sysmocom

sysmocom - s.f.m.c. GmbH

OsmoBSC User Manual

by Holger Freyther, Harald Welte, and Neels Hofmeyr

Copyright © 2012-2018 sysmocom - s.f.m.c. GmbH

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with the Invariant Sections being just 'Foreword', 'Acknowledgements' and 'Preface', with no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

The Asciidoc source code of this manual can be found at http://git.osmocom.org/osmo-gsm-manuals/

HISTORY			
NUMBER	DATE	DESCRIPTION	NAME
1	February 2016	Initial OsmoBSC manual, recycling OsmoNITB sections	HW
2	October 2018	Add Handover chapter: document new neighbor configuration, HO algorithm 2 and inter-BSC handover.	NH

Contents

1 Foreword **1**
- 1.1 Acknowledgements . . . 1
- 1.2 Endorsements . . . 2

2 Preface **2**
- 2.1 FOSS lives by contribution! . . . 2
- 2.2 Osmocom and sysmocom . . . 3
- 2.3 Corrections . . . 3
- 2.4 Legal disclaimers . . . 3
  - 2.4.1 Spectrum License . . . 3
  - 2.4.2 Software License . . . 3
  - 2.4.3 Trademarks . . . 3
  - 2.4.4 Liability . . . 4
  - 2.4.5 Documentation License . . . 4

3 Introduction **4**
- 3.1 Required Skills . . . 4
- 3.2 Getting assistance . . . 5

4 Overview **5**
- 4.1 About OsmoBSC . . . 5
- 4.2 Software Components . . . 5
  - 4.2.1 A-bis Implementation . . . 5
  - 4.2.2 A Implementation . . . 5
    - 4.2.2.1 BSSAP/SCCPlite . . . 6
    - 4.2.2.2 BSSAP/SCCP/M3UA . . . 6
  - 4.2.3 BSC Implementation . . . 6
  - 4.2.4 Speech traffic . . . 7

5 Running OsmoBSC **7**
- 5.1 SYNOPSIS . . . 7
- 5.2 OPTIONS . . . 7
- 5.3 Multiple instances . . . 8
- 5.4 Configure limits . . . 8
- 5.5 Configure primary links . . . 8
  - 5.5.1 Connect to an MSC's *A* interface . . . 8
    - 5.5.1.1 Configure SCCP/M3UA (AoIP) . . . 8
    - 5.5.1.2 Configure SCCPlite . . . 9
  - 5.5.2 Configure MGCP to connect to an MGW . . . 9
    - 5.5.2.1 Pinning a BTS to a specific MGW . . . 9
  - 5.5.3 Configure Lb to connect to an SMLC . . . 10
  - 5.5.4 Configure BSC co-located PCU . . . 10

6 The Osmocom VTY Interface **11**
6.1 Accessing the telnet VTY . . . 12
6.2 VTY Nodes . . . 13
6.3 Interactive help . . . 13
6.3.1 The question-mark (`?`) command . . . 13
6.3.2 TAB completion . . . 14
6.3.3 The `list` command . . . 15
6.3.4 The attribute system . . . 17
6.3.5 The expert mode . . . 18

7 libosmocore Logging System **19**
7.1 Log categories . . . 19
7.2 Log levels . . . 19
7.3 Log printing options . . . 20
7.4 Log filters . . . 20
7.5 Log targets . . . 20
7.5.1 Logging to the VTY . . . 21
7.5.2 Logging to the ring buffer . . . 21
7.5.3 Logging via gsmtap . . . 21
7.5.4 Logging to a file . . . 22
7.5.5 Logging to syslog . . . 23
7.5.6 Logging to systemd-journal . . . 23
7.5.7 Logging to stderr . . . 25

8 Signaling Networks: SS7 and SIGTRAN **25**
8.1 Physical Layer . . . 25
8.2 Message Transfer Part (MTP) . . . 25
8.2.1 Point Codes . . . 26
8.3 Higher-Layer Protocols . . . 26
8.4 Signaling Connection Control Part (SCCP) . . . 26
8.4.1 SCCP Addresses . . . 26
8.4.2 Global Titles . . . 27
8.4.3 Global Title Translation (GTT) . . . 27
8.4.4 Peculiarities of Connection Oriented SCCP . . . 28
8.5 SIGTRAN - SS7 over IP Networks . . . 28
8.5.1 SIGTRAN Concepts / Terminology . . . 28
8.5.1.1 Signaling Gateway (SG) . . . 28
8.5.1.2 Application Server (AS) . . . 28
8.5.1.3 Application Server Process (ASP) . . . 29

8.5.2 SIGTRAN variants / stackings . . . 29
8.5.2.1 MTP3 User Adaptation (M3UA) . . . 29
8.5.2.2 SCCP User Adaptation (SUA) . . . 29
8.5.2.3 MTP2 User Adaptation (M2UA) . . . 29
8.5.2.4 MTP2-User Peer-to-Peer Adaptation (M2PA) . . . 29
8.5.3 SIGTRAN security . . . 29
8.5.4 IPv6 support . . . 30
8.5.5 SCTP multi-homing in SIGTRAN . . . 30
8.5.6 SCTP Primary Address . . . 30
8.5.7 SCTP Peer Primary Address . . . 30
8.5.8 SCTP INIT Parameters . . . 31
8.5.9 SCTP role . . . 32
8.5.10 M3UA/SUA role . . . 32
8.5.11 Traffic Modes in SIGTRAN . . . 32

9 Osmocom SS7 + SIGTRAN support 32
9.1 History / Background . . . 32
9.1.1 The Past (before 2017) . . . 33
9.1.2 The present (2017) . . . 33
9.2 Osmocom extensions to SIGTRAN . . . 33
9.2.1 Osmocom M3UA Routing Key Management Extensions . . . 34
9.2.2 IPA / SCCPlite backwards compatibility . . . 34
9.3 Minimal Osmocom SIGTRAN configurations for small networks . . . 35
9.3.1 A minimal 2G configuration to get started . . . 35
9.3.2 A minimal 3G configuration to get started . . . 35
9.4 Osmocom SS7 Instances . . . 36
9.5 Osmocom SS7 xUA Server . . . 37
9.6 Osmocom SS7 Users . . . 37
9.7 Osmocom SS7 Links . . . 37
9.8 Osmocom SS7 Linksets . . . 37
9.9 Osmocom SS7 Application Servers . . . 37
9.10 Osmocom SS7 Application Server Processes . . . 38
9.11 Osmocom SS7 Routes . . . 38
9.12 Osmocom SCCP Instances . . . 39
9.13 Osmocom SCCP User . . . 39
9.14 Osmocom SCCP Connection . . . 39
9.15 Osmocom SCCP User SAP . . . 39
9.16 Osmocom MTP User SAP . . . 39

10 Configure SCCP/M3UA 39
10.1 Connect to STP Instance . . . 41
10.2 Local Point-Code . . . 41
10.3 Remote Point-Code . . . 41
10.4 Point-Code Format . . . 42
10.5 AS and ASP . . . 43
10.6 Subsystem Number (SSN) . . . 43
10.7 Routing Context / Routing Key . . . 44
10.7.1 M3UA without Routing Context IE / Routing Context *0* . . . 44
10.7.2 Example: Static Routing . . . 44

11 Reviewing and Provisioning BTS configuration 47
11.1 Reviewing current BTS status and configuration . . . 47
11.2 Provisioning a new BTS . . . 48
11.3 System Information configuration . . . 49
11.4 Neighbor List configuration . . . 49
11.5 Configuring GPRS PCU parameters of a BTS . . . 49
11.6 More explanation about the PCU config parameters . . . 50
11.6.1 `gprs mode (none|gprs|egprs)` . . . 50
11.6.2 `gprs cell bvci <2-65535>` . . . 50
11.6.3 `gprs nsei <0-65535>` . . . 50
11.6.4 `gprs nsvc <0-1> nsvci <0-65535>` . . . 50
11.6.5 `gprs nsvc <0-1> local udp port <0-65535>` . . . 50
11.6.6 `gprs nsvc <0-1> remote udp port <0-65535>` . . . 50
11.6.7 `gprs nsvc <0-1> remote ip A.B.C.D` . . . 50
11.6.8 `gprs ns timer (tns-block|tns-block-retries|tns-reset|tns-reset-retries|tns-test` <0-255> . . . 50
11.7 Dynamic Timeslot Configuration (TCH / PDCH) . . . 51
11.7.1 Osmocom Style Dynamic Timeslots (DYNAMIC/OSMOCOM) . . . 51
11.7.2 ip.access Style Dynamic Timeslots (DYNAMIC/IPACCESS) . . . 51
11.7.3 Avoid PDCH Exhaustion . . . 52
11.7.4 Dynamic Timeslot Configuration Examples . . . 52
11.8 Tuning Access to the BTS . . . 52
11.8.1 Access Control Class Load Management . . . 53
11.9 Configuring FACCH/SACCH repetition . . . 54
11.9.1 RACH Parameter Configuration . . . 55
11.10 Configuring Ericsson RBS Interface Switch (IS) . . . 56
11.10.1 Understanding the is-connection-list VTY option . . . 56
11.10.2 E1 port and TRU ICP numbers . . . 57

12 OsmoBSC example configuration files **58**
12.1 Example configuration for OsmoBSC with one single-TRX nanoBTS . . . 58
12.2 Example configuration for OsmoBSC with multi-TRX nanoBTS . . . 60
12.3 Example configuration for OsmoBSC with E1 BTS . . . 61
12.4 Example configuration for OsmoBSC with Ericsson RBS E1 BTS and EGPRS . . . 63
12.5 E1 Line number and MGCP trunk number . . . 65

13 BSC level configuration **65**
13.1 Hand-over . . . 65
13.1.1 Hand-over in GSM . . . 65
13.1.2 Configuration of hand-over in OsmoBSC . . . 65
13.2 Timer Configuration . . . 66
13.3 Discontinuous Transmission (DTX) . . . 66

14 Channel allocation **67**
14.1 Channel allocation parameters . . . 67
14.1.1 Channel allocation modes . . . 67
14.1.1.1 Dynamic channel allocation mode . . . 68
14.1.2 Interference aware channel allocation . . . 69
14.1.3 TCH sigalling policy . . . 69

15 Power control **69**
15.1 Power control parameters . . . 69
15.1.1 When the parameters come into effect? . . . 70
15.2 Power control configuration . . . 70
15.2.1 Power control mode . . . 71
15.2.2 Power control interval . . . 72
15.2.3 Power change step size . . . 73
15.2.4 RxLev and RxQual thresholds . . . 74
15.2.5 Carrier-to-Interference (C/I) thresholds . . . 75
15.2.6 Measurement averaging process . . . 75
15.3 BCCH carrier power reduction operation . . . 77
15.3.1 Supported BTS models . . . 77
15.3.2 Interworking with static and dynamic power control . . . 77
15.3.3 Managing BCCH carrier power reduction . . . 77
15.4 Temporary ACCH overpower . . . 78

16 Interference reporting **79**
16.1 Interference reporting parameters . . . 80
16.2 PDCH and dynamic timeslot handling . . . 80

17 CS Handover **81**
17.1 How Handover Works . . . 81
17.1.1 Internal / Intra-BSC Handover . . . 82
17.1.2 External / Inter-BSC Handover . . . 83
17.2 Configuring Neighbors . . . 84
17.2.1 Default: All Local Cells are Neighbors . . . 84
17.2.2 Local-BSS Neighbors . . . 85
17.2.3 Remote-BSS Neighbors . . . 86
17.2.4 Reconfiguring Neighbors in a Running OsmoBSC . . . 86
17.3 Configuring Handover Decisions . . . 87
17.3.1 Common Configuration . . . 87
17.3.2 Handover Algorithm 1 . . . 88
17.3.3 Handover Algorithm 2 . . . 88
17.3.3.1 Load Distribution . . . 88
17.3.4 External / Inter-BSC Handover Considerations . . . 90
17.4 Advertising 3G/4G neighbors . . . 90
17.4.1 UMTS/UTRAN/3G neighbors . . . 90
17.4.2 LTE/EUTRAN/4G neighbors . . . 91

18 PS Handover **92**
18.1 Neighbor Address Resolution Service . . . 92
18.1.1 Neighbor Address Resolution Service CTRL interface (deprecated) . . . 93

19 SMSCB (Cell Broadcast) **93**
19.1 Enabling a CBCH channel combination . . . 93
19.2 Configuring the CBSP connection . . . 94
19.3 Counters . . . 95

20 MSC Pooling **95**
20.1 Configuring MSC Pooling . . . 96
20.1.1 Connecting Multiple MSCs . . . 96
20.1.2 NRI Value Bit Length . . . 96
20.1.3 NULL-NRI . . . 96
20.1.4 Assigning NRI Ranges to MSCs . . . 97
20.1.5 MSC Offloading . . . 98

21 MGW Pooling **98**
21.1 MGW pool VTY configuration . . . 98
21.2 MGW pool management . . . 99
21.2.1 MGW pool status . . . 99
21.2.2 Adding an MGW / MGCP-Client to the MGW pool . . . 99
21.2.3 Reconnecting an MGW / MGCP-Client . . . 100
21.2.4 Monitor MGCP link (keepalive) / MGCP-Client . . . 101
21.2.5 Blocking an MGW / MGCP-Client . . . 101
21.2.6 Removing an MGW / MGCP-Client . . . 102

22 Location Services: Lb interface to SMLC **102**
22.1 Configure Lb-interface . . . 103

23 QoS, DSCP/TOS, Priority and IEEE 802.1q PCP **104**
23.1 IP Level (DSCP) . . . 104
23.2 Packet Priority . . . 104
23.3 Ethernet Level (PCP) . . . 105
23.4 Putting things together . . . 106
23.4.1 Full example of QoS for osmo-bsc downlink QoS . . . 107

24 Osmocom Counters **107**
24.1 Osmo Counters (deprecated) . . . 108
24.2 Rate Counters . . . 108
24.3 Stat Item . . . 108
24.4 Statistic Levels . . . 108
24.4.1 Global . . . 108
24.4.2 Peer . . . 108
24.4.3 Subscriber . . . 109
24.5 Stats Reporter . . . 109
24.5.1 Configuring a stats reporter . . . 109
24.6 Socket stats . . . 110
24.6.1 Configuration . . . 110
24.6.2 Generated stats items . . . 110

25 Implemented Counters **111**
25.1 Rate Counters . . . 111

26 Osmo Stat Items **114**

27 Osmo Counters **115**

28 Abis/IP Interface **115**
28.1 A-bis Operation & Maintenance Link . . . 115
28.2 A-bis Radio Signalling Link . . . 115
28.3 Locate Abis/IP based BTS . . . 116
28.3.1 abisip-find . . . 116
28.4 Deploying a new nanoBTS . . . 116
28.4.1 ipaccess-config . . . 116

29 Osmocom Control Interface **117**
29.1 Control Interface Protocol . . . 117
29.1.1 GET operation . . . 118
29.1.2 SET operation . . . 119
29.1.3 TRAP operation . . . 119
29.2 Common variables . . . 119
29.3 Control Interface python examples . . . 120
29.3.1 Getting rate counters . . . 120
29.3.2 Setting a value . . . 120
29.3.3 Getting a value . . . 121
29.3.4 Listening for traps . . . 121

30 Control interface **121**
30.1 notification . . . 126
30.2 inform-msc-v1 . . . 126
30.3 channel-load . . . 126
30.4 gprs-mode . . . 126
30.5 rf_state . . . 126
30.6 rf_locked . . . 126
30.7 max-power-reduction . . . 126
30.8 add/del neighbor cell . . . 127

31 Osmux **127**
31.1 Osmux and NAT . . . 128
31.2 CID allocation . . . 128
31.3 3GPP AoIP network setup with Osmux . . . 129
31.4 SCCPLite network setup with Osmux . . . 131
31.5 SCCPLite network setup with Osmux + BSC-NAT . . . 133
31.6 Osmux and MGCP . . . 135
31.6.1 `X-Osmux` Format . . . 135
31.6.2 `X-Osmux` Considerations . . . 135
31.6.3 `X-Osmux` Support . . . 136

31.7 Abis setup with Osmux . . . 136
31.8 Osmux Support in OsmoBSC . . . 137
31.8.1 OsmoBSC in a A/IP with IPA/SCCPlite network setup . . . 137
31.8.2 OsmoBSC in a 3GPP AoIP network setup . . . 137
31.8.3 Osmux in the ip.access Abis interface . . . 138

32 VTY Process and Thread management 138
32.1 Scheduling Policy . . . 139
32.2 CPU-Affinity Mask . . . 139

33 AoIP message flow examples 140
33.1 AoIP interface bring-up . . . 140
33.1.1 SCTP multi-homing . . . 141
33.1.2 MSC pooling . . . 141
33.2 MO call establishment on AoIP with user plane . . . 142

34 Glossary 144

A Osmocom TCP/UDP Port Numbers 152

B Bibliography / References 153
B.0.0.0.1 References . . . 153

C GNU Free Documentation License 158
C.1 PREAMBLE . . . 158
C.2 APPLICABILITY AND DEFINITIONS . . . 158
C.3 VERBATIM COPYING . . . 159
C.4 COPYING IN QUANTITY . . . 159
C.5 MODIFICATIONS . . . 159
C.6 COMBINING DOCUMENTS . . . 161
C.7 COLLECTIONS OF DOCUMENTS . . . 161
C.8 AGGREGATION WITH INDEPENDENT WORKS . . . 161
C.9 TRANSLATION . . . 161
C.10 TERMINATION . . . 161
C.11 FUTURE REVISIONS OF THIS LICENSE . . . 162
C.12 RELICENSING . . . 162
C.13 ADDENDUM: How to use this License for your documents . . . 162

1 Foreword

Digital cellular networks based on the GSM specification were designed in the late 1980s and first deployed in the early 1990s in Europe. Over the last 25 years, hundreds of networks were established globally and billions of subscribers have joined the associated networks.

The technological foundation of GSM was based on multi-vendor interoperable standards, first created by government bodies within CEPT, then handed over to ETSI, and now in the hands of 3GPP. Nevertheless, for the first 17 years of GSM technology, the associated protocol stacks and network elements have only existed in proprietary *black-box* implementations and not as Free Software.

In 2008 Dieter Spaar and I started to experiment with inexpensive end-of-life surplus Siemens GSM BTSs. We learned about the A-bis protocol specifications, reviewed protocol traces and started to implement the BSC-side of the A-bis protocol as something originally called `bs11-abis`. All of this was *just for fun*, in order to learn more and to boldly go where no Free Software developer has gone before. The goal was to learn and to bring Free Software into a domain that despite its ubiquity, had not yet seen any Free / Open Source software implementations.

`bs11-abis` quickly turned into `bsc-hack`, then *OpenBSC* and its *OsmoNITB* variant: A minimal implementation of all the required functionality of an entire GSM network, exposing A-bis towards the BTS. The project attracted more interested developers, and surprisingly quickly also commercial interest, contribution and adoption. This allowed adding support for more BTS models.

After having implemented the network-side GSM protocol stack in 2008 and 2009, in 2010 the same group of people set out to create a telephone-side implementation of the GSM protocol stack. This established the creation of the Osmocom umbrella project, under which OpenBSC and the OsmocomBB projects were hosted.

Meanwhile, more interesting telecom standards were discovered and implemented, including TETRA professional mobile radio, DECT cordless telephony, GMR satellite telephony, some SDR hardware, a SIM card protocol tracer and many others.

Increasing commercial interest particularly in the BSS and core network components has lead the way to 3G support in Osmocom, as well as the split of the minimal *OsmoNITB* implementation into separate and fully featured network components: OsmoBSC, OsmoMSC, OsmoHLR, OsmoMGW and OsmoSTP (among others), which allow seamless scaling from a simple "Network In The Box" to a distributed installation for serious load.

It has been a most exciting ride during the last eight-odd years. I would not have wanted to miss it under any circumstances.

— Harald Welte, Osmocom.org and OpenBSC founder, December 2017.

1.1 Acknowledgements

My deep thanks to everyone who has contributed to Osmocom. The list of contributors is too long to mention here, but I'd like to call out the following key individuals and organizations, in no particular order:

- Dieter Spaar for being the most amazing reverse engineer I've met in my career
- Holger Freyther for his many code contributions and for shouldering a lot of the maintenance work, setting up Jenkins - and being crazy enough to co-start sysmocom as a company with me ;)
- Andreas Eversberg for taking care of Layer2 and Layer3 of OsmocomBB, and for his work on OsmoBTS and OsmoPCU
- Sylvain Munaut for always tackling the hardest problems, particularly when it comes closer to the physical layer
- Chaos Computer Club for providing us a chance to run real-world deployments with tens of thousands of subscribers every year
- Bernd Schneider of Netzing AG for funding early ip.access nanoBTS support
- On-Waves ehf for being one of the early adopters of OpenBSC and funding a never ending list of features, fixes and general improvement of pretty much all of our GSM network element implementations
- sysmocom, for hosting and funding a lot of Osmocom development, the annual Osmocom Developer Conference and releasing this manual.

- Jan Luebbe, Stefan Schmidt, Daniel Willmann, Pablo Neira, Nico Golde, Kevin Redon, Ingo Albrecht, Alexander Huemer, Alexander Chemeris, Max Suraev, Tobias Engel, Jacob Erlbeck, Ivan Kluchnikov
- NLnet Foundation, for providing funding for a number of individual work items within the Osmocom universe, such as LTE support in OsmoCBC or GPRS/EGPRS support for Ericsson RBS6000.
- WaveMobile Ltd, for many years of sponsoring.

May the source be with you!

— Harald Welte, Osmocom.org and OpenBSC founder, January 2016.

1.2 Endorsements

This version of the manual is endorsed by Harald Welte as the official version of the manual.

While the GFDL license (see Appendix C) permits anyone to create and distribute modified versions of this manual, such modified versions must remove the above endorsement.

2 Preface

First of all, we appreciate your interest in Osmocom software.

Osmocom is a Free and Open Source Software (FOSS) community that develops and maintains a variety of software (and partially also hardware) projects related to mobile communications.

Founded by people with decades of experience in community-driven FOSS projects like the Linux kernel, this community is built on a strong belief in FOSS methodology, open standards and vendor neutrality.

2.1 FOSS lives by contribution!

If you are new to FOSS, please try to understand that this development model is not primarily about “free of cost to the GSM network operator”, but it is about a collaborative, open development model. It is about sharing ideas and code, but also about sharing the effort of software development and maintenance.

If your organization is benefiting from using Osmocom software, please consider ways how you can contribute back to that community. Such contributions can be many-fold, for example

- sharing your experience about using the software on the public mailing lists, helping to establish best practises in using/operating it,
- providing qualified bug reports, workarounds
- sharing any modifications to the software you may have made, whether bug fixes or new features, even experimental ones
- providing review of patches
- testing new versions of the related software, either in its current “master” branch or even more experimental feature branches
- sharing your part of the maintenance and/or development work, either by donating developer resources or by (partially) funding those people in the community who do.

We’re looking forward to receiving your contributions.

2.2 Osmocom and sysmocom

Some of the founders of the Osmocom project have established *sysmocom - systems for mobile communications GmbH* as a company to provide products and services related to Osmocom.

sysmocom and its staff have contributed by far the largest part of development and maintenance to the Osmocom mobile network infrastructure projects.

As part of this work, sysmocom has also created the manual you are reading.

At sysmocom, we draw a clear line between what is the Osmocom FOSS project, and what is sysmocom as a commercial entity. Under no circumstances does participation in the FOSS projects require any commercial relationship with sysmocom as a company.

2.3 Corrections

We have prepared this manual in the hope that it will guide you through the process of installing, configuring and debugging your deployment of cellular network infrastructure elements using Osmocom software. If you do find errors, typos and/or omissions, or have any suggestions on missing topics, please do take the extra time and let us know.

2.4 Legal disclaimers

2.4.1 Spectrum License

As GSM and UMTS operate in licensed spectrum, please always double-check that you have all required licenses and that you do not transmit on any ARFCN or UARFCN that is not explicitly allocated to you by the applicable regulatory authority in your country.

Warning
Depending on your jurisdiction, operating a radio transmitter without a proper license may be considered a felony under criminal law!

2.4.2 Software License

The software developed by the Osmocom project and described in this manual is Free / Open Source Software (FOSS) and subject to so-called *copyleft* licensing.

Copyleft licensing is a legal instrument to ensure that this software and any modifications, extensions or derivative versions will always be publicly available to anyone, for any purpose, under the same terms as the original program as developed by Osmocom.

This means that you are free to use the software for whatever purpose, make copies and distribute them - just as long as you ensure to always provide/release the *complete and corresponding* source code.

Every Osmocom software includes a file called `COPYING` in its source code repository which explains the details of the license. The majority of programs is released under GNU Affero General Public License, Version 3 (AGPLv3).

If you have any questions about licensing, don't hesitate to contact the Osmocom community. We're more than happy to clarify if your intended use case is compliant with the software licenses.

2.4.3 Trademarks

All trademarks, service marks, trade names, trade dress, product names and logos appearing in this manual are the property of their respective owners. All rights not expressly granted herein are reserved.

For your convenience we have listed below some of the registered trademarks referenced herein. This is not a definitive or complete list of the trademarks used.

Osmocom® and *OpenBSC*® are registered trademarks of Holger Freyther and Harald Welte.

sysmocom® and *sysmoBTS*® are registered trademarks of *sysmocom - systems for mobile communications GmbH*.

ip.access® and *nanoBTS*® are registered trademarks of *ip.access Ltd.*

2.4.4 Liability

The software is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the License text included with the software for more details.

2.4.5 Documentation License

Please see Appendix C for further information.

3 Introduction

3.1 Required Skills

Please note that even while the capital expenses of running mobile networks has decreased significantly due to Osmocom software and associated hardware like sysmoBTS, GSM networks are still primarily operated by large GSM operators.

Neither the GSM specification nor the GSM equipment was ever designed for networks to be installed and configured by anyone but professional GSM engineers, specialized in their respective area like radio planning, radio access network, back-haul or core network.

If you do not share an existing background in GSM network architecture and GSM protocols, correctly installing, configuring and optimizing your GSM network will be tough, irrespective whether you use products with Osmocom software or those of traditional telecom suppliers.

GSM knowledge has many different fields, from radio planning through site installation to core network configuration/administration.

The detailed skills required will depend on the type of installation and/or deployment that you are planning, as well as its associated network architecture. A small laboratory deployment for research at a university is something else than a rural network for a given village with a handful of cells, which is again entirely different from an urban network in a dense city.

Some of the useful skills we recommend are:

- general understanding about RF propagation and path loss in order to estimate coverage of your cells and do RF network planning.
- general understanding about GSM network architecture, its network elements and key transactions on the Layer 3 protocol
- general understanding about voice telephony, particularly those of ISDN heritage (Q.931 call control)
- understanding of GNU/Linux system administration and working on the shell
- understanding of TCP/IP networks and network administration, including tcpdump, tshark, wireshark protocol analyzers.
- ability to work with text based configuration files and command-line based interfaces such as the VTY of the Osmocom network elements

3.2 Getting assistance

If you do have a support package / contract with sysmocom (or want to get one), please contact support@sysmocom.de with any issues you may have.

If you don't have a support package / contract, you have the option of using the resources put together by the Osmocom community at https://projects.osmocom.org/, checking out the wiki and the mailing-list for community-based assistance. Please always remember, though: The community has no obligation to help you, and you should address your requests politely to them. The information (and software) provided at osmocom.org is put together by volunteers for free. Treat them like a friend whom you're asking for help, not like a supplier from whom you have bought a service.

If you would like to obtain professional/commercial support on Osmocom CNI, you can always reach out to sales@sysmocom.de to discuss your support needs. Purchasing support from sysmocom helps to cover the ongoing maintenance of the Osmocom CNI software stack.

4 Overview

This manual should help you getting started with OsmoBSC. It will cover aspects of configuring and running the OsmoBSC.

4.1 About OsmoBSC

OsmoBSC is the Osmocom implementation of a Base Station Controller. It implements:

- an *A-bis* interface towards BTSs and
- an *A* interface towards an MSC. It is important to be aware that there are two variants of the *A* interface, see Section 4.2.2.

4.2 Software Components

OsmoBSC contains a variety of different software components, which we'll briefly describe in this section.

4.2.1 A-bis Implementation

OsmoBSC implements the ETSI/3GPP specified A-bis interface, including TS 08.56 (LAPD), TS 08.58 (RSL) and TS 12.21 (OML). In addition, it supports a variety of vendor-specific extensions and dialects in order to communicate with BTSs from Siemens, Nokia, Ericsson, ip.access, Octasic and sysmocom, as well as various USRP based BTS implementations, using OsmoBTS and OsmoTRX (like the Ettus B200 series, the Fairwaves XTRX or the LimeSDR, to name a few).

For more information, see Section 11 and Section 12.

4.2.2 A Implementation

OsmoBSC implements a sub-set of the GSM A interface as specified in TS 08.08 (BSSAP) and TS 04.08 (DTAP).

Osmocom offers two variants of the *A* interface's protocol stacking:

- *BSSAP/SCCPlite*
- *BSSAP/SCCP/M3UA* (called AoIP)

Traditionally, OsmoBSC only implemented the BSSAP/SCCPlite protocol, but since a proper M3UA implementation became available in 2017 as part of *libosmo-sigtran* (*libosmo-sccp.git*), the stock OsmoBSC now supports BSSAP/SCCP/M3UA. SCCPlite support has been subsequently added to libosmo-sigtran in 2018, and now both variants of the *A* interface are supported by `osmo-bsc`.

The difference between an BSSAP/SCCPlite and BSSAP/SCCP/M3UA is illustrated in Figure 1 and Figure 2.

4.2.2.1 BSSAP/SCCPlite

Unlike classic A interface implementations for E1 interfacs, `osmo-bsc` implements a variant of encapsulating the A interface over IP. To do so, the SCCP messages are wrapped in an IPA multiplex and then communicated over TCP. The audio channels are mapped to RTP streams.

This protocol stacking is sometimes called "SCCPlite".

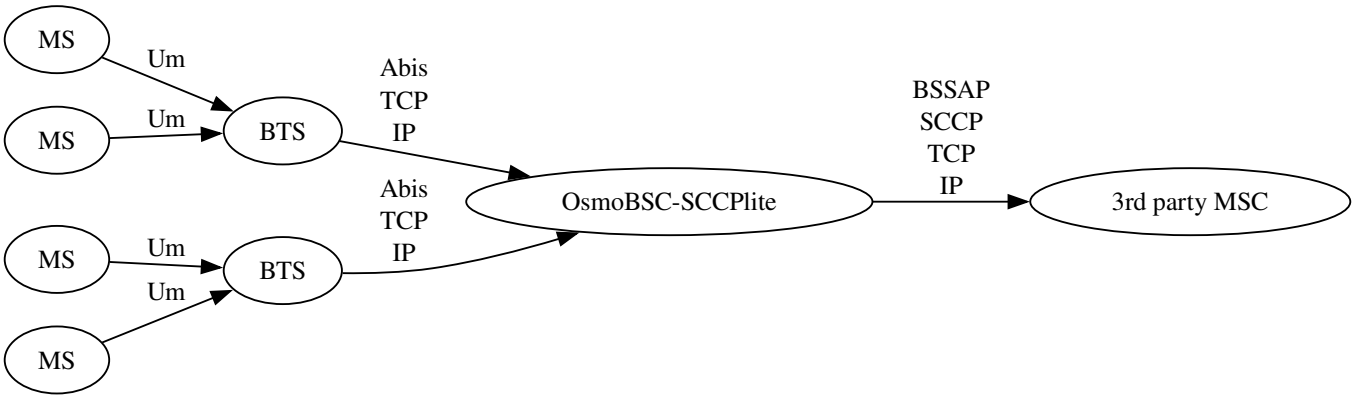


Figure 1: `osmo-bsc-sccplite` operation using *BSSAP/SCCPlite*

4.2.2.2 BSSAP/SCCP/M3UA

The default OsmoBSC's A interface uses the M3UA variant of SIGTRAN protocol stacking:

BSSAP
SCCP
M3UA
SCTP
IP

It is recommended to use the M3UA variant, which is required to operate with OsmoMSC.

To route SCCP/M3UA messages between OsmoBSC and and MSC, an STP instance like OsmoSTP is required.

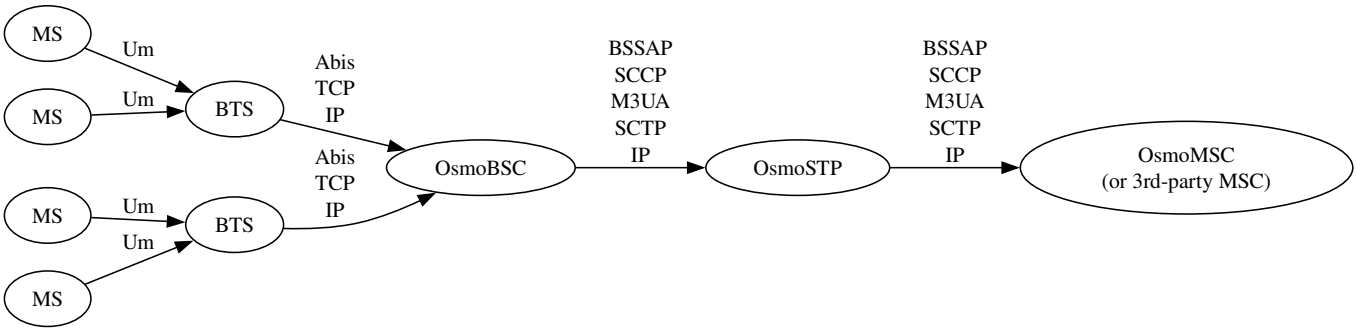


Figure 2: `osmo-bsc` operation using *BSSAP/SCCP/M3UA*

4.2.3 BSC Implementation

The BSC implementation covers the classic functionality of a GSM Base Station Controller, i.e.

- configuring and bringing up BTSs with their TRXs and TSs
- implementing the A-bis interface / protocols for signalling and actual voice data (TRAU frames).
- processing measurement results from the mobile stations in dedicated mode, performing hand-over decision and execution.
- Terminating the TS 04.08 RR (Radio Resource) sub-layer from the MS.

For more information, see [?], Section 11 and Section 12.

4.2.4 Speech traffic

OsmoBSC, by itself, does not perform any transcoding or relaying of user plane speech traffic. This task is handled entirely by a BSC co-located media gateway, such as OsmoMGW, which will take care of relaying the RTP traffic from the BTS into the core network.

In case classic E1 based BTSs are used, OsmoBSC will instruct the MGW to convert between TRAU frames on the E1 side and RTP frames on the IP based core network side.

5 Running OsmoBSC

The OsmoBSC executable (`osmo-bsc`) offers the following command-line arguments:

5.1 SYNOPSIS

osmo-bsc [-hl-V] [-d *DBGMASK*] [-D] [-c *CONFIGFILE*] [-s] [-T] [-e *LOGLEVEL*] [-l *IP*] [-r *RFCTL*]

5.2 OPTIONS

-h, --help
Print a short help message about the supported options

-V, --version
Print the compile-time version number of the program

-d, --debug *DBGMASK,DBGLEVELS*
Set the log subsystems and levels for logging to stderr. This has mostly been superseded by VTY-based logging configuration, see Section 7 for further information.

-D, --daemonize
Fork the process as a daemon into background.

-c, --config-file *CONFIGFILE*
Specify the file and path name of the configuration file to be used. If none is specified, use `osmo-bsc.cfg` in the current working directory.

-s, --disable-color
Disable colors for logging to stderr. This has mostly been deprecated by VTY based logging configuration, see Section 7 for more information.

-T, --timestamp
Enable time-stamping of log messages to stderr. This has mostly been deprecated by VTY based logging configu- ration, see Section 7 for more information.

-e, --log-level *LOGLEVEL*
Set the global log level for logging to stderr. This has mostly been deprecated by VTY based logging configuration, see Section 7 for more information.

-r, --rf-ctl *RFCTL*
Offer a Unix domain socket for RF control at the path/filename *RFCTL* in the file system.

5.3 Multiple instances

Running multiple instances of `osmo-bsc` on the same host is possible if all interfaces (VTY, CTRL) are separated using the appropriate configuration options. The IP based interfaces are binding to local host by default. In order to separate the processes, the user has to bind those services to specific but different IP addresses and/or ports.

The VTY and the Control interface can be bound to IP addresses from the loopback address range, for example:

```
line vty
 bind 127.0.0.2
ctrl
 bind 127.0.0.2
```

For the following links, OsmoBSC acts as a client and does not listen/bind to a specific interface, and will hence not encounter conflicts for multiple instances running on the same interface:

- The SCCP/M3UA links are established by OsmoBSC contacting an STP.
- The MGCP link is established by OsmoMSC contacting an MGW.

To run multiple OsmoBSC instances on the same A-interface (SCCP/M3UA), each BSC has to configure a distinct point-code. See Section 10.

5.4 Configure limits

When connecting hundreds of TRX to OsmoBSC, it may be necessary to adjust the operating system's limit on open file descriptors for the osmo-bsc process. A typical default limit imposed by operating systems is 1024; this would be exceeded by, for example, about 205 BTS with 4 TRX each. (Each BTS with 4 TRX requires 5 file descriptors for Abis; 205 * 5 already exceeds 1024, sockets for other interfaces not considered yet.)

It should be ok to set an OS limit on open file descriptors as high as 65536 for osmo-bsc, which practically rules out failure from running out of file descriptors anywhere (<50,000 TRX).

When using systemd, the file descriptor limit may be adjusted in the service file by the `LimitNOFILE` setting ("Number of Open FILE descriptors"). OsmoBSC ships a systemd service file with a high LimitNOFILE setting.

5.5 Configure primary links

5.5.1 Connect to an MSC's *A* interface

5.5.1.1 Configure SCCP/M3UA (AoIP)

OsmoBSC acts as client to contact an STP instance and establish an SCCP/M3UA link.

An example configuration of OsmoBSC's AoIP SCCP link, assuming the BSC at point-code 1.23.3 and the MSC reachable at point-code 0.23.1 via an SG listening for M3UA at 127.0.0.1:2905:

```
cs7 instance 0
 point-code 1.23.3
 asp asp-clnt-msc-0 2905 0 m3ua
  remote-ip 127.0.0.1
  role asp
  sctp-role client
 sccp-address msc
  point-code 0.23.1
msc 0
 msc-addr msc
```

This configuration is explained in detail in Section 10.

5.5.1.2 Configure SCCPlite

Traditionally, OsmoBSC implemented only an SCCPlite based A-interface, an ad-hoc standard encapsulating BSSAP in an IPA Multiplex. Since 2017, OsmoBSC supports primarily a proper 3GPP compliant SCCP/M3UA A-interface known as AoIP, by a new libosmo-sigtran implementation. In 2018, SCCPlite compatibility was added to libosmo-sigtran, re-enabling the option of using an SCCPlite based A-interface. For details, see the OsmoSTP manual, chapter "IPA / SCCPlite backwards compatibility".

Here is an example configuration of OsmoBSC for SCCPlite, assuming the BSC at point-code 1.23.3 and an SCCPlite MSC listening on 127.0.0.1:5000 with own point-code 0.23.1:

```
cs7 instance 0
 point-code 1.23.3
 asp asp-clnt-msc-0 5000 0 ipa
  remote-ip 127.0.0.1
 as as-clnt-msc-0 ipa
  asp asp-clnt-msc-0
  routing-key 0 1.23.3
  point-code override dpc 0.23.1
 sccp-address remote_msc
  point-code 0.23.1
msc 0
 msc-addr remote_msc
```

5.5.2 Configure MGCP to connect to an MGW

OsmoBSC uses a media gateway (typically OsmoMGW) to direct RTP streams. By default, an MGW is expected to receive MGCP requests on the IANA-registered default port for MGCP (2427) on local host (127.0.0.1).

Here is an example configuration for a remote MGW:

```
network
 mgw 0
  remote-ip 10.9.8.7
  remote-port 2427
  reset-endpoint rtpbridge/* ❶
```

❶ The *reset-endpoint* setting instructs the OsmoBSC to send a wildcarded DLCX to the media gateway. This helps to clear lingering calls from the media gateway when the OsmoBSC is restarted.

OsmoBSC is also able to handle a pool of media gateways for load distribution since mid 2021. See also Section 21.

Note
Previous versions of OsmoBSC didn't have the *mgw* VTY node and hence didn't support the MGW pooling feature. Therefore, historically the MGW related commands where placed under the `msc` VTY node. The MGW related commands under the `msc` VTY are still parsed and used but its use is deprecated and hence discouraged in favour of the new `mgw` node. Writing the config to a file from within OsmoBSC will automatically convert the config to use the new `mgw` node.

5.5.2.1 Pinning a BTS to a specific MGW

It is sometimes desirable to assign a specific MGW to a given BTS, so that all calls where the BTS is involved use the assigned MGW with a higher precedence if possible.

This is specially important if the BTS is configured to serve calls using Osmux instead of RTP. Osmux features trunking optimizations, which allow transmission of audio payload from different concurrent calls inside the same underlaying UDP packet, hence reducing the total required throughput and saving costs on the required link.

In order for Osmux trunking optimization to work, the source and destination IP address of uderlaying UDP packet must be of course the same for all the calls involved. That essentially boils down to having all the concurrent calls of the BTS be connected to the same MGW so that they can be trunked over the same UDP connection.

The pinning to a specific MGW can be configured per BTS, and hence it is configured under the `bts` VTY node:

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# network
OsmoBSC(config-net)# bts 1
OsmoBSC(config-bts)# mgw pool-target 1 ❶
OsmoBSC(config-bts)# exit
OsmoBSC(config-net)# bts 2
OsmoBSC(config-mgw)# mgw pool-target 7 strict ❷
OsmoBSC(config-net)# bts 3
OsmoBSC(config-mgw)# no mgw pool-target ❸
```

❶ Pin BTS1 to prefer MGW1 (node `mgw 1`). If MGW1 is not configured, administrateivly blocked or not connected at the time a new call is to be established, then another MGW from the pool is selected following the usual procedures. This allows applying pinning in the usual scenario while still keeping call service ongoing against another MGW if the preferred MGW is not available at a given time.

❷ Pin BTS2 to prefer MGW7 (node `mgw 7`). If MGW7 is not configured, administrateivly blocked or not connected at the time a new call is to be established, then the MGW assignment will fail and ultimately the call will be terminated during establishment.

❸ Apply no pinning at all (default). The MGW with the lowest load is the one being selected for each new call.

5.5.3 Configure Lb to connect to an SMLC

Enable the Lb interface. OsmoBSC will then use the default point-codes to establish a connection to the SMLC.

```
smlc
 enable
```

More detailed configuration is described in Section 22.1.

5.5.4 Configure BSC co-located PCU

While small IP based BTSs usually come with a built in PCU (BTS co-located PCU), this does not have to be the case with any BTS. Especially larger E1 BTS usually make use of a BSC co-located PCU.

In the case of OsmoBSC this means that an instance of OsmoPCU is running next to OsmoBSC. Both processes share a unix domain socket to exchange signaling traffic and configuration parameters.

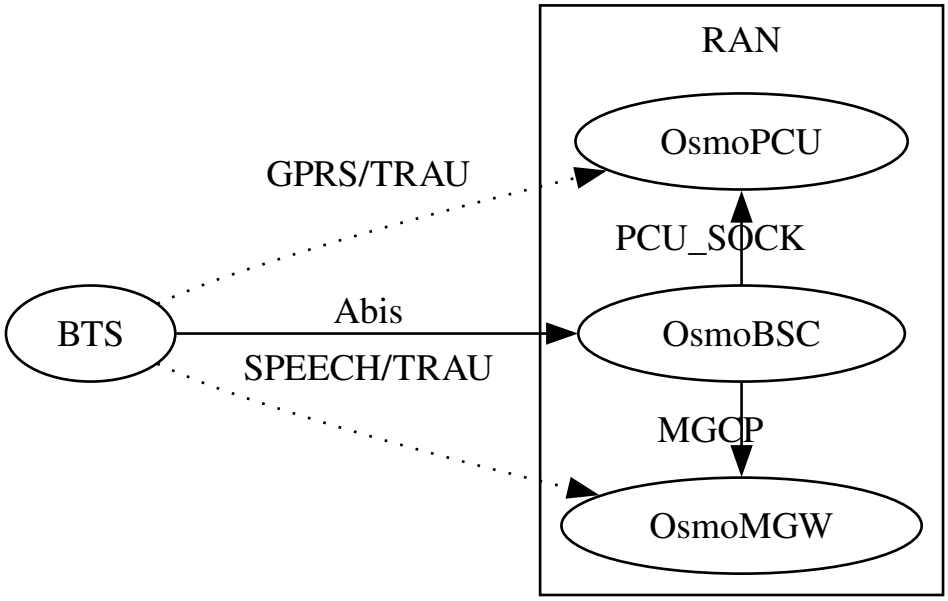


Figure 3: OsmoBSC with co-located OsmoPCU'

Apart from the configuration of the PCU socket path the configuration is not much different from those where the PCU is integrated inside the BTS. See also see also Section 11.5 for a detailed description.

Configure socket path to co-located PCU

```
network
 pcu-socket /tmp/pcu_bts
```

6 The Osmocom VTY Interface

All human interaction with Osmocom software is typically performed via an interactive command-line interface called the *VTY*.

Note
Integration of your programs and scripts should **not** be done via the telnet VTY interface, which is intended for human interaction only: the VTY responses may arbitrarily change in ways obvious to humans, while your scripts' parsing will likely break often. For external software to interact with Osmocom programs (besides using the dedicated protocols), it is strongly recommended to use the Control interface instead of the VTY, and to actively request / implement the Control interface commands as required for your use case.

The interactive telnet VTY is used to

- explore the current status of the system, including its configuration parameters, but also to view run-time state and statistics,
- review the currently active (running) configuration,
- perform interactive changes to the configuration (for those items that do not require a program restart),
- store the current running configuration to the config file,
- enable or disable logging; to the VTY itself or to other targets.

The Virtual Tele Type (VTY) has the concept of *nodes* and *commands*. Each command has a name and arguments. The name may contain a space to group several similar commands into a specific group. The arguments can be a single word, a string, numbers, ranges or a list of options. The available commands depend on the current node. there are various keyboard shortcuts to ease finding commands and the possible argument values.

Configuration file parsing during program start is actually performed the VTY's CONFIG node, which is also available in the telnet VTY. Apart from that, the telnet VTY features various interactive commands to query and instruct a running Osmocom program. A main difference is that during config file parsing, consistent indenting of parent vs. child nodes is required, while the interactive VTY ignores indenting and relies on the *exit* command to return to a parent node.

Note
In the *CONFIG* node, it is not well documented which commands take immediate effect without requiring a program restart. To save your current config with changes you may have made, you may use the `write file` command to **overwrite** your config file with the current configuration, after which you should be able to restart the program with all changes taking effect.

This chapter explains most of the common nodes and commands. A more detailed list is available in various programs' VTY reference manuals, e.g. see [vty-ref-osmomsc].

There are common patterns for the parameters, these include IPv4 addresses, number ranges, a word, a line of text and choice. The following will explain the commonly used syntactical patterns:

Table 3: VTY Parameter Patterns

Pattern	Example	Explanation
`A.B.C.D`	`127.0.0.1`	An IPv4 address
`A.B.C.D/M`	`192.168.1.0/24`	An IPv4 address and mask
`X:X::X:X`	`::1`	An IPv6 address
`X:X::X:X/M`	`::1/128`	An IPv6 address and mask
`TEXT`	`example01`	A single string without any spaces, tabs
`.TEXT`	`Some information`	A line of text
`(OptionA|OptionB|OptionC)`	`OptionA`	A choice between a list of available options
`<0-10>`	`5`	A number from a range

6.1 Accessing the telnet VTY

The VTY of a given Osmocom program is implemented as a telnet server, listening to a specific TCP port.

Please see Appendix A to check for the default TCP port number of the VTY interface of the specific Osmocom software you would like to connect to.

As telnet is insecure and offers neither strong authentication nor encryption, the VTY by default only binds to localhost (127.0.0.1) and will thus not be reachable by other hosts on the network.

Warning
By default, any user with access to the machine running the Osmocom software will be able to connect to the VTY. We assume that such systems are single-user systems, and anyone with local access to the system also is authorized to access the VTY. If you require stronger security, you may consider using the packet filter of your operating system to restrict access to the Osmocom VTY ports further.

6.2 VTY Nodes

The VTY by default has the following minimal nodes:

VIEW
When connecting to a telnet VTY, you will be on the *VIEW* node. As its name implies, it can only be used to view the system status, but it does not provide commands to alter the system state or configuration. As long as you are in the non-privileged *VIEW* node, your prompt will end in a > character.

ENABLE
The *ENABLE* node is entered by the `enable` command, from the *VIEW* node. Changing into the *ENABLE* node will unlock all kinds of commands that allow you to alter the system state or perform any other change to it. The *ENABLE* node and its children are signified by a # character at the end of your prompt.
You can change back from the *ENABLE* node to the *VIEW* node by using the `disable` command.

CONFIG
The *CONFIG* node is entered by the `configure terminal` command from the *ENABLE* node. The config node is used to change the run-time configuration parameters of the system. The prompt will indicate that you are in the config node by a `(config)#` prompt suffix.
You can always leave the *CONFIG* node or any of its children by using the `end` command.
This node is also automatically entered at the time the configuration file is read. All configuration file lines are processed as if they were entered from the VTY *CONFIG* node at start-up.

Other
Depending on the specific Osmocom program you are running, there will be few or more other nodes, typically below the *CONFIG* node. For example, the OsmoBSC has nodes for each BTS, and within the BTS node one for each TRX, and within the TRX node one for each Timeslot.

6.3 Interactive help

The VTY features an interactive help system, designed to help you to efficiently navigate is commands.

Note
The VTY is present on most Osmocom GSM/UMTS/GPRS software, thus this chapter is present in all the relevant manuals. The detailed examples below assume you are executing them on the OsmoMSC VTY. They will work in similar fashion on the other VTY interfaces, while the node structure will differ in each program.

6.3.1 The question-mark (?) command

If you type a single `?` at the prompt, the VTY will display possible completions at the exact location of your currently entered command.

If you type `?` at an otherwise empty command (without having entered even only a partial command), you will get a list of the first word of all possible commands available at this node:

Example: Typing ? at start of OsmoMSC prompt

```
OsmoMSC> ❶
  show        Show running system information
  list        Print command list
  exit        Exit current mode and down to previous mode
  help        Description of the interactive help system
  enable      Turn on privileged mode command
  terminal    Set terminal line parameters
  who         Display who is on vty
  logging     Configure logging
  no          Negate a command or set its defaults
  sms         SMS related commands
  subscriber  Operations on a Subscriber
```

❶ Type ? here at the prompt, the ? itself will not be printed.

If you have already entered a partial command, ? will help you to review possible options of how to continue the command. Let's say you remember that show is used to investigate the system status, but you don't remember the exact name of the object. Hitting ? after typing show will help out:

Example: Typing ? after a partial command

```
OsmoMSC> show ❶
  version          Displays program version
  online-help      Online help
  history          Display the session command history
  cs7              ITU-T Signaling System 7
  logging          Show current logging configuration
  alarms           Show current logging configuration
  talloc-context   Show talloc memory hierarchy
  stats            Show statistical values
  asciidoc         Asciidoc generation
  rate-counters    Show all rate counters
  fsm              Show information about finite state machines
  fsm-instances    Show information about finite state machine instances
  sgs-connections  Show SGs interface connections / MMEs
  subscriber       Operations on a Subscriber
  bsc              BSC
  connection       Subscriber Connections
  transaction      Transactions
  statistics       Display network statistics
  sms-queue        Display SMSqueue statistics
  smpp          SMPP Interface
```

❶ Type ? after the show command, the ? itself will not be printed.

You may pick the bsc object and type ? again:

Example: Typing ? after show bsc

```
OsmoMSC> show bsc
  <cr>
```

By presenting <cr> as the only option, the VTY tells you that your command is complete without any remaining arguments being available, and that you should hit enter, a.k.a. "carriage return".

6.3.2 TAB completion

The VTY supports tab (tabulator) completion. Simply type any partial command and press <tab>, and it will either show you a list of possible expansions, or completes the command if there's only one choice.

Example: Use of <tab> pressed after typing only s as command

```
OsmoMSC> s❶
show       sms        subscriber
```

❶ Type <tab> here.

At this point, you may choose show, and then press <tab> again:

Example: Use of <tab> pressed after typing show command

```
OsmoMSC> show ❶
version    online-help history    cs7        logging    alarms
talloc-context stats      asciidoc   rate-counters fsm        fsm-instances
sgs-connections subscriber bsc        connection transaction statistics
sms-queue smpp
```

❶ Type <tab> here.

6.3.3 The `list` command

The `list` command will give you a full list of all commands and their arguments available at the current node:

Example: Typing list at start of OsmoMSC VIEW node prompt

```
OsmoMSC> list
  show version
  show online-help
  list
  exit
  help
  enable
  terminal length <0-512>
  terminal no length
  who
  show history
  show cs7 instance <0-15> users
  show cs7 (sua|m3ua|ipa) [<0-65534>]
  show cs7 instance <0-15> asp
  show cs7 instance <0-15> as (active|all|m3ua|sua)
  show cs7 instance <0-15> sccp addressbook
  show cs7 instance <0-15> sccp users
  show cs7 instance <0-15> sccp ssn <0-65535>
  show cs7 instance <0-15> sccp connections
  show cs7 instance <0-15> sccp timers
  logging enable
  logging disable
  logging filter all (0|1)
  logging color (0|1)
  logging timestamp (0|1)
  logging print extended-timestamp (0|1)
  logging print category (0|1)
  logging print category-hex (0|1)
  logging print level (0|1)
  logging print file (0|1|basename) [last]
  logging set-log-mask MASK
  logging level (rll|cc|mm|rr|mncc|pag|msc|mgcp|ho|db|ref|ctrl|smpp|ranap|vlr|iucs|bssap| ↩
      sgs|lglobal|llapd|linp|lmux|lmi|lmib|lsms|lctrl|lgtp|lstats|lgsup|loap|lss7|lsccp|lsua ↩
      |lm3ua|lmgcp|ljibuf|lrspro) (debug|info|notice|error|fatal)
  logging level set-all (debug|info|notice|error|fatal)
  logging level force-all (debug|info|notice|error|fatal)
  no logging level force-all
  show logging vty
  show alarms
  show talloc-context (application|all) (full|brief|DEPTH)
  show talloc-context (application|all) (full|brief|DEPTH) tree ADDRESS
  show talloc-context (application|all) (full|brief|DEPTH) filter REGEXP
  show stats
  show stats level (global|peer|subscriber)
  show asciidoc counters
  show rate-counters
```

```
  show fsm NAME
  show fsm all
  show fsm-instances NAME
  show fsm-instances all
  show sgs-connections
  show subscriber (msisdn|extension|imsi|tmsi|id) ID
  show subscriber cache
  show bsc
  show connection
  show transaction
  sms send pending
  sms delete expired
  subscriber create imsi ID
  subscriber (msisdn|extension|imsi|tmsi|id) ID sms sender (msisdn|extension|imsi|tmsi|id)  ↩
      SENDER_ID send .LINE
  subscriber (msisdn|extension|imsi|tmsi|id) ID silent-sms sender (msisdn|extension|imsi| ↩
      tmsi|id) SENDER_ID send .LINE
  subscriber (msisdn|extension|imsi|tmsi|id) ID silent-call start (any|tch/f|tch/any|sdcch)
  subscriber (msisdn|extension|imsi|tmsi|id) ID silent-call stop
  subscriber (msisdn|extension|imsi|tmsi|id) ID ussd-notify (0|1|2) .TEXT
  subscriber (msisdn|extension|imsi|tmsi|id) ID ms-test close-loop (a|b|c|d|e|f|i)
  subscriber (msisdn|extension|imsi|tmsi|id) ID ms-test open-loop
  subscriber (msisdn|extension|imsi|tmsi|id) ID paging
  show statistics
  show sms-queue
  logging filter imsi IMSI
  show smpp esme
```

Tip
Remember, the list of available commands will change significantly depending on the Osmocom program you are accessing, its software version and the current node you're at. Compare the above example of the OsmoMSC *VIEW* node with the list of the OsmoMSC *NETWORK* config node:

Example: Typing list at start of OsmoMSC NETWORK config node prompt

```
OsmoMSC(config-net)# list
  help
  list
  write terminal
  write file
  write memory
  write
  show running-config
  exit
  end
  network country code <1-999>
  mobile network code <0-999>
  short name NAME
  long name NAME
  encryption a5 <0-3> [<0-3>] [<0-3>] [<0-3>]
  authentication (optional|required)
  rrlp mode (none|ms-based|ms-preferred|ass-preferred)
  mm info (0|1)
  timezone <-19-19> (0|15|30|45)
  timezone <-19-19> (0|15|30|45) <0-2>
  no timezone
  periodic location update <6-1530>
  no periodic location update
```

6.3.4 The attribute system

The VTY allows to edit the configuration at runtime. For many VTY commands the configuration change is immediately valid but for some commands a change becomes valid on a certain event only. In some cases it is even necessary to restart the whole process.

To give the user an overview, which configuration change applies when, the VTY implemets a system of attribute flags, which can be displayed using the `show` command with the parameter `vty-attributes`

Example: Typing show vty-attributes at the VTY prompt

```
OsmoBSC> show vty-attributes
  Global attributes:
    ^  This command is hidden (check expert mode)
    !  This command applies immediately
    @  This command applies on VTY node exit
  Library specific attributes:
    A  This command applies on ASP restart
    I  This command applies on IPA link establishment
    L  This command applies on E1 line update
  Application specific attributes:
    o  This command applies on A-bis OML link (re)establishment
    r  This command applies on A-bis RSL link (re)establishment
    l  This command applies for newly created lchans
```

The attributes are symbolized through a single ASCII letter (flag) and do exist in three levels. This is more or less due to the technical aspects of the VTY implementation. For the user, the level of an attribute has only informative purpose.

The global attributes, which can be found under the same attribute letter in every osmocom application, exist on the top level. The Library specific attributes below are used in various osmocom libraries. Like with the global attributes the attribute flag letter stays the same throughout every osmocom application here as well. On the third level one can find the application specific attributes. Those are unique to each osmocom application and the attribute letters may have different meanings in different osmocom applications. To make the user more aware of this, lowercase letters were used as attribute flags.

The `list` command with the parameter `with-flags` displays a list of available commands on the current VTY node, along with attribute columns on the left side. Those columns contain the attribute flag letters to indicate to the user how the command behaves in terms of how and when the configuration change takes effect.

Example: Typing list with-flags at the VTY prompt

```
OsmoBSC(config-net-bts)# list with-flags
  . ...  help
  . ...  list [with-flags]
  . ...  show vty-attributes
  . ...  show vty-attributes (application|library|global)
  . ...  write terminal
  . ...  write file [PATH]
  . ...  write memory
  . ...  write
  . ...  show running-config ❶
  . ...  exit
  . ...  end
  . o..  type (unknown|bs11|nanobts|rbs2000|nokia_site|sysmobts) ❷
  . ...  description .TEXT
  . ...  no description
  . o..  band BAND
  . .r.  cell_identity <0-65535> ❸
  . .r.  dtx uplink [force]
  . .r.  dtx downlink
  . .r.  no dtx uplink
  . .r.  no dtx downlink
  . .r.  location_area_code <0-65535>
  . o..  base_station_id_code <0-63>
```

```
  . o..  ipa unit-id <0-65534> <0-255>
  . o..  ipa rsl-ip A.B.C.D
  . o..  nokia_site skip-reset (0|1)
  ! ...  nokia_site no-local-rel-conf (0|1) ❹
  ! ...  nokia_site bts-reset-timer  <15-100> ❺
```

❶ This command has no attributes assigned.

❷ This command applies on A-bis OML link (re)establishment.

❸ This command applies on A-bis RSL link (re)establishment.

❹, ❺ This command applies immediately.

There are multiple columns because a single command may be associated with multiple attributes at the same time. To improve readability each flag letter gets a dedicated column. Empty spaces in the column are marked with a dot (".")

In some cases the listing will contain commands that are associated with no flags at all. Those commands either play an exceptional role (interactive commands outside "configure terminal", vty node navigation commands, commands to show / write the config file) or will require a full restart of the overall process to take effect.

6.3.5 The expert mode

Some VTY commands are considered relatively dangerous if used in production operation, so the general approach is to hide them. This means that they don't show up anywhere but the source code, but can still be executed. On the one hand, this approach reduces the risk of an accidental invocation and potential service degradation; on the other, it complicates intentional use of the hidden commands.

The VTY features so-called *expert* mode, that makes the hidden commands appear in the interactive help, as well as in the XML VTY reference, just like normal ones. This mode can be activated from the *VIEW* node by invoking the `enable` command with the parameter `expert-mode`. It remains active for the individual VTY session, and gets disabled automatically when the user switches back to the *VIEW* node or terminates the session.

A special attribute in the output of the `list with-flags` command indicates whether a given command is hidden in normal mode, or is a regular command:

Example: Hidden commands in the output of the list with-flags command

```
OsmoBSC> enable expert-mode ❶
OsmoBSC# list with-flags
  ...
  ^   bts <0-255> (activate-all-lchan|deactivate-all-lchan) ❷
  ^   bts <0-255> trx <0-255> (activate-all-lchan|deactivate-all-lchan) ❸
  .   bts <0-255> trx <0-255> timeslot <0-7> sub-slot <0-7> mdcx A.B.C.D <0-65535> ❹
  ^   bts <0-255> trx <0-255> timeslot <0-7> sub-slot <0-7> (borken|unused) ❺
  .   bts <0-255> trx <0-255> timeslot <0-7> sub-slot <0-7> handover <0-255> ❻
  .   bts <0-255> trx <0-255> timeslot <0-7> sub-slot <0-7> assignment ❼
  .   bts <0-255> smscb-command (normal|schedule|default) <1-4> HEXSTRING ❽
  ...
```

❶ This command enables the *expert* mode.

❷, ❸, ❺ This is a hidden command (only shown in the *expert* mode).

❹, ❻, ❼, ❽ This is a regular command that is always shown regardless of the mode.

7 libosmocore Logging System

In any reasonably complex software it is important to understand how to enable and configure logging in order to get a better insight into what is happening, and to be able to follow the course of action. We therefore ask the reader to bear with us while we explain how the logging subsystem works and how it is configured.

Most Osmocom Software (like `osmo-bts`, `osmo-bsc`, `osmo-nitb`, `osmo-sgsn` and many others) uses the same common logging system.

This chapter describes the architecture and configuration of this common logging system.

The logging system is composed of

- log targets (where to log),
- log categories (who is creating the log line),
- log levels (controlling the verbosity of logging), and
- log filters (filtering or suppressing certain messages).

All logging is done in human-readable ASCII-text. The logging system is configured by means of VTY commands that can either be entered interactively, or read from a configuration file at process start time.

7.1 Log categories

Each sub-system of the program in question typically logs its messages as a different category, allowing fine-grained control over which log messages you will or will not see. For example, in OsmoBSC, there are categories for the protocol layers `rsl`, `rr`, `mm`, `cc` and many others. To get a list of categories interactively on the vty, type: `logging level ?`

7.2 Log levels

For each of the log categories (see Section 7.1), you can set an independent log level, controlling the level of verbosity. Log levels include:

fatal
: Fatal messages, causing abort and/or re-start of a process. This *shouldn't happen*.

error
: An actual error has occurred, its cause should be further investigated by the administrator.

notice
: A noticeable event has occurred, which is not considered to be an error.

info
: Some information about normal/regular system activity is provided.

debug
: Verbose information about internal processing of the system, used for debugging purpose. This will log the most.

The log levels are inclusive, e.g. if you select *info*, then this really means that all events with a level of at least *info* will be logged, i.e. including events of *notice*, *error* and *fatal*.

So for example, in OsmoBSC, to set the log level of the Mobility Management category to info, you can use the following command: `log level mm info`.

There is also a special command to set all categories as a one-off to a desired log level. For example, to silence all messages but those logged as notice and above issue the command: `log level set-all notice`

Afterwards you can adjust specific categories as usual.

A similar command is `log level force-all <level>` which causes all categories to behave as if set to log level <level> until the command is reverted with `no log level force-all` after which the individually-configured log levels will again take effect. The difference between `set-all` and `force-all` is that `set-all` actually changes the individual category settings while `force-all` is a (temporary) override of those settings and does not change them.

7.3 Log printing options

The logging system has various options to change the information displayed in the log message.

log color 1
With this option each log message will log with the color of its category. The color is hard-coded and can not be changed. As with other options a *0* disables this functionality.

log timestamp 1
Includes the current time in the log message. When logging to syslog this option should not be needed, but may come in handy when debugging an issue while logging to file.

log print extended-timestamp 1
In order to debug time-critical issues this option will print a timestamp with millisecond granularity.

log print category 1
Prefix each log message with the category name.

log print category-hex 1
Prefix each log message with the category number in hex (*<000b>*).

log print level 1
Prefix each log message with the name of the log level.

log print file 1
Prefix each log message with the source file and line number. Append the keyword `last` to append the file information instead of prefixing it.

7.4 Log filters

The default behavior is to filter out everything, i.e. not to log anything. The reason is quite simple: On a busy production setup, logging all events for a given subsystem may very quickly be flooding your console before you have a chance to set a more restrictive filter.

To request no filtering, i.e. see all messages, you may use: `log filter all 1`

In addition to generic filtering, applications can implement special log filters using the same framework to filter on particular context.

For example in OsmoBSC, to only see messages relating to a particular subscriber identified by his IMSI, you may use: `log filter imsi 262020123456789`

7.5 Log targets

Each of the log targets represent certain destination for log messages. It can be configured independently by selecting levels (see Section 7.2) for categories (see Section 7.1) as well as filtering (see Section 7.4) and other options like `logging timestamp` for example.

7.5.1 Logging to the VTY

Logging messages to the interactive command-line interface (VTY) is most useful for occasional investigation by the system administrator.

Logging to the VTY is disabled by default, and needs to be enabled explicitly for each such session. This means that multiple concurrent VTY sessions each have their own logging configuration. Once you close a VTY session, the log target will be destroyed and your log settings be lost. If you re-connect to the VTY, you have to again activate and configure logging, if you wish.

To create a logging target bound to a VTY, you have to use the following command: `logging enable` This doesn't really activate the generation of any output messages yet, it merely creates and attaches a log target to the VTY session. The newly-created target still doesn't have any filter installed, i.e. *all log messages will be suppressed by default*

Next, you can configure the log levels for desired categories in your VTY session. See Section 7.1 for more details on categories and Section 7.2 for the log level details.

For example, to set the log level of the Call Control category to debug, you can use: `log level cc debug`

Finally, after having configured the levels, you still need to set the filter as it's described in Section 7.4.

Tip
If many messages are being logged to a VTY session, it may be hard to impossible to still use the same session for any commands. We therefore recommend to open a second VTY session in parallel, and use one only for logging, while the other is used for interacting with the system. Another option would be to use different log target.

To review the current vty logging configuration, you can use: `show logging vty`

7.5.2 Logging to the ring buffer

To avoid having separate VTY session just for logging output while still having immediate access to them, one can use `alarms` target. It lets you store the log messages inside the ring buffer of a given size which is available with `show alarms` command.

It's configured as follows:

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# log alarms 98
OsmoBSC(config-log)#
```

In the example above 98 is the desired size of the ring buffer (number of messages). Once it's filled, the incoming log messages will push out the oldest messages available in the buffer.

7.5.3 Logging via gsmtap

GSMTAP is normally a pseudo-header format that enables the IP-transport of GSM (or other telecom) protocols that are not normally transported over IP. For example, the most common situation is to enable GSMTAP in OsmoBTS or OsmoPCU to provide GSM-Um air interface capture files over IP, so they can be analyzed in wireshark.

GSMTAP logging is now a method how Osmocom software can also encapsulate its own log output in GSMTAP frames. We're not trying to re-invent rsyslog here, but this is very handy When debugging complex issues. It enables the reader of the pcap file containing GSMTAP logging together with other protocol traces to reconstruct exact chain of events. A single pcap file can then contain both the log output of any number of Osmocom programs in the same timeline of the messages on various interfaces in and out of said Osmocom programs.

It's configured as follows:

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# log gsmtap 192.168.2.3
OsmoBSC(config-log)#
```

The hostname/ip argument is optional: if omitted the default 127.0.0.1 will be used. The log strings inside GSMTAP are already supported by Wireshark. Capturing for `port 4729` on appropriate interface will reveal log messages including source file name and line number as well as application. This makes it easy to consolidate logs from several different network components alongside the air frames. You can also use Wireshark to quickly filter logs for a given subsystem, severity, file name etc.

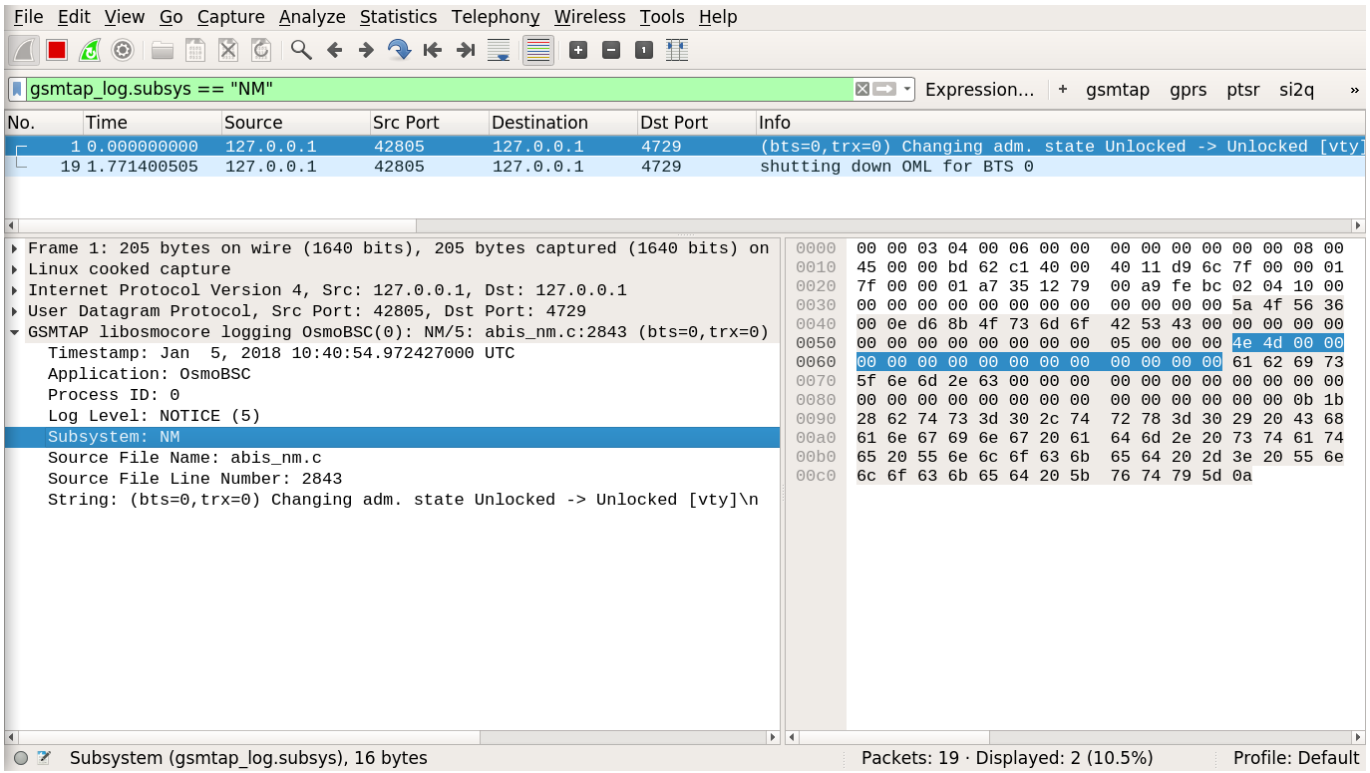


Figure 4: Wireshark with logs delivered over GSMTAP

Note: the logs are also duplicated to stderr when GSMTAP logging is configured because stderr is the default log target which is initialized automatically. To descrease stderr logging to absolute minimum, you can configure it as follows:

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# log stderr
OsmoBSC(config-log)# logging level force-all fatal
```

Note

Every time you generate GSMTAP messages and send it to a unicast (non-broadcast/multicast) IP address, please make sure that the destination IP address actually has a socket open on the specified port, or drops the packets in its packet filter. If unicast GSMTAP messages arrive at a closed destination UDP port, the operating system will likely generate ICMP port unreachable messages. Those ICMP messages in turn will, when arriving at the source (the host on which you run the Osmocom software sending GSMTAP), suppress generation of further GSMTAP messages for some time, resulting in incomplete files. In case of doubt, either send GSMTAP to multicast IP addresses, or run something like `nc -l -u -p 4729 > /dev/null` on the destination host to open the socket at the GSMTAP port and discard anything arriving at it.

7.5.4 Logging to a file

As opposed to Logging to the VTY, logging to files is persistent and stored in the configuration file. As such, it is configured in sub-nodes below the configuration node. There can be any number of log files active, each of them having different settings regarding levels / subsystems.

To configure a new log file, enter the following sequence of commands:

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# log file /path/to/my/file
OsmoBSC(config-log)#
```

This leaves you at the config-log prompt, from where you can set the detailed configuration for this log file. The available commands at this point are identical to configuring logging on the VTY, they include `logging filter`, `logging level` as well as `logging color` and `logging timestamp`.

Tip
Don't forget to use the `copy running-config startup-config` (or its short-hand `write file`) command to make your logging configuration persistent across application re-start.

Note
libosmocore provides file close-and-reopen support by SIGHUP, as used by popular log file rotating solutions such as https://github.com/logrotate/logrotate found in most GNU/Linux distributions.

7.5.5 Logging to syslog

syslog is a standard for computer data logging maintained by the IETF. Unix-like operating systems like GNU/Linux provide several syslog compatible log daemons that receive log messages generated by application programs.

libosmocore based applications can log messages to syslog by using the syslog log target. You can configure syslog logging by issuing the following commands on the VTY:

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# log syslog daemon
OsmoBSC(config-log)#
```

This leaves you at the config-log prompt, from where you can set the detailed configuration for this log file. The available commands at this point are identical to configuring logging on the VTY, they include `logging filter`, `logging level` as well as `logging color` and `logging timestamp`.

Note
Syslog daemons will normally automatically prefix every message with a time-stamp, so you should disable the libosmocore time-stamping by issuing the `logging timestamp 0` command.

7.5.6 Logging to systemd-journal

systemd has been adopted by the majority of modern GNU/Linux distributions. Along with various daemons and utilities it provides systemd-journald [1] - a daemon responsible for event logging (syslog replacement). libosmocore based applications can log messages directly to systemd-journald.

The key difference from other logging targets is that systemd based logging allows to offload rendering of the meta information, such as location (file name, line number), subsystem, and logging level, to systemd-journald. Furthermore, systemd allows to attach arbitrary meta fields to the logging messages [2], which can be used for advanced log filtering.

[1] https://www.freedesktop.org/software/systemd/man/systemd-journald.service.html [2] https://www.freedesktop.org/software/-systemd/man/systemd.journal-fields.html

It was decided to introduce libsystemd as an optional dependency, so it needs to be enabled explicitly at configure/build time:

```
$ ./configure --enable-systemd-logging
```

Note
Recent libosmocore packages provided by Osmocom for Debian and CentOS are compiled **with** libsystemd (https://gerrit.osmocom.org/c/libosmocore/+/22651).

You can configure systemd based logging in two ways:

Example: systemd-journal target with offloaded rendering

```
log systemd-journal raw ❶
 logging filter all 1
 logging level set-all notice
```

❶ `raw` logging handler, rendering offloaded to systemd.

In this example, logging messages will be passed to systemd without any meta information (time, location, level, category) in the text itself, so all the printing parameters like `logging print file` will be ignored. Instead, the meta information is passed separately as *fields* which can be retrieved from the journal and rendered in any preferred way.

```
# Show Osmocom specific fields
$ journalctl --fields | grep OSMO

# Filter messages by logging subsystem at run-time
$ journalctl OSMO_SUBSYS=DMSC -f

# Render specific fields only
$ journalctl --output=verbose \
        --output-fields=SYSLOG_IDENTIFIER,OSMO_SUBSYS,CODE_FILE,CODE_LINE,MESSAGE
```

See `man 7 systemd.journal-fields` for a list of default fields, and `man 1 journalctl` for general information and available formatters.

Example: systemd-journal target with libosmocore based rendering

```
log systemd-journal ❶
 logging filter all 1
 logging print file basename
 logging print category-hex 0
 logging print category 1
 logging print level 1
 logging timestamp 0 ❷
 logging color 1 ❸
 logging level set-all notice
```

❶ Generic logging handler, rendering is done by libosmocore.

❷ Disable timestamping, systemd will timestamp every message anyway.

❸ Colored messages can be rendered with `journalctl --output=cat`.

In this example, logging messages will be pre-processed by libosmocore before being passed to systemd. No additional fields will be attached, except the logging level (`PRIORITY`). This mode is similar to *syslog* and *stderr*.

7.5.7 Logging to stderr

If you're not running the respective application as a daemon in the background, you can also use the stderr log target in order to log to the standard error file descriptor of the process.

In order to configure logging to stderr, you can use the following commands:

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# log stderr
OsmoBSC(config-log)#
```

8 Signaling Networks: SS7 and SIGTRAN

Classic digital telephony networks (whether wired or wireless) use the ITU-T SS7 (Signaling System 7) to exchange signaling information between network elements.

Most of the ETSI/3GPP interfaces in the GSM and UMTS network are also based on top of [parts of] SS7. This includes, among others, the following interfaces:

- *A* interface between BSC and MSC
- *IuCS* interface between RNC (or HNB-GW) and MSC
- *IuPS* interface between RNC (or HNB-GW) and SGSN

Note
This does not include the A-bis interface between BTS and BSC. While Abis traditionally is spoken over the same physical TDM circuits as SS7, the protocol stack from L2 upwards is quite different (Abis uses LAPD, while SS7 uses MTP)!

8.1 Physical Layer

The traditional physical layer of SS7 is based on TDM (time division multiplex) links of the PDH/SDH family, as they were common in ISDN networks. Some people may know their smallest incarnation as so-called E1/T1 links. It can run either on individual 64kBps timeslots of such a link, or on entire 2Mbps/1.5MBps E1/T1 links.

There are also specifications for SS7 over ATM, though it is unclear to the author if this is actually still used anywhere.

On top of the Physical Layer is the Message Transfer Part (MTP).

8.2 Message Transfer Part (MTP)

MTP is the lower layer of the SS7 protocol stack. It is comprised of two sub-layers, called MTP2 and MTP3.

Nodes in a MTP network are addressed by their unique PC (Point Code).

A *MTP Routing Label* is in the MTP header and indicates the *Originationg Point Code* (OPC) as well as the *Destination Point Code* (DPC) and the *Service Indicator Octet* (SIO). The SIO is used to de-multiplex between different upper-layer protocol such as ISUP, TUP or SCCP.

Routing is performed by means of routers with routing tables, similar to routing is performed in IP networks. Even the concept of a *point code mask* analogous to the *netmask* exists.

Routers are connected with one another over one or more *Link Sets*, each comprised of one or multiple *Links*. Multiple Links in a Linkset exist both for load sharing as well as for fail over purposes.

8.2.1 Point Codes

The length of point codes depends on the particular MTP dialect that is used. In the 1980s, when international telephony signaling networks were established, most countries had their own national dialects with certain specifics.

Today, mostly the ITU and ANSI variants survive. The ITU variant uses 14bit point codes, while the ANSI variant uses 24 bit point code length.

Point Codes can be represented either as unsigned integers, or grouped. Unfortunately there is no standard as to their representation. In ITU networks, the *3.8.3* notation is commonly used, i.e. one decimal for the first 3 bits, followed by one decimal for the center 8 bits, followed by another decimal for the final 3 bits.

Example
The Point Code **1.5.3** (in 3.8.3 notation) is $1*2^{11} + 5*2^3 + 3$ = **2091 decimal**.

8.3 Higher-Layer Protocols

There are various higher-layer protocols used on top of MTP3, such as TUP, ISUP, BICC as well as SCCP. Those protocols exist side-by-side on top of MTP3, similar to e.g. ICMP, TCP and UDP existing side-by-side on top of IP.

In the context of cellular networks, SCCP is the most relevant part.

8.4 Signaling Connection Control Part (SCCP)

SCCP runs on top of MTP3 and creates something like an overlay network on top of it. SCCP communication can e.g. span multiple different isolated MTP networks, each with their own MTP dialect and addressing.

SCCP provides both connectionless (datagram) and connection-oriented services. Both are used in the context of cellular networks.

8.4.1 SCCP Addresses

SCCP Addresses are quite complex. This is due to the fact that it is not simply one address format, but in fact a choice of one or multiple different types of addresses.

SCCP Addresses exist as *Calling Party* and *Called Party* addresses. In the context of connectionless datagram services, the sender is always the Calling Party, and the receiver the Called Party. In connection-oriented SCCP, they resemble the initiator and recipient of the connection.

Table 4: SCCP Address Parts

Acronym	Name	Description
SSN	Sub-System Number	Describes a given application such as e.g. a GSM MSC, BSC or HLR. Can be compared to port numbers on the Internet
PC	Point Code	The Point Code of the underlying MTP network
GT	Global Title	What most people would call a "phone number". However, Global Titles come in many different numbering plans, and only one of them (E.164) resembles actual phone numbers.
RI	Routing Indicator	Determines if message shall be routed on PC+SSN or on GT basis

8.4.2 Global Titles

A Global Title is a (typically) globally unique address in the global telephony network. The body of the Global Title consists of a series of BCD-encoded digits similar to what everyone knows as phone numbers.

A GT is however not only the digits of the "phone number", but also some other equally important information, such as the *Numbering Plan* as well as the *Nature of Address Indication*.

Table 5: Global Title Parts

Acronym	Name	Description
GTI	Global Title Indicator	Determines the GT Format. Ranges from no GT (0) to GT+TT+NP+ES+NAI (4)
NAI	Nature of Address Indicator	Exists in GTI=1 and is sort of a mixture of TON + NPI
TT	Translation Type	Used as a look-up key in Global Title Translation Tables
NP	Numbering Plan	Indicates ITU Numbering Plan, such as E.164, E.212, E.214
ES	Encoding Scheme	Just a peculiar way to indicate the length of the digits
-	Signals	The actual "phone number digits"

For more information about SCCP Addresses and Global Titles, please refer to [itu-t-q713]

8.4.3 Global Title Translation (GTT)

Global Title Translation is a process of re-writing the Global Title on-the-fly while a signaling message passes a STP.

Basically, a SCCP message is first transported by MTP3 on the MTP level to the Destination Point Code indicated in the MTP Routing Label. This process uses MTP routing and is transparent to SCCP.

Once the SCCP message arrives at the MTP End-Node identified by the Destination Point Code, the message is handed up to the local SCCP stack, which then may implement Global Title Translation.

The input to the GTT process is

- the destination address of the SCCP message
- a local list/database of Global Title Translation Rules

The successful output of he GTT includes

- A new Routing Indicator
- The Destination Point Code to which the message is forwarded on MTP level
- a Sub-system Number (if RI is set to "Route on SSN")
- a new Global Title (if RI is set to "Route on GT"), e.g. with translated digits.

Between sender and recipient of a signaling message, there can be many instances of Global Title Translation (up to 15 as per the hop counter).

For more information on Global Title Translation, please refer to [itu-t-q714].

8.4.4 Peculiarities of Connection Oriented SCCP

Interestingly, Connection-Oriented SCCP messages carry SCCP Addresses **only during connection establishment**. All data messages during an ongoing connection do not contain a Called or Calling Party Address. Instead, they are routed only by the MTP label, which is constructed from point code information saved at the time the connection is established.

This means that connection-oriented SCCP can not be routed across MTP network boundaries the same way as connectionless SCCP messages. Instead, an STP would have to perform *connection coupling*, which is basically the equivalent of an application-level proxy between two SCCP connections, each over one of the two MTP networks.

This is probably mostly of theoretical relevance, as connection-oriented SCCP is primarily used between RAN and CN of cellular network inside one operator, i.e. not across multiple MTP networks.

8.5 SIGTRAN - SS7 over IP Networks

At some point, IP based networks became more dominant than classic ISDN networks, and 3GPP as well as IETF were working out methods in which telecom signaling traffic can be adapted over IP based networks.

Initially, only the edge of the network (i.e. the applications talking to the network, such as HLR or MSC) were attached to the existing old SS7 backbone by means as SUA and M3UA. Over time, even the links of the actual network backbone networks became more and more IP based.

In order to replace existing TDM-based SS7 links/linksets with SIGTRAN, the M2UA or M2PA variants are used as a kind of drop-in replacement for physical links.

All SIGTRAN share that while they use IP, they don't use TCP or UDP but operate over a (then) newly-introduced Layer 4 transport protocol on top of IP: SCTP (Stream Control Transmission Protocol).

Despite first being specified in October 2000 as IETF RFC 2960, it took a long time until solid implementations of SCTP ended up in general-purpose operating systems. SCTP is not used much outside the context of SIGTRAN, which means implementations often suffer from bugs, and many parts of the public Internet do not carry SCTP traffic due to restrictive firewalls and/or ignorant network administrators.

8.5.1 SIGTRAN Concepts / Terminology

Like every protocol or technology, SIGTRAN brings with it its own terminology and concepts. This section tries to briefly introduce them. For more information, please see the related IETF RFCs.

8.5.1.1 Signaling Gateway (SG)

The Signaling Gateway (SG) interconnects the SS7 network with external applications. It translates (parts of) the SS7 protocol stack into an IP based SIGTRAN protocol stack. Which parts at which level of the protocol stack are translated to what depends on the specific SIGTRAN dialect.

A SG is traditionally attached to the TDM-Based SS7 network and offers SIGTRAN/IP based applications a way to remotely attach to the SS7 network.

A SG typically has STP functionality built-in, but it is not mandatory.

8.5.1.2 Application Server (AS)

An Application Server is basically a logical entity representing one particular external application (from the SS7 point of view) which is interfaced with the SS7 network by means of one of the SIGTRAN protocols.

An Application Server can have one or more Application Server Processes associated with it. This functionality can be used for load-balancing or fail-over scenarios.

8.5.1.3 Application Server Process (ASP)

An Application Server Process represents one particular SCTP connection used for SIGTRAN signaling between an external application (e.g. a BSC) and the Signaling Gateway (SG).

One Application Server Process can route traffic for multiple Application Servers. In order to differentiate traffic for different Application Servers, the Routing Context header is used.

8.5.2 SIGTRAN variants / stackings

SIGTRAN is the name of an IETF working group, which has released an entire group of different protocol specifications. So rather than one way of transporting classic telecom signaling over IP, there are now half a dozen different ones, and all can claim to be an official IETF standard.

FIXME: Overview picture comparing the different stackings

8.5.2.1 MTP3 User Adaptation (M3UA)

M3UA basically "chops off" everything up to and including the MTP3 protocol layer of the SS7 protocol stack and replaces it with a stack comprised of M3UA over SCTP over IP.

M3UA is specified in [ietf-rfc4666].

M3UA is the SIGTRAN variant chosen by 3GPP for A, IuCs and IuPS interfaces over IP.

8.5.2.2 SCCP User Adaptation (SUA)

SUA basically "chops off" everything up to and including the SCCP protocol layer of the SS7 protocol stack and replaces it with a stack comprised of SUA over SCTP over IP.

This means that SUA can only be used for SCCP based signaling, but not for other SS7 protocols like e.g. TUP and ISUP.

SUA is specified in [ietf-rfc3868].

8.5.2.3 MTP2 User Adaptation (M2UA)

M2UA is specified in [ietf-rfc3331].

Note
M2UA is not supported in Osmocom SIGTRAN up to this point. Let us know if we can implement it for you!

8.5.2.4 MTP2-User Peer-to-Peer Adaptation (M2PA)

M2PA is specified in [ietf-rfc4165].

Note
M2PA is not supported in Osmocom SIGTRAN up to this point. Let us know if we can implement it for you!

8.5.3 SIGTRAN security

There simply is none. There are some hints that TLS shall be used over SCTP in order to provide authenticity and/or confidentiality for SIGTRAN, but this is not widely used.

As telecom signaling is not generally carried over public networks, private networks/links by means of MPLS, VLANs or VPNs such as IPsec are often used to isolate and/or secure SIGTRAN.

Under no circumstances should you use unsecured SIGTRAN with production data over the public internet!

8.5.4 IPv6 support

SCTP (and thus all the higher layer protocols of the various SIGTRAN stackings) operates on top of both IPv4 and IPv6. As the entire underlying IP transport is transparent to the SS7/SCCP applications, there is no restriction on whether to use SIGTRAN over IPv4 or IPv6.

8.5.5 SCTP multi-homing in SIGTRAN

SCTP, unlike more traditional IP L4 protocols (TCP, UDP) doesn't work based on a *connection* between source IP:port and Destination IP:port.

Instead, SCTP creates *associations* between two endpoints, both of which can have any number of IP addresses. This means that in case of network outage, traffic can continue to flow through any of the IP addresses of that association.

The Linux kernel by default advertises all IP addresses of the local system to the peer. This can be seen when inspecting the SCTP INIT chunk e.g. in wireshark. While this may be a reasonable default in some use cases, it is not always the best idea. Imagine addresses of internal/private IP networks, for example local bridge devices between lxc or docker containers, or local VMs. Such addresses have no significance beyond the local machine.

Subsequently, libosmo-sigtran allows the user to explicitly select which local IP addresses shall be used in SCTP multi-homing for the SIGTRAN associations it manages. The user can achieve this by specifying multiple `local-ip` VTY commands within one `asp` (SCTP client role) or within one `listen m3ua 2905` (SCTP server role).

8.5.6 SCTP Primary Address

SCTP has the concept of "primary address" in an association. The primary address is a remote address selected from those announced by the peer, and it is the "active" one chosen to transmit user data. The other remote addresses, that are not used, are kept as backups. They are in general only used to transmit user data whenever the SCTP implementation decides to change the primary address, be it due to user policy configuration change or due to the previous primary link becoming unusable. Only confirmed remote addresses (through HEARTBEAT mechanism) are electable to be used as primary address.

By default, the Linux kernel SCTP stack implementation will probably take the first remote address provided at connect() time in order to start the initial handshake, and continue with next provided remote addresses if the first one fails to confirm the handshake. The remote address which successfully confirmed the handshake is then used as a primary address (since it's likely the only confirmed so far), and will be kept until the link is considered down.

Some deployment setups may have requirements on preferred links to be used when transmitting data (eg. network setups with primary and secondary paths). This can be accomplished by explicitly notifying the kernel to use one of the remote addresses through the SCTP_PRIMARY_ADDR sockopt, plus monitoring the address availability changes on the socket and re-enforcing the primary address when it becomes available again. This is supported in the Osmocom SIGTRAN stack by using the `primary` parameter in one of the `remote-ip` commands under the `asp` node:

```
cs7 instance 0
 asp my-asp 2905 0 m3ua
  remote-ip 10.11.12.13
  remote-ip 16.17.18.19 primary ❶
  ...
```

❶ Use 16.17.18.19 as primary address for the SCTP association. User data will be in general transmitted over this path.

8.5.7 SCTP Peer Primary Address

The SCTP extension ASCONF (RFC5061) allows, when negotiated and supported by both peers, to dynamically announce to the peer the addition or deletion of IP addresses to the association. It also allows one peer announcing to the other peer the desired IP address it should be using as a primary address when sending data to it.

In the Linux kernel SCTP stack, this is accomplished by setting the sockopt SCTP_SET_PEER_PRIMARY_ADDR, which will trigger an ASCONF SCTP message to the peer with the provided local IP address. This is supported in the Osmocom SIGTRAN stack by using the `primary` parameter in one of the `local-ip` commands under the `asp` node:

```
cs7 instance 0
 asp my-asp 2905 0 m3ua
  local-ip 10.11.12.13
  local-ip 16.17.18.19 primary ❶
  ...
```

❶ Announce 16.17.18.19 to the peer as the primary address to be used when transmitting user data to us.

In order to be able to use this feature, the SCTP association peer must support the ASCONF extension. The extension support is negotiated during the INIT handshake of the association. Furthermore, for ASCONF features to work properly, the assoc also needs to announce/use the AUTH extension, as per RFC5061 section 4.2.7. Otherwise, the peer receiving an SCTP INIT with `ExtensionFeatures=ASCONF,ASCONF_ACK` but without AUTH, will reject the association with an ABORT since it's not complying with specifications (this behavior can be tweaked through sysctl "net.sctp.addip_noauth_enable").

As of the time of writing this documentation (linux 6.4.12) and since basically ever, those extensions are runtime-disabled by default. They can be enabled per socket using the kernel sockopts SCTP_ASCONF_SUPPORTED and SCTP_AUTH_SUPPORTED, and that's what the Osmocom stack is currently doing for all its SCTP sockets.

However, those sockopts are farily new (linux v5.4), which means user running older kernels will see in the logs setting those sockopts fail, but connection will keep ongoing, simply without those features available (so setting `primary` in the configuration won't have any effect here). On those older kernels, if this feature is still desired, it can be used by means of enabling the SCTP extensions in all socket system-wide through sysctl:

```
net.sctp.auth_enable=1
net.sctp.addip_enable=1
```

8.5.8 SCTP INIT Parameters

Several SCTP INIT parameters can be configured through VTY, which will be passed to the Linux Kernel SCTP stack and used whenever an association is being established.

On the client side (see Section 8.5.9), the parameters are configured in the `asp` node:

```
cs7 instance 0
 asp my-asp 2905 0 m3ua
  sctp-role client
  sctp-param init num-ostreams 250 ❶
  sctp-param init max-instreams 300 ❷
  sctp-param init max-attempts 3 ❸
  sctp-param init timeout 10000 ❹
  ...
```

❶ The number of streams from the server to the client. This value is transmitted during SCTP INIT ACK packet.

❷ Announce to the server that a maximum of up to 300 inbound SCTP streams are supported. This value is transmitted during SCTP INIT packet.

❸ Initial SCTP handshake will be attempted 3 times before considering the connection failed.

❹ Retransmit an SCTP INIT message after 10000 ms if no answer is received.

On the server side (see Section 8.5.9), the parameters are configured in the `listen` node:

```
cs7 instance 0
 asp my-asp 2905 0 m3ua
  sctp-role server
 listen m3ua 2905
  sctp-param init num-ostreams 250 ❶
  sctp-param init max-instreams 300 ❷
  ...
```

1. Announce to the server that up to 250 outbound SCTP streams (server to client) may be requested. This value is transmitted during SCTP INIT packet, and should be equal or lower to the `max-instreams` value received from the client during SCTP INIT packet.
2. Announce to the server that a maximum of up to 300 inbound SCTP streams are supported. This value is transmitted during SCTP INIT ACK packet.

8.5.9 SCTP role

The *SCTP role* defines which of the two L4 protocol roles SCTP assumes:

- The *SCTP server* role binds to a local port and handles incoming connections from clients
- The *SCTP client* role connects to a remote SCTP sever.

8.5.10 M3UA/SUA role

The *M3UA role* (or *SUA role*) determines which role a given peer of a M3UA connection implements. 3GPP specifies the following role:

- *SGP* (Signaling Gateway): The entity connected to the larger SS7 network
- *ASP* (Application Server Process): A client application that connects to the SGW to talk to the SS7 network
- *IPSP* (IP Server Process): M3UA in point-to-point mode

Osmocom (libosmo-sigtran) implements both the SGP and ASP roles, but not the IPSP role.

8.5.11 Traffic Modes in SIGTRAN

Whenever an AS consists of multiple ASPs, the traffic mode expresses how messages are distributed between those ASPs.

- *Override*: There is always one active ASP and multiple hot standby ASPs. If the active ASP fails, one of the remaining ASPs will become the new active ASP.
- *Loadshare*: The messages will be distributed between the different ASPs in a way to distribute the load among them. Details are implementation specific.
- *Broadcast*: A copy of every incoming signaling message is sent to *all* the ASPs in broadcast traffic mode.

Osmocom (libosmo-sigtran) implements all above-mentioned traffic modes.

9 Osmocom SS7 + SIGTRAN support

9.1 History / Background

If you're upgrading from earlier releases of the Osmocom stack, this section will give you some background about the evolution.

9.1.1 The Past (before 2017)

In the original implementation of the GSM BSC inside Osmocom (the OsmoBSC program, part of OpenBSC), no SS7 support was included.

This is despite the fact that ETSI/3GPP mandated the use of SCCP over MTP over E1/T1 TDM lines for the A interface at that time.

Instead of going down to the TDM based legacy physical layers, OsmoBSC implemented something called an IPA multiplex, which apparently some people also refer to as SCCPlite. We have never seen any specifications for this interface, but implemented it from scratch using protocol traces.

The IPA protocol stack is based on a minimal sub-set of SCCP (including connection oriented SCCP) wrapped into a 3-byte header to packetize a TCP stream.

The IPA/SCCPlite based A interface existed at a time when the ETSI/3GPP specifications did not offer any IP based transport for the A interface. An official as added only in Release FIXME of the 3GPP specifications.

The A interface BSSMAP protocol refers to voice circuits (E1/T1 timeslots) using circuit identity codes (CICs). As there are no physical timeslots on a TCP/IP based transport layer, the CICs get mapped to RTP streams for circuit-switched data using out-of-band signaling via MGCP, the IETF-standardized Media Gateway Control Protocol.

9.1.2 The present (2017)

In 2017, sysmocom was tasked with implementing a 3GPP AoIP compliant A interface. This meant that a lot of things had to change in the existing code:

- removal of the existing hard-wired SCCPlite/IPA code from OsmoBSC
- introduction of a formal SCCP User SAP at the lower boundary of BSSMAP
- introduction of libosmo-sigtran, a comprehensive SS7 and SIGTRAN library which includes a SCCP implementation for connectionless and connection-oriented procedures, offering the SCCP User SAP towards BSSAP
- introduction of an A interface in OsmoMSC (which so far offered Iu only)
- port of the existing SUA-based IuCS and IuPS over to the SCCP User SAP of libosmo-sigtran.
- Implementation of ETSI M3UA as preferred/primary transport layer for SCCP
- Implementation of an IPA transport layer inside libosmo-sigtran, in order to keep backwards-compatibility.

This work enables the Osmocom universe to become more compliant with modern Releases of 3GPP specifications, which enables interoperability with other MSCs or even BSCs. However, this comes at a price: Increased complexity in set-up and configuration.

Using SS7 or SIGTRAN based transport of the A interface adds an entirely new domain that needs to be understood by system and network administrators setting up cellular networks based on Osmocom.

One of the key advantages of the Osmocom architecture with OsmoNITB was exactly this simplification and reduction of complexity, enabling more people to set-up and operate cellular networks.

So we have put some thought into how we can achieve compatibility with SS7/SIGTRAN and the 3GPP specifications, while at the same time enabling some degree of auto-configuration where a small network can be set up without too many configuration related to the signaling network. We have achieved this by "abusing" (or extending) the M3UA Routing Key Management slightly.

9.2 Osmocom extensions to SIGTRAN

Osmocom has implemented some extensions to the SIGTRAN protocol suite. Those extensions will be documented below.

9.2.1 Osmocom M3UA Routing Key Management Extensions

In classic M3UA, a peer identifies its remote peer based on IP address and port details. So once an ASP connects to an SG, the SG will check if there is any configuration that matches the source IP (and possibly source port) of that connection in order to understand which routing context is used - and subsequently which traffic is to be routed to this M3UA peer.

This is quite inflexible, as it means that every BSC in a GSM network needs to be manually pre-configured at the SG/STP, and that configuration on the BSC and MSC must match to enable communication.

M3UA specifies an optional Routing Key Management (RKM) sub-protocol. Using RKM, an ASP can dynamically tell the SG/STP, which traffic it wants to receive. However, the idea is still that the SG has some matching configuration.

In OsmoSTP based on libosmo-sigtran, we decided to (optionally) enable fully dynamic registration. This means that any ASP can simply connect to the SG and request the dynamic creation of an ASP and AS with a corresponding routing key for a given point code. As long as the SG doesn't already have a route to this requested point code, The SG will simply trust any ASP and set a corresponding route.

To enable dynamic creation of ASPs within an AS from any source IP/port, the corresponding xUA Server (Section 9.5) must be configured with `accept-asp-connections dynamic-permitted`.

To enable dynamic registration of routing keys via RKM, the corresponding SS7 Instance (Section 9.4) must be configured with `xua rkm routing-key-allocation dynamic-permitted`.

This is of course highly insecure and can only be used in trusted, internal networks. However, it is quite elegant in reducing the amount of configuration complexity. All that is needed, is that an unique point code is configured at each of the ASPs (application programs) that connect to the STP.

To put things more concretely: Each BSC and MSC connecting to OsmoSTP simply needs to be configured to have a different point code, and to know to which IP/port of the STP to connect. There's no other configuration required for a small, autonomous, self-contained network. OsmoSTP will automatically install ASP, AS and route definitions on demand, and route messages between all connected entities.

The same above of course also applies to HNB-GW and OsmoSGSN in the case of Iu interfaces.

9.2.2 IPA / SCCPlite backwards compatibility

The fundamental problem with IPA/SCCPlite is that there's no MTP routing label surrounding the SCCP message. This is generally problematic in the context of connection-oriented SCCP, as there is no addressing information inside the SCCP messages after the connection has been established. Instead, the messages are routed based on the MTP label, containing point codes established during connection set-up time.

This means that even if the SCCP messages did contain Called/Calling Party Addresses with point codes or global titles, it would only help us for routing connectionless SCCP. The A interface, however, is connection-oriented.

So in order to integrate IPA/SCCPlite with a new full-blown SS7/SIGTRAN stack, there are the following options:

1. implement SCCP connection coupling. This is something like a proxy for connection-oriented SCCP, and is what is used in SS7 to route beyond a given MTP network (e.g. at gateways between different MTP networks).
2. consider all SCCP messages to be destined for the local point code of the receiver. This then means that the SG functionality must be included inside the MSC, and the MSC be bound to the SSN on the local point code.
3. hard-code some DPC when receiving a message from an IPA connection. It could be any remote PC and we'd simply route the message towards that point code.

But then we also have the return direction:

1. We could "assign" a unique SPC to each connected IPA client (BSC), and then announce that PC towards the SS7 side. Return packets would then end up at our IPA-server-bearing STP, which forwards them to the respective IPA connection and thus BSC. On the transmit side, we'd simply strip the MTP routing label and send the raw SCCP message over IPA.
2. If the IPA server / SGW resides within the MSC, one could also have some kind of handle/reference to the specific TCP connection through which the BSC connected. All responses for a given peer would then have to be routed back to the same connection. This is quite ugly as it completely breaks the concepts of the SCCP User SAP, where a user has no information (nor to worry about) any "physical" signaling links.

9.3 Minimal Osmocom SIGTRAN configurations for small networks

If you're not an SS7 expert, and all you want is to run your own small self-contained cellular network, this section explains what you need to do.

In general, you can consider OsmoSTP as something like an IP router. On the application layer (in our case the BSSAP/BSSMAP or RANAP protocols between Radio Access Network and Core Network), it is completely invisible/transparent. The BSC connects via SCCP to the MSC. It doesn't know that there's an STP in between, and that this STP is performing some routing function. Compares this to your web browser not knowing about IP routers, it just establishes an http connection to a web server.

This is also why most GSM network architecture diagrams will not explicitly show an STP. It is not part of the cellular network. Rather, one or many STPs are part of the underlying SS7 signaling transport network, on top of which the cellular network elements are built.

9.3.1 A minimal 2G configuration to get started

You will be running the following programs:

- OsmoBSC as the base-station controller between your BTS (possibly running OsmoBTS) and the MSC
- OsmoMSC as the mobile switching center providing SMS and telephony service to your subscribers
- OsmoSTP as the signal transfer point, routing messages between one or more BSCs and the MSC

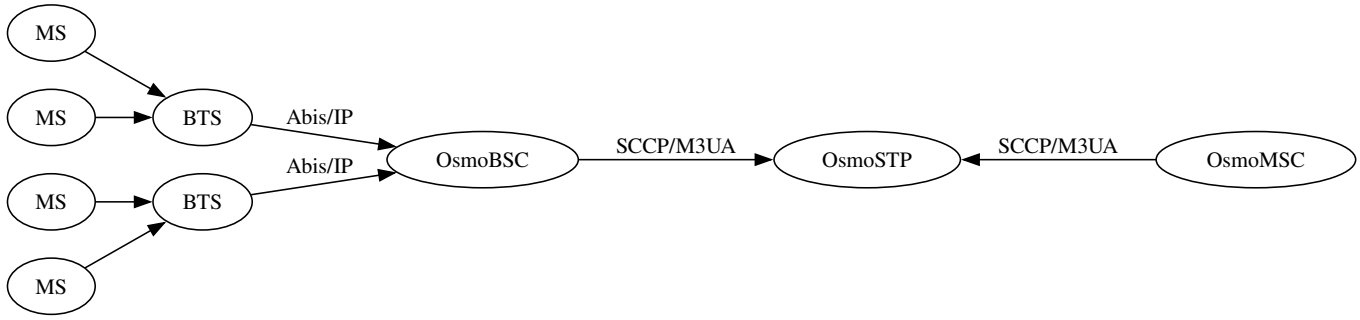


Figure 5: Simple signaling network for 2G (GSM)

You can use the OsmoSTP fully dynamic registration feature, so the BSCs and the MSC will simply register with their point codes to the STP, and the STP will create most configuration on the fly.

All you need to make sure is:

- to assign one unique point code to each BSC and MSC
- to point all BSCs and the MSC to connect to the IP+Port of the STP
- to configure the point code of the MSC in the BSCs

9.3.2 A minimal 3G configuration to get started

You will be running the following programs:

- OsmoHNBGW as the homeNodeB Gateway between your femtocells / small cells and the MSC+SGSN
- OsmoMSC as the mobile switching center providing SMS and telephony service to your subscribers

- OsmoSGSN as the Serving GPRS Support Node, providing packet data (internet) services to your subscribers
- OsmoSTP as the signal transfer point, routing messages between one or more HNBGWs and the MSC and SGSN

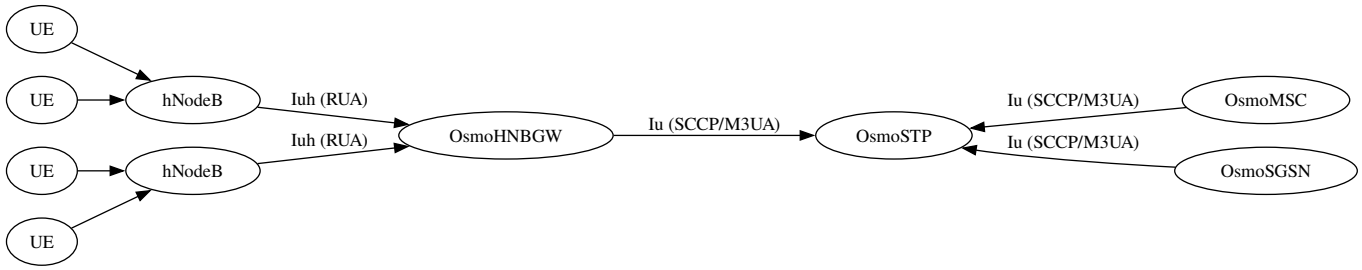


Figure 6: Simple signaling network for 3G (UMTS)

You can use the OsmoSTP fully dynamic registration feature, so the HNBGWs, the MSC and the SGSN will simply register with their point codes to the STP, and the STP will create most configuration on the fly.

All you need to make sure is:

- to assign one unique point code to each HNBGW, MSC and SGSN
- to point all HNBGWs and the MSC and SGSN to connect to the IP+Port of STP
- to configure the point code of the MSC in the HNBGWs
- to configure the point code of the SGSN in the HNBGWs

9.4 Osmocom SS7 Instances

The entire SS7 stack can be operated multiple times within one application/program by means of so-called SS7 Instances.

There can be any number of SS7 Instances, and each instance has its own set of XUA Servers, ASPs, ASs, Routes, etc.

Each SS7 Instance can have different point code formats / lengths.

Table 6: Major Attributes of an Osmocom SS7 Instance

Name	VTY Command	Description
ID	(config)# cs7 instance ID	The numeric identifier of this instance
Name	(config-cs7)# name NAME	A human-readable name for this instance
Description	(config-cs7)# description DESC	More verbose description
Primary PC	(config-cs7)# point-code PC	Primary local point code
Network Indicator	(config-cs7)# network-indicator	Network Indicator used in MTP3 Routing Label
Point Code Format	(config-cs7)# point-code format	Point Code Format (Default: 3.8.3)
Point Code Delimiter	(config-cs7)# point-code delimiter	Point Code Delimiter: . or -

9.5 Osmocom SS7 xUA Server

A **xUA Server** is a server that binds + listens to a given SCTP (SIGTRAN) or TCP (IPA) port and accepts connections from remote peers (ASPs).

There can be any number of xUA Servers within one SS7 Instance, as long as they all run on a different combination of IP address and port.

Table 7: Major Attributes of an Osmocom SS7 xUA Server

Name	Description
Local IP	Local Port Number to which the server shall bind/listen
Local Port	Local IP Address to which the server shall bind/listen
Protocol	Protocol (M3UA, SUA, IPA) to be operated by this server
Accept Dynamic ASPs	Should we accept connections from ASPs that are not explicitly pre-configured with their source IP and port?

9.6 Osmocom SS7 Users

A SS7 User is part of a program that binds to a given MTP-Layer Service Indicator (SI). The Osmocom SS7 stack offers an API to register SS7 Users, as well as the VTY command `show cs7 instance <0-15> users` to list the currently registered users.

9.7 Osmocom SS7 Links

Conceptually, SS7 links are on the same level as SIGTRAN ASPs. The details of SS7 Links in the Osmocom implementation are TBD.

9.8 Osmocom SS7 Linksets

Conceptually, SS7 Linksets are on the same level as SIGTRAN ASs. The details of SS7 Links in the Osmocom implementation are TBD.

9.9 Osmocom SS7 Application Servers

This corresponds 1:1 to the SIGTRAN concept of an Application Server, i.e. a given external Application that interfaces the SS7 network via a SS7 protocol variant such as M3UA.

In the context of Osmocom, for each program connecting to a STP (like a BSC or MSC), you will have one Application Server definition.

An AS has the following properties:

Table 8: Major Attributes of an Osmocom SS7 Application Server

Name	Description
Name	A human-readable name for this instance
Description	More verbose description (for human user only)
Protocol	Protocol (M3UA, SUA, IPA) to be operated by this server
Routing Key	Routing Key (mostly Point Code) routed to this AS
Traffic Mode	Broadcast, Loadshare or Override

Table 8: (continued)

Name	Description
Recovery Timeout	Duration of the AS T(r) recovery timer. During this time, outgoing messages are queued. If the AS is ACTIVE before timer expiration, the queue is drained. At expiration, the queue is flushed.
State	Application Server State (Down, Inactive, Active, Pending)
ASPs	Which ASPs are permitted to transfer traffic for this AS

9.10 Osmocom SS7 Application Server Processes

An Application Server Process corresponds to a given SCTP (or TCP) connection. From the STP/SG (Server) point-of-view, those are incoming connections from Application Servers such as the BSCs. From the ASP (Client) Point of view, it has one `osmo_ss7_asp` object for each outbound SIGTRAN connection.

An ASP has the following properties:

Table 9: Major Attributes of an Osmocom SS7 Application Server Process

Name	Description
Name	A human-readable name for this instance
Description	More verbose description (for human user only)
Protocol	Protocol (M3UA, SUA, IPA) to be operated by this server
Role	Server (SG) or Client (ASP)?
Local Port	Port Number of the local end of the connection
Local IP	IP Address of the local end of the connection
Remote Port	Port Number of the remote end of the connection
Remote IP	IP Address of the remote end of the connection
State	ASP State (Down, Inactive, Active)

9.11 Osmocom SS7 Routes

An Osmocom SS7 Route holds information to route traffic with a matching destination point code and point code mask (similar to IP Address + Netmask) towards a specified SS7 Linkset or Application Server. The Linkset or Application Servers are identified by their name.

Routes are inserted in a routing table, where they are ordered first based on prefix mask length (longest prefix first) and then on priority (lowest value first). As a result, more specific routes (eg. a single Point Code) will match and be used before more generic ones (eg. a subnet).

Table 10: Major Attributes of an Osmocom SS7 Application Server Process

Name	Description
Point Code	Destination Point Code for this route
Mask	Destination Mask for this route (like an IP netmask)
Linkset/AS Name	Destination Linkset or AS, identified by name
Priority	Priority of the route in the routing table; lower value means higher priority

The `show cs7 instance <0-15> route` command can be used on the VTY to obtain the list of routes inserted in the system routing table.

The `show cs7 instance <0-15> route-lookup <DPC> from <OPC> sls <SLS>` command can be used on the VTY to test and debug routing of specific traffic on the currently running routing table, similar to what `ip route get` linux command does in the IP world.

9.12 Osmocom SCCP Instances

An Osmocom SCCP Instance can be bound to an Osmocom SS7 Instance. It will register/bind for the ITU-standard Service Indicator (SI).

9.13 Osmocom SCCP User

An Program (like a BSC) will *bind* itself to a given well-known sub-system number (SSN) in order to receive SCCP messages destined for this SSN.

There is an API to bind a program to a SSN, which implicitly generates an SCCP User object.

The `show cs7 instance <0-15> sccp users` command can be used on the VTY to obtain a list of currently bound SCCP users, as well as their corresponding SSNs.

9.14 Osmocom SCCP Connection

This is how Osmocom represents each individual connection of connection-oriented SCCP.

To illustrate the practical application: For the common use case of the A or Iu interfaces, this means that every dedicated radio channel that is currently active to any UE/MS has one SCCP connection to the MSC and/or SGSN.

The `show cs7 instance <0-15> sccp connections` command can be used on the VTY to obtain a list of currently active SCCP connections, as well as their source/destination and current state.

9.15 Osmocom SCCP User SAP

The Osmocom SCCP User SAP (Service Access Point) is the programming interface between the SCCP Provider (libosmo-sigtran) and the SCCP User (such as osmo-bsc, osmo-msc, osmo-hnbgw, etc.). It follows primitives as laid out in [itu-t-q711], encapsulated in `osmo_prim` structures.

9.16 Osmocom MTP User SAP

The Osmocom MTP User SAP (Service Access Point) is the programming interface between the MTP Provider and the MTP User (e.g. SCCP). It follows primitives as laid out in [itu-t-q711], encapsulated in `osmo_prim` structures.

10 Configure SCCP/M3UA

All CNI programs using SCCP/M3UA act as M3UA ASP role and SCTP client, expecting to connect to a Signalling Gateway (STP/SG) implementing the M3UA SG role as SCTP server. The STP/SG then routes M3UA messages between its ASPs, typically by point-codes.

For an introduction about SCCP/M3UA/SS7/SIGTRAN technology, please see the chapter *Signaling Networks: SS7 and SIGTRAN* in the OsmoSTP user manual.

In an all-Osmocom CNI, the typical simple/minimal usage is:

- OsmoSTP acts as the STP/SG (server role) and routes between the ASP,
- All other Osmocom CNI programs act as SCTP client and provide ASP implementations.

For example, in an all-Osmocom minimal setup,

- OsmoMSC contacts an OsmoSTP and subscribes its point-code 0.23.1;
- then OsmoBSC also contacts the same OsmoSTP, subscribes with its own point-code 1.23.3.
- Using these established links, OsmoBSC initiates an A-interface link by directing a BSSAP RESET message to the MSC's point-code 0.23.1,
- and the RESET ACK response from the MSC is routed back to the BSC's point-code 1.23.3.

The details of SCCP/M3UA are configured in the *cs7* section of the VTY configuration.

Osmocom programs automatically configure missing SCCP/M3UA configuration, by assuming sane defaults for small/minimal all-Osmocom installations, which may not be what you want in larger networks integrating with non-Osmocom core network elements.

If no explicit `routing-key` is set, it may be determined at runtime by negotiation with OsmoSTP — see OsmoSTP manual chapter "Osmocom M3UA Routing Key Management Extensions", regarding config option `accept-asp-connections dynamic-permitted`.

The complete active configuration of an Osmocom program can be obtained by the VTY command `show cs7 config` (the usual `show running-config` omits automatically configured items). Here is an example of OsmoMSC's default configuration:

```
OsmoMSC> show cs7 config
cs7 instance 0
 point-code 0.23.1
 asp asp-clnt-OsmoMSC-A-Iu 2905 0 m3ua
  remote-ip 127.0.0.1
  role asp
  sctp-role client
 as as-clnt-OsmoMSC-A-Iu m3ua
  asp asp-clnt-OsmoMSC-A-Iu
  routing-key 2 0.23.1
```

At the time of writing, SCCP/M3UA links involving Osmocom program are:

- A-interface: OsmoBSC to OsmoMSC
- IuCS-interface: OsmoHNBGW to OsmoMSC
- IuPS-interface: OsmoHNBGW to OsmoSGSN
- Lb-interface: OsmoSMLC to OsmoBSC

On the SCTP/IP level, those connections are actually all established from the respective program (BSC, MSC, HNBGW, SGSN, SMLC) to OsmoSTP. Hence, if you look at the traffic in a protocol analyzer like wireshark, at IP level, you will see each of those programs establishing an SCTP association from a random local IP to the well-known SCTP port for M3UA (2905) at the OsmoSTP.

Those star-connections for M3UA/SCTP then are the transport network for higher level protocols like SCCP. OsmoSTP then acts as central router for SCCP-level message exchange between all the connected programs.

10.1 Connect to STP Instance

Establishing an SCCP/M3UA link towards a remote STP instance can be configured as:

```
cs7 instance 0
 asp my-asp 2905 0 m3ua
  # IP address of the remote STP:
  remote-ip 10.23.24.1
  # optional: local bind to a specific IP
  local-ip 10.9.8.7
  role asp
  sctp-role client
```

Be aware that such an `asp` needs to be linked to an `as`, see Section 10.5.

By default, an STP instance is assumed to listen on the default M3UA port (2905) on the local host. That means in general `127.0.0.1` will be used as default remote SCTP address, and `::1` will be added to the SCTP association if IPv6 support is available on the system.

Note
OsmoSTP listens by default on `::` if IPv6 is enabled on the system, and on `0.0.0.0` otherwise. Address `::` actually superseeds `0.0.0.0`, meaning it will listen on all IPv4 and IPv6 addresses available on the system.

Caution
Some applications overwrite the default target remote address to be `localhost`. If IPv6 support is available on the system, `localhost` will usually resolve to `::1`, otherwise it will usually resolve to `127.0.0.1`.

10.2 Local Point-Code

Each CNI program on an SCCP/M3UA link typically has a local point-code, configurable by:

```
cs7 instance 0
 point-code 7.65.4
```

If an explicit routing context is configured, this point-code is repeated in the `routing-key` configuration:

```
cs7 instance 0
 point-code 0.23.1
 as my-as m3ua
  routing-key 2 0.23.1
```

See also Section 10.4.

10.3 Remote Point-Code

Programs establishing communication across SCCP links need a remote SCCP address, typically by point-code, to contact.

For example,

- OsmoBSC needs to know the MSC's point-code, to be able to establish the A-interface.
- OsmoHNBGW needs to know the MSC's point-code, to be able to establish the IuCS-interface.
- OsmoHNBGW needs to know the SGSN's point-code, to be able to establish the IuPS-interface.

To maintain remote SCCP addresses, each `cs7` instance maintains an SCCP address book:

```
cs7 instance 0
 sccp-address remote-pc-example
  point-code 1.23.1
```

This address book entry on its own has no effect. It is typically referenced by specific configuration items depending on the individual programs.

Examples:

- An OsmoBSC configures the MSC's remote SCCP address:

```
cs7 instance 0
 sccp-address my-remote-msc
  point-code 1.23.1
msc 0
 msc-addr my-remote-msc
```

- An HNBGW configures both the remote MSC's and SGSN's SCCP addresses:

```
cs7 instance 0
 sccp-address my-msc
  point-code 0.23.1
 sccp-address my-sgsn
  point-code 0.23.2
hnbgw
 iucs
  remote-addr my-msc
 iups
  remote-addr my-sgsn
```

Besides a point-code, an SCCP address can have several routing indicators:

- PC: routing by point-code is the default for Osmocom.
- GT: routing by Global Title is configurable by `routing-indicator GT`.
- IP: routing by IP address is configurable by `routing-indicator IP`.

In OsmoSTP, only routing by point-code is currently implemented.

10.4 Point-Code Format

Point-codes can be represented in various formats. For details, see OsmoSTP manual, chapter "Point Codes".

By default, Osmocom uses a point-code representation of 3.8.3, i.e. first digit of 3 bit, second digit of 8 bit, and third digit of 3 bit.

```
cs7 instance 0
 point-code format 3 8 3
 point-code 0.23.1
```

Often, point-codes are also represented as a single decimal number:

```
cs7 instance 0
 point-code format 24
 point-code 185
```

It is also possible to use a dash as delimiter.

```
cs7 instance 0
 point-code delimiter dash
 point-code 0-23-1
```

10.5 AS and ASP

Each CNI program needs at least one Application Server `as` and one Application Server Process `asp` configured on its `cs7` to be able to communicate on SCCP/M3UA. An `asp` needs to be part of at least one `as`. For details, see the OsmoSTP manual, chapters "Application Server" and "Application Server Process".

In Osmocom's `cs7`, any amount of `as` and `asp` can be configured by name, and an `as` references the `asp` entries belonging to it by their names.

In a simple/minimal Osmocom setup, an Osmocom CNI program would have exactly one `as` with one `asp`.

For example:

```
cs7 instance 0
 asp my-asp 2905 0 m3ua
  # where to reach the STP:
  remote-ip 127.0.0.1
  role asp
  sctp-role client
 as my-as m3ua
  asp my-asp
```

In Osmocom CNI programs, it is possible to omit the `as` and/or `asp` entries, which the program will then attempt to configure automatically.

When configuring both `as` and `asp` manually, make sure to link them by name. For example, the following configuration will **fail**, because `as` and `asp` are not linked:

```
cs7 instance 0
 asp my-asp 2905 0 m3ua
  remote-ip 127.0.0.1
  role asp
  sctp-role client
 as my-as m3ua
  routing-key 2 0.23.1
```

To **fix** above config, link the `asp` to an `as` by adding `asp my-asp`:

```
cs7 instance 0
 asp my-asp 2905 0 m3ua
  remote-ip 127.0.0.1
  role asp
  sctp-role client
 as my-as m3ua
  asp my-asp
  routing-key 2 0.23.1
```

10.6 Subsystem Number (SSN)

Osmocom CNI programs typically route SCCP/M3UA messages by PC+SSN: each ASP, having a given SCCP address, receives messages for one or more specific subsystems, identified by a Subsystem Number (SSN).

For example, the A-interface between BSC and MSC uses SSN = BSSAP (254). In Osmocom programs, SSNs do not need to be configured; they implicitly, naturally relate to the interfaces that a program implements.

For example, OsmoBSC takes the configured remote MSC's SCCP address and adds the SSN = BSSAP to it in order to contact the MSC's A-interface. To receive A-interface messages from the MSC, OsmoBSC subscribes a local user for this SSN on the ASP.

10.7 Routing Context / Routing Key

In SCCP/M3UA, messages can be routed by various Routing Indicators (PC+SSN, PC, GT, ...). Osmocom CNI programs typically use PC+SSN as Routing Indicator.

On the SG (for example OsmoSTP), each ASP's distinct Routing Indicator needs to be indexed by a distinct Routing Context (a simple index number scoped per SG), to forward M3UA to the correct peer.

The Osmocom SG implementation employs Routing Key Management (RKM, see OsmoSTP manual) to automatically determine a distinct Routing Context index for each connected ASP. Routing Contexts can also be configured manually — some non-Osmocom SG implementations require this.

Each Routing Context is associated with a Routing Indicator and address; this association is called a Routing Key.

For example, to configure an OsmoBSC with a local point-code of 1.23.3 to receive M3UA with Routing Context of 2 and RI=PC:

```
cs7 instance 0
 point-code 1.23.3
 as my-as m3ua
  routing-key 2 1.23.3
```

Osmocom programs so far implement Routing Keys by Destination Point Code (DPC), plus optional Subsystem Number (SSN) and/or Service Indicator (SI):

```
routing-key RCONTEXT DPC
routing-key RCONTEXT DPC si (aal2|bicc|b-isup|h248|isup|sat-isup|sccp|tup)
routing-key RCONTEXT DPC ssn SSN
routing-key RCONTEXT DPC si (aal2|bicc|b-isup|h248|isup|sat-isup|sccp|tup) ssn SSN
```

10.7.1 M3UA without Routing Context IE / Routing Context *0*

As per the M3UA specification, the use of the routing context IE is optional as long as there is only one AS within an ASP. As soon as there are multiple different AS within one ASP, the routing context IE is mandatory, as it is the only clue to differentiate which of the ASs a given message belongs to.

In the Osmocom M3UA implementation, it is generally assumed that a routing context IE is always used, for the sake of clarity.

However, the routing context ID of *0* has the special meaning of *do not encode a routing context IE on transmit*.

So if you configure an application like OsmoBSC to use routing context 0, then no routing context IE will be included in outbound M3UA messages.

This special interpretation of *0* within the Osmocom M3UA implementation however means that we can not represent M3UA with a routing context IE that actually contains *0* as a numeric identifier.

So you only have the following options: * Using M3UA with routing context (1..N) * Using M3UA without routing context (0)

10.7.2 Example: Static Routing

Osmocom SS7 supports dynamic routing key registration via M3UA Routing Key Management (RKM), allowing minimal SS7 configuration. If all of your components support dynamic RKM, you should probably use it: see `accept-asp-connections dynamic-permitted` in osmo-stp.cfg.

This chapter explains how to configure osmo-stp if dynamic RKM is not an option.

In this example, let's connect osmo-bsc via osmo-stp to osmo-msc using only static SS7 routing.

```
 BSC        <--RK-1-->  STP         <--RK-3--> MSC
 IP 1.1.1.1             IP 2.2.2.2             IP 3.3.3.3
 M3UA 2905              M3UA 2905              M3UA 2905
 PC 1.1.1                                      PC 3.3.3
```

Every one static route fanning out from STP gets assigned a distinct Routing Key — a simple integer number. Above, the BSC's link has RK 1, the MSC's link has RK 3.

For static routing, the M3UA port numbers must be fixed, i.e. there must be no `0` for a client's local port as in `asp foo 2905 0 m3ua`. Instead, you may use `asp foo 2905 2905 m3ua`.

The BSC needs to configure:

- its own point-code — has to match the PC configured for the BSC in osmo-stp.cfg
- the routing key — has to match the RK assigned to BSC's PC in osmo-stp.cfg
- the MSC's point-code — has to match the PC in osmo-stp.cfg and osmo-msc.cfg
- local and remote IP:ports for M3UA — have to match the IP:ports in osmo-stp.cfg

The MSC needs to configure:

- its own point-code — has to match the PC configured for the MSC in osmo-stp.cfg
- the routing key — has to match the RK assigned to MSC's PC in osmo-stp.cfg
- local and remote IP:ports for M3UA — have to match the IP:ports in osmo-stp.cfg

The STP needs to configure:

- all point-codes — they have to match the PCs in osmo-bsc.cfg and osmo-msc.cfg
- all routing keys — they have to match the RKs used in osmo-bsc.cfg and osmo-msc.cfg
- local and remote IP:ports for M3UA — have to match the IP:ports in osmo-bsc.cfg and osmo-msc.cfg

osmo-bsc.cfg

```
cs7 instance 0
 point-code 1.1.1

 asp mybsc-0 2905 2905 m3ua
  remote-ip 2.2.2.2
  local-ip 1.1.1.1
  sctp-role client
 as mybsc0 m3ua
  asp mybsc0-0
  routing-key 1 1.1.1

 sccp-address mymsc
  routing-indicator PC
  point-code 3.3.3

msc 0
 msc-addr mymsc
```

osmo-stp.cfg

```
cs7 instance 0
 xua rkm routing-key-allocation static-only
 listen m3ua 2905
  accept-asp-connections pre-configured
  local-ip 2.2.2.2

 # asp <name> <remote-port> <local-port|0> m3ua
 asp mybsc-0 2905 2905 m3ua
  remote-ip 1.1.1.1
```

```
  local-ip 2.2.2.2
 as mybsc m3ua
  asp bsc-0
  routing-key 1 1.1.1

 asp mymsc-0 2905 2905 m3ua
  remote-ip 3.3.3.3
  local-ip 2.2.2.2
 as mymsc m3ua
  asp mymsc-0
  routing-key 3 3.3.3

 route-table system
  update route 1.1.1 7.255.7 linkset mybsc
  update route 3.3.3 7.255.7 linkset mymsc
```

osmo-msc.cfg

```
cs7 instance 0
 point-code 3.3.3

 asp mymsc-0 2905 2905 m3ua
  remote-ip 2.2.2.2
  local-ip 3.3.3.3
  sctp-role client
 as mymsc0 m3ua
  asp mymsc0-0
  routing-key 3 3.3.3
```

For comparison, the same setup with dynamic routing key management is a lot shorter, especially at osmo-stp.cfg, and there is no need to manually configure point-codes and routing keys between STP and {BSC,MSC}:

osmo-bsc.cfg

```
cs7 instance 0
 point-code 1.1.1

 asp mybsc-0 2905 0 m3ua
  remote-ip 2.2.2.2
  local-ip 1.1.1.1
  sctp-role client
 as mybsc0 m3ua
  asp mybsc0-0

 sccp-address mymsc
  routing-indicator PC
  point-code 3.3.3

msc 0
 msc-addr mymsc
```

osmo-stp.cfg

```
cs7 instance 0
 xua rkm routing-key-allocation dynamic-permitted
 listen m3ua 2905
  accept-asp-connections dynamic-permitted
  local-ip 2.2.2.2
```

osmo-msc.cfg

```
cs7 instance 0
 point-code 3.3.3

 asp mymsc-0 2905 0 m3ua
  remote-ip 2.2.2.2
  local-ip 3.3.3.3
  sctp-role client
 as mymsc0 m3ua
  asp mymsc0-0
```

11 Reviewing and Provisioning BTS configuration

The main functionality of the BSC component is to manage BTSs. As such, provisioning BTSs within the BSC is one of the most common tasks during BSC operation. Just like about anything else in OsmoBSC, they are configured using the VTY.

BTSs are internally numbered with integer numbers starting from "0" for the first BTS. BTS numbers have to be contiguous, so you cannot configure 0,1,2 and then 5.

11.1 Reviewing current BTS status and configuration

In order to view the status and properties of a BTS, you can issue the `show bts` command. If used without any BTS number, it will display information about all provisioned BTS numbers.

```
OsmoBSC> show bts 0
BTS 0 is of nanobts type in band DCS1800, has CI 0 LAC 1, BSIC 63, TSC 7 and 1 TRX
Description: (null)
MS Max power: 15 dBm
Minimum Rx Level for Access: -110 dBm
Cell Reselection Hysteresis: 4 dBm
RACH TX-Integer: 9
RACH Max transmissions: 7
System Information present: 0x0000007e, static: 0x00000000
  Unit ID: 200/0/0, OML Stream ID 0xff
  NM State: Oper 'Enabled', Admin 2, Avail 'OK'
  Site Mgr NM State: Oper 'Enabled', Admin 0, Avail 'OK'
  Paging: 0 pending requests, 0 free slots
  OML Link state: connected.
  Current Channel Load:
                   TCH/F:   0% (0/5)
                  SDCCH8:   0% (0/8)
```

You can also review the status of the TRXs configured within the BTSs of this BSC by using `show trx`:

```
OsmoBSC> show trx 0 0
TRX 0 of BTS 0 is on ARFCN 871
Description: (null)
  RF Nominal Power: 23 dBm, reduced by 0 dB, resulting BS power: 23 dBm
  NM State: Oper 'Enabled', Admin 2, Avail 'OK'
  Baseband Transceiver NM State: Oper 'Enabled', Admin 2, Avail 'OK'
  IPA Abis/IP stream ID: 0x00
```

The output can be restricted to the TRXs of one specified BTS number (`show trx 0`) or even that of a single specified TRX within a specified BTS (`show trx 0 0`).

Furthermore, information on the individual timeslots can be shown by means of `show timeslot`. The output can be restricted to the timeslots of a single BTS (`show timeslot 0`) or that of a single TRX (`show timeslot 0 0`). Finally, you can restrict the output to a single timeslot by specifying the BTS, TRX and TS numbers (`show timeslot 0 0 4`).

```
OsmoBSC> show timeslot 0 0 0
BTS 0, TRX 0, Timeslot 0, phys cfg CCCH, TSC 7
  NM State: Oper 'Enabled', Admin 2, Avail 'OK'
OsmoBSC> show timeslot 0 0 1
BTS 0, TRX 0, Timeslot 1, phys cfg SDCCH8, TSC 7
  NM State: Oper 'Enabled', Admin 2, Avail 'OK'
```

11.2 Provisioning a new BTS

In order to provision BTSs, you have to enter the BTS config node of the VTY. In order to configure BTS 0, you can issue the following sequence of commands:

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# network
OsmoBSC(config-net)# bts 0
OsmoBSC(config-net-bts)#
```

At this point, you have a plethora of commands, in fact an entire hierarchy of commands to configure all aspects of the BTS, as well as each of its TRX and each timeslot within each TRX. For a full reference, please consult the telnet VTY integrated help or the respective chapter in the VTY reference.

BTS configuration depends quite a bit on the specific BTS vendor and model. The section below provides just one possible example for the case of a sysmoBTS.

Note that from the `configure terminal` command onwards, the telnet VTY commands above are identical to configuration file settings, for details see Section 6.

Starting with `network` as above, your complete sysmoBTS configuration may look like this:

```
network
 bts 0
  type osmo-bts
  band DCS1800
  description The new BTS in Baikonur
  location_area_code 0x0926
  cell_identity 5
  base_station_id_code 63
  ip.access unit_id 8888 0
  ms max power 40
  trx 0
   arfcn 871
   nominal power 23
   max_power_red 0
   timeslot 0
    phys_chan_config CCCH+SDCCH4
   timeslot 1
    phys_chan_config TCH/F
   timeslot 2
    phys_chan_config TCH/F
   timeslot 3
    phys_chan_config TCH/F
   timeslot 4
    phys_chan_config TCH/F
   timeslot 5
    phys_chan_config TCH/F
   timeslot 6
    phys_chan_config TCH/F
   timeslot 7
    phys_chan_config PDCH
```

11.3 System Information configuration

A GSM BTS periodically transmits a series of *SYSTEM INFORMATION* messages to mobile stations, both via the BCCH in idle mode, was well as via the SACCH in dedicated mode. There are many different types of such messages. For their detailed contents and encoding, please see *3GPP TS 24.008* [3gpp-ts-24-008].

For each of the *SYSTEM INFORMATION* message types, you can configure to have the BSC generate it automatically (*computed*), or you can specify the respective binary message as a string of hexadecimal digits.

The default configuration is to compute all (required) *SYSTEM INFORMATION* messages automatically.

Please see the *OsmoBSC VTY Reference Manual* [vty-ref-osmobsc] for further information, particularly on the following commands:

- `system-information (1|2|3|4|5|6|7|8|9|10|13|16|17|18|19|20|2bis|2ter|2quater|5bis|5ter) mode (static|computed)`
- `system-information (1|2|3|4|5|6|7|8|9|10|13|16|17|18|19|20|2bis|2ter|2quater|5bis|5ter) static HEXSTRING`

11.4 Neighbor List configuration

Every BTS sends a list of ARFCNs of neighbor cells . within its *SYSTEM INFORMATION 2* (and 2bis/2ter) messages on the BCCH . within its *SYSTEM INFORMATION 5* messages on SACCH in dedicated mode

For every BTS config node in the VTY, you can specify the behavior of the neighbor list using the `neighbor list mode` VTY command:

automatic
: Automatically generate a list of neighbor cells using all other BTSs configured in the VTY

manual
: Manually specify the neighbor list by means of `neighbor-list (add|del) arfcn <0-1023>` commands, having identical neighbor lists on BCCH (SI2) and SACCH (SI5)

manual-si5
: Manually specify the neighbor list by means of `neighbor-list (add|del) arfcn <0-1023>` for BCCH (SI2) and a separate neighbor list by means of `si5 neighbor-list (add|del) arfcn <0-1023>` for SACCH (SI5).

11.5 Configuring GPRS PCU parameters of a BTS

In the case of BTS models using Abis/IP (IPA), the GPRS PCU is located inside the BTS. The BTS then establishes a Gb connection to the SGSN.

All the BTS-internal PCU configuration is performed via A-bis OML by means of configuring the *CELL*, *NSVC* (NS Virtual Connection and *NSE* (NS Entity).

There is one *CELL* node and one *NSE* node, but there are two *NSVC* nodes. At the time of this writing, only the NSVC 0 is supported by OsmoBTS, while both NSVC are supported by the ip.access nanoBTS.

The respective VTY configuration parameters are described below. They all exist beneath each BTS VTY config node.

But let's first start with a small example

Example configuration of GPRS PCU parameters at VTY BTS node

```
OsmoBSC(config-net-bts)# gprs mode gprs
OsmoBSC(config-net-bts)# gprs routing area 1
OsmoBSC(config-net-bts)# gprs cell bvci 1234
OsmoBSC(config-net-bts)# gprs nsei 1234
OsmoBSC(config-net-bts)# gprs nsvc 0 nsvci 1234
OsmoBSC(config-net-bts)# gprs nsvc 0 local udp port 23000
OsmoBSC(config-net-bts)# gprs nsvc 0 remote udp port 23000
OsmoBSC(config-net-bts)# gprs nsvc 0 remote ip 192.168.100.239
```

11.6 More explanation about the PCU config parameters

11.6.1 `gprs mode (none|gprs|egprs)`

This command determines if GPRS (or EGPRS) services are to be enabled in this cell at all.

11.6.2 `gprs cell bvci <2-65535>`

Configures the *BSSGP Virtual Circuit Identifier*. It must be unique between all BSSGP connections to one SGSN.

Note
It is up to the system administrator to ensure all PCUs are allocated an unique bvci. OsmoBSC will not ensure this policy.

11.6.3 `gprs nsei <0-65535>`

Configures the *NS Entity Identifier*. It must be unique between all NS connections to one SGSN.

Note
It is up to the system administrator to ensure all PCUs are allocated an unique bvci. OsmoBSC will not ensure this policy.

11.6.4 `gprs nsvc <0-1> nsvci <0-65535>`

Configures the *NS Virtual Connection Identifier*. It must be unique between all NS virtual connections to one SGSN.

Note
It is up to the system administrator to ensure all PCUs are allocated an unique nsvci. OsmoBSC will not ensure this policy.

11.6.5 `gprs nsvc <0-1> local udp port <0-65535>`

Configures the local (PCU side) UDP port for the NS-over-UDP link.

11.6.6 `gprs nsvc <0-1> remote udp port <0-65535>`

Configures the remote (SGSN side) UDP port for the NS-over-UDP link.

11.6.7 `gprs nsvc <0-1> remote ip A.B.C.D`

Configures the remote (SGSN side) UDP port for the NS-over-UDP link.

11.6.8 `gprs ns timer (tns-block|tns-block-retries|tns-reset|tns-reset-retries|tns-test|tns-a <0-255>`

Configures the various GPRS NS related timers. Please check the GPRS NS specification for the detailed meaning of those timers.

11.7 Dynamic Timeslot Configuration (TCH / PDCH)

A dynamic timeslot is in principle a timeslot that is used to serve GPRS data (PDCH), but that can be switched to be used either for voice (TCH) or signalling (SDCCH8) when all other static timeslots are already in use. This enhances GPRS bandwidth while there is no CS load, and is dynamically scaled down as CS services need to be served. This is a tremendous improvement in service over statically assigning a fixed number of timeslots for voice and data.

The causality is as follows: to establish a voice call, the MSC requests a logical channel of a given TCH kind from the BSC. The BSC assigns such a channel from a BTS' TRX's timeslot of its choice. The knowledge that a given timeslot is dynamic exists only on the BSC level. When the MSC asks for a logical channel, the BSC may switch off PDCH on a dynamic timeslot and then assign a logical TCH channel on it. Hence, though compatibility with the BTS needs to be ensured, any MSC is compatible with dynamic timeslots by definition.

OsmoBSC supports two kinds of dynamic timeslot handling, configured via the `network/bts/trx/timeslot/phys_chan_co` configuration. Not all BTS models support dynamic channels.

Table 11: Dynamic timeslot support by various BTS models

	`DYNAMIC/OSMOCOM`	`DYNAMIC/IPACCESS`
ip.access nanoBTS	-	supported
Ericsson RBS	supported	-
sysmoBTS using *osmo-bts-sysmo*	supported	supported
various SDR platforms using *osmo-bts-trx*	supported	supported
Nutaq Litecell 1.5 using *osmo-bts-litecell15*	supported	supported
Octasic OctBTS using *osmo-bts-octphy*	supported	supported

The *OsmoBTS Abis Protocol Specification* [osmobts-abis-spec] describes the non-standard RSL messages used for these timeslot kinds.

Note
Same as for dedicated PDCH timeslots, you need to enable GPRS and operate a PCU, SGSN and GGSN to provide the actual data service.

11.7.1 Osmocom Style Dynamic Timeslots (DYNAMIC/OSMOCOM)

`DYNAMIC/OSMOCOM` is an alias for `TCH/F_TCH/H_SDCCH8_PDCH`.

Timeslots of the `DYNAMIC/OSMOCOM` type dynamically switch between TCH/F, TCH/H, SDCCH8 and PDCH, depending on the channel kind requested by the MSC. The RSL messaging for these timeslots is compatible with Ericsson RBS.

BTS models supporting this timeslot kind are shown in Table 11.

In the lack of transcoding capabilities, this timeslot type may cause mismatching codecs to be selected for two parties of the same call, which would cause call routing to fail (`"Cannot patch through call with different channel types: local = TCH_F, remote = TCH_H"`). A workaround is to disable TCH/F on this timeslot type, i.e. to allow only TCH/H. To disable TCH/F on Osmocom style dynamic timeslots, use a configuration of

```
network
 dyn_ts_allow_tch_f 0
```

In OsmoNITB, disabling TCH/F on Osmocom dynamic timeslots is the default. In OsmoBSC, the default is to allow both.

11.7.2 ip.access Style Dynamic Timeslots (DYNAMIC/IPACCESS)

`DYNAMIC/IPACCESS` is an alias for `TCH/F_PDCH`.

Timeslots of the `DYNAMIC/IPACCESS` type dynamically switch between TCH/F and PDCH. The RSL messaging for `DYNAMIC/IPAC` timeslots is compatible with ip.access nanoBTS.

BTS models supporting this timeslot kind are shown in Table 11.

11.7.3 Avoid PDCH Exhaustion

To avoid disrupting GPRS, configure at least one timeslot as dedicated PDCH. With only dynamic timeslots, a given number of voice calls would convert all timeslots to TCH, and no PDCH timeslots would be left for GPRS service.

11.7.4 Dynamic Timeslot Configuration Examples

This is an extract of an `osmo-bsc` config file. A timeslot configuration with five Osmocom style dynamic timeslots and one dedicated PDCH may look like this:

```
network
 bts 0
  trx 0
   timeslot 0
    phys_chan_config CCCH+SDCCH4
   timeslot 1
    phys_chan_config SDCCH8
   timeslot 2
    phys_chan_config DYNAMIC/OSMOCOM
   timeslot 3
    phys_chan_config DYNAMIC/OSMOCOM
   timeslot 4
    phys_chan_config DYNAMIC/OSMOCOM
   timeslot 5
    phys_chan_config DYNAMIC/OSMOCOM
   timeslot 6
    phys_chan_config DYNAMIC/OSMOCOM
   timeslot 7
    phys_chan_config PDCH
```

With the ip.access nanoBTS, only `DYNAMIC/IPACCESS` dynamic timeslots are supported, and hence a nanoBTS configuration may look like this:

```
network
 bts 0
  trx 0
   timeslot 0
    phys_chan_config CCCH+SDCCH4
   timeslot 1
    phys_chan_config SDCCH8
   timeslot 2
    phys_chan_config DYNAMIC/IPACCESS
   timeslot 3
    phys_chan_config DYNAMIC/IPACCESS
   timeslot 4
    phys_chan_config DYNAMIC/IPACCESS
   timeslot 5
    phys_chan_config DYNAMIC/IPACCESS
   timeslot 6
    phys_chan_config DYNAMIC/IPACCESS
   timeslot 7
    phys_chan_config PDCH
```

11.8 Tuning Access to the BTS

OsmoBSC offers several configuration options to fine-tune access to the BTS. It can allow only a portion of the subscribers access to the network. This can also be used to ramp up access to the network on startup by slowly letting in more and more subscribers. This is especially useful for isolated cells with a huge number of subscribers.

Other options control the behaviour of the MS when it needs to access the random access channel before a dedicated channel is established.

If the BTS is connected to the BSC via a high-latency connection the MS should wait longer for an answer to a RACH request. If it does not the network will have to deal with an increased load due to duplicate RACH requests. However, in order to minimize the delay when a RACH request or response gets lost the MS should not wait too long before retransmitting.

11.8.1 Access Control Class Load Management

Every SIM card is member of one of the ten regular ACCs (0-9). Access to the BTS can be restricted to SIMs that are members of certain ACCs.

Furthermore, high priority users (such as PLMN staff, public or emergency services, etc.) may be members of one or more ACCs from 11-15.

Since the ACCs 0-9 are distributed uniformly across all SIMs, for instance allowing only ACCs 0-4 to connect to the BTS should reduce its load by 50% at the expense of not serving 50% of the subscribers.

The default is to allow all ACCs to connect.

OsmoBSC supports several levels of ACC management to allow or restrict access either permanently or temporarily on each BTS.

The first level of management consists of an access list to flag specific ACCs as permanently barred (the list can be updated at any time through VTY as seen below). As indicated above, the default is to allow all ACCs (0-15).

Example: Restrict permanent access to the BTS by ACC

```
network
 bts 0
  rach access-control-class 1 barred ❶
  rach access-control-class 9 allowed ❷
```

❶ Disallow SIMs with access-class 1 from connecting to the BTS

❷ Permit SIMs with access-class 9 to connect to the BTS.

On really crowded areas, a BTS may struggle to service all mobile stations willing to use it, and which may end up in collapse. In this kind of scenarios it is a good idea to temporarily further restrict the amount of allowed ACCs (hence restrict the amount of subscribers allowed to reach the BTS). However, doing so on a permanent basis would be unfair to subscribers from barred ACCs. Hence, OsmoBSC can be configured to temporarily generate ACC subsets of the permanent set presented above, and rotate them over time to allow fair access to all subscribers. This feature is only aimed at ACCs 0-9, since ACCs 11-15 are considered high priority and hence are always configured based on permanent list policy.

Example: Configure rotative access to the BTS

```
network
 bts 0
  access-control-rotate 3 ❶
  access-control-rotate-quantum 20 ❷
```

❶ Only allow up to 3 concurrent allowed ACCs from the permanent list

❷ Rotate the generated permanent list subsets every 20 seconds in a fair fashion

Furthermore, cells with large number of subscribers and limited overlapping coverage may become overwhelmed with traffic after the cell starts broadcasting. This is especially true in areas with little to no reception from other networks. To manage the load, OsmoBSC has an option to further restrict the rotating ACC subset during startup and slowly increment it over time and taking current channel load into account. The channel load will always be checked, even after the start up procedure, at an interval specified by the `access-control-class-ramping-step-interval` VTY command. It will

either keep, increase or decrease the size of the current rotating ACC subset based on certain thresholds configured by the `access-control-class-ramping-chan-load` VTY command. As a result, if ACC ramping is enabled (`access-control-` the number of concurrent allowed ACCs will start with **1** and will fluctuate over time based on channel load in the interval **[1, `access-control-rotate`]**. This means at any time there will be at least **1** allowed ACC which, together with ACC rotation, will prevent from subscriber being unable to use the network.

Example: Ramp up access to the BTS after startup

```
network
 bts 0
  access-control-class-ramping ❶
  access-control-class-ramping-step-size 1 ❷
  access-control-class-ramping-step-interval 30 ❸
  access-control-class-ramping-chan-load 71 89 ❹
```

❶ Turn on access-control-class ramping

❷ At each step enable one more ACC

❸ Check whether to allow more/less ACCs every 30 seconds

❹ The rotate subset size is increased if channel load is < 71%, and decreased if channel load is > 89%.

Here a full example of all the mechanisms combined can be found:

Example: Full ACC Load Management config setup

```
bts 0
  rach access-control-class 5 barred ❶
  rach access-control-class 6 barred
  rach access-control-class 7 barred
  rach access-control-class 8 barred
  rach access-control-class 9 barred
  access-control-class-rotate 3 ❷
  access-control-class-rotate-quantum 20 ❸
  access-control-class-ramping ❹
  access-control-class-ramping-step-size 1 ❺
  access-control-class-ramping-step-interval 30 ❻
  access-control-class-ramping-chan-load 71 89 ❼
```

❶ ACCs 5-9 are administratively barred, ie they will never be used until somebody manually enables them in VTY config

❷ Allow access through temporary subsets of len=3 from ACC set 0-4: (0,1,2) → (1,2,3) → (2,3,4) → (3,4,0), etc.

❸ Each subset iteration will happen every 20 seconds

❹ Ramping is enabled: during startup it will further restrict the rotate subset size parameter (start at len=1, end at len=3)

❺ The rotate subset size parameter will be increased or decreased one ACC slot at a time: len=1 → len=2 → len=3

❻ Check to further increase or decrease the rotate subset size based on current channel load is triggered every 30 seconds

❼ The rotate subset size is increased if channel load is < 71%, and decreased if channel load is > 89%.

11.9 Configuring FACCH/SACCH repetition

osmo-bts supports repetition of FACCH, uplink SACCH and downlink SACCH as described in *3GPP TS 44.006* [?]. When the feature is enabled it is applied dynamically, depending on the rf signal quality and MS capabilities. FACCH/SACCH repetition (or ACCH repetition) repeats the channel block transmission two times. This allows the transceiver to combine the symbols from two separate transmissions, which increases the probability that even a weak signal can be decoded.

Enabling ACCH repetition is especially recommended when using the AMR speech codec. AMR already provides a forward error correction that is superior to the forward error correction used with FACCH or SACCH. ACCH repetition is a good way to even out this imbalance.

The VTY configuration allows to enable repetition for all three channel types separately. For FACCH the operator has the option to restrict the repetition to LAPDM command frames only. Alternatively it is also possible to allow all LAPDM frame types for repetition. The following example shows a typical configuration where ACCH repetition is fully enabled.

Example typical configuration of ACCH repetition parameters at VTY BTS node

```
OsmoBSC(config-net-bts)# repeat dl-facch all
OsmoBSC(config-net-bts)# repeat ul-sacch
OsmoBSC(config-net-bts)# repeat dl-sacch
OsmoBSC(config-net-bts)# repeat rxqual 4
```

It should be noted that unless the repetition is enabled explicitly, the repetition is turned off by default. If no threshold (see Table 12) is set, the default value 4 (BER >= 1.6%) will be used. The following example shows a minimal configuration where the repetition is only activated for FACCH LAPDM command frames.

Example minimal configuration of ACCH repetition parameters at VTY BTS node

```
OsmoBSC(config-net-bts)# repeat dl-facch command
```

Since it is not worthwhile to apply any repetition when the signal conditions are good enough to ensure a reliable transmission in one round, the operator has the option to set a threshold based on RXQUAL/BER at which the repetition is switched on. The threshold mechanism implements a hysteresis to prevent bouncing between repetition on and repetition off. Only when the signal quality is increased again by two rxqual levels, the repetition is turned off again. It is even possible to permanently enable repetition, regardless of the signal quality.

Table 12: ACCH repetition thresholds

rxqual	enable threshold	disable threshold
0	(repetition always on)	(repetition always on)
1	asciimath:[BER >= 0.2%]	asciimath:[BER = 0%]
2	asciimath:[BER >= 0.4%]	asciimath:[BER = 0%]
3	asciimath:[BER >= 0.8%]	asciimath:[BER ⇐ 0.2%]
4	asciimath:[BER >= 1.6%]	asciimath:[BER ⇐ 0.4%]
5	asciimath:[BER >= 3.2%]	asciimath:[BER ⇐ 0.8%]
6	asciimath:[BER >= 6.4%]	asciimath:[BER ⇐ 1.6%]
7	asciimath:[BER >= 12.8%]	asciimath:[BER ⇐ 3.2%]

Note
osmo-bsc only sets the ACCH repetition parameters via RSL. Whether ACCH repetition can be used depends on the BTS model and osmo-bts version. To find out if a BTS supports ACCH repetition (BTS_FEAT_ACCH_REP), the VTY command `show bts` can be used.

11.9.1 RACH Parameter Configuration

The following parameters allow control over how the MS can access the random access channel (RACH). It is possible to set a minimum receive level under which the MS will not even attempt to access the network.

The RACH is a shared channel which means multiple MS can choose to send a request at the same time. To minimize the risk of a collision each MS will choose a random number of RACH slots to wait before trying to send a RACH request.

On very busy networks the range this number is chosen from should be high to avoid collisions, but a lower range reduces the overall delay when trying to establish a channel.

The option `rach tx integer N` controls the range from which this number X is chosen. It is `0 <= X < max(8,N)`.

After sending a RACH request the MS will wait a random amount of slots before retransmitting its RACH request. The range it will wait is also determined by the option `rach tx integer N`, but calculating it is not so straightforward. It is defined as `S <= X < S+N` where `S` is determined from a table.

In particular `S` is lowest when `N` is one of 3, 8, 14 or 50 and highest when `N` is 7, 12 or 32.

For more information see *3GPP TA 44.018* [3gpp-ts-44-018] Ch. 3.3.1.1.2 and Table 3.3.1.1.2.1 in particular.

The amount of times the MS attempts to retransmit RACH requests can also be changed. A higher number means more load on the RACH while a lower number can cause channel establishment to fail due to collisions or bad reception.

Example: Configure RACH Access Parameters

```
network
 bts 0
  rxlev access min 20 ❶
  rach tx integer 50❷
  rach max transmission ❸
```

❶ Allow access to the network if the MS receives the BCCH of the cell at -90dBm or better (20dB above -110dBm).

❷ This number affects how long the MS waits before (re-)transmitting RACH requests.

❸ How often to retransmit the RACH request.

11.10 Configuring Ericsson RBS Interface Switch (IS)

Ericsson RBS2000/RBS6000 base stations feature a so called "Interface Switch" (IS), which is a built-in switchboard that interconnects between internal components of the BTS. It also connects to the external E1 connections. This allows to adapt the BTS to specific E1 networking requirements that may differ from the usual timeslot configuration.

The internals of an Ericsson RBS are quite complex. In the following we will only explain how to connect transceiver units (TRU) to an E1 interface pointing to the outside world.

11.10.1 Understanding the is-connection-list VTY option

The IS operates on 16kbps subslots (ICPs), which means that there are no fixed borders between E1 timeslots. Any number of consecutive subslots may be connected through. However, depending on the components that are connected it may still be a requirement to align on E1 timeslot borders.

The configuration of the IS is done using the is-connection-list command. The first two numbers are the ICP numbers that specify the first subslot on both sides that shall be interconnected. The third number (contiguity index) specifies how many of the following subslots shall be connected.

In the following example we connect 4 blocks with 12 subslot each. The numbers on the left are the ICP numbers of the E1 connection pointing to the outside. The numbers in the middle are the ICP numbers of the subslots occupied by the transceivers (one TRX per block). The third number is the contiguity index that spans over 12 subslots or 3 E1 timeslots.

Example: 4 TRX BTS (4 x 12 subslots)

```
network
 bts 0
  is-connection-list add 4 512 12
  is-connection-list add 16 524 12
  is-connection-list add 28 536 12
  is-connection-list add 40 548 12
```

11.10.2 E1 port and TRU ICP numbers

On the outside connection, the ICP counting begins at E1 timeslot 0 (port A) but since E1 TS 0 is reserved for framing and synchronization of the E1 line itself the first usable subslot is subslot 4 (beginning of E1 TS 1). Depending on the configuration the BTS may have multiple E1 ports. The counting scheme will repeat itself. This means the next usable ICP can be found at an offset of 128.

Table 13: External connections of a BTS with two E1 ports

Function	Subslot offset (ICP)	ICP count
E1 port A	4	124
E1 port B	132	124

Depending on the transceiver configuration, a RBS2000/RBS6000 base station usually features two sets of ICPs for each TRX. The reason for this is that with the introduction of EGPRS more bandwidth than a single 16kbps subslot could deliver was required. The solution to this was to add an entirely new set of IS ICPs where full 64kbps E1 timeslots instead of 16kbps subslots could be used to serve a single air interface timeslot. The two sets of ICPs must not be mixed. Only one set may be used at a time.

Table 14: ICPs to use TRU with 16kbps subslots per TRAU

Function	Subslot offset (ICP)	ICP count
TRU-0, RSL/OML	512	4
TRU-0, TRAU TS0..TS7	516	8
TRU-1, RSL/OML	524	4
TRU-1, TRAU TS0..TS7	528	8
TRU-2, RSL/OML	536	4
TRU-2, TRAU TS0..TS7	540	8
TRU-3, RSL/OML	548	4
TRU-3, TRAU TS0..TS7	552	8
TRU-4, RSL/OML	560	4
TRU-4, TRAU TS0..TS7	564	8
TRU-5, RSL/OML	572	4
TRU-5, TRAU TS0..TS7	576	8
TRU-6, RSL/OML	640	4
TRU-6, TRAU TS0..TS7	644	8
TRU-7, RSL/OML	652	4
TRU-7, TRAU TS0..TS7	656	8
TRU-8, RSL/OML	664	4
TRU-8, TRAU TS0..TS7	668	8
TRU-9, RSL/OML	676	4
TRU-9, TRAU TS0..TS7	680	8
TRU-10, RSL/OML	688	4
TRU-10, TRAU TS0..TS7	692	8
TRU-11, RSL/OML	700	4
TRU-11, TRAU TS0..TS7	704	8

Note
Each air interface timeslot is served by its individual TRAU, so it is possible to route each subslot (ICP) dedicated to TRAU individually. The connections on the other end may contain gaps and do not have to be consecutive.

Table 15: ICPs to use TRU with 64kbps subslots per TRAU

Function	Subslot offset (ICP)	ICP count
TRU-0, RSL/OML	712	4
TRU-0, TRAU TS0..TS7	716	32
TRU-1, RSL/OML	748	4
TRU-1, TRAU TS0..TS7	752	32
TRU-2, RSL/OML	784	4
TRU-2, TRAU TS0..TS7	788	32
TRU-3, RSL/OML	820	4
TRU-3, TRAU TS0..TS7	824	32
TRU-4, RSL/OML	856	4
TRU-4, TRAU TS0..TS7	860	32
TRU-5, RSL/OML	928	4
TRU-5, TRAU TS0..TS7	932	32
TRU-6, RSL/OML	964	4
TRU-6, TRAU TS0..TS7	968	32
TRU-7, RSL/OML	1000	4
TRU-7, TRAU TS0..TS7	1004	32
TRU-8, RSL/OML	1036	4
TRU-8, TRAU TS0..TS7	1040	32
TRU-9, RSL/OML	1072	4
TRU-9, TRAU TS0..TS7	1076	32
TRU-10, RSL/OML	1108	4
TRU-10, TRAU TS0..TS7	1112	32
TRU-11, RSL/OML	1144	4
TRU-11, TRAU TS0..TS7	1148	32

Note
In case voice TRAU frames are transferred, only the first of the four 16kbps subslots is used. When the timeslot is switched to GPRS/EGPRS, the full 64kbps bandwidth will be used. This also means that the set of four ICPs per TRAU must be connected consecutively. Also the connection to the outside must be aligned to E1 timeslot borders.

12 OsmoBSC example configuration files

The `osmo-bsc/doc/examples/osmo-bsc` directory in the OpenBSC source tree contains a collection of example configuration files, sorted by BTS type.

This chapter is illustrating some excerpts from those examples

12.1 Example configuration for OsmoBSC with one single-TRX nanoBTS

Example 12.1 OsmoBSC with one single-TRX nanoBTS

```
e1_input
 e1_line 0 driver ipa ❶
network
 network country code 1
 mobile network code 1
 encryption a5 0
 neci 1
```

```
handover 0
bts 0
 type nanobts ❷
 band DCS1800 ❸
 cell_identity 0
 location_area_code 0x0001
 training_sequence_code 7
 base_station_id_code 63
 ms max power 15
 cell reselection hysteresis 4
 rxlev access min 0
 channel allocator mode set-all ascending
 rach tx integer 9
 rach max transmission 7
 ipa unit-id 1801 0 ❹
 oml ipa stream-id 255 line 0
 gprs mode none
 trx 0
  rf_locked 0
  arfcn 871 ❺
  nominal power 23
  max_power_red 20 ❻
  rsl e1 tei 0
   timeslot 0
    phys_chan_config CCCH+SDCCH4
   timeslot 1
    phys_chan_config SDCCH8
   timeslot 2
    phys_chan_config TCH/F
   timeslot 3
    phys_chan_config TCH/F
   timeslot 4
    phys_chan_config TCH/F
   timeslot 5
    phys_chan_config TCH/F
   timeslot 6
    phys_chan_config TCH/F
   timeslot 7
    phys_chan_config TCH/F
```

❶ You have to configure one virtual E1 line with the IPA driver in order to use Abis/IP. One e1_line is sufficient for any number of A-bis/IP BTSs, there is no limit like in physical E1 lines.

❷ The BTS type must be set using `type nanobts`

❸ The GSM band must be set according to the BTS hardware.

❹ The IPA Unit ID parameter must be set to what has been configured on the BTS side using the *BTS Manager* or `ipaccess-config`.

❺ The ARFCN of the BTS.

❻ All known nanoBTS units have a nominal transmit power of 23 dBm. If a `max_power_red` of 20 (dB) is configured, the resulting output power at the BTS Tx port is 23 - 20 = 3 dBm.

Note
The `nominal_power` setting does *not* influence the transmitted power to the BTS! It is a setting by which the system administrator tells the BSC about the nominal output power of the BTS. The BSC uses this as basis for calculations.

12.2 Example configuration for OsmoBSC with multi-TRX nanoBTS

Example 12.2 OsmoBSC configured for dual-TRX (stacked) nanoBTS

```
e1_input
 e1_line 0 driver ipa
network
 network country code 1
 mobile network code 1
 encryption a5 0
 neci 1
 handover 0
 bts 0
  type nanobts
  band DCS1800
  cell_identity 0
  location_area_code 0x0001
  training_sequence_code 7
  base_station_id_code 63
  ms max power 15
  cell reselection hysteresis 4
  rxlev access min 0
  channel allocator mode set-all ascending
  rach tx integer 9
  rach max transmission 7
  ipa unit-id 1800 0 ❶
  oml ipa stream-id 255 line 0
  gprs mode none
  trx 0
   rf_locked 0
   arfcn 871
   nominal power 23
   max_power_red 0
   rsl e1 tei 0
    timeslot 0
     phys_chan_config CCCH+SDCCH4
    timeslot 1
     phys_chan_config SDCCH8
    timeslot 2
     phys_chan_config TCH/F
    timeslot 3
     phys_chan_config TCH/F
    timeslot 4
     phys_chan_config TCH/F
    timeslot 5
     phys_chan_config TCH/F
    timeslot 6
     phys_chan_config TCH/F
    timeslot 7
     phys_chan_config TCH/F
  trx 1
   rf_locked 0
   arfcn 873
   nominal power 23
   max_power_red 0
   rsl e1 tei 0
    timeslot 0
     phys_chan_config SDCCH8
    timeslot 1
     phys_chan_config TCH/F
    timeslot 2
```

```
     phys_chan_config TCH/F
    timeslot 3
     phys_chan_config TCH/F
    timeslot 4
     phys_chan_config TCH/F
    timeslot 5
     phys_chan_config TCH/F
    timeslot 6
     phys_chan_config TCH/F
    timeslot 7
     phys_chan_config TCH/F
```

❶ In this example, the IPA Unit ID is specified as `1800 0`. Thus, the first nanoBTS unit (`trx 0`) needs to be configured to 1800/0/0 and the second nanoBTS unit (`trx 1`) needs to be configured to 1800/0/1. You can configure the BTS unit IDs using the `ipaccess-config` utility included in OsmoBSC.

Note
For building a multi-TRX setup, you also need to connect the TIB cables between the two nanoBTS units, as well as the coaxial/RF AUX cabling.

12.3 Example configuration for OsmoBSC with E1 BTS

The following configuration sample illustrates the usage of BTSs that are connected via an E1/T1 backhaul.

Example 12.3 OsmoBSC configured for single-TRX E1 Ericsson DUG20

```
e1_input ❶
 e1_line 0 driver dahdi
 e1_line 0 port 3
network
 network country code 1
 mobile network code 1
 encryption a5 0
 neci 1
 handover 0
 bts 0
  type rbs2000
  band GSM900
  om2000 version-limit oml gen 12 rev 10 ❷
  cell_identity 0
  location_area_code 0x0001
  training_sequence_code 7
  base_station_id_code 63
  ms max power 15
  cell reselection hysteresis 4
  rxlev access min 0
  channel allocator mode set-all ascending
  rach tx integer 9
  rach max transmission 7
  oml e1 line 0 timeslot 1 sub-slot full ❸
  oml e1 tei 62 ❹
  gprs mode none
  is-connection-list add 4 512 12 ❺
  is-connection-list add 16 524 12
  is-connection-list add 28 536 12
  is-connection-list add 40 548 12
  trx 0
```

```
  rf_locked 0
  arfcn 123
  nominal power 42
  max_power_red 12
  rsl e1 line 0 timeslot 1 sub-slot full ❻
  rsl e1 tei 0 ❼
  timeslot 0
   phys_chan_config CCCH+SDCCH4
   hopping enabled 0
   e1 line 0 timeslot 1 sub-slot full ❽
  timeslot 1
   phys_chan_config TCH/F
   hopping enabled 0
   e1 line 0 timeslot 2 sub-slot 1 ❾
  timeslot 2
   phys_chan_config TCH/F
   hopping enabled 0
   e1 line 0 timeslot 2 sub-slot 2
  timeslot 3
   phys_chan_config TCH/F
   hopping enabled 0
   e1 line 0 timeslot 2 sub-slot 3
  timeslot 4
   phys_chan_config TCH/F
   hopping enabled 0
   e1 line 0 timeslot 3 sub-slot 0
  timeslot 5
   phys_chan_config TCH/F
   hopping enabled 0
   e1 line 0 timeslot 3 sub-slot 1
  timeslot 6
   phys_chan_config TCH/F
   hopping enabled 0
   e1 line 0 timeslot 3 sub-slot 2
  timeslot 7
   phys_chan_config TCH/F
   hopping enabled 0
   e1 line 0 timeslot 3 sub-slot 3
```

---

❶ In this example we use a dahdi E1 card. This card has 4 ports. Here we use port numer 3. It should be noted that the dahdi driver also requires additional configuration, which is not covered by this manual.

❷ In this example we use an E1 Ericsson DUG20, which uses an OML dialect, called "OM2000".

❸ The first usable timeslot on an E1 line is TS1. In this example we will assume that TS1-TS3 are connected to the BTS stright through. TS1 will handle all signaling traffic. Here we assign this timeslot to OML.

❹ OML always requires a TEI (Terminal Equipment Identifier) to set up. This number can be found in the manual of the BTS.

❺ This BTS has an built in “Interface Switch” (IS) that offers flexible way to reconfigure the interconnection between the internal components of the BTS and the external E1 line. This depends on the exact BTS type and configuration. See also Section 11.10

❻ Similar to OML we assign TS1 to RSL as well.

❼ Like with OML, RSL also requires a TEI to be configured. Usually each TRX will have a specific TEI assigned.

❽ CCCH+SDCCH4 will also be mapped on TS1. The traffic for those control channels will be multiplexed alongside the RSL and OML traffic.

❾ The bandwidth of one E1 timeslot matches the bandwidth of 4 GSM air interface timeslots. The E1 timeslot is split up into four sub-slots, which are then assigned to one GSM air interface timeslot each. Since the first timeslot on the first TRX is already used for signaling we begin the sub-slot counting with sub-slot 1 for alignment reasons.

12.4 Example configuration for OsmoBSC with Ericsson RBS E1 BTS and EGPRS

The following example illustrates the usage of Ericsson RBS2000/RBS6000 BTSs. This classic E1 BTS has no built in PCU and therefore requires the configuration of a BSC co-located OsmoPCU (see also: Section 5.5.4).

It should also be noted that the Ericsson RBS2000/RBS6000 series is the first BTS of this type to be supported by OsmoBTS and OsmoPCU. The implementation has been made possible through funding by the NLnet Foundation.

Ericsson RBS2000/RBS6000 BTSs feature two GPRS modes. A 16kbps GPRS mode where only CS1 and CS2 are supported and an EGPRS mode where MCS1 to MCS9 are supported. OsmoPCU offers support for both modes but since the 16kbps mode only supports classic GPRS with CS1 and CS2 it is more of experimental interest and shall not be discussed further. The following example will describe how to configure the 64kbps mode with EGPRS.

In the following example we also expect that the user is already familliar with the E1 configuration example above (see also: Section 12.3)

Example 12.4 OsmoBSC configured for single-TRX E1 Ericsson DUG20 with EGPRS

```
e1_input
 e1_line 0 driver dahdi
 e1_line 0 port 3
network
 network country code 1
 mobile network code 1
 encryption a5 0
 neci 1
 handover 0
 pcu-socket /tmp/pcu_bts ❶
 bts 0
  type rbs2000
  band GSM900
  om2000 version-limit oml gen 12 rev 10
  cell_identity 0
  location_area_code 0x0001
  training_sequence_code 7
  base_station_id_code 63
  ms max power 15
  cell reselection hysteresis 4
  rxlev access min 0
  channel allocator mode set-all ascending
  rach tx integer 9
  rach max transmission 7
  oml e1 line 0 timeslot 1 sub-slot full
  oml e1 tei 62
  gprs mode egprs ❷
  gprs routing area 0
  gprs network-control-order nc0
  gprs cell bvci 2
  gprs nsei 101
  gprs nsvc 0 nsvci 101
  gprs nsvc 0 local udp port 23100
  gprs nsvc 0 remote udp port 23000
  gprs nsvc 0 remote ip 127.0.0.1
  is-connection-list add 4 712 36 ❸
  trx 0
   rf_locked 0
   arfcn 123
   nominal power 42
```

```
  max_power_red 12
  rsl e1 line 0 timeslot 1 sub-slot full
  rsl e1 tei 0
  timeslot 0
   phys_chan_config CCCH+SDCCH4
   hopping enabled 0
   e1 line 0 timeslot 1 sub-slot full
  timeslot 1
   phys_chan_config TCH/F
   hopping enabled 0
   e1 line 0 timeslot 3 sub-slot full ❹
  timeslot 2
   phys_chan_config TCH/F
   hopping enabled 0
   e1 line 0 timeslot 4 sub-slot full
  timeslot 3
   phys_chan_config TCH/F
   hopping enabled 0
   e1 line 0 timeslot 5 sub-slot full
  timeslot 4
   phys_chan_config TCH/F_TCH/H_SDCCH8_PDCH ❺
   hopping enabled 0
   e1 line 0 timeslot 6 sub-slot full
  timeslot 5
   phys_chan_config TCH/F
   hopping enabled 0
   e1 line 0 timeslot 7 sub-slot full
  timeslot 6
   phys_chan_config TCH/F
   hopping enabled 0
   e1 line 0 timeslot 8 sub-slot full
  timeslot 7
   phys_chan_config TCH/F
   hopping enabled 0
   e1 line 0 timeslot 9 sub-slot full
```

❶ This configures the PCU socket path (see also: Section 5.5.4)

❷ This configures the general GPRS parameters. The configuration is no different from BTS with built-in PCU.

❸ The Ericsson RBS2000/RBS6000 series has an built in "Interface Switch" (IS) that offers flexible way to reconfigure the interconnection between the internal components of the BTS and the external E1 line. Since 16kbps subslots cannot supply the bandwidth required for EGPRS the IS must be configured to connect the 64kbps interface of the TRU to the external E1 line. For a more detailed description of the IS see Section 11.10.

❹ Since we are using the 64kbps TRU interface we must configure a full E1 timeslot per air interface time slot. For Speech this will have no effect on the TRAU frame format. The only difference is that always the first 16kbps subslot of the assigned E1 timeslot is used. OsmoMGW will be instructed accordingly by OsmoBSC, so no re-configuration of OsmoMGW is required.

❺ In this example we will use air interface TS 4 as PDCH. As mentioned earlier Ericsson RBS2000/RBS6000 supports the *DYNAMIC/OSMOCOM* timeslot model. PDCH timeslots must be configured as dynamic timeslots. It is not possible to configure static PDCHs. Therefore the phys_chan_config must be set to TCH/F_TCH/H_SDCCH8_PDCH in order to use the air interface timeslot as PDCH.

Note
As of March 2023 the BSC co-located PCU support for Ericsson RBS was tested only with a single BTS. Even though OsmoBSC and OsmoPCU should be able to handle multiple BTS, unexpected bahviour should be taken into account.

12.5 E1 Line number and MGCP trunk number

The switching of the voice channels is done via OsmoMGW, which acts as a media converter between E1 and VoIP (RTP). OsmoBSC will use the E1 line number to address the trunk via MGCP.

When configuring OsmoMGW, one needs to make sure that the trunk number that is set up on OsmoMGW, matches the line number that is set up on OsmoBSC. When those numbers mismatch the trunk cannot be addressed correctly.

Example 12.5 OsmoMGW trunk configuration that matches the OsmoBSC configuration above

```
trunk 0
 rtp keep-alive once
 no rtp keep-alive
 line 0
```

13 BSC level configuration

13.1 Hand-over

13.1.1 Hand-over in GSM

Hand-over is the process of changing a MS with a currently active dedicated channel from one BTS to another BTS. As opposed to idle mode, where the MS autonomously performs cell re-selection, in dedicated mode this happens under network control.

In order to determine when to perform hand-over, and to which cells, the network requests the MS to perform measurements on a list of neighbor cell channels, which the MS then reports back to the network in the form of GSM RR *Measurement Result* messages. Those messages contain the downlink measurements as determined by the MS.

Furthermore, the BTS also performs measurements on the uplink, and communicates those by means of RSL to the BSC.

The hand-over decision is made by an algorithm that processes those measurement results and determines when to perform the hand-over.

13.1.2 Configuration of hand-over in OsmoBSC

OsmoBSC only support so-called intra-BSC hand-over, where the hand-over is performed between two BTSs within the same BSC.

Hand-over is enabled and configured by the use of a set of `handover` commands. Using those, you can tune the key parameters of the hand-over algorithm and adapt it to your specific environment.

Example handover configuration snippet

```
handover 1 ❶
handover window rxlev averaging 10 ❷
handover window rxqual averaging 1 ❸
handover window rxlev neighbor averaging 10 ❹
handover power budget interval 6 ❺
handover power budget hysteresis 3 ❻
handover maximum distance 9999 ❼
```

❶ Enable hand-over

❷ Set the RxLev averaging window for the serving cell to 10 measurements

❸ Set the RxQual averaging window for the serving cell to 1 measurement (no window)

❹ Set the RxLev averaging for neighbor cells to 10 measurements

5. Check for the conditions of a power budget hand-over every 6 SACCH frames
6. A neighbor cell must be at least 3 dB stronger than the serving cell to be considered a candidate for hand-over
7. Perform a maximum distance hand-over if TA is larger 9999 (i.e. never)

13.2 Timer Configuration

The GSM specification specifies a variety of timers both on the network as well as on the mobile station side.

Those timers can be configured using the `timer tXXXX` command.

Table 16: Configurable Timers

node	timer	default	description
network	t3101	3	Timeout for *Immediate Assignment* (sec)
network	t3103	5	Timeout for Handover (sec)
network	t3105	100	Repetition of *Physical Information* (millisec)
network	t3107	5	?
network	t3109	5	RSL SACCH deactivation timeout (sec)
network	t3111	2	RSL timeout to wait before releasing the RF channel (sec)
network	t3113	7	Time to try paging for a subscriber (sec)
network	t3115	10	?
network	t3117	10	?
network	t3119	10	?
network	t3122	10	Waiting time after *Immediate Assignment Reject*
network	t3141	10	?

13.3 Discontinuous Transmission (DTX)

GSM provides a full-duplex voice call service. However, in any civilized communication between human beings, only one of the participants is speaking at any given point in time. This means that most of the time, one of the two directions of the radio link is transmitting so-called *silence frames*.

During such periods of quiescence in one of the two directions, it is possible to suppress transmission of most of the radio bursts, as there is no voice signal to transport. GSM calls this feature *Discontinuous Transmission*. It exists separately for uplink (DTXu) and downlink (DTXd).

Downlink DTX is only permitted on non-primary transceivers (!= TRX0), as TRX0 must always transmit at constant output power to ensure it is detected during cell selection.

Uplink DTX is possible on any TRX, and serves primarily two uses:

1. reducing the MS battery consumption by transmitting at a lower duty cycle
2. reducing the uplink interference caused in surrounding cells that re-use the same ARFCN.

DTX for both uplink and downlink is implemented in the BTS. Not all BTS models support it.

The Osmocom BSC component can instruct the BTS to enable or disable uplink and/or downlink DTX by means of A-bis OML.

14 Channel allocation

Radio resource management is one of the main tasks of the Base Station Controller. This involves selection, activation, and deactivation of logical channels, which are maintained by connected Base Stations. The number of usable logical channels is limited by total number of radio carriers and may vary depending on the physical channel combinations assigned to their timeslots. Thus a major goal of the this task is to manage all the available resources in an efficient way, shifting the balance between service quality and the overall capacity.

14.1 Channel allocation parameters

OsmoBSC's channel allocator can be configured via the VTY interface. All relevant parameters are limited by the scope of a BTS node they belong to. There is currently no way to define global configuration for all BTS.

All parameters with their respective default values are listed below:

```
network
 bts 0
  channel allocator mode chan-req ascending
  channel allocator mode assignment ascending
  channel allocator mode handover ascending
  channel allocator avoid-interference 0
  channel allocator tch-signalling-policy always
```

14.1.1 Channel allocation modes

Currently the following channel allocation modes are supported:

- ascending (default): allocates channels in ascending order, starting from timeslot 0 of the first TRX (also called C0, the BCCH carrier);
- descending: allocates channels in descending order, starting from timeslot 7 of the last TRX;
- dynamic (only for assignment): dynamically choose between ascending and descending order depending on some additional parameters (see Section 14.1.1.1).

Note
Regardless of the chosen mode, logical channels (sub-slots) are always selected in ascending order. For example, if a timeslot is configured as SDCCH/8 and all 8 sub-slots are not in use, then the first SDCCH(0) sub-slot will be selected in both ascending and descending modes.

The allocation mode to be used can be configured using the following VTY command:

```
OsmoBSC(config-net-bts)# channel allocator mode ? ❶
  set-all     Set a single mode for all variants
  chan-req    Channel allocation for CHANNEL REQUEST (RACH)
  assignment  Channel allocation for assignment
  handover    Channel allocation for handover

OsmoBSC(config-net-bts)# channel allocator mode set-all ? ❷
  ascending   Allocate Timeslots and Transceivers in ascending order
  descending  Allocate Timeslots and Transceivers in descending order

OsmoBSC(config-net-bts)# channel allocator mode assignment ? ❸
  ascending   Allocate Timeslots and Transceivers in ascending order
  descending  Allocate Timeslots and Transceivers in descending order
  dynamic     Dynamic lchan selection based on configured parameters ❹
```

❶ It's optionally possible to configure different allocation modes for different allocation causes, e.g. `ascending` for `chan-req` and `descending` for both `assignment` and `handover`.

❷ `set-all` is equivalent to the old (deprecated) command syntax: `channel allocator (ascending|descending)`.

❸, ❹ The `dynamic` mode can be selected only for `assignment`.

14.1.1.1 Dynamic channel allocation mode

There exists an additional channel allocation mode, which can be employed during a TCH channel allocation for assignment. This mode selects between ascending and descending order depending on pre-configured parameters:

- Uplink RxLev threshold and number of samples for averaging,
- C0 (BCCH carrier) channel load threshold.

This is useful in setups where Tx power of the RF carriers cannot be adjusted dynamically at run-time and thus no BS Power Control can be performed. In such setups the BCCH carrier is transmitting at relatively higher power than the other RF carriers. The key idea is to allocate channels in a smarter way, so that UEs with poor signal would get channels on carriers with high Tx power, while UEs with good signal could use carriers with lower Tx power.

The configuration parameters for dynamic selection are listed below:

```
OsmoBSC(config-net-bts)# channel allocator dynamic-param ?
  sort-by-trx-power  Whether to sort TRX instances by their respective power levels
  ul-rxlev           Uplink RxLev
  c0-chan-load       C0 (BCCH carrier) channel load

channel allocator dynamic-param sort-by-trx-power ?
  0  Do not sort, use the same order as in the configuration file
  1  Sort TRX instances by their power levels in descending order

OsmoBSC(config-net-bts)# channel allocator dynamic-param ul-rxlev thresh ?
  <0-63>  Uplink RxLev threshold
OsmoBSC(config-net-bts)# channel allocator dynamic-param ul-rxlev thresh 50 avg-num ?
  <1-10>  Minimum number of RxLev samples for averaging
OsmoBSC(config-net-bts)# channel allocator dynamic-param c0-chan-load thresh ?
  <0-100>  Channel load threshold (in %)
```

The default values are:

```
network
 bts 0
  channel allocator dynamic-param sort-by-trx-power 0 ❶
  channel allocator dynamic-param ul-rxlev thresh 50 avg-num 2 ❷
  channel allocator dynamic-param c0-chan-load thresh 60 ❸
```

❶ Assume that RF carriers are listed in descending order sorted by Tx power.

❷ Use descending order if AVG of at least two Uplink RxLev samples >= 50 (-60 dBm).

❸ Use descending order if more than 60% logical channels of C0 are occupied.

Note
The final ascending/descending order decision is based on the two conditions. The descending order will be used only if **both conditions are met**, otherwise the allocator will use ascending order.

14.1.2 Interference aware channel allocation

The channel allocator can be configured to prefer logical channels with least interference, based on interference measurements periodically sent by the BTSs (see Section 16). This is an optional feature, which is disabled by default.

```
OsmoBSC(config-net-bts)# channel allocator avoid-interference ?
  0  Ignore interference levels (default). Always assign lchans
     in a deterministic order.
  1  In channel allocation, prefer lchans with less interference.
```

Note
Interference levels are compared within the scope of the whole BTS. This means that the selection logic may pick channels on the other TRXes, if they are better according to the interference reports from the BTS. This feature makes the allocation order non-deterministic and therefore nullifies the meaning of channel allocation modes described above.

14.1.3 TCH sigalling policy

By default, in a situation when all SDCCHs are exhausted, OsmoBSC will be using TCH channels for signalling (e.g for Location Updating or call establishment). This behavior can be restricted to certain kinds of signalling or disabled completely.

```
OsmoBSC(config-net-bts)# channel allocator tch-signalling-policy ?
  never      Never allow TCH for signalling purposes
  emergency  Only allow TCH for signalling purposes when establishing an emergency call
  voice      Allow TCH for signalling purposes when establishing any voice call
  always     Always allow TCH for signalling purposes (default)
```

15 Power control

The objective of power control is to regulate the transmit power of the MS (Uplink) as well as the BTS (Downlink) in order to achieve the optimal reception conditions, i.e. a desired signal strength and a desired signal quality.

There are two advantages of power control:

- reduction of the average power consumption (especially in the MS), and
- reduction of the co-channel interference for adjacent channel users.

Power control can be performed either by the BSC, or by the BTS autonomously. OsmoBSC currently lacks the power control logic, so it cannot act as the regulating entity, however it's capable to instruct a BTS that supports autonomous power control to perform the power regulation. This is achieved by including vendor- specific IEs with power control parameters in the channel activation messages on the A-bis/RSL interface.

15.1 Power control parameters

Unfortunately, 3GPP specifications do not specify the exact list of power control parameters and their encoding on the A-bis/RSL interface, so it's up to a BTS/BSC vendor what to send and in which format. Furthermore, there is no public documentation on which parameters are accepted by particular BTS models.

3GPP TS 44.008 nonetheless defines a minimal set of parameters for a general power control algorithm. OsmoBSC allows to configure these parameters via the VTY interface, this is further described in the next sections.

So far only the ip.access specific format is implemented, so it should be possible to enable power control for nanoBTS. OsmoBTS also accepts this format, but may ignore some of the received parameters due to incomplete implementation. On the other hand, OsmoBTS may support some extra parameters coming in Osmocom specific IEs not supported by nanoBTS, such as those configuring C/I measurement thresholds.

15.1.1 When the parameters come into effect?

It depends on how the power control parameters are signaled to the BTS. If a given BTS vendor/model requires *each* RSL CHANnel ACTIVation message to contain the full set of parameters, then changing them in the BSC at run-time would affect all newly established logical channels immediately. The existing connections would continue to use parameters which were in use during the time of channel activation.

For both ip.access nanoBTS and OsmoBTS, the configured parameters are being sent only once when the A-bis/RSL link is established. In all subsequent RSL messages, the MS/BS Power Parameters IE will be sent empty. Therefore, changing most of dynamic power control parameters at run-time would affect neither the existing nor newly established logical channels.

It's still possible to "push" a modified set of MS/BS power control parameters to a BTS that accepts the default parameters at startup without triggering the A-bis/RSL link re-establishment and thus interrupting the service. The following command triggers resending of both MS/BS power control parameters:

Example: Resending default power control parameters via the VTY

```
# Resending from the 'enable' node:
OsmoBSC# bts 0 ❶ resend-power-control-defaults

# Resending from any configuration node (note prefix 'do'):
OsmoBSC(config-ms-power-ctrl)# do bts 0 ❷ resend-power-control-defaults
```

❶, ❷ BTS number for which to resend default power control parameters.

Example: Resending default power control parameters via the CTRL

```
$ osmo_ctrl.py \
        --host 127.0.0.1 ❶ -p 4249 \
        --set "bts.0.send-power-control-defaults" 1 ❷
```

❶ Remote address of the host running osmo-bsc (localhost in this example).

❷ An arbitrary dummy value (ignored). Required because SET command is used.

Note
The above statement mostly applies to parameters for dynamic power control mode (see below). Switching between power control modes, as well as changing static/maximum power values, does not necessarily require resending of parameters.

15.2 Power control configuration

Two identical groups of parameters are available for both MS (Uplink) and BS (Downlink) power control. This chapter is aimed to put some light on them.

All parameters can be set via the VTY interface, currently within the scope of a BTS node. This means that all transceivers will "inherit" the same configuration.

```
OsmoBSC(config)# network
OsmoBSC(config-net)# bts 0
OsmoBSC(config-net-bts)# ?
...
  bs-power-control              BS (Downlink) power control parameters
  ms-power-control              MS (Uplink) power control parameters
...
```

Either of these commands would lead to a separate node:

```
OsmoBSC(config-net-bts)# ms-power-control
OsmoBSC(config-ms-power-ctrl)# list with-flags
...
  . l.  mode (static|dyn-bts) [reset]
  . l.  bs-power (static|dyn-max) <0-30>
  . lv  ctrl-interval <0-31>
  . lv  step-size inc <2-6> red <2-4>
  . lv  rxlev-thresh lower <0-63> upper <0-63>
  . lv  rxqual-thresh lower <0-7> upper <0-7>
  . lv  ci-thresh (fr-efr|hr|amr-fr|amr-hr|sdcch|gprs) lower <0-30> upper <0-30>
  . lv  rxlev-thresh-comp lower <0-31> <0-31> upper <0-31> <0-31>
  . lv  rxqual-thresh-comp lower <0-31> <0-31> upper <0-31> <0-31>
  . lv  ci-thresh-comp (fr-efr|hr|amr-fr|amr-hr|sdcch|gprs) lower <0-31> <0-31> upper  ↩
      <0-31> <0-31>
  . lv  no (rxlev-avg|rxqual-avg)
  . lv  (rxlev-avg|rxqual-avg) params hreqave <1-31> hreqt <1-31>
  . lv  (rxlev-avg|rxqual-avg) algo (unweighted|weighted|mod-median)
  . lv  (rxlev-avg|rxqual-avg) algo osmo-ewma beta <1-99>
  . lv  no ci-avg (fr-efr|hr|amr-fr|amr-hr|sdcch|gprs)
  . lv  ci-avg (fr-efr|hr|amr-fr|amr-hr|sdcch|gprs) params hreqave <1-31> hreqt <1-31>
  . lv  ci-avg (fr-efr|hr|amr-fr|amr-hr|sdcch|gprs) algo (unweighted|weighted|mod-median)
  . lv  ci-avg (fr-efr|hr|amr-fr|amr-hr|sdcch|gprs) algo osmo-ewma beta <1-99>
```

Note
Flag `v` indicates that a given parameter is vendor specific, so different BTS vendors/models may ignore or even reject it. Flag `l` indicates that changing a given parameter at run-time would affect only the new connections.

15.2.1 Power control mode

Three power control modes exist:

```
OsmoBSC(config-ms-power-ctrl)# mode ?
  static   Instruct the MS/BTS to use a static power level ❶
  dyn-bts  Power control to be performed dynamically by the BTS itself ❷
OsmoBSC(config-net-bts)# no (bs-power-control|ms-power-control) ❸
```

❶ Send RSL MS/BS Power IE alone indicating a static power level to the BTS.

❷ Send both RSL MS/BS Power IE and vendor-specific MS/BS Power Parameters IE.

❸ Do not send any power control IEs in RSL CHANnel ACTIVation messages.

By default, `static` mode is used for BS power control, while `dyn-bts` mode is automatically enabled for MS power control if vendor-specific format of the power control parameters (see above) is implemented for particular BTS model. Otherwise `static` mode is used too. Changing the mode at run-time would not affect already established connections, only the new ones (check flag `l`).

For BS power control, there is an additional parameter:

```
OsmoBSC(config-bs-power-ctrl)# bs-power ?
  static   Fixed BS Power reduction value (for static mode)
  dyn-max  Maximum BS Power reduction value (for dynamic mode)
```

that allows to configure the maximum BS power reduction value in `dyn-bts` mode, and a fixed power reduction value in `static` mode. In the later case, no attenuation (0 dB) is applied by default (full power).

15.2.2 Power control interval

Having requested a transmit power level, the MS/BS power control loop may optionally be suspended for a certain number of SACCH multiframes defined by VTY parameter `ctrl-interval`. Given that SACCH is relatively slow and transmission of a data block takes 480 ms, suspension allows an observation of the effect of one power control decision before initiating the next one. This is mostly important due to the fact that MS must change the transmit power at nominal steps of 2dB every 60ms (TS 45.008 sec 4.7.1). As a result, if the network requests the MS to change its transmit power by several MS Power Levels at a time, the MS will do so gradually over the next measurement period. Hence, upon next received L1 SACCH block, the *MS_PWR* value announced by the MS will match only the one used to transmit the last block, and so the related measured RxLevel/RxQual values will be inaccurate. By skipping one or several SACCH blocks, the algorithm will always use values which match correctly the announced *MS_PWR* and the measured RxLevel/RxQual (because the *MS_PWR* will already have changed and hence will be kept stable over that measurement period).

```
OsmoBSC(config-bs-power-ctrl)# ctrl-interval ?
  <0-31>  P_CON_INTERVAL, in units of 2 SACCH periods (0.96 seconds)
```

3GPP TS 45.008 briefly mentions this parameter in table A.1 (`P_Con_INTERVAL`).

A small time graph is depicted below for better understanding of the meaning of values for this parameter, since it is not obvious at all.

Example: Suspension interval accomplished by several values of P_CON_INTERVAL

```
|<-->| - one SACCH multi-frame period
|    |
|----|----|----|----|----|----|----|----|----> SACCH multi-frames
a) *    *    *    *    *    *    *    *    *      P_CON_INTERVAL=0 (0.48 s)
b) *         *         *         *         *      P_CON_INTERVAL=1 (0.96 s)
c) *                   *                   *      P_CON_INTERVAL=2 (1.92 s, default)
d) *                              *                P_CON_INTERVAL=3 (2.88 s)
e) *                                        *      P_CON_INTERVAL=4 (3.84 s)
```

The value to use for this parameter is closely related to that of VTY option `step-size inc <2-6> red <2-4>`, which configures the maximum step (in dB) at which the MS Power can be requested to changed when the MS Power Control Loop is triggered. The higher the `step-size`, the more time it will need the MS to do the necessary ramping of 2 dB steps, and hence the amount of time required for the MS to settle on the requested MS Power Level for an entire SACCH block. Or equally, the time the network can start using the full measurement period to trigger the MS Power Control Loop again with reliable measurements (`P_CON_INTERVAL`).

By default, increment `step-size` is set to 4 dB and the decrement `step-size` is set to 2 dB, hence the MS requiring `4 * 60 = 240` milliseconds. That's less than 1 measurement period (480 ms), hence only the first measurement period needs to be skipped. Therefore, a suspension interval of 1 for both MS/BS power control loops can be used, and so the power control decision is taken every 960 ms (every second SACCH block period).

However, OsmoBSC currently uses a default value of `ctrl-interval 2` (`P_CON_INTERVAL=2`, 1.92s, 3/4 received SACCH blocks are skipped), because that's the minimum amount of frames required for one loop step to run completely, that is: BTS fetching measurements and transmitting the new MS Power Level, then the MS retrieving the MS Power Level, transmitting with that exact MS Power level during the entire period and then finally submitting the Measurement Result containing that same MS Power Level. Using a value of `P_CON_INTERVAL=1` also provides good results, but in that case the loop tends to produce more temporary power oscillations due to the loop acting on periods where an older (earlier requested) MS Power level is still in use (and announced) by the MS.

Example: Timeline showing propagation of a new MS Power Level (P_CON_INTERVAL=2)

```
|<------------->| - one SACCH multi-frame period
| 1             | 2             | 3             | 4             | ---> SACCH multi-frames
|SA0|SA1|SA2|SA3|SA0|SA1|SA2|SA3|SA0|SA1|SA2|SA3|SA0|SA1|SA2|SA3| ---> SACCH bursts
|❶|...|...|❷|❸|...|...|❹|❺|...|...|❻|❼|...|...|❽|
```

❶ BTS sends new requested MS Power Level in header of SACCH

❷ MS receives SACCH block (new MS Power Level)

❸ MS starts ramping towards new MS Power Level (hence potentially variable power used over the period)

❹ MS should already be in desired MS Power Level (for step increments of less-or-equal than 8 dB)

❺ MS starts transmitting at desired MS Power level constantly (ramping is over)

❻ MS builds Measurement Results to be sent on next SACCH period

❼ MS sends the Measurement Results of the previous multiframe to the BTS

❽ BTS receives the Measurement Results from MS on SACCH, starts next loop iteration

Setting `ctrl-interval` to 0 increases the interval to 480 ms, so basically no SACCH block is skipped and MS Power Control loop is triggered upon receival of every UL SACCH block.

15.2.3 Power change step size

In order to slow down the reactivity of the power control loop and thus make it more robust against sporadic fluctuations of the input values (RxLev and either RxQual or C/I), the transmit power on both Uplink and Downlink is changed gradually, step by step.

OsmoBSC allows to configure the step sizes for both increasing and reducing directions separately. The corresponding power control loop would apply different delta values to the current transmit power level in order to raise or lower it.

Example: Power change step size

```
network
 bts 0
  bs-power-control
   mode dyn-bts ❶
   bs-power dyn-max 12 ❷
   step-size inc 6 red 4 ❸
  ms-power-control
   mode dyn-bts ❹
   step-size inc 4 red 2 ❺
```

❶, ❹ Both MS and BS power control is to be performed by the BTS autonomously.

❷ The BTS is allowed to reduce the power on Downlink up to 12 dB.

❸ On Downlink, BS power can be increased by 6 dB or reduced by 4 dB at once.

❺ On Uplink, MS power can be increased by 4 dB or reduced by 2 dB at once.

Note
In the context of BS power control, terms *increase* and *decrease* have the same meaning as in the context of MS power control: to make the output power stronger or weaker respectively. Even despite the BS power loop in fact controls the attenuation.

Tip
It's recommended to pick the values in a way that the increase step is greater than the reduce step. This way the system would be able to react on signal degradation quickly, while a good signal would not trigger radical power reduction.

Both parameters are mentioned in 3GPP TS 45.008, table A.1:

- Pow_Incr_Step_Size (range 2, 4 or 6 dB),
- Pow_Red_Step_Size (range 2 or 4 dB).

15.2.4 RxLev and RxQual thresholds

The general idea of power control is to maintain the signal level (RxLev) and quality (RxQual) within the target ranges. Each of these ranges can be defined as a pair of the lowest and the highest acceptable values called thresholds.

The process of RxLev / RxQual threshold comparison is described in 3GPP TS 45.008, section A.3.2.1. All parameters involved in the process can be found in table A.1 with the recommended default values.

Example: RxLev and RxQual threshold configuration

```
network
 bts 0
  bs-power-control
   mode dyn-bts ❶
   rxlev-thresh lower 32 upper 38 ❷
   rxqual-thresh lower 3 upper 0 ❸
```

❶ BS power control is to be performed by the BTS autonomously.

❷ RxLev is to be maintained in range 32 .. 38 (-78 .. -72 dBm).

❸ RxQual is to be maintained in range 3 .. 0 (less is better).

Note
For both RxLev and RxQual thresholds, the lower and upper values are included in the tolerance window. In the example above, RxQual=3 would not trigger the control loop to increase BS power, as well as RxLev=38 (-72 dBm) would not trigger power reduction.

Tip
It's recommended to harmonize the increase step size with the RxLev threshold window in a way that the former is less or equal than/to the later. For example, if the RxLev threshold is 32 .. 36 (-78 .. -74 dBm), then the window size is 4 dB, and thus the increase step should be less or equal (e.g. 2 or 4 dB).

In 3GPP TS 45.008, lower and upper RxLev thresholds are referred as `L_RXLEV_XX_P` and `U_RXLEV_XX_P`, while the RxQual thresholds are referred as `L_RXQUAL_XX_P` and `U_RXQUAL_XX_P`, where the `XX` is either `DL` (Downlink) or `UL` (Uplink).

The process of threshold comparison actually involves more than just upper and lower values for RxLev and RxQual. The received "raw" measurements are being averaged and stored in a circular buffer, so the power change is triggered only if Pn averages out of Nn averages exceed the corresponding thresholds.

Example: RxLev and RxQual threshold comparators

```
network
 bts 0
  bs-power-control
   mode dyn-bts ❶
   rxlev-thresh lower 32 upper 38 ❷
   rxlev-thresh-comp lower 10 12 ❸ upper 19 20 ❹
   rxqual-thresh lower 3 upper 0 ❺
   rxqual-thresh-comp lower 5 7 ❻ upper 15 18 ❼
```

❶ BS power control is to be performed by the BTS autonomously.

❷ L_RXLEV_XX_P=32, U_RXLEV_XX_P=38.

❸ P1=10 out of N1=12 averages < L_RXLEV_XX_P ⇒ increase power.

❹ P2=19 out of N2=20 averages > U_RXLEV_XX_P ⇒ decrease power.

❺ L_RXQUAL_XX_P=3, U_RXQAUL_XX_P=0.

❻ P3=5 out of N3=7 averages > L_RXQUAL_XX_P ⇒ increase power.

❼ P4=15 out of N4=18 averages < U_RXQUAL_XX_P ⇒ decrease power.

15.2.5 Carrier-to-Interference (C/I) thresholds

Carrier-to-Interference (C/I) provides a similar description of link quality to that provided by RxQual. However, C/I provides higher granularity than RxQual levels (0-7), hence providing the operator with a more refined way to set up targets levels and thresholds. C/I measurements can only be used for MS Power Control, since values are only available on the Uplink when computed by the BTS, and the MS doesn't provide figure for the Downlink to the BTS/BSC during measurement Reports.

Usual C/I value range for MS uplink channels are between 0dB to 30dB, with 9dB being a usual average target around cell edges. Furthermore, publicly available studies conclude that different channel types with different codecs used have different target C/I where signal is considered good enough. This means MS using a given channel type with better codec capabilities can be instructed to transmit at lower levels, hence reducing noise or channel interference among MS.

OsmoBTS MS Power Control Loop algorithm supports using C/I computed measurements. Related parameters can be configured similar to those of RxLev or RxQual (see previous section), with the main difference being that instead of having a global set of parameters, there's one set per channel type, hence allowing different parametrization based on the channel type in use by the MS.

Example: C/I threshold comparators for AMR-FR

```
network
 bts 0
  ms-power-control
   mode dyn-bts ❶
   ci-thresh amr-fr enable ❷
   ci-thresh amr-fr lower 7 upper 11 ❸
   ci-thresh-comp amr-fr lower 2 10 ❹ upper 3 4 ❺
```

❶ MS power control is to be performed by the BTS autonomously.

❷ MS power control loop should take C/I into account.

❸ L_CI_AMR_FR_XX_P=7, U_CI_AMR_FR_XX_P=11.

❹ P0=2 out of N1=10 averages < L_CI_AMR_FR_XX_P ⇒ increase power.

❺ P1=3 out of N2=4 averages > U_CI_AMR_FR_XX_P ⇒ decrease power.

Note
The BSC can instruct a BTS to disable C/I related logic in its autonomous MS Power Control Loop for a given channel type (hence not taking C/I measurements into account) by means of setting both related LOWER_CMP_N and UPPER_CMP_N parameters to zero (see *ci-thresh-comp* VTY command). For the sake of easing configuration, a placeholder VTY command to disable C/I for all channel types is available under VTY node *ms-power-control* as ***ci-thresh all disable***. Afterwards, the new configuration must be deployed to the target BTS (see Section 15.1.1).

15.2.6 Measurement averaging process

3GPP 45.008, section A.3.1 requires that the measurement values reported by both an MS and the BTS are being pre-processed before appearing on the input of the corresponding power control loops in any of the following ways:

- Unweighted average;

- Weighted average, with the weightings determined by O&M;
- Modified median calculation, with exceptionally high and low values (outliers) removed before the median calculation.

The pre-processing is expected to be performed by both MS and BS power control loops independently, for every input parameter (i.e. RxLev, RxQual and C/I).

```
OsmoBSC(config-bs-power-ctrl)# rxlev-avg algo ?
  unweighted  Un-weighted average
  weighted    Weighted average
  mod-median  Modified median calculation
  osmo-ewma   Exponentially Weighted Moving Average (EWMA)
OsmoBSC(config-bs-power-ctrl)# rxqual-avg algo ?
  unweighted  Un-weighted average
  weighted    Weighted average
  mod-median  Modified median calculation
  osmo-ewma   Exponentially Weighted Moving Average (EWMA)
```

OsmoBTS features a non-standard Osmocom specific EWMA (Exponentially Weighted Moving Average) based pre-processing. Other BTS models may support additional non-standard methods too, the corresponding VTY options can be added on request.

Among with the averaging methods, 3GPP 45.008 also defines two pre-processing parameters in section A.3.1:

- Hreqave - defines the period over which an average is produced, in terms of the number of SACCH blocks containing measurement results, i.e. the number of measurements contributing to each averaged measurement;
- Hreqt - is the number of averaged results that are maintained.

By default, OsmoBSC would not send any pre-processing parameters, so the BTS may apply its default pre-processing algorithm with default parameters, or may not apply any pre-processing at all - this is up to the vendor. The pre-processing parameters need to be configured explicitly as shown in the example below.

Example: Explicit pre-processing configuration

```
network
 bts 0
  bs-power-control
   mode dyn-bts ❶
   rxlev-avg algo unweighted ❷
   rxlev-avg params hreqave 4 hreqt 6 ❸
   rxqual-avg algo osmo-ewma beta 50 ❹
   rxqual-avg params hreqave 2 hreqt 3 ❺
  ms-power-control
   mode dyn-bts ❻
   rxlev-avg algo unweighted ❼
   rxlev-avg params hreqave 4 hreqt 6 ❽
   rxqual-avg algo osmo-ewma beta 50 ❾
   rxqual-avg params hreqave 2 hreqt 3 ❿
   ci-avg amr-fr algo osmo-ewma beta 50 ⓫
   ci-avg amr-fr params hreqave 2 hreqt 3 ⓬
```

❶, ❻ Both MS and BS power control is to be performed by the BTS autonomously.

❷, ❼ Unweighted average is applied to RxLev values.

❸, ❽ RxLev: Hreqave and Hreqt values: 4 out of 6 SACCH blocks produce an averaged measurement.

❹, ❾ Osmocom specific EWMA is applied to RxQual values with smoothing factor = 50% (beta=0.5).

❺, ❿ RxQual: Hreqave and Hreqt values: 2 out of 3 SACCH blocks produce an averaged measurement.

⓫ Osmocom specific EWMA is applied to C/I values on AMR-FR channels with smoothing factor = 50% (beta=0.5).

⓬ C/I AMR-FR: Hreqave and Hreqt values: 2 out of 3 SACCH blocks produce an averaged measurement.

15.3 BCCH carrier power reduction operation

According to 3GPP TS 45.008, section 7.1, the BCCH carrier (sometimes called C0) of a BTS shall maintain continuous Downlink transmission at full power in order to stay "visible" to the mobile stations. Because of that, early versions of this 3GPP document prohibited BS power reduction on C0. However, a new feature was introduced in version 13.0.0 (2015-11) - "BCCH carrier power reduction operation".

This is a special mode of operation, in which the variation of RF power level for some timeslots is relaxed for the purpose of energy saving. In other words, the output power on some timeslots, except the timeslot(s) carrying BCCH/CCCH, can be lower than the full power. In this case the maximum allowed difference is 6 dB.

Of course, energy saving comes at a price and has impacts to the network KPI. In particular, it does negatively affect cell reselection performance and does increase handover failure and call drop rates. This is why BCCH carrier power reduction operation mode is not enabled by default. More information on potential impact and the simulation results can be found in 3GPP TR 45.926.

15.3.1 Supported BTS models

At the time of writing this manual, the only BTS model that can be instructed to enter or leave the BCCH power reduction mode is osmo-bts-trx. Support for other BTS vendors/models may be added in the future.

Tip
If you're using OsmoBTS, make sure that it reports feature #021 "BCCH carrier power reduction mode" in the feature vector. This can be checked by issuing `show bts` command in OsmoBSC's VTY interface.

15.3.2 Interworking with static and dynamic power control

The key difference between BCCH carrier power reduction and the BS power control is that the former affects **inactive** timeslots (or sub-channels), so only dummy bursts are affected. The later depends on the Downlink measurement reports sent by the MS, and thus applies to **active** channels only. However, both features are interconnected: the maximum BCCH carrier power reduction value constrains the BS Power value that can be used for dynamic or static BS power control.

BS power control on the BCCH carrier will not be enabled unless the BTS is in BCCH carrier power reduction mode of operation. Once it is, the BS power reduction value in either of `dyn-bts` or `static` modes would be constrained by currently applied BCCH power reduction value, and thus would never exceed the maximum of 6 dB.

For example, consider a BTS with BS power control configured to use *dynamic* mode and the maximum power reduction of 16 dB. Once this BTS is switched into the BCCH carrier power reduction mode with the maximum attenuation of 4 dB, the maximum power reduction value for the BS power loop on the C0 carrier would be 4 dB.

Moreover, according to 3GPP TS 45.008, between a timeslot used for BCCH/CCCH and the timeslot preceding it, the difference in output power actually transmitted by the BTS shall not exceed 3 dB. This means that on some timeslots the power reduction value can be constrained even further.

15.3.3 Managing BCCH carrier power reduction

The BCCH carrier power reduction can be controlled via the CTRL and VTY interfaces. There is currently no logic in OsmoBSC for automatic activation and deactivation of this mode, so it's up to the network operator (or an external monitoring suite) when and depending on which factors to toggle it. Setting a value greater than zero enables the BCCH power reduction mode; setting zero disables it completely.

Example: Activating BCCH carrier power reduction via the VTY

```
OsmoBSC> enable
OsmoBSC# bts 0 ❶ c0-power-reduction ?
  <0-6>  Power reduction value (in dB, even numbers only)
OsmoBSC# bts 0 ❷ c0-power-reduction 4 ❸
```

❶, ❷ BTS number for which to activate BCCH carrier power reduction

❸ Maximum BCCH carrier power reduction (in 2 dB steps, 4 dB in this example)

Example: Activating BCCH carrier power reduction via the CTRL

```
$ osmo_ctrl.py \
        --host 127.0.0.1 ❶ -p 4249 \
        --set "bts.0.c0-power-reduction" 4 ❷
```

❶ Remote address of the host running osmo-bsc (localhost in this example)

❷ Maximum BCCH carrier power reduction (in 2 dB steps, 4 dB in this example)

Once activated, it's possible to introspect the current maximum reduction value:

Example: Checking BCCH carrier power reduction state via the VTY

```
OsmoBSC> enable
OsmoBSC# show bts 0 ❶
BTS 0 is of osmo-bts type in band DCS1800, has CI 0 LAC 1, BSIC 63 (NCC=7, BCC=7) and 2 TRX
  Description: (null)
  ARFCNs: 751 753
  BCCH carrier power reduction (maximum): 4 dB ❷
  ...
```

❶ BTS number for which to show BCCH carrier power reduction state

❷ Maximum BCCH carrier power reduction currently applied

Example: Checking BCCH carrier power reduction state via the CTRL

```
$ osmo_ctrl.py \
        --host 127.0.0.1 ❶ -p 4249 \
        --get "bts.0.c0-power-reduction"
Got message: b'GET_REPLY 3652121201381481804 bts.0.c0-power-reduction 4 ❷'
```

❶ Remote address of the host running osmo-bsc (localhost in this example)

❷ Maximum BCCH carrier power reduction currently applied

15.4 Temporary ACCH overpower

Temporary overpower (TOP) is a power control technique that allows to improve SACCH/FACCH performance in case of bad C/I. The key idea of TOP is to increment the BS transmit power by 2..4 dB only for FACCH/SACCH bursts, while keeping all voice bursts at the lower (normal) level as determined by the downlink power control loop. This allows to reduce call drop rate and increase capacity in deployments with tight frequency reuse.

Note
It's not possible to increase the current BS power beyond the maximum transmit power level supported by the PHY. Thus if the BTS is already transmitting at full power, the overpower logic cannot increase it even further. This is also why TOP must be employed **together with BS power control**, either static or dynamic.

The main area of use for TOP is traffic channels employing the AMR (Adaptive Multi Rate) codec, which is more robust to interference than the associated signalling channels. While AMR provides sufficient speech quality even at very low C/I levels, the associated signalling channels may be suffering from channel coding errors. This imbalance can be compensated by employing TOP, which can be efficiently combined with the ACCH repetition technique.

This feature requires no support on the mobile station side and can be used with UEs implementing the most recent 3GPP relese features, as well as legacy UEs. However, it needs to be implemented in the BTS. Given that TOP itself is not specified in 3GPP specifications, osmo-bsc uses Osmocom specific A-bis/RSL IEs in order to activate it. Therefore, only the recent osmo-bts versions may be instructed to activate this feature. Make sure that feature #023 "FACCH/SACCH Temporary overpower" is present in the feature vector. This can be checked by issuing `show bts` command in OsmoBSC's VTY interface.

TOP is disabled by default. Below is a configuration example enabling it:

```
network
 bts 0
  overpower dl-acch 2 ❶
  overpower rxqual 4 ❷
  overpower chan-mode speech-amr ❸
```

❶ Overpower of maximum 2 dB for both SACCH and FACCH.

❷ Enable TOP only if RxQual is worse than 4 (BER >= 1.6%).

❸ Permit TOP only for speech channels using AMR codec.

For advanced use cases, OsmoBSC can be configured to:

- enable TOP only for FACCH or SACCH selectively, and/or
- keep TOP enabled permanently regardless of the reported RxQual, and/or
- permit TOP for any kind of dedicated channels.

```
OsmoBSC(config-net-bts)# overpower ?
  dl-acch   Enable overpower for both SACCH and FACCH
  dl-sacch  Enable overpower for SACCH only
  dl-facch  Enable overpower for FACCH only

OsmoBSC(config-net-bts)# overpower rxqual 0?
  0  BER >= 0% (always on)

OsmoBSC(config-net-bts)# overpower chan-mode ?
  speech-amr  Speech channels using AMR codec (default)
  any         Any kind of channel mode
```

These parameters are indicated to the BTS during a logical channel activation or modifications procedures, so they can be changed at run-time.

16 Interference reporting

According to 3GPP 48.058, section 6.1, the BTS shall periodically report the interference levels on **idle** channels using the "Radio resource indication" procedure. This is done by sending the `RF RESource INDication` message, which is specified in sections 8.6.1 and 9.3.21.

16.1 Interference reporting parameters

The interference band is calculated by the BTS based on the `Interference level Boundaries` and the `Averaging period`. These parameters are sent by the BSC over the A-bis/OML, and can be configured via the VTY interface.

Below are the default values for them:

```
network
 bts 0
  interference-meas avg-period 6 ❶
  interference-meas level-bounds -115 ❷ -109 -103 -97 -91 -85 ❸
```

❶ Averaging period (`Intave`) in SACCH multiframe periods (480ms).

❷ Interference level boundary `0` (in dBm).

❸ Interference level boundary `X5` (in dBm).

The `Intave` parameter defines the averaging and reporting period. With the default value of 6 SACCH multiframe periods the BTS is instructed to report averaged interference levels approximately every 3 seconds.

According to 3GPP TS 48.008, there exist five interference bands and six `Interference level Boundaries` (`0`, `X1`, ... `X5`). The BTS shall map the averaged interference levels (initially in dBm) into these 5 bands.

```
-115 dBm   -109 dBm   -103 dBm    -97 dBm    -91 dBm    -85 dBm
    | ❶      | ❷      | ❸      | ❹      | ❺      | ❻
    +----------+----------+----------+----------+----------+
    |  band 1  |  band 2  |  band 3  |  band 4  |  band 5  |
    +----------+----------+----------+----------+----------+
```

❶ Interference level boundary `0` (outer).

❷ Interference level boundary `X1`.

❸ Interference level boundary `X2`.

❹ Interference level boundary `X3`.

❺ Interference level boundary `X4`.

❻ Interference level boundary `X5` (outer).

Unfortunately, it's not clearly defined by 3GPP how the BTS is supposed to map dBm values outside of the outer boundaries (`0` and `X5`) to band values. The ip.access nanoBTS, for example, would map values -120 dBm and -75 dBm to bands 1 and 5, respectively. osmo-bts replicates this behavior.

16.2 PDCH and dynamic timeslot handling

The BTS may optionally report interference levels for PDCH timeslots. This may be useful for the BSC to determine whether dynamic PDCH timeslots might be better used for new circuit switched connections, or whether alternative PDCH resources should be allocated for interference reasons.

Note
Currently osmo-bsc makes no use of PDCH interference reports, neither they get forwarded to the BSC co-located PCU over the PCUIF.

For dynamic timeslots (`DYNAMIC/OSMOCOM` and `DYNAMIC/IPACCESS`), the following expectations apply:

- when in TCH/F mode: no interference reports, because the only sub-channel is active;
- when in TCH/H mode: interference reports for **inactive** sub-channels only;
- when in SDCCH mode: interference reports for **inactive** sub-channels only;
- when in PDCH mode: optional interference reports;
  - measurements can be performed during IDLE TDMA frames.

17 CS Handover

Handover is the process of moving a continuously used channel (lchan) from one cell to another. Usually, that is an ongoing call, so that phones are able to move across cell coverage areas without interrupting the voice transmission.

A handover can

- stay within one given cell (intra-cell, i.e. simply a new RR Assignment Command);
- occur between two cells that belong to the same BSS (intra-BSC, via RR Handover Command);
- cross BSS boundaries (inter-BSC, via BSSMAP handover procedures);
- move to another MSC (inter-MSC, inter-PLMN);
- move to another RAN type, e.g. from 2G to 3G (inter-RAT, inter-Radio-Access-Technology).

The physical distance is by definition always very near, but handover negotiation may range from being invisible to the MSC all the way to orchestrating completely separate RAN stacks.

OsmoBSC currently supports handover within one BSS and between separate BSS. Whether inter-MSC is supported depends on the MSC implementation (to the BSC, inter-MSC handover looks identical to inter-BSC handover). Inter-RAT handover is currently not implemented. However, you may still advertise 3G and 4G neighbor cells in order to facilitate cell/RAT re-selection to those neighbors.

Since 2019, OsmoMSC fully supports both inter-BSC and inter-MSC handover.

Table 17: Handover support in Osmocom at the time of writing

	intra-BSC HO (local BSS)	inter-BSC HO (remote BSS)	inter-MSC HO	inter-RAT HO
OsmoBSC	rxlev, load-based	rxlev	(planned)	-
OsmoMSC	(not involved, except for codec changes)	(planned)	(planned)	-

Most handover related procedures are explained in 3GPP TS 48.008.

17.1 How Handover Works

This chapter generally explains handover operations between 2G cells.

17.1.1 Internal / Intra-BSC Handover

The BSS is configured to know which cell is physically adjacent to which other cells, its "neighbors". On the MS/BTS/BSS level, individual cells are identified by ARFCN+BSIC (frequency + 6-bit identification code).

The BSC instructs each BTS with a list of ARFCNs (i.e. GSM frequency bands) that qualify as neighbor cells, as part of the System Information Type 2. Each MS served by a BTS receives the System Information Type 2 and thus knows which ARFCNs to measure for potential handover. Each MS with an active channel then returns up to 6 measurements of reception levels (RXLEV) to the BTS, to be forwarded to the BSC in RSL Measurement Report messages.

Note that the BTS and MS are told only the ARFCNs, not the BSICs, of neighbor cells; the BSICs are however included in the measurements that an MS returns to BTS and BSC. Commonly, each ARFCN is owned by one specific operator, so, an MS considers all visible cells on a given ARFCN as possible neighbors. However, as soon as an MS reports RXLEV of a specific neighbor cell, the BSC needs to know which exact cell to possibly handover to, which is why the MS pinpoints the specific BSIC that it reported measurements for.

The BSC is the point of decision whether to do handover or not. This can be a hugely complex combination of heuristics, knowledge of cell load and codec capabilities. The most important indicator for handover though is: does an MS report a neighbor with a better signal than the current cell? See Figure 7.

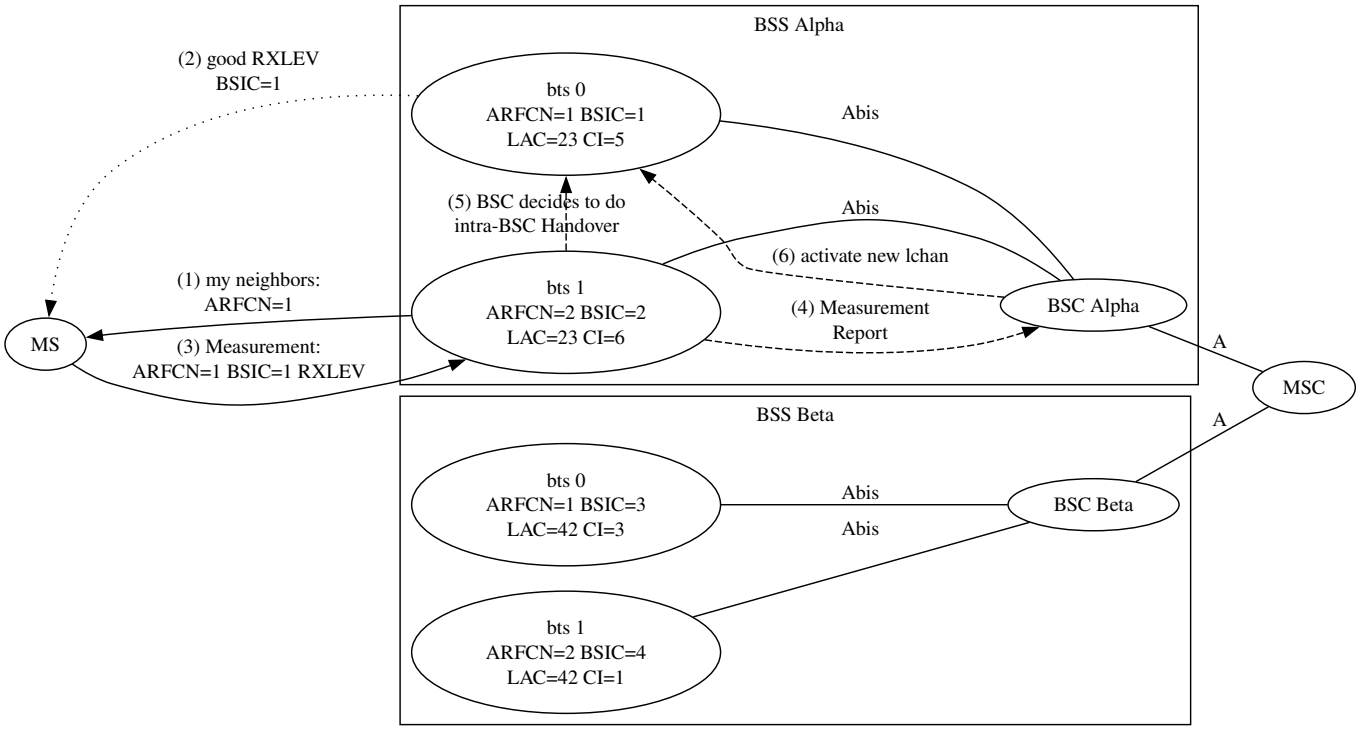


Figure 7: Intra-BSC Handover stays within the BSS (shows steps only up to activation of the new lchan — this would be followed by an RR Handover Command, RACH causing Handover Detection, Handover Complete, . . .)

If the BSC sees the need for handover, it will:

- activate a new lchan (with a handover reference ID),
- send an RR Handover Command to the current lchan, and
- wait for the MS to send a Handover RACH to the new lchan ("Handover Detect").
- The RTP stream then is switched over to the new lchan,
- an RSL Establish Indication is expected on the new lchan,

- and the old lchan is released.

Should handover fail at any point, e.g. the new lchan never receives a RACH, or the MS reports a Handover Failure, then the new lchan is simply released again, and the old lchan remains in use. If the RTP stream has already been switched over to the new lchan, it is switched back to the old lchan.

This is simple enough if the new cell is managed by the same BSC: the OsmoMGW is simply instructed to relay the BTS-side of the RTP stream to another IP address and port, and the BSC continues to forward DTAP to the MSC transparently. The operation happens completely within the BSS, except for the BSSMAP Handover Performed message sent to the MSC once the handover is completed (see 3GPP TS 48.008).

17.1.2 External / Inter-BSC Handover

If the handover target cell belongs to a different BSS, the RR procedure for handover remains the same, but we need to tell the *remote* BSC to allocate the new lchan.

The only way to reach the remote BSC is via the MSC, so the MSC must be able to:

- identify which other BSC we want to talk to,
- forward various BSSMAP Handover messages between old and new BSC,
- redirect the core-side RTP stream to the new BSS at the appropriate time,
- and must finally BSSMAP Clear the connection to the old BSS to conclude the inter-BSC handover.

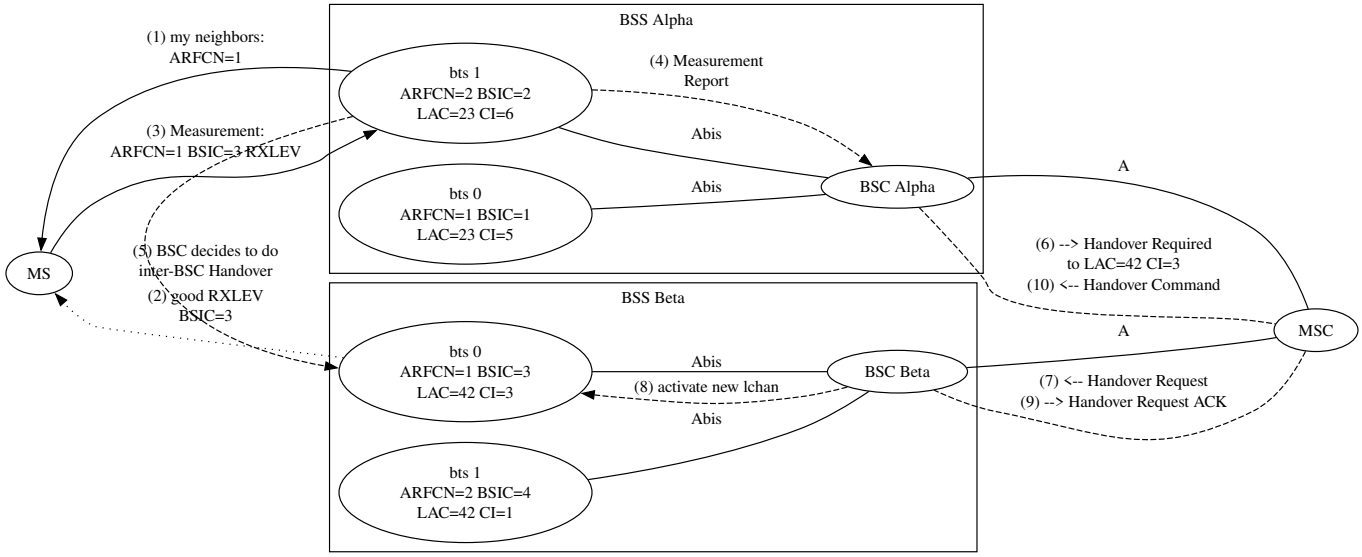


Figure 8: Inter-BSC Handover requires the MSC to relay between two BSCs (shows steps only up to the BSSMAP Handover Command — this would be followed by an RR Handover Command, RACH causing Handover Detection, Handover Complete, . . .)

The first part, identifying the remote BSC, is not as trivial as it sounds: as mentioned above, on the level of cell information seen by BTS and MS, the neighbor cells are identified by ARFCN+BSIC. However, on the A-interface and in the MSC, there is no knowledge of ARFCN+BSIC configurations. Instead, each cell is identified by a LAC and CI (Location Area Code and Cell Identifier).

Note
There are several different cell identification types on the A-interface: from Cell Global Identifier (MCC+MNC+LAC+CI) down to only LAC. OsmoBSC supports most of these (see Table 18). For simplicity, this description focuses on LAC+CI identification.

Hence:

- the BSC needs to know which remote-BSS cells' ARFCN+BSIC correspond to exactly which global LAC+CI, and
- the MSC needs to know which LAC+CI are managed by which BSC.

In other words, each BSC requires prior knowledge about the cell configuration of its remote-BSS neighbor cells, and the MSC requires prior knowledge about each BSC's cell identifiers; i.e. these config items are spread reduntantly.

The most obvious reason for using LAC+CI in BSSMAP is that identical ARFCN+BSIC are typically re-used across many cells of the same network operator: an operator will have only very few ARFCNs available, and the 6bit BSIC opens only a very limited range of distinction between cells. As long as each cell has no more than one neighbor per given ARFCN+BSIC, these values can be re-used any number of times across a network, and even between cells managed by one and the same BSC.

17.2 Configuring Neighbors

The most important step to enable handover in OsmoBSC is to configure each cell with the ARFCN+BSIC identities of its adjacent neighbors — both local-BSS and remote-BSS.

For a long time, OsmoBSC has offered configuration to manually enter the ARFCN+BSIC sent out as neighbors on various System Information messages (all `neighbor-list` related commands). This is still possible; however, particularly for re-using ARFCN+BSIC within one BSS, this method will not work well.

With the addition of inter-BSC handover support, the new `neighbor` config item has been added to the `bts` config node, to maintain explicit cell-to-cell neighbor relations, with the possibility to re-use ARFCN+BSIC in each cell.

It is recommended to completely replace `neighbor-list` configurations with the new `neighbor` configuration described below.

Table 18: Overview of neighbor configuration on the `bts` config node

Local	Remote BSS	type of `neighbor` config line, by example
✓		`neighbor bts 5`
✓		`neighbor lac 200`
✓		`neighbor lac-ci 200 3`
✓		`neighbor cgi 001 01 200 3`
✓	✓	`neighbor lac 200 arfcn 123 bsic 1`
✓	✓	`neighbor lac-ci 200 3 arfcn 123 bsic 1`
✓	✓	`neighbor cgi 001 01 200 3 arfcn 123 bsic 1`

17.2.1 Default: All Local Cells are Neighbors

For historical reasons, the default behavior of OsmoBSC is to add all local-BSS cells as neighbors for every other cell. To maintain a backwards compatible configuration file format, this is still the case: as long as no explicit neighbor cell is configured with a `neighbor` command (either none was configured, or all configured `neighbor` lines have been removed again), a cell automatically lists all of the local-BSS cells as neighbors. These are implicit mappings in terms of the legacy neighbor configuration scheme, and re-using ARFCN+BSIC combinations within a BSS will not work well this way.

As soon as the first explicit `neighbor` relation is added to a cell, the legacy behavior is switched off, and only explicit neighbors are in effect.

Note
If a cell is required to not have any neighbors, it is recommended to switch off handover for that cell with `handover 0`.

17.2.2 Local-BSS Neighbors

Local neighbors can be configured by just the local BTS number, or by LAC+CI, or any other supported A-interface type cell identification; also including the ARFCN+BSIC is optional, it will be derived from the local configuration if omitted.

OsmoBSC will log errors in case the configuration includes ambiguous ARFCN+BSIC relations (when one given cell has more than one neighbor for any one ARFCN+BSIC).

Neighbor relations must be configured explicitly in both directions, i.e. each cell has to name all of its neighbors, even if the other cell already has an identical neighbor relation in the reverse direction.

Example: configuring neighbors within the local BSS in osmo-bsc.cfg, identified by local BTS number

```
network
 bts 0
  neighbor bts 1
 bts 1
  neighbor bts 0
```

Example: configuring neighbors within the local BSS in osmo-bsc.cfg, identified by LAC+CI

```
network

 bts 0
  # this cell's LAC=0x0017 CI=5
  location_area_code 0x0017
  cell_identity 5
  # reference bts 1
  neighbor lac-ci 23 6

 bts 1
  # this cell's LAC=0x0017 CI=6
  location_area_code 0x0017
  cell_identity 6
  # reference bts 0
  neighbor lac-ci 23 5
```

It is allowed to include the ARFCN and BSIC of local neighbor cells, even though that is redundant with the already known local configuration of the target cell. The idea is to ease generating the neighbor configuration automatically, in that local-BSS and remote-BSS neighbors can have identical configuration formatting. If the cell identification (LAC+CI) matches a local cell but a mismatching ARFCN+BSIC follows on the same config line, OsmoBSC will report errors. For human readability and maintainability, it may instead be desirable to use the `neighbor bts <0-65535>` format, or omit the redundant `arfcn` and `bsic`.

Example: configuring neighbors within the local BSS in osmo-bsc.cfg, redundantly identified by LAC+CI as well as ARFCN+BSIC

```
network

 bts 0
  # this cell's LAC=0x0017 CI=5
  location_area_code 0x0017
  cell_identity 5
  # this cell's ARFCN=1 BSIC=1
  trx 0
   arfcn 1
  base_station_id_code 1
  # reference bts 1
  neighbor lac-ci 23 6 arfcn 2 bsic 2

 bts 1
  # LAC=0x0017 CI=6
  location_area_code 0x0017
```

```
  cell_identity 6
  # this cell's ARFCN=2 BSIC=2
  trx 0
   arfcn 2
  base_station_id_code 2
  # reference bts 0
  neighbor lac-ci 23 5 arfcn 1 bsic 1
```

17.2.3 Remote-BSS Neighbors

Remote-BSS neighbors always need to be configured with full A-interface identification *and* ARFCN+BSIC, to allow mapping a cell's neighbor ARFCN+BSIC to a BSSMAP Cell Identifier (see 3GPP TS 48.008 3.1.5.1 Handover Required Indication and 3.2.1.9 HANDOVER REQUIRED).

Example: configuring remote-BSS neighbors in osmo-bsc.cfg, identified by LAC+CI (showing both BSCs' configurations)

```
# BSC Alpha's osmo-bsc.cfg
network
 bts 0
  # this cell's LAC=0x0017 CI=6
  location_area_code 0x0017
  cell_identity 6
  # this cell's ARFCN=2 BSIC=2
  trx 0
   arfcn 2
  base_station_id_code 2
  # fully describe the remote cell by LAC+CI and ARFCN+BSIC
  neighbor lac-ci 42 3 arfcn 1 bsic 3

# BSC Beta's osmo-bsc.cfg
network
 bts 0
  # this cell's LAC=0x002A CI=3
  location_area_code 0x002A
  cell_identity 3
  # this cell's ARFCN=1 BSIC=3
  trx 0
   arfcn 1
  base_station_id_code 3
  # fully describe the remote cell by LAC+CI and ARFCN+BSIC
  neighbor lac-ci 23 6 arfcn 2 bsic 2
```

Note
It is strongly recommended to stick to a single format for remote-BSS neighbors' cell identifiers all across an OsmoBSC configuration; i.e. decide once to use `lac`, `lac-ci` or `cgi` and then stick to that within a given osmo-bsc.cfg. The reason is that the *Cell Identifier List* sent in the *BSSMAP Handover Required* message must have one single cell identifier type for all list items. Hence, to be able to send several alternative remote neighbors to the MSC, the configured cell identifiers must be of the same type. If in doubt, use the full CGI identifier everywhere.

17.2.4 Reconfiguring Neighbors in a Running OsmoBSC

When modifying a cell's neighbor configuration in a telnet VTY session while a cell is already active, the neighbor configuration will merely be cached in the BSC's local config. To take actual effect, it is necessary to

- either, re-connect the cell to the BSC (e.g. via `drop bts connection <0-65535> oml`)
- or, re-send the System Information using `bts <0-65535> resend-system-information`.

17.3 Configuring Handover Decisions

For a long time, OsmoBSC has supported handover based on reception level hysteresis (RXLEV) and distance (TA, Timing Advance), known as `algorithm 1`.

Since 2018, OsmoBSC also supports a load-based handover decision algorithm, known as `algorithm 2`, which also takes cell load, available codecs and oscillation into consideration. Algorithm 2 had actually been implemented for the legacy OsmoNITB program many years before the OsmoMSC split, but remained on a branch, until it was forward-ported to OsmoBSC in 2018.

Table 19: What handover decision algorithms take into account

algorithm 1	algorithm 2	
✓	✓	RXLEV
✓	✓	RXQUAL
✓	✓	TA (distance)
✓	✓	interference (good RXLEV, bad RXQUAL)
	✓	load (nr of free lchans, minimum RXLEV and RXQUAL)
	✓	penalty time to avoid oscillation
	✓	voice rate / codec bias
✓		inter-BSC: RXLEV hysteresis
	✓	inter-BSC: only below minimum RXLEV, RXQUAL

17.3.1 Common Configuration

Handover is disabled by default; to disable/enable handover, use `handover (0|1)`.

Once enabled, algorithm 1 is used by default; choose a handover algorithm with `handover algorithm (1|2)`:

```
network
 # Enable handover
 handover 1

 # Choose algorithm
 handover algorithm 2

 # Tweak parameters for algorithm 2 (optional)
 handover2 min-free-slots tch/f 4
 handover2 penalty-time failed-ho 30
 handover2 retries 1
```

All handover algorithms share a common configuration scheme, with an overlay of three levels:

- immutable compile-time default values,
- configuration on the `network` level for all cells,
- individual cells' configuration on each `bts` node.

Configuration settings relevant for algorithm 1 start with `handover1`, for algorithm 2 with `handover2`.

The following example overrides the compile-time default for all cells, and furthermore sets one particular cell on its own individual setting, for the `min-free-slots tch/f` value:

```
network
 handover2 min-free-slots tch/f 4
 bts 23
  handover2 min-free-slots tch/f 2
```

The order in which these settings are issued makes no difference for the overlay; i.e., the following configuration is perfectly identical to the above, and the individual cell's value remains in force:

```
network
 bts 23
  handover2 min-free-slots tch/f 2
 handover2 min-free-slots tch/f 4
```

Each setting can be reset to a default value with the `default` keyword. When resetting an individual cell's value, the globally configured value is used. When resetting the global value, the compile-time default is used (unless individual cells still have explicit values configured). For example, this telnet VTY session removes above configuration first from the cell, then from the global level:

```
OsmoBSC(config)# network
OsmoBSC(config-net)# bts 23
OsmoBSC(config-net-bts)# handover2 min-free-slots tch/f default
% 'handover2 min-free-slots tch/f' setting removed, now is 4
OsmoBSC(config-net-bts)# exit
OsmoBSC(config-net)# handover2 min-free-slots tch/f default
% 'handover2 min-free-slots tch/f' setting removed, now is 0
```

17.3.2 Handover Algorithm 1

Algorithm 1 takes action only when RR Measurement Reports are received from a BTS. As soon as a neighbor's average RXLEV is higher than the current cell's average RXLEV plus a hysteresis distance, handover is triggered.

If a handover fails, algorithm 1 will again attempt handover to the same cell with the next Measurement Report received.

Configuration settings relevant for algorithm 1 start with `handover1`. For further details, please refer to the OsmoBSC VTY Reference ([vty-ref-osmobsc]) or the telnet VTY online documentation. See the `handover1` settings on the `config-net` and `config-net-bts` nodes.

17.3.3 Handover Algorithm 2

Algorithm 2 is specifically designed to distribute load across cells. A subscriber will not necessarily remain attached to the cell that has the best RXLEV average, if that cell is heavily loaded and a less loaded neighbor is above the minimum allowed RXLEV.

Algorithm 2 also features penalty timers to avoid oscillation: for each subscriber, if handover to a specific neighbor failed (for a configurable number of retries), a holdoff timer prevents repeated attempts to handover to that same neighbor. Several hold-off timeouts following specific situations are configurable (see `handover2 penalty-time` configuration items).

Configuration settings relevant for algorithm 2 start with `handover2`. For further details, please refer to the OsmoBSC VTY Reference [vty-ref-osmobsc] or the telnet VTY online documentation. See the `handover2` settings on the `config-net` and `config-net-bts` nodes.

17.3.3.1 Load Distribution

Load distribution is only supported by algorithm 2.

Load distribution occurs:

- explicitly: every N seconds, OsmoBSC considers all local cells and actively triggers handover operations to reduce congestion, if any. See `min-free-slots` below, and the `congestion-check` setting.
- implicitly: when choosing the best neighbor candidate for a handover triggered otherwise, a congested cell (in terms of `min-free-slots`) is only used as handover target if there is no alternative that causes less cell load.

In either case, load distribution will only occur towards neighbor cells that adhere to minimum reception levels and distance, see `min rxlev` and `max distance`.

Load distribution will take effect only for already established channels. For example, an MS will always first establish a voice call with its current cell choice; in load situations, it might be moved to another cell shortly after that. Considering the best neighbor *before* starting a new voice call might be desirable, but is currently not implemented. Consider that RXLEV/RXQUAL ratings are averaged over a given number of measurement reports, so that the neighbor ratings may not be valid/reliable yet during early call establishment. In consequence, it is recommended to ensure a sufficient number of unused logical channels at all times, though there is no single correct configuration for all situations.

Most important for load distribution are the `min-free-slots tch/f` and `min-free-slots tch/h` settings. The default is zero, meaning *no* load distribution. To enable, set `min-free-slots` >= 1 for `tch/f` and/or `tch/h` as appropriate. This setting refers to the minimum number of voice channels that should ideally remain unused in each individual BTS at all times.

Note
it is not harmful to configure `min-free-slots` for a TCH kind that is not actually present. Such settings will simply be ignored.

Note
the number of TCH/F timeslots corresponds 1:1 to the number indicated by `min-free-slots tch/f`, because each TCH/F physical channel has exactly one logical channel. In contrast, for each TCH/H timeslot, there are two logical channels, hence `min-free-slots tch/h` corresponds to twice the number of TCH/H timeslots configured per cell. In fact, a more accurate naming would have been "min-free-lchans".

Think of the `min-free-slots` setting as the threshold at which load distribution is considered. If as many logical channels as required by this setting are available in a given cell, only changes in RXLEV/RXQUAL/TA trigger handover away from that cell. As soon as less logical channels remain free, the periodical congestion check attempts to distribute MS to less loaded neighbor cells. Every time, the one MS that will suffer the least RXLEV loss while still reducing congestion will be instructed to move first.

If a cell and its neighbors are all loaded past their `min-free-slots` settings, the algorithmic aim is to improve the percentage of load above the `min-free-slots` setting: a load-based handover always requires the target cell to have a lower load percentage after handover than the source cell had before handover.

The min-free-slots setting is a tradeoff between immediate voice service availability and optimal reception levels. A sane choice could be:

- Start off with `min-free-slots` set to half the available logical channels.
- Increase `min-free-slots` if you see MS being rejected too often even though close neighbors had unused logical channels.
- Decrease `min-free-slots` if you see too many handovers happening for no apparent reason.

Choosing the optimal setting is not trivial, consider these examples:

- Configure `min-free-slots` = 1: load distribution to other cells will occur exactly when the last available logical channel has become occupied. The next time the congestion check runs, at most one handover will occur, so that one channel is available again. In the intermediate time, all channels will be occupied, and some MS might be denied immediate voice service because of that, even though, possibly, other neighbor cells would have provided excellent reception levels and were completely unloaded. For those MS that are already in an ongoing voice call and are not physically moving, though, this almost guarantees service by the closest/best cell.
- Set `min-free-slots` = 2: up to two MS can successfully request voice service simultaneously (e.g. one MS is establishing a new voice call while another MS is travelling into this cell). Ideally, two slots have been kept open and are immediately available. But if a third MS is also traveling into this cell at the same time, it will not be able to handover into this cell until load distribution has again taken action to make logical channels available. The same scenario applies to any arbitrary number of MS asking for voice channels simultaneously. The operator needs to choose where to draw the line.

- Set `min-free-slots` >= the number of available channels: as soon as any neighbor is less loaded than a given cell, handover will be attempted. But imagine there are only two active voice calls on this cell with plenty of logical channels still unused, and the closest neighbor rates only just above `min rxlev`; then moving one of the MS *for no good reason* causes all of: increased power consumption, reduced reception stability and channel management overhead.

Note
In the presence of dynamic timeslots to provide GPRS service, the number of voice timeslots left unused also determines the amount of bandwidth available for GPRS.

17.3.4 External / Inter-BSC Handover Considerations

There currently is a profound difference for inter-BSC handover between algorithm 1 and 2:

For algorithm 1, inter-BSC handover is triggered as soon as the Measurement Reports and hysteresis indicate a better neighbor than the current cell, period.

For algorithm 2, a subscriber is "sticky" to the current BSS, and inter-BSC handover is only even considered when RXLEV/TA drop below minimum requirements.

- If your network topology is such that each OsmoBSC instance manages a single BTS, and you would like to encourage handover between these, choose handover algorithm 1. Load balancing will not be available, but RXLEV hysteresis will.
- If your network topology has many cells per BSS, and/or if your BSS boundaries in tendency correspond to physical/semantic boundaries that favor handover to remain within a BSS, then choose handover algorithm 2.

The reason for the difference between algorithm 1 and 2 for remote-BSS handovers is, in summary, the young age of the inter-BSC handover feature in OsmoBSC:

- So far the mechanisms to communicate cell load to remote-BSS available in the BSSMAP Handover messages are not implemented, so, a handover due to cell load across BSS boundaries would be likely to cause handover oscillation between the two BSS (continuous handover of the same MS back and forth between the same two cells).
- Algorithm 1 has no `min rxlev` setting.
- Algorithm 1 does not actually use any information besides the Measurement Reports, and hence can trivially treat all neighbor cells identically.

17.4 Advertising 3G/4G neighbors

Despite osmo-bsc not supporting inter-RAT hand-over at this point, it still makes sense to advertise neighbor cells of other network technologies like UMTS/UTRAN (3G) and LTE/EUTRAN (4G). This will help phones with idle-mode re-selection of the best available radio access technology (RAT).

For more information on the inter-RAT cell re-selection algorithm and its parameters, see 3GPP TS 45.008 - particularly Section 6.6.4 describing measurements on cells of other (non-GSM) RATs.

Such neighbors are advertised as part of the SI2quater (System Information Type 2quater).

17.4.1 UMTS/UTRAN/3G neighbors

In order to advertise a 3G neighbor cell you have to specify the following properties:

- the UARFCN (UTRAN Absolute Radio Channel Number) on which the cell broadcasts
- the Scrambling Code of the cell
- whether or not the cell uses diversity

In the following example, we're configuring a 3G neighbor cell on UARFCN 1234 using the scrambling code 511 with no diversity:

```
network
 bts 0
  si2quater neighbor-list add uarfcn 1234 511 0
```

3G neighbor cells can be removed using the same command, just replacing `add` with `del`.

17.4.2 LTE/EUTRAN/4G neighbors

In order to advertise a 4G neighbor cell you have to specify the following properties:

- EARFCN (EUTRAN Absolute Radio Channel Number) on which the cell broadcasts
- Reselection thresholds towards E-UTRAN cells:

0	0 dB
1	2 dB
2	4 dB
3	6 dB
...	...
31	62 dB

- Priority of E-UTRAN frequency: 0 = lowest priority, ..., 7 = highest priority
- QRXLEVMIN parameter: Minimum required RX level in the UTRAN FDD cell (dBm), see 3GPP TS 25.304.

0	-140 dBm
1	-138 dBm
2	-136 dBm
...	...
31	-78 dBm

- Measurement bandwidth in MHz, see 3GPP TS 44.018 and 3GPP TS 44.060. This field specifies the minimum value of the channel bandwidth of all valid E-UTRAN cells on the specified EARFCN. It is defined by the parameter Transmission Bandwidth Configuration, N RB (see 3GPP TS 36.104). The values indicate the number of resource blocks over which the mobile station could measure if the mobile station does not support wideband RSRQ measurements (see 3GPP TS 24.008). A mobile station supporting wideband RSRQ measurements shall measure over the indicated number of resource blocks. The field is coded according to the following table:

0	N_RB = 6
1	N_RB = 15
2	N_RB = 25
3	N_RB = 50
4	N_RB = 75
5	N_RB = 100

In the following example we're configuring a 4G neighbor on EARFCN 2342 with a higher reselection threshold of 40dB, a lower reselection threshold of 20dB, priority 5, QRXLEVMIN of -140 dBm and a measurement bandwidth of 100 resource blocks:

```
network
 bts 0
  si2quater neighbor-list add earfcn 2342 thresh-hi 20 thresh-lo 10 prio 5 qrxlv 0 meas 5
```

4G neighbor cells can be removed using the same command, just replacing `add` with `del`.

18 PS Handover

Packet Switch Handover is mostly managed by the packet switching nodes such as PCU, SGSN and GGSN. However, the PCU encounters a similar difficulty to that explained in Section 17: that is, it must find a way to translate ARFCN+BSIC identifiers provided by the MS into the sort of identifiers understood by the Core Network (SGSN). These identifiers are in this case composed of RAC+CI (instead LAC+CI), or more specifically, CGI+RAC (named as "Packet Switched CGI" or "CGI-PS" from now on for convenience).

Hence, it feels natural extending the Section 17.2 storage in the BSC to also provide Packet Switched related identifiers in order to provide the PCU the required information to do the translations, instead of duplicating the whole neighbor information in two different network nodes.

This can be done, similarly to already presented CS related commands in Section 17.2, by using the `neighbor cgi-ps` VTY command, which allows for specifying the extra identifier (RAC) required by PS related translations:

Example: configuring PS neighbors within the local BSS in osmo-bsc.cfg, identified by CGI-PS (CGI+RAC)

```
network
 bts 0
neighbor cgi-ps <MCC> <MNC> <LAC> <RAC> <CI> arfcn <0-1023> bsic (<0-63>|any)
```

This information should already be present by default if one uses the `local BTS number` to identify the network, as long as that BTS is properly configured to have GPRS enabled and the RAC code is set by `gprs routing area <0-255>` command:

Example: configuring PS neighbors within the local BSS in osmo-bsc.cfg, identified by local BTS number

```
network
 bts 0
  gprs mode gprs
  gprs routing area 45
  neighbor bts 1
  ...
 bts 1
  gprs mode egprs
  gprs routing area 48
  neighbor bts 0
  ...
```

This PS information is solely used by the Neighbor Resolution Service, aimed at nodes other than BSC itself, and described below.

18.1 Neighbor Address Resolution Service

This service is provided in order to provide the PCU with a way to translate ARCFCN+BSIC into CGI-PS in order to use RIM services and accomplish PS Handovers.

This interface is Osmocom specific, since the standard doesn't provide any specifications on how to provide this kind of information to the PCU.

Since the PCU can be either BSC co-located or BTS co-located (hence on a different host than BSC), messages may need to be sent over the network at some point.

In consequence, it was decided implement a new *Container* type message in PCUIF which the PCU can use to directly communicate with the BSC. If the PCU is BSC co-located, then the BSC can directly communicate over the unix socket. If the PCU is BTS co-located, then PCU sends PCUIF messages over the unix socket to the BTS, and the BTS forwards the PCUIF messages transparently over the the IPA multiplex of the OML connection towards the BSC and back.

Support for this interface is available by default nowadays in OsmoPCU, OsmoBTS and OsmoBSC, and requires no specific configuration, it will just work out of the box.

18.1.1 Neighbor Address Resolution Service CTRL interface (deprecated)

Before the existence of the PCUIF forwarded over IPA multiplex interface (see above), the Neighbor Address Resolution Service was implemented by means of a CTRL interface, similar to the one used in most Osmocom processes (see Section 30).

Caution
This interface is nowadays considered deprecated and should not be used anymore. Any related VTY options should be dropped from configuration files, to let OsmoPCU use the new interface instead. This section is kept here for a while as a reference for old deployments using old versions of the programs.

Due to security concerns, the set of CTRL commands available in this service is configured in a different IP address and port, since the service needs to be reachable by all PCU under the BSC. This way the user can still constrain the regular CTRL port (which may contains lots of configuration related commands) listening in a more constrained address (eg. localhost).

By default, this interface is disabled, and it is enabled by configuring the desired IP address to bind to (and optionally a different port other than well-known `4248`):

Example: Enable and configure Neighbor Resolution Service CTRL interface

```
network
 neighbor-resolution bind 127.0.0.1 5000
```

osmo-pcu will then be able to connect to the BSC and query for resolution during eg. NACC requests from MS.

The relevant commands are

- neighbor_resolve_cgi_ps_from_lac_ci:

```
GET neighbor_resolve_cgi_ps_from_lac_ci.<src_lac>.<src_cell_id>.<dst_arfcn>.<dst_bsic>
GET_REPLY <MCC>-<MNC>-<LAC>-<RAC>-<CI>
```

Where the `src_lac` and `src_ci` are provided by BTS/BSC over `PCUIF` interface during startup (`INFO IND`), and `dst_arfcn` and `dst_bsic` are those of the neighbor the PCU is willing to request the CGI-PS information. Hence, from the `src` params the BSC is able to look up the BTS and afterwards apply the translation of `dst` parameters based on the neighbor information available for that BTS.

19 SMSCB (Cell Broadcast)

OsmoBSC supports SMS Cell Broadcast (SMSCB) services (CBS). This includes the CBSP protocol to interact with a CBC (Cell Broadcast Centre) such as OsmoCBC, as well as the scheduling of SMSCB messages on both the BASIC and EXTENDED CBCH and transmission of related RSL messages to the attached BTS.

More high-level information can be found at https://en.wikipedia.org/wiki/Cell_Broadcast and the related specification is [3gpp-ts-23-041].

In order to use SMSCB with OsmoBSC, you will need to

- Configure OsmoBSC as either CBSP server or client
- Use a channel combination including a CBCH on the BTSs

19.1 Enabling a CBCH channel combination

On the Um interface, SMSCB are transmitted via the CBCH (Cell Broadcast Channel). The CBCH is a separate downlink-only logical channel which must be activated on any of the BTSs requiring CBSP support.

The channel combination is configured in the `timeslot` node of each TRX.

The two `phys_chan_config` supporting CBCH are `CCCH+SDCCH4+CBCH` and `SDCCH/8+CBCH`. Please note that the CBCH steals one of the SDCCH, so a SDCCH/4 will only have three remaining SDCCH, and a SDCCH/8 will have only seven remaining SDCCH.

19.2 Configuring the CBSP connection

CBSP is the protocol between BSC and CBC. It operates over TCP.

According to 3GPP TS 48.049, a BSC typically operates as a TCP server, and the CBC connects as TCP client. This would require the CBC to have out-of-band knowledge of all the BSCs in the network (and their IP addresses).

In order to comply with the specifications, OsmoBSC supports this mode of operation as CBSP TCP server. However, to make network operation and configuration more simple, it also can operate in TCP client mode, connecting to the CBC. This way the BSCs need to know the CBC IP address, but not vice-versa.

The BSC can operate in either CBSP TCP server mode or CBSP TCP client mode.

The CBC related configuration of OsmoBSC can be found in the `cbc` configuration node of the VTY interface.

The default port number for the CBSP server is 48049, according to the CBSP specification. Hence it normally suffices to configure only the IP addresses for the remote CBC server or the local CBSP server:

Example: Configure CBSP TCP client to connect to CBC at 1.2.3.4:48049 in osmo-bsc.cfg

```
cbc
 mode client
 client
  remote-ip 1.2.3.4
```

In server mode, the default configuration is 127.0.0.1:48049, so it suffices to set `mode server` to accept CBSP connections from localhost:

```
cbc
 mode server
```

To also listen for inbound CBSP connections on all interfaces, both IPv4 and IPv6:

Example: Configure CBSP TCP server to listen on all interfaces in osmo-bsc.cfg

```
cbc
 mode server
 server
  local-ip ::
```

Should non-standard port numbers be required, these can be configured with the `client` / `local-port` or the `server` / `remote-port` settings.

The `client` config also supports an explicit local bind for connecting to the remote CBC, using `client` / `local-ip` and `local-port`.

IP addresses for client and server can remain configured at the same time, and the `mode` command can be used to switch between client and server operation. The `mode` command takes immediate effect, no restart of OsmoBSC is required. After changing `cbc` IP addresses in the telnet VTY, it is required to switch `mode` to `disabled` and back to `client` or `server` to take effect.

Example: Disable the CBSP link in the telnet VTY

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# cbc
OsmoBSC(config-cbc)# mode disabled
OsmoBSC(config-cbc)# end
```

For more details on the available configuration commands, please check the OsmoBSC VTY Reference.

19.3 Counters

OsmoBSC has two Cell Broadcast related rate counter groups for each BTS:

- the basic CBCH ("cell broadcast channel 0")
- the extended CBCH ("cell broadcast channel 1")

See below example for a situation where no CBS messages were received yet from the CBC, and hence only NULL messages have been sent on both basic and extended CBCH:

Example: CBCH related rate counters on a BTS that didn't receive any CBS messages

```
cell broadcast channel 1:
         cbch:rcvd_queued:          0 (0/s 0/m 0/h 0/d) Received + queued CBCH messages ( ↩
             Abis)
        cbch:rcvd_dropped:          0 (0/s 0/m 0/h 0/d) Received + dropped CBCH messages ( ↩
            Abis)
         cbch:sent_single:          0 (0/s 0/m 0/h 0/d) Sent single CBCH messages (Um)
        cbch:sent_default:          0 (0/s 0/m 0/h 0/d) Sent default CBCH messages (Um)
           cbch:sent_null:      31366 (0/s 45/m 2677/h 30588/d) Sent NULL CBCH messages (Um ↩
               )
cell broadcast channel 0:
         cbch:rcvd_queued:          0 (0/s 0/m 0/h 0/d) Received + queued CBCH messages ( ↩
             Abis)
        cbch:rcvd_dropped:          0 (0/s 0/m 0/h 0/d) Received + dropped CBCH messages ( ↩
            Abis)
         cbch:sent_single:          0 (0/s 0/m 0/h 0/d) Sent single CBCH messages (Um)
        cbch:sent_default:          0 (0/s 0/m 0/h 0/d) Sent default CBCH messages (Um)
           cbch:sent_null:      31366 (1/s 46/m 2676/h 30588/d) Sent NULL CBCH messages (Um ↩
               )
```

20 MSC Pooling

MSC pooling is described in 3GPP TS 23.236 [3gpp-ts-23-236], and is supported by OsmoBSC since mid 2020.

The aim of MSC pooling is to distribute load from a BSC across multiple MSCs, which are equivalent and redundant infrastructure for the same core network.

The main mechanism for MSC pooling is the TMSI identity, which an MSC hands out to its attached subscribers. Typically 10 bits of the TMSI are designated as a Network Resource Identifier (NRI) that identifies the originating MSC, and allows OsmoBSC to direct a subscriber back to the same MSC instance that previously negotiated the IMSI Attach procedure. Typically, the full NRI value range available is divided into N even ranges, where each MSC is assigned one NRI range.

Subscribers attaching without a TMSI identity, or those with unknown NRI value, are evenly distributed across MSC instances. OsmoBSC uses a round-robin approach to distribute load across all connected MSCs.

A Paging Response from a subscriber is always returned back to whichever MSC initiated the Paging, regardless of the Mobile Identity used.

Finally, a NULL-NRI is a special NRI value that indicates that the MSC wishes to offload this subscriber to a different MSC. A NULL-NRI is an arbitrary NRI value that is chosen distinctly for each PLMN served by a BSC, so that a subscriber can be reassigned within that PLMN. Upon (periodic) Location Updating, an offloading MSC hands out a NULL-NRI value in the assigned TMSI, along with a non-broadcast LAI. The subscriber will notice the LAI mismatch, and immediately re-attempt the attach using the TMSI containing the NULL-NRI. The BSC recognises the NULL-NRI and redirects the subscriber to one of the other MSCs. A prerequisite for this to work well is that the particular MSC is previously marked as not accepting new subscribers, in the BSC's configuration.

The mechanisms described above make up the NAS node selection function implemented in the BSC.

3GPP TS 23.236 also defines that an offloading MSC hands subscriber information to the newly assigned MSC, which takes place outside the scope of the BSC.

20.1 Configuring MSC Pooling

The NRI ranges assigned to each MSC must match in the BSC and the MSC configuration. If MSC and BSC had inconsistent NRI value ranges configured, attached subscribers would be redirected MSC instances that did not perform the attach, possibly rendering the core network unusable.

20.1.1 Connecting Multiple MSCs

The `cs7 instance` configuration defines the SCCP addresses to reach the MSCs at. In addition, each MSC is configured by its own `msc` section in the configuration. An example osmo-bsc.cfg serving three MSCs:

```
cs7 instance 0
 # SCCP address book entries for the three MSCs
 sccp-address my-msc-0
  point-code 0.23.0
 sccp-address my-msc-1
  point-code 0.23.1
 sccp-address my-msc-2
  point-code 0.23.2

# assign each MSC configuration its remote SCCP address
msc 0
 msc-addr my-msc-0
msc 1
 msc-addr my-msc-1
msc 2
 msc-addr my-msc-2

# configure NRI value ranges
network
 nri bitlen 10
 nri null add 0
msc 0
 nri add 1 341
msc 1
 nri add 342 682
msc 2
 nri add 683 1023
```

20.1.2 NRI Value Bit Length

In OsmoBSC, the NRI value's bit length is freely configurable from 1 to 15 bits. 3GPP TS 23.236 suggests a typical bit length of 10, which is OsmoBSC's default. The NRI bit length must be identical across the entire MSC pool.

Change the NRI value bit length in OsmoBSC's VTY configuration like this:

```
network
 nri bitlen 10
```

In the TMSI bits, regardless of the NRI bit length, the NRI value always starts just after the most significant octet of a TMSI (most significant bit at TMSI's bit 23).

20.1.3 NULL-NRI

Since OsmoBSC supports serving only one PLMN, NULL-NRI are configured globally. Even though 3GPP TS 23.236 indicates that there is a single NULL-NRI per PLMN, OsmoBSC allows configuring multiple NULL-NRI values.

```
network
 nri null add 0
 nri null add 423
```

20.1.4 Assigning NRI Ranges to MSCs

Each MSC configured in OsmoBSC must be assigned a distinct NRI value range. Overlapping NRI value ranges will cause failure to serve subscribers.

NRI values are typically configured in ranges, here dividing a 10bit range (0..1023) into three equal ranges, while leaving 0 available to be configured as NULL-NRI:

```
msc 0
 nri add 1 341
msc 1
 nri add 342 684
msc 2
 nri add 685 1023
```

NRI can also be assigned in single values:

```
msc 0
 nri add 23
```

Ranges can be constructed arbitrarily by a sequence of `add` and `del` configurations, here a contrived example:

```
msc 0
 nri add 0 342
 nri del 23
 nri del 42 235
 nri add 1000 1023
```

To view the current NRI config in a running OsmoBSC instance, use the `show nri` command, here showing the result of the contrived example:

```
OsmoBSC(config-msc)# show nri
msc 0
 nri add 0 22
 nri add 24 41
 nri add 236 342
 nri add 1000 1023
```

On the VIEW and ENABLE VTY nodes, `show nri` shows all MSCs:

```
OsmoBSC> show nri
msc 0
 nri add 1 341
msc 1
 nri add 342 684
msc 2
 nri add 685 1023
```

When configuring overlapping NRI value ranges across MSCs, the telnet VTY warns about it, and starting OsmoBSC with such a configuration will fail:

```
msc 0
 nri add 1 511
msc 1
 nri add 512 1023
msc 2
 nri add 500 555
```

This results in:

```
$ osmo-bsc
DMSC ERROR msc 2: NRI range [500..555] overlaps between msc 2 and msc 0. For overlaps, msc ↩
    0 has higher priority than msc 2
DMSC ERROR msc 2: NRI range [500..555] overlaps between msc 2 and msc 1. For overlaps, msc ↩
    1 has higher priority than msc 2
```

20.1.5 MSC Offloading

To effectively offload a particular MSC, it must be marked as no longer taking new subscribers in OsmoBSC. This can be achieved in the telnet VTY by:

```
msc 0
 no allow-attach
```

This MSC will, as long as it is connected, continue to serve subscribers already attached to it: those that yield an NRI matching this MSC, and those that are being paged by this MSC. But OsmoBSC will no longer direct new subscribers to this MSC.

To re-enable an MSC for attaching new subscribers:

```
msc 0
 allow-attach
```

21 MGW Pooling

OsmoBSC is able to use a pool of media gateway (MGW) instances. The aim of MGW pooling is to evenly distribute the RTP voice stream load across multiple MGW instances. This can help to scale out over multiple VMs or physical machines. Until osmo-mgw includes multithreading support, it may also be used to scale-out to multiple cores on a single host.

The load distribution is managed in such a way that when a new call is placed, the pool will automatically assign the call to the available MGW with the lowest load.

MGW pooling is recommended for larger RAN or CN installations. For small networks and lab installations the classic method with one MGW per OsmoBSC offers sufficient performance.

21.1 MGW pool VTY configuration

In OsmoBSC the MGW is controlled via an MGCP-Client. The VTY commands to configure the MGCP-Client are part of the several *mgw* nodes, one per MGCP-Client to set up.

To setup an MGW pool, the user must first install multiple OsmoMGW instances, so that they won't interfere with each other. This can be done using different local host IP addresses or different ports. When OsmoMGW is installed from packages, the systemd configuration may also require adjustment.

By default, MGCP-Client will use whatever source IP address is resolved by the kernel routing subsystem, and will also ask the kernel to pick a free UDP port.

Example configuration with two MGCP-Client instances in a pool:

```
 mgw 0
  description media-gw-0 ❶
  remote-ip 127.0.0.1
  remote-port 2432
  local-ip 127.0.0.1
  local-port 2431
  endpoint-domain mgw0 ❷
 mgw 1
  description media-gw-1 ❸
```

```
  remote-ip 127.0.0.1
  remote-port 2430
  local-ip 127.0.0.1
  local-port 2429
  endpoint-domain mgw1 ❹
```

❷, ❹ When working with multiple MGW / MGCP-Client instances, the domain name for each MGW should be different. Otherwise it won't be possible to distinguish the endpoint names in the log. It should also be noted that the domain name must match the configuration of the related OsmoMGW instance.

❶, ❸ It is also possible to assign a descriptive name to each MGW instance. The MGCP client specific log lines will then use this name as logging context. If no description is set, the domain name will be used.

21.2 MGW pool management

The MGW pool is fully runtime-manageable. The only limitation is that an MGCP-Client can not be restarted or removed as long as it is serving calls (see also: Section 21.2.5).

21.2.1 MGW pool status

The VTY implements a *show mgw-pool* command that lists the currently configured MGW pool members, their status and call utilization. The following snippet shows an output example run on OsmoBSC, but it should be also available on other applications supporting the MGW pooling VTY features:

```
OsmoBSC> show mgw-pool
% MGW-Pool:
%  MGW 0:media-gw-0
%   MGCP link:     connected,UP
%   service:       unblocked
%   ongoing calls: 1
%  MGW 1:media-gw-1
%   MGCP link:     connected,UP
%   service:       unblocked
%   ongoing calls: 0
```

21.2.2 Adding an MGW / MGCP-Client to the MGW pool

To add a new MGCP-Client to the pool, the *mgw* node is used. Like with the *bts* or the *msc* node a reference number is used that usually starts at 0. However it is still possible to assign any number from 0-255. The enumeration also may contain gaps. The following snippet shows an output example run on OsmoBSC, but it should be also available on other applications supporting the MGW pooling VTY features:

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# network
OsmoBSC(config-net)# mgw 2
OsmoBSC(config-mgw)# ?
  keepalive        Monitor if the MGCP link against MGW is still usable
  local-ip         local bind to connect to MGW from
  local-port       local port to connect to MGW from
  remote-ip        remote IP address to reach the MGW at
  remote-port      remote port to reach the MGW at
  endpoint-domain  Set the domain name to send in MGCP messages, e.g. the part 'foo' in ' ↩
      rtpbridge/*@foo'.
  reset-endpoint   Add an endpoint name that should be reset (DLCX) on connect to the reset ↩
      -endpoint list,e.g. 'rtpbridge/*'
```

The newly added MGW will immediately appear in the mgw-pool list but it won't be used until its configuration finished by reconnecting it.

```
% MGW-Pool:
%  MGW 0:media-gw-0
%   MGCP link:     connected,UP
%   service:       unblocked
%   ongoing calls: 2
%  MGW 1:media-gw-1
%   MGCP link:     connected,UP
%   service:       unblocked
%   ongoing calls: 3
%  MGW 2:mgw ❶
%   MGCP link:     disconnected,DOWN
%   service:       unblocked
%   ongoing calls: 0
```

❶ In this example a description is not set yet, so the domain name ("mgw") is displayed.

21.2.3 Reconnecting an MGW / MGCP-Client

It may become necessary to reconnect an MGCP-Client. This is the case when the VTY configuration was changed at runtime. In order to make the changes effective the MGW configuration must be reloaded by reconnecting the MGW connection. Also newly created MGW instances require a reconnect once their configuration is done.

To reconnect an MGCP-Client use the *reconnect* VTY command:

```
OsmoBSC# mgw 2 reconnect
```

The mgcp-client status should immediately change to *connected*. The MGW is now ready to be used for new calls.

```
OsmoBSC# show mgw-pool
% MGW-Pool:
%  MGW 0:media-gw-0
%   MGCP link:     connected,UP
%   service:       unblocked
%   ongoing calls: 2
%  MGW 1:media-gw-1
%   MGCP link:     connected,UP
%   service:       unblocked
%   ongoing calls: 3
%  MGW 2:mgw
%   MGCP link:     connected,UP
%   service:       unblocked
%   ongoing calls: 0
```

It should be noted that MGCP a protocol is used via UDP, the connect only happens locally to forward the UDP datagrams properly (state printed in `mgcp-client: (dis)connected` above). Also (unless a reset endpoint is configured like in the example config above) there will be no immediate interaction with the MGW. However, the log should at least confirm the connect worked and the MGCP client has been created successfully.

```
Mon Aug  2 17:15:00 2021 DLMGCP mgcp_client.c:788 MGCP client: using endpoint domain '@mgw'
Mon Aug  2 17:15:00 2021 DLMGCP mgcp_client.c:908 MGCP GW connection: r=127.0.0.1:2427<->l ↩
    =127.0.0.1:2727
```

For that reason, it is strongly advised to enable the `keepalive` feature in OsmoBSC to schedule periodical MGCP queries against the MGW, in order to make sure it is reachable (the state `MGCP link: UP|DOWN` printed above). See section Section 21.2.4 below for more information.

21.2.4 Monitor MGCP link (keepalive) / MGCP-Client

The `keepalive` feature in OsmoBSC allows scheduling periodical queries on the MGCP layer in order to make sure it is reachable and hence obtain information on the state of the MGCP link. This is in turn used by the MGW Pool when picking an MGW from the pool: MGWs whose link is considered to be DOWN are skipped.

The feature consists of: - A `keepalive request-interval` which will trigger a transmission of an MGCP AuditEndpoint command targeting endpoint with name `keepalive request-endpoint`. This interval is updated every time any message is transmitted in the MGCP link, meaning the MGCP AuditEndpoint message is only triggered if no message has been transmitted since `keepalive request-interval` seconds ago. - A `keepalive timeout` which upon triggering (because no message was received over that amount of time) will then consider the MGW to be non-reachable (link DOWN).

The `keepalive` parameters are to be preferrably configured so that `"keepalive request-interval" * 2 < "keepalive timeout"`.

Example VTY configuration of `keepalive` feature in OsmoBSC:

```
mgw 0
 ...
 keepalive request-interval 20 ❶
 keepalive request-endpoint null ❷
 keepalive timeout 50 ❸
```

❶ Transmit an MGCP AuditEndpoint message to the MGW if no message has been sent to it over last 20 seconds

❷ The MGCP AuditEndpoint targets the `null` endpoint. This is a special endpoint available at OsmoMGW for those purposes, but any available endpoint can be configured and used instead.

❸ Consider the MGCP link to be DOWN if no message is received from the MGW over the last 50 seconds

Note
The `keepalive` feature is disabled by default, and must be explicitly configured in order to enable it.

21.2.5 Blocking an MGW / MGCP-Client

If it becomes apparent that an MGCP-Client must be restarted or removed from the config (maintenance) the operator can put that MGCP-Client into a blocked mode. A blocked MGCP-Client will still serve the ongoing calls but it will not be picked for the assignment of new calls.

To block an MGCP-Client use the *block* VTY command:

```
OsmoBSC# mgw 2 block
OsmoBSC# show mgw-pool
% MGW-Pool:
%  MGW 0:media-gw-0
%   MGCP link:     connected,UP
%   service:       unblocked
%   ongoing calls: 11
%  MGW 1:media-gw-1
%   MGCP link:     connected,UP
%   service:       unblocked
%   ongoing calls: 12
%  MGW 2:mgw
%   MGCP link:     connected,UP
%   service:       blocked
%   ongoing calls: 10
```

When the number of ongoing calls has tapered off, the MGW / MGCP-Client can be restarted or removed if necessary.

```
OsmoBSC# show mgw-pool
% MGW-Pool:
%  MGW 0:media-gw-0
%   MGCP link:     connected,UP
%   service:       unblocked
%   ongoing calls: 15
%  MGW 1:media-gw-1
%   MGCP link:     connected,UP
%   service:       unblocked
%   ongoing calls: 14
%  MGW 2:mgw
%   MGCP link:     connected,UP
%   service:       blocked
%   ongoing calls: 0
```

If the blockade should be reverted, the *unblock* VTY command can be used in the same way to remove the blockade. (Reconnecting will not remove the blockade.)

21.2.6 Removing an MGW / MGCP-Client

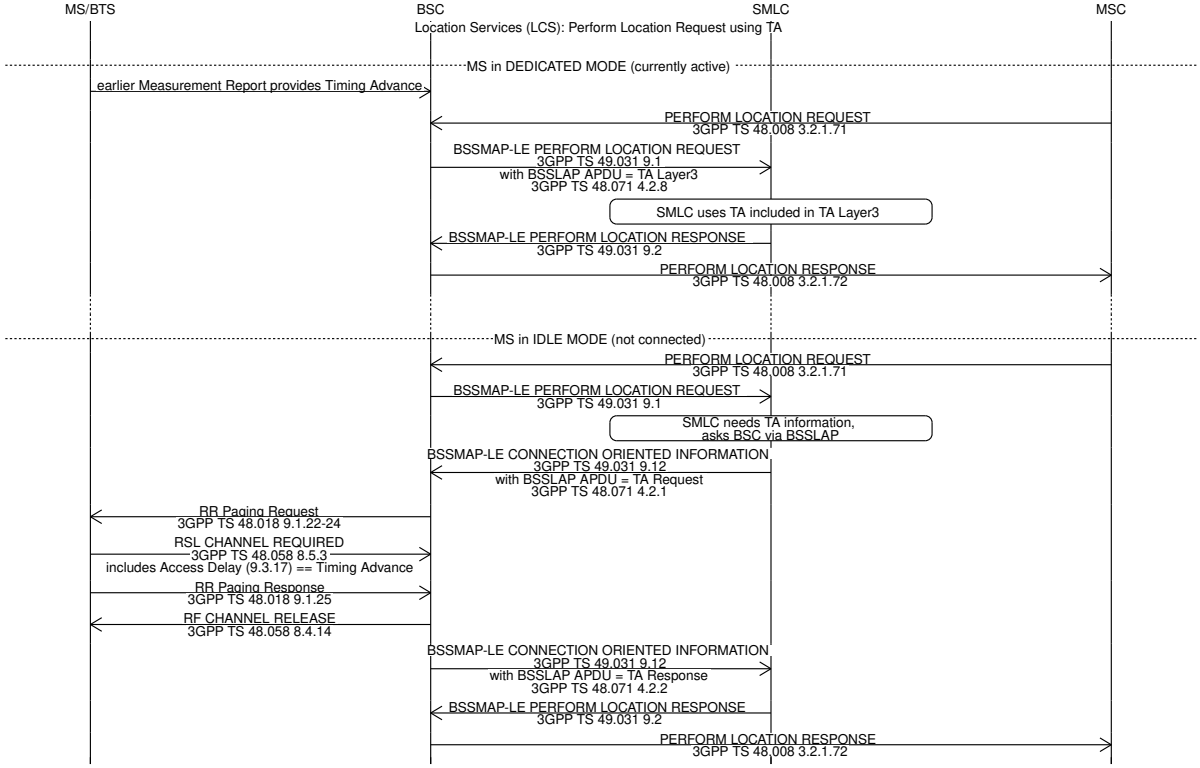
An MGCP-Client is removed from the pool using the *no mgw* command from the configure terminal. The MGCP-Client instance will automatically be terminated and the related resources are freed. The only requirement is that there are no ongoing calls on the selected instance.

```
OsmoBSC# configure terminal
OsmoBSC(config)# network
OsmoBSC(config-net)# no mgw 2
```

22 Location Services: Lb interface to SMLC

OsmoBSC and OsmoSMLC support positioning by Timing-Advance (TA), since October 2020.

A Perform Location Request is initiated by the MSC via BSSMAP on the A-interface, for a specific subscriber. The request is typically passed on via BSSMAP-LE on the Lb-interface to the SMLC. If required, the SMLC may request the subscriber's Timing Advance (a.k.a. Access Delay) from the BSC via BSSLAP (encapsulated BSSLAP APDU in a BSSMAP-LE Connection Oriented Information message). The SMLC may combine several location and velocity estimate methods to form a GAD PDU containing the resulting geographic location information. In TA-based positioning, the Timing-Advance information from the BSC is combined with the preconfigured latitude and longitude of the serving cell to form a location estimate. This is returned to the BSC via the Lb-interface, and in turn to the MSC via the A-interface.



Location Services (LCS) are described in 3GPP TS 43.059 [?]. Messages for LCS on the A-interface (BSSMAP, between BSC and MSC) are described in 3GPP TS 48.008 [3gpp-ts-48-008], on the Lb-interface (BSSMAP-LE between BSC and SMLC) in 3GPP TS 49.031 [?]. The resulting geographic location and possibly velocity information is encoded in GAD, described in 3GPP TS 23.032 [?].

22.1 Configure Lb-interface

All Lb-interface related configuration is found in the `smlc` section of OsmoBSC's configuration.

By default, the Lb-interface is disabled in OsmoBSC. It is started by `enable`.

```
smlc
 enable
```

On the Lb-interface, OsmoBSC always uses SSN "BSC (BSSMAP-LE)" (SSN code 250) and contacts the remote SMLC on SSN "SMLC (BSSMAP-LE)" (SSN code 252).

The point-codes are configurable, and default to OsmoBSC's local point-code 0.23.3 (187), and remote SMLC point-code 0.23.6 (190).

Typically, multiple BSCs connect to the same SMLC, so that each BSC needs to have a distinct point-code, while the SMLC has a single, fixed point-code.

To configure a different remote SMLC point-code, first configure an arbitrarily named SCCP address in the `cs7` address book, and then apply that to the `smlc-addr` configuration:

```
cs7 instance 0
 sccp-address my-smlc
  point-code 0.42.6
smlc
 enable
 smlc-addr my-smlc
```

For the BSC side, it suffices to configure a point-code in the `cs7` section, so that the BSC typically uses the same point-code on A and Lb interfaces. In this example, the BSC has point-code `1.2.3` on the Lb interface:

```
cs7 instance 0
 point-code 1.2.3
smlc
 enable
```

It is also possible to configure a distinct BSC's point-code on Lb, using the `bsc-addr` configuration. In the following example, the BSC uses point-code `0.42.3` only on the Lb interface, while the A interface remains unchanged:

```
cs7 instance 0
 sccp-address my-bsc-on-lb
  point-code 0.42.3
smlc
 enable
 bsc-addr my-bsc-on-lb
```

The geographic locations of individual cells are configured at the SMLC. See for example OsmoSMLC's user manual [userman-osmosmlc].

23 QoS, DSCP/TOS, Priority and IEEE 802.1q PCP

In many use cases operators want to apply different QoS classes for user plane vs. control plane traffic. IP Routers, Ethernet switches and other network gear can then perform intelligent queue management as required for the respective service.

For example, voice user plane frames need a rather stable and short latency, while IP user plane and control plane traffic has less critical latency requirements.

23.1 IP Level (DSCP)

At IP level, different priorities / classes of traffic are expressed in accordance to [ietf-rfc2474] by the DSCP (Differentiated Services Code Point) field of the IP header. DSCP resembles the upper 6 bits of the field formerly known as the TOS bits as per [ietf-rfc791].

On Linux and other operating systems with BSD-style sockets API, the applications can request a specific DSCP value to be used for packets generated by those sockets.

Osmocom CNI software such as osmo-bts and osmo-mgw support setting the DSCP value via VTY commands, see e.g. the `rtp ip-dscp` setting of the `bts` node in osmo-bts.

23.2 Packet Priority

In the Linux network stack, every packet is represented by `struct sk_buff`, which has an associated *priority*. Furthermore, every socket through which applications send data have an associated *socket priority*. Each time a packet is transmitted through a given socket, the packet inherits the packet priority from the socket priority.

Furthermore, there is a mapping table that maps DSCP/TOS bits to priority. The sixteen different TOS bit values are mapped to priority values as follows:

Table 23: Linux kernel default DSCP/TOS → priority mapping

TOS (binary)	DSCP (binary)	Priority (decimal)
xxx0000x	xxx000	0
xxx0001x	xxx000	0
xxx0010x	xxx001	0
xxx0011x	xxx001	0

Table 23: (continued)

TOS (binary)	DSCP (binary)	Priority (decimal)
xxx0100x	xxx010	2
xxx0101x	xxx010	2
xxx0110x	xxx011	2
xxx0111x	xxx011	2
xxx1000x	xxx100	6
xxx1001x	xxx100	6
xxx1010x	xxx101	6
xxx1011x	xxx101	6
xxx1100x	xxx110	4
xxx1101x	xxx110	4
xxx1110x	xxx111	4
xxx1111x	xxx111	4

This table of default DSCP/TOS → priority bit mappings cannot be modified.

However, the per-packet *priority* values can be set by various means of network policy, including

- by packet filter rules (iptables, ip6tables, nftables)
  - if you use `iptables`, using `CLASSIFY --set-class` in the `mangle` table
  - if you use `nftables`, using `meta priority set` in the `mangle` table
- by the application using the SO_PRIORITY socket option (currently not yet supported by Osmocom CNI)

23.3 Ethernet Level (PCP)

At Ethernet level, different priorities / QoS classes are expressed by the so-called PCP (Priority Code Point) field in the IEEE 802.1q (VLAN) header.

NOTE
This means that PCP functionality requires the use of IEEE 802.q VLAN. You cannot use PCP without VLAN.

The Linux kernel assigns IEEE 802.1q PCP bits based on a *mapping* between the *priority* and the PCP value. Each VLAN network device maintains a separate map for both egress (transmit) and ingress (receive) path.

The current priority mappings can be inspected via the `/proc` filesystem. For example, if you have a VLAN device `eth0.9` for VLAN ID 9 on the net-device `eth0`, you can use the following example:

Example: Inspecting the current egress QoS map

```
$ sudo cat /proc/net/vlan/eth0.9❶
eth0.9  VID: 9   REORDER_HDR: 1  dev->priv_flags: 1021
         total frames received       123340
          total bytes received     40668066
      Broadcast/Multicast Rcvd         1106

      total frames transmitted        10499
       total bytes transmitted      1570809
Device: eth0
INGRESS priority mappings: 0:0  1:0  2:0  3:0  4:0  5:0  6:0 7:0 ❷
 EGRESS priority mappings: ❸
```

❶ make sure to specify your specific VLAN interface name here instead of `eth0.9`

❷ ingress priority mappings (all PCP values mapped to priority 0)

❸ egress priority mappings (empty)

As we can see in the above example, there are no egress priority mappings yet. Let's create three new mappings, mapping *priority* value 1 to PCP 1, *priority* 2 to PCP 2, and *priority* 3 to PCP 3:

Example: Creating three new egress QoS mappings

```
$ sudo ip link set dev eth0.9❶ type vlan egress-qos-map 1:1 2:2 3:3 ❷
$ sudo cat /proc/net/vlan/eth0.9 ❸
eth0.9  VID: 9   REORDER_HDR: 1  dev->priv_flags: 1021
         total frames received       123898
          total bytes received     40843611
      Broadcast/Multicast Rcvd         1106

      total frames transmitted        10517
       total bytes transmitted      1574357
Device: eth0
INGRESS priority mappings: 0:0  1:0  2:0  3:0  4:0  5:0  6:0 7:0
 EGRESS priority mappings: 1:1 2:2 3:3 ❹
```

❶ make sure to specify your specific VLAN interface name here instead of `eth0.9`

❷ command to define three new egress QoS maps

❸ command to re-display the current status

❹ three new egress mappings are shown as given in `ip` command

NOTE

The settings of the `ip` command are volatile and only active until the next reboot (or the network device or VLAN is removed). Please refer to the documentation of your specific Linux distribution in order to find out how to make such settings persistent by means of an `ifup` hook whenever the interface comes up. For CentOS/RHEL 8 this can e.g. be achieved by means of an `/sbin/ifup-local script` (when using `network-scripts` and not NetworkManager). For Debian or Ubuntu, this typically involves adding `up` lines to `/etc/network/interfaces` or a `/etc/network/if-up.d` script.

23.4 Putting things together

Assuming one needs to set both the DSCP bits as well as the PCP for certain traffic, the above-mentioned mechanisms need to be combined as follows:

1. configure the osmocom program to set the DSCP value
2. use the default DSCP → priority mapping, if possible
3. configure an egress QoS map to map from priority to PCP

If the desired combination of DSCP + PCP cannot be achieved that way, due to the rather static default kernel mapping table, one needs to go one step further:

1. configure the osmocom program to set the DSCP value
2. use packet filter rules to set the priority based on DSCP
3. configure an egress QoS map to map from priority to PCP

23.4.1 Full example of QoS for osmo-bsc downlink QoS

In the below example we will show the full set of configuration required for both DSCP and PCP differentiation of downlink Abis traffic by osmo-bsc.

What we want to achieve in this example is the following configuration:

Table 24: DSCP and PCP assignments for osmo-bsc downlink traffic in this example

Traffic	DSCP	PCP
A-bis RSL	56	7
A-bis RTP	46	6
A-bis OML	34	5

1. configure the osmocom program to set the DSCP value
2. configure an egrees QoS map to map from priority to PCP

Example Step 1: add related VTY configuration to osmo-bsc.cfg

```
...
e1_input
 ipa ip-dscp oml 34
 ipa socket-priority oml 5
 ipa ip-dscp rsl 56
 ipa socket-priority rsl 7
...
```

Example Step 2: egress QoS map to map from socket priority to PCP values

```
$ sudo ip link set dev eth0.9❶ type vlan egress-qos-map 0:0 5:5 6:6 7:7 ❷
```

❶ make sure to specify your specific VLAN interface name here instead of `eth0.9`.

❷ create a egress QoS map that maps the priority value 1:1 to the PCP. We also include the mapping 6:6 from the osmo-mgw example (see [userman-osmomgw]) here.

NOTE
The settings of the `ip` command are volatile and only active until the next reboot (or the network device or VLAN is removed). Please refer to the documentation of your specific Linux distribution in order to find out how to make such settings persistent by means of an `ifup` hook whenever the interface comes up. For CentOS/RHEL 8 this can e.g. be achieved by means of an `/sbin/ifup-local script` (when using `network-scripts` and not NetworkManager). For Debian or Ubuntu, this typically involves adding `up` lines to `/etc/network/interfaces` or a `/etc/network/if-up.d` script.

24 Osmocom Counters

The following gives an overview of all the types of counters available:

24.1 Osmo Counters (deprecated)

Osmo counters are the oldest type of counters added to Osmocom projects. They are not grouped.

- Printed as part of VTY show stats
- Increment, Decrement
- Accessible through the control interface: counter.<counter_name>

24.2 Rate Counters

Rate counters count rates of events.

- Printed as part of VTY show stats
- Intervals: per second, minute, hour, day or absolute value
- Increment only
- Accessible through the control interface
- Rate counters are grouped and different instances per group can exist

The control interface command to get a counter (group) is:

rate_ctr.per_{sec,min,hour,day,abs}.<group_name>.<idx>.[counter_name]

It is possible to get all counters in a group by omitting the counter name

24.3 Stat Item

Stat items are a grouped replacement for osmo counters.

- Printed as part of VTY show stats
- Replacement for osmo counters
- Not yet available through the control interface
- Grouped and indexed like rate counters
- Items have a unit
- Keeps a list of the last values measured, so could return an average, min, max, std. deviation. So far this is not implemented in any of the reporting options.

24.4 Statistic Levels

There are three levels on which a statistic can be aggregated in Osmocom projects: globally, per-peer and per-subscriber.

24.4.1 Global

These are global statistics.

24.4.2 Peer

These statistics relate to a peer the program connects to such as the NSVC in an SGSN.

This level also includes reporting global statistics.

24.4.3 Subscriber

These statistics are related to an individual mobile subscriber. An example would be bytes transferred in an SGSN PDP context.

This level also includes global and peer-based statistics.

24.5 Stats Reporter

The stats reporter periodically collects osmo counter, rate counter and stat item values and sends them to a backend. Currently implemented are outputting to the configured log targets and a statsd connector.

24.5.1 Configuring a stats reporter

Periodically printing the statistics to the log can be done in the following way:

Example 24.1 Log statistics

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# stats interval 60 ❶
OsmoBSC(config)# stats reporter log ❷
OsmoBSC(config-stats)# level global ❸
OsmoBSC(config-stats)# enable ❹
```

❶ The interval determines how often the statistics are reported.

❷ Write the statistic information to any configured log target.

❸ Report only `global` statistics (can be `global`, `peer`, or `subscriber`).

❹ Enable the reporter, `disable` will disable it again.

The counter values can also be sent to any aggregation/visualization tool that understands the statsd format, for example a statsd server with graphite or prometheus using the statsd_exporter together with grafana.

The statsd format is specified in https://github.com/b/statsd_spec

Example 24.2 Report statistics to statsd

```
OsmoBSC> enable
OsmoBSC# configure terminal
OsmoBSC(config)# stats interval 10
OsmoBSC(config)# stats reporter statsd ❶
OsmoBSC(config-stats)# prefix BSC1 ❷
OsmoBSC(config-stats)# level subscriber ❸
OsmoBSC(config-stats)# remote-ip 1.2.3.4 ❹
OsmoBSC(config-stats)# remote-port 8125 ❺
OsmoBSC(config-stats)# enable
```

❶ Configure the statsd reporter.

❷ Prefix the reported statistics. This is useful to distinguish statistics from multiple instances of the same service.

❸ Report only `global` statistics or include `peer` or `subscriber` statistics as well.

❹ IP address of the statsd server.

❺ UDP port of the statsd server. Statsd by default listens to port 8125.

You can use Netdata (https://learn.netdata.cloud/) as a statsd server which does not require special configuration to show rate counters. By default all the rate counters will be exposed to the StatsD plugin (listening on 127.0.0.1:8125) and displayed on the Netdata dashboard available via: http://localhost:19999 The list of available charts which includes all the rate counters reported via statsD is available through: http://localhost:19999/api/v1/charts

24.6 Socket stats

libosmocore provides features to monitor the status of TCP connections. This can be a helpful source of information when the links between network components are unreliable (e.g. satellite link between BTS and BSC).

Note
This feature is only available for certain types of TCP connections. At the moment only RSL/OML connections between OsmoBSC and the connected BTSs can be monitored.

24.6.1 Configuration

The gathering of the TCP connection statistics is done via syscalls. This has to be taken into account for the configuration. Since syscalls are rather expensive and time consuming the overall performance of the application may suffer when many TCP connections are present. This may be the case for BSCs with a large number of BTSs connected to it.

The statistics are gathered in batches per interval. A batch size of 5 would mean that only 5 TCP connections per interval are evaluated and the next 5 connections in the next interval and so on.

It is recommended to choose a large reporting interval and a reasonable small batch size to distribute the syscall load as even as possible.

Example 24.3 Report statistics to statsd

```
OsmoBSC> enable
OsmoBSC# configure terminal
stats-tcp interval 10 ❶
stats-tcp batch-size 5 ❷
```

❶ Set the gathering interval (sec.)

❷ Set how many TCP sockets statistics to gather per interval.

24.6.2 Generated stats items

Name	Description
tcp:unacked	unacknowledged packets.
tcp:lost	unacknowledged packets.
tcp:retrans	lost packets.
tcp:rtt	retransmitted packets.
tcp:rcv_rtt	roundtrip-time (receive).
tcp:notsent_bytes	bytes not yet sent.
tcp:rwnd_limited	time (usec) limited by receive window.
tcp:sndbuf_limited	Time (usec) limited by send buffer.
tcp:reord_seen	reordering events seen.

The item group index is the file descriptor number. The item group name consists of a static prefix (e.g. "ipa-rsl"), followed by the IP addresses and ports of both peers.

Example 24.4 VTY output of a stats item group of a TCP connection

```
stats tcp (15)('ipa-rsl,r=10.9.1.143:38455<->l=10.9.1.162:3003'):
 unacknowledged packets:        0
 lost packets:        0
 retransmitted packets:        0
 roundtrip-time:      583
 roundtrip-time (receive):        0
 bytes not yet sent:        0
 time (usec) limited by receive window:        0
 Time (usec) limited by send buffer:        0
 reordering events seen:        0
```

25 Implemented Counters

These counters and their description based on OsmoBSC 1.4.0.84-3f1f8 (OsmoBSC).

25.1 Rate Counters

Table 26: bts - base transceiver station

Name	Reference	Description
chreq:total	[?]	Received channel requests.
chreq:no_channel	[?]	Sent to MS no channel available.
chan:rf_fail	[?]	Received a RF failure indication from BTS.
chan:rll_err	[?]	Received a RLL failure with T200 cause from BTS.
oml_fail	[?]	Received a TEI down on a OML link.
rsl_fail	[?]	Received a TEI down on a OML link.
codec:amr_f	[?]	Count the usage of AMR/F codec by channel mode requested.
codec:amr_h	[?]	Count the usage of AMR/H codec by channel mode requested.
codec:efr	[?]	Count the usage of EFR codec by channel mode requested.
codec:fr	[?]	Count the usage of FR codec by channel mode requested.
codec:hr	[?]	Count the usage of HR codec by channel mode requested.
paging:attempted	[?]	Paging attempts for a subscriber.
paging:already	[?]	Paging attempts ignored as subscriber was already being paged.
paging:responded	[?]	Paging attempts with successful paging response.
paging:expired	[?]	Paging Request expired because of timeout T3113.
chan_act:total	[?]	Total number of Channel Activations.
chan_act:nack	[?]	Number of Channel Activations that the BTS NACKed
rsl:unknown	[?]	Number of unknown/unsupported RSL messages received from BTS

Table 26: (continued)

Name	Reference	Description
rsl:ipa_nack	[?]	Number of IPA (RTP/dyn-PDCH) related NACKs received from BTS
chan:mode_modify_nack	[?]	Number of Channel Mode Modify NACKs received from BTS

Table 27: bts - base transceiver station

Name	Reference	Description
chreq:total	[?]	Received channel requests.
chreq:no_channel	[?]	Sent to MS no channel available.
chan:rf_fail	[?]	Received a RF failure indication from BTS.
chan:rll_err	[?]	Received a RLL failure with T200 cause from BTS.
oml_fail	[?]	Received a TEI down on a OML link.
rsl_fail	[?]	Received a TEI down on a OML link.
codec:amr_f	[?]	Count the usage of AMR/F codec by channel mode requested.
codec:amr_h	[?]	Count the usage of AMR/H codec by channel mode requested.
codec:efr	[?]	Count the usage of EFR codec by channel mode requested.
codec:fr	[?]	Count the usage of FR codec by channel mode requested.
codec:hr	[?]	Count the usage of HR codec by channel mode requested.
paging:attempted	[?]	Paging attempts for a subscriber.
paging:already	[?]	Paging attempts ignored as subscriber was already being paged.
paging:responded	[?]	Paging attempts with successful paging response.
paging:expired	[?]	Paging Request expired because of timeout T3113.
chan_act:total	[?]	Total number of Channel Activations.
chan_act:nack	[?]	Number of Channel Activations that the BTS NACKed
rsl:unknown	[?]	Number of unknown/unsupported RSL messages received from BTS
rsl:ipa_nack	[?]	Number of IPA (RTP/dyn-PDCH) related NACKs received from BTS
chan:mode_modify_nack	[?]	Number of Channel Mode Modify NACKs received from BTS

Table 28: bts - base transceiver station

Name	Reference	Description
chreq:total	[?]	Received channel requests.
chreq:no_channel	[?]	Sent to MS no channel available.

Table 28: (continued)

Name	Reference	Description
chan:rf_fail	[?]	Received a RF failure indication from BTS.
chan:rll_err	[?]	Received a RLL failure with T200 cause from BTS.
oml_fail	[?]	Received a TEI down on a OML link.
rsl_fail	[?]	Received a TEI down on a OML link.
codec:amr_f	[?]	Count the usage of AMR/F codec by channel mode requested.
codec:amr_h	[?]	Count the usage of AMR/H codec by channel mode requested.
codec:efr	[?]	Count the usage of EFR codec by channel mode requested.
codec:fr	[?]	Count the usage of FR codec by channel mode requested.
codec:hr	[?]	Count the usage of HR codec by channel mode requested.
paging:attempted	[?]	Paging attempts for a subscriber.
paging:already	[?]	Paging attempts ignored as subscriber was already being paged.
paging:responded	[?]	Paging attempts with successful paging response.
paging:expired	[?]	Paging Request expired because of timeout T3113.
chan_act:total	[?]	Total number of Channel Activations.
chan_act:nack	[?]	Number of Channel Activations that the BTS NACKed
rsl:unknown	[?]	Number of unknown/unsupported RSL messages received from BTS
rsl:ipa_nack	[?]	Number of IPA (RTP/dyn-PDCH) related NACKs received from BTS
chan:mode_modify_nack	[?]	Number of Channel Mode Modify NACKs received from BTS

Table 29: e1inp - E1 Input subsystem

Name	Reference	Description
hdlc:abort	[?]	HDLC abort
hdlc:bad_fcs	[?]	HLDC Bad FCS
hdlc:overrun	[?]	HDLC Overrun
alarm	[?]	Alarm
removed	[?]	Line removed

Table 30: bsc - base station controller

Name	Reference	Description
assignment:attempted	[?]	Assignment attempts.
assignment:completed	[?]	Assignment completed.

Table 30: (continued)

Name	Reference	Description
assignment:stopped	[?]	Connection ended during Assignment.
assignment:no_channel	[?]	Failure to allocate lchan for Assignment.
assignment:timeout	[?]	Assignment timed out.
assignment:failed	[?]	Received Assignment Failure message.
assignment:error	[?]	Assignment failed for other reason.
handover:attempted	[?]	Intra-BSC handover attempts.
handover:completed	[?]	Intra-BSC handover completed.
handover:stopped	[?]	Connection ended during HO.
handover:no_channel	[?]	Failure to allocate lchan for HO.
handover:timeout	[?]	Handover timed out.
handover:failed	[?]	Received Handover Fail messages.
handover:error	[?]	Re-assignment failed for other reason.
interbsc_ho_out:attempted	[?]	Attempts to handover to remote BSS.
interbsc_ho_out:completed	[?]	Handover to remote BSS completed.
interbsc_ho_out:stopped	[?]	Connection ended during HO.
interbsc_ho_out:timeout	[?]	Handover timed out.
interbsc_ho_out:error	[?]	Handover to remote BSS failed for other reason.
interbsc_ho_in:attempted	[?]	Attempts to handover from remote BSS.
interbsc_ho_in:completed	[?]	Handover from remote BSS completed.
interbsc_ho_in:stopped	[?]	Connection ended during HO.
interbsc_ho_in:no_channel	[?]	Failure to allocate lchan for HO.
interbsc_ho_in:failed	[?]	Received Handover Fail message.
interbsc_ho_in:timeout	[?]	Handover from remote BSS timed out.
interbsc_ho_in:error	[?]	Handover from remote BSS failed for other reason.
paging:attempted	[?]	Paging attempts for a subscriber.
paging:detached	[?]	Paging request send failures because no responsible BTS was found.
paging:responded	[?]	Paging attempts with successful response.
abis:unknown_unit_id	[?]	Connection attempts from unknown IPA CCM Unit ID.

26 Osmo Stat Items

base transceiver station .bts - base transceiver station

Name	Reference	Description	Unit
chanloadavg	[?]	Channel load average.	%
T3122	[?]	T3122 IMMEDIATE ASSIGNMENT REJECT wait indicator.	s
rach_busy	[?]	RACH slots with signal above threshold	%

Name	Reference	Description	Unit
rach_access	[?]	RACH slots with access bursts in them	%

base transceiver station .bts - base transceiver station

Name	Reference	Description	Unit
chanloadavg	[?]	Channel load average.	%
T3122	[?]	T3122 IMMEDIATE ASSIGNMENT REJECT wait indicator.	s
rach_busy	[?]	RACH slots with signal above threshold	%
rach_access	[?]	RACH slots with access bursts in them	%

base transceiver station .bts - base transceiver station

Name	Reference	Description	Unit
chanloadavg	[?]	Channel load average.	%
T3122	[?]	T3122 IMMEDIATE ASSIGNMENT REJECT wait indicator.	s
rach_busy	[?]	RACH slots with signal above threshold	%
rach_access	[?]	RACH slots with access bursts in them	%

27 Osmo Counters

28 Abis/IP Interface

28.1 A-bis Operation & Maintenance Link

The GSM Operation & Maintenance Link (OML) is specified in 3GPP TS 12.21 and is used between a GSM Base-Transceiver-Station (BTS) and a GSM Base-Station-Controller (BSC). The default TCP port for OML is *3002*. The connection will be opened from the BTS to the BSC.

Abis OML is only specified over E1 interfaces. The Abis/IP implementation of OsmoBTS and OsmoBSC extend and/or deviate from the TS 12.21 specification in several ways. Please see the *OsmoBTS Abis Protocol Specification* [osmobts-abis-spec] for more information.

28.2 A-bis Radio Signalling Link

The GSM Radio Signalling Link (RSL) is specified in 3GPP TS 08.58 and is used between a GSM Base-Transceiver-Station and a GSM Base-Station-Controller (BSC). The default TCP port for RSL is *3003*. The connection will be opened from the BTS to BSC after it has been instructed by the BSC.

Abis RSL is only specified over E1 interfaces. The Abis/IP implementation of OsmoBTS and OsmoBSC extend and/or deviate from the TS 08.58 specification in several ways. Please see the *OsmoBTS Abis Protocol Specification* [osmobts-abis-spec] for more information.

28.3 Locate Abis/IP based BTS

We can use a tool called abisip-find to be able to find BTS which is connected in the network. This tool is located in the OsmoBSC project repository under: *./src/ipaccess*

28.3.1 abisip-find

abisip-find is a small command line tool which is used to search and find BTS devices in your network (e.g. sysmoBTS, nanoBTS).

It uses broadcast packets of the UDP variant of the Abis-IP protocol on port 3006, and thus will find any BTS that can be reached by the all-network broadcast address 255.255.255.255

When program is started it will print one line for each BTS it can find.

Example: using abisip-find to find BTS in your network

```
$ ./abisip-find
abisip-find (C) 2009 by Harald Welte
This is FREE SOFTWARE with ABSOLUTELY NO WARRANTY

you might need to specify the outgoing
network interface, e.g. ``abisip-find eth0``
Trying to find ip.access BTS by broadcast UDP...

MAC_Address='24:62:78:01:02:03' IP_Address='192.168.0.171' Serial_Number='123'
Unit_ID='sysmoBTS 1002'

MAC_Address='24:62:78:04:05:06' IP_Address='192.168.0.182' Serial_Number='456'
Unit_ID='sysmoBTS 1002'

MAC Address='00:01:02:03:04:05' IP Address='192.168.100.123' Unit ID='65535/0/0'
Location_1='' Location 2='BTS_NBT131G'  Equipment Version='165a029_55'
Software Version='168a302_v142b13d0' Unit Name='nbts-00-02-95-00-4E-B3'
Serial Number='00123456'

^C
```

You may have to start the program as a root:

```
$ sudo ./abisip-find eth0
```

28.4 Deploying a new nanoBTS

A tool called ipaccess-config can be used to configure a new ip.access nanoBTS.

28.4.1 ipaccess-config

This program is very helpful tool which is used to configure Unit ID and Primary OML IP. You can find this tool in the OsmoBSC repository under: *./src/ipaccess*

Example: using ipaccess-config to configure Unit ID and Primary OML IP of nanoBTS

```
$ ./ipaccess-config -u 1801/0/0❶  10.9.1.195❷ -o 10.9.1.154❸

ipaccess-config (C) 2009-2010 by Harald Welte and others
This is FREE SOFTWARE with ABSOLUTELY NO WARRANTY

Trying to connect to ip.access BTS ...
```

```
abis_nm.c:316 OC=SITE-MANAGER(00) INST=(ff,ff,ff) STATE CHG:
OP_STATE=Disabled AVAIL=Not installed(07)
abis_nm.c:316 OC=BTS(01) INST=(00,ff,ff) STATE CHG:
OP_STATE=Disabled AVAIL=Not installed(07) ADM=Locked
abis_nm.c:316 OC=BASEBAND-TRANSCEIVER(04) INST=(00,00,ff) STATE CHG:
OP_STATE=Disabled AVAIL=Not installed(07) ADM=Locked
OML link established using TRX 0
setting Unit ID to '1801/0/0'
setting primary OML link IP to '10.9.1.154'
abis_nm.c:316 OC=CHANNEL(03) INST=(00,00,00) STATE CHG:
OP_STATE=Disabled AVAIL=Not installed(07) ADM=Locked
...
abis_nm.c:2433 OC=BASEBAND-TRANSCEIVER(04) INST=(00,00,ff) IPACCESS(0xf0):
SET NVATTR ACK
Set the NV Attributes.
```

1. Unit ID
2. IP address of the NITB
3. IP address of the nanoBTS

29 Osmocom Control Interface

The VTY interface as described in Section 6 is aimed at human interaction with the respective Osmocom program.

Other programs **should not** use the VTY interface to interact with the Osmocom software, as parsing the textual representation is cumbersome, inefficient, and will break every time the formatting is changed by the Osmocom developers.

Instead, the *Control Interface* was introduced as a programmatic interface that can be used to interact with the respective program.

29.1 Control Interface Protocol

The control interface protocol is a mixture of binary framing with text based payload.

The protocol for the control interface is wrapped inside the IPA multiplex header with the stream identifier set to IPAC_PROTO_OSMO (0xEE).

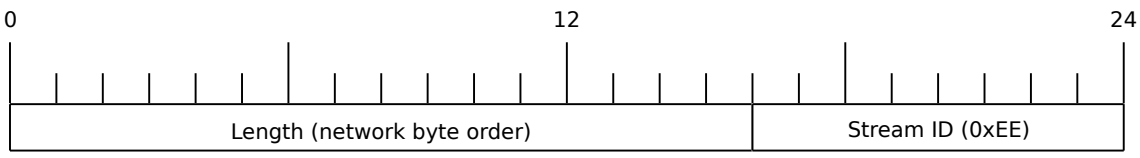


Figure 9: IPA header for control protocol

Inside the IPA header is a single byte of extension header with protocol ID 0x00 which indicates the control interface.

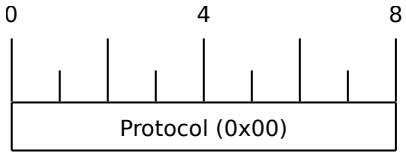


Figure 10: IPA extension header for control protocol

After the concatenation of the two above headers, the plain-text payload message starts. The format of that plain text is illustrated for each operation in the respective message sequence chart in the chapters below.

The fields specified below follow the following meaning:

<id>
A numeric identifier, uniquely identifying this particular operation. Value `0` is not allowed unless it's a TRAP message. It will be echoed back in any response to a particular request.

<var>
The name of the variable / field affected by the GET / SET / TRAP operation. Which variables/fields are available is dependent on the specific application under control.

<val>
The value of the variable / field

<reason>
A text formatted, human-readable reason why the operation resulted in an error.

29.1.1 GET operation

The GET operation is performed by an external application to get a certain value from inside the Osmocom application.

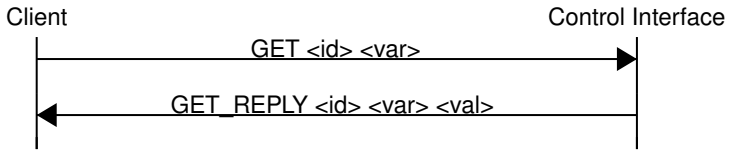


Figure 11: Control Interface GET operation (successful outcome)

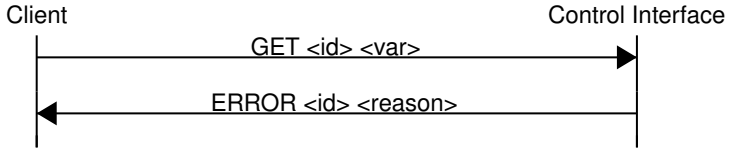


Figure 12: Control Interface GET operation (unsuccessful outcome)

29.1.2 SET operation

The SET operation is performed by an external application to set a value inside the Osmocom application.

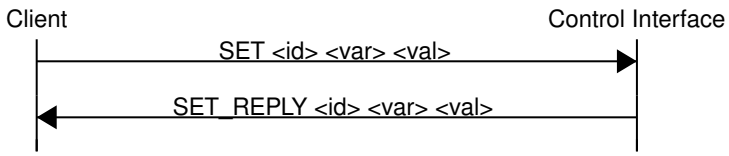


Figure 13: Control Interface SET operation (successful outcome)

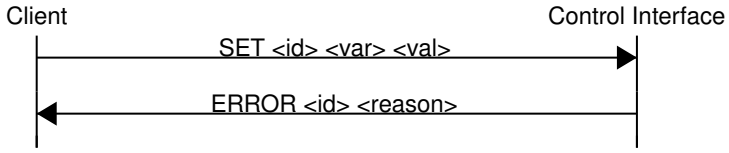


Figure 14: Control Interface SET operation (unsuccessful outcome)

29.1.3 TRAP operation

The program can at any time issue a trap. The term is used in the spirit of SNMP.

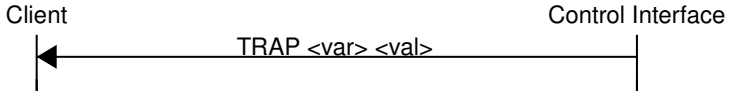


Figure 15: Control Interface TRAP operation

29.2 Common variables

There are several variables which are common to all the programs using control interface. They are described in the following table.

Table 34: Variables available over control interface

Name	Access	Value	Comment
counter.*	RO		Get counter value.
rate_ctr.*	RO		Get list of rate counter groups.
rate_ctr.IN.GN.GI.name	RO		Get value for interval IN of rate counter name which belong to group named GN with index GI.

Those read-only variables allow to get value of arbitrary counter using its name.

For example "`rate_ctr.per_hour.bsc.0.handover:timeout`" is the number of handover timeouts per hour.

Of course for that to work the program in question have to register corresponding counter names and groups using libosmocore functions.

In the example above, "`bsc`" is the rate counter group name and "`0`" is its index. It is possible to obtain all the rate counters in a given group by requesting "`rate_ctr.per_sec.bsc.*`" variable.

The list of available groups can be obtained by requesting "`rate_ctr.*`" variable.

The rate counter group name have to be prefixed with interval specification which can be any of "**per_sec**", "**per_min**", "**per_hour**", "**per_day**" or "**abs**" for absolute value.

The old-style counters available via "`counter.*`" variables are superseded by "`rate_ctr.abs`" so its use is discouraged. There might still be some applications not yet converted to rate_ctr.

29.3 Control Interface python examples

In the `osmo-python-tests` repository, there is an example python script called `scripts/osmo_ctrl.py` which implements the Osmocom control interface protocol.

You can use this tool either stand-alone to perform control interface operations against an Osmocom program, or you can use it as a reference for developing your own python software talking to the control interface.

Another implementation is in `scripts/osmo_rate_ctr2csv.py` which will retrieve performance counters for a given Osmocom program and output it in csv format. This can be used to periodically (using systemd timer for example) retrieve data to build KPI and evaluate how it changes over time.

Internally it uses "`rate_ctr.*`" variable described in Section 29.2 to get the list of counter groups and than request all the counters in each group. Applications interested in individual metrics can request it directly using `rate_ctr2csv.py` as an example.

29.3.1 Getting rate counters

Example: Use rate_ctr2csv.py to get rate counters from OsmoBSC

```
$ ./scripts/osmo_rate_ctr2csv.py --header
Connecting to localhost:4249...
Getting rate counter groups info...
"group","counter","absolute","second","minute","hour","day"
"e1inp.0","hdlc:abort","0","0","0","0","0"
"e1inp.0","hdlc:bad_fcs","0","0","0","0","0"
"e1inp.0","hdlc:overrun","0","0","0","0","0"
"e1inp.0","alarm","0","0","0","0","0"
"e1inp.0","removed","0","0","0","0","0"
"bsc.0","chreq:total","0","0","0","0","0"
"bsc.0","chreq:no_channel","0","0","0","0","0"
...
"msc.0","call:active","0","0","0","0","0"
"msc.0","call:complete","0","0","0","0","0"
"msc.0","call:incomplete","0","0","0","0","0"
Completed: 44 counters from 3 groups received.
```

29.3.2 Setting a value

Example: Use osmo_ctrl.py to set the short network name of OsmoBSC

```
$ ./osmo_ctrl.py -d localhost -s short-name 32C3
Got message: SET_REPLY 1 short-name 32C3
```

29.3.3 Getting a value

Example: Use osmo_ctrl.py to get the mnc of OsmoBSC

```
$ ./osmo_ctrl.py -d localhost -g mnc
Got message: GET_REPLY 1 mnc 262
```

29.3.4 Listening for traps

You can use `osmo_ctrl.py` to listen for traps the following way:

Example: Using osmo_ctrl.py to listen for traps:

```
$ ./osmo_ctrl.py -d localhost -m
❶
```

❶ the command will not return and wait for any TRAP messages to arrive

30 Control interface

The actual protocol is described in Section 29, the variables common to all programs using it are described in Section 29.2. Here we describe variables specific to OsmoBSC. The commands starting with prefix "bts.N." are specific to a certain BTS so N have to be replaced with BTS number when issuing command e. g. "bts.1.channel-load". Similarly the TRX-specific commands are additionally prefixed with TRX number e. g. "bts.1.trx.2.arfcn".

Table 35: Variables available over control interface

Name	Access	Trap	Value	Comment
msc_connection_status	RO	Yes	"connected", "disconnected"	Indicate the status of connection to MSC.
bts_connection_status	RO	Yes	"connected", "disconnected"	Indicate the status of connection to BTS.
location	RW	Yes	"<unixtime>,(invalid|fix2d|fix3d),<lat>,<lon>,<height>"	Set/Get location data.
timezone	RW	No	"<hours>,<mins>,<dst>", "off"	-19 <= hours <= 19, mins in {0, 15, 30, 45}, and 0 <= dst <= 2
apply-configuration	WO	No	"restart"	Restart all BTSes.
mnc	RW	No	"<mnc>"	Set/Get MNC (value between (0, 999)).
mcc	RW	No	"<mcc>"	Set/Get MCC (value between (1, 999)).
mcc-mnc-apply	WO	No	"<mcc>,<mnc>"	Apply new MCC/MNC values if different from currently used one.
notification	WO	Yes	Arbitrary value	See Section 30.1 for details.
inform-msc-v1	WO	Yes	Arbitrary value	See Section 30.2 for details.
rf_locked	RW	No	"0","1"	See Section 30.6 for details.
number-of-bts	RO	No	"<num>"	Get number of configured BTS.

Table 35: (continued)

Name	Access	Trap	Value	Comment
apply-config-file	WO	No	"<filename>"	Apply VTY config file snippet from file.
write-config-file	WO	No	"overwrite", "<filename>"	Write running configuration to file.
bts.N.location-area-code	RW	No	"<lac>"	Set/Get LAC (value between (0, 65535)).
bts.N.cell-identity	RW	No	"<id>"	Set/Get Cell Identity (value between (0, 65535)).
bts.N.bsic	RW	No	"<bsic>"	Set/Get BSIC (value between (0, 63)).
bts.N.rach-max-delay	RW	No	"<delay>"	Set/Get RACH max delay (value between (1, 127)).
bts.N.apply-configuration	WO	No	Ignored	Restart BTS via OML.
bts.N.send-new-system-informations	WO	No	Ignored	Regenerate and resend System Information messages for given BTS.
bts.N.send-power-control-defaults	WO	No	Ignored	Resend default power control parameters for given BTS.
bts.N.channel-load	RO	No	"<name>,<used>,<total>"	See Section 30.3 for details.
bts.N.oml-connection-state	RO	No	"connected", "disconnected", "degraded"	Indicate the status of OML connection of BTS.
bts.N.oml-uptime	RO	No	<uptime>	Return OML link uptime in seconds.
bts.N.gprs-mode	RW	No	"<mode>"	See Section 30.4 for details.
bts.N.rf_state	RO	No	"<oper>,<admin>,<pol>"	See Section 30.5 for details.
bts.N.cell-reselection-offset	RW	No	"<cro>"	Set/Get cell reselection offset (value between (0, 126), steps of 2).
bts.N.cell-reselection-penalty-time	RW	No	"<penalty-time>","reserved"	Set/Get cell reselection penalty time (value between (20, 620), steps of 20).
bts.N.cell-reselection-hysteresis	RW	No	"<crh>"	Set/Get cell reselection hysteresis (value between (0, 14), steps of 2).
bts.N.rach-access-control-classes	RO	No	"<class>,(barred	allowed)"
Get concatenated pairs of RACH access control classes.	bts.N.rach-access-control-class.bar	WO	No	"<class>","emergency"

Table 35: (continued)

Name	Access	Trap	Value	Comment
Set RACH access control class as barred.	bts.N.rach-access-control-class.allow	W-O	No	"<class>","emergency"
Set RACH access control class as allowed.	bts.N.trx.arfcn	RW	No	"<arfcn>"
Set/Get ARFCN (value between (0, 1023)).	bts.N.trx.max-power-reduction	RW	No	"<mpr>"
See Section 30.7 for details.	[bts.N.]handover.active	RW	No	"0","1","default"
Enable/disable handover.	[bts.N.]handover.algorithm	RW	No	"1","2","default"
Choose algorithm for handover decision (hodec1 or hodec2).	[bts.N.]handover.window.rxlev.averaging	RW	No	<1-10>,"default"
How many RxLev measurements to use for averaging.	[bts.N.]handover.window.rxqual.averaging	RW	No	<1-10>,"default"
How many RxQual measurements to use for averaging.	[bts.N.]handover.window.rxlev.neighbor.averaging	RW	No	<1-10>,"default"
How many Neighbor RxLev measurements to use for averaging.	[bts.N.]handover.power.budget.interval	RW	No	<1-99>,"default"
How often to check for a better cell (SACCH frames).	[bts.N.]handover.power.budget.hysteresis	RW	No	<0-999>,"default"
How many dB stronger must a neighbor be to become a HO candidate.	[bts.N.]handover.maximum.distance	RW	No	<0-9999>,"default"
Maximum Timing-Advance value (i.e. MS distance) before triggering HO.	[bts.N.]handover2.window.rxlev.averaging	RW	No	<1-10>,"default"
How many RxLev measurements to use for averaging.	[bts.N.]handover2.window.rxqual.averaging	RW	No	<1-10>,"default"
How many RxQual measurements to use for averaging.	[bts.N.]handover2.window.rxlev.neighbor.averaging	RW	No	<1-10>,"default"
window rxlev neighbor averaging.	[bts.N.]handover2.power.budget.interval	RW	No	<1-99>,"default"
How many dB stronger must a neighbor be to become a HO candidate.	[bts.N.]handover2.power.budget.hysteresis	RW	No	<0-999>,"default"

Table 35: (continued)

Name	Access	Trap	Value	Comment
How many dB stronger must a neighbor be to become a HO candidate.	[bts.N.]hand	RW	2.maximum.distance	<0-9999>,"default"
Maximum Timing-Advance value (i.e. MS distance) before triggering HO.	[bts.N.]hand	RW	2.assignment	"0","1","default"
Enable or disable in-call channel re-assignment within the same cell.	[bts.N.]hand	RW	2.tdma-measurement	"full","subset","default"
Define measurement set of TDMA frames.	[bts.N.]hand	RW	2.min.rxlev	←110—50>,"default"
How weak may RxLev of an MS become before triggering HO.	[bts.N.]hand	RW	2.min.rxqual	<0-7>,"default"
How bad may RxQual of an MS become before triggering HO.	[bts.N.]hand	RW	2.afs-bias.rxlev	<0-20>,"default"
RxLev improvement bias for AFS over other codecs.	[bts.N.]hand	RW	2.afs-bias.rxqual	<0-7>,"default"
RxQual improvement bias for AFS over other codecs.	[bts.N.]hand	RW	2.min-free-slots.tch-f	<0-9999>,"default"
Minimum free TCH/F timeslots before cell is considered congested.	[bts.N.]hand	RW	2.min-free-slots.tch-h	<0-9999>,"default"
Minimum free TCH/H timeslots before cell is considered congested.	[bts.N.]hand	RW	2.max-handovers	<1-9999>,"default"
Maximum number of concurrent handovers allowed per cell.	[bts.N.]hand	RW	2.penalty-time.max-distance	<0-99999>,"default"
ime to suspend handover for a subscriber after leaving this cell due to exceeding max distance.	[bts.N.]hand	RW	2.penalty-time.failed-ho	<0-99999>,"default"

Table 35: (continued)

Name	Access	Trap	Value	Comment
Time to suspend handover for a subscriber after a failed handover into this cell.	[bts.N.]handover2.penalty-time.failed-assignment	RW	No	<0-99999>,"default"
Time to suspend handover for a subscriber after a failed re-assignment within this cell.	[bts.N.]handover2.retries	RW	No	<0-9>,"default"
Number of times to immediately retry a failed handover/assignment, before a penalty time is applied.	handover2.congestion-check	RW	No	"disabled",<1-999>,"now"
Congestion check interval in seconds, "now" triggers immediate congestion check.	bts.N.neighbor-list.mode	WO	No	"automatic","manual","manual-si5"
Mode of Neighbor List generation.	bts.N.neighbor-list.add	WO	No	<0-1023>
Add to manual neighbor list.	bts.N.neighbor-list.del	WO	No	<0-1023>
Delete from manual neighbor list.	bts.N.neighbor-list.si5-add	WO	No	<0-1023>
Add to manual SI5 neighbor list.	bts.N.neighbor-list.si5-del	WO	No	<0-1023>
Delete from manual SI5 neighbor list.	bts.N.neighbor-list.si2	RO	No	"<arfcn>"
Get space concatenated list of SI2 neighbor ARFCNs.	bts.N.neighbor-list.si5	RO	No	"<arfcn>"
Get space concatenated list of SI5 neighbor ARFCNs.	bts.N.neighbor-list.si2quater.uarfcns	RO	No	"<uarfcn>,<scrambling code>,<diversity bit>"
Get space concatenated list of UARFCN neighbors.	bts.N.neighbor-list.si2quater.earfcns	RO	No	"<earfcn>,<thresh-hi>,<thresh-lo>,<prio>,<qrxlv>,<meas>"
Get space concatenated list of EARFCN neighbors.	bts.N.si2quater-neighbor-list.add.uarfcn	WO	No	"<uarfcn>,<scrambling code>,<diversity bit>"
Add UARFCN neighbor.	bts.N.si2quater-neighbor-list.del.uarfcn	WO	No	"<uarfcn>,<scrambling code>"
Delete UARFCN neighbor.	bts.N.si2quater-neighbor-list.add.earfcn	WO	No	"<earfcn>,<thresh-hi>,<thresh-lo>,<prio>,<qrxlv>,<meas>"

Table 35: (continued)

Name	Access	Trap	Value	Comment
Add EARFCN neighbor.	bts.N.si2quater-neighbor-list.del.earfcn	WO	No	"<earfcn>"

30.1 notification

Setting this variable initiate TRAP "notification" to all the clients connected to control interface with the value supplied in SET operation. This is not intended to be used outside of local systems.

30.2 inform-msc-v1

Setting this variable initiate TRAP "inform-msc-v1" to all connected MSCs control interfaces with the value supplied in SET operation.

30.3 channel-load

Obtain channel load for given BTS. Returns concatenated set of triplets ("<name>,<used>,<total>") for all channel types configured on the BTS. The "<name>" is the channel type. The "<used>" is the number of channels of that type currently in use. The "<total>" is the number of channels of that type configured on the BTS.

30.4 gprs-mode

Set/Get the GPRS mode of the BTS. One of the following is accepted/returned: "none", "gprs", "egprs".

30.5 rf_state

Following triplet is returned: "<oper>,<admin>,<pol>". The "<oper>" might be "operational" or "inoperational" representing different operational states. The "<admin>" might be "locked" or "unlocked" representing administrative status. The "<pol>" might be "off", "on", "grace" or "unknown" representing different RF policies.

30.6 rf_locked

Set/Get RF locked status. The GET operation will return either "0" or "1" depending on the RF lock status. The SET operation will set RF lock status if RF Ctrl is enabled in the BSC Configuration.

30.7 max-power-reduction

Set/Get the value of maximum power reduction. Even values between 0 and 22 are accepted.

30.8 add/del neighbor cell

The control interface allows for editing the neighbor cell configuration. Neighbor cells can be added or removed during runtime. It is also possible to clear the entire neighbor list if necessary.

Table 36: Variables available over control interface

Name	Access	Trap	Value	Comment
bts.N.neighbor-bts.add	WO	No	"<num>"	Add neighbor cell by local BTS number.
bts.N.neighbor-bts.del	WO	No	"<num>"	Delete neighbor cell by local BTS number.
bts.N.neighbor-lac.add	WO	No	"<lac>[-<arfcn>-<bsic>]"	Add neighbor cell by LAC.
bts.N.neighbor-lac.del	WO	No	"<lac>[-<arfcn>-<bsic>]"	Delete neighbor cell by LAC.
bts.N.neighbor-lac-ci.add	WO	No	"<lac>-<ci>[-<arfcn>-<bsic>]"	Add neighbor cell by LAC and CI.
bts.N.neighbor-lac-ci.del	WO	No	"<lac>-<ci>[-<arfcn>-<bsic>]"	Delete neighbor cell by LAC and CI.
bts.N.neighbor-cgi.add	WO	No	"<mcc>-<mnc>-<lac>-<ci>[-<arfcn>-<bsic>]"	Add neighbor cell by cgi.
bts.N.neighbor-cgi.del	WO	No	"<mcc>-<mnc>-<lac>-<ci>[-<arfcn>-<bsic>]"	Delete neighbor cell by cgi.
bts.N.neighbor-cgi-ps.add	WO	No	"<mcc>-<mnc>-<lac>-<rac>-<ci>[-<arfcn>-<bsic>]"	Add neighbor cell by cgi (Packet Switched, with RAC)
bts.N.neighbor-cgi-ps.del	WO	No	"<mcc>-<mnc>-<lac>-<rac>-<ci>[-<arfcn>-<bsic>]"	Delete neighbor cell by cgi (Packet Switched, with RAC).
bts.N.neighbor-clear	WO	No	Ignored	Delete all neighbor cells.

Note
The bsic-number (<bsic>) can also be set to "any" if no explcit bsic shall be given

31 Osmux

`Osmux` is a protocol aimed at multiplexing and transmitting voice and signalling traffic from multiple sources in order to reduce the overall bandwidth consumption. This feature becomes specially meaningful in case of satellite based GSM systems, where the transmission cost on the back-haul is relatively expensive. In such environment, even seemingly small protocol optimizations, eg. message batching and trunking, can result in significant cost reduction.

Full reference document for the osmux protocol can be found here: https://ftp.osmocom.org/docs/latest/osmux-reference.pdf

In Osmocom satellite based GSM networks, the satellite link is envisioned to be in between the BSS and the core network, that is, between the BSC and the MSC (or BSC-NAT). Hence, Osmocom components can make use of `Osmux` protocol to multiplex payload audio streams from call legs between OsmoBSC and OsmoMSC (or OsmoBSCNAT). The MGW attached those components need of course also be aware of `Osmux` existence in order to properly set up the audio data plane.

Under some specific circumstances, the operator may decide to set up the network with a bandwidth-limited (e.g. satellite) link between BTS and BSC. Hence, use of the `Osmux` protocol is also possible between an Osmux capable BTS (like OsmoBTS) and OsmoBSC (and its co-located MGW).

31.1 Osmux and NAT

It is quite usual for satellite based links to use NATs, which means any or both of the two components at each side of the satellite link (BSC and MSC/BSC-NAT) may end up being behind a NAT and being unable to provide the real public address to its peer on the other side of the satellite.

As a result, upon call parameter negotiation (RTP/Osmux IP address and port), those parameters won't be entirely useful and some specific logic needs to be introduced into the network components to circumvent the NAT under those cases.

For instance, if the BSC and its co-located MGW (BSC/MGW from now on) is under a NAT, it may end up providing its private address and port as RTP/Osmux parameters to the MSC/MGW through GSM protocols, but MSC will fail to send any message to that tuple because of the NAT or routing issues (due to IP address being a private address). In that scenario, MSC/MGW needs to be aware that there's a NAT and wait until an RTP/Osmux message arrives from the BSC/MGW host. It then can, from that message source IP address and port (and CID in case of Osmux), discover the real public IP address and port of the peer (BSC/MGW). From that point on, the BSC/MGW punched a hole in the NAT (its connection table is updated) and MSC/MGW is able to send data back to it on that connection.

In order to make use of the features above, OsmoMGW must be made aware explicitly through VTY configuration that its peers are located behind a NAT. This is done through the `osmux peer-behind-nat (on|off)` VTY commands.

If OsmoMGW itself is behind a NAT, it must use the VTY config `rtp keep-alive` (used for both RTP and Osmux) to at least the value `once`, in order for it to punch the hole in its NAT so that its peer can know where to send packets back to it.

Another characteristic of NATs is that they tend to drop connections from their connection tables after some inactivity time, meaning a peer may receive notice about the other end not being available while it actually is. This means the GSM network needs to be configured in a way to ensure inactivity periods are short enough that this cannot occur.

Hence, if OsmoMGW is behind a NAT, it is actually desirable to have the VTY config `rtp keep-alive` configured with the `<1-120>` value in order to force transmission of dummy packets ever few seconds.

Osmux implementations such as OsmoMGW also come with the `osmux dummy` VTY command to enable sending dummy AMR payloads to the peer even if no real data was received (for instance if DTX is used). This is useful under some specific satellite links which were proven to work unreliably if the total throughput in use over the link changes over time. This way throughput resources are kept pre-allocated until they are needed again (audio is received again).

31.2 CID allocation

Each peer (BSC/MGW and MSC/MGW) allocates its own *local CID*, and receives from its peer a *remote CID* (aka the peer's *local CID*) through the used GSM protocol. This *remote CID* is then used to send Osmux frames to that peer.

```
BSC/MGW(localCID=Y,remoteCID=?)<-X--MSC/MGW(localCID=X,remoteCID=?)
BSC/MGW(localCID=Y,remoteCID=X)--Y->MSC/MGW(localCID=X,remoteCID=Y)
```

This way each peer is responsible for allocating and managing their own local address (CID) space. This is basically the same that happens with regular IP address and port in the RTP case (and those also apply when Osmux is used, but an extra identifier, the CID, is allocated).

In an ideal scenario, without NAT, each BSC/MGW would have a public, differentiated and unique IP and port set tuple, And MSC/MGW should be able to identify messages coming from them by easily matching source IP address, port (and CID in Osmux case) against the parameters negotiated during call set up.

In this kind of scenario, MSC/MGW could easily open and manage one Osmux socket per BSC (based on SDP IPaddr and port parameters), with same `<localIPaddr, localPort>` tuple, allowing for 256 Osmux CIDs per BSC and hence call legs per BSC. Each of the peers could actually have more than one Osmux socket towards the other peer, by using a pool of ports or IP addresses, so there's really not limit if required as long as there's a way to infer the initially negotiated `<srcIP, srcPort, dstIP, dstPort, remoteCID>` tuple from the received audio packets.

However, due to some constrains from in between NATs explained in section above, BSC/MGW IP address and port are not a priory known, and could change between different connections coming from it. As a result, it is difficult to infer the entire tuple, so for now MGW needs to allocate its Osmux *local CID* in a clever way, in order to be able to identify the full tuple from it.

Hence, currently OsmoMGW CID allocation implementation shares CID between all connections, which means it can only handle up to 256 concurrent Osmux connections (call legs).

Future work in OsmoMGW (OS#4092) plans to use a set of local ports for Osmux sockets instead of only 1 currently used. This way local ports can be matched against specific `<remoteIP, remotePort>` tuples and have an entire 256 Osmux CID space per `<remoteIP, remotePort>` (aka per peer).

31.3 3GPP AoIP network setup with Osmux

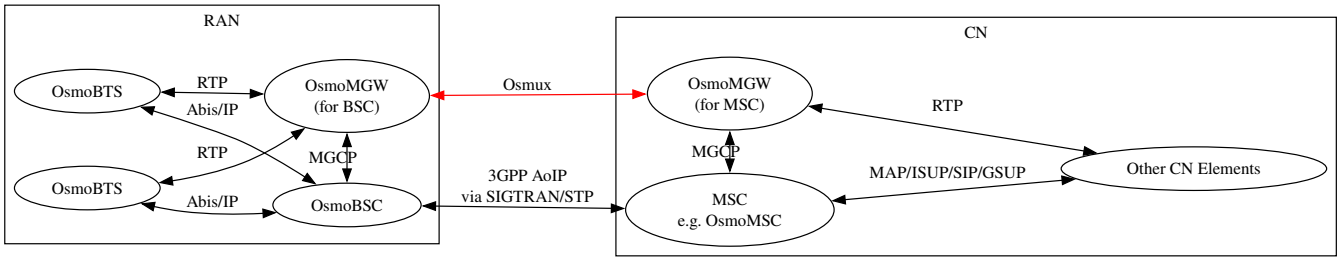


Figure 16: Sample node diagram of a 3GPP AoIP network with Osmux enabled

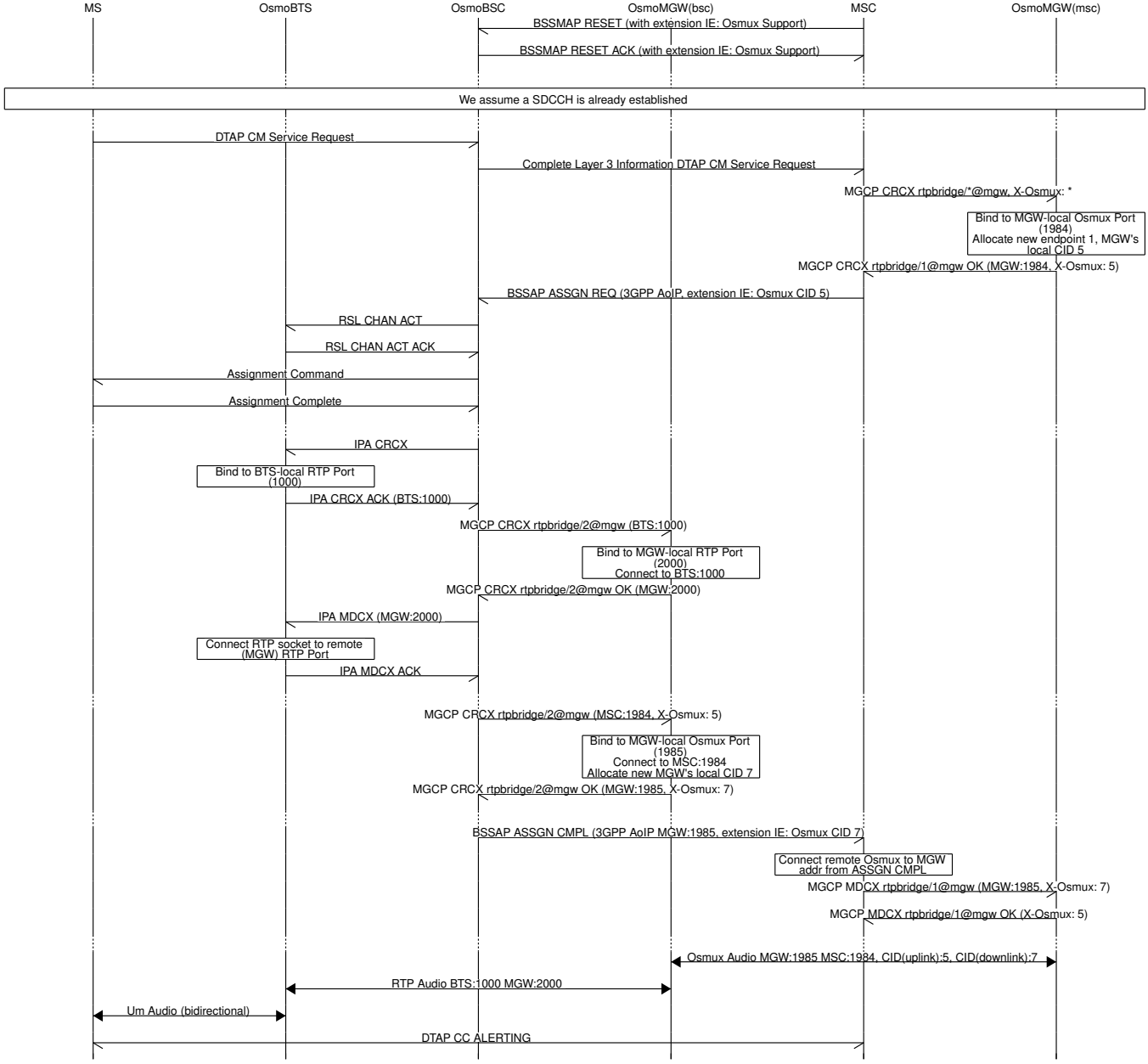


Figure 17: MO-call with Osmux enable on 3GPP AoIP

31.4 SCCPLite network setup with Osmux

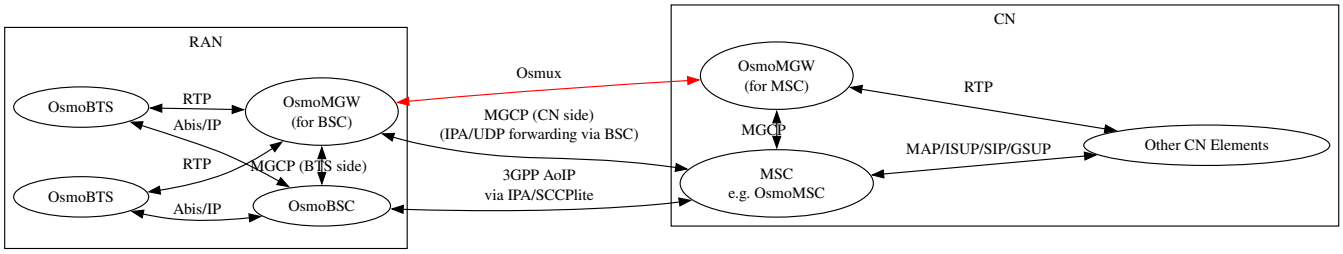


Figure 18: Sample node diagram of a 3GPP AoIP using A/IP with IPA/SCCPlite network with Osmux enabled

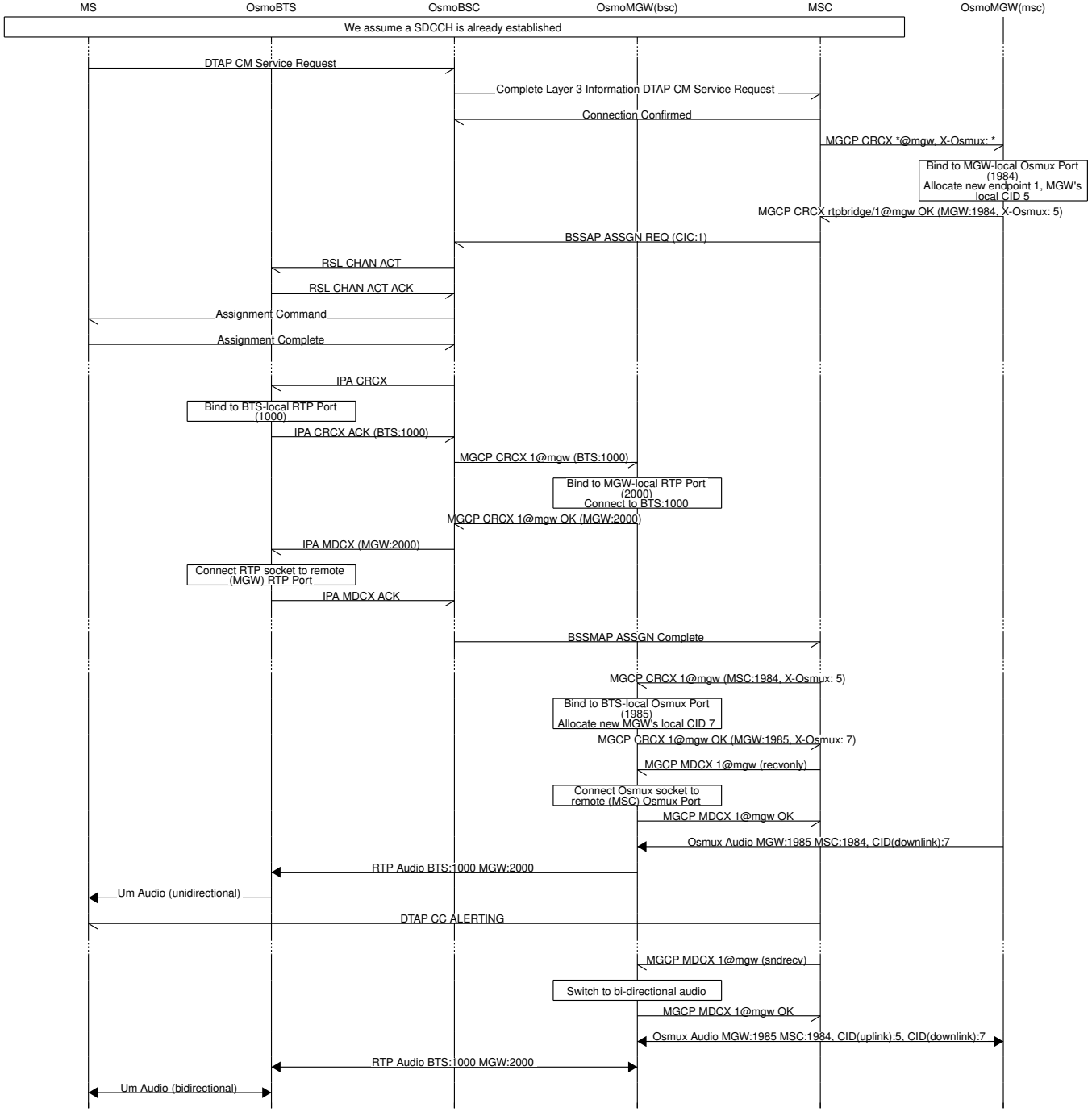


Figure 19: MO-call with Osmux enable on 3GPP AoIP using A/IP with IPA/SCCPlite

31.5 SCCPLite network setup with Osmux + BSC-NAT

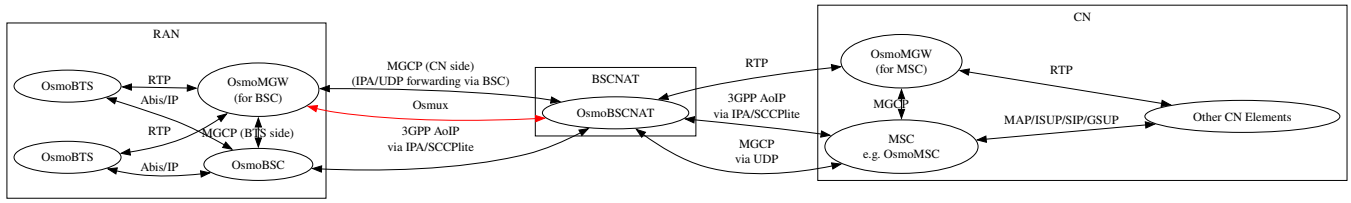


Figure 20: Sample node diagram of a 3GPP AoIP using A/IP with IPA/SCCPlite network and BSC-NAT with Osmux enabled

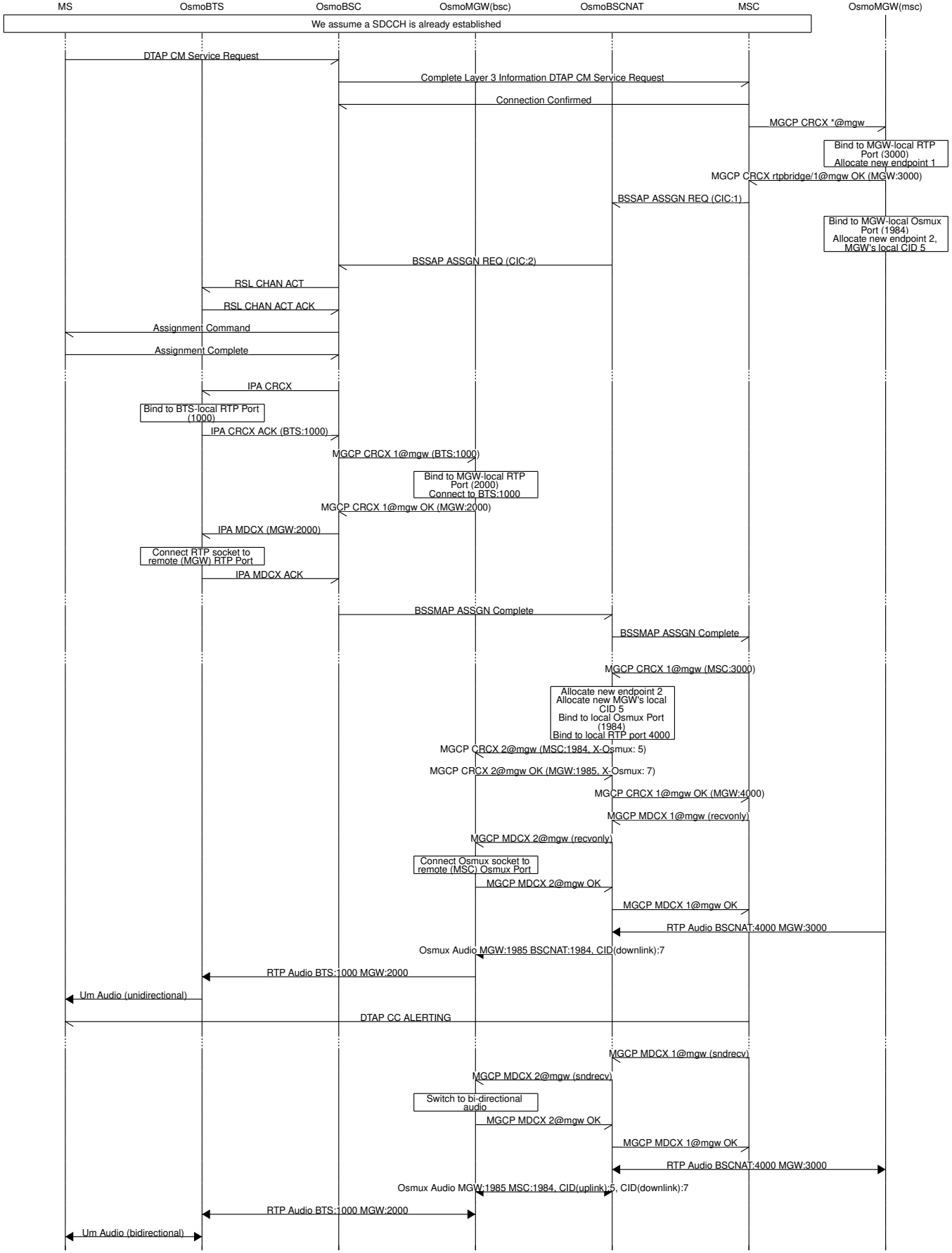


Figure 21: MO-call with Osmux enable on 3GPP AoIP using A/IP with IPA/SCCPlite with a BSC-NAT between BSC and MSC

31.6 Osmux and MGCP

`X-Osmux` indicates to OsmoMGW that a given connection of an `rtpbridge` endpoint has to be configured in order to handle Osmux frames instead of RTP messages on the data plane.

31.6.1 `X-Osmux` Format

The value part of `X-Osmux` must be one integer in range [0..255], or alternatively only on request messages, an asterisk (*) if the value is not yet known.

`X-Osmux` must be issued in the MGCP header section (typically as its last item), before the SDP section starts.

`X-Osmux` can be included inside `CRCX` and `MDCX` request messages, as well as their respective response messages.

In request messages, the value part of `X-Osmux` specifies the CID to be used by OsmoMGW to *send* Osmux frames to the remote peer for that connection, also known as the MGW's *remote CID* or the peer's *local CID*.

In response messages, the value part of `X-Osmux` specifies the CID where OsmoMGW expect to *receive* Osmux frames from the remote peer for that connection, also known as the MGW's *local CID* or the peer's *remote CID*.

Example: X-Osmux format with a known CID 3.

```
X-Osmux: 3
```

Example: X-Osmux format with a wildcard (not yet known) CID.

```
X-Osmux: *
```

31.6.2 `X-Osmux` Considerations

If the MGCP client is willing to use Osmux for a given connection, it shall specify so during `CRCX` time, and not later. If at `CRCX` time the MGCP client doesn't yet know the MGW's *remote CID*, it can use an astersik (*) and provide *remote CID* later within `MDCX` messages.

All subsequent `MDCX` messages sent towards an Osmux connection must contain the original MGW's *remote CID* sent during `CRCX`. The same way, all `MDCX` response shall contain the *local CID* sent during `CRCX`.

The other required connection address parameters, such as IP address, port, and codecs, are negotiated through MGCP and SDP as usual, but in this case the IP address and port specific the Osmux socket IP address and port to use, that together with the Osmux CID conform the entire tuple identifying a Osmux stream.

Since Osmux only supports AMR codec payloads, the SDP must specify use of AMR codec.

Example: CRCX message that instructs OsmoMGW to create an Osmux connection

```
CRCX 189 rtpbridge/1@mgw MGCP 1.0
C: 36
M: sendrecv
X-Osmux: 2

v=0
o=- 36 23 IN IP4 172.18.2.20
s=-
c=IN IP4 1.2.3.4
t=0 0
m=audio 2342 RTP/AVP 112
a=fmtp:112
a=rtpmap:112 AMR/8000/1
a=ptime:20
```

Example: response to CRCX containing the MGW's local CID

```
200 189 OK
I: 07E41584
X-Osmux: 2
Z: rtpbridge/1@mgw

v=0
o=- foo 21 IN IP4 172.18.1.20
s=-
c=IN IP4 172.18.1.20
t=0 0
m=audio 11002 RTP/AVP 112
a=rtpmap:112 AMR/8000
a=ptime:20
```

31.6.3 `X-Osmux` Support

`X-Osmux` is known to be supported by OsmoMGW on the MGCP server side, and by OsmoBSC as well as OsmoMSC on the MGCP client side (through libosmo-mgcp-cli). No other programs supporting this feature are known or envisioned at the time of writing this document.

In OmoMGW, Osmux support is managed through VTY.

Example: Sample config file section with Osmux configuration

```
mgcp
 ...
 osmux on ❶
 osmux bind-ip 172.18.1.20 ❷
 osmux port 1984 ❸
 osmux batch-factor 4 ❹
 osmux dummy on ❺
```

❶ Allow clients to set allocate Osmux connections in `rtpbridge` endpoints, while still allowing RTP connections

❷ Bind the Osmux socket to the provided IP address

❸ Bind the Osmux socket to the provided UDP port

❹ Batch up to 4 RTP payloads of the same stream on each Osmux frame

❺ Periodically send Osmux dummy frames, useful to punch a hole in NATs and maintain connections opened.

31.7 Abis setup with Osmux

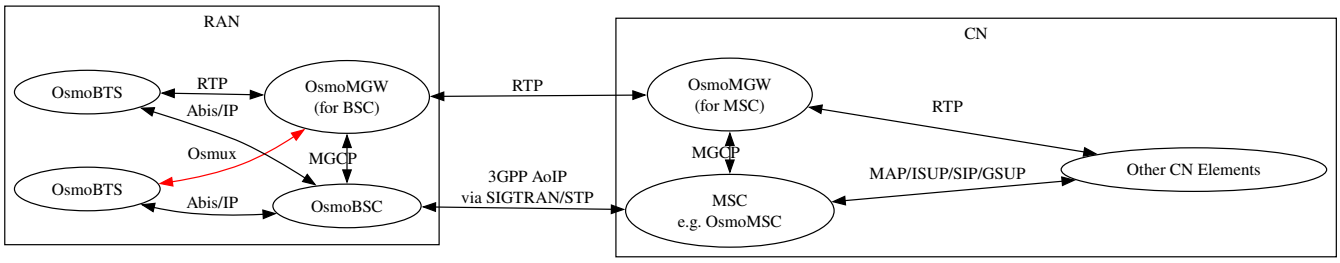


Figure 22: Sample node diagram of Osmux enabled in the Abis interface

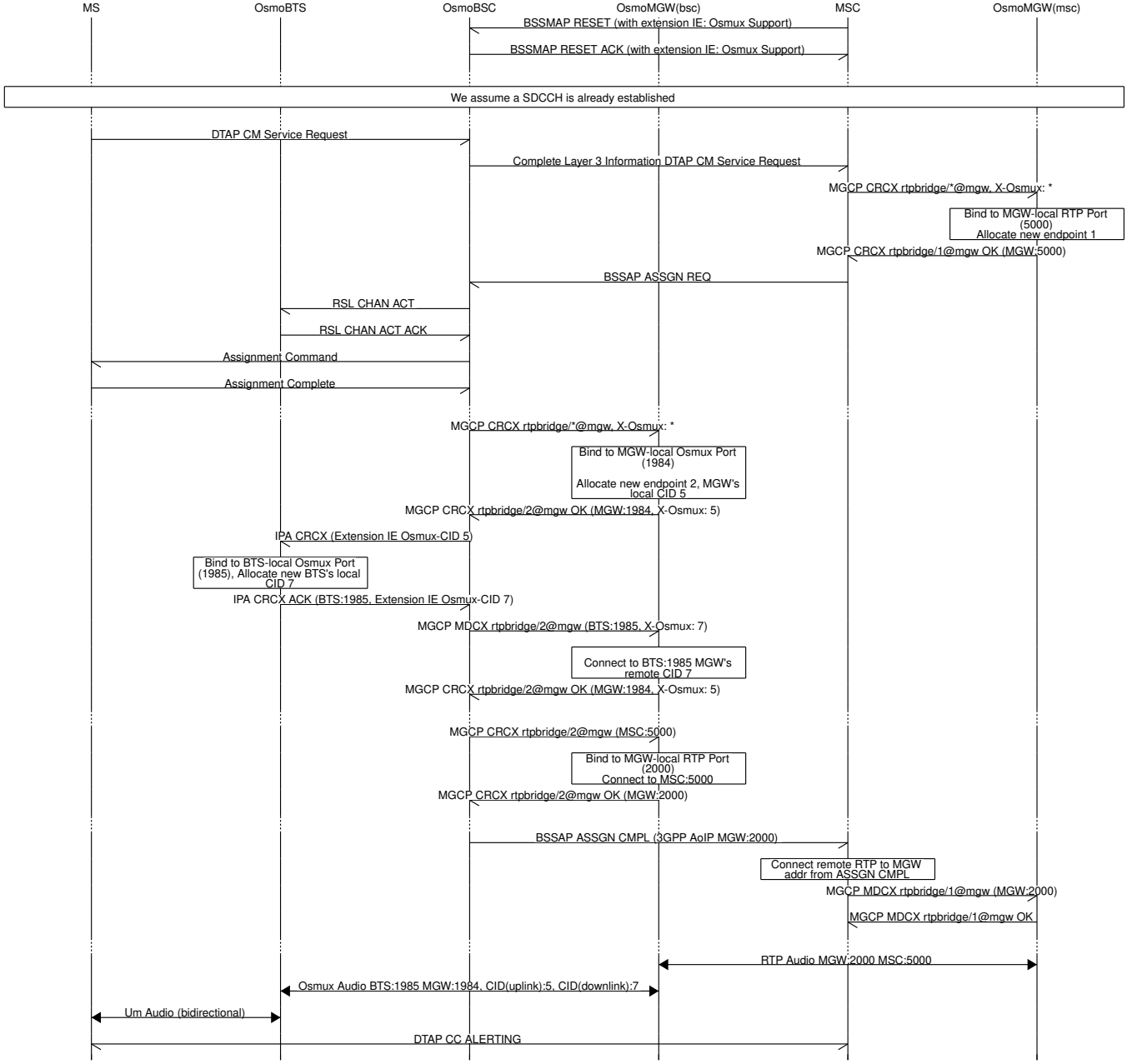


Figure 23: MO-call with Osmux enabled on Abis

31.8 Osmux Support in OsmoBSC

31.8.1 OsmoBSC in a A/IP with IPA/SCCPlite network setup

In this kind of setup, Osmux is transparent to OsmoBSC and no specific configuration is required here, since the CN-side of the BSC-attached MGW is managed directly by the MSC.

So, in this case, only MSC and MGW (both for MSC-attached one and BSC-attached one) need to be configured explicitly.

31.8.2 OsmoBSC in a 3GPP AoIP network setup

Osmux usage in OsmoBSC in managed through the VTY command `osmux (on|off|only)` under the `msc` node. Once enabled (`on` or `only`), OsmoBSC will start appending the vendor specific *Osmux Support* IE in *BSSMAP RESET* and *BSSMAP*

RESET-ACK message towards the MSC in order to announce it supports Osmux. This way, the MSC can decide whether to use Osmux or not based on this information when setting up a call (this time using *Osmux CID* IE). It should be noted that this option should not be enabled unless MSC managing OsmoBSC supports handling this extension IE (like OsmoMSC), a 3rd-party MSC might otherwise refuse the related *RESET/RESET-ACK* messages.

OsmoBSC will behave differently during call set up based on the VTY command presented above:

- `off`: If *BSSMAP Assign Request* from MSC contains *Osmux CID* IE, meaning MSC wants to use Osmux for this call, then OsmoBSC will reject the assignment and the call set up will fail.
- `on`: BSC will support and accept both Osmux and non-Osmux (RTP) upon call set up. If *BSSMAP Assign Request* from MSC contains *Osmux CID* IE, OsmoBSC will instruct its MGW to set up an Osmux connection on the CN-side of the MGCP endpoint, and will provide the MSC with its *recvCID* through the extension IE *Osmux CID* appended to the *BSSMAP Assign Complete* message. On the other hand, if *BSSMAP Assign Request* doesn't contain an *Osmux CID* IE, OsmoBSC will instruct its MGW to set up a regular RTP connection on the CN-side of the MGCP endpoint.
- `only`: Same as per `on`, except that OsmoBSC will accept only Osmux calls on the CN-side, this is, if *BSSMAP Assign Request* from MSC doesn't contain an *Osmux CID* IE, it will reject the assignment and the call set up will fail.

31.8.3 Osmux in the ip.access Abis interface

OsmoBSC can also talk Osmux instead of RTP to an OsmoBTS which supports the feature. Osmux usage agains the BTS in OsmoBSC in managed through the VTY command `osmux (on|off|only)` under the `bts` node.

If a BTS supports Osmux, it may announce the *OSMUX* BTS feature towards the BSC over OML. This way, the OsmoBSC becomes aware that this BTS supports using Osmux to transfer voice call user data when the AMR codec is selected.

It is then up to OsmoBSC to decide whether to use Osmux or not when establishing a new call. If OsmoBSC decides to use Osmux for a given call, it will instruct its co-located MGW to set up an Osmux connection in the endpoint (using the 'X-Osmux extension`) and then it will forward the received Osmux CID to the BTS in the the *IPACC CRCX/MDCX* messages by means of an extra *Osmux CID* IE appended to it. The IP address and port provided in the same messages refer to the address and port where Osmux frames with the provided CID are expected to be received. Similarly, the BTS appends an *Osmux CID* IE to the *IPACC CRCX/MDCX ACK* message it generates, this time with its own local Osmux CID, which OsmoBSC will in turn forward back to the co-located MGW. Same goes for the BTS' local IP address and port where Osmux frames are expected to be received.

OsmoBSC will behave differently during call set up based on the VTY command `use (on|off|only)` under each `bts` node presented above:

- `off`: OsmoBSC will never attempt use of Osmux against this BTS (default).
- `on`: OsmoBSC will use Osmux against the BTS if the BTS announced Osmux support during OML bringup, and if MGW provided a valid Osmux CID during *MGCP CRCX*. Otherwise BSC will simply automatically fall back to using RTP for each call. For non-AMR calls, RTP will always be used.
- `only`: Same as per `on`, except that OsmoBSC will accept only Osmux calls on the BTS-side. This is, if *MGCP CRCX ACK* from MGW doesn't contain an *Osmux CID* IE or *IPACC CRCX ACK* from BSC doesn't contain an *Osmux CID* IE, it will reject the assignment and the call set up will fail. This means also that only AMR calls (`Channel Mode GSM3`) are allowed.

32 VTY Process and Thread management

Most Osmocom programs provide, some support to tune some system settings related to the running process, its threads, its scheduling policies, etc.

All of these settings can be configured through the VTY, either during startup by means of usual config files or through direct human interaction at the telnet VTY interface while the process is running.

32.1 Scheduling Policy

The scheduler to use as well as some of its properties (such as realtime priority) can be configured at any time for the entire process. This sort of functionality is useful in order to increase priority for processes running time-constrained procedures, such as those acting on the Um interface, like *osmo-trx* or *osmo-bts*, where use of this feature is highly recommended.

Example: Set process to use RR scheduler

```
cpu-sched
 policy rr 1 ❶
```

❶ Configure process to use *SCHED_RR* policy with real time priority 1

32.2 CPU-Affinity Mask

Most operating systems allow for some sort of configuration on restricting the amount of CPUs a given process or thread can run on. The procedure is sometimes called as *cpu-pinning* since it allows to keep different processes pinned on a subset of CPUs to make sure the scheduler won't run two CPU-hungry processes on the same CPU.

The set of CPUs where each thread is allowed to run on is expressed by means of a bitmask in hexadecimal representation, where the right most bit relates to CPU 0, and the Nth most significant bit relates to CPU *N-1*. Setting the bit means the process is allowed to run on that CPU, while clearing it means the process is forbidden to run on that CPU.

Hence, for instance a cpu-affinity mask of *0x00* means the thread is not allowed on any CPU, which will cause the thread to stall until a new value is applied. A mask of *0x01* means the thread is only allowed to run on the 1st CPU (CPU 0). A mask of *0xff00* means CPUs 8-15 are allowed, while 0-7 are not.

For single-threaded processes (most of Osmocom are), it is usually enough to set this line in VTY config file as follows:

```
cpu-sched
 cpu-affinity self 0x01 ❶
```

❶ Allow main thread (the one managing the VTY) only on CPU 0

Or otherwise:

```
cpu-sched
 cpu-affinity all 0x01 ❶
```

❶ Allow all threads only on CPU 0

For multi-threaded processes, it may be desired to run some threads on a subset of CPUs while another subset may run on another one. In order to identify threads, one can either use the TID of the thread (each thread has its own PID in Linux), or its specific Thread Name in case it has been set by the application.

The related information on all threads available in the process can be listed through VTY. This allows identifying quickly the different threads, its current cpu-affinity mask, etc.

Example: Get osmo-trx Thread list information from VTY

```
OsmoTRX> show cpu-sched threads
Thread list for PID 338609:
 TID: 338609, NAME: 'osmo-trx-uhd', cpu-affinity: 0x3
 TID: 338610, NAME: 'osmo-trx-uhd', cpu-affinity: 0x3
 TID: 338611, NAME: 'osmo-trx-uhd', cpu-affinity: 0x3
 TID: 338629, NAME: 'osmo-trx-uhd', cpu-affinity: 0x3
 TID: 338630, NAME: 'osmo-trx-uhd', cpu-affinity: 0x3
 TID: 338631, NAME: 'osmo-trx-uhd', cpu-affinity: 0x3
 TID: 338634, NAME: 'UHDAsyncEvent', cpu-affinity: 0x3
```

```
 TID: 338635, NAME: 'TxLower', cpu-affinity: 0x3
 TID: 338636, NAME: 'RxLower', cpu-affinity: 0x3
 TID: 338637, NAME: 'RxUpper0', cpu-affinity: 0x3
 TID: 338638, NAME: 'TxUpper0', cpu-affinity: 0x3
 TID: 338639, NAME: 'RxUpper1', cpu-affinity: 0x3
 TID: 338640, NAME: 'TxUpper1', cpu-affinity: 0x3
```

At runtime, one can change the cpu-affinity mask for a given thread identifying it by either TID or name:

Example: Set CPU-affinity from VTY telnet interface

```
OsmoTRX> cpu-affinity TxLower 0x02 ❶
OsmoTRX> cpu-affinity TxLower 0x03 ❷
```

❶ Allow thread named *TxLower* (*338635*) only on CPU 1

❷ Allow with TID *338636* (*RxLower*) only on CPU 0 and 1

Since thread names are set dynamically by the process during startup or at a later point after creating the thread itself, One may need to specify in the config file that the mask must be applied by the thread itself once being configured rather than trying to apply it immediately. To specify so, the *delay* keyword is using when configuring in the VTY. If the *delay* keyword is not used, the VTY will report and error and fail at startup when trying to apply a cpu-affinity mask for a yet-to-be-created thread.

Example: Set CPU-affinity from VTY config file

```
cpu-sched
 cpu-affinity TxLower 0x01 delay ❶
```

❶ Allow thread named *TxLower* (*338635*) only on CPU 1. It will be applied by the thread itself when created.

33 AoIP message flow examples

The flow diagrams / ladder diagrams of this section are intended to provide some examples on how AoIP procedures work. We hope they will be useful in understanding the interface better and aid in debugging any related issues.

33.1 AoIP interface bring-up

This Figure shows the exchange of messages of a BSC when it is establishing its AoIP interface from scratch, for example because it has just been started up. We assume the BSC/CN has already been fully brought up, so no SCTP/M3U*A bring-up between MSC and STP is dieplayed.

The diagram shows only one possible scenario.

Depending on the MSC implementation, in between the BSC and the MSC there may be either

- a dedicated STP (or multiple replicated STPs)
- no dedicated STP, as the functionality is implemented inside the MSC
- an entire SS7 network between BSC and MSC, with multiple STP, SGW, elements switching messages from the BSCs to the MSCs.

The configuration details that need to be known to the BSC at start-up time are:

- at SCTP level
  - remote IP addresses to which it should establish a SCTP association

  - SCTP port number for M3UA at the STP
- at M3UA level
  - routing key (0 for none)
  - local BSC-side SS7 point code
  - remote MSC-side SS7 point code

There possibly may be more configuration details, such as

- multiple local and/or remote IP addresses for SCTP multi-homing
- a fixed local (BSC side) IP address and/or SCTP port (default: dynamic/random)

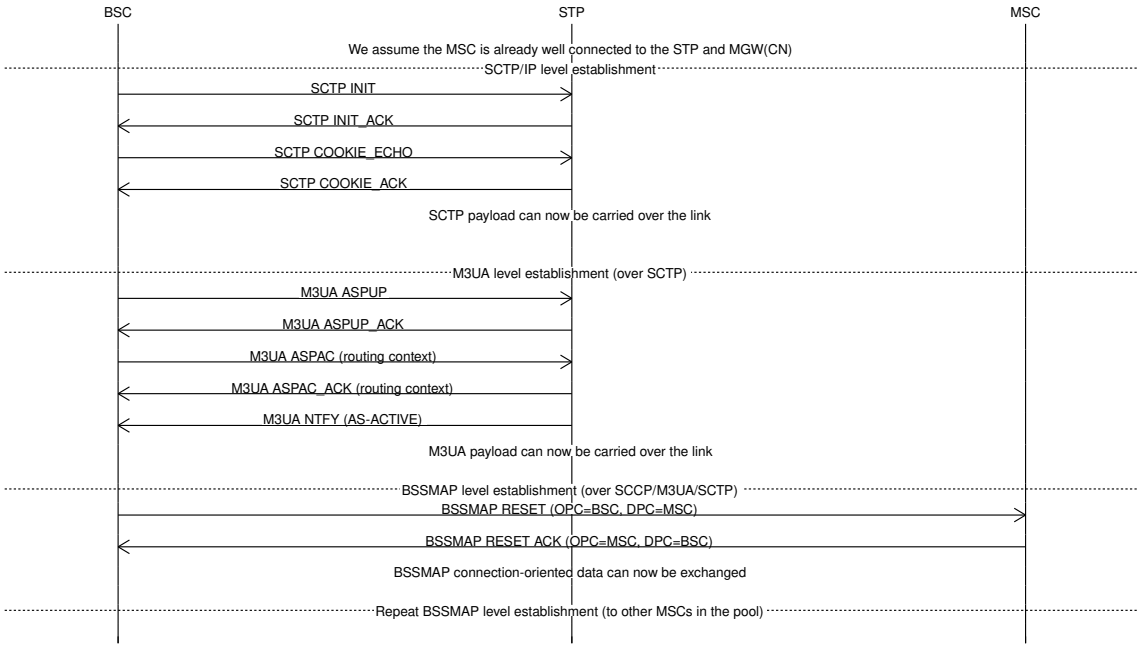


Figure 24: AoIP interface bring-up between BSC and MSC

For the purpose of clairty, SCTP-level acknowledgement chunks are not shown. Those are automatically generated by the receiver for every DATA chunk received in order to confirm its reception and to allow the transmitter to re-transmit in case of packet loss.

33.1.1 SCTP multi-homing

If SCTP multi-homing is used, the additional IP addresses are typically exchanged via additional information elements in the INIT/INIT_ACK chunks at connection establishment. They may also change at a later point.

33.1.2 MSC pooling

If there is MSC pooling configured, there is typically still only one M3UA ASP / SCTP association. The different MSCs are addressed on the SCCP point-code level. It's the STPs job to route the messages based on point codes to the respective MSC.

The BSC will try to establish BSSAP to each of the MSCs in the pool, using a separate BSSAP reset procedure to each of the pool members point code.

See the Chapter *MSC Pooling* in the OsmoBSC user manual for configuration examples of this situation.

33.2 MO call establishment on AoIP with user plane

The following figure shows a simplified version of the messages between MS, OsmoBTS, OsmoBSC, OsmoMGW@BSC, MSC[-Server] and MSC-MGW in during the establishment and release of a MO voice call. Particular focus is given on messages related to the establishment of the RTP based user plane.

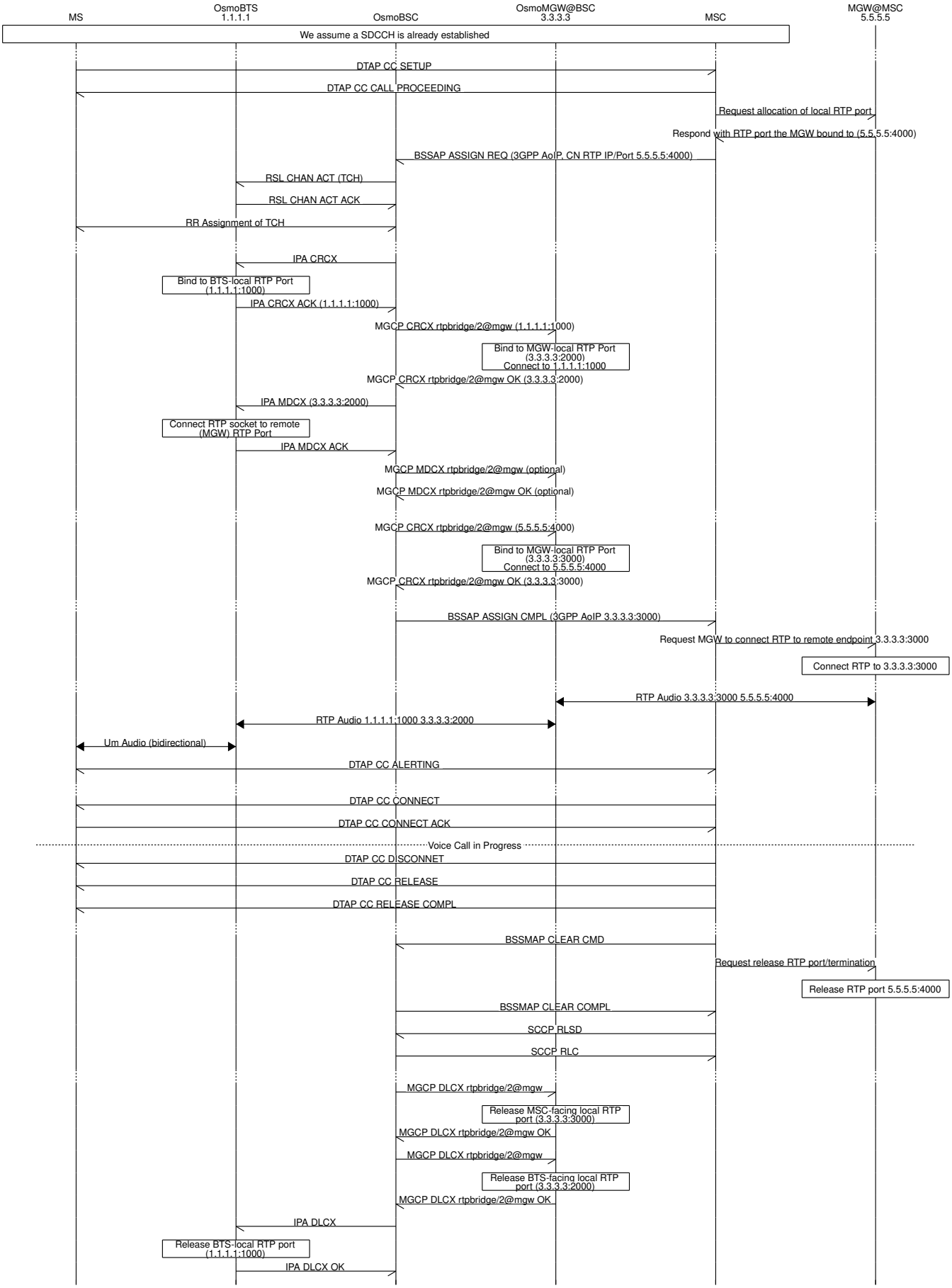
The fact whether or not the RAN or the CN use media gateways, how they control their respective media gateway, and whether there are multiple media gateways for load distribution is a private implementation detail of either RAN or CN. Either side does not need to know the internal structure of the other side, since the RTP endpoint parameters are signaled for each call individually over the A interface.

The signaling between the BSC-colocated MGW and OsmoBSC is IETF MGCP (Media Gateway Control Protocol).

The signaling between the MSC[-Server] and the MGW is internal to the CN. It is typically based on MEGACO/H.248.

As only the BSC and the MSC exchange 3GPP specified signaling messages, there is no direct interaction between the RAN and the CN side MGW. They only exchange RTP and associated RTCP.

In many real deployments, OsmoMGW will have a different IP address on the BTS/Abis facing interface than on the MSC/A facing interface. As a simplification, this has been omitted in the figure.


MS
OsmoBTS 1.1.1.1
OsmoBSC
OsmoMGW@BSC 3.3.3.3
MSC
MGW@MSC 5.5.5.5
We assume a SDCCH is already established
DTAP CC SETUP
DTAP CC CALL PROCEEDING
Request allocation of local RTP port
Respond with RTP port the MGW bound to (5.5.5.5:4000)
BSSAP ASSIGN REQ (3GPP AoIP, CN RTP IP/Port 5.5.5.5:4000)
RSL CHAN ACT (TCH)
RSL CHAN ACT ACK
RR Assignment of TCH
IPA CRCX
Bind to BTS-local RTP Port (1.1.1.1:1000)
IPA CRCX ACK (1.1.1.1:1000)
MGCP CRCX rtpbridge/2@mgw (1.1.1.1:1000)
Bind to MGW-local RTP Port (3.3.3.3:2000) Connect to 1.1.1.1:1000
MGCP CRCX rtpbridge/2@mgw OK (3.3.3.3:2000)
IPA MDCX (3.3.3.3:2000)
Connect RTP socket to remote (MGW) RTP Port
IPA MDCX ACK
MGCP MDCX rtpbridge/2@mgw (optional)
MGCP MDCX rtpbridge/2@mgw OK (optional)
MGCP CRCX rtpbridge/2@mgw (5.5.5.5:4000)
Bind to MGW-local RTP Port (3.3.3.3:3000) Connect to 5.5.5.5:4000
MGCP CRCX rtpbridge/2@mgw OK (3.3.3.3:3000)
BSSAP ASSIGN CMPL (3GPP AoIP 3.3.3.3:3000)
Request MGW to connect RTP to remote endpoint 3.3.3.3:3000
Connect RTP to 3.3.3.3:3000
RTP Audio 3.3.3.3:3000 5.5.5.5:4000
RTP Audio 1.1.1.1:1000 3.3.3.3:2000
Um Audio (bidirectional)
DTAP CC ALERTING
DTAP CC CONNECT
DTAP CC CONNECT ACK
Voice Call in Progress
DTAP CC DISCONNET
DTAP CC RELEASE
DTAP CC RELEASE COMPL
BSSMAP CLEAR CMD
Request release RTP port/termination
Release RTP port 5.5.5.5:4000
BSSMAP CLEAR COMPL
SCCP RLSD
SCCP RLC
MGCP DLCX rtpbridge/2@mgw
Release MSC-facing local RTP port (3.3.3.3:3000)
MGCP DLCX rtpbridge/2@mgw OK
MGCP DLCX rtpbridge/2@mgw
Release BTS-facing local RTP port (3.3.3.3:2000)
MGCP DLCX rtpbridge/2@mgw OK
IPA DLCX
Release BTS-local RTP port (1.1.1.1:1000)
IPA DLCX OK

34 Glossary

2FF
2nd Generation Form Factor; the so-called plug-in SIM form factor

3FF
3rd Generation Form Factor; the so-called microSIM form factor

3GPP
3rd Generation Partnership Project

4FF
4th Generation Form Factor; the so-called nanoSIM form factor

A Interface
Interface between BTS and BSC, traditionally over E1 (*3GPP TS 48.008* [3gpp-ts-48-008])

A3/A8
Algorithm 3 and 8; Authentication and key generation algorithm in GSM and GPRS, typically COMP128v1/v2/v3 or MILENAGE are typically used

A5
Algorithm 5; Air-interface encryption of GSM; currently only A5/0 (no encryption), A5/1 and A5/3 are in use

Abis Interface
Interface between BTS and BSC, traditionally over E1 (*3GPP TS 48.058* [3gpp-ts-48-058] and *3GPP TS 52.021* [3gpp-ts-52-021])

ACC
Access Control Class; every BTS broadcasts a bit-mask of permitted ACC, and only subscribers with a SIM of matching ACC are permitted to use that BTS

AGCH
Access Grant Channel on Um interface; used to assign a dedicated channel in response to RACH request

AGPL
GNU Affero General Public License, a copyleft-style Free Software License

AQPSK
Adaptive QPSK, a modulation scheme used by VAMOS channels on Downlink

ARFCN
Absolute Radio Frequency Channel Number; specifies a tuple of uplink and downlink frequencies

AUC
Authentication Center; central database of authentication key material for each subscriber

BCCH
Broadcast Control Channel on Um interface; used to broadcast information about Cell and its neighbors

BCC
Base Station Color Code; short identifier of BTS, lower part of BSIC

BTS
Base Transceiver Station

BSC
Base Station Controller

BSIC
Base Station Identity Code; 16bit identifier of BTS within location area

BSSGP
Base Station Subsystem Gateway Protocol (*3GPP TS 48.018* [3gpp-ts-48-018])

BVCI
BSSGP Virtual Circuit Identifier

CBC
Cell Broadcast Centre; central entity of Cell Broadcast service

CBCH
Cell Broadcast Channel; used to transmit Cell Broadcast SMS (SMS-CB)

CBS
Cell Broadcast Service

CBSP
Cell Broadcast Service Protocol (*3GPP TS 48.049* [3gpp-ts-48-049])

CC
Call Control; Part of the GSM Layer 3 Protocol

CCCH
Common Control Channel on Um interface; consists of RACH (uplink), BCCH, PCH, AGCH (all downlink)

Cell
A cell in a cellular network, served by a BTS

CEPT
Conférence européenne des administrations des postes et des télécommunications; European Conference of Postal and Telecommunications Administrations.

CGI
Cell Global Identifier comprised of MCC, MNC, LAC and BSIC

CSFB
Circiut-Switched Fall Back; Mechanism for switching from LTE/EUTRAN to UTRAN/GERAN when circuit-switched services such as voice telephony are required.

dB
deci-Bel; relative logarithmic unit

dBm
deci-Bel (milliwatt); unit of measurement for signal strength of radio signals

DHCP
Dynamic Host Configuration Protocol (*IETF RFC 2131* [ietf-rfc2131])

downlink
Direction of messages / signals from the network core towards the mobile phone

DSCP
Differentiated Services Code Point (*IETF RFC 2474* [ietf-rfc2474])

DSP
Digital Signal Processor

dvnixload
Tool to program UBL and the Bootloader on a sysmoBTS

EDGE
Enhanced Data rates for GPRS Evolution; Higher-speed improvement of GPRS; introduces 8PSK

EGPRS
Enhanced GPRS; the part of EDGE relating to GPRS services

EIR
Equipment Identity Register; core network element that stores and manages IMEI numbers

ESME
External SMS Entity; an external application interfacing with a SMSC over SMPP

ETSI
European Telecommunications Standardization Institute

FPGA
Field Programmable Gate Array; programmable digital logic hardware

Gb
Interface between PCU and SGSN in GPRS/EDGE network; uses NS, BSSGP, LLC

GERAN
GPRS/EDGE Radio Access Network

GFDL
GNU Free Documentation License; a copyleft-style Documentation License

GGSN
GPRS Gateway Support Node; gateway between GPRS and external (IP) network

GMSK
Gaussian Minimum Shift Keying; modulation used for GSM and GPRS

GPL
GNU General Public License, a copyleft-style Free Software License

Gp
Gp interface between SGSN and GGSN; uses GTP protocol

GPRS
General Packet Radio Service; the packet switched 2G technology

GPS
Global Positioning System; provides a highly accurate clock reference besides the global position

GSM
Global System for Mobile Communications. ETSI/3GPP Standard of a 2G digital cellular network

GSMTAP
GSM tap; pseudo standard for encapsulating GSM protocol layers over UDP/IP for analysis

GSUP
Generic Subscriber Update Protocol. Osmocom-specific alternative to TCAP/MAP

GT
Global Title; an address in SCCP

GTP
GPRS Tunnel Protocol; used between SGSN and GGSN

HLR
Home Location Register; central subscriber database of a GSM network

HNB-GW
Home NodeB Gateway. Entity between femtocells (Home NodeB) and CN in 3G/UMTS.

HPLMN
Home PLMN; the network that has issued the subscriber SIM and has his record in HLR

IE
Information Element

IMEI
International Mobile Equipment Identity; unique 14-digit decimal number to globally identify a mobile device, optionally with a 15th checksum digit

IMEISV
IMEI software version; unique 14-digit decimal number to globally identify a mobile device (same as IMEI) plus two software version digits (total digits: 16)

IMSI
International Mobile Subscriber Identity; 15-digit unique identifier for the subscriber/SIM; starts with MCC/MNC of issuing operator

IP
Internet Protocol (*IETF RFC 791* [ietf-rfc791])

IPA
ip.access GSM over IP protocol; used to multiplex a single TCP connection

Iu
Interface in 3G/UMTS between RAN and CN

IuCS
Iu interface for circuit-switched domain. Used in 3G/UMTS between RAN and MSC

IuPS
Iu interface for packet-switched domain. Used in 3G/UMTS between RAN and SGSN

LAC
Location Area Code; 16bit identifier of Location Area within network

LAPD
Link Access Protocol, D-Channel (*ITU-T Q.921* [itu-t-q921])

LAPDm
Link Access Protocol Mobile (*3GPP TS 44.006* [3gpp-ts-44-006])

LLC
Logical Link Control; GPRS protocol between MS and SGSN (*3GPP TS 44.064* [3gpp-ts-44-064])

Location Area
Location Area; a geographic area containing multiple BTS

LU
Location Updating; can be of type IMSI-Attach or Periodic. Procedure that indicates a subscriber's physical presence in a given radio cell.

M2PA
MTP2 Peer-to-Peer Adaptation; a SIGTRAN Variant (*RFC 4165* [ietf-rfc4165])

M2UA
MTP2 User Adaptation; a SIGTRAN Variant (*RFC 3331* [ietf-rfc3331])

M3UA
MTP3 User Adaptation; a SIGTRAN Variant (*RFC 4666* [ietf-rfc4666])

MCC
Mobile Country Code; unique identifier of a country, e.g. 262 for Germany

MFF
Machine-to-Machine Form Factor; a SIM chip package that is soldered permanently onto M2M device circuit boards.

MGW
Media Gateway

MM
Mobility Management; part of the GSM Layer 3 Protocol

MNC
Mobile Network Code; identifies network within a country; assigned by national regulator

MNCC
Mobile Network Call Control; Unix domain socket based Interface between MSC and external call control entity like osmo-sip-connector

MNO
Mobile Network Operator; operator with physical radio network under his MCC/MNC

MO
Mobile Originated. Direction from Mobile (MS/UE) to Network

MS
Mobile Station; a mobile phone / GSM Modem

MSC
Mobile Switching Center; network element in the circuit-switched core network

MSC pool
A number of redundant MSCs serving the same core network, which a BSC / RNC distributes load across; see also the "MSC Pooling" chapter in OsmoBSC's user manual [userman-osmobsc] and *3GPP TS 23.236* [3gpp-ts-23-236]

MSISDN
Mobile Subscriber ISDN Number; telephone number of the subscriber

MT
Mobile Terminated. Direction from Network to Mobile (MS/UE)

MTP
Message Transfer Part; SS7 signaling protocol (*ITU-T Q.701* [itu-t-q701])

MVNO
Mobile Virtual Network Operator; Operator without physical radio network

NCC
Network Color Code; assigned by national regulator

NITB
Network In The Box; combines functionality traditionally provided by BSC, MSC, VLR, HLR, SMSC functions; see OsmoNITB

NRI
Network Resource Indicator, typically 10 bits of a TMSI indicating which MSC of an MSC pool attached the subscriber; see also the "MSC Pooling" chapter in OsmoBSC's user manual [userman-osmobsc] and *3GPP TS 23.236* [3gpp-ts-23-236]

NSEI
NS Entity Identifier

NVCI
NS Virtual Circuit Identifier

NWL
Network Listen; ability of some BTS to receive downlink from other BTSs

NS
Network Service; protocol on Gb interface (*3GPP TS 48.016* [3gpp-ts-48-016])

OCXO
Oven Controlled Crystal Oscillator; very high precision oscillator, superior to a VCTCXO

OML
Operation & Maintenance Link (ETSI/*3GPP TS 52.021* [3gpp-ts-52-021])

OpenBSC
Open Source implementation of GSM network elements, specifically OsmoBSC, OsmoNITB, OsmoSGSN

OpenGGSN
Open Source implementation of a GPRS Packet Control Unit

OpenVPN
Open-Source Virtual Private Network; software employed to establish encrypted private networks over untrusted public networks

Osmocom
Open Source MObile COMmunications; collaborative community for implementing communications protocols and systems, including GSM, GPRS, TETRA, DECT, GMR and others

OsmoBSC
Open Source implementation of a GSM Base Station Controller

OsmoNITB
Open Source implementation of a GSM Network In The Box, combines functionality traditionally provided by BSC, MSC, VLR, HLR, AUC, SMSC

OsmoSGSN
Open Source implementation of a Serving GPRS Support Node

OsmoPCU
Open Source implementation of a GPRS Packet Control Unit

OTA
Over-The-Air; Capability of operators to remotely reconfigure/reprogram ISM/USIM cards

PC
Point Code; an address in MTP

PCH
Paging Channel on downlink Um interface; used by network to page an MS

PCP
Priority Code Point (*IEEE 802.1Q* [?])

PCU
Packet Control Unit; used to manage Layer 2 of the GPRS radio interface

PDCH
Packet Data Channel on Um interface; used for GPRS/EDGE signalling + user data

PIN
Personal Identification Number; a number by which the user authenticates to a SIM/USIM or other smart card

PLMN
Public Land Mobile Network; specification language for a single GSM network

PUK
PIN Unblocking Code; used to unblock a blocked PIN (after too many wrong PIN attempts)

RAC
Routing Area Code; 16bit identifier for a Routing Area within a Location Area

RACH
Random Access Channel on uplink Um interface; used by MS to request establishment of a dedicated channel

RAM
Remote Application Management; Ability to remotely manage (install, remove) Java Applications on SIM/USIM Card

RF
Radio Frequency

RFM
Remote File Management; Ability to remotely manage (write, read) files on a SIM/USIM card

Roaming
Procedure in which a subscriber of one network is using the radio network of another network, often in different countries; in some countries national roaming exists

Routing Area
Routing Area; GPRS specific sub-division of Location Area

RR
Radio Resources; Part of the GSM Layer 3 Protocol

RSL
Radio Signalling Link (*3GPP TS 48.058* [3gpp-ts-48-058])

RTP
Real-Time Transport Protocol (*IETF RFC 3550* [ietf-rfc3550]); Used to transport audio/video streams over UDP/IP

SACCH
Slow Associate Control Channel on Um interface; bundled to a TCH or SDCCH, used for signalling in parallel to active dedicated channel

SCCP
Signaling Connection Control Part; SS7 signaling protocol (*ITU-T Q.711* [itu-t-q711])

SDCCH
Slow Dedicated Control Channel on Um interface; used for signalling and SMS transport in GSM

SDK
Software Development Kit

SGs
Interface between MSC (GSM/UMTS) and MME (LTE/EPC) to facilitate CSFB and SMS.

SGSN
Serving GPRS Support Node; Core network element for packet-switched services in GSM and UMTS.

SIGTRAN
Signaling Transport over IP (*IETF RFC 2719* [ietf-rfc2719])

SIM
Subscriber Identity Module; small chip card storing subscriber identity

Site
A site is a location where one or more BTSs are installed, typically three BTSs for three sectors

SMPP
Short Message Peer-to-Peer; TCP based protocol to interface external entities with an SMSC

SMSC
Short Message Service Center; store-and-forward relay for short messages

SS7
Signaling System No. 7; Classic digital telephony signaling system

SS
Supplementary Services; query and set various service parameters between subscriber and core network (e.g. USSD, 3rd-party calls, hold/retrieve, advice-of-charge, call deflection)

SSH
Secure Shell; *IETF RFC 4250* [ietf-rfc4251] to 4254

SSN
Sub-System Number; identifies a given SCCP Service such as MSC, HLR

STP
Signaling Transfer Point; A Router in SS7 Networks

SUA
SCCP User Adaptation; a SIGTRAN Variant (*RFC 3868* [ietf-rfc3868])

syslog
System logging service of UNIX-like operating systems

System Information
A set of downlink messages on the BCCH and SACCH of the Um interface describing properties of the cell and network

TCH
Traffic Channel; used for circuit-switched user traffic (mostly voice) in GSM

TCP
Transmission Control Protocol; (*IETF RFC 793* [ietf-rfc793])

TFTP
Trivial File Transfer Protocol; (*IETF RFC 1350* [ietf-rfc1350])

TOS
Type Of Service; bit-field in IPv4 header, now re-used as DSCP (*IETF RFC 791* [ietf-rfc791])

TRX
Transceiver; element of a BTS serving a single carrier

TS
Technical Specification

u-Boot
Boot loader used in various embedded systems

UBI
An MTD wear leveling system to deal with NAND flash in Linux

UBL
Initial bootloader loaded by the TI Davinci SoC

UDP
User Datagram Protocol (*IETF RFC 768* [ietf-rfc768])

UICC
Universal Integrated Chip Card; A smart card according to *ETSI TR 102 216* [etsi-tr102216]

Um interface
U mobile; Radio interface between MS and BTS

uplink
Direction of messages: Signals from the mobile phone towards the network

USIM
Universal Subscriber Identity Module; application running on a UICC to provide subscriber identity for UMTS and GSM networks

USSD
Unstructured Supplementary Service Data; textual dialog between subscriber and core network, e.g. **100 → Your extension is 1234*

VAMOS
Voice services over Adaptive Multi-user channels on One Slot; an optional extension for GSM specified in Release 9 of 3GPP GERAN specifications (*3GPP TS 48.018* [3gpp-ts-48-018]) allowing two independent UEs to transmit and receive simultaneously on traffic channels

VCTCXO
Voltage Controlled, Temperature Compensated Crystal Oscillator; a precision oscillator, superior to a classic crystal oscillator, but inferior to an OCXO

VLAN
Virtual LAN in the context of Ethernet (*IEEE 802.1Q* [ieee-802.1q])

VLR
Visitor Location Register; volatile storage of attached subscribers in the MSC

VPLMN
Visited PLMN; the network in which the subscriber is currently registered; may differ from HPLMN when on roaming

VTY
Virtual TeletYpe; a textual command-line interface for configuration and introspection, e.g. the OsmoBSC configuration file as well as its telnet link on port 4242

A Osmocom TCP/UDP Port Numbers

The Osmocom GSM system utilizes a variety of TCP/IP based protocols. The table below provides a reference as to which port numbers are used by which protocol / interface.

Table 37: TCP/UDP port numbers

L4 Protocol	Port Number	Purpose	Software
UDP	1984	Osmux	osmo-mgw, osmo-bts
UDP	2427	MGCP GW	osmo-bsc_mgcp, osmo-mgw
TCP	2775	SMPP (SMS interface for external programs)	osmo-nitb
TCP	3002	A-bis/IP OML	osmo-bts, osmo-bsc, osmo-nitb
TCP	3003	A-bis/IP RSL	osmo-bts, osmo-bsc, osmo-nitb
TCP	4227	telnet (VTY)	osmo-pcap-client
TCP	4228	telnet (VTY)	osmo-pcap-server
TCP	4236	Control Interface	osmo-trx
TCP	4237	telnet (VTY)	osmo-trx
TCP	4238	Control Interface	osmo-bts
TCP	4239	telnet (VTY)	osmo-stp
TCP	4240	telnet (VTY)	osmo-pcu
TCP	4241	telnet (VTY)	osmo-bts
TCP	4242	telnet (VTY)	osmo-nitb, osmo-bsc, cellmgr-ng
TCP	4243	telnet (VTY)	osmo-bsc_mgcp, osmo-mgw
TCP	4244	telnet (VTY)	osmo-bsc_nat
TCP	4245	telnet (VTY)	osmo-sgsn
TCP	4246	telnet (VTY)	osmo-gbproxy
TCP	4247	telnet (VTY)	OsmocomBB
TCP	4249	Control Interface	osmo-nitb, osmo-bsc
TCP	4250	Control Interface	osmo-bsc_nat
TCP	4251	Control Interface	osmo-sgsn
TCP	4252	telnet (VTY)	sysmobts-mgr
TCP	4253	telnet (VTY)	osmo-gtphub
TCP	4254	telnet (VTY)	osmo-msc

Table 37: (continued)

L4 Protocol	Port Number	Purpose	Software
TCP	4255	Control Interface	osmo-msc
TCP	4256	telnet (VTY)	osmo-sip-connector
TCP	4257	Control Interface	osmo-ggsn, ggsn (OpenGGSN)
TCP	4258	telnet (VTY)	osmo-hlr
TCP	4259	Control Interface	osmo-hlr
TCP	4260	telnet (VTY)	osmo-ggsn
TCP	4261	telnet (VTY)	osmo-hnbgw
TCP	4262	Control Interface	osmo-hnbgw
TCP	4263	Control Interface	osmo-gbproxy
TCP	4264	telnet (VTY)	osmo-cbc
TCP	4265	Control Interface	osmo-cbc
TCP	4266	D-GSM MS Lookup: mDNS serve	osmo-hlr
TCP	4267	Control Interface	osmo-mgw
TCP	4268	telnet (VTY)	osmo-uecups
SCTP	4268	UECUPS	osmo-uecups
TCP	4269	telnet (VTY)	osmo-e1d
TCP	4270	telnet (VTY)	osmo-isdntap
TCP	4271	telnet (VTY)	osmo-smlc
TCP	4272	Control Interface	osmo-smlc
TCP	4273	telnet (VTY)	osmo-hnodeb
TCP	4274	Control Interface	osmo-hnodeb
TCP	4275	telnet (VTY)	osmo-upf
TCP	4276	Control Interface	osmo-upf
TCP	4277	telnet (VTY)	osmo-pfcp-tool
TCP	4278	Control Interface	osmo-pfcp-tool
UDP	4729	GSMTAP	Almost every osmocom project
TCP	5000	A/IP	osmo-bsc, osmo-bsc_nat
UDP	23000	GPRS-NS over IP default port	osmo-pcu, osmo-sgsn, osmo-gbproxy
TCP	48049	BSC-CBC (CBSP) default port	osmo-bsc, osmo-cbc

B Bibliography / References

B.0.0.0.1 References

[1] [userman-ice1usb] Osmocom Project: icE1usb User Manual.

[2] [userman-ogt] Pau Espin: osmo-gsm-tester User Manual.

[3] [userman-remsim] Harald Welte: osmo-remsim User Manual.

[4] [osmobts-abis-spec] Neels Hofmeyr & Harald Welte. OsmoBTS Abis Protocol Specification. https://ftp.osmocom.org/docs/latest/osmobts-abis.pdf

[5] [userman-osmobsc] Osmocom Project: OsmoBSC User Manual. https://ftp.osmocom.org/docs/latest/osmobsc-usermanual.pdf

[6] [vty-ref-osmobsc] Osmocom Project: OsmoBSC VTY Reference Manual. https://ftp.osmocom.org/docs/-latest/osmobsc-vty-reference.pdf

[7] [userman-osmobts] Osmocom Project: OsmoBTS User Manual. https://ftp.osmocom.org/docs/latest/osmobts-usermanual.pdf

[8] [vty-ref-osmobts] Osmocom Project: OsmoBTS VTY Reference Manual. https://ftp.osmocom.org/docs/-latest/osmobts-trx-vty-reference.pdf https://ftp.osmocom.org/docs/latest/osmobts-sysmo-vty-reference.pdf https://ftp.osmocom.org/docs/latest/osmobts-lc15-vty-reference.pdf https://ftp.osmocom.org/docs/latest/-osmobts-oc2g-vty-reference.pdf https://ftp.osmocom.org/docs/latest/osmobts-octphy-vty-reference.pdf https://ftp.osmocom.org/docs/latest/osmobts-virtual-vty-reference.pdf

[9] [userman-osmocbc] Osmocom Project: OsmoCBC User Manual. https://ftp.osmocom.org/docs/latest/-osmocbc-usermanual.pdf

[10] [vty-ref-osmocbc] Osmocom Project: OsmoCBC VTY Reference Manual. https://ftp.osmocom.org/docs/-latest/osmocbc-vty-reference.pdf

[11] [userman-osmogbproxy] Osmocom Project: OsmoGBProxy User Manual. https://ftp.osmocom.org/docs/-latest/osmogbproxy-usermanual.pdf

[12] [vty-ref-osmogbproxy] Osmocom Project: OsmoGBPRoxy VTY Reference Manual. https://ftp.osmocom.org/-docs/latest/osmogbproxy-vty-reference.pdf

[13] [userman-osmoggsn] Osmocom Project: OpenGGSN User Manual. https://ftp.osmocom.org/docs/latest/-osmoggsn-usermanual.pdf

[14] [vty-ref-osmoggsn] Osmocom Project: OsmoGGSN VTY Reference Manual. https://ftp.osmocom.org/docs/-latest/osmoggsn-vty-reference.pdf

[15] [userman-osmohlr] Osmocom Project: OsmoHLR User Manual. https://ftp.osmocom.org/docs/latest/osmohlr-usermanual.pdf

[16] [vty-ref-osmohlr] Osmocom Project: OsmoHLR VTY Reference Manual. https://ftp.osmocom.org/docs/latest/-osmohlr-vty-reference.pdf

[17] [userman-osmohnbgw] Osmocom Project: OsmoHNBGW User Manual. https://ftp.osmocom.org/docs/latest/-osmohnbgw-usermanual.pdf

[18] [vty-ref-osmohnbgw] Osmocom Project: OsmoHNBGW VTY Reference Manual. https://ftp.osmocom.org/-docs/latest/osmohnbgw-vty-reference.pdf

[19] [userman-osmomgw] Osmocom Project: OsmoMGW User Manual. https://ftp.osmocom.org/docs/latest/-osmomgw-usermanual.pdf

[20] [vty-ref-osmomgw] Osmocom Project: OsmoMGW VTY Reference Manual. https://ftp.osmocom.org/docs/-latest/osmomgw-vty-reference.pdf

[21] [userman-osmomsc] Osmocom Project: OsmoMSC User Manual. https://ftp.osmocom.org/docs/latest/-osmomsc-usermanual.pdf

[22] [vty-ref-osmomsc] Osmocom Project: OsmoMSC VTY Reference Manual. https://ftp.osmocom.org/docs/-latest/osmomsc-vty-reference.pdf

[23] [userman-osmonitb] Osmocom Project: OsmoNITB User Manual. https://ftp.osmocom.org/docs/latest/-osmonitb-usermanual.pdf

[24] [vty-ref-osmonitb] Osmocom Project: OsmoNITB VTY Reference Manual. https://ftp.osmocom.org/docs/-latest/osmonitb-vty-reference.pdf

[25] [userman-osmopcu] Osmocom Project: OsmoPCU User Manual. https://ftp.osmocom.org/docs/latest/-osmopcu-usermanual.pdf

[26] [vty-ref-osmopcu] Osmocom Project: OsmoPCU VTY Reference Manual. https://ftp.osmocom.org/docs/-latest/osmopcu-vty-reference.pdf

[27] [userman-osmosgsn] Osmocom Project: OsmoSGSN User Manual. https://ftp.osmocom.org/docs/latest/-osmosgsn-usermanual.pdf

[28] [vty-ref-osmosgsn] Osmocom Project: OsmoSGSN VTY Reference Manual. https://ftp.osmocom.org/docs/-latest/osmosgsn-vty-reference.pdf

[29] [userman-osmosipconnector] Osmocom Project: OsmoSIPconnector User Manual. https://ftp.osmocom.org/-docs/latest/osmosipconnector-usermanual.pdf

[30] [vty-ref-osmosipconnector] Osmocom Project: OsmoSIPconnector VTY Reference Manual. https://ftp.osmocom.org/docs/latest/osmosipconnector-vty-reference.pdf

[31] [userman-osmosmlc] Osmocom Project: OsmoSMLC User Manual. https://ftp.osmocom.org/docs/latest/-osmosmlc-usermanual.pdf

[32] [vty-ref-osmosmlc] Osmocom Project: OsmoSMLC VTY Reference Manual. https://ftp.osmocom.org/docs/-latest/osmosmlc-vty-reference.pdf

[33] [userman-osmostp] Osmocom Project: OsmoSTP User Manual. https://ftp.osmocom.org/docs/latest/osmostp-usermanual.pdf

[34] [vty-ref-osmostp] Osmocom Project: OsmoSTP VTY Reference Manual. https://ftp.osmocom.org/docs/latest/-osmostp-vty-reference.pdf

[35] [userman-osmotrx] Osmocom Project: OsmoTRX User Manual. https://ftp.osmocom.org/docs/latest/osmotrx-usermanual.pdf

[36] [vty-ref-osmotrx] Osmocom Project: OsmoTRX VTY Reference Manual. https://ftp.osmocom.org/docs/-latest/osmotrx-uhd-vty-reference.pdf https://ftp.osmocom.org/docs/latest/osmotrx-lms-vty-reference.pdf https://ftp.osmocom.org/docs/latest/osmotrx-ipc-vty-reference.pdf https://ftp.osmocom.org/docs/latest/-osmotrx-usrp1-vty-reference.pdf

[37] [3gpp-ts-23-041] 3GPP TS 23.041: Technical realization of Cell Broadcast Service (CBS)

[38] [3gpp-ts-23-048] 3GPP TS 23.048: Security mechanisms for the (U)SIM application toolkit; Stage 2 https://www.3gpp.org/DynaReport/23048.htm

[39] [3gpp-ts-23-236] 3GPP TS 23.236: Intra-domain connection of Radio Access Network (RAN) nodes to multiple Core Network (CN) nodes https://www.3gpp.org/DynaReport/23236.htm

[40] [3gpp-ts-24-007] 3GPP TS 24.007: Mobile radio interface signalling layer 3; General Aspects https://www.3gpp.org/DynaReport/24007.htm

[41] [3gpp-ts-24-008] 3GPP TS 24.008: Mobile radio interface Layer 3 specification; Core network protocols; Stage 3. https://www.3gpp.org/dynareport/24008.htm

[42] [3gpp-ts-31-101] 3GPP TS 31.101: UICC-terminal interface; Physical and logical characteristics https://www.3gpp.org/DynaReport/31101.htm

[43] [3gpp-ts-31-102] 3GPP TS 31.102: Characteristics of the Universal Subscriber Identity Module (USIM) application https://www.3gpp.org/DynaReport/31102.htm

[44] [3gpp-ts-31-103] 3GPP TS 31.103: Characteristics of the IMS Subscriber Identity Module (ISIM) application https://www.3gpp.org/DynaReport/31103.htm

[45] [3gpp-ts-31-111] 3GPP TS 31.111: Universal Subscriber Identity Module (USIM) Application Toolkit (USAT) https://www.3gpp.org/DynaReport/31111.htm

[46] [3gpp-ts-31-115] 3GPP TS 31.115: Secured packet structure for (Universal) Subscriber Identity Module (U)SIM Toolkit applications https://www.3gpp.org/DynaReport/31115.htm

[47] [3gpp-ts-31-116] 3GPP TS 31.116: Remote APDU Structure for (U)SIM Toolkit applications https://www.3gpp.org/DynaReport/31116.htm

[48] [3gpp-ts-35-205] 3GPP TS 35.205: 3G Security; Specification of the MILENAGE algorithm set: General

[49] [3gpp-ts-35-206] 3GPP TS 35.206: 3G Security; Specification of the MILENAGE algorithm set: Algorithm specification https://www.3gpp.org/DynaReport/35206.htm

[50] [3gpp-ts-44-006] 3GPP TS 44.006: Mobile Station - Base Station System (MS - BSS) interface; Data Link (DL) layer specification https://www.3gpp.org/DynaReport/44006.htm

[51] [3gpp-ts-44-018] 3GPP TS 44.018: Mobile radio interface layer 3 specification; Radio Resource Control (RRC) protocol https://www.3gpp.org/DynaReport/44018.htm

[52] [3gpp-ts-44-064] 3GPP TS 44.064: Mobile Station - Serving GPRS Support Node (MS-SGSN); Logical Link Control (LLC) Layer Specification https://www.3gpp.org/DynaReport/44064.htm

[53] [3gpp-ts-45-002] 3GPP TS 45.002: Digital cellular telecommunications system (Phase 2+) (GSM); GSM/EDGE Multiplexing and multiple access on the radio path https://www.3gpp.org/DynaReport/45002.htm

[54] [3gpp-ts-48-008] 3GPP TS 48.008: Mobile Switching Centre - Base Station system (MSC-BSS) interface; Layer 3 specification https://www.3gpp.org/DynaReport/48008.htm

[55] [3gpp-ts-48-016] 3GPP TS 48.016: General Packet Radio Service (GPRS); Base Station System (BSS) - Serving GPRS Support Node (SGSN) interface; Network service https://www.3gpp.org/DynaReport/48016.htm

[56] [3gpp-ts-48-018] 3GPP TS 48.018: General Packet Radio Service (GPRS); Base Station System (BSS) - Serving GPRS Support Node (SGSN); BSS GPRS protocol (BSSGP) https://www.3gpp.org/DynaReport/48018.htm

[57] [3gpp-ts-48-049] 3GPP TS 48.049: Digital cellular communications system; Base Station Controller - Cell Broadcast Centre (BSC-CBC) interface specification; Cell Broadcast Service Protocol (CBSP) https://www.3gpp.org/DynaReport/48049.htm

[58] [3gpp-ts-48-056] 3GPP TS 48.056: Base Station Controller - Base Transceiver Station (BSC - BTS) interface; Layer 2 specification https://www.3gpp.org/DynaReport/48056.htm

[59] [3gpp-ts-48-058] 3GPP TS 48.058: Base Station Controller - Base Transceiver Station (BSC - BTS) Interface; Layer 3 specification https://www.3gpp.org/DynaReport/48058.htm

[60] [3gpp-ts-51-011] 3GPP TS 51.011: Specification of the Subscriber Identity Module - Mobile Equipment (SIM-ME) interface

[61] [3gpp-ts-51-014] 3GPP TS 51.014: Specification of the SIM Application Toolkit for the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface https://www.3gpp.org/DynaReport/51014.htm

[62] [3gpp-ts-52-021] 3GPP TS 52.021: Network Management (NM) procedures and messages on the A-bis interface https://www.3gpp.org/DynaReport/52021.htm

[63] [etsi-tr102216] ETSI TR 102 216: Smart cards https://www.etsi.org/deliver/etsi_tr/102200_102299/102216/-03.00.00_60/tr_102216v030000p.pdf

[64] [etsi-ts102221] ETSI TS 102 221: Smart Cards; UICC-Terminal interface; Physical and logical characteristics https://www.etsi.org/deliver/etsi_ts/102200_102299/102221/13.01.00_60/ts_102221v130100p.pdf

[65] [etsi-ts101220] ETSI TS 101 220: Smart Cards; ETSI numbering system for telecommunication application providers https://www.etsi.org/deliver/etsi_ts/101200_101299/101220/12.00.00_60/ts_101220v120000p.pdf

[66] [etsi-ts102671] ETSI TS 102 671: Smart Cards; Machine to Machine UICC; Physical and logical characteristics https://www.etsi.org/deliver/etsi_ts/102600_102699/102671/18.01.00_60/ts_102671v180100p.pdf

[67] [ieee-802.1q] IEEE 802.1Q: Bridges and Bridged Networks https://ieeexplore.ieee.org/document/6991462

[68] [ietf-rfc768] IETF RFC 768: User Datagram Protocol https://tools.ietf.org/html/rfc768

[69] [ietf-rfc791] IETF RFC 791: Internet Protocol https://tools.ietf.org/html/rfc791

[70] [ietf-rfc793] IETF RFC 793: Transmission Control Protocol https://tools.ietf.org/html/rfc793

[71] [ietf-rfc1035] IETF RFC 1035: Domain Names - Implementation and Specification https://tools.ietf.org/html/-rfc1035

[72] [ietf-rfc1350] IETF RFC 1350: Trivial File Transfer Protocol https://tools.ietf.org/html/rfc1350

[73] [ietf-rfc2131] IETF RFC 2131: Dynamic Host Configuration Protocol https://tools.ietf.org/html/rfc2131

[74] [ietf-rfc2474] IETF RFC 2474: Definition of the Differentiated Services Field (DS Field) in the IPv44 and IPv6 Headers https://tools.ietf.org/html/rfc2474

[75] [ietf-rfc2719] IETF RFC 2719: Signal Transport over IP https://tools.ietf.org/html/rfc2719

[76] [ietf-rfc3331] IETF RFC 3331: Message Transfer Part 2 User Adaptation Layer https://tools.ietf.org/html/-rfc3331

[77] [ietf-rfc3550] IETF RFC 3550: RTP: A Transport protocol for Real-Time Applications https://tools.ietf.org/-html/rfc3550

[78] [ietf-rfc3596] IETF RFC 3596: DNS Extensions to Support IP Version 6 https://tools.ietf.org/html/rfc3596

[79] [ietf-rfc3868] IETF RFC 3868: SCCP User Adaptation Layer https://tools.ietf.org/html/rfc3868

[80] [ietf-rfc4165] IETF RFC 4165: Message Transfer Part 2 Peer-to-Peer Adaptation Layer https://tools.ietf.org/-html/rfc4165

[81] [ietf-rfc4251] IETF RFC 4251: The Secure Shell (SSH) Protocol Architecture https://tools.ietf.org/html/-rfc4251

[82] [ietf-rfc4666] IETF RFC 4666: Message Transfer Part 3 User Adaptation Layer https://tools.ietf.org/html/-rfc4666

[83] [ietf-rfc5771] IETF RFC 5771: IANA Guidelines for IPv4 Multicast Address Assignments https://tools.ietf.org/html/rfc5771

[84] [itu-t-q701] ITU-T Q.701: Functional Description of the Message Transfer Part (MTP) https://www.itu.int/rec/-T-REC-Q.701/en/

[85] [itu-t-q711] ITU-T Q.711: Functional Description of the Signalling Connection Control Part https://www.itu.int/rec/T-REC-Q.711/en/

[86] [itu-t-q713] ITU-T Q.713: Signalling connection control part formats and codes https://www.itu.int/rec/T-REC-Q.713/en/

[87] [itu-t-q714] ITU-T Q.714: Signalling connection control part procedures https://www.itu.int/rec/T-REC-Q.714/en/

[88] [itu-t-q921] ITU-T Q.921: ISDN user-network interface - Data link layer specification https://www.itu.int/rec/-T-REC-Q.921/en

[89] [smpp-34] SMPP Develoepers Forum. Short Message Peer-to-Peer Protocol Specification v3.4 https://docs.nimta.com/SMPP_v3_4_Issue1_2.pdf

[90] [gnu-agplv3] Free Software Foundation. GNU Affero General Public License. https://www.gnu.org/licenses/-agpl-3.0.en.html

[91] [freeswitch_pbx] FreeSWITCH SIP PBX https://freeswitch.org

[92] [tw-ts-001] TW-TS-001: Enhanced RTP transport of FR and EFR codec frames in an IP-based GSM RAN https://www.freecalypso.org/specs/tw-ts-001-v010100.txt

[93] [tw-ts-002] TW-TS-002: Enhanced RTP transport of HRv1 codec frames in an IP-based GSM RAN https://www.freecalypso.org/specs/tw-ts-002-v010100.txt

[94] [tw-ts-003] TW-TS-003: BSSMAP extension for selection of enhanced RTP transport formats https://www.freecalypso.org/specs/tw-ts-003-v010002.txt

C GNU Free Documentation License

Version 1.3, 3 November 2008

Copyright © 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc. http://fsf.org/

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

C.1 PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

C.2 APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Section whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not Transparent is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary

formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, Title Page means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

C.3 VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section Section C.4.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

C.4 COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

C.5 MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

a. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

b. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

c. State on the Title Page the name of the publisher of the Modified Version, as the publisher.

d. Preserve all the copyright notices of the Document.

e. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

f. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

g. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

h. Include an unaltered copy of this License.

i. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

j. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

k. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

l. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

m. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

n. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Sections.

o. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Section and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

C.6 COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

C.7 COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

C.8 AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

C.9 TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

C.10 TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

C.11 FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

C.12 RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

C.13 ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (c)  YEAR  YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.3
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled ``GNU
Free Documentation License''.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with. . . Texts." line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.