# ETSI TS 145 003 V16.0.0 (2020-09)



Digital cellular telecommunications system (Phase 2+) (GSM); GSM/EDGE Channel coding (3GPP TS 45.003 version 16.0.0 Release 16)



Reference
RTS/TSGR-0645003vg00

Keywords
GSM

#### **ETSI**

650 Route des Lucioles F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C Association à but non lucratif enregistrée à la Sous-Préfecture de Grasse (06) N° 7803/88

### Important notice

The present document can be downloaded from: <u>http://www.etsi.org/standards-search</u>

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI deliverable is the one made publicly available in PDF format at <a href="https://www.etsi.org/deliver">www.etsi.org/deliver</a>.

Users of the present document should be aware that the document may be subject to revision or change of status.

Information on the current status of this and other ETSI documents is available at

<a href="https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx">https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx</a>

If you find errors in the present document, please send your comment to one of the following services: https://portal.etsi.org/People/CommitteeSupportStaff.aspx

### **Copyright Notification**

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.

The content of the PDF version shall not be modified without the written authorization of ETSI.

The copyright and the foregoing restriction extend to reproduction in all media.

© ETSI 2020. All rights reserved.

**DECT™**, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners.

oneM2M™ logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners.

GSM® and the GSM logo are trademarks registered and owned by the GSM Association.

## Intellectual Property Rights

#### **Essential patents**

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: "Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards", which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

#### **Trademarks**

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

# **Legal Notice**

This Technical Specification (TS) has been produced by ETSI 3rd Generation Partnership Project (3GPP).

The present document may refer to technical specifications or reports using their 3GPP identities. These shall be interpreted as being references to the corresponding ETSI deliverables.

The cross reference between 3GPP and ETSI identities can be found under http://webapp.etsi.org/key/queryform.asp.

### Modal verbs terminology

In the present document "shall", "shall not", "should", "should not", "may", "need not", "will", "will not", "can" and "cannot" are to be interpreted as described in clause 3.2 of the <u>ETSI Drafting Rules</u> (Verbal forms for the expression of provisions).

"must" and "must not" are NOT allowed in ETSI deliverables except when used in direct citation.

# Contents

Intell	ectual Property Rights	2
Legal	Notice	2
Moda	al verbs terminology	2
Forev	word	18
1	Scope	10
1.1	References	
1.2	Abbreviations	
2	General	20
2.1	General organization	20
2.2	Naming Convention	
3	Traffic Channels (TCH)	
3.1	Speech channel at full rate (TCH/FS and TCH/EFS)	43
3.1.1	Preliminary channel coding for EFR only	43
3.1.1.	1 CRC calculation	43
3.1.1.2	<b>2</b>	
3.1.1.3	Correspondence between input and output of preliminary channel coding	44
3.1.2	Channel coding for FR and EFR	44
3.1.2.1	Parity and tailing for a speech frame	44
3.1.2.2	2 Convolutional encoder	45
3.1.3	Interleaving	45
3.1.4	Mapping on a Burst	45
3.2	Speech channel at half rate (TCH/HS)	46
3.2.1	Parity and tailing for a speech frame	46
3.2.2	Convolutional encoder	46
3.2.3	Interleaving	47
3.2.4	Mapping on a burst	47
3.3	Data channel at full rate, 12.0 kbit/s radio interface rate (9.6 kbit/s services (TCH/F9.6))	47
3.3.1	Interface with user unit	48
3.3.2	Block code	48
3.3.3	Convolutional encoder	48
3.3.4	Interleaving	48
3.3.5	Mapping on a Burst	48
3.4	Data channel at full rate, 6.0 kbit/s radio interface rate (4.8 kbit/s services (TCH/F4.8))	49
3.4.1	Interface with user unit	49
3.4.2	Block code	49
3.4.3	Convolutional encoder	49
3.4.4	Interleaving	49
3.4.5	Mapping on a Burst	
3.5	Data channel at half rate, 6.0 kbit/s radio interface rate (4.8 kbit/s services (TCH/H4.8))	50
3.5.1	Interface with user unit	50
3.5.2	Block code	50
3.5.3	Convolutional encoder	50
3.5.4	Interleaving	50
3.5.5	Mapping on a Burst	50
3.6	Data channel at full rate, 3.6 kbit/s radio interface rate (2.4 kbit/s and less services (TCH/F2.4))	50
3.6.1	Interface with user unit	50
3.6.2	Block code	
3.6.3	Convolutional encoder	50
3.6.4	Interleaving	51
3.6.5	Mapping on a Burst	51
3.7	Data channel at half rate, 3.6 kbit/s radio interface rate (2.4 kbit/s and less services (TCH/H2.4))	
3.7.1	Interface with user unit	
3.7.2	Block code	
3.7.3	Convolutional encoder	51

3.7.4	Interleaving	
3.7.5	Mapping on a Burst	
3.8	Data channel at full rate, 14.5 kbit/s radio interface rate (14.4 kbit/s services (TCH/F14.4))	
3.8.1	Interface with user unit	
3.8.2	Block code	
3.8.3	Convolutional encoder	
3.8.4	Interleaving	
3.8.5	Mapping on a Burst	
3.9	Adaptive multi rate speech channel at full rate (TCH/AFS)	52
3.9.1	SID_UPDATE	
3.9.1.1	Coding of in-band data	
3.9.1.2	Parity and convolutional encoding for the comfort noise parameters	
3.9.1.3	Identification marker	
3.9.1.4	Interleaving	
3.9.1.5	Mapping on a Burst	
3.9.2	SID_FIRST	
3.9.2.1	Coding of in-band data	
3.9.2.2	Identification marker	
3.9.2.3	Interleaving	
3.9.2.4	Mapping on a Burst	
3.9.3	ONSET	
3.9.3.1	Coding of in-band data	
3.9.3.2	Interleaving	
3.9.3.3	Mapping on a Burst	
3.9.4	SPEECH	
3.9.4.1	Coding of the in-band data	
3.9.4.2	Ordering according to subjective importance	
3.9.4.3	Parity for speech frames	
3.9.4.4	Convolutional encoder	
3.9.4.5	Interleaving	
3.9.4.6	Mapping on a Burst	
3.9.5	RATSCCH	
3.9.5.1	Coding of in-band data	
3.9.5.2	Parity and convolutional encoding for the RATSCCH message	
3.9.5.3	Identification marker	
3.9.5.4	Interleaving	
3.9.5.5	Mapping on a Burst	
3.10	Adaptive multi rate speech channel at half rate (TCH/AHS)	
3.10.1	SID_UPDATE  Coding of in-band data	68
3.10.1.1 3.10.1.2	Parity and convolutional encoding for the comfort noise parameters	
3.10.1.2	Identification marker	
3.10.1.4 3.10.1.5	Interleaving	
3.10.1.3	** *	
3.10.2.1	SID_UPDATE_INH  Coding of in-band data	
3.10.2.1	Identification marker	
3.10.2.2	Interleaving	
3.10.2.3	Mapping on a Burst	
3.10.2.4	SID_FIRST_P1	
3.10.3.1	Coding of in-band data	
3.10.3.1	Identification marker	
3.10.3.2	Interleaving	
3.10.3.4	Mapping on a Burst	
3.10.3.4	SID_FIRST_P2	
3.10.4	Coding of in-band data	
3.10.4.1	Interleaving	
3.10.4.2	Mapping on a Burst	
3.10.4.3	SID_FIRST_INH	
3.10.5.1	Coding of in-band data	
3.10.5.2	Identification marker	
3.10.5.3	Interleaving	

3.10.5.4	Mapping on a Burst	
3.10.6	ONSET	
3.10.6.1	Coding of in-band data	
3.10.6.2	Interleaving	73
3.10.6.3	Mapping on a Burst	
3.10.7	SPEECH	
3.10.7.1	Coding of the in-band data	
3.10.7.2	Ordering according to subjective importance	73
3.10.7.3	Parity for speech frames	
3.10.7.4	Convolutional encoder	75
3.10.7.5	Interleaving	
3.10.7.6	Mapping on a Burst	
3.10.8	RATSCCH_MARKER	
3.10.8.1	Coding of in-band data	
3.10.8.2	Identification marker	
3.10.8.3	Interleaving	80
3.10.8.4	Mapping on a Burst	
3.10.9	RATSCCH_DATA	
3.10.9.1	Coding of in-band data	
3.10.9.2	Parity and convolutional encoding for the RATSCCH message	
3.10.9.3	Interleaving	
3.10.9.4	Mapping on a Burst	
3.11	Data channel for ECSD at full rate, 29.0 kbit/s radio interface rate (28.8 kbit/s services (E-TCH/F28.8))	
3.11.1	Interface with user unit	
3.11.2	Block code	
3.11.2.1	Repetition bits	
3.11.2.2	Reed Solomon encoder	
3.11.3	Convolutional encoder	
3.11.3.1	Tailing bits for a data frame	
3.11.3.2	Convolutional encoding for a data frame	
3.11.4	Interleaving	
3.11.5	Mapping on a Burst	
3.12	Data channel for ECSD at full rate, 32.0 kbit/s radio interface rate (32.0 kbit/s services (E-TCH/F32.0))	
3.12.1	Interface with user unit	
3.12.2	Void	
3.12.3	Convolutional encoder	
3.12.3.1 3.12.3.2	Tailing bits for a data frame	
	Convolutional encoding for a data frame	
3.12.4 3.12.5	Interleaving	
	Mapping on a Burst	
3.13 3.13.1	Interface with user unit	
3.13.1	Convolutional encoder	
3.13.2.1	Tailing bits for a data frame	
3.13.2.1	Convolutional encoding for a data frame	
3.13.2.2	Interleaving	
3.13.4	Mapping on a Burst	
3.14	Wideband Adaptive multi rate speech channel at full rate (TCH/WFS)	
3.14.1	SID_UPDATE	
3.14.2	SID_FIRST	
3.14.3	ONSET	
3.14.4	SPEECH	
3.14.4.1	Coding of the in-band data	
3.14.4.2	Ordering according to subjective importance	
3.14.4.3	Parity for speech frames	
3.14.4.4	Convolutional encoder	
3.14.4.5	Interleaving	
3.14.4.6	Mapping on a Burst	
3.14.5	RATSCCH	
3.15	Adaptive multi rate speech channel at 8-PSK half rate (O-TCH/AHS)	
3.15.1	SID_UPDATE	
3 15 1 1	Coding of in hand data	03

3.15.1.2	Parity and convolutional encoding for the comfort noise parameters	
3.15.1.3	Identification marker	93
3.15.1.4	Repetition	93
3.15.1.5	Interleaving	93
3.15.1.6	Mapping on a Burst	93
3.15.2	SID_UPDATE_INH	94
3.15.2.1	Coding of in-band data	94
3.15.2.2	Identification marker	94
3.15.2.3	Repetition	94
3.15.2.4	Interleaving	94
3.15.2.5	Mapping on a Burst	94
3.15.3	SID_FIRST_P1	94
3.15.3.1	Coding of in-band data	94
3.15.3.2	Identification marker	94
3.15.3.3	Repetition	94
3.15.3.4	Interleaving	95
3.15.3.5	Mapping on a Burst	95
3.15.4	SID_FIRST_P2	95
3.15.4.1	Coding of in-band data	95
3.15.4.2	Repetition	95
3.15.4.3	Interleaving	95
3.15.4.4	Mapping on a Burst	95
3.15.5	SID_FIRST_INH	95
3.15.5.1	Coding of in-band data	95
3.15.5.2	Identification marker	96
3.15.5.3	Repetition	96
3.15.5.4	Interleaving	96
3.15.5.5	Mapping on a Burst	96
3.15.6	ONSET	96
3.15.6.1	Coding of in-band data	96
3.15.6.2	Repetition	96
3.15.6.3	Interleaving	96
3.15.6.4	Mapping on a Burst	96
3.15.7	SPEECH	96
3.15.7.1	Coding of the in-band data	96
3.15.7.2	Ordering according to subjective importance	97
3.15.7.3	Parity for speech frames	
3.15.7.4	Convolutional encoder	99
3.15.7.5	Interleaving	
3.15.7.6	Mapping on a Burst	114
3.15.8	RATSCCH_MARKER	114
3.15.8.1	Coding of in-band data	114
3.15.8.2	Identification marker	114
3.15.8.3	Interleaving	114
3.15.8.4	Mapping on a Burst	
3.15.9	RATSCCH_DATA	
3.15.9.1	Coding of in-band data	
3.15.9.2	Parity and convolutional encoding for the RATSCCH message	115
3.15.9.3	Interleaving	
3.15.9.4	Mapping on a Burst	
3.16	Wideband Adaptive multi rate speech channel at 8-PSK full rate (O-TCH/WFS)	
3.16.1	SID_UPDATE	
3.16.1.1	Coding of in-band data	
3.16.1.2	Parity and convolutional encoding for the comfort noise parameters	116
3.16.1.3	Identification marker	116
3.16.1.4	Repetition	116
3.16.1.5	Interleaving	
3.16.1.6	Mapping on a Burst	
3.16.2	SID_FIRST	
3.16.2.1	Coding of in-band data	
3.16.2.2	Identification marker	
3.16.2.3	Repetition	117

3.16.2.4	Interleaving	
3.16.2.5	Mapping on a Burst	117
3.16.3	ONSET	
3.16.3.1	Coding of in-band data	117
3.16.3.2	Repetition	117
3.16.3.3	Interleaving	117
3.16.3.4	Mapping on a Burst	117
3.16.4	SPEECH	117
3.16.4.1	Coding of the in-band data	118
3.16.4.2	Ordering according to subjective importance	
3.16.4.3	Parity for speech frames	
3.16.4.4	Convolutional encoder	
3.16.4.5	Interleaving	
3.16.4.6	Mapping on a Burst	
3.16.5	RATSCCH	
3.16.5.1	Coding of in-band data	
3.16.5.2	Parity and convolutional encoding for the RATSCCH message	
3.16.5.3	Identification marker	
3.16.5.4	Interleaving	
3.16.5.5	Mapping on a Burst	
3.17	Wideband Adaptive multi rate speech channel at 8-PSK half rate (O-TCH/WHS)	
3.17.1	SID_UPDATE	
3.17.1.1	Coding of in-band data	
3.17.1.2	Parity and convolutional encoding for the comfort noise parameters	
3.17.1.2	Identification marker	
3.17.1.3	Repetition	
3.17.1.4	•	
	Interleaving	
3.17.1.6	Mapping on a Burst	
3.17.2	SID_UPDATE_INH	
3.17.2.1	Coding of in-band data	
3.17.2.2	Identification marker	
3.17.2.3	Repetition	
3.17.2.4	Interleaving	
3.17.2.5	Mapping on a Burst	
3.17.3	SID_FIRST_P1	
3.17.3.1	Coding of in-band data	
3.17.3.2	Identification marker	
3.17.3.3	Repetition	
3.17.3.4	Interleaving	
3.17.3.5	Mapping on a Burst	
3.17.4	SID_FIRST_P2	
3.17.4.1	Coding of in-band data	
3.17.4.2	Repetition	
3.17.4.3	Interleaving	
3.17.4.4	Mapping on a Burst	
3.17.5	SID_FIRST_INH	
3.17.5.1	Coding of in-band data	
3.17.5.2	Identification marker	
3.17.5.3	Repetition	
3.17.5.4	Interleaving	138
3.17.5.5	Mapping on a Burst	138
3.17.6	ONSET	138
3.17.6.1	Coding of in-band data	138
3.17.6.2	Repetition	138
3.17.6.3	Interleaving	
3.17.6.4	Mapping on a Burst	
3.17.7	SPEECH	
3.17.7.1	Coding of the in-band data	
3.17.7.2	Ordering according to subjective importance	
3.17.7.3	Parity for speech frames	
3.17.7.4	Convolutional encoder	
3.17.7.5	Interleaving	

3.17.7.6	Mapping on a Burst	146
3.17.8	RATSCCH_MARKER	146
3.17.8.1	Coding of in-band data	146
3.17.8.2	Identification marker	146
3.17.8.3	Interleaving	146
3.17.8.4	Mapping on a Burst	146
3.17.9	RATSCCH_DATA	146
3.17.9.1	Coding of in-band data	146
3.17.9.2	Parity and convolutional encoding for the RATSCCH message	147
3.17.9.3	Interleaving	147
3.17.9.4	Mapping on a Burst	147
4 (	Control Channels	1.47
4.1	Slow associated control channel (SACCH)	
4.1.1	Block constitution	
4.1.1	Block code	
4.1.3	Convolutional encoder	
4.1.4	Interleaving	
4.1.5	Mapping on a Burst	
4.1.3	Fast associated control channel at full rate (FACCH/F)	
4.2.1	Block constitution	
4.2.2	Block code	
4.2.3	Convolutional encoder	
4.2.3	Interleaving	
4.2.5	Mapping on a Burst	
4.2.3	Fast associated control channel at half rate (FACCH/H)	
4.3.1	Block constitution	
4.3.2	Block code	
4.3.3	Convolutional encoder	
4.3.4	Interleaving	
4.3.5	Mapping on a Burst	
4.4	Broadcast control, Paging, Access grant, Notification and Cell broadcast channels (BCCH, PCH	
	AGCH, NCH, CBCH), CTS Paging and Access grant channels (CTSPCH, CTSAGCH)	
4.5	Stand-alone dedicated control channel (SDCCH)	
4.6	Random access channel (RACH)	
4.6.1	RACH carrying 8 information bits	
4.6.2	(EC-)RACH carrying 30 information bits	
4.6.2.1	Block constitution	
4.6.2.2	Parity bits	
4.6.2.3	Tail bits	
4.6.2.4	Convolutional code	152
4.6.2.5	Burst mapping	152
4.7	Synchronization channel (SCH), Compact synchronization channel (CSCH), CTS Beacon and A	ccess
	request channels (CTSBCH-SB, CTSARCH)	
4.7a	Extended Coverage Synchronization channel (EC-SCH)	153
4.7a.1	Block constitution	153
4.7a.2	Coding	
4.7a.3	Blind physical layer transmission	
4.7a.4	Cyclic shift	
4.7a.5	Mapping onto a physical channel	
4.8	Access Burst on circuit switched channels other than RACH	
4.9	Access Bursts for uplink access on a channel used for VGCS or VBS	
4.10a	Fast associated control channel at ECSD E-TCH/F (E-FACCH/F)	
4.10a.1	Block constitution	
4.10a.2	Block code	
4.10a.3	Convolutional encoder	
4.10a.4	Interleaving	
4.10a.5	Mapping on a Burst	
4.10b	Octal fast associated control channel at half rate (O-FACCH/H)	
4.10b.1	Block constitution	
4.10b.2	Block code	
4.10b.3	Convolutional encoder	156

4.10b.4	Reordering	
4.10b.5	Interleaving	
4.10b.6	Mapping on a burst	
4.10c	Octal fast associated control channel at full rate (O-FACCH/F)	157
4.10c.1	Block constitution	157
4.10c.2	Block code	
4.10c.3	Convolutional encoder	157
4.10c.4	Reordering	157
4.10c.5	Interleaving	158
4.10c.6	Mapping on a burst	158
4.11	Slow associated control channel with embedded enhanced power control (SACCH/TP)	158
4.11.1	Block constitution	
4.11.2	Block code	
4.11.3	Convolutional encoder	
4.11.4	Dummy bits insertion	
4.11.5	Interleaving	
4.11.6	Mapping on a Burst	
4.12	Enhanced power control channel (EPCCH)	
4.12.1	Block code	
4.12.2	Mapping on a Burst	160
5 P	acket Switched Channels	161
5.1	Packet data traffic channel (PDTCH)	
5.1.1	Packet data block type 1 (CS-1)	
5.1.2	Packet data block type 2 (CS-2)	
5.1.2.1	Block constitution	
5.1.2.2	Block code	
5.1.2.3	Convolutional encoder	
5.1.2.4	Interleaving	
5.1.2.5	Mapping on a burst	
5.1.3	Packet data block type 3 (CS-3)	163
5.1.3.1	Block constitution	163
5.1.3.2	Block code	163
5.1.3.3	Convolutional encoder	
5.1.3.4	Interleaving	
5.1.3.5	Mapping on a burst	
5.1.4	Packet data block type 4 (CS-4)	
5.1.4.1	Block constitution	
5.1.4.2	Block code	
5.1.4.3	Convolutional encoder	
5.1.4.4	Interleaving	
5.1.4.5	Mapping on a burst	
5.1.4a	Packet data block type 5a (MCS-0)	
5.1.4a.1 5.1.4a.1.	Downlink (MCS-0 DL) Block constitution.	
5.1.4a.1. 5.1.4a.1.		
5.1.4a.1 5.1.4a.1		
5.1.4a.1.	· · · · · · · · · · · · · · · · · · ·	
5.1.4a.1	e e e e e e e e e e e e e e e e e e e	
5.1.4a.1.		
5.1. <del>4</del> a.1.	Packet data block type 5 (MCS-1)	
5.1.5.1	Downlink (MCS-1 DL)	
5.1.5.1.1	·	
5.1.5.1.2		
5.1.5.1.2		
5.1.5.1.2		
5.1.5.1.3	e	
5.1.5.1.4	Data coding	169
5.1.5.1.4		
5.1.5.1.5	Č	171
5.1.5.1.6	11 6	
51516	1 BTTI configuration	171

5.1.5.1.6.2	RTTI configuration	
5.1.5.2	Uplink (MCS-1 UL)	173
5.1.5.2.1	Block constitution	173
5.1.5.2.2	Header coding	173
5.1.5.2.3	Data coding	173
5.1.5.2.3a	Piggy-backed Ack/Nack coding	
5.1.5.2.4	Interleaving	
5.1.5.2.5	Mapping on a burst	
5.1.6	Packet data block type 6 (MCS-2)	
5.1.6.1	Downlink (MCS-2 DL)	
5.1.6.1.1		
	Block constitution	
5.1.6.1.2	USF precoding	
5.1.6.1.2.1	BTTI configuration	
5.1.6.1.2.2	RTTI configuration	
5.1.6.1.3	Header coding	
5.1.6.1.4	Data coding	
5.1.6.1.4a	Piggy-backed Ack/Nack coding	176
5.1.6.1.5	Interleaving	176
5.1.6.1.6	Mapping on a burst	176
5.1.6.2	Uplink (MCS-2 UL)	176
5.1.6.2.1	Block constitution	
5.1.6.2.2	Header coding	
5.1.6.2.3	Data coding	
5.1.6.2.3a	Piggy-backed Ack/Nack coding	
5.1.6.2.4	Interleaving	
5.1.6.2.5	Mapping on a burst	
5.1.7		
	Packet data block type 7 (MCS-3)	
5.1.7.1	Downlink (MCS-3 DL)	
5.1.7.1.1	Block constitution	
5.1.7.1.2	USF precoding	
5.1.7.1.2.1	BTTI configuration	
5.1.7.1.2.2	RTTI configuration	
5.1.7.1.3	Header coding	
5.1.7.1.4	Data coding	177
5.1.7.1.4a	Piggy-backed Ack/Nack coding	179
5.1.7.1.5	Interleaving	179
5.1.7.1.6	Mapping on a burst	179
5.1.7.2	Uplink (MCS-3 UL)	179
5.1.7.2.1	Block constitution	179
5.1.7.2.2	Header coding	
5.1.7.2.3	Data coding	
5.1.7.2.3a	Piggy-backed Ack/Nack coding	
5.1.7.2.4	Interleaving	
5.1.7.2.5	Mapping on a burst	
5.1.8	11 0	
5.1.8.1	Packet data block type 8 (MCS-4)	
	,	
5.1.8.1.1	Block constitution.	
5.1.8.1.2	USF precoding	
5.1.8.1.2.1	BTTI configuration	
5.1.8.1.2.2	RTTI configuration	
5.1.8.1.3	Header coding	
5.1.8.1.4	Data coding	
5.1.8.1.5	Interleaving	
5.1.8.1.6	Mapping on a burst	
5.1.8.2	Uplink (MCS-4 UL)	181
5.1.8.2.1	Block constitution	
5.1.8.2.2	Header coding	
5.1.8.2.3	Data coding	
5.1.8.2.4	Interleaving	
5.1.8.2.5	Mapping on a burst	
5.1.9	Packet data block type 9 (MCS-5)	
5 1 9 1	Downlink (MCS-5 DL)	181

5.1.9.1.1	Block constitution	
5.1.9.1.2	USF precoding	
5.1.9.1.2.1	BTTI configuration	
5.1.9.1.2.2	RTTI configurations	182
5.1.9.1.3	Header coding	182
5.1.9.1.4	Data coding	183
5.1.9.1.4a	Piggy-backed Ack/Nack coding	184
5.1.9.1.5	Interleaving	185
5.1.9.1.6	Mapping on a burst	186
5.1.9.2	Uplink (MCS-5 UL)	
5.1.9.2.1	Block constitution.	
5.1.9.2.2	Header coding	
5.1.9.2.3	Data coding	
5.1.9.2.3a	Piggy-backed Ack/Nack coding	
5.1.9.2.4	Interleaving	
5.1.9.2.5	Mapping on a burst	
5.1.10	Packet data block type 10 (MCS-6)	
5.1.10.1	Downlink (MCS-6 DL)	
5.1.10.1.1	Block constitution.	
5.1.10.1.2	USF precoding.	
5.1.10.1.2.1	BTTI configuration	
5.1.10.1.2.2	RTTI configuration	
5.1.10.1.2.2	Header coding	
5.1.10.1.3	Data coding	
5.1.10.1.4 5.1.10.1.4a	Piggy-backed Ack/Nack coding	
5.1.10.1.4a 5.1.10.1.5	Interleaving	
5.1.10.1.5	Mapping on a burst	
5.1.10.2	Uplink (MCS-6 UL)	
5.1.10.2.1	Block constitution	
5.1.10.2.2	Header coding	
5.1.10.2.3	Data coding	
5.1.10.2.3a	Piggy-backed Ack/Nack coding	
5.1.10.2.4	Interleaving	
5.1.10.2.5	Mapping on a burst	
5.1.11	Packet data block type 11 (MCS-7)	
5.1.11.1	Downlink (MCS-7 DL)	
5.1.11.1.1	Block constitution	
5.1.11.1.2	USF precoding	
5.1.11.1.2.2	RTTI configuration	
5.1.11.1.3	Header coding	
5.1.11.1.4	Data coding	
5.1.11.1.4a	Piggy-backed Ack/Nack coding	
5.1.11.1.5	Interleaving	
5.1.11.1.6	Mapping on a burst	
5.1.11.2	Uplink (MCS-7 UL)	
5.1.11.2.1	Block constitution	
5.1.11.2.2	Header coding	
5.1.11.2.3	Data coding	199
5.1.11.2.3a	Piggy-backed Ack/Nack coding	
5.1.11.2.4	Interleaving	
5.1.11.2.5	Mapping on a burst	200
5.1.12	Packet data block type 12 (MCS-8)	
5.1.12.1	Downlink (MCS-8 DL)	
5.1.12.1.1	Block constitution.	
5.1.12.1.2	USF precoding	
5.1.12.1.2.1	BTTI configuration	
5.1.12.1.2.2	RTTI configuration	
5.1.12.1.3	Header coding	
5.1.12.1.4	Data coding	
5.1.12.1.4a	Piggy-backed Ack/Nack coding	
5.1.12.1.5	Interleaving	
5.1.12.1.6	Mapping on a burst	
- · - · - <del>- · - · · · ·</del>		

5.1.12.2	Uplink (MCS-8 UL)	204
5.1.12.2.1	Block constitution	
5.1.12.2.2	Header coding	
5.1.12.2.3	Data coding	
5.1.12.2.3a	Piggy-backed Ack/Nack coding	
5.1.12.2.4	Interleaving	
5.1.12.2.5	Mapping on a burst	
5.1.13	Packet data block type 13 (MCS-9)	
5.1.13.1	Downlink (MCS-9 DL)	
5.1.13.1.1	Block constitution	
5.1.13.1.2	USF precoding	
5.1.13.1.2.1	BTTI configuration	
5.1.13.1.2.2	RTTI configuration	
5.1.13.1.3	Header coding	
5.1.13.1.4	Data coding	
5.1.13.1.5	Interleaving	
5.1.13.1.6	Mapping on a burst	
5.1.13.2	Uplink (MCS-9 UL)	
5.1.13.2.1	Block constitution	
5.1.13.2.2	Header coding	
5.1.13.2.3	Data coding	
5.1.13.2.4	Interleaving	
5.1.13.2.4	Mapping on a burst	
	cket data traffic channels (PDTCH) for EGPRS2	
5.1a Fac		
5.1a.1 5.1a.1.1	General descriptions of channel coding functions	
	Header	
5.1a.1.2	Data encoded with convolutional code	
5.1a.1.3	Data encoded with turbo code	
5. 1a.1.3.1	Parity bits	
5.1a.1.3.2	Turbo encoding	
5.1a.1.3.3	Trellis termination for Turbo coder	
5.1a.1.3.4	Turbo code internal interleaver	
5.1a.1.3.4.1	Bits-input to rectangular matrix with padding	
5.1a.1.3.5	Turbo code puncturing	
5.1a.1.3.5.2.1	i &	
5.1a.1.3.5.2.2	1 0 11	
5.1a.1.3.5.2.3	1 51	
5.1a.1.3.5.2.4		
5.1a.1.3.5.3	PAN Parameters Handling	
5.1a.1.4	PAN	
5.1a.2	General descriptions of interleaving functions	
5.1a.2.1	Interleaver type 1	
5.1a.2.2	Interleaver type 2	
5.1a.3	Packet data block type 14 (UAS-7)	
5.1a.3.1	Block constitution	222
5.1a.3.2	Header coding	
5.1a.3.3	Data coding	223
5.1a.3.4	PAN coding	
5.1a.3.5	Interleaving	223
5.1a.3.6	Mapping on a burst	224
5.1a.4	Packet data block type 15 (UAS-8)	
5.1a.4.1	Block constitution	
5.1a.4.2	Header coding	
5.1a.4.3	Data coding	
5.1a.4.4	PAN coding	
5.1a.4.5	Interleaving	
5.1a.4.6	Mapping on a burst	
5.1a.5	Packet data block type 16 (UAS-9)	
5.1a.5.1	Block constitution	
5.1a.5.2	Header coding	
5.1a.5.3	Data coding	
5.1a.5.4	PAN coding	

5.1a.5.5	Interleaving	228
5.1a.5.6	Mapping on a burst	
5.1a.6	Packet data block type 17 (UAS-10)	
5.1a.6.1	Block constitution	
5.1a.6.2	Header coding	
5.1a.6.3	Data coding	
5.1a.6.4	PAN coding	
5.1a.6.5	Interleaving	
5.1a.6.6	Mapping on a burst	
5.1a.0.0	Packet data block type 18 (UAS-11)	
5.1a.7 5.1a.7.1	Block constitution	
5.1a.7.1 5.1a.7.2	Header coding	
5.1a.7.2 5.1a.7.3	Data coding	
5.1a.7.3 5.1a.7.4		
	PAN coding	
5.1a.7.5	Interleaving	
5.1a.7.6	Mapping on a burst	
5.1a.8	Packet data block type 19 (UBS-5)	
5.1a.8.1	Block constitution	
5.1a.8.2	Header coding	
5.1a.8.3	Data coding	
5.1a.8.4	PAN coding	
5.1a.8.5	Interleaving	236
5.1a.8.6	Mapping on a burst	236
5.1a.9	Packet data block type 20 (UBS-6)	236
5.1a.9.1	Block constitution	236
5.1a.9.2	Header coding	237
5.1a.9.3	Data coding	237
5.1a.9.4	PAN coding	
5.1a.9.5	Interleaving	
5.1a.9.6	Mapping on a burst	
5.1a.10	Packet data block type 21 (UBS-7)	
5.1a.10.1	Block constitution	
5.1a.10.2	Header coding	
5.1a.10.3	Data coding	
5.1a.10.4	PAN coding	
5.1a.10.4	Interleaving	
5.1a.10.5 5.1a.10.6	Mapping on a burst	
	Packet data block type 22 (UBS-8)	
5.1a.11		
5.1a.11.1	Block constitution	
5.1a.11.2	Header coding	
5.1a.11.3	Data coding	
5.1a.11.4	PAN coding	
5.1a.11.5	Interleaving	
5.1a.11.6	Mapping on a burst	
5.1a.12	Packet data block type 23 (UBS-9)	
5.1a.12.1	Block constitution	
5.1a.12.2	Header coding	
5.1a.12.3	Data coding	
5.1a.12.4	PAN coding	
5.1a.12.5	Interleaving	243
5.1a.12.6	Mapping on a burst	244
5.1a.13	Packet data block type 24 (UBS-10)	246
5.1a.13.1	Block constitution	
5.1a.13.2	Header coding	
5.1a.13.3	Data coding	
5.1a.13.4	PAN coding	
5.1a.13.5	Interleaving	
5.1a.13.6	Mapping on a burst	
5.1a.14	Packet data block type 25 (UBS-11)	
5.1a.14.1	Block constitution	
5.1a.14.2	Header coding	
5.1a.14.2 5.1a.14.3	Data coding	
J.1a.14.3	Data Coung	230

5.1a.14.4	PAN coding	251
5.1a.14.5	Interleaving	251
5.1a.14.6	Mapping on a burst	252
5.1a.15	Packet data block type 26 (UBS-12)	
5.1a.15.1	Block constitution	
5.1a.15.2	Header coding	
5.1a.15.3	Data coding	
5.1a.15.4	PAN coding	
5.1a.15.5	Interleaving	
5.1a.15.6	Mapping on a burst	
5.1a.16	Packet data block type 27 (DAS-5)	
5.1a.16.1	Block constitution	
5.1a.16.2 5.1a.16.3	USF coding Header coding	
5.1a.16.3 5.1a.16.4	Data coding	
5.1a.16.4 5.1a.16.5	PAN coding	
5.1a.16.6	Interleaving	
5.1a.16.7	Mapping on a burst	
5.1a.17	Packet data block type 28 (DAS-6)	
5.1a.17.1	Block constitution	
5.1a.17.2	USF coding	
5.1a.17.3	Header coding	
5.1a.17.4	Data coding	
5.1a.17.5	PAN coding	
5.1a.17.6	Interleaving	
5.1a.17.7	Mapping on a burst	257
5.1a.18	Packet data block type 29 (DAS-7)	257
5.1a.18.1	Block constitution	
5.1a.18.2	USF coding	
5.1a.18.3	Header coding	
5.1a.18.4	Data coding	
5.1a.18.5	PAN coding	
5.1a.18.6	Interleaving	
5.1a.18.7	Mapping on a burst	
5.1a.19	Packet data block type 30 (DAS-8)	
5.1a.19.1	Block constitution	
5.1a.19.2 5.1a.19.2.1	USF coding BTTI configuration	
5.1a.19.2.1 5.1a.19.2.2	RTTI configurations	
5.1a.19.2.2 5.1a.19.3	Header coding	
5.1a.19.4	Data coding	
5.1a.19.5	PAN coding	
5.1a.19.6	Interleaving	
5.1a.19.7	Mapping on a burst	
5.1a.20	Packet data block type 31 (DAS-9)	
5.1a.20.1	Block constitution	
5.1a.20.2	USF coding	263
5.1a.20.3	Header coding	263
5.1a.20.4	Data coding	263
5.1a.20.5	PAN coding	264
5.1a.20.6	Interleaving	
5.1a.20.7	Mapping on a burst	
5.1a.21	Packet data block type 32 (DAS-10)	
5.1a.21.1	Block constitution	
5.1a.21.2	USF coding	
5.1a.21.2.1	BTTI configuration	
5.1a.21.2.2	RTTI configurations	
5.1a.21.3	Header coding	
5.1a.21.4	Data coding	
5.1a.21.5 5.1a.21.6	PAN codingInterleaving	
5.1a.21.6 5.1a.21.7	Mapping on a burst	
.,.14.41./	171G1/1/1112 VIII G 1/U15L	

5.1a.22	Packet data block type 33 (DAS-11)	269
5.1a.22.1	Block constitution	
5.1a.22.2	USF coding	
5.1a.22.3	Header coding	
5.1a.22.4	Data coding	
5.1a.22.5	PAN coding	
5.1a.22.6	Interleaving	
5.1a.22.7	Mapping on a burst	
5.1a.23	Packet data block type 34 (DAS-12)	
5.1a.23.1	Block constitution	
5.1a.23.2	USF coding	
5.1a.23.3	Header coding	
5.1a.23.4	Data coding	
5.1a.23.5	PAN coding	
5.1a.23.6	Interleaving	
5.1a.23.7	Mapping on a burst	
5.1a.24	Packet data block type 35 (DBS-5)	
5.1a.24.1	Block constitution	
5.1a.24.2	USF coding	
5.1a.24.2.1	BTTI configuration	
5.1a.24.2.2	RTTI configurations	
5.1a.24.3	Header coding	
5.1a.24.4	Data coding	
5.1a.24.5	PAN coding	
5.1a.24.6	Interleaving	
5.1a.24.7	Mapping on a burst	
5.1a.25	Packet data block type 36 (DBS-6)	
5.1a.25.1	Block constitution	
5.1a.25.2	USF coding	
5.1a.25.3	Header coding	
5.1a.25.4	Data coding	
5.1a.25.5	PAN coding	
5.1a.25.6	Interleaving	
5.1a.25.7	Mapping on a burst	
5.1a.26	Packet data block type 37 (DBS-7)	
5.1a.26.1	Block constitution	
5.1a.26.2	USF coding	
5.1a.26.2.1	BTTI configuration	
5.1a.26.2.2	RTTI configurations	
5.1a.26.3	Header coding	
5.1a.26.4	Data coding	
5.1a.26.5	PAN coding	
5.1a.26.6	Interleaving	
5.1a.26.7	Mapping on a burst	
5.1a.27	Packet data block type 38 (DBS-8)	
5.1a.27.1	Block constitution	
5.1a.27.1 5.1a.27.2	USF coding	
5.1a.27.3	Header coding	
5.1a.27.4	Data coding	
5.1a.27.5	PAN coding	
5.1a.27.6	Interleaving	
5.1a.27.7	Mapping on a burst	
5.1a.28	Packet data block type 39 (DBS-9)	
5.1a.28.1	Block constitution	
5.1a.28.2	USF coding	
5.1a.28.3	Header coding	
5.1a.28.4	Data coding	
5.1a.28.5	PAN coding	
5.1a.28.6	Interleaving	
5.1a.28.7	Mapping on a burst	
5.1a.29	Packet data block type 40 (DBS-10)	
5.1a.29.1	Block constitution	
J.14.47.1	2104 CONDUCTION	ر ۵ کے ۔۔۔۔۔۔۔۔۔۔۔۔۔

5.1a.29.2	USF coding	289
5.1a.29.2.	1 BTTI configuration	289
5.1a.29.2.	6	
5.1a.29.3	Header coding	
5.1a.29.4	Data coding	
5.1a.29.5	PAN coding	
5.1a.29.6	Interleaving	
5.1a.29.7	Mapping on a burst	
5.1a.30	Packet data block type 41 (DBS-11)	
5.1a.30.1	Block constitution	
5.1a.30.2	USF coding	
5.1a.30.3	Header coding	
5.1a.30.4	Data coding	
5.1a.30.5	PAN coding	
5.1a.30.6	Interleaving	
5.1a.30.7	Mapping on a burst	
5.1a.31	Packet data block type 42 (DBS-12)	
5.1a.31.1	Block constitution	
5.1a.31.2	USF coding	
5.1a.31.3	Header coding	
5.1a.31.4	Data coding	
5.1a.31.5	PAN coding	
5.1a.31.6 5.1a.31.7	Interleaving	
	Mapping on a burst	
5.1b 5.1b.1	General	
5.1b.1 5.1b.2	Downlink MCS-1/M	
5.1b.2.1	Block constitution	
5.1b.2.1 5.1b.2.2	Encoding and blind physical layer transmission	
5.1b.2.2	Mapping onto PDCHs	
5.1b.3	Uplink MCS-1/M	
5.1b.3.1	Block constitution	
5.1b.3.2	Encoding and blind physical layer transmission	
5.1b.3.3	Mapping onto PDCHs	
5.1b.4	Uplink MCS-1'/48	
5.1b.4.1	Block constitution	
5.1b.4.2	Header coding	
5.1b.4.3	Data coding	
5.1b.4.4	Interleaving	
5.1b.4.5	Mapping on a burst	303
5.1b.4.6	Blind physical layer transmission	303
5.1b.4.7	Mapping onto PDCHs	304
5.2	Packet control channels (PACCH, PBCCH, PAGCH, PPCH, PTCCH, CPBCCH, CPAGCH and	
	CPPCH)	
5.2a	Extended Coverage broadcast control channel (EC-BCCH)	
5.2a.1	Block constitution	
5.2a.2	Coding	
5.2a.3	Blind physical layer transmission	
5.2a.4	Mapping onto a physical channel	305
5.2b	Extended coverage common control channel downlink (EC-CCCH/D/M), Extended coverage paging	
	indication channel (EC-PICH) and Extended coverage packet associated control channel (EC-	201
5 Ol 1	PACCH/M)	
5.2b.1	Block constitution	
5.2b.2	Data coding	
5.2b.3	USF coding (only for EC-PACCH/D)	
5.2b.4 5.2b.4.1	Mapping on a Burst EC-CCCH/D/M and EC-PICH	306
5.2b.4.1 5.2b.4.2	EC-PACCH/D/M and EC-PICH	
5.2b.4.2 5.2b.4.3	EC-PACCH/U/M	
5.2b.4.5 5.2b.5	Blind physical layer transmission	
5.2b.6	Bit stealing for USF (only for EC-PACCH/D)	
5.2b.7	Mapping onto physical channels.	

5.2b.7.1		150 NAV	
5.2b.7.2		and EC-PICH	
5.3		ss channel (PRACH, CPRACH and MPRACH)	
5.3.1		urst	
5.3.2		Access Burst	
5.3.2.1		ution	
5.3.2.2	•	ora	
5.3.2.3		SIC	
5.3.2.4		l code	
5.3a		andom access channel (EC-RACH/M)	
5.3a.0			
5.3a.1		n	
5.3a.2			
5.3a.3		yer transmissions	
5.3a.4		ysical channel(s)	
5.3a.4.1		CH	
5.3a.4.2		CH	
5.3a.5		ing 30 information bits	
5.3a.6		ge random access channel (EC-RACH/66)	
5.3a.6.0			
5.3a.6.1		ution	
5.3a.6.2			
5.3a.6.3		SIC	
5.3a.6.4		onvolutional code	
5.3a.6.5		l layer transmissions	
5.3a.6.6		p physical channel(s)	
5.3a.7		ge random access channel (EC-RACH/132)	
5.3a.7.0		-tion	
5.3a.7.1		ution	
5.3a.7.2	•	SIC	
5.3a.7.3 5.3a.7.4		l code	
5.3a.7.4 5.3a.7.5		l layer transmissions	
5.3a.7.5		p physical channel(s)	
5.4		ket switched channels other than PRACH, CPRACH and MPRACH	
	1		
	•		
6.1			
6.2		oding/multiplexing	
6.2.1			
6.2.2	_		
6.2.3			
6.2.4	•	el multiplexing	
6.2.5			
6.2.6	_	g encoding	
6.2.7		pping	
6.2.8	C		
6.2.9		ırst	
6.2.10	Signalling on Ha	If Rate Channels	343
Annex A	A (informative):	Summary of Channel Types	344
Annex 1	B (informative):	Summary of Polynomials Used for Convolutional Codes and Turbo	
		Codes	347
Annex	C (informative):	Change history	348
History			352

### **Foreword**

This Technical Specification has been produced by the 3<sup>rd</sup> Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

#### where:

- x the first digit:
  - 1 presented to TSG for information;
  - 2 presented to TSG for approval;
  - 3 or greater indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the document.

### 1 Scope

A reference configuration of the transmission chain is shown in 3GPP TS 45.001. According to this reference configuration, the present document specifies the data blocks given to the encryption unit.

It includes the specification of encoding, reordering, interleaving and the stealing flag. It does not specify the channel decoding method.

The definition is given for each kind of logical channel, starting from the data provided to the channel encoder by the speech coder, the data terminal equipment, or the controller of the Mobile Station (MS) or Base Transceiver Station (BTS). The definitions of the logical channel types used in this technical specification are given in 3GPP TS 45.002, a summary is in annex A.

Additionally, the present document describes the characteristics of the coding/multiplexing unit for the Flexible Layer One (FLO) starting from the transport blocks provided by higher layers. An overview of FLO is given in 3GPP TR 45.902.

### 1.1 References

[15]

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1]	3GPP TR 21.905: "Vocabulary for 3GPP Specifications".
[2]	3GPP TS 26.090: "AMR speech Codec; Transcoding Functions".
[3]	3GPP TS 26.190: "Mandatory Speech Codec speech processing functions AMR Wideband speech codec; Transcoding functions".
[4]	3GPP TS 44.018: "Mobile radio interface layer 3 specification, Radio Resource Control Protocol".
[5]	3GPP TS 44.021: "Rate adaption on the Mobile Station - Base Station System (MS - BSS) interface".
[6]	3GPP TS 44.060: "General Packet Radio Service (GPRS); Mobile Station (MS) - Base Station System (BSS) interface; Radio Link Control/ Medium Access Control (RLC/MAC) protocol".
[7]	3GPP TS 45.001: "Physical Layer on the Radio Path (General Description)".
[8]	3GPP TS 45.002: "Multiplexing and multiple access on the radio path".
[9]	3GPP TS 45.004: "Modulation".
[10]	3GPP TS 45.008: "Radio subsystem link control".
[11]	3GPP TS 45.009: "Link adaptation".
[12]	3GPP TR 45.902: "Flexible Layer One".
[13]	3GPP TS 46.010: "Full rate speech transcoding".
[14]	3GPP TS 46.020: "Half rate speech transcoding".

3GPP TS 46.060: "Enhanced full rate speech transcoding".

### 1.2 Abbreviations

Abbreviations used in the present document are listed in 3GPP TR 21.905. In addition to abbreviations in 3GPP TR 21.905 the following abbreviations apply:

**BTTI Basic Transmission Time Interval ESAB Extended Synchronization Access Burst EDAB** Extended Dual slot Access Burst eTFI Extended Temporary Flow Identity **FANR** Fast Ack/Nack Reporting **PAN** Piggy-backed Ack/Nack Piggy-backed Ack/Nack Indicator **PANI RTTI** Reduced Transmission Time Interval

TFI Temporary Flow Identity
TTI Transmission Time Interval

### 2 General

### 2.1 General organization

Each channel has its own coding and interleaving scheme. However, the channel coding and interleaving is organized in such a way as to allow, as much as possible, a unified decoder structure.

Each channel uses the following sequence and order of operations:

- the information bits are coded with a systematic block code, building words of information + parity bits;
- these information + parity bits are encoded with a convolutional code or a turbo code, building the coded bits;
- reordering and interleaving the coded bits, and adding a stealing flag, gives the interleaved bits.

All these operations are made block by block, the size of which depends on the channel. However, most of the channels use a block of either 456 coded bits or 1368 coded bits, corresponding to 456 coded symbols, which is interleaved and mapped onto bursts in a very similar way for all of them. This block of 456 coded symbols is the basic structure of the channel coding scheme. Figures 1a, 1b, 1c, 1d, 1ea, 1eb, 1f, 1g, 1h, 1i, 1j, 1k, 1l, 2 and 2aa give diagrams showing the general structure of the channel coding.

In the case of full rate speech TCH, a block of 456 coded bits carries the information of one speech frame. In case of control channels, it carries one message.

In the case of half rate speech TCH, the information of one speech frame is carried in a block of 228 coded bits.

In the case of the Enhanced full rate speech the information bits coming out of the source codec first go through a preliminary channel coding. Then the channel coding as described above takes place.

In the case of 8-PSK modulated speech TCH, the information of one speech frame is carried in a block of 1368 coded bits (456 coded symbols) for full rate channels or 684 coded bits (228 coded symbols) for half rate channels.

In the case of a packet switched channel the block of 456, 1096, 1384, 1848, 2200, 2312 or 2748 coded bits carries one RLC/MAC block.

In the case of E-TCH/F28.8 or E-TCH/F43.2, the block of 1368 coded bits (456 coded symbols) carries one radio interface data block. In the case of E-TCH/F32.0, the block of 1392 coded bits (464 coded symbols) carries one radio interface data block.

In the case of FACCH, a coded message block of 456 bits is divided into eight sub-blocks. The first four sub-blocks are sent by stealing the even numbered bits of four timeslots in consecutive frames used for the TCH. The other four sub-blocks are sent by stealing the odd numbered bits of the relevant timeslot in four consecutive used frames delayed 2 or 4 frames relative to the first frame. Along with each block of 456 coded bits there is, in addition, a stealing flag (8 bits), indicating whether the block belongs to the TCH or to the FACCH. In the case of SACCH, BCCH, CCCH or

CTSCCH, this stealing flag is dummy. In the case of a packet switched channel, these bits are used to indicate the coding scheme used.

In the case of E-FACCH/F, a coded message block of 456 bits is divided into four sub-blocks. The four sub-blocks are sent by stealing all symbols of four timeslots in consecutive frames used for the E-TCH and using GMSK modulation. The indication of the E-FACCH/F is based on the identification of the modulation. Along with each block of 456 coded bits there is, in addition, a stealing flag (8 bits), indicating whether the block belongs to the E-FACCH, FACCH or TCH

Some cases do not fit in the general organization, and use short blocks of coded bits which are sent completely in one timeslot. They are the random access messages of:

- the RACH;
- or PRACH, CPRACH and MPRACH:

on uplink and the synchronization information broadcast on the SCH or CSCH on the downlink. In CTS, they are the access request message of the CTSARCH on uplink and the information broadcast on the CTSBCH-SB on downlink.

In the coding/multiplexing unit of FLO, error detection, forward error correction and rate matching is applied to each transport channel independently. However the transport channels share a common multiplexing, TFCI mapping, interleaving and burst mapping. All these operations are made every transmission time interval and the number of coded bits produced by the coding/multiplexing unit depends on the basic physical subchannel. In the case of full rate GMSK basic physical subchannel, blocks of 464 bits are produced. In the case of half rate GMSK basic physical subchannel, blocks of 1392 bits are produced. In the case of half rate 8PSK basic physical subchannel, blocks of 696 bits are produced.

For EC-GSM-IoT, the same nine modulation and coding schemes as for EGPRS are defined for the extended coverage packet data traffic channels (EC-PDTCH). In addition three modulation and coding schemes employing MCS-1 and blind physical layer transmissions are defined for EC-PDTCH. For extended coverage common control channels in downlink (EC-CCCH/D) and the extended coverage packet associated control channel (EC-PACCH/U and EC-PACCH/D), three coding schemes having the same structure, differing mainly in message size and puncturing, are defined. The encoded bits are mapped onto a single burst, which is transmitted using blind physical layer transmissions. The extended coverage broadcast control channel (EC-BCCH) uses the same coding scheme as the broadcast control channel (BCCH) with subsequent blind physical layer transmissions. The extended coverage synchronization channel (EC-SCH) uses the same coding scheme as the synchronization channel (EC-RACH) uses the same coding scheme as the random access channel (RACH), with subsequent blind physical layer transmissions, with the exception of the EC-RACH used for coverage class CC5, that applies modified coding schemes with subsequent blind physical layer transmissions. Figures 1m, 1n, 2ab, 2ac, 2ad, 2ae,2af and 2ag give diagrams showing the general structure of the channel coding for EC-channels.

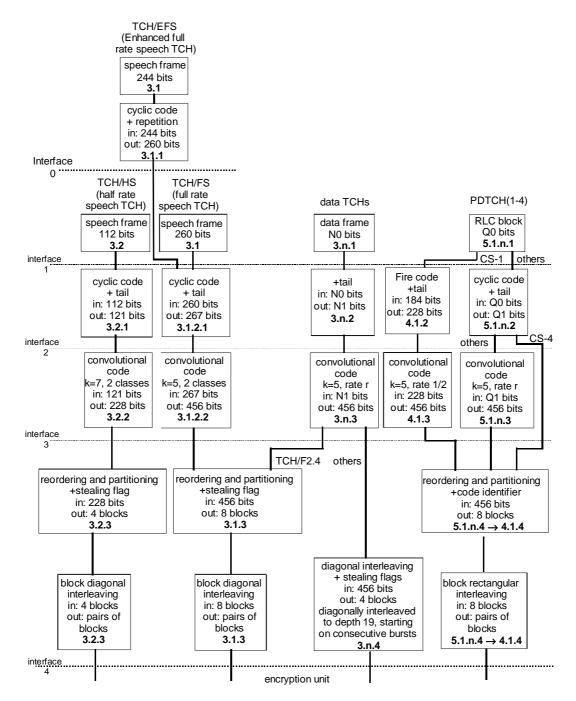


Figure 1a: Channel Coding and Interleaving Organization for speech, circuit switched data and GPRS packet data channels

In each box, the last line indicates the subclause defining the function. In the case of data TCHs, N0, N1 and n depend on the type of data TCH. In the case of PDTCH, Q0, Q1 and n depend on the coding scheme.

### Interfaces:

- 0) speech bits from the speech encoder (s);
- 1) information bits (d);
- 2) information + parity + tail bits (u);
- 3) coded bits (c);
- 4) interleaved bits (e).

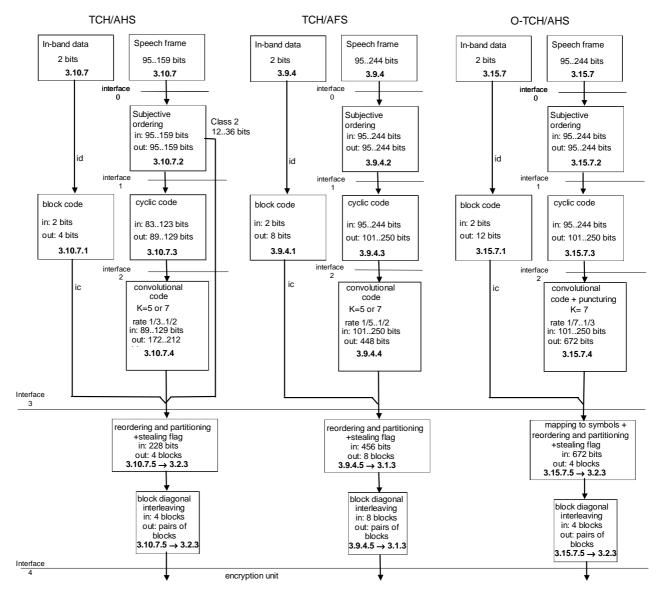


Figure 1b: Channel Coding and Interleaving Organization, adaptive multi-rate speech

### Interfaces:

- 0) speech bits from the speech encoder (s);
- 1) reordered speech bits (d);
- 2) speech + parity + tail bits (u);
- 3) coded bits (c);
- 4) interleaved bits (e).

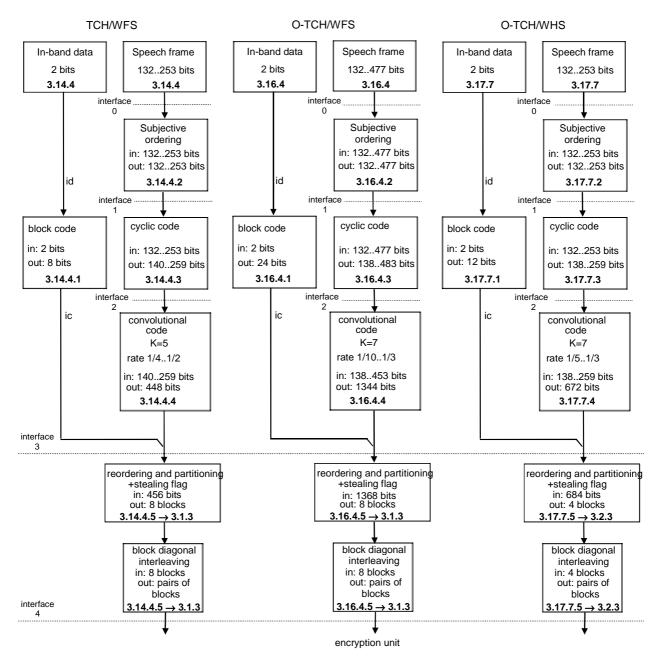


Figure 1c: Channel Coding and Interleaving Organization, wide-band adaptive multi-rate speech

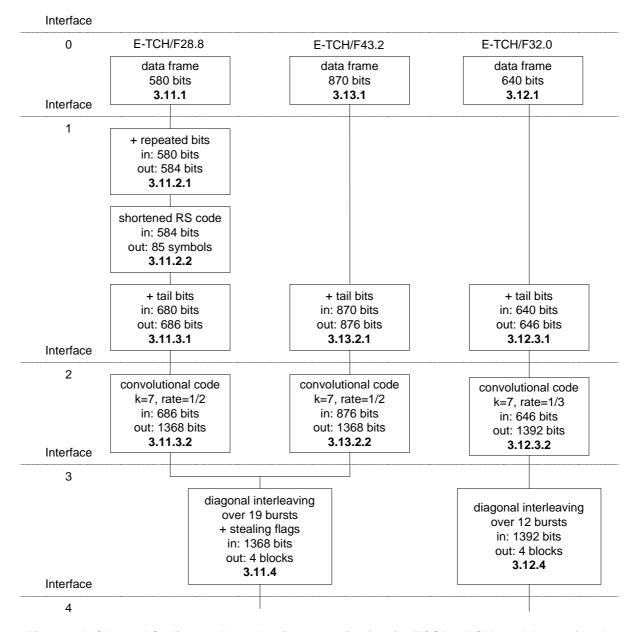


Figure 1d: Channel Coding and Interleaving Organization for ECSD 8-PSK modulated signals

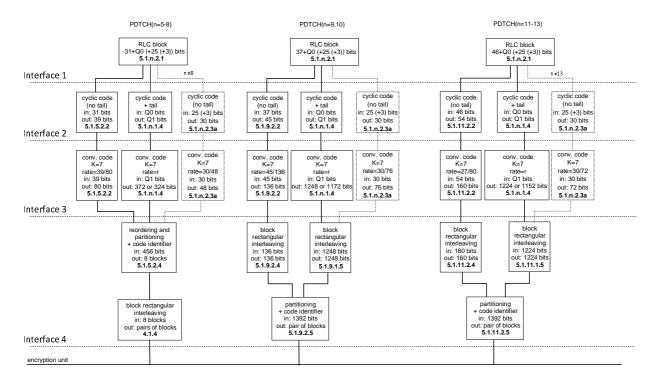


Figure 1ea: Channel Coding and Interleaving Organization for EGPRS Uplink Packet Data Channels

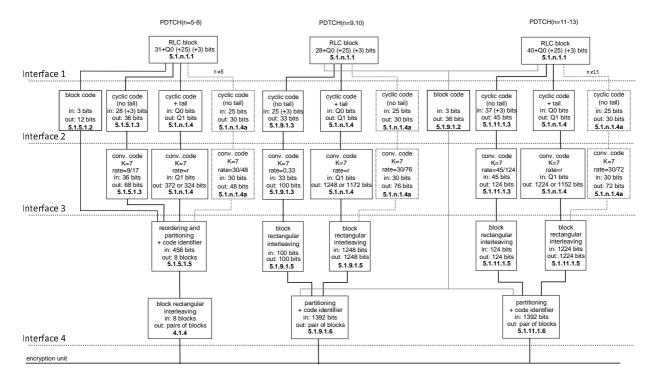


Figure 1eb: Channel Coding and Interleaving Organization for EGPRS Downlink Packet Data Channels

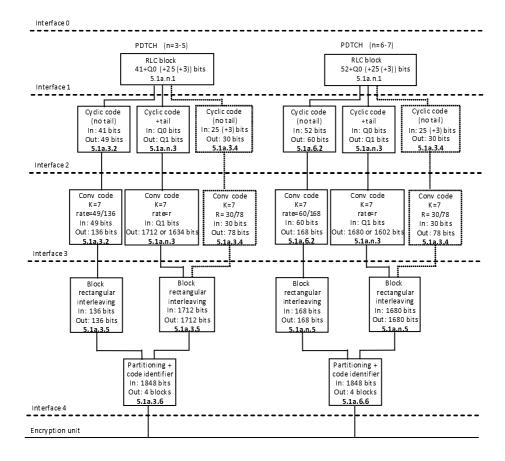


Figure 1f: Channel Coding and Interleaving Organization for EGPRS2-A Uplink Packet Data Channels

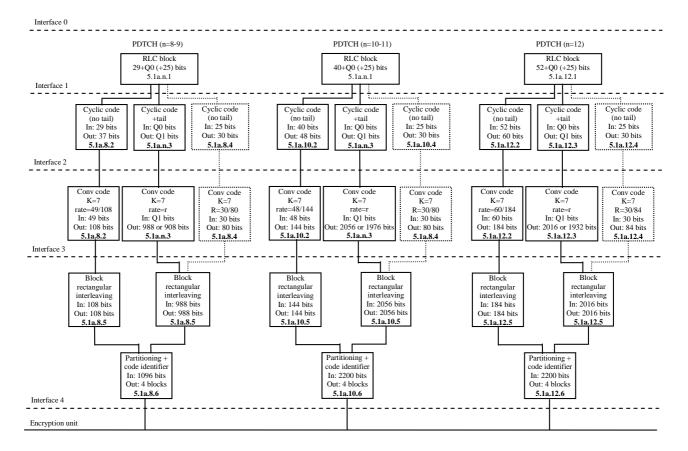


Figure 1g: Channel Coding and Interleaving Organization for EGPRS2-B Uplink Packet Data Channels, UBS-5 to UBS-9

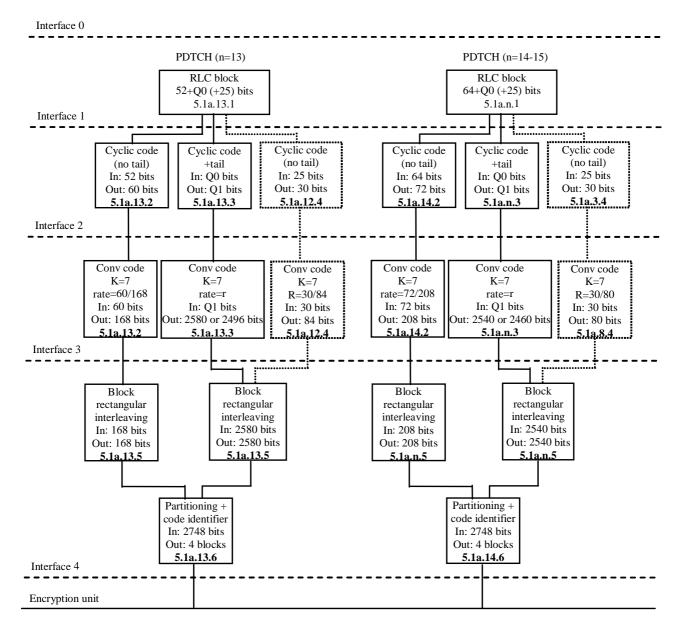


Figure 1h: Channel Coding and Interleaving Organization for EGPRS2-B Uplink Packet Data Channels, UBS-10 to UBS-12

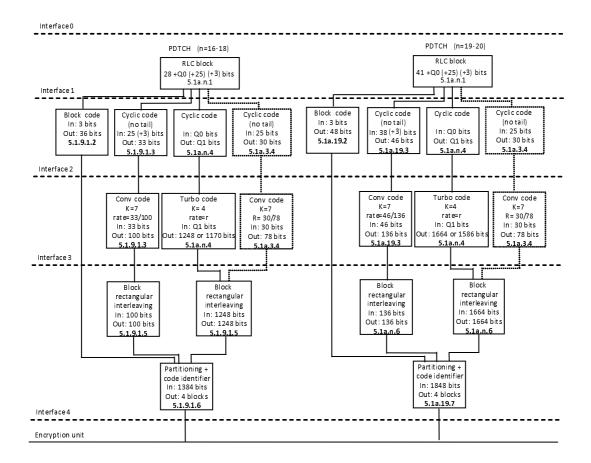


Figure 1i: Channel Coding and Interleaving Organization for EGPRS2-A Downlink Packet Data Channels, DAS-5 to DAS-9

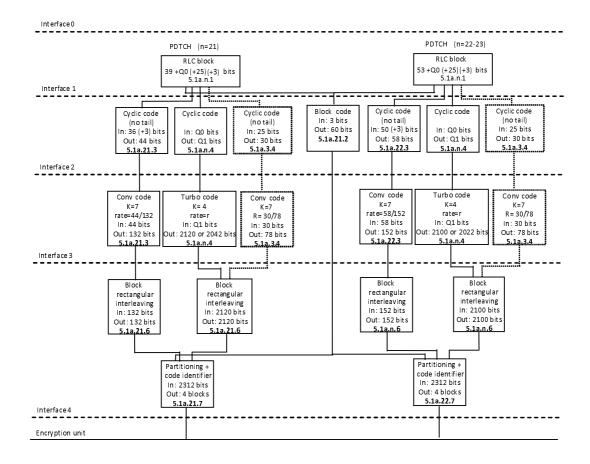


Figure 1j: Channel Coding and Interleaving Organization for EGPRS2-A Downlink Packet Data Channels, DAS-10 to DAS-12

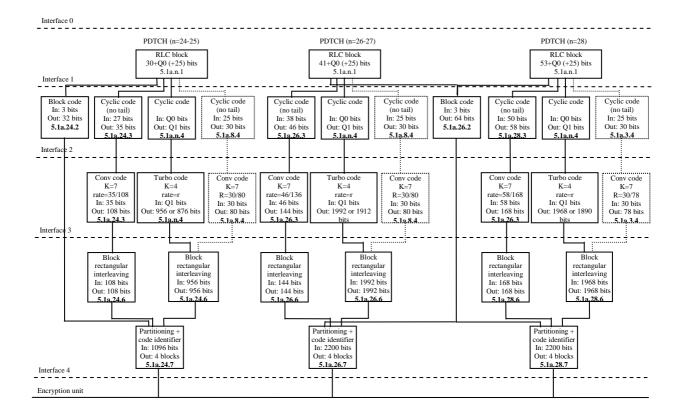


Figure 1k: Channel Coding and Interleaving Organization for EGPRS2-B Downlink Packet Data Channels, DBS-5 to DBS-9

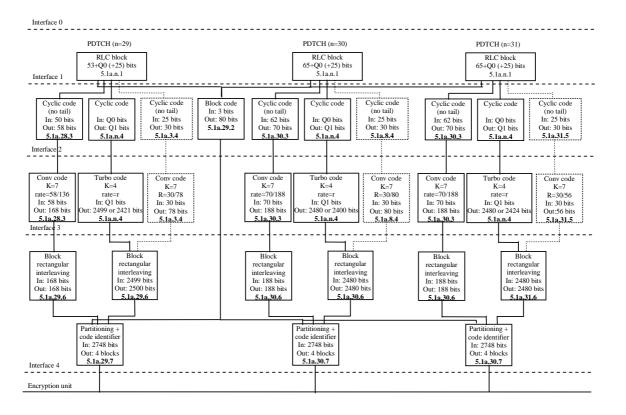


Figure 1I: Channel Coding and Interleaving Organization for EGPRS2-B Downlink Packet Data Channels, DBS-10 to DBS-12

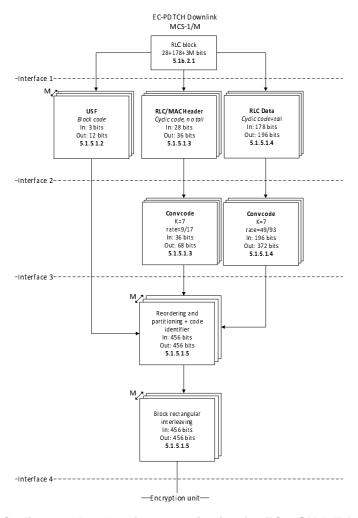


Figure 1m: Channel Coding and Interleaving Organization for EC-GSM-IoT Downlink Packet Data Traffic Channels

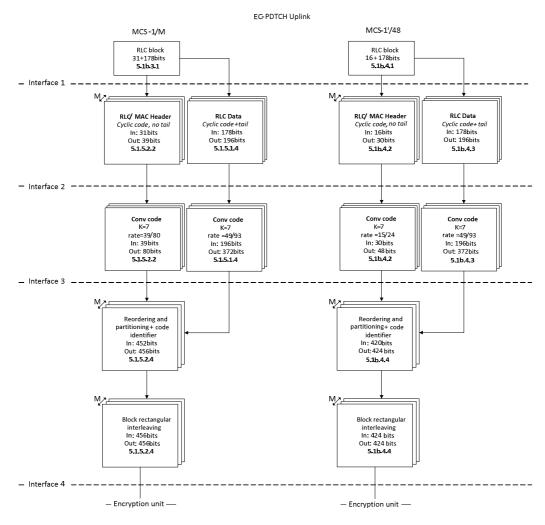


Figure 1n: Channel Coding and Interleaving Organization for EC-GSM-IoT Uplink Packet Data Traffic Channels (MCS-1/M is used for CC1 to CC4, MCS-1'/48 for CC5.)

In each box, the last line indicates the subclause defining the function. M denotes the number of blind physical layer transmissions, see subclause 5.1b.2.2 and 5.1b.3.2 for the downlink and uplink respectively.

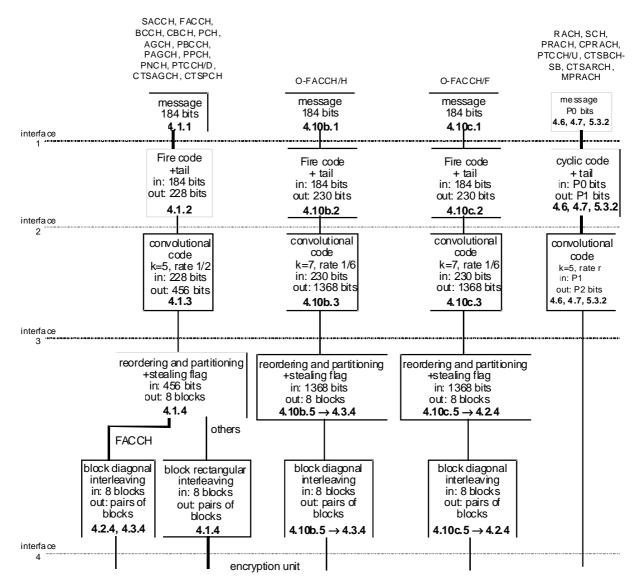


Figure 2: Channel Coding and Interleaving Organization for control channels except PACCH, EC-BCCH, EC-CCCH and EC-PACCH

In each box, the last line indicates the subclause defining the function. In the case of RACH, PRACH and of MPRACH using Packet Access Burst, P0 = 8 and P1 = 18; in the case of PRACH and of MPRACH using Extended Packet Access Burst, P0 = 11 and P1 = 21; in the case of SCH, CSCH, CTSBCH-SB and CTSARCH, P0 = 25 and P1 = 39.

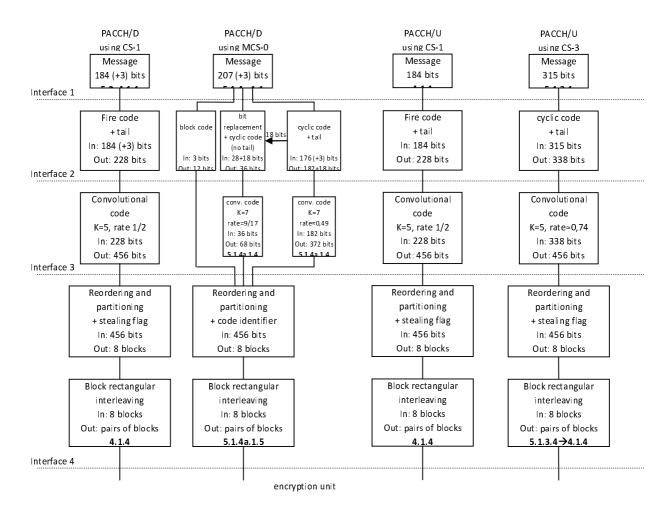


Figure 2aa: Channel Coding and Interleaving Organization for PACCH

In each box, the last line indicates the subclause defining the function.

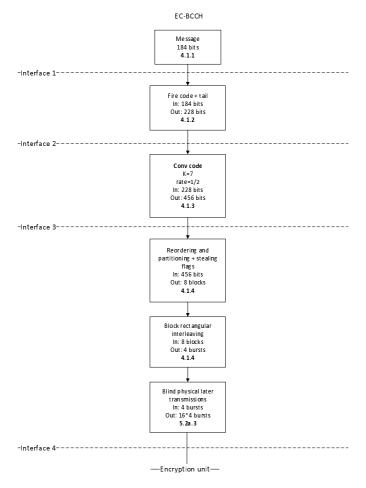


Figure 2ab: Channel Coding and Interleaving Organization for EC-BCCH

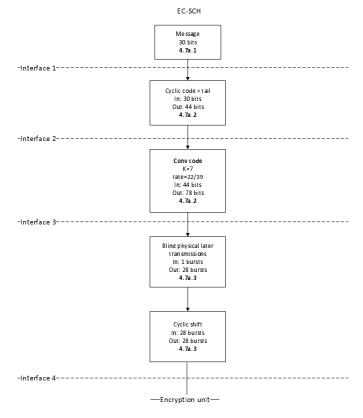


Figure 2ac: Channel Coding and Interleaving Organization for EC-SCH

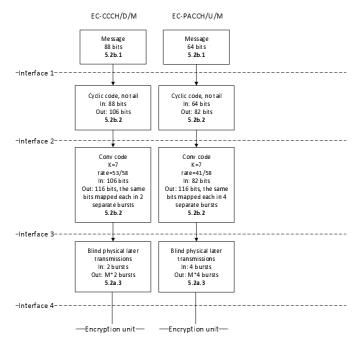


Figure 2ad: Channel Coding and Interleaving Organization for EC-CCCH/D and EC-PACCH/U

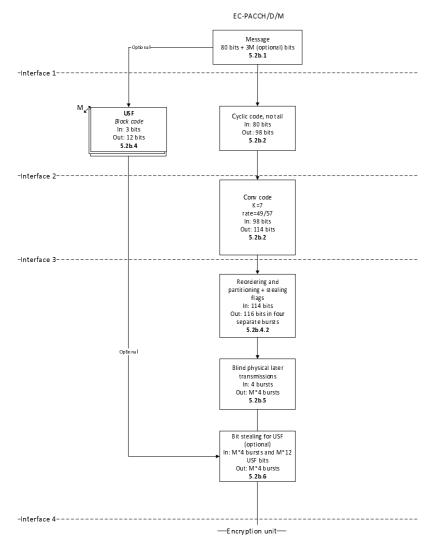


Figure 2ae: Channel Coding and Interleaving Organization for EC-PACCH/D

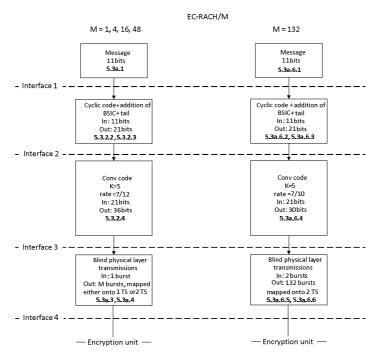


Figure 2af: Channel Coding and Interleaving Organization for EC-RACH (M=1, 4, 16, 48 is used for CC1, CC2, CC3, CC4 and M=132 for CC5).

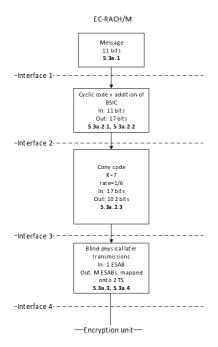


Figure 2ag: Channel Coding Organization for EC-RACH (M=66)

In each box, the last line indicates the subclause defining the function. M denotes the number of blind physical layer transmissions.

# 2.2 Naming Convention

For ease of understanding a naming convention for bits is given for use throughout the technical specification:

- General naming:

"k" and "j" for numbering of bits or symbols in data blocks and bursts;

" $K_x$ " gives the amount of bits or symbols in one block, where "x" refers to the data type;

"n" is used for numbering of delivered data blocks where;

"N" marks a certain data block;

"m" is used for numbering of blind physical layer transmissions where;

"M" gives the number of transmissions;

"B" is used for numbering of bursts or blocks where;

" $B_0$ " marks the first burst or block carrying bits from the data block with n = 0 (first data block in the transmission).

- Data delivered to the preliminary channel encoding unit (for EFR only):

$$s(k)$$
 for  $k = 1..., K_s$ 

- Data delivered by the preliminary channel encoding unit (for EFR only) before bits rearrangement

$$w(k)$$
 for  $k = 1..., K_w$ 

- Data bits delivered to the encoding unit (interface 1 in figures 1a, 1b, 1c, 1ea, 1eb, 2 and 2aa):

d(k) for 
$$k = 0,1,...,K_{d}-1$$

- Data bits delivered to the encoding unit for EC-GSM-IoT MCS-1/M:

a(k) for 
$$k = 0, 1, ..., K_a-1$$

- Data symbols delivered to the encoding unit (interface 1 in figure 1d):

D(k) for 
$$k = 0,1,...,K_{D}-1$$

- Input in-band data bits (for TCH/AMR only):

$$id(k)$$
 for  $k = 0, 1$ 

- Encoded in-band data bits (for TCH/AMR only):

ic(k) for k = 0, 1,...,3 TCH/AHS speech frames or 
$$k=0,\,1,...,7 \, \text{TCH/AFS or TCH/WFS speech frames or}$$
 
$$k=0,\,1,...,11 \, \text{O-TCH/AHS or O-TCH/WHS speech frames or}$$
 
$$k=0,\,1,...,23 \, \text{O-TCH/WFS speech frames or}$$
 
$$k=0,\,1,...,23 \, \text{O-TCH/WFS speech frames}$$
 or 
$$k=0,\,1,...,15 \, \text{TCH/AMR SID and RATSCCH frames}$$

- Code identifying the used coding scheme (for packet switched channels only):

$$q(k)$$
 for  $k = 0,1,...,7$ 

Data bits after the first encoding step (block code, cyclic code; interface 2 in figures 1a, 1b, 1c, 1ea, 1eb, 2 and 2aa):

$$u(k)$$
 for  $k = 0,1,...,K_{11}-1$ 

- Data symbols after the first encoding step (block code; interface 2 in figure 1d):

U(k) for 
$$k = 0,1,...,K_{IJ}-1$$

- Data put into the shift register of the convolutional code and calculated from the data bits u(k) and the feedback bits in recursive systematic convolutional codes

$$r(k)$$
 for  $k = 0,1,..., K_{r}-1$ 

- Data after the second encoding step (convolutional code; interface 3 in figures 1a, 1b, 1c, 1d, 1ea, 1eb, 2 and 2aa):

$$c(n,k)$$
 or  $c(k)$  for  $k = 0,1,...,K_c-1$ 

$$n = 0,1,...,N,N+1,...$$

- Interleaved data bits:

$$i(B,k)$$
 for  $k = 0,1,...,K_i-1$ 

$$B = B_0, B_0 + 1,...$$

- Interleaved data symbols:

$$I(B,k)$$
 for  $k = 0,1,...,K_{I}-1$ 

$$B = B_0, B_0 + 1,...$$

- Bits in one burst (interface 4 in figures 1a, 1b, 1c, 1ea, 1eb, 2 and 2aa):

$$e(B,k)$$
 for  $k = 0,1,...,114,115$ 

$$B = B_0, B_0 + 1, ...$$

- Symbols in one burst (interface 4 in figure 1d):

E(B,k) for 
$$k = 0,1,...,114,115$$

$$B = B_0, B_0 + 1, ...$$

- Bits in transmission m of burst B after blind physical layer transmissions:

$$R(m,B,k)$$
 for  $k = 0,1,...,K_R-1$ 

$$B = 0.1...$$

$$m = 0,...,M-1$$

- E-IACCH messages delivered to the block coding of inband signalling (for ECSD only):

im(k) or im(n,k)

for 
$$k = 0,1,2$$

$$n = 0,1,...,N,N+1,...$$

- E-IACCH bits delivered to the mapping on one burst (for ECSD only):

ib(B,k) for 
$$k = 0,1,...,5$$

$$B = B_0, B_0 + 1,...$$

- E-IACCH symbols in one burst (for ECSD only):

HL(B) and HU(B)

for 
$$B = B_0, B_0 + 1,...$$

- EPCCH messages delivered to the block coding (for SACCH/TP only):

pm(k) or pm(n,k)

for 
$$k = 0,1,2$$

$$n = 0,1,...,N,N+1,...$$

- EPCCH bits delivered to the mapping on one burst (for SACCH/TP only):

$$pb(B,k)$$
 for  $k = 0,1,...,11$  
$$B = B_0, B_0 + 1,....$$

# 3 Traffic Channels (TCH)

Two kinds of traffic channel are considered: speech and data. Both of them use the same general structure (see figure 1), and in both cases, a piece of information can be stolen by the FACCH.

# 3.1 Speech channel at full rate (TCH/FS and TCH/EFS)

The speech coder (whether Full rate or Enhanced full rate) delivers to the channel encoder a sequence of blocks of data. In case of a full rate and enhanced full rate speech TCH, one block of data corresponds to one speech frame.

For the full rate coder each block contains 260 information bits, including 182 bits of class 1 (protected bits), and 78 bits of class 2 (no protection), (see table 2).

The bits delivered by the speech coder are received in the order indicated in 3GPP TS 46.010 and have to be rearranged according to table 2 before channel coding as defined in subclauses 3.1.1 to 3.1.4. The rearranged bits are labelled  $\{d(0),d(1),...,d(259)\}$ , defined in the order of decreasing importance.

For the EFR coder each block contains 244 information bits. The block of 244 information bits, labelled s(1)..., s(244), passes through a preliminary stage, applied only to EFR (see figure 1) which produces 260 bits corresponding to the 244 input bits and 16 redundancy bits. Those 16 redundancy bits correspond to 8 CRC bits and 8 repetition bits, as described in subclause 3.1.1. The 260 bits, labelled w(1)..w(260), have to be rearranged according to table 6 before they are delivered to the channel encoding unit which is identical to that of the TCH/FS. The 260 bits block includes 182 bits of class 1(protected bits) and 78 bits of class 2 (no protection). The class 1 bits are further divided into the class 1a and class 1b, class 1a bits being protected by a cyclic code and the convolutional code whereas the class 1b are protected by the convolutional code only.

# 3.1.1 Preliminary channel coding for EFR only

## 3.1.1.1 CRC calculation

An 8-bit CRC is used for error-detection. These 8 parity bits (bits w253-w260) are generated by the cyclic generator polynomial:  $g(D) = D^8 + D^4 + D^3 + D^2 + 1$  from the 65 most important bits (50 bits of class 1a and 15 bits of class 1b). These 65 bits (b(1)-b(65)) are taken from the table 5 in the following order (read row by row, left to right):

s39	s40	s41	s42	s43	s44	s48	s87	s45	s2
s3	s8	s10	s18	s19	s24	s46	s47	s142	s143
s144	s145	s146	s147	s92	s93	s195	s196	s98	s137
s148	s94	s197	s149	s150	s95	s198	s4	s5	s11
s12	s16	s9	s6	s7	s13	s17	s20	s96	s199
s1	s14	s15	s21	s25	s26	s28	s151	s201	s190
s240	s88	s138	s191	s241					

The encoding is performed in a systematic form, which means that, in GF(2), the polynomial:

- $b(1)D^{72} + b(2)D^{71} + ... + b(65)D^8 + p(1)D^7 + p(2)D^6 + ... + p(7)D^1 + p(8);$
- p(1) p(8): the parity bits (w253-w260);
- b(1) b(65) = the data bits from the table above;

when divided by g(D), yields a remainder equal to 0.

#### 3.1.1.2 Repetition bits

The repeated bits are s70, s120, s173 and s223. They correspond to one of the bits in each of the PULSE\_5, the most significant one not protected by the channel coding stage.

## 3.1.1.3 Correspondence between input and output of preliminary channel coding

The preliminary coded bits w(k) for k = 1 to 260 are hence defined by:

$$w(k) = s(k)$$
 for  $k = 1$  to 71

$$w(k) = s(k-2)$$
 for  $k = 74$  to 123

$$w(k) = s(k-4)$$
 for  $k = 126$  to 178

$$w(k) = s(k-6)$$
 for  $k = 181$  to  $s230$ 

$$w(k) = s(k-8)$$
 for  $k = 233$  to  $s252$ 

#### Repetition bits:

$$w(k) = s(70)$$
 for  $k = 72$  and 73

$$w(k) = s(120)$$
 for  $k = 124$  and 125

$$w(k) = s(173)$$
 for  $k = 179$  and 180

$$w(k) = s(223)$$
 for  $k = 231$  and 232

Parity bits:

$$w(k = p(k-252) \text{ for } k = 253 \text{ to } 260$$

# 3.1.2 Channel coding for FR and EFR

#### 3.1.2.1 Parity and tailing for a speech frame

a) Parity bits:

The first 50 bits of class 1 (**known as class 1a for the EFR**) are protected by three parity bits used for error detection. These parity bits are added to the 50 bits, according to a degenerate (shortened) cyclic code (53,50,2), using the generator polynomial:

$$g(D) = D^3 + D + 1$$

The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

$$d(0)D^{52}+d(1)D^{51}+...+d(49)D^{3}+p(0)D^{2}+p(1)D+p(2)\\$$

where p(0), p(1), p(2) are the parity bits, when divided by g(D), yields a remainder equal to:

$$1 + D + D^2$$

b) Tailing bits and reordering:

The information and parity bits of class 1 are reordered, defining 189 information + parity + tail bits of class 1,  $\{u(0),u(1),...,u(188)\}$  defined by:

$$u(k) = d(2k)$$
 and  $u(184-k) = d(2k+1)$  for  $k = 0,1,...,90$ 

$$u(91+k) = p(k)$$
 for  $k = 0,1,2$ 

$$u(k) = 0$$
 for  $k = 185,186,187,188$  (tail bits)

#### 3.1.2.2 Convolutional encoder

The class 1 bits are encoded with the ½ rate convolutional code defined by the polynomials:

$$G0 = 1 + D^3 + D^4$$
  
 $G1 = 1 + D + D^3 + D^4$ 

The coded bits  $\{c(0), c(1), ..., c(455)\}$  are then defined by:

- class 1: 
$$c(2k) = u(k) + u(k-3) + u(k-4)$$
 
$$c(2k+1) = u(k) + u(k-1) + u(k-3) + u(k-4) \qquad \text{for } k = 0,1,...,188$$
 
$$u(k) = 0 \text{ for } k < 0$$
 - class 2:  $c(378+k) = d(182+k)$  for  $k = 0,1,...,77$ 

# 3.1.3 Interleaving

The coded bits are reordered and interleaved according to the following rule:

$$i(B,j) = c(n,k),$$
 for  $k = 0,1,...,455$   
 $n = 0,1,...,N,N+1,...$   
 $B = B_0 + 4n + (k \mod 8)$   
 $j = 2((49k) \mod 57) + ((k \mod 8) \operatorname{div} 4)$ 

See table 1. The result of the interleaving is a distribution of the reordered 456 bits of a given data block, n = N, over 8 blocks using the even numbered bits of the first 4 blocks ( $B = B_0 + 4N + 0$ , 1, 2, 3) and odd numbered bits of the last 4 blocks ( $B = B_0 + 4N + 4$ , 5, 6, 7). The reordered bits of the following data block, n = N+1, use the even numbered bits of the blocks  $B = B_0 + 4N + 4$ , 5, 6, 7 ( $B = B_0 + 4(N+1) + 0$ , 1, 2, 3) and the odd numbered bits of the blocks  $B = B_0 + 4(N+1) + 4$ , 5, 6, 7. Continuing with the next data blocks shows that one block always carries 57 bits of data from one data block (n = N) and 57 bits of data from the next block (n = N+1), where the bits from the data block with the higher number always are the even numbered data bits, and those of the data block with the lower number are the odd numbered bits.

The block of coded data is interleaved "block diagonal", where a new data block starts every 4<sup>th</sup> block and is distributed over 8 blocks.

# 3.1.4 Mapping on a Burst

The mapping is given by the rule:

$$e(B,j) = i(B,j)$$
 and  $e(B,59+j) = i(B,57+j)$  for  $j = 0,1,...,56$ 

and

$$e(B,57) = hl(B)$$
 and  $e(B,58) = hu(B)$ 

The two bits, labelled hl(B) and hu(B) on burst number B are flags used for indication of control channel signalling. For each TCH/FS block not stolen for signalling purposes:

```
hu(B) = 0 for the first 4 bursts (indicating status of even numbered bits)
```

$$hl(B) = 0$$
 for the last 4 bursts (indicating status of odd numbered bits)

For the use of hl(B) and hu(B) when a speech frame is stolen for signalling purposes see subclause 4.2.5.

# 3.2 Speech channel at half rate (TCH/HS)

The speech coder delivers to the channel encoder a sequence of blocks of data. In case of a half rate speech TCH, one block of data corresponds to one speech frame. Each block contains 112 bits, including 95 bits of class 1 (protected bits), and 17 bits of class 2 (no protection), see tables 3a and 3b.

The bits delivered by the speech coder are received in the order indicated in 3GPP TS 46.020 and have to be arranged according to either table 3a or table 3b before channel encoding as defined in subclauses 3.2.1 to 3.2.4. The rearranged bits are labelled  $\{d(0),d(1),...,d(111)\}$ . Table 3a has to be taken if parameter Mode = 0 (which means that the speech encoder is in unvoiced mode), while table 3b has to be taken if parameter Mode = 1, 2 or 3 (which means that the speech encoder is in voiced mode).

# 3.2.1 Parity and tailing for a speech frame

a) Parity bits:

The most significant 22 class 1 bits d(73),d(74),...,d(94) are protected by three parity bits used for error detection. These bits are added to the 22 bits, according to a cyclic code using the generator polynomial:

$$g(D) = D^3 + D + 1$$

The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

$$d(73)D^{24} + d(74)D^{23} + ... + d(94)D^{3} + p(0)D^{2} + p(1)D + p(2)$$

where p(0), p(1), p(2) are the parity bits, when divided by g(D), yields a remainder equal to:

$$1 + D + D^2$$
.

b) Tail bits and reordering:

The information and parity bits of class 1 are reordered, defining 104 information + parity + tail bits of class 1,  $\{u(0),u(1),...,u(103)\}\$  defined by:

$$\begin{array}{ll} u(k) = d(k) & \text{for } k = 0,1,...,94 \\ \\ u(k) = p(k-95) & \text{for } k = 95,96,97 \\ \\ u(k) = 0 & \text{for } k = 98,99,...,103 \text{ (tail bits)} \end{array}$$

#### 3.2.2 Convolutional encoder

The class 1 bits are encoded with the punctured convolutional code defined by the mother polynomials:

$$G4 = 1 + D^{2} + D^{3} + D^{5} + D^{6}$$
 
$$G5 = 1 + D + D^{4} + D^{6}$$
 
$$G6 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}$$

and the puncturing matrices:

In the puncturing matrices, a 1 indicates no puncture and a 0 indicates a puncture.

The coded bits  $\{c(0),c(1),...,c(227)\}$  are then defined by:

class 1 information bits:

$$c(2k) = u(k)+u(k-2)+u(k-3)+(k-5)+u(k-6)$$

$$c(2k+1) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-4) + u(k-6) \qquad \text{for } k = 0,1,...,94; \\ u(k) = 0 \text{ for } k < 0 \\ \text{parity bits:} \\ c(3k-95) = u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6) \\ c(3k-94) = u(k) + u(k-1) + u(k-4) + u(k-6) \\ c(3k-93) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-4) + u(k-6) \qquad \text{for } k = 95,96,97 \\ \text{tail bits:} \\ c(2k+3) = u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6) \\ c(2k+4) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-4) + u(k-6) \qquad \text{for } k = 98,99,...,103 \\ \text{class 2 information bits:} \\ c(k+211) = d(k+95) \text{ for } k = 0,1,...,16$$

# 3.2.3 Interleaving

The coded bits are reordered and interleaved according to the following rule:

$$i(B,j)=c(n,k)$$
 for  $k=0,1,...,227$  
$$n=0,1,...,N,N+1,... \label{eq:balance}$$
 
$$B=B0+2n+b$$

The values of b and j in dependence of k are given by table 4.

The result of the interleaving is a distribution of the reordered 228 bits of a given data block, n = N, over 4 blocks using the even numbered bits of the first 2 blocks (B = B0+2N+0,1) and the odd numbered bits of the last 2 blocks (B = B0+2N+2,3). The reordered bits of the following data block, n = N+1, use the even numbered bits of the blocks B = B0+2N+2,3 (B = B0+2(N+1)+0,1) and the odd numbered bits of the blocks B = B0+2(N+1)+2,3. Continuing with the next data blocks shows that one block always carries 57 bits of data from one data block (n = N) and 57 bits from the next block (n = N+1), where the bits from the data block with the higher number always are the even numbered data bits, and those of the data block with the lower number are the odd numbered bits. The block of coded data is interleaved "block diagonal", where a new data block starts every  $2^{nd}$  block and is distributed over 4 blocks.

# 3.2.4 Mapping on a burst

The mapping is given by the rule:

$$e(B,j)=i(B,j)$$
 and  $e(B,59+j)=i(B,57+j)$  for  $j=0,1,...,56$  and 
$$e(B,57)=hl(B) \text{ and } e(B,58)=hu(B)$$

The two bits, labelled hl(B) and hu(B) on burst number B are flags used for indication of control channel signalling. For each TCH/HS block not stolen for signalling purposes:

```
hu(B) = 0 for the first 2 bursts (indicating status of the even numbered bits)

hl(B) = 0 for the last 2 bursts (indicating status of the odd numbered bits)
```

For the use of hl(B) and hu(B) when a speech frame is stolen for signalling purposes, see subclause 4.3.5.

# 3.3 Data channel at full rate, 12.0 kbit/s radio interface rate (9.6 kbit/s services (TCH/F9.6))

The definition of a 12.0 kbit/s radio interface rate data flow for data services is given in 3GPP TS 44.021.

#### 3.3.1 Interface with user unit

The user unit delivers to the encoder a bit stream organized in blocks of 60 information bits (data frames) every 5 ms. Four such blocks are dealt with together in the coding process {d(0),...,d(239)}. For non-transparent services those four blocks shall align with one 240-bit RLP frame.

#### 3.3.2 Block code

The block of 4 \* 60 information bits is not encoded, but only increased with 4 tail bits equal to 0 at the end of the block.

$$u(k) = d(k)$$
 for  $k = 0,1,...,239$   $u(k) = 0$  for  $k = 240,241,242,243$  (tail bits)

## 3.3.3 Convolutional encoder

This block of 244 bits  $\{u(0),...,u(243)\}$  is encoded with the ½ rate convolutional code defined by the following polynomials:

$$G0 = 1 + D^3 + D^4$$
  
 $G1 = 1 + D + D^3 + D^4$ 

resulting in 488 coded bits {C(0), C(1),..., C(487)} with

$$\begin{split} &C(2k) &= u(k) + u(k-3) + u(k-4) \\ &C(2k+1) &= u(k) + u(k-1) + u(k-3) + u(k-4) \quad \text{for } k = 0,1,...,243 \ ; \ u(k) = 0 \ \text{for } k < 0 \end{split}$$

The code is punctured in such a way that the following 32 coded bits:

```
\{C(11+15j) \text{ for } j = 0,1,...,31\} are not transmitted.
```

The result is a block of 456 coded bits,  $\{c(0),c(1),...,c(455)\}$ 

# 3.3.4 Interleaving

The coded bits are reordered and interleaved according to the following rule:

```
\begin{split} i(B,j) &= c(n,k) \, \text{for} \, \, k \, = 0,1,...,455 \\ n &= 0,1,...,N,N+1,... \\ B &= B_0 + 4n + (k \, \text{mod} \, \, 19) + (k \, \, \text{div} \, \, 114) \\ j &= (k \, \, \text{mod} \, \, 19) + 19 \, (k \, \, \text{mod} \, \, 6) \end{split}
```

The result of the interleaving is a distribution of the reordered 114 bit of a given data block, n = N, over 19 blocks, 6 bits equally distributed in each block, in a diagonal way over consecutive blocks.

Or in other words the interleaving is a distribution of the encoded, reordered 456 bits from four given input data blocks, which taken together give n = N, over 22 bursts, 6 bits equally distributed in the first and  $22^{nd}$  bursts, 12 bits distributed in the second and  $21^{st}$  bursts, 18 bits distributed in the third and  $20^{th}$  bursts and 24 bits distributed in the other 16 bursts.

The block of coded data is interleaved "diagonal", where a new block of coded data starts with every fourth burst and is distributed over 22 bursts.

# 3.3.5 Mapping on a Burst

The mapping is done as specified for TCH/FS in subclause 3.1.4. On bitstealing by a FACCH, see subclause 4.2.5.

# 3.4 Data channel at full rate, 6.0 kbit/s radio interface rate (4.8 kbit/s services (TCH/F4.8))

The definition of a 6.0 kbit/s radio interface rate data flow for data services is given in 3GPP TS 44.021.

#### 3.4.1 Interface with user unit

The user unit delivers to the encoder a bit stream organized in blocks of 60 information bits (data frames) every 10 ms,  $\{d(0),d(1),...,d(59)\}$ .

In the case where the user unit delivers to the encoder a bit stream organized in blocks of 240 information bits every 40 ms (e.g. RLP frames), the bits  $\{d(0),d(1),...,d(59),d(60),...,d(60+59),d(2*60),...,d(2*60+59),d(3*60),...,d(3*60+59)\}$  shall be treated as four blocks of 60 bits each as described in the remainder of this clause. To ensure end-to-end synchronization of the 240 bit blocks, the resulting block after coding of the first 120 bits  $\{d(0),d(1),...,d(60+59)\}$  shall be transmitted in one of the transmission blocks B0, B2, B4 of the channel mapping defined in 3GPP TS 45.002.

#### 3.4.2 Block code

Sixteen bits equal to 0 are added to the 60 information bits, the result being a block of 76 bits,  $\{u(0),u(1),...,u(75)\}$ , with:

$$u(19k+p)=d(15k+p)$$
 for  $k=0,1,2,3$  and  $p=0,1,...,14$ ;  $u(19k+p)=0$  for  $k=0,1,2,3$  and  $p=15,16,17,18$ .

Two such blocks forming a block of 152 bits  $\{u'(0), u'(1), ..., u'(151)\}$  are dealt with together in the rest of the coding process:

$$u'(k) = u1(k), k = 0,1,...,75 (u1 = 1st block)$$
  
 $u'(k+76) = u2(k), k = 0,1,...,75 (u2 = 2nd block)$ 

#### 3.4.3 Convolutional encoder

This block of 152 bits is encoded with the convolutional code of rate 1/3 defined by the following polynomials:

$$G1 = 1 + D + D^3 + D^4$$
  
 $G2 = 1 + D^2 + D^4$   
 $G3 = 1 + D + D^2 + D^3 + D^4$ 

The result is a block of 3 \* 152 = 456 coded bits,  $\{c(0),c(1),...,c(455)\}$ :

$$\begin{split} c(3k) &= u'(k) + u'(k-1) + u'(k-3) + u'(k-4) \\ c(3k+1) &= u'(k) + u'(k-2) + u'(k-4) \\ c(3k+2) &= u'(k) + u'(k-1) + u'(k-2) + u'(k-3) + u'(k-4) \quad \text{for } k = 0,1,...,151; \\ u'(k) &= 0 \text{ for } k < 0 \end{split}$$

# 3.4.4 Interleaving

The interleaving is done as specified for the TCH/F9.6 in subclause 3.3.4.

# 3.4.5 Mapping on a Burst

The mapping is done as specified for the TCH/FS in subclause 3.1.4. On bitstealing for signalling purposes by a FACCH, see subclause 4.2.5.

# 3.5 Data channel at half rate, 6.0 kbit/s radio interface rate (4.8 kbit/s services (TCH/H4.8))

The definition of a 6.0 kbit/s radio interface rate data flow for data services is given in 3GPP TS 44.021.

#### 3.5.1 Interface with user unit

The user unit delivers to the encoder a bit stream organized in blocks of 60 information bits (data frames) every 10 ms. Four such blocks are dealt with together in the coding process,  $\{d(0),d(1),...,d(239)\}$ .

For non-transparent services those four blocks shall align with one complete 240-bit RLP frame.

#### 3.5.2 Block code

The block encoding is done as specified for the TCH/F9.6 in subclause 3.3.2.

#### 3.5.3 Convolutional encoder

The convolutional encoding is done as specified for the TCH/F9.6 in subclause 3.3.3.

## 3.5.4 Interleaving

The interleaving is done as specified for the TCH/F9.6 in subclause 3.3.4.

## 3.5.5 Mapping on a Burst

The mapping is done as specified for the TCH/FS in subclause 3.1.4. On bitstealing for signalling purposes by a FACCH, see subclause 4.3.5.

# 3.6 Data channel at full rate, 3.6 kbit/s radio interface rate (2.4 kbit/s and less services (TCH/F2.4))

The definition of a 3.6 kbit/s radio interface rate data flow for data services is given in 3GPP TS 44.021.

#### 3.6.1 Interface with user unit

The user unit delivers to the encoder a bit stream organized in blocks of 36 information bits (data frames) every 10 ms. Two such blocks are dealt with together in the coding process,  $\{d(0),d(1),...,d(71)\}$ .

### 3.6.2 Block code

This block of 72 information bits is not encoded, but only increased with four tail bits equal to 0 at the end of the block.

$$u(k) = d(k), \quad k = 0,1,...,71$$
  
 $u(k) = 0 \quad , \quad k = 72,73,74,75 \text{ (tail bits);}$ 

## 3.6.3 Convolutional encoder

This block of 76 bits  $\{u(0),u(1),...,u(75)\}$  is encoded with the convolutional code of rate 1/6 defined by the following polynomials:

$$G1 = 1 + D + D^3 + D^4$$
  
 $G2 = 1 + D^2 + D^4$ 

$$G3 = 1 + D + D^{2} + D^{3} + D^{4}$$

$$G1 = 1 + D + D^{3} + D^{4}$$

$$G2 = 1 + D^{2} + D^{4}$$

$$G3 = 1 + D + D^{2} + D^{3} + D^{4}$$

The result is a block of 456 coded bits:

 $\{c(0), c(1),...,c(455)\}, \text{ defined by}$  c(6k) = c(6k+3) = u(k) + u(k-1) + u(k-3) + u(k-4) c(6k+1) = c(6k+4) = u(k) + u(k-2) + u(k-4)  $c(6k+2) = c(6k+5) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-4), \text{for} \qquad k = 0,1,...,75;$  u(k) = 0 for k < 0

# 3.6.4 Interleaving

The interleaving is done as specified for the TCH/FS in subclause 3.1.3.

# 3.6.5 Mapping on a Burst

The mapping is done as specified for the TCH/FS in subclause 3.1.4.

# 3.7 Data channel at half rate, 3.6 kbit/s radio interface rate (2.4 kbit/s and less services (TCH/H2.4))

The definition of a 3.6 kbit/s radio interface rate data flow for data services is given in 3GPP TS 44.021.

#### 3.7.1 Interface with user unit

The user unit delivers to the encoder a bit stream organized in blocks of 36 information bits (data frames) every 10 ms. Two such blocks are dealt with together in the coding process,  $\{d(0),d(1),...,d(71)\}$ .

#### 3.7.2 Block code

The block of 72 information bits is not encoded, but only increased with 4 tail bits equal to 0, at the end of the block.

Two such blocks forming a block of 152 bits  $\{u(0),u(1),...,u(151)\}$  are dealt with together in the rest of the coding process.

$$\begin{array}{ll} u(k) & = d1(k), & k = 0,1,...,75 \; (d1 = 1^{st} \; information \; block) \\ \\ u(k+76) & = d2(k), & k = 0,1,...,75 \; (d2 = 2^{nd} \; information \; block) \\ \\ u(k) & = 0, & k = 72,73,74,75,148,149,150,151 \; (tail \; bits) \end{array}$$

#### 3.7.3 Convolutional encoder

The convolutional encoding is done as specified for the TCH/F4.8 in subclause 3.4.3.

# 3.7.4 Interleaving

The interleaving is done as specified for the TCH/F9.6 in subclause 3.3.4.

## 3.7.5 Mapping on a Burst

The mapping is done as specified for the TCH/FS in subclause 3.1.4. On bit stealing for signalling purposes by a FACCH, see subclause 4.3.5.

# 3.8 Data channel at full rate, 14.5 kbit/s radio interface rate (14.4 kbit/s services (TCH/F14.4))

The definition of a 14.5 kbit/s radio interface rate data flow for data services is given in 3GPP TS 44.021.

### 3.8.1 Interface with user unit

The user unit delivers to the encoder a bit stream organized in blocks of 290 information bits (data frames) every 20 ms.

#### 3.8.2 Block code

The block of 290 information bits is not encoded, but only increased with 4 tail bits equal to 0 at the end of the block.

$$u(k) = d(k)$$
 for  $k = 0,1,...,289$  
$$u(k) = 0$$
 for  $k = 290,291,292,293$  (tail bits)

#### 3.8.3 Convolutional encoder

This block of 294 bits  $\{u(0),...,u(293)\}$  is encoded with the ½ rate convolutional code defined by the following polynomials:

$$G0 = 1 + D^3 + D^4$$
  
 $G1 = 1 + D + D^3 + D^4$ 

resulting in 588 coded bits {C(0), C(1),..., C(587)} with

$$\begin{split} &C(2k) = u(k) + u(k-3) + u(k-4) \\ &C(2k+1) = u(k) + u(k-1) + u(k-3) + u(k-4) \text{ for } k = 0,1,...,293 \text{ ; } u(k) = 0 \text{ for } k < 0 \end{split}$$

The code is punctured in such a way that the following 132 coded bits:

 $\{C(18*j+1), C(18*j+6), C(18*j+11), C(18*j+15) \text{ for } j=0,1,...,31\}$  and the bits C(577), C(582), C(584) and C(587) are not transmitted.

The result is a block of 456 coded bits,  $\{c(0),c(1),...,c(455)\}$ 

# 3.8.4 Interleaving

The interleaving is done as specified for the TCH/F9.6 in section 3.3.4.

# 3.8.5 Mapping on a Burst

The mapping is done as specified for TCH/FS in section 3.1.4. On bitstealing by a FACCH, see section 4.2.5.

# 3.9 Adaptive multi rate speech channel at full rate (TCH/AFS)

This section describes the coding for the different frame formats used for TCH/AFS. The formats used are (in the order they are described):

SID\_UPDATE Used to convey comfort noise parameters during DTX

SID\_FIRST Marker to define end of speech, start of DTX

ONSET Used to signal the Codec mode for the first speech frame after DTX

SPEECH Speech frames

RATSCCH Frames used to convey RATSCCH messages

In this chapter, sub chapters 3.9.1 to 3.9.5 describe the channel coding for the different formats listed above.

Common to all the formats is that in-band information is conveyed, the coding for the in-band channel is described in the table below.

Identifier (defined in 3GPP TS 45.009)	Received in-band data id(1), id(0)	Encoded in-band data for SID and RATSCCH frames ic(15),, ic(0)	Encoded in-band data for speech frames ic(7),, ic(0)
CODEC_MODE_1	00	0101001100001111	00000000
CODEC_MODE_2	01	0011111010111000	10111010
CODEC_MODE_3	10	1000100001100011	01011101
CODEC_MODE_4	11	1110010111010100	11100111

# 3.9.1 SID\_UPDATE

The speech encoder delivers 35 bits of comfort noise parameters. Also delivered is two in-band channels, id0(0,1) and id1(0,1), id0 corresponding to Mode Commands or Mode Requests and id1 to Mode Indication. The general coding is as: the two in-band data channels are coded to 16 bits each, a 14-bit CRC is added to the 35 CN bits which are then coded by a rate  $\frac{1}{4}$  RSC coder to 212 bits. Finally a 212 bit identification field is added thereby giving a total size of 456 bits. These 456 bits are then block interleaved in the same way as SACCH frames.

#### 3.9.1.1 Coding of in-band data

The two in-band data fields, id0(0,1) and id1(0,1), are encoded, giving ic0(0..15) and ic1(0..15).

The ic0 and ic1 data is moved to the coded data c as:

$$c(k) = ic0(k) \qquad \qquad \text{for } k = 0,1,2,3$$
 
$$c(k) = ic1(k-4) \qquad \qquad \text{for } k = 4, 5, 6, 7$$
 
$$c(k) = ic0(k-4) \qquad \qquad \text{for } k = 8, 9, 10, 11$$
 
$$c(k) = ic1(k-8) \qquad \qquad \text{for } k = 12, 13, 14, 15$$
 
$$c(k) = ic0(k-8) \qquad \qquad \text{for } k = 16, 17, 18, 19$$
 
$$c(k) = ic1(k-12) \qquad \qquad \text{for } k = 20, 21, 22, 23$$
 
$$c(k) = ic0(k-12) \qquad \qquad \text{for } k = 24, 25, 26, 27$$
 
$$c(k) = ic1(k-16) \qquad \qquad \text{for } k = 28, 29, 30, 31$$

#### 3.9.1.2 Parity and convolutional encoding for the comfort noise parameters

#### a) Parity bits:

A 14-bit CRC is used for error-detection. These 14 parity bits are generated by the cyclic generator polynomial:  $g(D) = D^{14} + D^{13} + D^5 + D^3 + D^2 + I$  from the 35 comfort noise parameter bits. The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

$$d(0)D(48) + d(1)D(47) + ... + d(34)D(14) + p(0)D^{\left(13\right)} + ... + p(12)D + p(13)$$

where p(0), p(1) ... p(13) are the parity bits, when divided by g(D), yields a remainder equal to  $1 + D + D^2 + D^3 + D^4 + D^5 + D^6 + D^7 + D^8 + D^9 + D^{10} + D^{11} + D^{12} + D^{13}$ 

The information and parity bits are merged:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 34$ 

$$u(k) = p(k-35)$$
 for  $k = 35, 36, ..., 48$ 

#### b) Convolutional encoder

The comfort noise parameters with parity bits (u(0..48)) are encoded with the  $\frac{1}{4}$  rate

convolutional code defined by the polynomials:

$$G1/G3 = 1 + D + D^{3} + D^{4} / 1 + D + D^{2} + D^{3} + D^{4}$$

$$G2/G3 = 1 + D^{2} + D^{4} / 1 + D + D^{2} + D^{3} + D^{4}$$

$$G3/G3 = 1$$

$$G3/G3 = 1$$

resulting in 212 coded bits,  $\{C(0)...C(211)\}\$  defined by:

$$\begin{split} r(k) &= u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) \\ C(4k) &= r(k) + r(k-1) + r(k-3) + r(k-4) \\ C(4k+1) &= r(k) + r(k-2) + r(k-4) \\ C(4k+2) &= u(k) \\ C(4k+3) &= u(k) \qquad \qquad \text{for } k = 0, 1, ..., 48; \ r(k) = 0 \ \text{for } k < 0 \end{split}$$

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(4k) &= r(k) + r(k-1) + r(k-3) + r(k-4) \\ C(4k+1) &= r(k) + r(k-2) + r(k-4) \\ C(4k+2) &= r(k-1) + r(k-2) + r(k-3) + r(k-4) \\ C(4k+3) &= r(k-1) + r(k-2) + r(k-3) + r(k-4) & \text{for } k = 49, 50, ..., 52 \end{split}$$

This block of data is moved to the coded data (c) as:

$$c(8*k+32) = C(4*k)$$
  
 $c(8*k+33) = C(4*k+1)$   
 $c(8*k+34) = C(4*k+2)$   
 $c(8*k+35) = C(4*k+3)$  for  $k = 0, 1, ..., 52$ 

#### 3.9.1.3 Identification marker

The identification marker, IM(0..211), is constructed by repeating the following 9-bit sequence: { 0, 1, 0, 0, 1, 1, 1, 1, 0 } 24 times and then discarding the last 4 bits. This block of data is moved to the coded data (c) as:

$$c(8*k+36) = IM(4*k)$$
  
 $c(8*k+37) = IM(4*k+1)$   
 $c(8*k+38) = IM(4*k+2)$ 

$$c(8*k+39) = IM(4*k+3)$$
 for  $k = 0, 1, ..., 52$ 

#### 3.9.1.4 Interleaving

The interleaving is done as specified for the SACCH in subclause 4.1.4.

#### 3.9.1.5 Mapping on a Burst

The interleaving is done as specified for the SACCH in subclause 4.1.5 with the exception that hl(B) and hu(B) is set to "0".

## 3.9.2 SID\_FIRST

This frame type contains no source data from the speech coder, what is transmitted is the in-band channel (signalling Mode Indication or Mode Command/Mode Request depending on the current frame number) and an identification marker.

### 3.9.2.1 Coding of in-band data

The in-band data, id(0,1), is encoded to ic(0..15) which is moved to the coded data c as:

$$c(k) = ic(k)$$
 for  $k = 0,1,2,3$   
 $c(k) = ic(k-4)$  for  $k = 8, 9, 10, 11$   
 $c(k) = ic(k-8)$  for  $k = 16, 17, 18, 19$   
 $c(k) = ic(k-12)$  for  $k = 24, 25, 26, 27$ 

#### 3.9.2.2 Identification marker

The identification marker, IM(0..211), is constructed by repeating the following 9-bit sequence: { 0, 1, 0, 0, 1, 1, 1, 1, 0 } 24 times and then discarding the last 4 bits. This block of data is moved to the coded data (c) as:

```
c(8*k+32) = IM(4*k)

c(8*k+33) = IM(4*k+1)

c(8*k+34) = IM(4*k+2)

c(8*k+35) = IM(4*k+3) for k = 0, 1, ..., 52
```

#### 3.9.2.3 Interleaving

The interleaving is done as specified for the TCH/FS in subclause 3.1.3.

#### 3.9.2.4 Mapping on a Burst

The mapping is done as specified for the TCH/FS in subclause 3.1.4. The last 4 bursts shall not be transmitted unless the SID\_FIRST frame is immediately followed by a speech frame.

## 3.9.3 ONSET

Onset frames are used to preset the interleaver buffer after a period of no speech activity in DTX mode. This frame type contains no source data from the speech coder, what is transmitted is the in-band channel signalling the Mode Indication for the speech frame following the onset marker.

#### 3.9.3.1 Coding of in-band data

The in-band data, Mode Indication id1(0,1), is encoded to ic1(0..15). This sequence is then repeated 14 times more, and the last 12 bits are discarded (15\*16-12=228) giving the sequence ic1(0..227).

This sequence is then moved to c as:

$$c(8*k+4) = ic1(4*k)$$

$$c(8*k+5) = ic1(4*k+1)$$

$$c(8*k+6) = ic1(4*k+2)$$

$$c(8*k+7) = ic1(4*k+3)$$
 for k = 0, 1, ..., 56

#### 3.9.3.2 Interleaving

The coded bits are reordered and interleaved according to the following rule:

$$i(B,j) = c(n,k), \quad \text{for } k = 4,5,6,7,\ 12,13,14,15,20,21,22,23\ ...,455$$
 
$$n = 0,1,...,N,N+1,...$$
 
$$B = B_0 + 4n + (k\ mod\ 8) - 4$$
 
$$j = 2((49k)\ mod\ 57) + ((k\ mod\ 8)\ div\ 4)$$

See table 1. The result of the interleaving is a distribution of the defined 228 bits of a given data block of size 456 bits, n = N, over 4 blocks using the odd numbered bits. The even numbered bits of these 4 blocks will be filled by the speech frame for which this frame is the ONSET.

## 3.9.3.3 Mapping on a Burst

The mapping is given by the rule:

$$e(B,j) = i(B,j)$$
 and  $e(B,59+j) = i(B,57+j)$  for  $j = 0,1,...,56$ 

and

$$e(B,57) = hl(B)$$

The bit labelled hl(B) on burst number B is a flag used for indication of control channel signalling. For each ONSET block not stolen for signalling purposes:

```
hl(B) = 0 for the 4 bursts (indicating status of odd numbered bits)
```

For the use of hl(B) when an ONSET is stolen for signalling purposes see subclause 4.2.5.

#### 3.9.4 SPEECH

The speech coder delivers to the channel encoder a sequence of blocks of data. One block of data corresponds to one speech frame and the block length is different in each of the eight channel codec modes. Adjoining each block of data is information of the channel codec mode to use when encoding the block. Also delivered is the in-band data id(0,1) representing Mode Indication or Mode Command/Mode Request depending on the current frame number.

#### 3.9.4.1 Coding of the in-band data

The two input in-band bits (id(0,1)) are coded to eight coded in-band bits (ic(0..7)).

The encoded in-band bits are moved to the coded bits, c, as

$$c(k) = ic(k)$$
 for  $k = 0, 1, ..., 7$ .

## 3.9.4.2 Ordering according to subjective importance

The bits delivered by the speech encoder,  $\{s(1), s(2), ..., s(K_s)\}$ , are rearranged according to subjective importance before channel coding. Tables 7 to 16 define the correct rearrangement for the speech codec modes 12.2 kbit/s, 10.2 kbit/s, 7.95 kbit/s, 7.40 kbit/s, 6.70 kbit/s, 5.90 kbit/s, 5.15 kbit/s and 4.75 kbit/s, respectively. In the tables speech codec parameters are numbered in the order they are delivered by the corresponding speech encoder according to 3GPP TS 26.090 and the rearranged bits are labelled  $\{d(0),d(1),...,d(K_d-1)\}$ , defined in the order of decreasing importance. Index  $K_d$  refers to the number of bits delivered by the speech encoder, see below:

Codec mode	Number of speech bits delivered per block (K <sub>d</sub> )
TCH/AFS12.2	244
TCH/AFS10.2	204
TCH/AFS7.95	159
TCH/AFS7.4	148
TCH/AFS6.7	134
TCH/AFS5.9	118
TCH/AFS5.15	103
TCH/AFS4.75	95

The ordering algorithm is in pseudo code as:

for 
$$j = 0$$
 to  $K_{d-1}$   $d(j) := s(table(j)+1);$  where  $table(j)$  is read line by line left to right

The rearranged bits are further divided into two different classes to perform unequal error protection for different bits according to subjective importance.

The protection classes are:

- 1a Data protected with the CRC and the convolution code.
- 1b Data protected with the convolution code.

No unprotected bits are used.

The number of class 1 (sum of class 1a and 1b), class 1a and class 1b bits for each codec mode is shown below:

Codec Mode	Number of speech bits delivered per block	Number of class 1 bits per block	Number of class 1a bits per block	Number of class 1b bits per block
TCH/AFS12.2	244	244	81	163
TCH/AFS10.2	204	204	65	139
TCH/AFS7.95	159	159	75	84
TCH/AFS7.4	148	148	61	87
TCH/AFS6.7	134	134	55	79
TCH/AFS5.9	118	118	55	63
TCH/AFS5.15	103	103	49	54
TCH/AFS4.75	95	95	39	56

## 3.9.4.3 Parity for speech frames

The basic parameters for each codec mode for the first encoding step are shown below:

Codec mode	Speech encoded bits (K <sub>d</sub> )	CRC protected bits (K <sub>d1a</sub> )	Number of bits after first encoding step (K <sub>u</sub> = K <sub>d</sub> + 6)
TCH/AFS12.2	244	81	250
TCH/AFS10.2	204	65	210
TCH/AFS7.95	159	75	165
TCH/AFS7.4	148	61	154
TCH/AFS6.7	134	55	140
TCH/AFS5.9	118	55	124
TCH/AFS5.15	103	49	109
TCH/AFS4.75	95	39	101

A 6-bit CRC is used for error-detection. These 6 parity bits are generated by the cyclic generator polynomial:  $g(D) = D^6 + D^5 + D^3 + D^2 + D^1 + I$  from the first  $K_{d1a}$  bits of class 1, where  $K_{d1a}$  refers to number of bits in protection class 1a as shown above for each codec mode. The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

$$d(0)D(K_{d1a}+5) + d(1)D(K_{d1a}+4) + ... + d(K_{d1a}-1)D(6) + p(0)D(5) + ... + p(4)D + p(5)$$

where p(0), p(1) ... p(5) are the parity bits, when divided by g(D), yields a remainder equal to:

$$1 + D + D^2 + D^3 + D^4 + D^5$$
.

The information and parity bits are merged:

$$\begin{array}{ll} u(k)=d(k) & \text{ for } k=0,\,1,\,...,\,K_{d1a}\text{-}1 \\ \\ u(k)=p(k\text{-}K_{d1a}) & \text{ for } k=K_{d1a},\,K_{d1a}\text{+}1,\,...,\,K_{d1a}\text{+}5 \\ \\ u(k)=d(k\text{-}6) & \text{ for } k=K_{d1a}\text{+}6,\,K_{d1a}\text{+}7,\,...,\,K_{u}\text{-}1 \end{array}$$

Thus, after the first encoding step u(k) will be defined by the following contents for each codec mode:

#### TCH/AFS12.2:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 80$   $u(k) = p(k-81)$  for  $k = 81, 82, ..., 86$   $u(k) = d(k-6)$  for  $k = 87, 88, ..., 249$ 

#### TCH/AFS10.2:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 64$   $u(k) = p(k-65)$  for  $k = 65, 66, ..., 70$   $u(k) = d(k-6)$  for  $k = 71, 72, ..., 209$ 

#### **TCH/AFS7.95**:

$$\begin{array}{ll} u(k) = d(k) & \text{ for } k = 0,\,1,\,...,\,74 \\ \\ u(k) = p(k\text{-}75) & \text{ for } k = 75,\,76,\,...,\,80 \\ \\ u(k) = d(k\text{-}6) & \text{ for } k = 81,\,82,\,...,\,164 \end{array}$$

#### TCH/AFS7.4:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 60$   
 $u(k) = p(k-61)$  for  $k = 61, 62, ..., 66$   
 $u(k) = d(k-6)$  for  $k = 67, 68, ..., 153$ 

#### TCH/AFS6.7:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 54$ 

$$u(k) = p(k-55)$$
 for  $k = 55, 56, ..., 60$ 

$$u(k) = d(k-6)$$
 for  $k = 61, 62, ..., 139$ 

#### TCH/AFS5.9:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 54$ 

$$u(k) = p(k-55)$$
 for  $k = 55, 56, ..., 60$ 

$$u(k) = d(k-6)$$
 for  $k = 61, 62, ..., 123$ 

#### **TCH/AFS5.15**:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 48$ 

$$u(k) = p(k-49)$$
 for  $k = 49, 50, ..., 54$ 

$$u(k) = d(k-6)$$
 for  $k = 55, 56, ..., 108$ 

#### **TCH/AFS4.75**:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 38$ 

$$u(k) = p(k-39)$$
 for  $k = 39, 40, ..., 44$ 

$$u(k) = d(k-6)$$
 for  $k = 45, 46, ..., 100$ 

#### 3.9.4.4 Convolutional encoder

The bits from the first encoding step (u(k)) are encoded with the recursive systematic convolutional codes as summarised below. The number of output bits after puncturing is 448 for all codec modes.

Codec mode	Rate	Number of input bits to conv. coder	Number of output bits from conv. coder	Number of punctured bits
TCH/AFS12.2	1/2	250	508	60
TCH/AFS10.2	1/3	210	642	194
TCH/AFS7.95	1/3	165	513	65
TCH/AFS7.4	1/3	154	474	26
TCH/AFS6.7	1/4	140	576	128
TCH/AFS5.9	1/4	124	520	72
TCH/AFS5.15	1/5	109	565	117
TCH/AFS4.75	1/5	101	535	87

Below the coding for each codec mode is specified in detail.

#### **TCH/AFS12.2**:

The block of 250 bits  $\{u(0)...u(249)\}$  is encoded with the ½ rate convolutional code defined by polynomials:

$$G0/G0 = 1$$

$$G1/G0 = 1 + D + D^3 + D^4 / 1 + D^3 + D^4$$

resulting in 508 coded bits,  $\{C(0)...C(507)\}\$  defined by:

$$r(k) = u(k) + r(k-3) + r(k-4)$$

$$C(2k) = u(k)$$

$$C(2k+1) = r(k)+r(k-1)+r(k-3)+r(k-4)$$
 for  $k = 0, 1, ..., 249$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$r(k) = 0$$

$$C(2k) = r(k-3) + r(k-4)$$

$$C(2k+1) = r(k)+r(k-1)+r(k-3)+r(k-4)$$
 for  $k = 250, 251, ..., 253$ 

The code is punctured in such a way that the following 60 coded bits:

are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the in-band bits in c as

$$c(k+8) = P(k)$$
 for  $k = 0, 1, ..., 447$ .

#### TCH/AFS10.2:

The block of 210 bits  $\{u(0)...\ u(209)\}$  is encoded with the 1/3 rate convolutional code defined by the following polynomials:

$$G1/G3 = 1 + D + D^{3} + D^{4} / 1 + D + D^{2} + D^{3} + D^{4}$$
 
$$G2/G3 = 1 + D^{2} + D^{4} / 1 + D + D^{2} + D^{3} + D^{4}$$
 
$$G3/G3 = 1$$

resulting in 642 coded bits, {C(0)... C(641)} defined by:

$$r(k) = u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4)$$

$$C(3k) = r(k) + r(k-1) + r(k-3) + r(k-4)$$

$$C(3k+1) = r(k)+r(k-2)+r(k-4)$$

$$C(3k+2) = u(k)$$
 for  $k = 0, 1, ..., 209$ 

and (for termination of the coder):

$$r(k) = 0$$

$$C(3k) = r(k)+r(k-1) + r(k-3) + r(k-4)$$

$$C(3k+1) = r(k)+r(k-2)+r(k-4)$$

$$C(3k+2) = r(k-1)+r(k-2)+r(k-3)+r(k-4)$$
 for  $k = 210, 211, ..., 213$ 

The code is punctured in such a way that the following 194 bits:

```
C(1), C(4), C(7), C(10), C(16), C(19), C(22), C(28), C(31), C(34), C(40), C(43), C(46), C(52), C(55), C(58), C(64), C(67), C(70), C(76), C(79), C(82), C(88), C(91), C(94), C(100), C(103), C(106), C(112), C(115), C(118), C(124), C(127), C(130), C(136), C(139), C(142), C(148), C(151), C(154), C(160), C(163), C(166), C(172), C(175), C(178), C(184), C(187), C(190), C(196), C(199), C(202), C(208), C(211), C(214), C(220), C(223), C(226), C(232), C(235), C(238), C(244), C(247), C(250), C(256), C(259), C(262), C(268), C(271), C(274), C(280), C(283), C(286), C(292), C(295), C(298), C(304), C(307), C(310), C(316), C(319), C(322), C(325), C(328), C(331), C(334), C(337), C(340), C(343), C(346), C(349), C(352), C(355), C(358), C(361), C(364), C(367), C(370), C(373), C(376), C(379), C(382), C(388), C(391), C(394), C(397), C(400), C(403), C(406), C(409), C(412), C(415), C(418), C(421), C(424), C(427), C(430), C(433), C(436), C(439), C(439), C(439), C(436), C(
```

C(442), C(445), C(448), C(451), C(454), C(457), C(460), C(463), C(466), C(469), C(472), C(475), C(478), C(481), C(484), C(487), C(490), C(493), C(496), C(499), C(502), C(505), C(508), C(511), C(514), C(517), C(520), C(523), C(526), C(529), C(532), C(535), C(538), C(541), C(544), C(547), C(550), C(553), C(556), C(559), C(562), C(565), C(568), C(571), C(574), C(577), C(580), C(583), C(586), C(589), C(592), C(595), C(598), C(601), C(604), C(607), C(609), C(610), C(613), C(616), C(619), C(621), C(622), C(625), C(627), C(628), C(631), C(633), C(634), C(636), C(637), C(639) and C(640)

are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the in-band bits in c as:

$$c(k+8) = P(k)$$
 for  $k = 0, 1, ..., 447$ .

#### **TCH/AFS7.95**:

The block of 165 bits  $\{u(0)...u(164)\}$  is encoded with the 1/3 rate convolutional code defined by the following polynomials:

$$G4/G4 = 1$$

$$G5/G4 = 1 + D + D^4 + D^6/1 + D^2 + D^3 + D^5 + D^6$$

$$G6/G4 = 1 + D + D^2 + D^3 + D^4 + D^6/1 + D^2 + D^3 + D^5 + D^6$$

resulting in 513 coded bits,  $\{C(0)...C(512)\}\$  defined by:

$$r(k) = u(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(3k) = u(k)$$

$$C(3k+1) = r(k)+r(k-1)+r(k-4)+r(k-6)$$

$$C(3k+2) = r(k)+r(k-1)+r(k-2)+r(k-3)+r(k-4)+r(k-6)$$
 for  $k = 0, 1, ..., 164$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$r(k) = 0$$

$$C(3k) = r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(3k+1) = r(k)+r(k-1)+r(k-4)+r(k-6)$$

$$C(3k+2) = r(k)+r(k-1)+r(k-2)+r(k-3)+r(k-4)+r(k-6)$$
 for  $k = 165, 166, ..., 170$ 

The code is punctured in such a way that the following 65 coded bits:

```
C(1), C(2), C(4), C(5), C(8), C(22), C(70), C(118), C(166), C(214), C(262), C(310), C(317), C(319), C(325), C(332), C(334), C(341), C(343), C(349), C(356), C(358), C(365), C(367), C(373), C(380), C(382), C(385), C(389), C(391), C(397), C(404), C(406), C(409), C(413), C(415), C(421), C(428), C(430), C(433), C(437), C(439), C(445), C(452), C(454), C(457), C(461), C(463), C(469), C(476), C(478), C(481), C(485), C(487), C(490), C(493), C(500), C(502), C(503), C(505), C(506), C(508), C(509), C(511) and C(512)
```

are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the in-band bits in c as

$$c(k+8) = P(k)$$
 for  $k = 0, 1, ..., 447$ .

#### TCH/AFS7.4:

The block of 154 bits  $\{u(0)...\ u(153)\}$  is encoded with the 1/3 rate convolutional code defined by the following polynomials:

$$G1/G3 = 1 + D + D^{3} + D^{4} / 1 + D + D^{2} + D^{3} + D^{4}$$

$$G2/G3 = 1 + D^{2} + D^{4} / 1 + D + D^{2} + D^{3} + D^{4}$$

$$G3/G3 = 1$$

resulting in 474 coded bits, {C(0)... C(473)} defined by:

$$r(k) = u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4)$$

$$C(3k) = r(k) + r(k-1) + r(k-3) + r(k-4)$$

$$C(3k+1) = r(k)+r(k-2)+r(k-4)$$

$$C(3k+2) = u(k)$$
 for  $k = 0, 1, ..., 153$ 

and (for termination of the coder):

$$r(k) = 0$$

$$C(3k) = r(k)+r(k-1) + r(k-3) + r(k-4)$$

$$C(3k+1) = r(k)+r(k-2)+r(k-4)$$

$$C(3k+2) = r(k-1)+r(k-2)+r(k-3)+r(k-4)$$
 for  $k = 154, 155, ..., 157$ 

The code is punctured in such a way that the following 26 bits:

are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the in-band bits in c as:

$$c(k+8) = P(k)$$
 for  $k = 0, 1, ..., 447$ .

#### TCH/AFS6.7:

The block of 140 bits  $\{u(0)...u(139)\}$  is encoded with the  $\frac{1}{4}$  rate convolutional code defined by polynomials:

$$G1/G3 = 1 + D + D^3 + D^4 / 1 + D + D^2 + D^3 + D^4$$

$$G2/G3 = 1 + D^2 + D^4 / 1 + D + D^2 + D^3 + D^4$$

$$G3/G3 = 1$$

$$G3/G3 = 1$$

resulting in 576 coded bits,  $\{C(0)...C(575)\}\$  defined by:

$$r(k) = u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4)$$

$$C(4k)$$
 =  $r(k) + r(k-1) + r(k-3) + r(k-4)$ 

$$C(4k+1) = r(k)+r(k-2)+r(k-4)$$

$$C(4k+2) = u(k)$$

$$C(4k+3) = u(k)$$
 for  $k = 0, 1, ..., 139$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$r(k) = 0$$

$$C(4k) = r(k)+r(k-1) + r(k-3) + r(k-4)$$

$$C(4k+1) = r(k)+r(k-2)+r(k-4)$$

$$C(4k+2) = r(k-1)+r(k-2)+r(k-3)+r(k-4)$$

$$C(4k+3) = r(k-1)+r(k-2)+r(k-3)+r(k-4)$$
 for  $k = 140, 141, ..., 143$ 

The code is punctured in such a way that the following 128 coded bits:

C(1), C(3), C(7), C(11), C(15), C(27), C(39), C(55), C(67), C(79), C(95), C(107), C(119), C(135), C(147), C(159), C(175), C(187), C(199), C(215), C(227), C(239), C(255), C(267), C(279), C(287), C(291), C(295), C(299), C(303), C(307), C(311), C(315), C(319), C(323), C(327), C(331), C(335), C(339), C(343), C(347), C(351), C(355), C(359), C(363), C(367), C(369), C(371), C(375), C(377), C(379), C(383), C(385), C(387), C(391), C(393), C(395), C(399), C(401), C(403), C(407), C(409), C(411), C(415), C(417), C(419), C(423), C(425), C(427), C(431), C(433), C(435), C(439), C(441), C(443), C(447), C(449), C(451), C(455), C(457), C(459), C(463), C(465), C(467), C(471), C(473), C(475), C(479), C(481), C(483), C(487), C(489), C(491), C(495), C(497), C(499), C(503), C(505), C(507), C(511), C(513), C(515), C(519), C(521), C(523), C(527), C(529), C(531), C(563), C(565), C(567), C(569), C(571), C(573) and C(575)

are not transmitted. The result is a block of 448 coded bits, P(0)...P(447) which are appended to the in-band bits in c as

$$c(k+8) = P(k)$$
 for  $k = 0, 1, ..., 447$ .

#### TCH/AFS5.9:

The block of 124 bits  $\{u(0)...u(123)\}$  is encoded with the  $\frac{1}{4}$  rate convolutional code defined by polynomials:

$$G4/G6 = 1 + D^{2} + D^{3} + D^{5} + D^{6} / 1 + D + D^{2} + D^{3} + D^{4} + D^{6}$$

$$G5/G6 = 1 + D + D^{4} + D^{6} / 1 + D + D^{2} + D^{3} + D^{4} + D^{6}$$

$$G6/G6 = 1$$

$$G6/G6 = 1$$

resulting in 520 coded bits,  $\{C(0)...C(519)\}\$  defined by:

and (for termination of the coder):

The code is punctured in such a way that the following 72 coded bits:

```
C(0), C(1), C(3), C(5), C(7), C(11), C(15), C(31), C(47), C(63), C(79), C(95), C(111), C(127), C(143), C(159), C(175), C(191), C(207), C(223), C(239), C(255), C(271), C(287), C(303), C(319), C(327), C(331), C(335), C(343), C(347), C(351), C(359), C(363), C(367), C(375), C(379), C(383), C(391), C(395), C(399), C(407), C(411), C(415), C(423), C(427), C(431), C(439), C(443), C(447), C(455), C(459), C(463), C(467), C(471), C(475), C(479), C(483), C(487), C(491), C(495), C(499), C(503), C(507), C(509), C(511), C(512), C(513), C(515), C(516), C(517) and C(519)
```

are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the in-band bits in c as

$$c(8+k) = P(k)$$
 for  $k = 0, 1, ..., 447$ .

#### **TCH/AFS5.15**:

The block of 109 bits  $\{u(0)...u(108)\}$  is encoded with the 1/5 rate convolutional code defined by the following polynomials:

$$\begin{split} G1/G3 &= 1 + D + D^3 + D^4 / 1 + D + D^2 + D^3 + D^4 \\ G1/G3 &= 1 + D + D^3 + D^4 / 1 + D + D^2 + D^3 + D^4 \\ G2/G3 &= 1 + D^2 + D^4 / 1 + D + D^2 + D^3 + D^4 \\ G3/G3 &= 1 \\ G3/G3 &= 1 \end{split}$$

resulting in 565 coded bits,  $\{C(0)...C(564)\}\$  defined by:

$$\begin{array}{ll} r(k) &= u(k) + r(k\text{-}1) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}4) \\ \\ C(5k) &= r(k) + r(k\text{-}1) + r(k\text{-}3) + r(k\text{-}4) \\ \\ C(5k+1) &= r(k) + r(k\text{-}1) + r(k\text{-}3) + r(k\text{-}4) \\ \\ C(5k+2) &= r(k) + r(k\text{-}2) + r(k\text{-}4) \\ \\ C(5k+3) &= u(k) \\ \\ C(5k+4) &= u(k) \\ \\ \end{array}$$

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(5k) &= r(k) + r(k-1) + r(k-3) + r(k-4) \\ C(5k+1) &= r(k) + r(k-1) + r(k-3) + r(k-4) \\ C(5k+2) &= r(k) + r(k-2) + r(k-4) \\ C(5k+3) &= r(k-1) + r(k-2) + r(k-3) + r(k-4) \\ C(5k+4) &= r(k-1) + r(k-2) + r(k-3) + r(k-4) & \text{for } k = 109, 110, ..., 112 \end{split}$$

The code is punctured in such a way that the following 117 coded bits:

```
C(0), C(C(4), C(5), C(9), C(10), C(14), C(15), C(20), C(25), C(30), C(35), C(40), C(50), C(60), C(70), C(80), C(90), C(100), C(110), C(120), C(130), C(140), C(150), C(160), C(170), C(180), C(190), C(200), C(210), C(220), C(230), C(240), C(250), C(260), C(270), C(280), C(290), C(300), C(310), C(315), C(320), C(325), C(330), C(334), C(335), C(340), C(344), C(345), C(350), C(354), C(355), C(360), C(364), C(365), C(370), C(374), C(375), C(380), C(384), C(385), C(390), C(394), C(395), C(400), C(404), C(405), C(410), C(414), C(415), C(420), C(424), C(425), C(430), C(434), C(435), C(440), C(444), C(445), C(450), C(454), C(455), C(460), C(464), C(465), C(470), C(474), C(475), C(480), C(484), C(485), C(490), C(494), C(495), C(500), C(504), C(505), C(510), C(514), C(515), C(520), C(524), C(525), C(529), C(530), C(534), C(535), C(539), C(540), C(544), C(545), C(549), C(550), C(554), C(555), C(559), C(560) and C(564)
```

are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the in-band bits in c as

$$c(8+k) = P(k)$$
 for  $k = 0, 1, ..., 447$ .

#### **TCH/AFS4.75**:

The block of 101 bits  $\{u(0)...u(100)\}$  is encoded with the 1/5 rate convolutional code defined by the following polynomials:

$$G4/G6 = 1 + D^{2} + D^{3} + D^{5} + D^{6} / 1 + D + D^{2} + D^{3} + D^{4} + D^{6}$$

$$G4/G6 = 1 + D^{2} + D^{3} + D^{5} + D^{6} / 1 + D + D^{2} + D^{3} + D^{4} + D^{6}$$

$$G5/G6 = 1 + D + D^{4} + D^{6} / 1 + D + D^{2} + D^{3} + D^{4} + D^{6}$$

$$G6/G6 = 1$$

$$G6/G6 = 1$$

resulting in 535 coded bits,  $\{C(0)...C(534)\}\$  defined by:

$$r(k) = u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6)$$

$$C(5k) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(5k+1) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(5k+2) = r(k) + r(k-1) + r(k-4) + r(k-6)$$

$$C(5k+3) = u(k)$$

$$C(5k+4) = u(k)$$

for 
$$k = 0, 1, ..., 100$$
;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(5k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(5k+1) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(5k+2) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(5k+3) &= r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(5k+4) &= r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ \end{split}$$

The code is punctured in such a way that the following 87 coded bits:

for k = 101, 102, ..., 106

 $C(0), C(1), C(2), C(4), C(5), C(7), C(9), C(15), C(25), C(35), C(45), C(55), C(65), C(75), C(85), C(95), \\ C(105), C(115), C(125), C(135), C(145), C(155), C(165), C(175), C(185), C(195), C(205), C(215), C(225), \\ C(235), C(245), C(255), C(265), C(275), C(285), C(295), C(305), C(315), C(325), C(335), C(345), C(355), \\ C(365), C(375), C(385), C(395), C(400), C(405), C(410), C(415), C(420), C(425), C(430), C(435), C(440), \\ C(445), C(450), C(455), C(459), C(460), C(465), C(470), C(475), C(479), C(480), C(485), C(490), C(495), \\ C(499), C(500), C(505), C(509), C(510), C(515), C(517), C(519), C(520), C(522), C(524), C(525), C(526), \\ C(527), C(529), C(530), C(531), C(532) \text{ and } C(534) \\ \\ \\ \\ C(527), C(529), C(530), C(531), C(532) \text{ and } C(534) \\ \\ \\ \\ C(527), C(529), C(520), C(530), C(531), C(532) \text{ and } C(534) \\ \\ \\ \\ C(527), C(529), C(520), C(530), C(531), C(532) \text{ and } C(534) \\ \\ \\ \\ C(527), C(529), C(530), C(531), C(532), C(532), C(535), C(355), C(355), C(355), C(355), C(354), C(355), C($ 

are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the inband bits in c as

$$c(8+k) = P(k)$$
 for  $k = 0, 1, ..., 447$ .

#### 3.9.4.5 Interleaving

The interleaving is done as specified for the TCH/FS in subclause 3.1.3.

## 3.9.4.6 Mapping on a Burst

The mapping is done as specified for the TCH/FS in subclause 3.1.4.

#### 3.9.5 RATSCCH

The RATSCCH message consists of 35 bits. Also delivered are two in-band channels, id0(0,1) and id1(0,1), id0 corresponding to Mode Commands or Mode Requests and id1 to Mode Indication. The general coding is as: the two inband data channels are coded to 16 bits each, a 14-bit CRC is added to the 35 RATSCCH bits which are then coded by a rate ½ RSC coder to 212 bits. Finally a 212 bit identification field is added thereby giving a total size of 456 bits. These 456 bits are then block interleaved in the same way as a normal speech frame.

## 3.9.5.1 Coding of in-band data

The two in-band data fields, idO(0,1) and id1(0,1), are encoded, giving icO(0..15) and ic1(0..15). These bits are moved to the coded bits c as:

$$c(k) = ic1(k)$$
 for  $k = 0,1, ..., 15$   $c(k+228) = ic0(k)$  for  $k = 0, 1, ..., 15$ 

#### 3.9.5.2 Parity and convolutional encoding for the RATSCCH message

a) Parity bits:

A 14-bit CRC is used for error-detection. These 14 parity bits are generated by the cyclic generator polynomial:  $g(D) = D^{14} + D^{13} + D^5 + D^3 + D^2 + I$  from the 35 comfort noise parameter bits. The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

$$d(0)D(48) + d(1)D(47) + ... + d(34)D(14) + p(0)D(13) + ... + p(12)D + p(13)$$
 where p(0), p(1) ... p(13) are the parity bits, when divided by g(D), yields a remainder equal to 1+ D + D<sup>2</sup> + D<sup>3</sup> + D<sup>4</sup> + D<sup>5</sup> + D<sup>6</sup> + D<sup>7</sup> + D<sup>8</sup> + D<sup>9</sup> + D<sup>10</sup> + D<sup>11</sup> + D<sup>12</sup> + D<sup>13</sup>

The information and parity bits are merged:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 34$   $u(k) = p(k-35)$  for  $k = 35, 36, ..., 48$ 

#### b) Convolutional encoder

The comfort noise parameters with parity and tail bits (u(0..48)) are encoded with the  $\frac{1}{4}$  rate

convolutional code defined by the polynomials:

$$G1/G3 = 1 + D + D^{3} + D^{4} / 1 + D + D^{2} + D^{3} + D^{4}$$

$$G2/G3 = 1 + D^{2} + D^{4} / 1 + D + D^{2} + D^{3} + D^{4}$$

$$G3/G3 = 1$$

$$G3/G3 = 1$$

resulting in 212 coded bits,  $\{C(0)...C(211)\}\$  defined by:

$$\begin{array}{ll} r(k) & = u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) \\ \\ C(4k) & = r(k) + r(k-1) + r(k-3) + r(k-4) \\ \\ C(4k+1) & = r(k) + r(k-2) + r(k-4) \\ \\ C(4k+2) & = u(k) \\ \\ C(4k+3) & = u(k) \end{array} \qquad \text{for } k = 0, 1, ..., 48; r(k) = 0 \text{ for } k < 0.$$

and (for termination of the coder):

$$r(k) = 0$$

$$C(4k) = r(k)+r(k-1)+r(k-3)+r(k-4)$$

$$C(4k+1) = r(k)+r(k-2)+r(k-4)$$

$$C(4k+2) = r(k-1)+r(k-2)+r(k-3)+r(k-4)$$

$$C(4k+3) = r(k-1)+r(k-2)+r(k-3)+r(k-4)$$

for 
$$k = 49, 50, ..., 52$$

This block of data is moved to the coded data (c) as:

$$c(k+244) = C(k)$$

for 
$$k = 0, 1, ..., 211$$

#### 3.9.5.3 Identification marker

The identification marker, IM(0..211), is constructed by repeating the following 11-bit sequence: { 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1 } 20 times and then discarding the last 8 bits. This block of data is moved to the coded data (c) as:

$$c(k+16) = IM(k)$$

for 
$$k = 0, 1, ..., 211$$

#### 3.9.5.4 Interleaving

The interleaving is done as specified for the TCH/FS in subclause 3.1.3.

#### 3.9.5.5 Mapping on a Burst

The mapping is done as specified for the TCH/FS in subclause 3.1.4.

# 3.10 Adaptive multi rate speech channel at half rate (TCH/AHS)

This section describes the coding for the different frame formats used for TCH/AHS. The formats used are (in the order they are described):

SID\_UPDATE Used to convey comfort noise parameters during DTX

SID\_UPDATE\_INH Used to inhibit the second part of a SID\_UPDATE frame if there is a speech onset

SID\_FIRST\_P1 First part of marker to define end of speech, start of DTX

SID\_FIRST\_P2 Second part of marker to define end of speech, start of DTX

SID\_FIRST\_INH Used to inhibit the second part of a SID\_FIRST\_P1 frame if there is a speech onset

ONSET Used to signal the Codec mode for the first speech frame after DTX

SPEECH Speech frames

RATSCCH\_MARKER Marker to identify RATSCCH frames

RATSCCH\_DATA Frame that conveys the actual RATSCCH message

In this chapter, sub chapters 3.10.1 to 3.10.9 describe the channel coding for the different formats listed above.

Common to all the formats is that in-band information is conveyed, the coding for the in-band channel is described in the table below:

Identifier (defined in 3GPP TS 45.009)	Received in-band data id(1), id(0)	Encoded in-band data for SID and RATSCCH frames ic(15),, ic(0)	Encoded in-band data for speech framesic(3),, ic(0)
CODEC_MODE_1	00	0101001100001111	0000
CODEC_MODE_2	01	0011111010111000	1001
CODEC_MODE_3	10	1000100001100011	0111
CODEC_MODE_4	11	1110010111010100	1110

## 3.10.1 SID\_UPDATE

The speech encoder delivers 35 bits of comfort noise parameters. Also delivered is two in-band channels, id0(0,1) and id1(0,1), id0 corresponding to Mode Commands/Mode Requests and id1 to Mode Indication. The general coding is as: the two in-band data channels are coded to 16 bits each, a 14-bit CRC is added to the 35 CN bits which are then coded by a rate ½ RSC coder to 212 bits. Finally a 212 bit identification field is added thereby giving a total size of 456 bits. These 456 bits are block interleaved over 4 bursts.

## 3.10.1.1 Coding of in-band data

The two in-band data fields, idO(0,1) and id1(0,1), are encoded, giving icO(0..15) and ic1(0..15).

The ic0 and ic1 data is moved to the coded data c as:

$$c(k) = ic1(k)$$
 for  $k = 0,1, ..., 15$   
 $c(k) = ic0(k-228)$  for  $k = 228, 229, ..., 243$ 

## 3.10.1.2 Parity and convolutional encoding for the comfort noise parameters

a) Parity bits:

A 14-bit CRC is used for error-detection. These 14 parity bits are generated by the cyclic generator polynomial:  $g(D) = D^{14} + D^{13} + D^5 + D^3 + D^2 + I$  from the 35 comfort noise parameter bits. The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

$$d(0)D(48) + d(1)D(47) + ... + d(34)D(14) + p(0)D(13) + ... + p(12)D + p(13)$$
 where p(0), p(1) ... p(13) are the parity bits, when divided by g(D), yields a remainder equal to 
$$1 + D + D^2 + D^3 + D^4 + D^5 + D^6 + D^7 + D^8 + D^9 + D^{10} + D^{11} + D^{12} + D^{13}$$

The information and parity bits are merged:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 34$   
 $u(k) = p(k-35)$  for  $k = 35, 36, ..., 48$ 

#### b) Convolutional encoder

The comfort noise parameters with parity bits (u(0..48)) are encoded with the  $\frac{1}{4}$  rate

convolutional code defined by the polynomials:

$$G1/G3 = 1 + D + D^{3} + D^{4} / 1 + D + D^{2} + D^{3} + D^{4}$$

$$G2/G3 = 1 + D^{2} + D^{4} / 1 + D + D^{2} + D^{3} + D^{4}$$

$$G3/G3 = 1$$

$$G3/G3 = 1$$

resulting in 212 coded bits,  $\{C(0)...C(211)\}\$  defined by:

$$r(k)$$
 =  $u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4)$ 

$$\begin{array}{ll} C(4k) & = r(k) + r(k-1) + r(k-3) + r(k-4) \\ \\ C(4k+1) & = r(k) + r(k-2) + r(k-4) \\ \\ C(4k+2) & = u(k) \\ \\ C(4k+3) & = u(k) \end{array} \qquad \text{for } k = 0, 1, ..., 48; r(k) = 0 \text{ for } k < 0 \\ \\ C(4k+3) & = u(k) \end{array}$$

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(4k) &= r(k) + r(k-1) + r(k-3) + r(k-4) \\ C(4k+1) &= r(k) + r(k-2) + r(k-4) \\ C(4k+2) &= r(k-1) + r(k-2) + r(k-3) + r(k-4) \\ C(4k+3) &= r(k-1) + r(k-2) + r(k-3) + r(k-4) & \text{for } k = 49, 50, ..., 52 \end{split}$$

This block of data is moved to the coded data (c) as:

$$c(k+244) = C(k)$$
 for  $k = 0, 1, ..., 211$ 

#### 3.10.1.3 Identification marker

The identification marker, IM(0..211), is constructed by repeating the following 9-bit sequence: { 1, 0, 1, 1, 0, 0, 0, 0, 1 } 24 times and then discarding the last 4 bits. This block of data is moved to the coded data (c) as:

$$c(k+16) = IM(k)$$
 for  $k = 0, 1, ..., 211$ 

#### 3.10.1.4 Interleaving

The coded bits are reordered and interleaved according to the following rule:

$$\begin{split} i(B,j) &= c(n,k) \, \text{for} \, \, k = 0,1,...,227 \\ &\quad n = 0,1,...,N,N+1,... \\ &\quad B = B0 + 2n + b \\ \\ i(B,j) &= c(n,k+228) \quad \text{for} \, \, k = 0,1,...,227 \\ &\quad n = 0,1,...,N,N+1,... \\ &\quad B = B0 + 2n + ((b+2) \ \text{mod} \, \, 4) \end{split}$$

The values of b and j in dependence of k are given by table 4.

The result of the interleaving is a distribution of the 456 bits of a given data block, n = N, over 4 blocks using all bits for each block. The block of coded data is interleaved "block rectangular" where a new data block starts every  $4^{th}$  block and is distributed over 4 blocks.

### 3.10.1.5 Mapping on a Burst

The mapping is given by the rule:

$$e(B,j)=i(B,j) \mbox{ and } e(B,59+j)=i(B,57+j) \mbox{ for } j=0,1,...,56$$
 and 
$$e(B,57)=hl(B) \mbox{ and } e(B,58)=hu(B)$$

The two bits, labelled hl(B) and hu(B) on burst number B are flags used for indication of control channel signalling. For each block not stolen for FACCH signalling purposes:

$$hu(B)$$
 = 0 for all 4 bursts  
 $hl(B)$  = 0 for all 4 bursts

For the use of hl(B) and hu(B) when frame is stolen for signalling purposes, see subclause 4.3.5.

## 3.10.2 SID UPDATE INH

This special frame is used when the first 2 burst of a SID\_UPDATE frame have been transmitted but the second two bursts cannot be transmitted due to a speech frame. The general coding is as: the in-band data (Note that this must be the same Mode Indication bits as id1(0,1) for the SID\_UPDATE frame that is being inhibited) is encoded, a marker that is the opposite of the SID\_UPDATE marker is appended and the data is interleaved in such a way that the odd bits of two bursts are filled.

## 3.10.2.1 Coding of in-band data

The in-band data, Mode Indication id1(0,1), is encoded to ic1(0..15) which is moved to the coded data c as:

$$c(k) = ic1(k)$$
 for  $k = 0,1, ..., 15$ 

#### 3.10.2.2 Identification marker

The identification marker, IM(0..211), is constructed by repeating the following 9-bit sequence: { 0, 1, 0, 0, 1, 1, 1, 1, 0 } 24 times and then discarding the last 4 bits. This block of data is moved to the coded data (c) as:

$$c(k+16) = IM(k)$$
 for  $k = 0, 1, ..., 211$ 

## 3.10.2.3 Interleaving

The coded bits are reordered and interleaved according to the following rule:

$$i(B,j) = c(n,k) \qquad \qquad for \qquad \qquad k = 1,3,5,7,...,227$$
 
$$n = 0,1,...,N,N+1,...$$
 
$$B = B0 + 2n + b - 2$$

The values of b and j in dependence of k are given by table 4.

The result of the interleaving is a distribution of 114 of the reordered 228 bits of a given data block, n = N, over 2 blocks using the odd numbered bits. The even numbered bits of these 2 blocks will be filled by the speech frame that following immediately after this frame.

#### 3.10.2.4 Mapping on a Burst

The mapping is given by the rule:

$$e(B,j) = i(B,j) \mbox{ and } e(B,59+j) = i(B,57+j) \mbox{ for } j = 0,1,...,56$$
 and 
$$e(B,57) = hl(B)$$

The bit labelled hl(B) on burst number B is a flag used for indication of control channel signalling. For each SID\_FIRST\_INH block not stolen for signalling purposes:

hl(B) = 0 for the 2 bursts (indicating status of the odd numbered bits)

For the use of hl(B) when a SID\_UPDATE\_INH is stolen for signalling purposes, see subclause 4.3.5.

## 3.10.3 SID\_FIRST\_P1

This frame type contains no source data from the speech coder. What is generated is the in-band channel and an identification marker. The in-band data id(0,1) represents Mode Indication or Mode Command/Mode Request depending on the current frame number.

#### 3.10.3.1 Coding of in-band data

The in-band data, id(0,1), is encoded to ic(0..15) which is moved to the coded data c as:

$$c(k) = ic(k)$$
 for  $k = 0,1,...,15$ 

#### 3.10.3.2 Identification marker

The identification marker, IM(0..211), is constructed by repeating the following 9-bit sequence: { 0, 1, 0, 0, 1, 1, 1, 1, 0 } 24 times and then discarding the last 4 bits. This block of data is moved to the coded data (c) as:

$$c(k+16) = IM(k)$$
 for  $k = 0, 1, ..., 211$ 

#### 3.10.3.3 Interleaving

The interleaving is done as specified for the TCH/HS in subclause 3.2.3.

#### 3.10.3.4 Mapping on a Burst

The mapping is done as specified for the TCH/HS in subclause 3.2.4.

## 3.10.4 SID\_FIRST\_P2

This frame type contains no source data from the speech coder. What is generated is the in-band channel and, derived from that, an identification marker. The in-band data id(0,1) represents Mode Indication or Mode Command/Mode Request depending on the current frame number.

#### 3.10.4.1 Coding of in-band data

The in-band data, id(0,1), is encoded to ic(0..15). This sequence is then repeated 7 times more, and the last 14 bits are discarded (8\*16-14=114) giving the sequence ic(0..113).

This sequence is then moved to c as:

$$c(2*k) = ic(k)$$
 for  $k = 0, 1, ..., 113$ 

#### 3.10.4.2 Interleaving

The coded bits are reordered and interleaved according to the following rule:

$$i(B,j) = c(n,k) \qquad \qquad for \qquad \qquad k = 0,2,4,6,...,226$$
 
$$n = 0,1,...,N,N+1,...$$
 
$$B = B0 + 2n + b$$

The values of b and j in dependence of k are given by table 4.

The result of the interleaving is a distribution of 114 of the reordered 228 bits of a given data block, n = N, over 2 blocks using the even numbered bits. The odd numbered bits of these 2 blocks have already been filled by the SID\_FIRST\_P1 frame.

# 3.10.4.3 Mapping on a Burst

The mapping is given by the rule:

$$\label{eq:ebspace} e(B,j) = i(B,j) \text{ and } e(B,59+j) = i(B,57+j) \text{ for } j = 0,1,...,56$$
 and

$$e(B,58) = hu(B)$$

The bit labelled hu(B) on burst number B is a flag used for indication of control channel signalling. For each SID\_FIRST\_P2 block not stolen for signalling purposes:

hu(B) = 0 for the 2 bursts (indicating status of the even numbered bits)

For the use of hu(B) when a SID\_FIRST\_P2 is stolen for signalling purposes, see subclause 4.3.5.

# 3.10.5 SID\_FIRST\_INH

This special frame is used when the first 2 burst of a SID\_FIRST\_P1 frame have been transmitted but the second two bursts cannot be transmitted due to a SPEECH frame. The general coding is as: the in-band data (Note that this must be the same data as for the SID\_FIRST\_P1 frame that is being inhibited) is encoded, a marker that is the opposite of the SID\_FIRST\_P1 marker is appended and the data is interleaved in such a way that the odd bits of two bursts are filled.

# 3.10.5.1 Coding of in-band data

The coding of the in-band data is done as specified for the SID\_FIRST\_P1 frame in subclause 3.10.3.1.

#### 3.10.5.2 Identification marker

The identification marker, IM(0..211), is constructed by repeating the following 9-bit sequence: { 1, 0, 1, 1, 0, 0, 0, 0, 1 } 24 times and then discarding the last 4 bits. This block of data is moved to the coded data (c) as:

$$c(k+16) = IM(k)$$
 for  $k = 0, 1, ..., 211$ 

#### 3.10.5.3 Interleaving

The interleaving is done as specified for the SID\_UPDATE\_INH in subclause 3.10.2.3.

#### 3.10.5.4 Mapping on a Burst

The mapping is done as specified for the SID\_UPDATE\_INH in subclause 3.10.2.4.

#### 3.10.6 ONSET

Onset frames are used to preset the interleaver buffer after a period of no speech activity in DTX mode. This frame type contains no source data from the speech coder. What is transmitted is the in-band channel signalling the Mode Indication for the speech frame following the onset marker.

# 3.10.6.1 Coding of in-band data

The in-band data, Mode Indication id1(0,1), will be encoded to ic1(0..15). This sequence is then repeated 7 times more, and the last 14 bits are discarded (8\*16-14=114) giving the sequence ic1(0..113).

This sequence is then moved to c as:

$$c(2*k+1) = ic1(k)$$
 for  $k = 0, 1, ..., 113$ 

# 3.10.6.2 Interleaving

The interleaving is done as specified for the SID\_UPDATE\_INH in subclause 3.10.2.3.

# 3.10.6.3 Mapping on a Burst

The mapping is done as specified for the SID\_UPDATE\_INH in subclause 3.10.2.4.

# 3.10.7 SPEECH

The speech coder delivers to the channel encoder a sequence of blocks of data. One block of data corresponds to one speech frame and the block length is different in each of the six channel codec modes. Adjoining each block of data is information of the channel codec mode to use when encoding the block Also delivered is the in-band data id(0,1) representing Mode Indication or Mode Command/Mode Request depending on the current frame number.

# 3.10.7.1 Coding of the in-band data

The two bits to be in-band encoded, id(0,1), are encoded into ic(0..3).

The encoded in-band data (4 bits) are then moved to c(k) as:

$$c(k) = ic(k)$$
 for  $k = 0, 1, ..., 3$ 

# 3.10.7.2 Ordering according to subjective importance

The bits delivered by the speech encoder,  $\{s(1), s(2), ..., s(K_s)\}$ , are rearranged according to subjective importance before channel coding, Tables 9, 10, 11, 12, 13, 14 define the correct rearrangement for the speech codec modes 7.95 kbit/s, 7.40 kbit/s, 6.70 kbit/s, 5.90 kbit/s, 5.15 kbit/s and 4.75 kbit/s, respectively. In the tables speech codec parameters are numbered in the order they are delivered by the corresponding speech encoder according to 3GPP TS 26.090 and the rearranged bits are labelled  $\{d(0),d(1),...,d(K_d-1)\}$ , defined in the order of decreasing importance. Index  $K_d$  refers to the number of bits delivered by the speech encoder, see below:

Codec mode	Number of speech bits delivered per block (K <sub>d</sub> )
TCH/AHS7.95	159
TCH/AHS7.4	148
TCH/AHS6.7	134
TCH/AHS5.9	118
TCH/AHS5.15	103
TCH/AHS4.75	95

The ordering algorithm is in pseudo code as:

for 
$$j = 0$$
 to  $K_{d-1}$   $d(j) := s(table(j)+1)$ ; where  $table(j)$  is read line by line left to right

The rearranged bits are further divided into three different classes to perform unequal error protection for different bits according to subjective importance.

The protection classes are:

1a - Data protected with the CRC and the convolution code.

1b - Data protected with the convolution code.

2 - Data sent without protection.

The number of class 1 (sum of class 1a and 1b), class 1a, class 1b and class 2 bits for each codec mode is shown below:

Codec mode	Number of speech bits delivered per block	Number of class 1 bits per block	Number of class 1a bits per block	Number of class 1b bits per block	Number of class 2 bits per block
TCH/AHS7.95	159	123	67	56	36
TCH/AHS7.4	148	120	61	59	28
TCH/AHS6.7	134	110	55	55	24
TCH/AHS5.9	118	102	55	47	16
TCH/AHS5.15	103	91	49	42	12
TCH/AHS4.75	95	83	39	44	12

# 3.10.7.3 Parity for speech frames

The basic parameters for each codec mode for the first encoding step are shown below:

Mode number	Number of class 1 bits (K <sub>d1</sub> )	CRC protected bits (K <sub>d1a</sub> )	Number of output bits from first encoding step $(K_{u} = K_{d1} + 6)$
TCH/AHS7.95	123	67	129
TCH/AHS7.4	120	61	126
TCH/AHS6.7	110	55	116
TCH/AHS5.9	102	55	108
TCH/AHS5.15	91	49	97
TCH/AHS4.75	83	39	89

A 6-bit CRC is used for error-detection. These 6 parity bits are generated by the cyclic generator polynomial:  $g(D) = D^6 + D^5 + D^3 + D^2 + D^1 + I$  from the first  $K_{d1a}$  bits of class 1, where  $K_{d1a}$  refers to number of bits in protection class 1a. The value of  $K_{d1a}$  for each codec mode is shown above.

The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:  $d(0)D(K_{d1a}+5) + d(1)D(K_{d1a}+4) + ... + d(K_{d1a}-1)D(6) + p(0)D(5) + ... + p(4)D + p(5)$ 

where p(0), p(1) ... p(5) are the parity bits, when divided by g(D), yields a remainder equal to:

$$1 + D + D^2 + D^3 + D^4 + D^5$$
.

The information and parity bits are merged:

$$\begin{split} u(k) &= d(k) & \text{for } k = 0,\,1,\,...,\,K_{d1a}\text{-}1 \\ \\ u(k) &= p(k\text{-}K_{d1a}) & \text{for } k = K_{d1a},\,K_{d1a}\text{+}1,\,...,\,K_{d1a}\text{+}5 \\ \\ u(k) &= d(k\text{-}6) & \text{for } k = K_{d1a}\text{+}6,\,K_{d1a}\text{+}7,\,...,\,K_{u}\text{-}1 \end{split}$$

Thus, after the first encoding step u(k) will be defined by the following contents for each codec mode:

#### **TCH/AHS7.95**:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 66$   $u(k) = p(k-67)$  for  $k = 67, 68, ..., 72$   $u(k) = d(k-6)$  for  $k = 73, 74, ..., 128$ 

#### **TCH/AHS7.4**:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 60$  
$$u(k) = p(k-61)$$
 for  $k = 61, 62, ..., 66$ 

$$u(k) = d(k-6)$$
 for  $k = 67, 68, ..., 125$ 

#### TCH/AHS6.7:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 54$ 

$$u(k) = p(k-55)$$
 for  $k = 55, 56, ..., 60$ 

$$u(k) = d(k-6)$$
 for  $k = 61, 62, ..., 115$ 

#### TCH/AHS5.9:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 54$ 

$$u(k) = p(k-55)$$
 for  $k = 55, 56, ..., 60$ 

$$u(k) = d(k-6)$$
 for  $k = 61, 62, ..., 107$ 

#### **TCH/AHS5.15**:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 48$ 

$$u(k) = p(k-49)$$
 for  $k = 49, 50, ..., 54$ 

$$u(k) = d(k-6)$$
 for  $k = 55, 56, ..., 96$ 

#### **TCH/AHS4.75**:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 38$ 

$$u(k) = p(k-39)$$
 for  $k = 39, 40, ..., 44$ 

$$u(k) = d(k-6)$$
 for  $k = 45, 46, ..., 88$ 

#### 3.10.7.4 Convolutional encoder

The bits from the first encoding step (u(k)) are encoded with the recursive systematic convolutional code as summarised below:

Codec mode	Number of input bits to conv. code	Rate	Number of output bits from conv. code	Number of punctured bits
TCH/AHS7.95	129	1/2	266	78
TCH/AHS7.4	126	1/2	260	64
TCH/AHS6.7	116	1/2	240	40
TCH/AHS5.9	108	1/2	224	16
TCH/AHS5.15	97	1/3	303	91
TCH/AHS4.75	89	1/3	285	73

Below the coding for each codec mode is specified in detail.

#### **TCH/AHS7.95**:

The block of 129 bits  $\{u(0)...u(128)\}$  is encoded with the ½ rate convolutional code defined by polynomials:

$$G0/G0 = 1$$

$$G1/G0 = 1 + D + D^3 + D^4 / 1 + D^3 + D^4$$

resulting in 266 coded bits,  $\{C(0)...C(265)\}\$  defined by:

$$r(k) = u(k) + r(k-3) + r(k-4)$$

$$C(2k) = u(k)$$

$$C(2k+1) = r(k)+r(k-1)+r(k-3)+r(k-4)$$
 for  $k = 0, 1, ..., 128$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$r(k) = 0$$

$$C(2k) = r(k-3) + r(k-4)$$

$$C(2k+1) = r(k)+r(k-1)+r(k-3)+r(k-4)$$
 for  $k = 129, 130 ..., 132$ 

The code is punctured in such a way that the following 78 coded bits:

are not transmitted. The result is a block of 188 coded and punctured bits, P(0)...P(187) which are appended to the in-band bits in c as

$$c(k+4) = P(k)$$
 for  $k = 0, 1, ..., 187$ .

Finally the 36 class 2 bits are appended to c

$$c(192+k) = d(123+k)$$
 for  $k = 0, 1, ..., 35$ .

#### **TCH/AHS7.4**:

The block of 126 bits  $\{u(0)...u(125)\}$  is encoded with the ½ rate convolutional code defined by polynomials:

$$G0/G0 = 1$$

$$G1/G0 = 1 + D + D^3 + D^4 / 1 + D^3 + D^4$$

resulting in 260 coded bits,  $\{C(0)...C(259)\}\$  defined by:

$$r(k) = u(k) + r(k-3) + r(k-4)$$

$$C(2k) = u(k)$$

$$C(2k+1) = r(k)+r(k-1)+r(k-3)+r(k-4)$$
 for  $k = 0, 1, ..., 125$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$r(k) = 0$$

$$C(2k) = r(k-3) + r(k-4)$$

$$C(2k+1) = r(k)+r(k-1)+r(k-3)+r(k-4)$$
 for  $k = 126, 127 ..., 129$ 

The code is punctured in such a way that the following 64 coded bits:

```
C(1), C(3), C(7), C(11), C(19), C(23), C(27), C(35), C(39), C(43), C(51), C(55), C(59), C(67), C(71), C(75), C(83), C(87), C(91), C(99), C(103), C(107), C(115), C(119), C(123), C(131), C(135), C(139), C(143), C(147), C(151), C(155), C(159), C(163), C(167), C(171), C(175), C(179), C(183), C(187), C(191), C(195), C(199), C(203), C(207), C(211), C(215), C(219), C(221), C(223), C(227), C(229), C(231), C(235), C(237), C(239), C(243), C(245), C(247), C(251), C(253), C(255), C(257) and C(259)
```

are not transmitted. The result is a block of 196 coded and punctured bits, P(0)...P(195) which are appended to the in-band bits in c as

$$c(k+4) = P(k)$$
 for  $k = 0, 1, ..., 195$ .

Finally the 28 class 2 bits are appended to c

$$c(200+k) = d(120+k)$$
 for  $k = 0, 1, ..., 27$ .

#### TCH/AHS6.7:

The block of 116 bits  $\{u(0)...u(115)\}$  is encoded with the ½ rate convolutional code defined by polynomials:

$$G0/G0 = 1$$

$$G1/G0 = 1 + D + D^3 + D^4 / 1 + D^3 + D^4$$

resulting in 240 coded bits,  $\{C(0)...C(239)\}\$  defined by:

$$r(k) = u(k) + r(k-3) + r(k-4)$$

$$C(2k) = u(k)$$

$$C(2k+1) = r(k)+r(k-1)+r(k-3)+r(k-4)$$
 for  $k = 0, 1, ..., 115$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$r(k) = 0$$

$$C(2k) = r(k-3) + r(k-4)$$

$$C(2k+1) = r(k)+r(k-1)+r(k-3)+r(k-4)$$
 for  $k = 116, 117 ..., 119$ 

The code is punctured in such a way that the following 40 coded bits:

are not transmitted. The result is a block of 200 coded and punctured bits, P(0)...P(199) which are appended to the in-band bits in c as

$$c(k+4) = P(k)$$
 for  $k = 0, 1, ..., 199$ .

Finally the 24 class 2 bits are appended to c

$$c(204+k) = d(110+k)$$
 for  $k = 0, 1, ..., 23$ .

#### TCH/AHS5.9:

The block of 108 bits  $\{u(0)...u(107)\}$  is encoded with the ½ rate convolutional code defined by polynomials:

$$G0/G0 = 1$$

$$G1/G0 = 1 \, + \, D + \, D^3 + \, D^4 \, / \, \, 1 \, + \, D^3 \, + \, D^4$$

resulting in 224 coded bits,  $\{C(0)...C(223)\}\$  defined by:

$$r(k) = u(k) + r(k-3) + r(k-4)$$

$$C(2k) = u(k)$$

$$C(2k+1) = r(k)+r(k-1)+r(k-3)+r(k-4)$$
 for  $k = 0, 1, ..., 107$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$r(k) = 0$$

$$C(2k) = r(k-3) + r(k-4)$$

$$C(2k+1) = r(k)+r(k-1)+r(k-3)+r(k-4)$$
 for  $k = 108, 109 ..., 111$ 

The code is punctured in such a way that the following 16 coded bits:

are not transmitted. The result is a block of 208 coded and punctured bits, P(0)...P(207) which are appended to the in-band bits in c as

$$c(k+4) = P(k)$$
 for  $k = 0, 1, ..., 207$ .

Finally the 16 class 2 bits are appended to c

$$c(212+k) = d(102+k)$$
 for  $k = 0, 1, ..., 15$ .

#### **TCH/AHS5.15**:

The block of 97 bits  $\{u(0)...u(96)\}$  is encoded with the 1/3 rate convolutional code defined by the following polynomials:

$$G1/G3 = 1 + D + D^{3} + D^{4} / 1 + D + D^{2} + D^{3} + D^{4}$$
 
$$G2/G3 = 1 + D^{2} + D^{4} / 1 + D + D^{2} + D^{3} + D^{4}$$

G3/G3 = 1

resulting in 303 coded bits,  $\{C(0)...C(302)\}\$  defined by:

$$r(k)$$
 =  $u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4)$ 

$$C(3k) = r(k) + r(k-1) + r(k-3) + r(k-4)$$

$$C(3k+1) = r(k)+r(k-2)+r(k-4)$$

$$C(3k+2) = u(k)$$
 for  $k = 0, 1, ..., 96$ 

and (for termination of the coder):

$$r(k) = 0$$

$$C(3k) = r(k)+r(k-1) + r(k-3) + r(k-4)$$

$$C(3k+1) = r(k)+r(k-2)+r(k-4)$$

$$C(3k+2) = r(k-1)+r(k-2)+r(k-3)+r(k-4)$$
 for  $k = 97, 98, ..., 100$ 

The code is punctured in such a way that the following 91 coded bits:

 $\begin{array}{l} C(0), C(1), C(3), C(4), C(6), C(9), C(12), C(15), C(18), C(21), C(27), C(33), C(39), C(45), C(51), C(54), \\ C(57), C(63), C(69), C(75), C(81), C(87), C(90), C(93), C(99), C(105), C(111), C(117), C(123), C(126), \\ C(129), C(135), C(141), C(147), C(153), C(159), C(162), C(165), C(168), C(171), C(174), C(177), C(180), \\ C(183), C(186), C(189), C(192), C(195), C(198), C(201), C(204), C(207), C(210), C(213), C(216), C(219), \\ C(222), C(225), C(228), C(231), C(234), C(237), C(240), C(243), C(244), C(246), C(249), C(252), C(255), \\ C(256), C(258), C(261), C(264), C(267), C(268), C(270), C(273), C(276), C(279), C(280), C(282), C(285), \\ C(288), C(289), C(291), C(294), C(295), C(297), C(298), C(300) \ \text{and} \ C(301) \end{array}$ 

are not transmitted. The result is a block of 212 coded and punctured bits, P(0)...P(211) which are appended to the in-band bits in c as

$$c(k+4) = P(k)$$
 for  $k = 0, 1, ..., 211$ .

Finally the 12 class 2 bits are appended to c

$$c(216+k) = d(91+k)$$
 for  $k = 0, 1, ..., 11$ .

# TCH/AHS4.75:

The block of 89 bits  $\{u(0)...u(88)\}$  is encoded with the 1/3 rate convolutional code defined by the following polynomials:

$$G4/G4 = 1$$

$$G5/G4 = 1 + D + D^4 + D^6/1 + D^2 + D^3 + D^5 + D^6$$

$$G6/G4 = 1 + D + D^2 + D^3 + D^4 + D^6/1 + D^2 + D^3 + D^5 + D^6$$

resulting in 285 coded bits,  $\{C(0)...C(284)\}\$  defined by:

$$r(k) = u(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(3k) = u(k)$$

$$C(3k+1) = r(k)+r(k-1)+r(k-4)+r(k-6)$$

$$C(3k+2) = r(k)+r(k-1)+r(k-2)+r(k-3)+r(k-4)+r(k-6)$$
 for  $k = 0, 1, ..., 88$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$r(k) = 0$$

$$C(3k) = r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(3k+1) = r(k)+r(k-1)+r(k-4)+r(k-6)$$

$$C(3k+2) = r(k)+r(k-1)+r(k-2)+r(k-3)+r(k-4)+r(k-6)$$
 for  $k = 89, 90, ..., 94$ 

The code is punctured in such a way that the following 73 coded bits:

C(1), C(2), C(4), C(5), C(7), C(8), C(10), C(13), C(16), C(22), C(28), C(34), C(40), C(46), C(52), C(58), C(64), C(70), C(76), C(82), C(88), C(94), C(100), C(106), C(112), C(118), C(124), C(130), C(136), C(142), C(148), C(151), C(154), C(160), C(163), C(166), C(172), C(175), C(178), C(184), C(187), C(190), C(196), C(199), C(202), C(208), C(211), C(214), C(220), C(223), C(226), C(232), C(235), C(238), C(241), C(244), C(247), C(250), C(253), C(256), C(259), C(262), C(265), C(268), C(271), C(274), C(275), C(277), C(278), C(280), C(281), C(283) and C(284)

are not transmitted. The result is a block of 212 coded and punctured bits, P(0)...P(211) which are appended to the in-band bits in c as

$$c(k+4) = P(k)$$
 for  $k = 0, 1, ..., 211$ .

Finally the 12 class 2 bits are appended to c

$$c(216+k) = d(83+k)$$
 for  $k = 0, 1, ..., 11$ .

### 3.10.7.5 Interleaving

The interleaving is done as specified for the TCH/HS in subclause 3.2.3.

# 3.10.7.6 Mapping on a Burst

The mapping is done as specified for the TCH/HS in subclause 3.2.4.

# 3.10.8 RATSCCH MARKER

This frame type contains the in-band channel and an identification marker. The in-band data id(0,1) represents Mode Indication or Mode Command/Mode Request depending on the current frame number.

#### 3.10.8.1 Coding of in-band data

The in-band data, ic(0,1), is encoded to ic(0..15) which is moved to the coded data c as:

$$c(k) = ic(k)$$
 for  $k = 0,1, ..., 15$ 

### 3.10.8.2 Identification marker

The identification marker, IM(0..211), is constructed by repeating the following 11-bit sequence: { 1, 0, 0, 1, 0, 0, 0, 1 } 20 times and then discarding the last 8 bits. This block of data is moved to the coded data (c) as:

$$c(k+16) = IM(k)$$
 for  $k = 0, 1, ..., 211$ 

# 3.10.8.3 Interleaving

The interleaving is done as specified for the TCH/HS in subclause 3.2.3.

# 3.10.8.4 Mapping on a Burst

The mapping is done as specified for the TCH/HS in subclause 3.2.4.

# 3.10.9 RATSCCH DATA

This frame contains the RATSCCH data and an inband channel. The RATSCCH data consists of 35 bits. The in-band data id(0,1) represents Mode Indication or Mode Command/Mode Request depending on the current frame number.

# 3.10.9.1 Coding of in-band data

The in-band data, ic(0,1), is encoded to ic(0..15) which is moved to the coded data c as:

$$c(k) = ic(k)$$
 for  $k = 0,1, ..., 15$ 

# 3.10.9.2 Parity and convolutional encoding for the RATSCCH message

a) Parity bits:

A 14-bit CRC is used for error-detection. These 14 parity bits are generated by the cyclic generator polynomial:  $g(D) = D^{14} + D^{13} + D^5 + D^3 + D^2 + I$  from the 35 comfort noise parameter bits. The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

$$d(0)D^{\left(48\right)}+d(1)D^{\left(47\right)}+...+d(34)D^{\left(14\right)}+p(0)D^{\left(13\right)}+...+p(12)D+p(13)$$

where p(0), p(1) ... p(13) are the parity bits, when divided by g(D), yields a remainder equal to  $1+D+D^2+D^3+D^4+D^5+D^6+D^7+D^8+D^9+D^{10}+D^{11}+D^{12}+D^{13}$ 

The information and parity bits are merged:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 34$   
 $u(k) = p(k-35)$  for  $k = 35, 36, ..., 48$ 

#### b) Convolutional encoder

The comfort noise parameters with parity and tail bits (u(0..48)) are encoded with the  $\frac{1}{4}$  rate

convolutional code defined by the polynomials:

$$G1/G3 = 1 + D + D^{3} + D^{4} / 1 + D + D^{2} + D^{3} + D^{4}$$

$$G2/G3 = 1 + D^{2} + D^{4} / 1 + D + D^{2} + D^{3} + D^{4}$$

$$G3/G3 = 1$$

$$G3/G3 = 1$$

resulting in 212 coded bits, {C(0)... C(211)} defined by:

$$\begin{split} r(k) &= u(k) + r(k\text{-}1) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}4) \\ C(4k) &= r(k) + r(k\text{-}1) + r(k\text{-}3) + r(k\text{-}4) \\ C(4k+1) &= r(k) + r(k\text{-}2) + r(k\text{-}4) \\ C(4k+2) &= u(k) \\ C(4k+3) &= u(k) \end{split} \qquad \text{for } k = 0, 1, ..., 48; r(k) = 0 \text{ for } k < 0 \end{split}$$

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(4k) &= r(k) + r(k-1) + r(k-3) + r(k-4) \\ C(4k+1) &= r(k) + r(k-2) + r(k-4) \\ C(4k+2) &= r(k-1) + r(k-2) + r(k-3) + r(k-4) \\ C(4k+3) &= r(k-1) + r(k-2) + r(k-3) + r(k-4) & \text{for } k = 49, 50, ..., 52 \end{split}$$

This block of data is moved to the coded data (c) as:

$$c(k+16) = C(k)$$
 for  $k = 0, 1, ..., 211$ 

#### 3.10.9.3 Interleaving

The interleaving is done as specified for the TCH/HS in subclause 3.2.3.

# 3.10.9.4 Mapping on a Burst

The mapping is done as specified for the TCH/HS in subclause 3.2.4.

# 3.11 Data channel for ECSD at full rate, 29.0 kbit/s radio interface rate (28.8 kbit/s services (E-TCH/F28.8))

The definition of a 28.8 kbit/s radio interface rate data flow for data services is given in 3GPP TS 44.021.

# 3.11.1 Interface with user unit

The user unit delivers to the encoder a bit stream organized in blocks of 580 information bits (data frames) every 20 ms.

# 3.11.2 Block code

# 3.11.2.1 Repetition bits

To match to RS alphabet 4 extra data bits are added to the end of each block of 580 bits: d(k)=0, k=580,...583.

# 3.11.2.2 Reed Solomon encoder

The block of 584 information bits is encoded by shortened systematic Reed Solomon (RS) code over Galois field  $GF(2^8)$ . The Galois field  $GF(2^8)$  is built as an extension of GF(2). The characteristic of  $GF(2^8)$  is equal to 2.

The code used is systematic RS<sub>8</sub>(85,73), which is shortened systematic RS<sub>8</sub>(255,243) code over GF( $2^8$ ) with the primitive polynomial  $p(x)=x^8+x^4+x^3+x^2+1$ . The primitive element a is the root of the primitive polynomial, i.e.

$$a^8 = a^4 + a^3 + a^2 + 1$$
.

Generator polynomial for RS<sub>8</sub>(255,243) code is:

 $g(x) = \prod_{i=1}^{n} (x - a^{i+122})$ ; that results in symmetrical form for the generator polynomial with coefficients given in decimal notation

$$g(x) = x^{12} + 18x^{11} + 157x^{10} + 162x^{9} + 134x^{8} + 157x^{7} + 253x^{6} + 157x^{5} + 134x^{4} + 162x^{3} + 157x^{2} + 18x + 157x^{2$$

where binary presentation of polynomial coefficients in GF(256) is  $\{a^7, a^6, a^5, a^4, a^3, a^2, a, 1\}$ .

Specifically, decimal, power and polynomial presentations for the generator polynomial coefficients are the following:

$$x^{11}$$
:  $18 = a^{224} = a^4 + a^4$   
 $x^{10}$ :  $157 = a^{32} = a^7 + a^4$ 

x<sup>12</sup>: 1

$$x^{10}$$
: 157 =  $a^{32}$  =  $a^7 + a^4 + a^3 + a^2 + 1$ 

$$x^9$$
:  $162 = a^{209} = a^7 + a^5 + a$ 

$$x^8$$
:  $134 = a^{99} = a^7 + a^2 + a$ 

$$x^7$$
: 157 =  $a^{32}$  =  $a^7 + a^4 + a^3 + a^2 + 1$ 

$$x^6$$
:  $253 = a^{80} = a^7 + a^6 + a^5 + a^4 + a^3 + a^2 + 1$ 

$$x^5$$
:  $157 = a^{32} = a^7 + a^4 + a^3 + a^2 + 1$ 

$$x^4$$
:  $134 = a^{99} = a^7 + a^2 + a$ 

$$x^3$$
:  $162 = a^{209} = a^7 + a^5 + a^8$ 

$$x^2$$
: 157 =  $a^{32}$  =  $a^7 + a^4 + a^3 + a^2 + 1$ 

$$x^1$$
:  $18 = a^{224} = a^4 + a$ 

$$x^0$$
:  $1 = a^{255} = 1$ 

The RS encoding is performed in the following three steps:

#### a) Bit to symbol conversion

The information bits  $\{d(0),d(1),...,d(583)\}$  are converted into 73 information 8-bit symbols  $\{D(0),...,D(72)\}$  as the following:

$$D(k) = 128d(8k+7) + 64d(8k+6) + 32d(8k+5) + 16d(8k+4) + 8d(8k+3) + 4d(8k+2) + 2d(8k+1) + d(8k) + 2d(8k+1) +$$

Resulting 8-bit symbols are presented as

$$D(k) = \{d(8k+7), d(8k+6), d(8k+5), d(8k+4), d(8k+3), d(8k+2), d(8k+1), d(8k)\}$$
 for  $k = 0,1,...,72$ 

where d(8k+7),...,d(8k) are ordered from the most significant bit (MSB) to the less significant bit (LSB).

The polynomial representation of a single information symbol over  $GF(2^8)$  in terms of a is given by

$$D_a(k) = a^7 d(8k+7) + a^6 d(8k+6) + a^5 d(8k+5) + a^4 d(8k+4) + a^3 d(8k+3) + a^2 d(8k+2) + ad(8k+1) + d(8k)$$

# b) Encoding

The information symbols D(0)...D(72) are encoded by shortened systematic  $RS_8(85,73)$  code with output symbols U(0)...U(84) ordered as

$$U(k)=D(k)$$
 for  $k=0,1,...72$ ;  $U(k)=R(k)$  for  $k=73,74,...,84$ ;

where R(k) are parity check symbols added by RS<sub>8</sub>(85,73) encoder.

Information symbols are ordered in the descending polynomial order such that  $D_a$  (72) corresponds to the lowest

degree term of  $D(x) = D_a(72) + D_a(71)x + ... + D_a(1)x^{71} + D_a(0)x^{72}$ , where D(x) is the polynomial representation of

information symbols  $\{D(0),D(1),...,D(72)\}$  over Galois field.

Parity check symbols in polynomial representation over Galois field are ordered in the descending polynomial order such that  $R_a(84)$  corresponds to the lowest degree of  $R(x)=R_a(84)+R_a(83)x+...+R_a(74)x^{10}+R_a(73)x^{11}$ . The parity check symbols are calculated as R(x)= remainder  $[x^{12}D(x)/g(x)]$ , and  $U(x)=R(x)+x^{12}D(x)$ , i.e.,

$$U_a(k) = D_a(k)$$
 for k=0,1,...72;  $U_a(k) = R_a(k)$  for k=73,74,...,84.

The encoding operation with the shortened RS<sub>8</sub>(85,73) code may be presented as the following:

- Expanding 73 information symbols to the block of 243 symbols by adding 170 dump (zero) symbols
- Encoding 243 symbols by systematic RS<sub>8</sub>(255,243) encoder with outer block of 255 symbols
- Removing 170 dump symbols, resulting in the output block of 85 symbols.

#### c) Symbol to bit conversion

The output symbols  $\{U_a(0),...,U_a(84)\}$  are converted back into symbols  $\{U(0),...,U(84)\}$  and then back into binary form with LSB coming out first, resulting in the block of 680 bits  $\{u(0),...u(679)\}$ .

# 3.11.3 Convolutional encoder

# 3.11.3.1 Tailing bits for a data frame

Before convolutional encoding 6 tail bits  $\{u(k)=0, k=680, \dots 685\}$  are added to the end of each data block .

# 3.11.3.2 Convolutional encoding for a data frame

This block of 686 bits  $\{u(0),...,u(685)\}$  is encoded with the ½ rate convolutional code defined by the following polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

resulting in 1372 coded bits  $\{c(0), c(1),..., c(1371)\}$  with

$$c(2k)=u(k)+u(k-2)+u(k-3)+u(k-5)+u(k-6);$$

$$c(2k+1) = u(k) + u(k-1) + \ u(k-2) + \ u(k-3) + \ u(k-6) \ \ \text{for} \ k = 0, 1, ..., 685; \ \ u(k) = 0 \ \text{for} \ k < 0$$

The code is punctured in such a way that the following 4 coded bits:

The result is a block of 1368 coded bits,  $\{c(0), c(1), ..., c(1367)\}$ .

# 3.11.4 Interleaving

The interleaving scheme is presented below.

The coded bits are reordered and interleaved according to the following rule:

$$\begin{split} i(B,j) &= c(n,k), \quad \text{ for } k = 0,1,...,1367 \\ n &= 0,1,...,N,N+1,... \\ B &= B_0 + 4n + (k \text{ mod } 19) + (k \text{ div } 342) \end{split}$$

$$i = (k \mod 19) + 19(k \mod 18)$$

The result of the interleaving is a distribution of the reordered 342 bit of a given data block, n = N, over 19 blocks, 18 bits equally distributed in each block, in a diagonal way over consecutive blocks.

Or in other words the interleaving is a distribution of the encoded, reordered 1368 bits from four given input data blocks, which taken together give n = N, over 22 bursts, 18 bits equally distributed in the first and  $22^{nd}$  bursts, 36 bits distributed in the second and  $21^{st}$  bursts, 54 bits distributed in the third and  $20^{th}$  bursts and 72 bits distributed in the other 16 bursts.

The block of coded data is interleaved "diagonal", where a new block of coded data starts with every fourth burst and is distributed over 22 bursts.

# 3.11.5 Mapping on a Burst

Before mapping on a burst the interleaved bits  $\{i(0)...i(1367)\}$  are converted into 3-bit symbols  $\{I(0),I(1),...,I(455)\}$  according to Table 1 in 3GPP TS 45.004, the symbol I(k) depends on i(3k+2), i(3k+1) and i(3k) for k=0,1,...,455.

The E-IACCH message delivered to the encoder on every 20ms has a fixed size of 3 information bits  $\{im(0), im(1), im(2)\}$ . The contents of the bits are defined in 3GPP TS 45.008 for both uplink and downlink.

The E-IACCH information bits  $\{im(n,0),im(n,1),im(n,2)\}$  are coded into 24 bits ib(B,k),  $B_0 + 4n \le B < B_0 + 4n + 4$ , k = 0,1,...5 according to the following table:

im(n,0),im(n,1),im(n,2)	<u>ib(</u> B <sub>0</sub> +4n,0 <u>),,ib(</u> B <sub>0</sub> +4n,5 <u>),,</u>	
	<u>ib(</u> B <sub>0</sub> +4n+3,0),ib(B <sub>0</sub> +4n+3,5)	
000	000000 000000 000000 000000	
<u>001</u>	<u>001111 110100 100101 110100</u>	
<u>010</u>	<u>011100 010111 111001 100011</u>	
<u>011</u>	<u>010011 100011 011100 010111</u>	
<u>100</u>	<u>100110 011001 110110 001101</u>	
<u>101</u>	<u>101001 101101 010011 111001</u>	
<u>110</u>	<u>111010 001110 001111 101110</u>	
<u>111</u>	<u>110101 111010 101010 011010</u>	

Before mapping on a burst the E-IACCH bits  $\{ib(B,0)...ib(B,5)\}$  are converted into 3-bit symbols  $\{HL(B),HU(B)\}$  according to Table 1 in 3GPP TS 45.004. The symbol HL(B) depends on ib(B,2), ib(B,1) and ib(B,0) and

the symbol HU(B) on ib(B,5), ib(B,4) and ib(B,3).

The mapping is given by the rule:

$$E(B,j) = I(B,j)$$
 and  $E(B,59+j) = I(B,57+j)$  for  $j = 0,1,...,56$ 

and

$$E(B,57) = HL(B)$$
 and  $E(B,58) = HU(B)$ .

The two symbols, labelled HL(B) and HU(B) on burst number B are flags used for E-IACCH.

# 3.12 Data channel for ECSD at full rate, 32.0 kbit/s radio interface rate (32.0 kbit/s services (E-TCH/F32.0))

The definition of a 32.0 kbit/s radio interface rate data flow for data services is given in 3GPP TS 44.021.

# 3.12.1 Interface with user unit

The user unit delivers to the encoder a bit stream organized in blocks of 640 information bits (data frames) every 20 ms.

#### 3.12.2 Void

# 3.12.3 Convolutional encoder

 $G4 = 1 + D^2 + D^3 + D^5 + D^6$ 

### 3.12.3.1 Tailing bits for a data frame

Before convolutional encoding 6 tail bits  $\{d(k)=0, k=640,...,645\}$  are added to the end of each data block.

# 3.12.3.2 Convolutional encoding for a data frame

This block of 646 bits  $\{d(0),...,d(645)\}$  is encoded with the 1/3 rate convolutional code (the same code as for MCS-1) defined by the following polynomials:

$$\begin{split} G7 &= 1 + D + D^2 + D^3 + D^6 \\ G5 &= 1 + D + D^4 + D^6 \\ \end{split}$$
 resulting in 1938 coded bits  $\{c(0), c(1), ..., c(1937)\}$  with 
$$c(3k) = d(k) + d(k-2) + d(k-3) + d(k-5) + d(k-6) \; ;$$
 
$$c(3k+1) = d(k) + d(k-1) + d(k-2) + d(k-3) + d(k-6) \; ;$$
 
$$c(3k+2) = d(k) + d(k-1) + d(k-4) + d(k-6) \; ;$$
 for  $k = 0,1, ..., 645 \; ;$   $d(k) = 0$  for  $k < 1$ 

The code is punctured using the rate matching algorithm of clause 6.2.3, such that the following 546 coded bits:

```
c(0), c(3), c(7), c(10), c(14), c(17), c(21), c(24), c(28), c(31), c(35), c(39), c(42), c(46), c(49), c(53), c(56), c(60),
     c(63), c(67), c(70), c(74), c(78), c(81), c(85), c(88), c(92), c(95), c(99), c(102), c(106), c(110), c(113), c(117),
     c(120), c(124), c(127), c(131), c(134), c(138), c(141), c(145), c(149), c(152), c(156), c(159), c(163), c(166),
     c(170), c(173), c(177), c(181), c(184), c(188), c(191), c(195), c(198), c(202), c(205), c(209), c(212), c(216),
     c(220), c(223), c(227), c(230), c(234), c(237), c(241), c(244), c(248), c(252), c(255), c(259), c(262), c(266),
     c(269), c(273), c(276), c(280), c(283), c(287), c(291), c(294), c(298), c(301), c(305), c(308), c(312), c(315),
     c(319), c(323), c(326), c(330), c(333), c(337), c(340), c(344), c(347), c(351), c(354), c(358), c(362), c(365),
     c(369), c(372), c(376), c(379), c(383), c(386), c(390), c(393), c(397), c(401), c(404), c(408), c(411), c(415),
     c(418), c(422), c(425), c(429), c(433), c(436), c(440), c(443), c(447), c(450), c(454), c(457), c(461), c(464),
     c(468), c(472), c(475), c(479), c(482), c(486), c(489), c(493), c(496), c(500), c(504), c(507), c(511), c(514),
     c(518), c(521), c(525), c(528), c(532), c(535), c(539), c(543), c(546), c(550), c(553), c(557), c(560), c(564),
     c(567), c(571), c(575), c(578), c(582), c(585), c(589), c(592), c(596), c(599), c(603), c(606), c(610), c(614),
     c(617), c(621), c(624), c(628), c(631), c(635), c(638), c(642), c(646), c(649), c(653), c(656), c(660), c(663), c(663), c(664), c(664), c(664), c(664), c(664), c(664), c(666), c(66
     c(667), c(670), c(674), c(677), c(681), c(685), c(688), c(692), c(695), c(699), c(702), c(706), c(709), c(713),
     c(716), c(720), c(724), c(727), c(731), c(734), c(738), c(741), c(745), c(748), c(752), c(756), c(759), c(763),
     c(766), c(770), c(773), c(777), c(780), c(784), c(787), c(791), c(795), c(798), c(802), c(805), c(809), c(812),
     c(816), c(819), c(823), c(827), c(830), c(834), c(837), c(841), c(844), c(848), c(851), c(855), c(858), c(862),
     c(866), c(869), c(873), c(876), c(880), c(883), c(887), c(890), c(894), c(898), c(901), c(905), c(908), c(912),
     c(915), c(919), c(922), c(926), c(929), c(933), c(937), c(940), c(944), c(947), c(951), c(954), c(958), c(961),
     c(965), c(969), c(972), c(976), c(979), c(983), c(986), c(990), c(993), c(997), c(1000), c(1004), c(1008), c(1011),
     c(1015), c(1018), c(1022), c(1025), c(1029), c(1032), c(1036), c(1039), c(1043), c(1047), c(1050), c(1054),
     c(1057), c(1061), c(1064), c(1068), c(1071), c(1075), c(1079), c(1082), c(1086), c(1089), c(1093), c(1096),
     c(1100), c(1103), c(1107), c(1110), c(1114), c(1118), c(1121), c(1125), c(1128), c(1132), c(1135), c(1139),
     c(1142), c(1146), c(1150), c(1153), c(1157), c(1160), c(1164), c(1167), c(1171), c(1174), c(1178), c(1181),
     c(1185), c(1189), c(1192), c(1196), c(1199), c(1203), c(1206), c(1210), c(1213), c(1217), c(1221), c(1224),
     c(1228), c(1231), c(1235), c(1238), c(1242), c(1245), c(1249), c(1252), c(1256), c(1260), c(1263), c(1267),
     c(1270), c(1274), c(1277), c(1281), c(1284), c(1288), c(1292), c(1295), c(1299), c(1302), c(1306), c(1309),
     c(1313), c(1316), c(1320), c(1323), c(1327), c(1331), c(1334), c(1338), c(1341), c(1345), c(1348), c(1352),
     c(1355), c(1359), c(1362), c(1366), c(1370), c(1373), c(1377), c(1380), c(1384), c(1387), c(1391), c(1394),
     c(1398), c(1402), c(1405), c(1409), c(1412), c(1416), c(1419), c(1423), c(1426), c(1430), c(1433), c(1437),
     c(1441), c(1444), c(1448), c(1451), c(1455), c(1458), c(1462), c(1465), c(1469), c(1473), c(1476), c(1480),
```

```
c(1483), c(1487), c(1490), c(1494), c(1497), c(1501), c(1504), c(1508), c(1512), c(1515), c(1519), c(1522), c(1526), c(1529), c(1533), c(1536), c(1540), c(1544), c(1547), c(1551), c(1554), c(1558), c(1561), c(1565), c(1568), c(1572), c(1575), c(1579), c(1583), c(1586), c(1590), c(1593), c(1597), c(1600), c(1604), c(1607), c(1611), c(1615), c(1618), c(1622), c(1625), c(1629), c(1632), c(1636), c(1639), c(1639), c(1643), c(1646), c(1650), c(1654), c(1657), c(1661), c(1664), c(1668), c(1671), c(1675), c(1678), c(1682), c(1685), c(1689), c(1693), c(1696), c(1700), c(1703), c(1707), c(1710), c(1714), c(1717), c(1721), c(1725), c(1728), c(1728), c(1732), c(1738), c(1739), c(1742), c(1746), c(1749), c(1753), c(1756), c(1760), c(1764), c(1767), c(1771), c(1771), c(1774), c(1778), c(1781), c(1785), c(1831), c(1835), c(1838), c(1842), c(1845), c(1849), c(1852), c(1856), c(1859), c(1863), c(1867), c(1870), c(1874), c(1877), c(1881), c(1884), c(1888), c(1891), c(1895), c(1898), c(1902), c(1906), c(1909), c(1913), c(1916), c(1920), c(1923), c(1927), c(1930), c(1934)
```

are not transmitted.

The result is a block of 1392 coded bits,  $\{c(0),c(1),...,c(1391)\}$ .

# 3.12.4 Interleaving

The coded bits are reordered and interleaved according to the following rule:

```
i(B,j) = c(n,k), for k = 0,1,...,1391

n = 0,1,...,N,N+1,...

B = B_0 + 4n + (k \mod 12)

j = 3*[(49*(k+int(k/348)) \mod 116) + int[(k \mod 12)/4]]
```

The result of the interleaving is a distribution of the reordered 348 bits of a given data block, n = N, over 12 blocks, 29 bits equally distributed in each block. The block of coded data is interleaved "diagonal", where a new block of coded data starts with every fourth burst and is distributed over 12 bursts.

# 3.12.5 Mapping on a Burst

The mapping is given by the rule:

$$e(B,j) = i(B,j)$$
 for  $j = 0,1,...,347$ 

NOTE: No stealing flags are used.

# 3.13 Data channel for ECSD at full rate, 43.5 kbit/s radio interface rate (43.2 kbit/s services (E-TCH/F43.2))

The definition of a 43.5 kbit/s radio interface rate data flow for data services is given in 3GPP TS 44.021.

#### 3.13.1 Interface with user unit

The user unit delivers to the encoder a bit stream organized in blocks of 870 information bits (data frames) every 20 ms.

# 3.13.2 Convolutional encoder

# 3.13.2.1 Tailing bits for a data frame

Before convolutional encoding 6 tail bits  $\{d(k)=0, k=870, \dots 875\}$  are added to the end of each data block .

#### 3.13.2.2 Convolutional encoding for a data frame

This block of 876 bits  $\{d(0),...,d(875)\}$  is encoded with the ½ rate convolutional code defined by the following polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

resulting in 1752 coded bits  $\{c(0), c(1),..., c(1751)\}$  with

$$c(2k)=d(k)+d(k-2)+d(k-3)+d(k-5)+d(k-6);$$

$$c(2k+1)=d(k)+d(k-1)+d(k-2)+d(k-3)+d(k-6)$$
 for  $k=0,1,...,875$ ;  $u(k)=0$  for  $k<0$ 

The code is punctured in such a way that the following 384 coded bits:

$$c(2+8(k-1))$$
 for  $k=1:219$ ;  $c(4+16(k-1))$  for  $k=1:110$ ;  $c(6+32(k-1))$  for  $k=1:55$ 

are not transmitted.

The result is a block of 1368 coded bits,  $\{c(0), c(1), ..., c(1367)\}$ .

# 3.13.3 Interleaving

The interleaving is done as specified for E-TCH/F28.8 in subclause 3.11.4.

# 3.13.4 Mapping on a Burst

The mapping is done as specified for E-TCH/F28.8 in subclause 3.11.5.

# 3.14 Wideband Adaptive multi rate speech channel at full rate (TCH/WFS)

This section describes the coding for the different frame formats used for TCH/WFS. The formats used are (in the order they are described):

SID\_UPDATE Used to convey comfort noise parameters during DTX

SID\_FIRST Marker to define end of speech, start of DTX

ONSET Used to signal the Codec mode for the first speech frame after DTX

SPEECH Speech frames

RATSCCH Frames used to convey RATSCCH messages

In this chapter, sub chapters 3.14.1 to 3.14.4 describe the channel coding for the different formats listed above.

Common to all the formats is that in-band information is conveyed, the coding for the in-band channel is described in the table below.

Identifier (defined in 3GPP TS 45.009)	Received in-band data id(1), id(0)	Encoded in-band data for SID and RATSCCH frames ic(15),, ic(0)	Encoded in-band data for speech frames ic(7),, ic(0)
CODEC_MODE_1	00	0101001100001111	00000000
CODEC_MODE_2	01	0011111010111000	10111010
CODEC_MODE_3	10	1000100001100011	01011101
CODEC_MODE_4	11	1110010111010100	11100111

# 3.14.1 SID\_UPDATE

The SID\_UPDATE frames are handled as specified for the TCH/AFS in subclause 3.9.1.

# 3.14.2 SID\_FIRST

The SID\_FIRST frames are handled as specified for the TCH/AFS in subclause 3.9.2.

# 3.14.3 ONSET

The Onset frames are handled as specified for the TCH/AFS in subclause 3.9.3.

# 3.14.4 SPEECH

The speech coder delivers to the channel encoder a sequence of blocks of data. One block of data corresponds to one speech frame and the block length is different in each of the seven channel codec modes. Adjoining each block of data is information of the channel codec mode to use when encoding the block. Also delivered is the in-band data id(0,1) representing Mode Indication or Mode Command/Mode Request depending on the current frame number.

# 3.14.4.1 Coding of the in-band data

The two input in-band bits (id(0,1)) are coded to eight coded in-band bits (ic(0..7)).

The encoded in-band bits are moved to the coded bits, c, as

$$c(k) = ic(k)$$
 for  $k = 0, 1, ..., 7$ .

# 3.14.4.2 Ordering according to subjective importance

The bits delivered by the speech encoder,  $\{s(1), s(2), ..., s(K_s)\}$ , are rearranged according to subjective importance before channel coding. Tables 16 to 18 define the correct rearrangement for the speech codec modes 12.65 kbit/s, 8.85 kbit/s and 6.60 kbit/s, respectively. In the tables speech codec parameters are numbered in the order they are delivered by the corresponding speech encoder according to 3GPP TS 26.190 and the rearranged bits are labelled  $\{d(0),d(1),...,d(K_d-1)\}$ , defined in the order of decreasing importance. Index  $K_d$  refers to the number of bits delivered by the speech encoder, see below:

Codec mode	Number of speech bits delivered per block (K <sub>d</sub> )
TCH/WFS12.65	253
TCH/WFS8.85	177
TCH/WFS6.60	132

The ordering algorithm is in pseudo code as:

for 
$$j = 0$$
 to  $K_{d-1}$   $d(j) := s(table(j)+1);$  where  $table(j)$  is read line by line left to right

The rearranged bits are further divided into two different classes to perform unequal error protection for different bits according to subjective importance.

The protection classes are:

1a - Data protected with the CRC and the convolution code.

1b - Data protected with the convolution code.

The number of class 1 (sum of class 1a and 1b), class 1a and class 1b bits for each codec mode is shown below:

Codec mode	Number of speech bits delivered per block	Number of class 1 bits per block	Number of class 1a bits per block	Number of class 1b bits per block
TCH/WFS12.65	253	253	72	181
TCH/WFS8.85	177	177	64	113
TCH/WFS6.60	132	132	54	78

# 3.14.4.3 Parity for speech frames

The basic parameters for each codec mode for the first encoding step are shown below:

Codec mode	Number of class 1 bits (K <sub>d1</sub> )	CRC Protected bits (K <sub>d1a</sub> )	CRC bits	Number of bits after first encoding step (K <sub>u</sub> = K <sub>d</sub> + 8/6)
TCH/WFS12.65	253	72	6	259
TCH/WFS8.85	177	64	6	183
TCH/WFS6.60	132	54	8	140

A 8-bit or 6-bit CRC is used for error-detection. These parity bits are generated by the cyclic generator polynomial:  $g8(D) = D^8 + D^4 + D^3 + D^2 + 1$  or  $g6(D) = D^6 + D^5 + D^3 + D^2 + D^1 + 1$  respectively from the first  $K_{d1a}$  bits of class 1, where  $K_{d1a}$  refers to number of bits in protection class 1a as shown above for each codec mode. The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

$$d(0)D(K_{d1a}+n-1)+d(1)D(K_{d1a}+n-2)+...+d(K_{d1a}-1)D(n)+p(0)D^{\left(n-1\right)}+...+p(n-2)D+p(n-1)$$

where p(0), p(1) ... p(n) are the parity bits (n=8 or 6), when divided by g8(D) or g6(D), yields a remainder equal to:

$$1+D+...+D^{n-1}$$
.

The information and parity bits are merged:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., K_{d1a}-1$ 

$$u(k) = p(k - K_{d1a}) \hspace{1cm} \text{for } k = K_{d1a}, \, K_{d1a} + 1, \, ..., \, K_{d1a} + \, (n - 1)$$

$$u(k) = d(k\text{-}n) \hspace{1cm} \text{for } k = K_{d1a} \text{+} n, \; K_{d1a} \text{+} n \text{+} 1, \; ..., \; K_u \text{-} 1$$

Thus, after the first encoding step u(k) will be defined by the following contents for each codec mode:

# TCH/WFS12.65:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 71$ 

$$u(k) = p(k-72)$$
 for  $k = 72, 73, ..., 77$ 

$$u(k) = d(k-6)$$
 for  $k = 78, 79, ..., 258$ 

### **TCH/WFS8.85**:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 63$   
 $u(k) = p(k-64)$  for  $k = 64, 65, ..., 69$ 

# u(k) = d(k-6) for k = 70, 71, ..., 182

# **TCH/WFS6.60**:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 53$ 

$$u(k) = p(k-54)$$
 for  $k = 54, 55, ..., 61$   
 $u(k) = d(k-8)$  for  $k = 62, 63, ..., 139$ 

#### 3.14.4.4 Convolutional encoder

The bits from the first encoding step (u(k)) are encoded with the recursive systematic convolutional codes as summarised below. The number of output bits after puncturing is 448 for all codec modes.

Codec Mode	Rate	Number of input bits to conv. coder	Number of output bits from conv. Coder	Number Of punctured bits
TCH/WFS12.65	1/2	259	526	78
TCH/WFS8.85	1/3	183	561	113
TCH/WFS6.60	1/4	140	576	128

Below the coding for each codec mode is specified in detail. The puncturing for each mode is designed to give an even protection of the class 1A bits while the protection within class 1B is not equal to reflect the individual error sensitivity of the class 1B bits.

#### TCH/WFS12.65:

The block of 259 bits  $\{u(0)...u(258)\}$  is encoded with the ½ rate convolutional code defined by polynomials:

$$G0/G0 = 1$$

$$G1/G0 = 1 + D + D^3 + D^4 / 1 + D^3 + D^4$$

resulting in 518 coded bits,  $\{C(0)...C(517)\}\$  defined by:

$$r(k) = u(k) + r(k-3) + r(k-4)$$

$$C(2k) = u(k)$$

$$C(2k+1) = r(k)+r(k-1)+r(k-3)+r(k-4)$$
 for  $k = 0, 1, ..., 258$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$r(k) = 0$$

$$C(2k) = r(k-3) + r(k-4)$$

$$C(2k+1) = r(k)+r(k-1)+r(k-3)+r(k-4)$$
 for  $k = 259, 260, ..., 262$ 

The code is punctured in such a way that the following 78 coded bits:

```
C(1), C(17), C(33), C(191), C(207), C(223), C(239), C(251), C(253), C(255), C(267), C(269), C(271), C(283), C(285), C(287), C(297), C(299), C(301), C(303), C(313), C(315), C(317), C(319), C(329), C(331), C(333), C(335), C(345), C(347), C(349), C(351), C(361), C(363), C(365), C(367), C(377), C(379), C(381), C(383), C(393), C(395), C(397), C(399), C(409), C(411), C(413), C(415), C(425), C(427), C(429), C(431), C(441), C(443), C(445), C(447), C(457), C(459), C(461), C(463), C(473), C(475), C(477), C(479), C(487), C(489), C(491), C(493), C(495), C(503), C(505), C(507), C(509), C(511), C(519), C(521), C(523), C(525)
```

are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the in-band bits in c as

$$c(8+k) = P(k)$$
 for  $k = 0, 1, ..., 447$ .

#### TCH/WFS8.85:

The block of 183 bits  $\{u(0)...u(182)\}$  is encoded with the 1/3 rate convolutional code defined by the following polynomials:

$$G1/G1 = 1$$

$$G2/G1 = 1 + D^2 + D^4 / 1 + D + D^3 + D^4$$

$$G3/G1 = 1 + D + D^2 + D^3 + D^4 / 1 + D + D^3 + D^4$$

resulting in 549 coded bits,  $\{C(0)...C(548)\}\$  defined by:

$$r(k) = u(k) + r(k-1) + r(k-3) + r(k-4)$$

$$C(3k) = u(k)$$

$$C(3k+1) = r(k)+r(k-2) + r(k-4)$$

$$C(3k+2) = r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4)$$
 for  $k = 0, 1, ..., 182$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$r(k) = 0$$

$$C(3k) = r(k-1) + r(k-3) + r(k-4)$$

$$C(3k+1) = r(k)+r(k-2) + r(k-4)$$

$$C(3k+2) = r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4)$$
 for  $k = 183, 184, ..., 186$ 

The code is punctured in such a way that the following 113 coded bits:

are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the in-band bits in c as

$$c(8+k) = P(k)$$
 for  $k = 0, 1, ..., 447$ .

#### **TCH/WFS6.60**:

The block of 140 bits  $\{u(0)...u(139)\}$  is encoded with the  $\frac{1}{4}$  rate convolutional code defined by the following polynomials:

$$G1/G1 = 1$$

$$G2/G1 = 1 + D^2 + D^4 / 1 + D + D^3 + D^4$$

$$G3/G1 = 1 + D + D^2 + D^3 + D^4 / 1 + D + D^3 + D^4$$

$$G1/G1 = 1$$

resulting in 560 coded bits,  $\{C(0)...C(559)\}\$  defined by:

$$r(k) = u(k) + r(k-1) + r(k-3) + r(k-4)$$

$$C(4k) = u(k)$$

$$C(4k+1) = r(k) + r(k-2) + r(k-4)$$

$$C(4k+2) = r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4)$$

$$C(4k+3) = u(k)$$
 for  $k = 0, 1, ..., 139$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(4k) &= r(k-1) + r(k-3) + r(k-4) \\ C(4k+1) &= r(k) + r(k-2) + r(k-4) \\ C(4k+2) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) \\ C(4k+3) &= r(k-1) + r(k-3) + r(k-4) \qquad \text{for } k = 140, 141, ..., 143 \end{split}$$

The code is punctured in such a way that the following 128 coded bits:

```
 \begin{array}{l} C(3), C(7), C(11), C(15), C(27), C(31), C(35), C(39), C(51), C(55), C(59), C(75), C(79), C(83), C(99), \\ C(103), C(107), C(123), C(127), C(131), C(147), C(151), C(155), C(171), C(175), C(179), C(195), C(199), \\ C(203), C(219), C(223), C(227), C(231), C(243), C(247), C(251), C(255), C(267), C(271), C(275), C(279), \\ C(283), C(291), C(295), C(299), C(303), C(307), C(311), C(315), C(319), C(323), C(327), C(331), C(335), \\ C(339), C(343), C(347), C(351), C(355), C(359), C(363), C(367), C(371), C(375), C(379), C(382), C(383), \\ C(387), C(391), C(395), C(399), C(403), C(406), C(407), C(411), C(415), C(419), C(423), C(427), C(430), \\ C(431), C(435), C(439), C(443), C(447), C(451), C(454), C(455), C(459), C(463), C(467), C(471), C(475), \\ C(478), C(479), C(483), C(487), C(491), C(495), C(499), C(502), C(503), C(507), C(511), C(515), C(519), \\ C(523), C(526), C(527), C(531), C(535), C(539), C(543), C(547), C(550), C(551), C(555), C(559), C(562), \\ C(563), C(566), C(567), C(569), C(570), C(571), C(573), C(574), C(575) \end{array}
```

are not transmitted. The result is a block of 448 coded and punctured bits, P(0)...P(447) which are appended to the inband bits in c as

$$c(8+k) = P(k)$$
 for  $k = 0, 1, ..., 447$ .

#### 3.14.4.5 Interleaving

The interleaving is done as specified for the TCH/FS in subclause 3.1.3.

#### 3.14.4.6 Mapping on a Burst

The mapping is done as specified for the TCH/FS in subclause 3.1.4.

# 3.14.5 RATSCCH

The RATSCCH frames are handled as specified for TCH/AFS in subclause 3.9.5.

# 3.15 Adaptive multi rate speech channel at 8-PSK half rate (O-TCH/AHS)

This section describes the coding for the different frame formats used for O-TCH/AHS. The formats used are (in the order they are described):

SID_UPDATE	Used to convey comfort noise parameters during DTX
SID_UPDATE_INH	Used to inhibit the second part of a SID_UPDATE frame if there is a speech onset
SID_FIRST_P1	First part of marker to define end of speech, start of DTX
SID_FIRST_P2	Second part of marker to define end of speech, start of DTX
SID_FIRST_INH	Used to inhibit the second part of a SID_FIRST_P1 frame if there is a speech onset
ONSET	Used to signal the Codec mode for the first speech frame after DTX

SPEECH Speech frames

RATSCCH\_MARKER Marker to identify RATSCCH frames

RATSCCH DATA Frame that conveys the actual RATSCCH message

In this chapter, sub chapters 3.15.1 to 3.15.9 describe the channel coding for the different formats listed above.

Common to all the formats is that in-band information is conveyed, the coding for the in-band channel is described in the table below.

Identifier	Received in-band	Encoded in-band data for SID	Encoded in-band data for
(defined in 3GPP	data	and RATSCCH frames	speech frames
45.009)	id(1), id(0)	ic(15),, ic(0)	ic(11),, ic(0)
CODEC_MODE_1	00	0101001100001111	00000000000
CODEC_MODE_2	01	0011111010111000	110110101110
CODEC_MODE_3	10	1000100001100011	101101110101
CODEC_MODE_4	11	1110010111010100	011011011011

# 3.15.1 SID\_UPDATE

The speech encoder delivers 35 bits of comfort noise parameters. Also delivered is two in-band channels, id0(0,1) and id1(0,1), id0 corresponding to Mode Commands/Mode Requests and id1 to Mode Indication. The general coding is as: the two in-band data channels are coded to 16 bits each, a 14-bit CRC is added to the 35 CN bits which are then coded by a rate ¼ RSC coder to 212 bits. A 212 bit identification field is added thereby giving a total size of 456 bits. Finally each bit is repeated 3 times and then converted into 3-bit symbols giving a total size of 456 symbols. These 456 symbols are block interleaved over 4 bursts.

### 3.15.1.1 Coding of in-band data

The coding of in-band data is done as specified for the SID\_UPDATE frame at half rate in subclause 3.10.1.1.

#### 3.15.1.2 Parity and convolutional encoding for the comfort noise parameters

The parity and convolutional encoding for the comfort noise parameters are done as specified for the SID\_UPDATE frame at half rate in subclause 3.10.1.2.

#### 3.15.1.3 Identification marker

The identification marker is constructed as specified for the SID\_UPDATE frame at half rate in subclause 3.10.1.3.

#### 3.15.1.4 Repetition

The coded bits (c) are repeated according to the following rule:

$$c'(3k+2) = c'(3k+1) = c'(3k) = c(k)$$
 for  $k=0,...,455$ 

#### 3.15.1.5 Interleaving

Before interleaving the coded bits  $\{c'(0) \dots c'(1367)\}$  are converted into 3-bit symbols  $\{C(0) \dots C(455)\}$  according to table 1 in 3GPP TS 45.004, the symbol C(k) depends on c'(3k+2), c'(3k+1) and c'(3k) for  $k=0,1,\dots,455$ .

The interleaving is done as specified for the SID\_UPDATE frame at half rate in subclause 3.10.1.4. The difference is that the interleaving is done by symbols instead of single bits, reusing the existing interleaving tables.

# 3.15.1.6 Mapping on a Burst

The mapping is done as specified for the SID\_UPDATE frame at half rate in subclause 3.10.1.5 with exception that it is done by symbols instead of single bits.

# 3.15.2 SID\_UPDATE\_INH

This special frame is used when the first 2 burst of a SID\_UPDATE frame have been transmitted but the second two bursts cannot be transmitted due to a speech frame. The general coding is as: the in-band data (Note that this must be the same Mode Indication bits as id1(0,1) for the SID\_UPDATE frame that is being inhibited) is encoded, a marker that is the opposite of the SID\_UPDATE marker is appended and the data is interleaved in such a way that the odd symbols of two bursts are filled.

# 3.15.2.1 Coding of in-band data

The coding of in-band data is done as specified for the SID\_UPDATE\_INH frame at half rate in subclause 3.10.2.1.

#### 3.15.2.2 Identification marker

The identification marker is constructed as specified for the SID\_UPDATE\_INH frame at half rate in subclause 3.10.2.2.

# 3.15.2.3 Repetition

The coded bits (c) are repeated according to the following rule:

$$c'(3k+2) = c'(3k+1) = c'(3k) = c(k)$$
 for  $k=0,...,227$ 

#### 3.15.2.4 Interleaving

Before interleaving the coded bits  $\{c'(0) \dots c'(683)\}$  are converted into 3-bit symbols  $\{C(0) \dots C(227)\}$  according to table 1 in 3GPP TS 45.004, the symbol C(k) depends on c'(3k+2), c'(3k+1) and c'(3k) for  $k=0,1,\dots,227$ .

The interleaving is done as specified for the SID\_UPDATE\_INH frame at half rate in subclause 3.10.2.3. The difference is that the interleaving is done by symbols instead of single bits, reusing the existing interleaving tables. The result of the interleaving is a distribution of 114 of the reordered 228 symbols of a given data block over 2 blocks using the odd numbered bits. The even numbered symbols of these 2 blocks will be filled by the speech frame that following immediately after this frame.

# 3.15.2.5 Mapping on a Burst

The mapping is done as specified for the SID\_UPDATE\_INH frame at half rate in subclause 3.10.2.4 with exception that it is done by symbols instead of single bits.

# 3.15.3 SID FIRST P1

This frame type contains no source data from the speech coder. What is generated is the in-band channel and an identification marker. The in-band data id(0,1) represents Mode Indication or Mode Command/Mode Request depending on the current frame number.

# 3.15.3.1 Coding of in-band data

The coding of in-band data is done as specified for the SID\_FIRST\_P1 frame at half rate in subclause 3.10.3.1.

# 3.15.3.2 Identification marker

The identification marker is constructed as specified for the SID\_FIRST\_P1 frame at half rate in subclause 3.10.3.2.

#### 3.15.3.3 Repetition

The coded bits (c) are repeated according to the following rule:

$$c'(3k+2) = c'(3k+1) = c'(3k) = c(k)$$
 for  $k=0,...,227$ 

#### 3.15.3.4 Interleaving

Before interleaving the coded bits  $\{c'(0) \dots c'(683)\}$  are converted into 3-bit symbols  $\{C(0) \dots C(227)\}$  according to table 1 in 3GPP TS 45.004, the symbol C(k) depends on c'(3k+2), c'(3k+1) and c'(3k) for  $k=0,1,\dots,227$ .

The interleaving is done as specified for the TCH/HS in subclause 3.2.3. The difference is that the interleaving is done by symbols instead of single bits, reusing the existing interleaving tables.

# 3.15.3.5 Mapping on a Burst

The mapping is done as specified for the TCH/HS in subclause 3.2.4 with exception that it is done by symbols instead of single bits.

# 3.15.4 SID FIRST P2

This frame type contains no source data from the speech coder. What is generated is the in-band channel and, derived from that, an identification marker. The in-band data id(0,1) represents Mode Indication or Mode Command/Mode Request depending on the current frame number.

# 3.15.4.1 Coding of in-band data

The coding of in-band data is done as specified for the SID\_FIRST\_P2 frame at half rate in subclause 3.10.4.1.

#### 3.15.4.2 Repetition

The coded bits (c) are repeated according to the following rule:

$$c'(3k+2) = c'(3k+1) = c'(3k) = c(k)$$
 for  $k=0,2,4...,226$ 

#### 3.15.4.3 Interleaving

Before interleaving the coded bits  $\{c'(0), c'(1), c'(2), c'(6), c'(7), c'(8) \dots c'(678), c'(679), c'(680)\}$  are converted into 3-bit symbols  $\{C(0), C(2), C(4) \dots C(226)\}$  according to table 1 in 3GPP TS 45.004, the symbol C(k) depends on c'(3k+2), c'(3k+1) and c'(3k) for  $k=0,2,4,\dots,226$ .

The interleaving is done as specified for the SID\_FIRST\_P2 frame at half rate in subclause 3.10.4.2. The difference is that the interleaving is done by symbols instead of single bits, reusing the existing interleaving tables. The result of the interleaving is a distribution of 114 of the reordered 228 symbols of a given data block over 2 blocks using the even numbered symbols. The odd numbered symbols of these 2 blocks have already been filled by the SID\_FIRST\_P1 frame.

# 3.15.4.4 Mapping on a Burst

The mapping is done as specified for the SID\_FIRST\_P2 frame at half rate in subclause 3.10.4.3 with exception that it is done by symbols instead of single bits.

# 3.15.5 SID FIRST INH

This special frame is used when the first 2 burst of a SID\_FIRST\_P1 frame have been transmitted but the second two bursts cannot be transmitted due to a SPEECH frame. The general coding is as: the in-band data (Note that this must be the same data as for the SID\_FIRST\_P1 frame that is being inhibited) is encoded, a marker that is the opposite of the SID\_FIRST\_P1 marker is appended and the data is interleaved in such a way that the odd symbols of two bursts are filled.

# 3.15.5.1 Coding of in-band data

The coding of the in-band data is done as specified for the SID\_FIRST\_P1 frame in subclause 3.10.3.1.

#### 3.15.5.2 Identification marker

The identification marker is done as specified for the SID\_FIRST\_INH frame at half rate in subclause 3.10.5.2.

# 3.15.5.3 Repetition

The coded bits (c) are repeated according to the following rule:

$$c'(3k+2) = c'(3k+1) = c'(3k) = c(k)$$
 for  $k=0,...,227$ 

#### 3.15.5.4 Interleaving

Before interleaving the coded bits  $\{c'(0) \dots c'(683)\}$  are converted into 3-bit symbols  $\{C(0) \dots C(227)\}$  according to table 1 in 3GPP TS 45.004, the symbol C(k) depends on c'(3k+2), c'(3k+1) and c'(3k) for  $k=0,1,\dots,227$ .

The interleaving is done as specified for the SID\_UPDATE\_INH in subclause 3.10.2.3. The difference is that the interleaving is done by symbols instead of single bits, reusing the existing interleaving tables.

#### 3.15.5.5 Mapping on a Burst

The mapping is done as specified for the SID\_UPDATE\_INH in subclause 3.10.2.4 with exception that it is done by symbols instead of single bits.

#### 3.15.6 ONSET

Onset frames are used to preset the interleaver buffer after a period of no speech activity in DTX mode. This frame type contains no source data from the speech coder. What is transmitted is the in-band channel signalling the Mode Indication for the speech frame following the onset marker.

# 3.15.6.1 Coding of in-band data

The coding of in-band data is done as specified for the ONSET frame at half rate in subclause 3.10.6.1.

#### 3.15.6.2 Repetition

The repetition is done as specified for the SID\_UPDATE\_INH frame in subclause 3.15.2.3.

# 3.15.6.3 Interleaving

The interleaving is done as specified for the SID\_UPDATE\_INH frame in subclause 3.15.2.4.

# 3.15.6.4 Mapping on a Burst

The mapping is done as specified for the SID\_UPDATE\_INH frame in subclause 3.15.2.5.

#### 3.15.7 SPEECH

The speech coder delivers to the channel encoder a sequence of blocks of data. One block of data corresponds to one speech frame and the block length is different in each of the eight channel codec modes. Adjoining each block of data is information of the channel codec mode to use when encoding the block. Also delivered is the in-band data id(0,1) representing Mode Indication or Mode Command/Mode Request depending on the current frame number.

# 3.15.7.1 Coding of the in-band data

The two input in-band bits (id(0,1)) are coded to twelve coded in-band bits (ic(0..11)).

# 3.15.7.2 Ordering according to subjective importance

The bits delivered by the speech encoder,  $\{s(1), s(2), ..., s(K_s)\}$ , are rearranged according to subjective importance before channel coding. Tables 7 to 14 define the correct rearrangement for the speech codec modes 12.2 kbit/s, 10.2 kbit/s, 7.95 kbit/s, 7.4 kbit/s, 6.7 kbit/s, 5.9 kbit/s, 5.15 kbit/s and 4.75 kbit/s, respectively. In the tables speech codec parameters are numbered in the order they are delivered by the corresponding speech encoder according to 3GPP TS 26.090 and the rearranged bits are labelled  $\{d(0),d(1),...,d(K_{d-1})\}$ , defined in the order of decreasing importance. Index  $K_d$  refers to the number of bits delivered by the speech encoder, see below:

Codec mode	Number of speech bits delivered per block (K <sub>d</sub> )
O-TCH/AHS12.2	244
O-TCH/AHS10.2	204
O-TCH/AHS7.95	159
O-TCH/AHS7.4	148
O-TCH/AHS6.7	134
O-TCH/AHS5.9	118
O-TCH/AHS5.15	103
O-TCH/AHS4.75	95

The ordering algorithm is in pseudo code as:

for 
$$j = 0$$
 to  $K_{d-1}$   $d(j) := s(table(j)+1);$  where  $table(j)$  is read line by line left to right

The rearranged bits are further divided into three different classes to perform unequal error protection for different bits according to subjective importance.

The protection classes are:

- 1a Data protected with the CRC and the convolution code.
- 1b Data protected with the convolution code.

The number of class 1 (sum of class 1a and 1b), class 1a and class 1b bits for each codec mode is shown below:

Codec mode	Number of speech bits delivered per block	Number of class 1a bits per block	Number of class 1b bits per block
O-TCH/AHS12.2	244	81	163
O-TCH/AHS10.2	204	65	139
O-TCH/AHS7.95	159	75	84
O-TCH/AHS7.4	148	61	87
O-TCH/AHS6.7	134	55	79
O-TCH/AHS5.9	118	55	63
O-TCH/AHS5.15	103	49	54
O-TCH/AHS4.75	95	39	56

# 3.15.7.3 Parity for speech frames

The basic parameters for each codec mode for the first encoding step are shown below:

Codec mode	Speech encoded bits (K <sub>d</sub> )	CRC protected bits (K <sub>d1a</sub> )	Number of bits after first encoding step (K <sub>u</sub> = K <sub>d</sub> + 6)
O-TCH/AHS12.2	244	81	250
O-TCH/AHS10.2	204	65	210
O-TCH/AHS7.95	159	75	165
O-TCH/AHS7.4	148	61	154
O-TCH/AHS6.7	134	55	140
O-TCH/AHS5.9	118	55	124
O-TCH/AHS5.15	103	49	109
O-TCH/AHS4.75	95	39	101

A 6-bit CRC is used for error-detection. These 6 parity bits are generated by the cyclic generator polynomial:  $g(D) = D^6 + D^5 + D^3 + D^2 + D^1 + I$  from the first  $K_{d1a}$  bits of class 1, where  $K_{d1a}$  refers to number of bits in protection class 1a as shown above for each codec mode. The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

$$d(0)D(K_{d1a}+5)+d(1)D(K_{d1a}+4)+...+d(K_{d1a}-1)D(6)+p(0)D^{(5)}+...+p(4)D+p(5)$$

where p(0), p(1) ... p(5) are the parity bits, when divided by g(D), yields a remainder equal to:

$$1+D+D^2+D^3+D^4+D^5$$
.

The information and parity bits are merged:

u(k) = d(k)	for $k = 0, 1,, K_{d1a}-1$
$u(k) = p(k-K_{d1a})$	for $k = K_{d1a}, K_{d1a}+1,, K_{d1a}+5$
u(k) = d(k-6)	for $k = K_{d1a}+6, K_{d1a}+7,, K_{u}-1$

Thus, after the first encoding step u(k) will be defined by the following contents for each codec mode:

### O-TCH/AHS12.2:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 80$   $u(k) = p(k-81)$  for  $k = 81, 82, ..., 86$   $u(k) = d(k-6)$  for  $k = 87, 88, ..., 249$ 

# **O-TCH/AHS10.2**:

$$\begin{array}{ll} u(k) = d(k) & \text{for } k = 0, \ 1, \ ..., \ 64 \\ \\ u(k) = p(k\text{-}65) & \text{for } k = 65, \ 66, \ ..., \ 70 \\ \\ u(k) = d(k\text{-}6) & \text{for } k = 71, \ 72, \ ..., \ 209 \end{array}$$

#### **O-TCH/AHS7.95**:

$$\begin{array}{ll} u(k) = d(k) & \text{for } k = 0, \, 1, \, ..., \, 74 \\ \\ u(k) = p(k\text{-}75) & \text{for } k = 75, \, 76, \, ..., \, 80 \\ \\ u(k) = d(k\text{-}6) & \text{for } k = 81, \, 82, \, ..., \, 164 \end{array}$$

### **O-TCH/AHS7.4**:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 60$   
 $u(k) = p(k-61)$  for  $k = 61, 62, ..., 66$   
 $u(k) = d(k-6)$  for  $k = 67, 68, ..., 153$ 

#### O-TCH/AHS6.7:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 54$ 

$$u(k) = p(k-55)$$
 for  $k = 55, 56, ..., 60$ 

$$u(k) = d(k-6)$$
 for  $k = 61, 62, ..., 139$ 

#### O-TCH/AHS5.9:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 54$ 

$$u(k) = p(k-55)$$
 for  $k = 55, 56, ..., 60$ 

$$u(k) = d(k-6)$$
 for  $k = 61, 62, ..., 123$ 

#### **O-TCH/AHS5.15**:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 48$ 

$$u(k) = p(k-49)$$
 for  $k = 49, 50, ..., 54$ 

$$u(k) = d(k-6)$$
 for  $k = 55, 56, ..., 108$ 

#### **O-TCH/AHS4.75**:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 38$ 

$$u(k) = p(k-39)$$
 for  $k = 39, 40, ..., 44$ 

$$u(k) = d(k-6)$$
 for  $k = 45, 46, ..., 100$ 

#### 3.15.7.4 Convolutional encoder

The bits from the first encoding step (u(k)) are encoded with the recursive systematic convolutional codes as summarised below. The number of output bits after puncturing is 672 for all codec modes.

Codec Mode	Rate	Number of input bits to conv. coder	Number of output bits from conv. coder	Number of punctured bits
O-TCH/AHS12.2	1/3	250	768	96
O-TCH/AHS10.2	1/4	210	864	192
O-TCH/AHS7.95	1/4	165	684	12
O-TCH/AHS7.4	1/5	154	800	128
O-TCH/AHS6.7	1/5	140	730	58
O-TCH/AHS5.9	1/6	124	780	108
O-TCH/AHS5.15	1/6	109	690	18
O-TCH/AHS4.75	1/7	101	749	77

Below the coding for each codec mode is specified in detail.

#### **O-TCH/AHS12.2:**

The block of 250 bits  $\{u(0)...u(249)\}$  is encoded with the 1/3 rate convolutional code defined by the following polynomials:

$$G4/G7 = 1 + D^2 + D^3 + D^5 + D^6/1 + D + D^2 + D^3 + D^6$$

$$G5/G7 = 1 + D + D^4 + D^6/1 + D + D^2 + D^3 + D^6$$

$$G7/G7 = 1$$

resulting in 768 coded bits,  $\{C(0)...C(767)\}\$  defined by:

$$\begin{split} r(k) &= u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-6) \\ C(3k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(3k+1) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(3k+2) &= u(k) \end{split} \qquad \qquad \text{for } k = 0, 1, ..., 249; \qquad r(k) = 0 \text{ for } k < 0 \end{split}$$

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(3k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(3k+1) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(3k+2) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) \qquad \text{for } k = 250, 251, ..., 255 \end{split}$$

The following 448 coded bits are moved to data block P<sub>G</sub>:

```
C(2), C(3), C(5), C(6), C(8), C(9), C(11), C(12), C(14), C(15), C(17), C(18), C(20), C(21), C(22), C(23),
C(24), C(26), C(27), C(29), C(30), C(32), C(33), C(34), C(35), C(36), C(38), C(39), C(41), C(42), C(44),
C(45), C(46), C(47), C(48), C(50), C(51), C(53), C(54), C(56), C(57), C(58), C(59), C(60), C(62), C(63),
C(65), C(66), C(68), C(69), C(70), C(71), C(72), C(74), C(75), C(77), C(78), C(80), C(81), C(82), C(83),
C(84), C(86), C(87), C(89), C(90), C(92), C(93), C(94), C(95), C(96), C(98), C(99), C(101), C(102), C(104),
C(105), C(106), C(107), C(108), C(110), C(111), C(113), C(114), C(116), C(117), C(118), C(119), C(120),
C(122), C(123), C(125), C(126), C(128), C(129), C(130), C(131), C(132), C(134), C(135), C(137), C(138),
C(140), C(141), C(142), C(143), C(144), C(146), C(147), C(149), C(150), C(152), C(153), C(154), C(155),
C(156), C(158), C(159), C(161), C(162), C(164), C(165), C(166), C(167), C(168), C(170), C(171), C(173),
C(174), C(176), C(177), C(178), C(179), C(180), C(182), C(183), C(185), C(186), C(188), C(189), C(190),
C(191), C(192), C(194), C(195), C(197), C(198), C(200), C(201), C(202), C(203), C(204), C(206), C(207),
C(209), C(210), C(212), C(213), C(214), C(215), C(216), C(218), C(219), C(221), C(222), C(224), C(225),
C(226), C(227), C(228), C(230), C(231), C(233), C(234), C(236), C(237), C(238), C(239), C(240), C(242),
C(243), C(245), C(246), C(248), C(249), C(250), C(251), C(252), C(254), C(255), C(257), C(258), C(260),
C(261), C(262), C(263), C(264), C(266), C(267), C(269), C(270), C(272), C(273), C(275), C(276), C(278),
C(279), C(281), C(282), C(284), C(285), C(287), C(288), C(290), C(291), C(293), C(294), C(296), C(297),
C(299), C(300), C(302), C(303), C(305), C(306), C(308), C(309), C(311), C(312), C(314), C(315), C(317),
C(318), C(320), C(321), C(323), C(324), C(326), C(327), C(329), C(330), C(332), C(333), C(335), C(336),
C(338), C(339), C(341), C(342), C(344), C(345), C(347), C(348), C(350), C(351), C(353), C(354), C(356),
C(357), C(359), C(360), C(362), C(363), C(365), C(366), C(368), C(369), C(371), C(372), C(374), C(375),
C(377), C(378), C(380), C(381), C(383), C(384), C(386), C(387), C(389), C(390), C(392), C(393), C(395),
C(396), C(398), C(399), C(401), C(402), C(404), C(405), C(407), C(410), C(411), C(413), C(414), C(416),
C(419), C(420), C(422), C(423), C(425), C(428), C(431), C(434), C(435), C(437), C(438), C(440), C(443),
C(444), C(446), C(449), C(450), C(452), C(455), C(458), C(461), C(464), C(467), C(468), C(470), C(471), C(471), C(481), C(48
C(473), C(476), C(479), C(482), C(483), C(485), C(486), C(488), C(491), C(494), C(497), C(498), C(500),
C(503), C(506), C(509), C(512), C(515), C(516), C(518), C(519), C(521), C(524), C(527), C(530), C(531),
C(533), C(534), C(536), C(539), C(542), C(545), C(546), C(548), C(551), C(554), C(557), C(560), C(563),
C(564), C(566), C(567), C(569), C(572), C(575), C(578), C(579), C(581), C(582), C(584), C(587), C(590),
C(593), C(594), C(596), C(599), C(602), C(605), C(608), C(611), C(612), C(614), C(615), C(617), C(620),
C(623), C(626), C(627), C(629), C(630), C(632), C(635), C(638), C(641), C(642), C(644), C(647), C(650),
C(653), C(656), C(659), C(660), C(662), C(663), C(665), C(668), C(671), C(674), C(675), C(677), C(678),
C(680), C(683), C(686), C(689), C(690), C(692), C(695), C(698), C(701), C(704), C(707), C(708), C(710),
C(711), C(713), C(716), C(719), C(722), C(723), C(725), C(726), C(728), C(731), C(734), C(737), C(738),
C(740), C(743), C(746), C(749), C(755), C(758), C(761), C(764)
```

And the following 224 coded bits are moved to data block P<sub>B</sub>:

```
C(16), C(19), C(25), C(28), C(31), C(37), C(40), C(49), C(52), C(55), C(61), C(64), C(67), C(73), C(76), C(79), C(85), C(88), C(97), C(100), C(103), C(109), C(112), C(115), C(121), C(124), C(127), C(133), C(136), C(145), C(148), C(151), C(157), C(160), C(163), C(169), C(172), C(175), C(181), C(184), C(193), C(196), C(199), C(205), C(208), C(211), C(217), C(220), C(223), C(229), C(232), C(241), C(244), C(247), C(253), C(256), C(259), C(265), C(268), C(271), C(274), C(277), C(280), C(283), C(286), C(289), C(295), C(301), C(307), C(310), C(316), C(322), C(328), C(337), C(343), C(349), C(355), C(358), C(364), C(370), C(376), C(385), C(391), C(397), C(403), C(406), C(408), C(412), C(417), C(418), C(424), C(426), C(429), C(429), C(429), C(426), C(429), C(426), C(429), C(426), C(429), C(426), C(426), C(429), C(426), C(426), C(429), C(426), C(426), C(429), C(426), C
```

```
 \begin{array}{l} C(432), C(433), C(439), C(441), C(445), C(447), C(451), C(453), C(454), C(456), C(459), C(460), C(462), \\ C(465), C(466), C(472), C(474), C(477), C(480), C(481), C(487), C(489), C(492), C(493), C(495), C(499), \\ C(501), C(502), C(504), C(507), C(508), C(510), C(513), C(514), C(520), C(522), C(525), C(528), C(529), \\ C(535), C(537), C(540), C(541), C(543), C(547), C(549), C(550), C(552), C(555), C(556), C(558), C(561), \\ C(562), C(568), C(570), C(573), C(576), C(577), C(583), C(585), C(588), C(589), C(591), C(595), C(597), \\ C(598), C(600), C(603), C(604), C(606), C(609), C(610), C(616), C(618), C(621), C(624), C(625), C(631), \\ C(633), C(636), C(637), C(639), C(643), C(645), C(646), C(648), C(651), C(652), C(654), C(657), C(658), \\ C(664), C(666), C(669), C(672), C(673), C(679), C(681), C(684), C(685), C(687), C(691), C(693), C(694), \\ C(696), C(699), C(700), C(702), C(705), C(706), C(712), C(714), C(717), C(720), C(721), C(727), C(729), \\ C(732), C(733), C(735), C(739), C(741), C(742), C(744), C(747), C(750), C(752), C(753), C(756), C(759), \\ C(767) \end{array}
```

The vectors P<sub>G</sub> and P<sub>B</sub> of coded and punctured bits is combined with in band bits to vector P<sub>C</sub> as

$$\begin{split} P_{C^{'}}(k) &= ic(k) & \text{for } k = 0,\, 1,\, 2,\, 3 \\ P_{C^{'}}(k+4) &= P_{G}(k) & \text{for } k = 0,\, 1,\, ...,\, 223 \\ P_{C^{'}}(k+224) &= ic(k) & \text{for } k = 4,\, 5,\, 6,\, 7 \\ P_{C^{'}}(k+8) &= P_{G}(k) & \text{for } k = 224,\, 225,\, ...,\, 447 \\ P_{C^{'}}(k+448) &= ic(k) & \text{for } k = 8,\, 9,\, 10,\, 11 \\ P_{C^{'}}(k+460) &= P_{B}(k) & \text{for } k = 0,\, 1,\, ...,\, 223 \end{split}$$

#### **O-TCH/AHS10.2:**

The block of 210 bits  $\{u(0)...u(209)\}$  is encoded with the  $\frac{1}{4}$  rate convolutional code defined by the following polynomials:

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$
 
$$G5/G7 = 1 + D + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$
 
$$G6/G7 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$
 
$$G7/G7 = 1$$

resulting in 864 coded bits,  $\{C(0)...C(863)\}\$  defined by:

$$\begin{split} r(k) &= u(k) + r(k\text{-}1) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}6) \\ C(4k) &= r(k) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}5) + r(k\text{-}6) \\ C(4k+1) &= r(k) + r(k\text{-}1) + r(k\text{-}4) + r(k\text{-}6) \\ C(4k+2) &= r(k) + r(k\text{-}1) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}4) + r(k\text{-}6) \\ C(4k+3) &= u(k) \end{split}$$
 for  $k = 0, 1, ..., 209$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(4k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(4k+1) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(4k+2) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(4k+3) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) \qquad \text{for } k = 210, 211, ..., 215 \end{split}$$

The following 448 coded bits are moved to data block P<sub>G</sub>:

```
C(3), C(7), C(8), C(11), C(12), C(15), C(16), C(19), C(20), C(23), C(24), C(27), C(28), C(31), C(32), C(33),
C(35), C(36), C(39), C(40), C(43), C(44), C(45), C(47), C(48), C(51), C(52), C(55), C(56), C(57), C(59),
C(60), C(63), C(64), C(67), C(68), C(69), C(71), C(72), C(75), C(76), C(79), C(80), C(81), C(83), C(84),
C(87), C(88), C(91), C(92), C(93), C(95), C(96), C(99), C(100), C(103), C(104), C(105), C(107), C(108),
C(111), C(112), C(115), C(116), C(117), C(119), C(120), C(123), C(124), C(127), C(128), C(129), C(131),
C(132), C(135), C(136), C(139), C(140), C(141), C(143), C(144), C(147), C(148), C(151), C(152), C(153),
C(155), C(156), C(159), C(160), C(163), C(164), C(165), C(167), C(168), C(171), C(172), C(175), C(176),
C(177), C(179), C(180), C(183), C(184), C(187), C(188), C(189), C(191), C(192), C(195), C(196), C(199),
C(200), C(201), C(203), C(204), C(207), C(208), C(211), C(212), C(213), C(215), C(216), C(219), C(220),
C(223), C(224), C(225), C(227), C(228), C(231), C(232), C(235), C(236), C(237), C(239), C(240), C(243),
C(244), C(247), C(248), C(251), C(252), C(255), C(256), C(259), C(260), C(261), C(263), C(264), C(267),
C(268), C(271), C(272), C(275), C(276), C(279), C(280), C(283), C(284), C(285), C(287), C(288), C(291),
C(292), C(295), C(296), C(299), C(300), C(303), C(304), C(307), C(308), C(311), C(312), C(315), C(316),
C(319), C(320), C(323), C(324), C(327), C(328), C(331), C(332), C(335), C(336), C(339), C(340), C(343),
C(344), C(347), C(348), C(351), C(352), C(355), C(356), C(359), C(360), C(363), C(364), C(367), C(368),
C(371), C(372), C(375), C(376), C(379), C(380), C(383), C(384), C(387), C(388), C(391), C(392), C(395),
C(396), C(399), C(400), C(403), C(404), C(407), C(408), C(411), C(412), C(415), C(416), C(419), C(420),
C(423), C(424), C(427), C(428), C(431), C(432), C(435), C(436), C(439), C(440), C(443), C(444), C(447),
C(448), C(451), C(452), C(455), C(456), C(459), C(460), C(463), C(464), C(467), C(468), C(471), C(472),
C(475), C(476), C(479), C(480), C(483), C(484), C(487), C(488), C(491), C(492), C(495), C(496), C(499),
C(500), C(503), C(504), C(507), C(508), C(511), C(512), C(515), C(516), C(519), C(520), C(523), C(524),
C(527), C(528), C(531), C(532), C(535), C(536), C(539), C(540), C(543), C(544), C(547), C(548), C(551),
C(552), C(555), C(556), C(559), C(560), C(563), C(564), C(567), C(568), C(571), C(572), C(575), C(576),
C(579), C(580), C(583), C(584), C(587), C(588), C(591), C(592), C(595), C(596), C(596), C(600), C(603),
C(604), C(607), C(608), C(611), C(612), C(615), C(616), C(619), C(620), C(623), C(624), C(627), C(628),
C(631), C(632), C(635), C(636), C(639), C(640), C(643), C(644), C(647), C(648), C(651), C(652), C(655),
C(656), C(659), C(660), C(663), C(664), C(667), C(668), C(671), C(672), C(675), C(676), C(679), C(680),
C(683), C(684), C(687), C(688), C(691), C(692), C(695), C(696), C(699), C(700), C(703), C(704), C(707),
C(708), C(711), C(712), C(715), C(716), C(719), C(720), C(723), C(724), C(727), C(728), C(731), C(732),
C(735), C(736), C(739), C(740), C(743), C(744), C(747), C(748), C(751), C(752), C(755), C(756), C(759),
C(760), C(763), C(764), C(767), C(768), C(771), C(772), C(775), C(776), C(779), C(780), C(783), C(784),
C(787), C(788), C(791), C(792), C(795), C(796), C(799), C(800), C(803), C(804), C(807), C(808), C(811),
C(812), C(815), C(816), C(819), C(820), C(823), C(824), C(827), C(828), C(831), C(832), C(835), C(836),
C(839), C(840), C(843), C(844), C(847), C(848), C(851), C(852), C(855), C(859), C(863)
```

And the following 224 coded bits are moved to data block P<sub>B</sub>:

```
C(4), C(17), C(21), C(25), C(26), C(29), C(34), C(37), C(41), C(42), C(46), C(49), C(50), C(53), C(58),
C(61), C(65), C(66), C(70), C(73), C(77), C(82), C(85), C(89), C(90), C(94), C(97), C(101), C(106), C(109),
C(113), C(114), C(118), C(121), C(125), C(130), C(133), C(137), C(138), C(142), C(145), C(149), C(154),
C(157), C(161), C(162), C(166), C(169), C(173), C(178), C(181), C(185), C(186), C(190), C(193), C(197),
C(202), C(205), C(209), C(210), C(214), C(217), C(221), C(226), C(229), C(233), C(234), C(238), C(241),
C(245), C(249), C(250), C(253), C(257), C(258), C(262), C(265), C(269), C(273), C(274), C(277), C(281),
C(282), C(286), C(289), C(293), C(297), C(298), C(301), C(305), C(306), C(309), C(310), C(313), C(317),
C(321), C(325), C(329), C(333), C(337), C(341), C(345), C(349), C(353), C(357), C(361), C(365), C(369),
C(373), C(377), C(381), C(385), C(389), C(393), C(397), C(401), C(405), C(409), C(413), C(417), C(421),
C(425), C(429), C(433), C(437), C(441), C(445), C(449), C(453), C(457), C(461), C(465), C(469), C(473),
C(477), C(481), C(485), C(489), C(493), C(497), C(501), C(505), C(509), C(513), C(517), C(521), C(525),
C(529), C(533), C(537), C(541), C(545), C(549), C(553), C(557), C(561), C(565), C(569), C(573), C(577),
C(581), C(585), C(589), C(593), C(597), C(601), C(605), C(609), C(613), C(617), C(621), C(625), C(629),
C(633), C(637), C(641), C(645), C(649), C(653), C(657), C(661), C(665), C(669), C(673), C(677), C(681),
C(685), C(689), C(693), C(697), C(701), C(705), C(709), C(713), C(717), C(721), C(725), C(729), C(733),
C(737), C(741), C(745), C(749), C(753), C(757), C(761), C(765), C(769), C(773), C(777), C(781), C(785),
C(789), C(793), C(797), C(801), C(805), C(809), C(813), C(817), C(821), C(825), C(829), C(833)
```

The vectors P<sub>G</sub> and P<sub>B</sub> of coded and punctured bits is combined with in band bits to vector P<sub>C</sub>' as

```
P_{C'}(k) = ic(k) for k = 0, 1, 2, 3

P_{C'}(k+4) = P_{G}(k) for k = 0, 1, ..., 223
```

$$\begin{split} P_{C}'\left(k{+}224\right) &= ic(k) & \text{for } k = 4, 5, 6, 7 \\ P_{C}'\left(k{+}8\right) &= P_{G}(k) & \text{for } k = 224, 225, ..., 447 \\ P_{C}'\left(k{+}448\right) &= ic(k) & \text{for } k = 8, 9, 10, 11 \\ P_{C}'\left(k{+}460\right) &= P_{B}(k) & \text{for } k = 0, 1, ..., 223 \end{split}$$

# **O-TCH/AHS7.95:**

The block of 165 bits  $\{u(0)...u(164)\}$  is encoded with the  $\frac{1}{4}$  rate convolutional code defined by polynomials:

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G5/G7 = 1 + D + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G6/G7 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G7/G7 = 1$$

resulting in 684 coded bits,  $\{C(0)...C(683)\}\$  defined by:

$$\begin{split} r(k) &= u(k) + r(k\text{-}1) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}6) \\ C(4k) &= r(k) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}5) + r(k\text{-}6) \\ C(4k+1) &= r(k) + r(k\text{-}1) + r(k\text{-}4) + r(k\text{-}6) \\ C(4k+2) &= r(k) + r(k\text{-}1) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}4) + r(k\text{-}6) \\ C(4k+3) &= u(k) \end{split}$$
 for  $k = 0, 1, ..., 164$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(4k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(4k+1) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(4k+2) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(4k+3) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) & \text{for } k = 165, 166, ..., 170 \end{split}$$

The following 448 coded bits are moved to data block P<sub>G</sub>:

```
C(3), C(7), C(11), C(15), C(16), C(19), C(20), C(23), C(24), C(27), C(28), C(29), C(31), C(32), C(33), C(35),
C(36), C(37), C(39), C(40), C(41), C(43), C(44), C(47), C(48), C(49), C(51), C(52), C(53), C(55), C(56),
C(57), C(59), C(60), C(61), C(63), C(64), C(65), C(67), C(68), C(71), C(72), C(73), C(75), C(76), C(77),
C(79), C(80), C(81), C(83), C(84), C(85), C(87), C(88), C(89), C(91), C(92), C(95), C(96), C(97), C(99),
C(100), C(101), C(103), C(104), C(105), C(107), C(108), C(109), C(111), C(112), C(113), C(115), C(116),
C(119), C(120), C(121), C(123), C(124), C(125), C(127), C(128), C(129), C(131), C(132), C(133), C(135),
C(136), C(137), C(139), C(140), C(143), C(144), C(145), C(147), C(148), C(149), C(151), C(152), C(153),
C(155), C(156), C(157), C(159), C(160), C(161), C(163), C(164), C(167), C(168), C(169), C(171), C(172),
C(173), C(175), C(176), C(177), C(179), C(180), C(181), C(183), C(184), C(185), C(187), C(188), C(191),
C(192), C(193), C(195), C(196), C(197), C(199), C(200), C(201), C(203), C(204), C(205), C(207), C(208),
C(209), C(211), C(212), C(215), C(216), C(217), C(219), C(220), C(221), C(223), C(224), C(225), C(227),
C(228), C(229), C(231), C(232), C(233), C(235), C(236), C(239), C(240), C(241), C(243), C(244), C(245),
C(247), C(248), C(249), C(251), C(252), C(253), C(255), C(256), C(257), C(259), C(260), C(263), C(264),
C(265), C(267), C(268), C(269), C(271), C(272), C(273), C(275), C(276), C(277), C(279), C(280), C(281),
C(283), C(284), C(287), C(288), C(289), C(291), C(292), C(293), C(295), C(296), C(297), C(299), C(300),
C(301), C(303), C(304), C(305), C(307), C(308), C(311), C(312), C(313), C(315), C(316), C(317), C(319),
C(320), C(321), C(323), C(324), C(325), C(327), C(328), C(329), C(331), C(332), C(335), C(336), C(337),
C(339), C(340), C(341), C(343), C(344), C(345), C(347), C(348), C(349), C(351), C(352), C(353), C(355),
C(356), C(359), C(360), C(361), C(363), C(364), C(365), C(367), C(368), C(369), C(371), C(372), C(373),
C(375), C(376), C(377), C(379), C(380), C(383), C(384), C(385), C(387), C(388), C(389), C(391), C(392),
```

```
C(393), C(395), C(396), C(397), C(399), C(400), C(401), C(403), C(404), C(407), C(408), C(409), C(411), C(412), C(413), C(413), C(415), C(416), C(417), C(419), C(420), C(421), C(423), C(424), C(425), C(427), C(428), C(431), C(432), C(433), C(435), C(436), C(437), C(439), C(440), C(441), C(443), C(444), C(445), C(447), C(448), C(449), C(451), C(452), C(455), C(456), C(457), C(459), C(460), C(461), C(463), C(464), C(465), C(467), C(468), C(469), C(471), C(472), C(473), C(475), C(476), C(479), C(480), C(481), C(483), C(484), C(487), C(488), C(489), C(491), C(492), C(495), C(496), C(497), C(499), C(500), C(503), C(504), C(505), C(507), C(508), C(511), C(512), C(513), C(515), C(516), C(519), C(520), C(521), C(523), C(524), C(527), C(528), C(529), C(531), C(532), C(535), C(536), C(537), C(539), C(540), C(543), C(544), C(545), C(547), C(548), C(551), C(572), C(575), C(576), C(577), C(579), C(580), C(581), C(583), C(584), C(585), C(587), C(588), C(591), C(592), C(593), C(595), C(596), C(599), C(600), C(601), C(603), C(604), C(607), C(608), C(609), C(611), C(612), C(615), C(616), C(617), C(619), C(640), C(641), C(643), C(644), C(647), C(648), C(651), C(655), C(656), C(659), C(660), C(663), C(664), C(667), C(667), C(671), C(675), C(679), C(683)
```

And the following 224 coded bits are moved to data block P<sub>B</sub>:

```
C(0), C(4), C(8), C(9), C(10), C(12), C(13), C(14), C(17), C(18), C(21), C(22), C(25), C(26), C(30), C(34),
C(38), C(42), C(45), C(46), C(50), C(54), C(58), C(62), C(66), C(69), C(70), C(74), C(78), C(82), C(86),
C(90), C(93), C(94), C(98), C(102), C(106), C(110), C(114), C(117), C(118), C(122), C(126), C(130),
C(134), C(138), C(141), C(142), C(146), C(150), C(154), C(158), C(162), C(165), C(166), C(170), C(174),
C(178), C(182), C(186), C(189), C(190), C(194), C(198), C(202), C(206), C(210), C(213), C(214), C(218),
C(222), C(226), C(230), C(234), C(237), C(238), C(242), C(246), C(250), C(254), C(258), C(261), C(262),
C(266), C(270), C(274), C(278), C(282), C(285), C(286), C(290), C(294), C(298), C(302), C(306), C(309),
C(310), C(314), C(318), C(322), C(326), C(330), C(333), C(334), C(338), C(342), C(346), C(350), C(354),
C(357), C(358), C(362), C(366), C(370), C(374), C(378), C(381), C(382), C(386), C(390), C(394), C(398),
C(402), C(405), C(406), C(410), C(414), C(418), C(422), C(426), C(429), C(430), C(434), C(438), C(442),
C(446), C(450), C(453), C(454), C(458), C(462), C(466), C(470), C(474), C(477), C(478), C(482), C(485),
C(486), C(490), C(493), C(494), C(498), C(501), C(502), C(506), C(509), C(510), C(514), C(517), C(518),
C(522), C(525), C(526), C(530), C(533), C(534), C(538), C(541), C(542), C(546), C(549), C(550), C(554),
C(557), C(558), C(562), C(565), C(566), C(570), C(573), C(574), C(578), C(581), C(582), C(586), C(589),
C(590), C(594), C(597), C(598), C(602), C(605), C(606), C(610), C(613), C(614), C(618), C(621), C(622),
C(626), C(629), C(630), C(634), C(637), C(638), C(642), C(645), C(646), C(649), C(650), C(652), C(653),
C(654), C(657), C(658), C(661), C(662), C(665), C(666), C(668), C(669), C(670), C(672)
```

The vectors  $P_G$  and  $P_B$  of coded and punctured bits is combined with in band bits to vector  $P_C$ ' as

$P_{C}'(k) = ic(k)$	for $k = 0, 1, 2, 3$
$P_{C}'(k+4) = P_{G}(k)$	for $k = 0, 1,, 223$
$P_{C}'(k+224) = ic(k)$	for $k = 4, 5, 6, 7$
$P_{C'}(k+8) = P_{G}(k)$	for k = 224, 225,, 447
$P_{C}'(k+448) = ic(k)$	for $k = 8, 9, 10, 11$
$P_{C}'(k+460) = P_{B}(k)$	for $k = 0, 1,, 223$

#### **O-TCH/AHS7.4**:

The block of 154 bits  $\{u(0)...u(153)\}$  is encoded with the 1/5 rate convolutional code defined by the following polynomials:

$$\begin{aligned} G4/G7 &= 1 + D^2 + D^3 + D^5 + D^6/1 + D + D^2 + D^3 + D^6 \\ G4/G7 &= 1 + D^2 + D^3 + D^5 + D^6/1 + D + D^2 + D^3 + D^6 \\ G5/G7 &= 1 + D + D^4 + D^6/1 + D + D^2 + D^3 + D^6 \\ G6/G7 &= 1 + D + D^2 + D^3 + D^4 + D^6/1 + D + D^2 + D^3 + D^6 \\ G7/G7 &= 1 \end{aligned}$$

r(k) = 0 for k < 0

for k = 0, 1, ..., 153;

resulting in 800 coded bits,  $\{C(0)...C(799)\}\$  defined by:

$$r(k) = u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-6)$$

$$C(5k) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(5k+1) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(5k+2) = r(k) + r(k-1) + r(k-4) + r(k-6)$$

$$C(5k+3) = r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6)$$

and (for termination of the coder):

C(5k+4) = u(k)

$$\begin{split} r(k) &= 0 \\ C(5k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(5k+1) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(5k+2) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(5k+3) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(5k+4) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) & \text{for } k = 154, 155, ..., 159 \end{split}$$

The following 448 coded bits are moved to data block P<sub>G</sub>:

```
C(4), C(9), C(14), C(17), C(19), C(22), C(24), C(27), C(29), C(32), C(34), C(37), C(39), C(49), C(43), C(44),
C(47), C(48), C(49), C(52), C(53), C(54), C(57), C(58), C(59), C(62), C(63), C(64), C(67), C(68), C(69),
C(72), C(73), C(74), C(77), C(78), C(79), C(82), C(83), C(84), C(87), C(88), C(89), C(92), C(93), C(94),
C(97), C(98), C(99), C(102), C(103), C(104), C(107), C(108), C(109), C(112), C(113), C(114), C(117),
C(118), C(119), C(122), C(123), C(124), C(127), C(128), C(129), C(132), C(133), C(134), C(137), C(138),
C(139), C(142), C(143), C(144), C(147), C(148), C(149), C(152), C(153), C(154), C(157), C(158), C(159),
C(162), C(163), C(164), C(167), C(168), C(169), C(172), C(173), C(174), C(177), C(178), C(179), C(182),
C(183), C(184), C(187), C(188), C(189), C(192), C(193), C(194), C(197), C(198), C(199), C(202), C(203),
C(204), C(207), C(208), C(209), C(212), C(213), C(214), C(217), C(218), C(219), C(222), C(223), C(224),
C(227), C(228), C(229), C(232), C(233), C(234), C(237), C(238), C(239), C(242), C(243), C(244), C(247),
C(248), C(249), C(252), C(253), C(254), C(257), C(258), C(259), C(262), C(263), C(264), C(267), C(268),
C(269), C(272), C(273), C(274), C(277), C(278), C(279), C(282), C(283), C(284), C(287), C(288), C(289),
C(292), C(293), C(294), C(297), C(298), C(299), C(302), C(303), C(304), C(307), C(308), C(309), C(312),
C(313), C(314), C(317), C(318), C(319), C(322), C(323), C(324), C(327), C(328), C(329), C(332), C(333),
C(334), C(337), C(338), C(339), C(342), C(343), C(344), C(347), C(348), C(349), C(352), C(353), C(354),
C(357), C(358), C(359), C(362), C(363), C(364), C(367), C(368), C(369), C(372), C(373), C(374), C(377),
C(378), C(379), C(382), C(383), C(384), C(387), C(389), C(392), C(393), C(394), C(397), C(398), C(399),
C(402), C(403), C(404), C(407), C(408), C(409), C(412), C(413), C(414), C(417), C(418), C(419), C(422),
C(423), C(424), C(427), C(428), C(429), C(432), C(433), C(434), C(437), C(439), C(442), C(443), C(444), C(44
C(447), C(448), C(449), C(452), C(453), C(454), C(457), C(458), C(459), C(462), C(463), C(464), C(467),
C(468), C(469), C(472), C(473), C(474), C(477), C(478), C(479), C(482), C(483), C(484), C(487), C(489),
C(492), C(493), C(494), C(497), C(498), C(499), C(502), C(503), C(504), C(507), C(508), C(509), C(512),
C(513), C(514), C(517), C(518), C(519), C(522), C(523), C(524), C(527), C(528), C(529), C(532), C(533),
C(534), C(537), C(539), C(542), C(543), C(544), C(547), C(548), C(549), C(552), C(553), C(554), C(557),
C(558), C(559), C(562), C(563), C(564), C(567), C(568), C(569), C(572), C(573), C(574), C(577), C(578),
C(579), C(582), C(583), C(584), C(587), C(589), C(592), C(593), C(594), C(597), C(598), C(599), C(602),
C(603), C(604), C(607), C(608), C(609), C(612), C(613), C(614), C(617), C(618), C(619), C(622), C(623), C(603), C(604), C(604), C(607), C(608), C(60
C(624), C(627), C(628), C(629), C(632), C(633), C(634), C(637), C(639), C(642), C(643), C(644), C(647),
C(648), C(649), C(652), C(653), C(654), C(657), C(658), C(659), C(662), C(663), C(664), C(667), C(668),
C(669), C(672), C(673), C(674), C(677), C(678), C(679), C(682), C(683), C(684), C(687), C(689), C(692),
C(693), C(694), C(697), C(698), C(699), C(702), C(703), C(704), C(707), C(708), C(709), C(712), C(713),
C(714), C(717), C(718), C(719), C(722), C(723), C(724), C(728), C(729), C(733), C(734), C(737), C(739),
C(742), C(743), C(744), C(747), C(748), C(749), C(752), C(753), C(754), C(758), C(759), C(763), C(764),
C(767), C(768), C(769), C(773), C(774), C(778), C(779), C(783), C(784), C(789), C(789), C(799)
```

And the following 224 coded bits are moved to data block P<sub>B</sub>:

```
C(12), C(13), C(16), C(18), C(21), C(23), C(26), C(28), C(31), C(33), C(36), C(38), C(41), C(45), C(46), 
C(50), C(51), C(55), C(56), C(60), C(61), C(65), C(66), C(71), C(75), C(76), C(80), C(81), C(85), C(86),
C(90), C(91), C(95), C(96), C(100), C(101), C(105), C(106), C(110), C(111), C(115), C(116), C(121),
C(125), C(126), C(130), C(131), C(135), C(136), C(140), C(141), C(145), C(146), C(150), C(151), C(155),
C(156), C(160), C(161), C(165), C(166), C(171), C(175), C(176), C(180), C(181), C(185), C(186), C(190),
C(191), C(195), C(196), C(200), C(201), C(205), C(206), C(210), C(211), C(215), C(216), C(221), C(225),
C(226), C(230), C(231), C(235), C(236), C(240), C(241), C(245), C(246), C(250), C(251), C(255), C(256),
C(260), C(261), C(265), C(266), C(271), C(275), C(276), C(280), C(281), C(285), C(286), C(290), C(291),
C(296), C(300), C(301), C(305), C(306), C(310), C(311), C(315), C(316), C(321), C(325), C(326), C(330),
C(331), C(335), C(336), C(340), C(341), C(346), C(350), C(351), C(355), C(356), C(360), C(361), C(365),
C(366), C(371), C(375), C(376), C(380), C(381), C(385), C(386), C(388), C(390), C(391), C(396), C(400),
C(401), C(405), C(406), C(410), C(411), C(415), C(416), C(421), C(425), C(426), C(430), C(431), C(435),
C(436), C(438), C(440), C(441), C(446), C(450), C(451), C(455), C(456), C(460), C(461), C(465), C(466),
C(471), C(475), C(476), C(480), C(481), C(485), C(486), C(488), C(491), C(496), C(500), C(501), C(505),
C(506), C(510), C(511), C(516), C(521), C(536), C(538), C(546), C(561), C(571), C(586), C(588), C(596),
C(611), C(621), C(636), C(638), C(646), C(661), C(671), C(686), C(688), C(696), C(711), C(721), C(727),
C(732), C(736), C(738), C(746), C(757), C(762), C(772), C(777), C(782), C(787), C(788), C(793)
```

The vectors  $P_G$  and  $P_B$  of coded and punctured bits is combined with in band bits to vector  $P_{C}$  as

$$\begin{split} P_{C^{'}}(k) &= ic(k) & \text{for } k = 0,\, 1,\, 2,\, 3 \\ \\ P_{C^{'}}(k+4) &= P_{G}(k) & \text{for } k = 0,\, 1,\, ...,\, 223 \\ \\ P_{C^{'}}(k+224) &= ic(k) & \text{for } k = 4,\, 5,\, 6,\, 7 \\ \\ P_{C^{'}}(k+8) &= P_{G}(k) & \text{for } k = 224,\, 225,\, ...,\, 447 \\ \\ P_{C^{'}}(k+448) &= ic(k) & \text{for } k = 8,\, 9,\, 10,\, 11 \\ \\ P_{C^{'}}(k+460) &= P_{B}(k) & \text{for } k = 0,\, 1,\, ...,\, 223 \end{split}$$

### O-TCH/AHS6.7:

The block of 140 bits  $\{u(0)...\ u(139)\}$  is encoded with the 1/5 rate convolutional code defined by the following polynomials:

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G5/G7 = 1 + D + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G6/G7 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G7/G7 = 1$$

resulting in 730 coded bits,  $\{C(0)...C(729)\}\$  defined by:

$$\begin{split} r(k) &= u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-6) \\ C(5k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(5k+1) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(5k+2) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(5k+3) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(5k+4) &= u(k) & \text{for } k = 0, 1, ..., 139; \qquad r(k) = 0 \text{ for } k < 0 \end{split}$$

and (for termination of the coder):

```
\begin{split} r(k) &= 0 \\ C(5k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(5k+1) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(5k+2) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(5k+3) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(5k+4) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) & \text{for } k = 140, 141, ..., 145 \end{split}
```

The following 448 coded bits are moved to data block P<sub>G</sub>:

```
C(4), C(9), C(12), C(13), C(14), C(17), C(18), C(19), C(21), C(22), C(23), C(24), C(27), C(28), C(29), C(32),
C(33), C(34), C(36), C(37), C(38), C(39), C(42), C(43), C(44), C(46), C(47), C(48), C(49), C(52), C(53),
C(54), C(57), C(58), C(59), C(61), C(62), C(63), C(64), C(67), C(68), C(69), C(71), C(72), C(73), C(74),
C(77), C(78), C(79), C(82), C(83), C(84), C(86), C(87), C(88), C(89), C(92), C(93), C(94), C(96), C(97),
C(98), C(99), C(102), C(103), C(104), C(107), C(108), C(109), C(111), C(112), C(113), C(114), C(117),
C(118), C(119), C(121), C(122), C(123), C(124), C(127), C(128), C(129), C(132), C(133), C(134), C(136),
C(137), C(138), C(139), C(142), C(143), C(144), C(146), C(147), C(148), C(149), C(152), C(153), C(154),
C(157), C(158), C(159), C(161), C(162), C(163), C(164), C(167), C(168), C(169), C(171), C(172), C(173),
C(174), C(177), C(178), C(179), C(182), C(183), C(184), C(186), C(187), C(188), C(189), C(192), C(193),
C(194), C(196), C(197), C(198), C(199), C(202), C(203), C(204), C(207), C(208), C(209), C(211), C(212),
C(213), C(214), C(217), C(218), C(219), C(221), C(222), C(223), C(224), C(227), C(228), C(229), C(232),
C(233), C(234), C(236), C(237), C(238), C(239), C(242), C(243), C(244), C(246), C(247), C(248), C(249),
C(252), C(253), C(254), C(257), C(258), C(259), C(261), C(262), C(263), C(264), C(267), C(268), C(269),
C(271), C(272), C(273), C(274), C(277), C(278), C(279), C(282), C(283), C(284), C(286), C(287), C(288),
C(289), C(292), C(293), C(294), C(296), C(297), C(298), C(299), C(302), C(303), C(304), C(307), C(308),
C(309), C(311), C(312), C(313), C(314), C(317), C(318), C(319), C(321), C(322), C(323), C(324), C(327),
C(328), C(329), C(332), C(333), C(334), C(336), C(337), C(338), C(339), C(342), C(343), C(344), C(347),
C(348), C(349), C(352), C(353), C(354), C(357), C(358), C(359), C(362), C(363), C(364), C(367), C(368),
C(369), C(372), C(373), C(374), C(377), C(378), C(379), C(382), C(383), C(384), C(387), C(388), C(389),
C(392), C(393), C(394), C(397), C(398), C(399), C(402), C(403), C(404), C(407), C(408), C(409), C(412),
C(413), C(414), C(417), C(418), C(419), C(422), C(423), C(424), C(427), C(428), C(429), C(432), C(433),
C(434), C(437), C(438), C(439), C(442), C(443), C(444), C(447), C(448), C(449), C(452), C(453), C(454),
C(457), C(458), C(459), C(462), C(463), C(464), C(467), C(468), C(469), C(472), C(473), C(474), C(477),
C(478), C(479), C(482), C(483), C(484), C(487), C(488), C(489), C(492), C(493), C(494), C(498), C(499),
C(502), C(503), C(504), C(507), C(508), C(509), C(512), C(513), C(514), C(517), C(518), C(519), C(523),
C(524), C(527), C(528), C(529), C(532), C(533), C(534), C(537), C(538), C(539), C(542), C(543), C(544),
C(548), C(549), C(552), C(553), C(554), C(557), C(558), C(559), C(562), C(563), C(564), C(567), C(568),
C(569), C(573), C(574), C(577), C(578), C(579), C(582), C(583), C(584), C(587), C(588), C(589), C(592),
C(593), C(594), C(598), C(599), C(602), C(603), C(604), C(607), C(608), C(609), C(612), C(613), C(614),
C(617), C(618), C(619), C(623), C(624), C(627), C(628), C(629), C(632), C(633), C(634), C(637), C(638),
C(639), C(642), C(643), C(644), C(648), C(649), C(652), C(653), C(654), C(657), C(658), C(659), C(662),
C(663), C(664), C(667), C(668), C(669), C(673), C(674), C(677), C(678), C(679), C(682), C(683), C(684),
C(687), C(688), C(689), C(692), C(693), C(694), C(698), C(699), C(702), C(703), C(704), C(707), C(708),
C(709), C(712), C(713), C(714), C(718), C(719), C(723), C(724), C(728), C(729)
```

And the following 224 coded bits are moved to data block P<sub>B</sub>:

```
 C(16), C(25), C(26), C(30), C(31), C(35), C(40), C(41), C(45), C(50), C(51), C(55), C(56), C(60), C(65), C(66), C(70), C(75), C(76), C(80), C(81), C(85), C(90), C(91), C(95), C(100), C(101), C(105), C(106), C(110), C(115), C(116), C(120), C(125), C(126), C(130), C(131), C(135), C(140), C(141), C(145), C(150), C(151), C(155), C(156), C(160), C(165), C(166), C(170), C(175), C(176), C(180), C(181), C(185), C(190), C(191), C(195), C(200), C(201), C(205), C(206), C(210), C(215), C(216), C(220), C(225), C(226), C(230), C(231), C(235), C(240), C(241), C(245), C(250), C(251), C(255), C(256), C(260), C(265), C(266), C(270), C(275), C(276), C(280), C(325), C(326), C(330), C(331), C(335), C(340), C(341), C(345), C(346), C(350), C(351), C(355), C(356), C(360), C(361), C(365), C(366), C(370), C(371), C(375), C(376), C(380), C(381), C(385), C(386), C(390), C(391), C(395), C(396), C(400), C(401), C(405), C(406), C(410), C(411), C(415), C(416), C(420), C(421), C(425), C(426), C(430), C(431), C(435), C(436), C(440), C(441), C(445), C(446), C(450), C(451), C(455), C(456), C(460), C(461), C(465), C(466), C(470), C(471), C(475), C(476), C(480), C(481), C(485), C(486), C(490), C(491), C(495), C(496), C(497), C(501), C(505), C(506), C(511), C(515), C(515), C(515), C(515), C(515), C(511), C(515), C(515), C(511), C(515), C(515), C(511), C(51
```

 $C(516), C(521), C(522), C(526), C(530), C(531), C(536), C(540), C(541), C(546), C(547), C(551), C(555), \\ C(556), C(561), C(565), C(566), C(571), C(572), C(576), C(580), C(581), C(586), C(590), C(591), C(596), \\ C(597), C(601), C(605), C(606), C(611), C(615), C(616), C(621), C(622), C(626), C(630), C(631), C(636), \\ C(640), C(641), C(646), C(647), C(651), C(656), C(661), C(666), C(671), C(672), C(676), C(681), C(697) \\ C(640), C(64$ 

The vectors P<sub>G</sub> and P<sub>B</sub> of coded and punctured bits is combined with in band bits to vector P<sub>C</sub> as

$$P_{C}'(k) = ic(k)$$
 for  $k = 0, 1, 2, 3$ 

$$P_{C}'(k+4) = P_{G}(k)$$
 for  $k = 0, 1, ..., 223$ 

$$P_{C}'(k+224) = ic(k)$$
 for  $k = 4, 5, 6, 7$ 

$$P_{C}'(k+8) = P_{G}(k)$$
 for  $k = 224, 225, ..., 447$ 

$$P_{C}'(k+448) = ic(k)$$
 for  $k = 8, 9, 10, 11$ 

$$P_{C}'(k+460) = P_{B}(k)$$
 for  $k = 0, 1, ..., 223$ 

#### O-TCH/AHS5.9:

The block of 124 bits  $\{u(0)...u(123)\}$  is encoded with the 1/6 rate convolutional code defined by the following polynomials:

$$G4/G6 = 1 + D^2 + D^3 + D^5 + D^6/1 + D + D^2 + D^3 + D^4 + D^6$$

$$G4/G6 = 1 + D^2 + D^3 + D^5 + D^6/1 + D + D^2 + D^3 + D^4 + D^6$$

$$G5/G6 = 1 + D + D^4 + D^6/1 + D + D^2 + D^3 + D^4 + D^6$$

G6/G6 = 1

G6/G6 = 1

$$G7/G6 = 1 + D + D^2 + D^3 + D^6/1 + D + D^2 + D^3 + D^4 + D^6$$

resulting in 780 coded bits,  $\{C(0)...C(779)\}\$  defined by:

$$r(k) = u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6)$$

$$C(6k) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(6k+1) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(6k+2) = r(k) + r(k-1) + r(k-4) + r(k-6)$$

C(6k+3) = u(k)

$$C(6k+4) = u(k)$$

$$C(6k+5) = r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-6)$$
 for  $k = 0, 1, ..., 123$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$r(k) = 0$$

$$C(6k) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(6k+1) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(6k+2) = r(k) + r(k-1) + r(k-4) + r(k-6)$$

$$C(6k+3) = r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6)$$

$$C(6k+4) = r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6)$$

```
C(6k+5) = r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-6) for k = 124, 125, ..., 129
```

The following 448 coded bits are moved to data block P<sub>G</sub>:

```
C(4), C(9), C(10), C(13), C(14), C(15), C(16), C(19), C(20), C(21), C(22), C(25), C(26), C(27), C(28), C(31),
C(32), C(33), C(34), C(37), C(38), C(39), C(40), C(43), C(44), C(45), C(46), C(49), C(50), C(51), C(52),
C(55), C(56), C(57), C(58), C(61), C(62), C(63), C(64), C(67), C(68), C(69), C(70), C(73), C(74), C(75),
C(76), C(79), C(80), C(81), C(82), C(85), C(86), C(87), C(88), C(91), C(92), C(93), C(94), C(97), C(98),
C(99), C(100), C(103), C(104), C(105), C(106), C(109), C(110), C(111), C(112), C(115), C(116), C(117),
C(118), C(121), C(122), C(123), C(124), C(127), C(128), C(129), C(130), C(133), C(134), C(135), C(136),
C(139), C(140), C(141), C(142), C(145), C(146), C(147), C(148), C(151), C(152), C(153), C(154), C(157),
C(158), C(159), C(160), C(163), C(164), C(165), C(166), C(169), C(170), C(171), C(172), C(175), C(176),
C(177), C(178), C(181), C(182), C(183), C(184), C(187), C(188), C(189), C(190), C(193), C(194), C(195),
C(196), C(199), C(200), C(201), C(202), C(205), C(206), C(207), C(208), C(211), C(212), C(213), C(214),
C(217), C(218), C(219), C(220), C(223), C(224), C(225), C(226), C(229), C(230), C(231), C(232), C(235),
C(236), C(237), C(238), C(241), C(242), C(243), C(244), C(247), C(248), C(249), C(250), C(253), C(254),
C(255), C(256), C(259), C(260), C(261), C(262), C(265), C(266), C(267), C(268), C(271), C(272), C(273),
C(274), C(277), C(278), C(279), C(280), C(283), C(284), C(285), C(286), C(289), C(290), C(291), C(292),
C(295), C(296), C(297), C(298), C(301), C(302), C(303), C(304), C(307), C(308), C(309), C(310), C(313),
C(314), C(315), C(316), C(319), C(320), C(321), C(322), C(325), C(326), C(327), C(328), C(331), C(332),
C(333), C(334), C(337), C(338), C(339), C(340), C(343), C(344), C(345), C(346), C(349), C(350), C(351),
C(352), C(355), C(356), C(357), C(358), C(361), C(362), C(363), C(364), C(367), C(368), C(369), C(370),
C(373), C(374), C(375), C(376), C(379), C(380), C(381), C(382), C(385), C(386), C(387), C(388), C(391),
C(392), C(393), C(394), C(397), C(398), C(399), C(400), C(403), C(404), C(405), C(406), C(409), C(410),
C(411), C(412), C(415), C(416), C(417), C(418), C(421), C(422), C(423), C(424), C(427), C(428), C(429),
C(430), C(434), C(435), C(436), C(439), C(440), C(441), C(442), C(446), C(447), C(448), C(451), C(452),
C(453), C(454), C(458), C(459), C(460), C(463), C(464), C(465), C(466), C(470), C(471), C(471), C(472), C(475),
C(476), C(477), C(478), C(482), C(483), C(484), C(487), C(488), C(489), C(490), C(494), C(495), C(496),
C(499), C(500), C(501), C(502), C(506), C(507), C(508), C(511), C(513), C(514), C(518), C(519), C(520),
C(523), C(525), C(526), C(530), C(531), C(532), C(535), C(537), C(538), C(542), C(543), C(544), C(547),
C(549), C(550), C(554), C(555), C(556), C(559), C(561), C(562), C(566), C(567), C(568), C(571), C(573),
C(574), C(578), C(579), C(580), C(583), C(585), C(586), C(590), C(591), C(592), C(595), C(597), C(598),
C(602), C(603), C(604), C(607), C(609), C(610), C(614), C(615), C(616), C(619), C(621), C(622), C(626),
C(627), C(628), C(631), C(633), C(634), C(638), C(639), C(640), C(643), C(645), C(646), C(650), C(651),
C(652), C(655), C(657), C(658), C(662), C(663), C(664), C(667), C(669), C(670), C(674), C(675), C(676),
C(679), C(681), C(682), C(686), C(687), C(688), C(691), C(693), C(694), C(698), C(699), C(700), C(703),
C(705), C(706), C(710), C(711), C(712), C(717), C(718), C(722), C(723), C(724), C(729), C(730), C(736),
C(741), C(742), C(748), C(753), C(754), C(760), C(765), C(766), C(777), C(778)
```

And the following 224 coded bits are moved to data block P<sub>B</sub>:

```
C(3), C(7), C(8), C(12), C(18), C(23), C(24), C(30), C(35), C(36), C(42), C(47), C(48), C(54), C(59), C(60),
C(66), C(71), C(72), C(78), C(83), C(84), C(90), C(95), C(96), C(102), C(107), C(108), C(114), C(119),
C(120), C(126), C(131), C(132), C(138), C(143), C(144), C(150), C(155), C(156), C(162), C(167), C(168),
C(174), C(179), C(180), C(186), C(191), C(192), C(198), C(203), C(204), C(210), C(215), C(216), C(222),
C(227), C(228), C(234), C(239), C(240), C(246), C(251), C(252), C(258), C(263), C(264), C(270), C(275),
C(276), C(282), C(287), C(288), C(294), C(299), C(300), C(306), C(311), C(312), C(318), C(323), C(324),
C(330), C(335), C(336), C(342), C(347), C(348), C(354), C(359), C(360), C(366), C(371), C(372), C(378),
C(383), C(384), C(390), C(395), C(396), C(402), C(407), C(408), C(414), C(419), C(420), C(426), C(431),
C(432), C(433), C(438), C(443), C(444), C(445), C(450), C(455), C(456), C(457), C(462), C(467), C(468),
C(469), C(474), C(479), C(480), C(481), C(486), C(491), C(492), C(493), C(498), C(503), C(504), C(505),
C(512), C(515), C(516), C(517), C(524), C(527), C(528), C(529), C(536), C(539), C(540), C(541), C(548),
C(551), C(552), C(553), C(560), C(563), C(564), C(565), C(572), C(575), C(576), C(577), C(584), C(587),
C(588), C(589), C(596), C(599), C(600), C(601), C(608), C(611), C(612), C(613), C(620), C(623), C(624),
C(625), C(632), C(635), C(636), C(637), C(644), C(647), C(648), C(649), C(656), C(659), C(660), C(661),
C(668), C(671), C(672), C(673), C(680), C(683), C(684), C(685), C(692), C(695), C(696), C(697), C(704),
C(707), C(708), C(709), C(715), C(716), C(720), C(721), C(727), C(728), C(734), C(735), C(739), C(740),
C(746), C(747), C(751), C(752), C(758), C(759), C(763), C(764), C(770), C(771), C(772), C(775)
```

The vectors P<sub>G</sub> and P<sub>B</sub> of coded and punctured bits is combined with in band bits to vector P<sub>C</sub> as

$$\begin{split} P_{C^{'}}(k) &= ic(k) & \text{for } k = 0,\, 1,\, 2,\, 3 \\ P_{C^{'}}(k+4) &= P_{G}(k) & \text{for } k = 0,\, 1,\, ...,\, 223 \\ P_{C^{'}}(k+224) &= ic(k) & \text{for } k = 4,\, 5,\, 6,\, 7 \\ P_{C^{'}}(k+8) &= P_{G}(k) & \text{for } k = 224,\, 225,\, ...,\, 447 \\ P_{C^{'}}(k+448) &= ic(k) & \text{for } k = 8,\, 9,\, 10,\, 11 \\ P_{C^{'}}(k+460) &= P_{B}(k) & \text{for } k = 0,\, 1,\, ...,\, 223 \end{split}$$

#### **O-TCH/AHS5.15**:

The block of 109 bits  $\{u(0)...u(108)\}$  is encoded with the 1/6 rate convolutional code defined by the following polynomials:

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G5/G7 = 1 + D + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G6/G7 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G6/G7 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G7/G7 = 1$$

resulting in 690 coded bits,  $\{C(0)...C(689)\}\$  defined by:

$$\begin{split} r(k) &= u(k) + r(k\text{-}1) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}6) \\ C(6k) &= r(k) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}5) + r(k\text{-}6) \\ C(6k+1) &= r(k) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}5) + r(k\text{-}6) \\ C(6k+2) &= r(k) + r(k\text{-}1) + r(k\text{-}4) + r(k\text{-}6) \\ C(6k+3) &= r(k) + r(k\text{-}1) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}4) + r(k\text{-}6) \\ C(6k+4) &= r(k) + r(k\text{-}1) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}4) + r(k\text{-}6) \\ C(6k+5) &= u(k) \end{split}$$

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(6k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(6k+1) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(6k+2) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(6k+3) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(6k+4) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(6k+5) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) & \text{for } k = 109, 110, ..., 114 \\ \end{split}$$

The following 448 coded bits are moved to data block P<sub>G</sub>:

C(5), C(8), C(9), C(11), C(14), C(15), C(17), C(20), C(21), C(23), C(26), C(27), C(29), C(30), C(32), C(35), C(36), C(38), C(38), C(39), C(41), C(42), C(44), C(45), C(47), C(48), C(50), C(51), C(53), C(54), C(56), C(5

```
C(57), C(59), C(60), C(61), C(62), C(63), C(65), C(66), C(68), C(69), C(71), C(72), C(73), C(74), C(75),
C(77), C(78), C(80), C(81), C(83), C(84), C(86), C(87), C(89), C(90), C(92), C(93), C(95), C(96), C(98),
C(99), C(101), C(102), C(104), C(105), C(107), C(108), C(109), C(110), C(111), C(113), C(114), C(116),
C(117), C(119), C(120), C(121), C(122), C(123), C(125), C(126), C(128), C(129), C(131), C(132), C(134),
C(135), C(137), C(138), C(140), C(141), C(143), C(144), C(146), C(147), C(149), C(150), C(152), C(153),
C(155), C(156), C(157), C(158), C(159), C(161), C(162), C(164), C(165), C(167), C(168), C(169), C(170),
C(171), C(173), C(174), C(176), C(177), C(179), C(180), C(182), C(183), C(185), C(186), C(188), C(189),
C(191), C(192), C(194), C(195), C(197), C(198), C(200), C(201), C(203), C(204), C(205), C(206), C(207),
C(209), C(210), C(212), C(213), C(215), C(216), C(217), C(218), C(219), C(221), C(222), C(224), C(225),
C(227), C(228), C(230), C(231), C(233), C(234), C(236), C(237), C(239), C(240), C(242), C(243), C(245), C(24
C(246), C(248), C(249), C(251), C(252), C(253), C(254), C(255), C(257), C(258), C(260), C(261), C(263),
C(264), C(265), C(266), C(267), C(269), C(270), C(272), C(273), C(275), C(276), C(278), C(279), C(281),
C(282), C(284), C(285), C(287), C(288), C(290), C(291), C(293), C(294), C(296), C(297), C(299), C(300),
C(301), C(302), C(303), C(305), C(306), C(308), C(309), C(311), C(312), C(313), C(314), C(315), C(317),
C(318), C(320), C(321), C(323), C(324), C(326), C(327), C(329), C(330), C(332), C(333), C(335), C(336),
C(338), C(339), C(341), C(342), C(344), C(345), C(347), C(348), C(349), C(350), C(351), C(353), C(354),
C(356), C(357), C(359), C(360), C(361), C(362), C(363), C(365), C(366), C(368), C(369), C(371), C(372),
C(374), C(375), C(377), C(378), C(380), C(381), C(383), C(384), C(386), C(387), C(389), C(390), C(392),
C(393), C(395), C(396), C(397), C(398), C(399), C(401), C(402), C(404), C(405), C(407), C(408), C(409),
C(410), C(411), C(413), C(414), C(416), C(417), C(419), C(420), C(422), C(423), C(425), C(426), C(428),
C(429), C(431), C(432), C(434), C(435), C(437), C(438), C(440), C(441), C(443), C(444), C(446), C(447),
C(449), C(450), C(452), C(453), C(455), C(456), C(458), C(459), C(461), C(462), C(464), C(465), C(467),
C(468), C(470), C(471), C(473), C(474), C(476), C(477), C(479), C(480), C(482), C(483), C(485), C(486),
C(488), C(489), C(491), C(494), C(495), C(497), C(498), C(500), C(501), C(503), C(504), C(506), C(507),
C(509), C(510), C(512), C(513), C(515), C(518), C(519), C(521), C(522), C(524), C(525), C(527), C(528),
C(530), C(531), C(533), C(534), C(536), C(537), C(539), C(542), C(543), C(545), C(546), C(548), C(549),
C(551), C(552), C(554), C(555), C(557), C(558), C(560), C(561), C(563), C(566), C(567), C(569), C(570),
C(572), C(573), C(575), C(576), C(578), C(579), C(581), C(582), C(584), C(585), C(587), C(590), C(591),
C(593), C(594), C(596), C(597), C(599), C(600), C(602), C(603), C(605), C(606), C(608), C(609), C(611),
C(614), C(615), C(617), C(618), C(620), C(621), C(623), C(624), C(626), C(627), C(629), C(630), C(632),
C(633), C(635), C(638), C(639), C(641), C(642), C(644), C(645), C(647), C(648), C(650), C(651), C(653),
C(654), C(656), C(657), C(662), C(663), C(666), C(668), C(669), C(671), C(675)
```

And the following 224 coded bits are moved to data block P<sub>B</sub>:

```
C(2), C(3), C(13), C(24), C(28), C(31), C(34), C(37), C(40), C(43), C(46), C(49), C(52), C(55), C(58), C(64),
C(67), C(70), C(76), C(79), C(82), C(85), C(88), C(91), C(94), C(97), C(100), C(103), C(106), C(112),
C(115), C(118), C(124), C(127), C(130), C(133), C(136), C(139), C(142), C(145), C(148), C(151), C(154),
C(160), C(163), C(166), C(172), C(175), C(178), C(181), C(184), C(187), C(190), C(193), C(196), C(199),
C(202), C(208), C(211), C(214), C(220), C(223), C(226), C(229), C(232), C(235), C(238), C(241), C(244),
C(247), C(250), C(256), C(259), C(262), C(268), C(271), C(274), C(277), C(280), C(283), C(286), C(289),
C(292), C(295), C(298), C(304), C(307), C(310), C(316), C(319), C(322), C(325), C(328), C(331), C(334),
C(337), C(340), C(343), C(346), C(352), C(355), C(358), C(364), C(367), C(370), C(373), C(376), C(379),
C(382), C(385), C(388), C(391), C(394), C(400), C(403), C(406), C(412), C(415), C(418), C(421), C(424),
C(427), C(430), C(433), C(436), C(439), C(442), C(445), C(448), C(451), C(454), C(457), C(460), C(463),
C(466), C(469), C(472), C(475), C(478), C(481), C(484), C(487), C(490), C(492), C(493), C(496), C(499),
C(502), C(505), C(508), C(511), C(514), C(516), C(517), C(520), C(523), C(526), C(529), C(532), C(535),
C(538), C(540), C(541), C(544), C(547), C(550), C(553), C(556), C(559), C(562), C(564), C(565), C(568),
C(571), C(574), C(577), C(580), C(583), C(586), C(588), C(589), C(592), C(595), C(598), C(601), C(604),
C(607), C(610), C(612), C(613), C(616), C(619), C(622), C(625), C(628), C(631), C(634), C(636), C(637),
C(640), C(643), C(646), C(649), C(652), C(655), C(658), C(659), C(660), C(661), C(664), C(665), C(667),
C(670), C(672), C(673), C(676), C(677), C(681), C(682), C(683), C(686), C(687), C(688), C(689)
```

The vectors P<sub>G</sub> and P<sub>B</sub> of coded and punctured bits is combined with in band bits to vector P<sub>C</sub>' as

```
P_{C'}(k) = ic(k) for k = 0, 1, 2, 3
```

$$P_{C}'(k+4) = P_{G}(k)$$
 for  $k = 0, 1, ..., 223$ 

$$\begin{split} P_{C^{'}}(k+224) &= ic(k) & \text{for } k=4,5,6,7 \\ P_{C^{'}}(k+8) &= P_{G}(k) & \text{for } k=224,225,...,447 \\ P_{C^{'}}(k+448) &= ic(k) & \text{for } k=8,9,10,11 \\ P_{C^{'}}(k+460) &= P_{B}(k) & \text{for } k=0,1,...,223 \end{split}$$

#### **O-TCH/AHS4.75**:

The block of 101 bits  $\{u(0)...u(100)\}$  is encoded with the 1/7 rate convolutional code defined by the following polynomials:

$$\begin{aligned} G4/G7 &= 1 + D^2 + D^3 + D^5 + D^6/1 + D + D^2 + D^3 + D^6 \\ G4/G7 &= 1 + D^2 + D^3 + D^5 + D^6/1 + D + D^2 + D^3 + D^6 \\ G5/G7 &= 1 + D + D^4 + D^6/1 + D + D^2 + D^3 + D^6 \\ G5/G7 &= 1 + D + D^4 + D^6/1 + D + D^2 + D^3 + D^6 \\ G6/G7 &= 1 + D + D^2 + D^3 + D^4 + D^6/1 + D + D^2 + D^3 + D^6 \\ G6/G7 &= 1 + D + D^2 + D^3 + D^4 + D^6/1 + D + D^2 + D^3 + D^6 \\ G7/G7 &= 1 \end{aligned}$$

resulting in 749 coded bits,  $\{C(0)...C(748)\}\$  defined by:

$$\begin{split} r(k) &= u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-6) \\ C(7k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(7k+1) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(7k+2) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(7k+3) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(7k+4) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(7k+5) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(7k+6) &= u(k) \end{split}$$

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(7k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(7k+1) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(7k+2) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(7k+3) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(7k+4) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(7k+5) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(7k+6) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) & \text{for } k = 101, 102, ..., 106 \end{split}$$

The following 448 coded bits are moved to data block P<sub>G</sub>:

$$C(6), C(13), C(15), C(18), C(20), C(22), C(25), C(27), C(30), C(32), C(34), C(36), C(37), C(39), C(41), \\ C(42), C(43), C(44), C(46), C(48), C(49), C(51), C(53), C(55), C(56), C(57), C(58), C(60), C(62), C(63), \\ C(48), C(48$$

```
C(64), C(65), C(67), C(69), C(70), C(72), C(74), C(76), C(77), C(78), C(79), C(81), C(83), C(84), C(85),
C(86), C(88), C(90), C(91), C(93), C(95), C(97), C(98), C(99), C(100), C(102), C(104), C(105), C(106),
C(107), C(109), C(111), C(112), C(114), C(116), C(118), C(119), C(120), C(121), C(123), C(125), C(126),
C(127), C(128), C(130), C(132), C(133), C(135), C(137), C(139), C(140), C(141), C(142), C(144), C(146),
C(147), C(148), C(149), C(151), C(153), C(154), C(156), C(158), C(160), C(161), C(162), C(163), C(165),
C(167), C(168), C(169), C(170), C(172), C(174), C(175), C(177), C(179), C(181), C(182), C(183), C(184),
C(186), C(188), C(189), C(190), C(191), C(193), C(195), C(196), C(198), C(200), C(202), C(203), C(204),
C(205), C(207), C(209), C(210), C(211), C(212), C(214), C(216), C(217), C(219), C(221), C(223), C(224),
C(225), C(226), C(228), C(230), C(231), C(232), C(233), C(235), C(237), C(238), C(240), C(242), C(244),
C(245), C(246), C(247), C(249), C(251), C(252), C(253), C(254), C(256), C(258), C(259), C(261), C(263),
C(265), C(266), C(267), C(268), C(270), C(272), C(273), C(274), C(275), C(277), C(279), C(280), C(282),
C(284), C(286), C(287), C(288), C(289), C(291), C(293), C(294), C(295), C(296), C(298), C(300), C(301),
C(303), C(305), C(307), C(308), C(309), C(310), C(312), C(314), C(315), C(316), C(317), C(319), C(321),
C(322), C(324), C(326), C(328), C(329), C(330), C(331), C(333), C(335), C(336), C(337), C(338), C(340),
C(342), C(343), C(345), C(347), C(349), C(350), C(351), C(352), C(354), C(356), C(357), C(358), C(359),
C(361), C(363), C(364), C(366), C(368), C(370), C(371), C(372), C(373), C(375), C(377), C(378), C(379),
C(380), C(382), C(384), C(385), C(387), C(389), C(391), C(392), C(393), C(394), C(396), C(398), C(399),
C(400), C(401), C(403), C(405), C(406), C(408), C(410), C(412), C(413), C(414), C(415), C(417), C(419),
C(420), C(421), C(422), C(424), C(426), C(427), C(429), C(431), C(433), C(434), C(436), C(438), C(440),
C(441), C(443), C(445), C(447), C(448), C(450), C(452), C(454), C(455), C(456), C(457), C(459), C(461),
C(462), C(463), C(464), C(466), C(468), C(469), C(471), C(473), C(475), C(476), C(478), C(480), C(482),
C(483), C(485), C(487), C(489), C(490), C(492), C(494), C(496), C(497), C(498), C(499), C(501), C(503),
C(504), C(505), C(506), C(508), C(510), C(511), C(513), C(515), C(517), C(518), C(520), C(522), C(524),
C(525), C(527), C(529), C(531), C(532), C(534), C(536), C(538), C(539), C(540), C(541), C(543), C(545),
C(546), C(547), C(548), C(550), C(552), C(553), C(555), C(557), C(559), C(560), C(562), C(564), C(566),
C(567), C(569), C(571), C(573), C(574), C(576), C(578), C(580), C(581), C(582), C(583), C(585), C(587),
C(588), C(590), C(592), C(594), C(595), C(597), C(599), C(601), C(602), C(604), C(606), C(608), C(609),
C(611), C(613), C(615), C(616), C(618), C(620), C(622), C(623), C(625), C(627), C(629), C(630), C(632),
C(634), C(636), C(637), C(639), C(641), C(643), C(644), C(646), C(648), C(650), C(653), C(655), C(657),
C(658), C(660), C(662), C(664), C(667), C(669), C(671), C(674), C(676), C(678), C(679), C(681), C(683),
C(685), C(688), C(690), C(692), C(695), C(697), C(699), C(700), C(702), C(704), C(706), C(709), C(711),
C(713), C(716), C(718), C(720), C(721), C(723), C(725), C(727), C(730), C(732), C(734), C(742)
```

And the following 224 coded bits are moved to data block P<sub>B</sub>:

```
C(1), C(4), C(8), C(9), C(11), C(24), C(26), C(29), C(31), C(33), C(40), C(45), C(47), C(50), C(52), C(54),
C(61), C(68), C(71), C(73), C(75), C(80), C(82), C(87), C(89), C(92), C(94), C(96), C(103), C(110), C(113),
C(115), C(117), C(122), C(124), C(129), C(131), C(134), C(136), C(138), C(145), C(152), C(155), C(157),
C(159), C(164), C(166), C(171), C(173), C(176), C(178), C(180), C(187), C(194), C(197), C(199), C(201),
C(206), C(208), C(213), C(215), C(218), C(220), C(222), C(229), C(236), C(239), C(241), C(243), C(248),
C(250), C(255), C(257), C(260), C(262), C(264), C(271), C(278), C(281), C(283), C(285), C(292), C(299),
C(302), C(304), C(306), C(313), C(320), C(323), C(325), C(327), C(334), C(341), C(344), C(346), C(348),
C(355), C(362), C(365), C(367), C(369), C(376), C(383), C(386), C(388), C(390), C(397), C(404), C(407),
C(409), C(411), C(418), C(425), C(428), C(430), C(432), C(435), C(439), C(442), C(446), C(449), C(451),
C(453), C(460), C(467), C(470), C(472), C(474), C(477), C(481), C(484), C(488), C(491), C(493), C(495),
C(502), C(509), C(512), C(514), C(516), C(519), C(523), C(526), C(530), C(533), C(535), C(537), C(544),
C(551), C(554), C(556), C(558), C(561), C(565), C(568), C(572), C(575), C(577), C(579), C(586), C(589),
C(593), C(596), C(598), C(600), C(603), C(607), C(610), C(614), C(617), C(619), C(621), C(624), C(628),
C(631), C(635), C(638), C(640), C(642), C(645), C(649), C(651), C(652), C(656), C(659), C(661), C(663),
C(665), C(666), C(670), C(672), C(673), C(677), C(680), C(682), C(684), C(686), C(687), C(691), C(693),
C(694), C(698), C(701), C(703), C(705), C(707), C(708), C(712), C(714), C(715), C(719), C(722), C(724),
C(726), C(728), C(729), C(733), C(737), C(739), C(741), C(743), C(744), C(746), C(748)
```

The vectors P<sub>G</sub> and P<sub>B</sub> of coded and punctured bits is combined with in band bits to vector P<sub>C</sub>' as

```
P_{C'}(k) = ic(k) for k = 0, 1, 2, 3
```

$$P_{C}'(k+4) = P_{G}(k)$$
 for  $k = 0, 1, ..., 223$ 

$$\begin{split} P_{C^{'}}(k+224) &= ic(k) & \text{for } k=4,5,6,7 \\ P_{C^{'}}(k+8) &= P_{G}(k) & \text{for } k=224,225,...,447 \\ P_{C^{'}}(k+448) &= ic(k) & \text{for } k=8,9,10,11 \\ P_{C^{'}}(k+460) &= P_{B}(k) & \text{for } k=0,1,...,223 \end{split}$$

#### 3.15.7.5 Interleaving

Before interleaving the bits  $\{P_C'(0) \dots P_C'(683)\}$  are converted into 3-bit symbols  $\{c(0) \dots c(227)\}$  according to table 1 in 3GPP TS 45.004, the symbol c(k) consists of  $d_{3k}=P_C'(k)$ ,  $d_{3k+1}=P_C'(k+228)$  and  $d_{3k+2}=P_C'(k+456)$  for  $k=0,1,\dots,227$ . The interleaving is done as specified for the TCH/HS in subclause 3.2.3. The difference is that the interleaving is done by symbols instead of single bits, reusing the existing interleaving tables.

## 3.15.7.6 Mapping on a Burst

The mapping is given by the rule:

$$e(B,j)=i(B,j) \mbox{ and } e(B,59+j)=i(B,57+j) \mbox{ for } j=0,1,...,56$$
 and 
$$e(B,57)=hl(B) \mbox{ and } e(B,58)=hu(B)$$

The two symbols, labelled hl(B) and hu(B) on burst number B are flags used for indication of control channel signalling. For each O-TCH/AHS block not stolen for signalling purposes:

 $hu(B) = \{0,0,0\}$  for the first 2 bursts (indicating status of the even numbered symbols)

 $hl(B) = \{0,0,0\}$  for the last 2 bursts (indicating status of the odd numbered symbols)

where {0,0,0} is the mapping of the three bits 0,0,0 onto a 3-bit symbol according to table 1 in 3GPP TS 45.004.

For the use of hl(B) and hu(B) when a speech frame is stolen for signalling purposes, see subclause 4.11.6.

# 3.15.8 RATSCCH MARKER

This frame type contains the in-band channel and an identification marker. The in-band data id(0,1) represents Mode Indication or Mode Command/Mode Request depending on the current frame number.

# 3.15.8.1 Coding of in-band data

The coding of in-band data is done as specified for the RATSCCH\_MARKER frame at half rate in subclause 3.10.8.1.

#### 3.15.8.2 Identification marker

The identification marker is done as specified for the RATSCCH\_MARKER frame at half rate in subclause 3.10.8.2.

# 3.15.8.3 Interleaving

Before interleaving the bits are repeated 3 times:

$$c'(3k+2) = c'(3k+1) = c'(3k) = c(k)$$
 for k=0,...,227

The bits  $\{c'(0) \dots c'(683)\}$  are then converted into 3-bit symbols  $\{C(0) \dots C(227)\}$  according to table 1 in 3GPP TS 45.004, the symbol C(k) depends on c'(3k+2), c'(3k+1) and c'(3k) for  $k=0,1,\dots,227$ .

The interleaving is done as specified for the TCH/HS in subclause 3.2.3. The difference is that the interleaving is done by symbols instead of single bits, reusing the existing interleaving tables.

# 3.15.8.4 Mapping on a Burst

The mapping is done as specified for the TCH/HS in subclause 3.2.4 with exception that it is done by symbols instead of single bits.

# 3.15.9 RATSCCH\_DATA

This frame contains the RATSCCH data and an inband channel. The RATSCCH data consists of 35 bits. The in-band data id(0,1) represents Mode Indication or Mode Command/Mode Request depending on the current frame number.

## 3.15.9.1 Coding of in-band data

The coding of in-band data is done as specified for the RATSCCH\_DATA frame at half rate in subclause 3.10.9.1.

## 3.15.9.2 Parity and convolutional encoding for the RATSCCH message

The parity and convolutional encoding for the RATSCCH message are done as specified for the RATSCCH\_DATA frame at half rate in subclause 3.10.9.2.

# 3.15.9.3 Interleaving

Before interleaving the bits are repeated 3 times:

$$c'(3k+2) = c'(3k+1) = c'(3k) = c(k)$$
 for  $k=0,...,227$ 

The bits  $\{c'(0) \dots c'(683)\}$  are then converted into 3-bit symbols  $\{C(0) \dots C(227)\}$  according to table 1 in 3GPP TS 45.004, the symbol C(k) depends on c'(3k+2), c'(3k+1) and c'(3k) for  $k=0,1,\dots,227$ .

The interleaving is done as specified for the TCH/HS in subclause 3.2.3. The difference is that the interleaving is done by symbols instead of single bits, reusing the existing interleaving tables.

# 3.15.9.4 Mapping on a Burst

The mapping is done as specified for the TCH/HS in subclause 3.2.4 with exception that it is done by symbols instead of single bits.

# 3.16 Wideband Adaptive multi rate speech channel at 8-PSK full rate (O-TCH/WFS)

This section describes the coding for the different frame formats used for O-TCH/WFS. The formats used are (in the order they are described):

SID UPDATE Used to convey comfort noise parameters during DTX

SID\_FIRST Marker to define end of speech, start of DTX

ONSET Used to signal the Codec mode for the first speech frame after DTX

SPEECH Speech frames

RATSCCH Frames used to convey RATSCCH messages

In this chapter, sub chapters 3.16.1 to 3.16.5 describe the channel coding for the different formats listed above.

Common to all the formats is that in-band information is conveyed, the coding for the in-band channel is described in the table below.

Identifier (defined in 3GPP TS 45.009)	Received in-band data id(1), id(0)	Encoded in-band data for SID and RATSCCH frames ic(15),, ic(0)	Encoded in-band data for speech frames ic(23),, ic(0)
CODEC_MODE_1	00	0101001100001111	000000000000000000000000000000000000000
CODEC_MODE_2	01	0011111010111000	110110101110110110101110
CODEC_MODE_3	10	1000100001100011	101101110101101101110101
CODEC_MODE_4	11	1110010111010100	011011011011011011011

# 3.16.1 SID UPDATE

The speech encoder delivers 35 bits of comfort noise parameters. Also delivered is two in-band channels, id0(0,1) and id1(0,1), id0 corresponding to Mode Commands or Mode Requests and id1 to Mode Indication. The general coding is as: the two in-band data channels are coded to 16 bits each, a 14-bit CRC is added to the 35 CN bits which are then coded by a rate ½ RSC coder to 212 bits. Finally a 212 bits identification field is added thereby giving a total size of 456 bits. These 456 bits are then block interleaved in the same way as SACCH frames.

## 3.16.1.1 Coding of in-band data

The coding of in-band data is done as specified for the SID\_UPDATE frame in TCH/AFS (subclause 3.9.1.1).

## 3.16.1.2 Parity and convolutional encoding for the comfort noise parameters

The parity and convolutional encoding for the comfort noise parameters are done as specified for the SID\_UPDATE frame in TCH/AFS (subclause 3.9.1.2).

#### 3.16.1.3 Identification marker

The identification marker is constructed as specified for the SID\_UPDATE frame in TCH/AFS (subclause 3.9.1.3).

#### 3.16.1.4 Repetition

The coded bits (c) are repeated according to the following rule:

$$c'(3k+2) = c'(3k+1) = c'(3k) = c(k)$$
 for  $k=0,...,455$ 

## 3.16.1.5 Interleaving

Before interleaving the coded bits  $\{c'(0) \dots c'(1367)\}$  are converted into 3-bit symbols  $\{C(0) \dots C(455)\}$  according to table 1 in 3GPP TS 45.004, the symbol C(k) depends on c'(3k+2), c'(3k+1) and c'(3k) for  $k=0,1,\dots,455$ .

The interleaving is done as specified for the SID\_UPDATE frame in TCH/AFS (subclause 3.9.1.4). The difference is that the interleaving is done by symbols instead of single bits, reusing the existing interleaving tables.

## 3.16.1.6 Mapping on a Burst

The mapping is done as specified for the SID\_UPDATE frame in TCH/AFS (subclause 3.9.1.5) with exception that it is done by symbols instead of single bits.

# 3.16.2 SID FIRST

This frame type contains no source data from the speech coder, what is transmitted is the in-band channel (signalling Mode Indication or Mode Command/Mode Request depending on the current frame number) and an identification marker.

# 3.16.2.1 Coding of in-band data

The coding of in-band data is done as specified for the SID\_FIRST frame in TCH/AFS (subclause 3.9.2.1).

#### 3.16.2.2 Identification marker

The identification marker is constructed as specified for the SID\_FIRST frame in TCH/AFS (subclause 3.9.2.2).

#### 3.16.2.3 Repetition

The coded bits (c) are repeated according to the following rule:

$$c'(3k+2) = c'(3k+1) = c'(3k) = c(k)$$
 for  $k=0,...,227$ 

#### 3.16.2.4 Interleaving

Before interleaving the coded bits  $\{c'(0) \dots c'(683)\}$  are converted into 3-bit symbols  $\{C(0) \dots C(227)\}$  according to table 1 in 3GPP TS 45.004, the symbol C(k) depends on c'(3k+2), c'(3k+1) and c'(3k) for  $k=0,1,\dots,227$ .

The interleaving is done as specified for the in TCH/AFS (subclause 3.9.2.3). The difference is that the interleaving is done by symbols instead of single bits, reusing the existing interleaving tables.

## 3.16.2.5 Mapping on a Burst

The mapping is done as specified for the TCH/AFS (subclause 3.9.2.4) with exception that it is done by symbols instead of single bits.

## 3.16.3 ONSET

Onset frames are used to preset the interleaver buffer after a period of no speech activity in DTX mode. This frame type contains no source data from the speech coder. What is transmitted is the in-band channel signalling the Mode Indication for the speech frame following the onset marker.

## 3.16.3.1 Coding of in-band data

The coding of in-band data is done as specified for the ONSET frame in TCH/AFS (subclause 3.9.3.1).

#### 3.16.3.2 Repetition

The coded bits (c) are repeated according to the following rule:

$$c'(3k+2) = c'(3k+1) = c'(3k) = c(k)$$
 for  $k=0,...,227$ 

# 3.16.3.3 Interleaving

Before interleaving the coded bits  $\{c'(0) \dots c'(683)\}$  are converted into 3-bit symbols  $\{C(0) \dots C(227)\}$  according to table 1 in 3GPP TS 45.004, the symbol C(k) depends on c'(3k+2), c'(3k+1) and c'(3k) for  $k=0,1,\dots,227$ .

The interleaving is done as specified for the ONSET frame in TCH/AFS (subclause 3.9.3.2). The difference is that the interleaving is done by symbols instead of single bits, reusing the existing interleaving tables. The result of the interleaving is a distribution of 114 of the reordered 228 symbols of a given data block over 2 blocks using the odd numbered bits. The even numbered symbols of these 2 blocks will be filled by the speech frame that following immediately after this frame.

# 3.16.3.4 Mapping on a Burst

The mapping is done as specified for the ONSET frame in TCH/AFS (subclause 3.9.3.3) with exception that it is done by symbols instead of single bits.

#### 3.16.4 SPEECH

The speech coder delivers to the channel encoder a sequence of blocks of data. One block of data corresponds to one speech frame and the block length is different in each of the nine channel codec modes. Adjoining each block of data is

information of the channel codec mode to use when encoding the block. Also delivered is the in-band data id(0,1) representing Mode Indication or Mode Command/Mode Request depending on the current frame number.

#### 3.16.4.1 Coding of the in-band data

The two input in-band bits (id(0,1)) are coded to twenty four coded in-band bits (ic(0..23)).

The encoded in-band bits are moved to the coded bits, c, as

$$c(k) = ic(k)$$
 for  $k = 0, 1, ..., 23$ .

## 3.16.4.2 Ordering according to subjective importance

The bits delivered by the speech encoder,  $\{s(1), s(2), ..., s(K_s)\}$ , are rearranged according to subjective importance before channel coding. Tables 16 to 20 define the correct rearrangement for the speech codec modes 12.65 kbit/s, 8.85 kbit/s 6.60 kbit/s, 23.85 kbit/s and 15.85 kbit/s, respectively. In the tables speech codec parameters are numbered in the order they are delivered by the corresponding speech encoder according to 3GPP TS 26.190 and the rearranged bits are labelled  $\{d(0),d(1),...,d(K_{d}-1)\}$ , defined in the order of decreasing importance. Index  $K_d$  refers to the number of bits delivered by the speech encoder, see below:

Codec mode	Number of speech bits delivered per block (K <sub>d</sub> )
O-TCH/WFS23.85	477
O-TCH/WFS15.85	317
O-TCH/WFS12.65	253
O-TCH/WFS8.85	177
O-TCH/WFS6.60	132

The ordering algorithm is in pseudo code as:

for 
$$j = 0$$
 to  $K_{d-1}$   $d(j) := s(table(j)+1)$ ; where table(j) is read line by line left to right

The rearranged bits are further divided into two different classes to perform unequal error protection for different bits according to subjective importance.

The protection classes are:

1a - Data protected with the CRC and the convolution code.

1b - Data protected with the convolution code.

The number of class 1 (sum of class 1a and 1b), class 1a and class 1b bits for each codec mode is shown below:

Codec mode	Number of speech bits delivered per block	Number of class 1 bits per block	Number of Class 1a bits per block	Number of class 1b bits per block
O-TCH/WFS23.85	477	477	72	405
O-TCH/WFS15.85	317	317	72	245
O-TCH/WFS12.65	253	253	72	181
O-TCH/WFS8.85	177	177	64	113
O-TCH/WFS6.60	132	132	54	78

## 3.16.4.3 Parity for speech frames

The basic parameters for each codec mode for the first encoding step are shown below:

Codec mode	Number of class 1 bits (K <sub>d1</sub> )	CRC Protected bits (K <sub>d1a</sub> )	CRC bits	Number of bits after first encoding step (K <sub>u</sub> = K <sub>d</sub> + 6)
O-TCH/WFS23.85	477	72	6	483
O-TCH/WFS15.85	317	72	6	323
O-TCH/WFS12.65	253	72	6	259
O-TCH/WFS8.85	177	64	6	183
O-TCH/WFS6.60	132	54	6	138

A 6-bit CRC is used for error-detection. These parity bits are generated by the cyclic generator polynomial:  $g(D) = D^6 + D^5 + D^3 + D^2 + D^1 + I$  from the first  $K_{d1a}$  bits of class 1, where  $K_{d1a}$  refers to number of bits in protection class 1a as shown above for each codec mode. The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

$$d(0)D(K_{d1a} + 5) + d(1)D(K_{d1a} + 4) + ... + d(K_{d1a} - 1)D(6) + p(0)D(5) + ... + p(4)D + p(5)$$

where p(0), p(1) ... p(5) are the parity bits, when divided by g(D), yields a remainder equal to:

$$1 + D + D^2 + D^3 + D^4 + D^5$$
.

The information and parity bits are merged:

u(k) = d(k)	for $k = 0, 1,, K_{d1a}$ -1
u(K) = u(K)	101 K = 0, 1,, $N_{d1a}$

$$u(k) = p(k - K_{d1a}) \hspace{1cm} \text{for } k = K_{d1a}, \, K_{d1a} + 1, \, ..., \, K_{d1a} + 5$$

$$u(k) = d(k-6)$$
 for  $k = K_{d1a}+6, K_{d1a}+7, ..., K_{u-1}$ 

#### O-TCH/WFS23.85:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 71$ 

$$u(k) = p(k-72)$$
 for  $k = 72, 73, ..., 77$ 

$$u(k) = d(k-6)$$
 for  $k = 78, 79, ..., 482$ 

#### O-TCH/WFS15.85:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 71$ 

$$u(k) = p(k-72)$$
 for  $k = 72, 73, ..., 77$ 

$$u(k) = d(k-6)$$
 for  $k = 78, 79, ..., 322$ 

#### O-TCH/WFS12.65:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 71$ 

$$u(k) = p(k-72)$$
 for  $k = 72, 73, ..., 77$ 

$$u(k) = d(k-6)$$
 for  $k = 78, 79, ..., 258$ 

#### **O-TCH/WFS8.85**:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 63$ 

$$u(k) = p(k-64)$$
 for  $k = 64, 65, ..., 69$ 

$$u(k) = d(k-6)$$
 for  $k = 70, 71, ..., 182$ 

#### O-TCH/WFS6.60:

$$u(k) = d(k)$$
 for  $k = 0, 1, ..., 53$ 

$$u(k) = p(k-54)$$
 for  $k = 54, 55, ..., 59$ 

$$u(k) = d(k-6)$$
 for  $k = 60, 61, ..., 137$ 

#### 3.16.4.4 Convolutional encoder

The bits from the first encoding step (u(k)) are encoded with the recursive systematic convolutional codes as summarised below. The number of output bits after puncturing is 1344 for all codec modes.

Codec Mode	Rate	Number of input bits to conv. coder	Number of output bits from conv. Coder	Number Of Punctured bits
O-TCH/WFS23.85	1/3	483	1467	123
O-TCH/WFS15.85	1/5	323	1645	301
O-TCH/WFS12.65	1/6	259	1590	246
O-TCH/WFS8.85	1/8	183	1512	168
O-TCH/WFS6.60	1/10	138	1440	96

Below the coding for each codec mode is specified in detail. The puncturing for each mode is designed to give an even protection of the class 1A bits while the protection within class 1B is not equal to reflect the individual error sensitivity of the class 1B bits.

#### O-TCH/WFS23.85:

The block of 483 bits  $\{u(0)...u(482)\}$  is encoded with the 1/3 rate convolutional code defined by the following

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$
 
$$G5/G7 = 1 + D + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$
 
$$G7/G7 = 1$$

resulting in 1467 coded bits,  $\{C(0)...C(1466)\}\$  defined by:

$$\begin{split} r(k) &= u(k) + r(k\text{-}1) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}6) \\ C(3k) &= r(k) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}5) + r(k\text{-}6) \\ C(3k+1) &= r(k) + r(k\text{-}1) + r(k\text{-}4) + r(k\text{-}6) \\ C(3k+2) &= u(k) \end{split} \qquad \qquad \text{for } k = 0, 1, ..., 482; \qquad r(k) = 0 \text{ for } k < 0 \end{split}$$

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(3k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(3k+1) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(3k+2) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) & \text{for } k = 483, 484, ..., 488 \end{split}$$

The following 896 coded bits are moved to data block P<sub>G</sub>:

```
C(2), C(3), C(5), C(6), C(8), C(9), C(11), C(12), C(14), C(15), C(17), C(18), C(20), C(21), C(23), C(24), C(26), C(27), C(29), C(30), C(32), C(33), C(35), C(36), C(38), C(39), C(41), C(42), C(44), C(45), C(47), C(48), C(50), C(51), C(53), C(54), C(56), C(57), C(59), C(60), C(62), C(63), C(65), C(66), C(68), C(69), C(71), C(72), C(74), C(75), C(77), C(78), C(80), C(81), C(83), C(84), C(86), C(87), C(89), C(90), C(93), C(95), C(96), C(98), C(99), C(101), C(102), C(104), C(105), C(107), C(108), C(110), C(111), C(113), C(114), C(116), C(117), C(119), C(120), C(122), C(123), C(125), C(126), C(128), C(129), C(131), C(132), C(134), C(135), C(137), C(138), C(140), C(141), C(143), C(144), C(146), C(147), C(149), C(150), C(152), C(153), C(155), C(156), C(158), C(159), C(161), C(162), C(164), C(165), C(166), C(188), C(189), C(171), C(173), C(174), C(176), C(177), C(179), C(180), C(182), C(183), C(185), C(186), C(188), C(189), C(189), C(180), C(182), C(183), C(185), C(186), C(188), C(189), C(189), C(180), C(181), C(181
```

```
C(191), C(192), C(194), C(195), C(197), C(198), C(200), C(201), C(203), C(204), C(206), C(207), C(209),
C(210), C(212), C(213), C(215), C(216), C(218), C(219), C(221), C(222), C(224), C(225), C(227), C(228),
C(230), C(231), C(233), C(234), C(236), C(237), C(239), C(240), C(242), C(243), C(245), C(246), C(248),
C(249), C(251), C(252), C(254), C(255), C(257), C(258), C(260), C(261), C(263), C(264), C(266), C(267),
C(269), C(270), C(272), C(273), C(275), C(276), C(278), C(279), C(281), C(282), C(284), C(285), C(287),
C(288), C(290), C(291), C(293), C(294), C(296), C(297), C(299), C(300), C(302), C(303), C(305), C(306),
C(308), C(309), C(311), C(312), C(314), C(315), C(317), C(318), C(320), C(321), C(323), C(324), C(326),
C(327), C(329), C(330), C(332), C(333), C(335), C(336), C(338), C(339), C(341), C(342), C(344), C(345),
C(347), C(348), C(350), C(351), C(353), C(354), C(356), C(357), C(359), C(360), C(362), C(363), C(365),
C(366), C(368), C(369), C(371), C(372), C(374), C(375), C(377), C(378), C(380), C(381), C(383), C(384),
C(386), C(387), C(389), C(390), C(392), C(393), C(395), C(396), C(398), C(399), C(401), C(402), C(404),
C(405), C(407), C(408), C(410), C(411), C(413), C(414), C(416), C(417), C(419), C(420), C(422), C(423),
C(425), C(426), C(428), C(429), C(431), C(432), C(434), C(435), C(437), C(438), C(440), C(441), C(443),
C(444), C(446), C(447), C(449), C(450), C(452), C(453), C(455), C(456), C(458), C(459), C(461), C(462),
C(464), C(465), C(467), C(468), C(470), C(471), C(473), C(474), C(476), C(477), C(479), C(480), C(482),
C(483), C(485), C(486), C(488), C(489), C(491), C(492), C(494), C(495), C(497), C(498), C(500), C(501),
C(503), C(504), C(506), C(507), C(509), C(510), C(512), C(513), C(515), C(518), C(519), C(521), C(522),
C(524), C(525), C(527), C(530), C(531), C(533), C(534), C(536), C(537), C(539), C(542), C(543), C(545),
C(546), C(548), C(549), C(551), C(554), C(555), C(557), C(558), C(560), C(561), C(563), C(566), C(567), C(567), C(568), C(56
C(569), C(570), C(572), C(573), C(575), C(578), C(579), C(581), C(582), C(584), C(585), C(587), C(590),
C(591), C(593), C(594), C(596), C(597), C(599), C(602), C(603), C(605), C(606), C(608), C(609), C(611),
C(614), C(615), C(617), C(618), C(620), C(621), C(623), C(626), C(627), C(629), C(630), C(632), C(633),
C(635), C(638), C(639), C(641), C(642), C(644), C(645), C(647), C(650), C(651), C(653), C(654), C(656),
C(657), C(659), C(662), C(663), C(665), C(666), C(668), C(669), C(671), C(674), C(675), C(677), C(678),
C(680), C(681), C(683), C(686), C(687), C(689), C(690), C(692), C(693), C(695), C(698), C(699), C(701),
C(702), C(704), C(705), C(707), C(710), C(711), C(713), C(714), C(716), C(717), C(719), C(722), C(723),
C(725), C(726), C(728), C(729), C(731), C(734), C(735), C(737), C(738), C(740), C(741), C(743), C(746),
C(747), C(749), C(750), C(752), C(753), C(755), C(758), C(759), C(761), C(762), C(764), C(765), C(767),
C(770), C(771), C(773), C(774), C(776), C(777), C(779), C(782), C(783), C(785), C(786), C(788), C(789),
C(791), C(794), C(795), C(797), C(798), C(800), C(801), C(803), C(806), C(807), C(809), C(810), C(812),
C(813), C(815), C(818), C(819), C(821), C(822), C(824), C(825), C(827), C(830), C(831), C(833), C(834),
C(836), C(837), C(839), C(842), C(843), C(845), C(846), C(848), C(849), C(851), C(854), C(855), C(857),
C(858), C(860), C(861), C(863), C(866), C(867), C(869), C(870), C(872), C(873), C(875), C(878), C(879),
C(881), C(882), C(884), C(885), C(887), C(890), C(891), C(893), C(894), C(896), C(897), C(899), C(902),
C(903), C(905), C(906), C(908), C(909), C(911), C(914), C(915), C(917), C(918), C(920), C(921), C(923),
C(926), C(927), C(929), C(930), C(932), C(933), C(935), C(938), C(939), C(941), C(942), C(944), C(945),
C(947), C(950), C(951), C(953), C(954), C(956), C(957), C(959), C(962), C(963), C(965), C(966), C(968),
C(969), C(971), C(974), C(975), C(977), C(978), C(980), C(981), C(983), C(986), C(987), C(989), C(990),
C(992), C(993), C(995), C(998), C(999), C(1001), C(1002), C(1004), C(1005), C(1007), C(1010), C(1011),
C(1013), C(1014), C(1016), C(1017), C(1019), C(1022), C(1023), C(1025), C(1026), C(1028), C(1029),
C(1031), C(1034), C(1035), C(1037), C(1038), C(1040), C(1041), C(1043), C(1046), C(1047), C(1049),
C(1050), C(1052), C(1053), C(1055), C(1058), C(1059), C(1061), C(1062), C(1064), C(1065), C(1067),
C(1070), C(1071), C(1073), C(1074), C(1076), C(1077), C(1079), C(1082), C(1083), C(1085), C(1086),
C(1088), C(1089), C(1091), C(1094), C(1095), C(1097), C(1098), C(1100), C(1101), C(1103), C(1106),
C(1107), C(1109), C(1110), C(1112), C(1113), C(1115), C(1118), C(1119), C(1121), C(1122), C(1124),
C(1125), C(1127), C(1130), C(1131), C(1133), C(1134), C(1136), C(1137), C(1139), C(1142), C(1143),
C(1145), C(1146), C(1148), C(1149), C(1151), C(1154), C(1155), C(1157), C(1158), C(1160), C(1161),
C(1163), C(1166), C(1167), C(1169), C(1170), C(1172), C(1173), C(1175), C(1178), C(1179), C(1181),
C(1182), C(1184), C(1185), C(1187), C(1190), C(1191), C(1193), C(1194), C(1196), C(1197), C(1199),
C(1202), C(1203), C(1205), C(1206), C(1208), C(1209), C(1211), C(1214), C(1215), C(1217), C(1218),
C(1220), C(1221), C(1223), C(1226), C(1227), C(1229), C(1230), C(1232), C(1233), C(1235), C(1238),
C(1239), C(1241), C(1242), C(1244), C(1245), C(1247), C(1250), C(1251), C(1253), C(1254), C(1256),
C(1257), C(1259), C(1262), C(1263), C(1265), C(1266), C(1268), C(1269), C(1271), C(1274), C(1275),
C(1277), C(1278), C(1280), C(1281), C(1283), C(1286), C(1287), C(1289), C(1290), C(1292), C(1293),
C(1295), C(1298), C(1299), C(1301), C(1302), C(1304), C(1305), C(1307), C(1310), C(1311), C(1313),
C(1314), C(1316), C(1317), C(1319), C(1322), C(1323), C(1325), C(1326), C(1328), C(1329), C(1331),
C(1334), C(1335), C(1337), C(1338), C(1340), C(1341), C(1343), C(1346), C(1347), C(1349), C(1350),
C(1352), C(1353), C(1355), C(1358), C(1359), C(1361), C(1362), C(1364), C(1365), C(1367), C(1370),
C(1371), C(1373), C(1374), C(1376), C(1377), C(1379), C(1382), C(1383), C(1385), C(1386), C(1388),
C(1389), C(1391), C(1394), C(1395), C(1397), C(1398), C(1400), C(1401), C(1403), C(1406), C(1407),
C(1409), C(1410), C(1412), C(1413), C(1415), C(1418), C(1419), C(1421), C(1422), C(1424), C(1425),
```

C(1427), C(1430), C(1431), C(1433), C(1434), C(1436), C(1437), C(1439), C(1442), C(1443), C(1445), C(1446), C(1448), C(1449), C(1451), C(1454), C(1455), C(1457), C(1458), C(1460), C(1463), C(1466)

And the following 448 coded bits are moved to data block P<sub>B</sub>:

```
C(1), C(4), C(7), C(10), C(13), C(16), C(19), C(22), C(25), C(28), C(31), C(34), C(37), C(40), C(43), C(46),
C(49), C(52), C(55), C(58), C(61), C(64), C(67), C(70), C(73), C(76), C(79), C(82), C(85), C(88), C(91),
C(94), C(97), C(100), C(103), C(106), C(109), C(112), C(115), C(118), C(121), C(124), C(127), C(130),
C(133), C(136), C(139), C(142), C(145), C(148), C(151), C(154), C(157), C(160), C(163), C(166), C(169),
C(172), C(175), C(178), C(181), C(184), C(187), C(190), C(193), C(196), C(199), C(202), C(205), C(208),
C(211), C(214), C(217), C(220), C(223), C(226), C(229), C(232), C(235), C(238), C(241), C(244), C(247),
C(250), C(253), C(256), C(259), C(262), C(265), C(268), C(271), C(274), C(277), C(280), C(283), C(286),
C(289), C(292), C(295), C(298), C(301), C(304), C(307), C(310), C(313), C(316), C(319), C(322), C(325),
C(328), C(331), C(334), C(337), C(340), C(343), C(346), C(349), C(352), C(355), C(358), C(361), C(364),
C(367), C(370), C(373), C(376), C(379), C(382), C(385), C(388), C(391), C(394), C(397), C(400), C(403),
C(406), C(409), C(412), C(415), C(418), C(421), C(424), C(427), C(430), C(433), C(436), C(439), C(442),
C(445), C(448), C(451), C(451), C(457), C(460), C(463), C(466), C(469), C(472), C(475), C(478), C(481),
C(484), C(487), C(490), C(493), C(496), C(499), C(502), C(505), C(508), C(511), C(514), C(516), C(517),
C(520), C(523), C(526), C(528), C(529), C(532), C(535), C(538), C(540), C(541), C(544), C(547), C(550), C(55
C(552), C(553), C(556), C(559), C(562), C(564), C(565), C(568), C(571), C(574), C(576), C(577), C(580),
C(583), C(586), C(588), C(589), C(592), C(595), C(598), C(600), C(601), C(604), C(607), C(610), C(612),
C(613), C(616), C(619), C(622), C(624), C(625), C(628), C(631), C(634), C(636), C(637), C(640), C(643),
C(646), C(648), C(649), C(652), C(655), C(658), C(660), C(661), C(664), C(667), C(670), C(672), C(673),
C(676), C(679), C(682), C(684), C(685), C(688), C(691), C(694), C(696), C(697), C(700), C(703), C(706),
C(708), C(709), C(712), C(715), C(718), C(720), C(721), C(724), C(727), C(730), C(732), C(733), C(736),
C(739), C(742), C(744), C(745), C(748), C(754), C(756), C(760), C(766), C(768), C(772), C(778), C(780),
C(784), C(790), C(792), C(796), C(802), C(804), C(808), C(814), C(816), C(820), C(826), C(828), C(832),
C(838), C(840), C(844), C(850), C(852), C(856), C(862), C(864), C(868), C(874), C(876), C(880), C(886),
C(888), C(892), C(898), C(900), C(904), C(910), C(912), C(916), C(922), C(924), C(928), C(934), C(936),
C(940), C(946), C(948), C(952), C(958), C(960), C(964), C(970), C(972), C(976), C(982), C(984), C(988),
C(994), C(996), C(1000), C(1006), C(1008), C(1012), C(1018), C(1020), C(1024), C(1030), C(1032),
C(1036), C(1042), C(1044), C(1048), C(1054), C(1056), C(1060), C(1066), C(1068), C(1072), C(1078),
C(1080), C(1084), C(1090), C(1092), C(1096), C(1102), C(1104), C(1108), C(1114), C(1116), C(1120),
C(1126), C(1128), C(1132), C(1138), C(1140), C(1144), C(1150), C(1152), C(1156), C(1162), C(1164),
C(1168), C(1174), C(1176), C(1180), C(1186), C(1188), C(1192), C(1198), C(1200), C(1204), C(1210),
C(1212), C(1216), C(1222), C(1224), C(1228), C(1234), C(1236), C(1240), C(1246), C(1248), C(1252),
C(1258), C(1260), C(1264), C(1270), C(1272), C(1276), C(1282), C(1284), C(1288), C(1294), C(1296),
C(1300), C(1306), C(1308), C(1312), C(1318), C(1320), C(1324), C(1330), C(1332), C(1336), C(1342),
C(1344), C(1348), C(1354), C(1356), C(1360), C(1366), C(1368), C(1372), C(1378), C(1380), C(1384),
C(1390), C(1392), C(1396), C(1402), C(1404), C(1408), C(1414), C(1416), C(1420), C(1426), C(1428),
C(1432), C(1438), C(1440), C(1444), C(1450), C(1452), C(1456), C(1462)
```

The vectors P<sub>G</sub> and P<sub>B</sub> of coded and punctured bits is combined with in band bits to vector P<sub>C</sub> as

$P_{C'}(k) = ic(k)$	for $k = 0, 1, 2, 3, 4, 5, 6, 7$
$P_{C}'(k+8) = P_{G}(k)$	for $k = 0, 1,, 447$
$P_{C}'(k+448) = ic(k)$	for k = 8, 9, 10, 11, 12, 13, 14, 15
$P_{C}'(k+16) = P_{G}(k)$	for k = 448, 449,, 895
$P_{C}'(k+896) = ic(k)$	for k = 16, 17, 18, 19, 20, 21, 22, 23
$P_{C}'(k+920) = P_{B}(k)$	for $k = 0, 1,, 447$

#### O-TCH/WFS15.85:

The block of 323 bits  $\{u(0)...u(322)\}$  is encoded with the 1/5 rate convolutional code defined by the following polynomials:

$$\begin{aligned} G4/G7 &= 1 + D^2 + D^3 + D^5 + D^6/1 + D + D^2 + D^3 + D^6 \\ G4/G7 &= 1 + D^2 + D^3 + D^5 + D^6/1 + D + D^2 + D^3 + D^6 \\ G5/G7 &= 1 + D + D^4 + D^6/1 + D + D^2 + D^3 + D^6 \\ G6/G7 &= 1 + D + D^2 + D^3 + D^4 + D^6/1 + D + D^2 + D^3 + D^6 \\ G7/G7 &= 1 \end{aligned}$$

resulting in 1645 coded bits,  $\{C(0)...C(1644)\}\$  defined by:

$$\begin{split} r(k) &= u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-6) \\ C(5k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(5k+1) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(5k+2) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(5k+3) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(5k+4) &= u(k) & \text{for } k = 0, 1, ..., 322; \qquad r(k) = 0 \text{ for } k < 0 \end{split}$$

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(5k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(5k+1) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(5k+2) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(5k+3) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(5k+4) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) & \text{for } k = 323, 324, ..., 238 \end{split}$$

The following 896 coded bits are moved to data block P<sub>G</sub>:

```
C(4), C(9), C(11), C(12), C(14), C(16), C(17), C(18), C(19), C(21), C(22), C(23), C(24), C(26), C(27),
C(28), C(29), C(31), C(32), C(34), C(36), C(37), C(38), C(39), C(41), C(42), C(43), C(44), C(46), C(47),
C(49), C(51), C(52), C(53), C(54), C(56), C(57), C(58), C(59), C(61), C(62), C(64), C(66), C(67), C(68),
C(69), C(71), C(72), C(73), C(74), C(76), C(77), C(79), C(81), C(82), C(83), C(84), C(86), C(87), C(88),
C(89), C(91), C(92), C(94), C(96), C(97), C(98), C(99), C(101), C(102), C(103), C(104), C(106), C(107),
C(109), C(111), C(112), C(113), C(114), C(116), C(117), C(118), C(119), C(121), C(122), C(124), C(126),
C(127), C(128), C(129), C(131), C(132), C(133), C(134), C(136), C(137), C(139), C(141), C(142), C(143),
C(144), C(146), C(147), C(148), C(149), C(151), C(152), C(154), C(156), C(157), C(158), C(159), C(161),
C(162), C(163), C(164), C(166), C(167), C(169), C(171), C(172), C(173), C(174), C(176), C(177), C(178),
C(179), C(181), C(182), C(184), C(186), C(187), C(188), C(189), C(191), C(192), C(193), C(194), C(196),
C(197), C(199), C(201), C(202), C(203), C(204), C(206), C(207), C(208), C(209), C(211), C(212), C(214),
C(216), C(217), C(218), C(219), C(221), C(222), C(223), C(224), C(226), C(227), C(229), C(231), C(232),
C(233), C(234), C(236), C(237), C(238), C(239), C(241), C(242), C(244), C(246), C(247), C(248), C(249),
C(251), C(252), C(253), C(254), C(256), C(257), C(259), C(261), C(262), C(263), C(264), C(266), C(267),
C(268), C(269), C(271), C(272), C(274), C(276), C(277), C(278), C(279), C(281), C(282), C(283), C(284),
C(286), C(287), C(289), C(291), C(292), C(293), C(294), C(296), C(297), C(298), C(299), C(301), C(302),
C(304), C(306), C(307), C(308), C(309), C(311), C(312), C(313), C(314), C(316), C(317), C(319), C(321),
C(322), C(323), C(324), C(326), C(327), C(328), C(329), C(331), C(332), C(334), C(336), C(337), C(338),
C(339), C(341), C(342), C(343), C(344), C(346), C(347), C(349), C(351), C(352), C(353), C(354), C(356),
C(357), C(358), C(359), C(361), C(362), C(364), C(366), C(367), C(369), C(371), C(372), C(374), C(376),
C(377), C(379), C(381), C(384), C(386), C(387), C(389), C(391), C(394), C(396), C(397), C(399), C(401),
C(404), C(406), C(407), C(409), C(411), C(414), C(416), C(417), C(419), C(421), C(424), C(426), C(427),
C(429), C(431), C(434), C(436), C(437), C(439), C(441), C(444), C(446), C(447), C(449), C(451), C(454),
C(456), C(457), C(459), C(461), C(464), C(466), C(467), C(469), C(471), C(474), C(476), C(477), C(479),
C(481), C(484), C(486), C(487), C(489), C(491), C(494), C(496), C(497), C(499), C(501), C(504), C(506),
C(507), C(509), C(511), C(514), C(516), C(517), C(519), C(521), C(524), C(526), C(527), C(529), C(531),
```

```
C(534), C(536), C(537), C(539), C(541), C(544), C(546), C(547), C(549), C(551), C(554), C(556), C(557),
C(559), C(561), C(564), C(566), C(567), C(569), C(571), C(574), C(576), C(577), C(579), C(581), C(584),
C(586), C(587), C(589), C(591), C(594), C(596), C(597), C(599), C(601), C(604), C(606), C(607), C(609),
C(611), C(614), C(616), C(617), C(619), C(621), C(624), C(626), C(627), C(629), C(631), C(634), C(636),
C(637), C(639), C(641), C(644), C(646), C(647), C(649), C(651), C(654), C(656), C(657), C(659), C(661),
C(664), C(666), C(667), C(669), C(671), C(674), C(676), C(677), C(679), C(681), C(684), C(686), C(687),
C(689), C(691), C(694), C(696), C(697), C(699), C(701), C(704), C(706), C(707), C(709), C(711), C(714),
C(716), C(717), C(719), C(721), C(724), C(726), C(727), C(729), C(731), C(734), C(736), C(737), C(739),
C(741), C(744), C(746), C(747), C(749), C(751), C(754), C(756), C(757), C(759), C(761), C(764), C(766),
C(767), C(769), C(771), C(774), C(776), C(777), C(779), C(781), C(784), C(786), C(787), C(789), C(791),
C(794), C(796), C(797), C(799), C(801), C(804), C(806), C(807), C(809), C(811), C(814), C(816), C(817),
C(819), C(821), C(824), C(826), C(827), C(829), C(831), C(834), C(836), C(837), C(839), C(841), C(844),
C(846), C(847), C(849), C(851), C(854), C(856), C(857), C(859), C(861), C(864), C(866), C(867), C(869),
C(871), C(874), C(876), C(877), C(879), C(881), C(884), C(886), C(887), C(889), C(891), C(894), C(896),
C(897), C(899), C(901), C(904), C(906), C(907), C(909), C(911), C(914), C(916), C(917), C(919), C(921),
C(924), C(926), C(927), C(929), C(931), C(934), C(936), C(937), C(939), C(941), C(944), C(946), C(947),
C(949), C(951), C(954), C(956), C(957), C(959), C(961), C(964), C(966), C(967), C(969), C(971), C(974),
C(976), C(977), C(979), C(981), C(984), C(986), C(987), C(989), C(991), C(994), C(996), C(997), C(999),
C(1001), C(1004), C(1006), C(1007), C(1009), C(1011), C(1014), C(1016), C(1017), C(1019), C(1021),
C(1024), C(1026), C(1027), C(1029), C(1031), C(1034), C(1036), C(1037), C(1039), C(1041), C(1044),
C(1046), C(1047), C(1049), C(1051), C(1054), C(1056), C(1057), C(1059), C(1061), C(1064), C(1066),
C(1067), C(1069), C(1071), C(1074), C(1076), C(1077), C(1079), C(1081), C(1084), C(1086), C(1087),
C(1089), C(1091), C(1094), C(1096), C(1097), C(1099), C(1101), C(1104), C(1106), C(1107), C(1109),
C(1111), C(1114), C(1116), C(1117), C(1119), C(1121), C(1124), C(1126), C(1127), C(1129), C(1131),
C(1134), C(1136), C(1137), C(1139), C(1141), C(1144), C(1146), C(1147), C(1149), C(1151), C(1154),
C(1156), C(1157), C(1159), C(1161), C(1164), C(1166), C(1167), C(1169), C(1171), C(1174), C(1176),
C(1177), C(1179), C(1181), C(1184), C(1186), C(1187), C(1189), C(1191), C(1194), C(1196), C(1197),
C(1199), C(1201), C(1204), C(1206), C(1207), C(1209), C(1211), C(1214), C(1216), C(1217), C(1219),
C(1221), C(1224), C(1226), C(1227), C(1229), C(1231), C(1234), C(1236), C(1237), C(1239), C(1241),
C(1244), C(1246), C(1247), C(1249), C(1251), C(1254), C(1256), C(1257), C(1259), C(1261), C(1264),
C(1266), C(1267), C(1269), C(1271), C(1274), C(1276), C(1277), C(1279), C(1281), C(1284), C(1286),
C(1287), C(1289), C(1291), C(1294), C(1296), C(1297), C(1299), C(1301), C(1304), C(1306), C(1307),
C(1309), C(1311), C(1314), C(1316), C(1317), C(1319), C(1321), C(1324), C(1326), C(1327), C(1329),
C(1331), C(1334), C(1336), C(1337), C(1339), C(1341), C(1344), C(1346), C(1347), C(1349), C(1351),
C(1354), C(1356), C(1357), C(1359), C(1361), C(1364), C(1366), C(1367), C(1369), C(1371), C(1374),
C(1376), C(1377), C(1379), C(1381), C(1384), C(1386), C(1387), C(1389), C(1391), C(1394), C(1396),
C(1397), C(1399), C(1401), C(1404), C(1406), C(1407), C(1409), C(1411), C(1414), C(1416), C(1417),
C(1419), C(1421), C(1424), C(1426), C(1427), C(1429), C(1431), C(1434), C(1436), C(1437), C(1439),
C(1441), C(1444), C(1446), C(1447), C(1449), C(1451), C(1454), C(1456), C(1457), C(1459), C(1461),
C(1464), C(1466), C(1467), C(1469), C(1471), C(1474), C(1476), C(1477), C(1479), C(1481), C(1484),
C(1486), C(1487), C(1489), C(1491), C(1494), C(1496), C(1497), C(1499), C(1501), C(1504), C(1506),
C(1507), C(1509), C(1511), C(1514), C(1516), C(1517), C(1519), C(1521), C(1524), C(1526), C(1527),
C(1529), C(1531), C(1534), C(1536), C(1537), C(1539), C(1541), C(1544), C(1546), C(1547), C(1549),
C(1551), C(1554), C(1556), C(1557), C(1559), C(1561), C(1564), C(1566), C(1567), C(1569), C(1571),
C(1574), C(1576), C(1577), C(1579), C(1581), C(1584), C(1586), C(1587), C(1589), C(1591), C(1594),
C(1596), C(1597), C(1599), C(1601), C(1604), C(1606), C(1609), C(1614), C(1616), C(1619), C(1624),
C(1626), C(1629), C(1631), C(1634), C(1636), C(1639), C(1644)
```

#### And the following 448 coded bits are moved to data block P<sub>B</sub>:

```
C(30), C(33), C(35), C(40), C(45), C(48), C(50), C(55), C(60), C(63), C(65), C(70), C(75), C(78), C(80), C(85), C(90), C(93), C(95), C(100), C(105), C(108), C(110), C(115), C(120), C(123), C(125), C(130), C(135), C(138), C(140), C(145), C(150), C(153), C(155), C(160), C(165), C(168), C(170), C(175), C(180), C(183), C(185), C(190), C(195), C(198), C(200), C(205), C(210), C(213), C(215), C(220), C(225), C(228), C(230), C(235), C(240), C(243), C(245), C(250), C(255), C(258), C(260), C(265), C(270), C(273), C(275), C(280), C(285), C(288), C(290), C(295), C(300), C(303), C(305), C(310), C(315), C(318), C(320), C(325), C(330), C(333), C(335), C(340), C(345), C(348), C(350), C(355), C(363), C(368), C(373), C(378), C(382), C(383), C(388), C(392), C(393), C(398), C(402), C(403), C(408), C(412), C(413), C(418), C(422), C(423), C(428), C(432), C(433), C(438), C(442), C(443), C(448), C(452), C(453), C(458), C(462), C(463), C(468), C(472), C(473), C(478), C(482), C(483), C(488), C(492), C(493), C(498), C(502), C(503), C(508), C(512), C(513), C(518), C(522), C(523), C(528), C(532), C(533), C(538), C(542), C(543), C(548), C(552), C(553), C(555), C(553), C(553), C(553), C(553), C(554), C(552), C(553), C(555), C(553), C(553), C(553), C(553), C(554), C(552), C(553), C(555), C(553), C(553), C(553), C(554), C(552), C(553), C(555), C(553), C(553), C(553), C(553), C(554), C(552), C(553), C(555), C(553), C(553), C(553), C(554), C(554), C(552), C(553), C(555), C(553), C(553), C(553), C(554), C(554), C(555), C
```

```
C(558), C(562), C(563), C(568), C(572), C(573), C(578), C(582), C(583), C(588), C(592), C(593), C(598),
C(602), C(608), C(612), C(613), C(618), C(622), C(623), C(628), C(632), C(633), C(638), C(642), C(643),
C(648), C(652), C(653), C(658), C(662), C(668), C(672), C(673), C(678), C(682), C(683), C(688), C(692),
C(698), C(702), C(703), C(708), C(712), C(713), C(718), C(722), C(723), C(728), C(732), C(733), C(738),
C(742), C(743), C(748), C(752), C(758), C(762), C(763), C(768), C(772), C(773), C(778), C(782), C(788),
C(792), C(793), C(798), C(802), C(803), C(808), C(812), C(813), C(818), C(822), C(823), C(828), C(832),
C(833), C(838), C(842), C(848), C(852), C(853), C(858), C(862), C(863), C(868), C(872), C(878), C(882),
C(883), C(888), C(892), C(893), C(898), C(902), C(903), C(908), C(912), C(913), C(918), C(922), C(923),
C(928), C(932), C(938), C(942), C(943), C(948), C(952), C(953), C(958), C(962), C(968), C(972), C(973),
C(978), C(982), C(983), C(988), C(992), C(993), C(998), C(1002), C(1003), C(1008), C(1012), C(1013),
C(1018), C(1022), C(1028), C(1032), C(1033), C(1038), C(1042), C(1043), C(1048), C(1052), C(1058),
C(1062), C(1063), C(1068), C(1072), C(1073), C(1078), C(1082), C(1083), C(1088), C(1092), C(1093),
C(1098), C(1102), C(1103), C(1108), C(1112), C(1118), C(1122), C(1123), C(1128), C(1132), C(1133),
C(1138), C(1142), C(1148), C(1152), C(1153), C(1158), C(1162), C(1163), C(1168), C(1168), C(1172), C(1173),
C(1178), C(1182), C(1183), C(1188), C(1192), C(1193), C(1198), C(1202), C(1208), C(1212), C(1213),
C(1218), C(1222), C(1223), C(1228), C(1232), C(1238), C(1242), C(1243), C(1248), C(1252), C(1253),
C(1258), C(1262), C(1263), C(1268), C(1272), C(1273), C(1278), C(1282), C(1283), C(1288), C(1292),
C(1298), C(1302), C(1303), C(1308), C(1312), C(1313), C(1318), C(1322), C(1328), C(1332), C(1333),
C(1338), C(1342), C(1343), C(1348), C(1352), C(1353), C(1358), C(1362), C(1363), C(1368), C(1372),
C(1373), C(1378), C(1382), C(1388), C(1392), C(1393), C(1398), C(1402), C(1403), C(1408), C(1412),
C(1418), C(1422), C(1423), C(1428), C(1432), C(1433), C(1438), C(1442), C(1443), C(1448), C(1452),
C(1453), C(1458), C(1462), C(1463), C(1468), C(1472), C(1478), C(1482), C(1483), C(1488), C(1492),
C(1493), C(1498), C(1502), C(1508), C(1512), C(1513), C(1518), C(1522), C(1523), C(1528), C(1532),
C(1533), C(1538), C(1542), C(1543), C(1548), C(1552), C(1553), C(1558), C(1562), C(1568), C(1572),
C(1573), C(1578), C(1582), C(1583), C(1588), C(1592), C(1598), C(1602), C(1603), C(1607), C(1611),
C(1612), C(1617), C(1621), C(1622), C(1623), C(1627), C(1632), C(1637), C(1641)
```

The vectors P<sub>G</sub> and P<sub>B</sub> of coded and punctured bits is combined with in band bits to vector P<sub>C</sub>' as

$P_{C}'(k) = ic(k)$	for $k = 0, 1, 2, 3, 4, 5, 6, 7$
$P_{C}'(k+8) = P_{G}(k)$	for $k = 0, 1,, 447$
$P_{C}'(k+448) = ic(k)$	for k = 8, 9, 10, 11, 12, 13, 14, 15
$P_{C}'(k+16) = P_{G}(k)$	for k = 448, 449,, 895
$P_{C}'(k+896) = ic(k)$	for k = 16, 17, 18, 19, 20, 21, 22, 23
$P_{C}'(k+920) = P_{B}(k)$	for $k = 0, 1,, 447$

#### **O-TCH/WFS12.65:**

The block of 259 bits  $\{u(0)...u(258)\}$  is encoded with the 1/6 rate convolutional code defined by the following polynomials:

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G5/G7 = 1 + D + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G6/G7 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G6/G7 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G7/G7 = 1$$

resulting in 1590 coded bits,  $\{C(0)...C(1589)\}\$  defined by:

$$\begin{split} r(k) &= u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-6) \\ C(6k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(6k+1) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \end{split}$$

$$\begin{split} &C(6k+2) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ &C(6k+3) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ &C(6k+4) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ &C(6k+5) &= u(k) \end{split} \qquad \qquad \text{for } k = 0, 1, ..., 258; \qquad r(k) = 0 \text{ for } k < 0.$$

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(6k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(6k+1) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(6k+2) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(6k+3) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(6k+4) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(6k+5) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) & \text{for } k = 259, 260, ..., 264 \end{split}$$

The following 896 coded bits are moved to data block P<sub>G</sub>:

```
C(5), C(7), C(11), C(13), C(14), C(17), C(19), C(20), C(23), C(25), C(26), C(27), C(29), C(31), C(32),
C(35), C(36), C(37), C(38), C(39), C(41), C(42), C(43), C(44), C(45), C(47), C(48), C(49), C(50), C(51),
C(53), C(54), C(55), C(56), C(59), C(60), C(61), C(62), C(63), C(65), C(66), C(67), C(68), C(71), C(72),
C(73), C(74), C(75), C(77), C(78), C(79), C(80), C(81), C(83), C(84), C(85), C(86), C(87), C(89), C(90),
C(91), C(92), C(95), C(96), C(97), C(98), C(99), C(101), C(102), C(103), C(104), C(107), C(108), C(109),
C(110), C(111), C(113), C(114), C(115), C(116), C(117), C(119), C(120), C(121), C(122), C(123), C(125),
C(126), C(127), C(128), C(131), C(132), C(133), C(134), C(135), C(137), C(138), C(139), C(140), C(143),
C(144), C(145), C(146), C(147), C(149), C(150), C(151), C(152), C(153), C(155), C(156), C(157), C(158),
C(159), C(161), C(162), C(163), C(164), C(167), C(168), C(169), C(170), C(171), C(173), C(174), C(175),
C(176), C(179), C(180), C(181), C(182), C(183), C(185), C(186), C(187), C(188), C(189), C(191), C(192),
C(193), C(194), C(195), C(197), C(198), C(199), C(200), C(203), C(204), C(205), C(206), C(207), C(209),
C(210), C(211), C(212), C(215), C(216), C(217), C(218), C(219), C(221), C(222), C(223), C(224), C(225),
C(227), C(228), C(229), C(230), C(231), C(233), C(234), C(235), C(236), C(239), C(240), C(241), C(242),
C(243), C(245), C(246), C(247), C(248), C(251), C(252), C(253), C(254), C(255), C(257), C(258), C(259),
C(260), C(261), C(263), C(264), C(265), C(266), C(267), C(269), C(270), C(271), C(272), C(275), C(276),
C(277), C(278), C(279), C(281), C(282), C(283), C(284), C(287), C(288), C(289), C(290), C(291), C(293),
C(294), C(295), C(296), C(297), C(299), C(300), C(301), C(302), C(303), C(305), C(306), C(307), C(308),
C(311), C(312), C(313), C(314), C(315), C(317), C(318), C(319), C(320), C(323), C(324), C(325), C(326),
C(327), C(329), C(330), C(331), C(332), C(333), C(335), C(336), C(337), C(338), C(339), C(341), C(342),
C(343), C(344), C(347), C(348), C(349), C(350), C(351), C(353), C(354), C(355), C(356), C(359), C(360),
C(361), C(362), C(363), C(365), C(366), C(367), C(368), C(369), C(371), C(372), C(373), C(374), C(375),
C(377), C(378), C(379), C(380), C(383), C(384), C(385), C(386), C(387), C(389), C(390), C(391), C(392),
C(395), C(396), C(397), C(398), C(399), C(401), C(402), C(403), C(404), C(405), C(407), C(408), C(409), C(40
C(410), C(411), C(413), C(414), C(415), C(416), C(419), C(420), C(421), C(422), C(423), C(425), C(426),
C(427), C(428), C(431), C(432), C(433), C(434), C(435), C(437), C(438), C(439), C(440), C(441), C(443),
C(444), C(445), C(446), C(447), C(449), C(450), C(451), C(452), C(455), C(456), C(457), C(458), C(459),
C(461), C(462), C(463), C(464), C(467), C(468), C(469), C(470), C(471), C(473), C(474), C(475), C(476),
C(477), C(479), C(480), C(481), C(482), C(483), C(485), C(486), C(487), C(488), C(491), C(492), C(493),
C(494), C(495), C(497), C(498), C(499), C(500), C(503), C(505), C(506), C(509), C(511), C(512), C(515),
C(517), C(518), C(521), C(523), C(524), C(527), C(529), C(530), C(533), C(535), C(536), C(539), C(541),
C(542), C(545), C(547), C(548), C(551), C(553), C(554), C(557), C(559), C(560), C(563), C(565), C(566),
C(569), C(571), C(572), C(575), C(577), C(578), C(581), C(583), C(584), C(587), C(589), C(590), C(593),
C(595), C(596), C(599), C(601), C(602), C(605), C(607), C(608), C(611), C(613), C(614), C(617), C(619),
C(620), C(623), C(625), C(626), C(629), C(631), C(632), C(635), C(637), C(638), C(641), C(643), C(644),
C(647), C(649), C(650), C(653), C(655), C(656), C(659), C(661), C(662), C(665), C(667), C(668), C(671),
C(673), C(674), C(677), C(679), C(680), C(683), C(685), C(686), C(689), C(691), C(692), C(695), C(697),
C(698), C(701), C(703), C(704), C(707), C(709), C(710), C(713), C(715), C(716), C(719), C(721), C(722),
C(725), C(727), C(728), C(731), C(733), C(734), C(737), C(739), C(740), C(743), C(745), C(746), C(749),
```

```
C(751), C(752), C(755), C(757), C(758), C(761), C(763), C(764), C(767), C(769), C(770), C(773), C(775),
C(776), C(779), C(781), C(782), C(785), C(787), C(788), C(791), C(793), C(794), C(797), C(799), C(800),
C(803), C(805), C(806), C(809), C(811), C(812), C(815), C(817), C(818), C(821), C(823), C(824), C(827),
C(829), C(830), C(833), C(835), C(836), C(839), C(841), C(842), C(845), C(847), C(848), C(851), C(853),
C(854), C(857), C(859), C(860), C(863), C(865), C(866), C(869), C(871), C(872), C(875), C(877), C(878),
C(881), C(883), C(884), C(887), C(889), C(890), C(893), C(895), C(896), C(899), C(901), C(902), C(905),
C(907), C(908), C(911), C(913), C(914), C(917), C(919), C(920), C(923), C(925), C(926), C(929), C(931),
C(932), C(935), C(937), C(938), C(941), C(943), C(944), C(947), C(949), C(950), C(953), C(955), C(956),
C(959), C(961), C(962), C(965), C(967), C(968), C(971), C(973), C(974), C(977), C(979), C(980), C(983),
C(985), C(986), C(989), C(991), C(992), C(995), C(997), C(998), C(1001), C(1003), C(1004), C(1007),
C(1009), C(1013), C(1015), C(1016), C(1019), C(1021), C(1022), C(1025), C(1027), C(1028), C(1031),
C(1033), C(1034), C(1037), C(1039), C(1040), C(1043), C(1045), C(1049), C(1051), C(1052), C(1055),
C(1057), C(1058), C(1061), C(1063), C(1064), C(1067), C(1069), C(1070), C(1073), C(1075), C(1076),
C(1079), C(1081), C(1085), C(1087), C(1088), C(1091), C(1093), C(1094), C(1097), C(1099), C(1100),
C(1103), C(1105), C(1106), C(1109), C(1111), C(1112), C(1115), C(1117), C(1121), C(1123), C(1124),
C(1127), C(1129), C(1130), C(1133), C(1135), C(1136), C(1139), C(1141), C(1142), C(1145), C(1147),
C(1148), C(1151), C(1153), C(1157), C(1159), C(1160), C(1163), C(1165), C(1166), C(1169), C(1171),
C(1172), C(1175), C(1177), C(1178), C(1181), C(1183), C(1184), C(1187), C(1189), C(1193), C(1195),
C(1196), C(1199), C(1201), C(1202), C(1205), C(1207), C(1208), C(1211), C(1213), C(1214), C(1217),
C(1219), C(1220), C(1223), C(1225), C(1229), C(1231), C(1232), C(1235), C(1237), C(1238), C(1241),
C(1243), C(1244), C(1247), C(1249), C(1250), C(1253), C(1255), C(1256), C(1259), C(1261), C(1265),
C(1267), C(1268), C(1271), C(1273), C(1274), C(1277), C(1279), C(1280), C(1283), C(1285), C(1286),
C(1289), C(1291), C(1292), C(1295), C(1297), C(1301), C(1303), C(1304), C(1307), C(1309), C(1310),
C(1313), C(1315), C(1316), C(1319), C(1321), C(1322), C(1325), C(1327), C(1328), C(1331), C(1333),
C(1337), C(1339), C(1340), C(1343), C(1345), C(1346), C(1349), C(1351), C(1352), C(1355), C(1357),
C(1358), C(1361), C(1363), C(1364), C(1367), C(1369), C(1373), C(1375), C(1376), C(1379), C(1381),
C(1382), C(1385), C(1387), C(1388), C(1391), C(1393), C(1394), C(1397), C(1399), C(1400), C(1403),
C(1405), C(1409), C(1411), C(1412), C(1415), C(1417), C(1418), C(1421), C(1423), C(1424), C(1427),
C(1429), C(1430), C(1433), C(1435), C(1436), C(1439), C(1441), C(1445), C(1447), C(1448), C(1451),
C(1453), C(1454), C(1457), C(1459), C(1460), C(1463), C(1465), C(1466), C(1469), C(1471), C(1472),
C(1475), C(1477), C(1481), C(1483), C(1484), C(1487), C(1489), C(1490), C(1493), C(1495), C(1496),
C(1499), C(1501), C(1502), C(1505), C(1507), C(1508), C(1511), C(1513), C(1517), C(1519), C(1520),
C(1523), C(1525), C(1529), C(1531), C(1532), C(1535), C(1537), C(1541), C(1543), C(1544), C(1547),
C(1549), C(1553), C(1555), C(1559), C(1565), C(1567), C(1571), C(1573), C(1577), C(1583), C(1589)
```

And the following 448 coded bits are moved to data block P<sub>B</sub>:

```
C(21), C(33), C(34), C(46), C(57), C(58), C(69), C(70), C(82), C(93), C(94), C(105), C(106), C(118),
C(129), C(130), C(141), C(142), C(154), C(165), C(166), C(177), C(178), C(190), C(201), C(202), C(213),
C(214), C(226), C(237), C(238), C(249), C(250), C(262), C(273), C(274), C(285), C(286), C(298), C(309),
C(310), C(321), C(322), C(334), C(345), C(346), C(357), C(358), C(370), C(381), C(382), C(393), C(394),
C(406), C(417), C(418), C(429), C(430), C(442), C(453), C(454), C(465), C(466), C(478), C(489), C(490),
C(501), C(502), C(504), C(507), C(510), C(513), C(516), C(519), C(522), C(525), C(528), C(531), C(534),
C(537), C(540), C(543), C(546), C(549), C(552), C(555), C(558), C(561), C(564), C(567), C(570), C(573),
C(576), C(579), C(582), C(585), C(588), C(591), C(594), C(597), C(600), C(603), C(606), C(609), C(612),
C(615), C(618), C(621), C(624), C(627), C(630), C(633), C(636), C(639), C(642), C(645), C(648), C(651),
C(654), C(657), C(660), C(663), C(666), C(669), C(672), C(675), C(678), C(681), C(684), C(687), C(690),
C(693), C(696), C(699), C(702), C(705), C(708), C(711), C(714), C(717), C(720), C(723), C(726), C(729),
C(732), C(735), C(738), C(741), C(744), C(747), C(750), C(753), C(756), C(759), C(762), C(765), C(768),
C(771), C(774), C(777), C(780), C(783), C(786), C(789), C(792), C(795), C(798), C(801), C(804), C(807),
C(810), C(813), C(816), C(819), C(822), C(825), C(828), C(831), C(834), C(837), C(840), C(843), C(846),
C(849), C(852), C(855), C(858), C(861), C(864), C(867), C(870), C(873), C(876), C(879), C(882), C(885),
C(888), C(891), C(894), C(897), C(900), C(903), C(906), C(909), C(912), C(915), C(918), C(921), C(924),
C(927), C(930), C(933), C(936), C(939), C(942), C(945), C(948), C(951), C(954), C(957), C(960), C(963),
C(966), C(969), C(972), C(975), C(978), C(981), C(984), C(987), C(990), C(993), C(996), C(999), C(1002),
C(1005), C(1008), C(1010), C(1011), C(1014), C(1017), C(1020), C(1023), C(1026), C(1029), C(1032),
C(1035), C(1038), C(1041), C(1044), C(1046), C(1047), C(1050), C(1053), C(1056), C(1059), C(1062),
C(1065), C(1068), C(1071), C(1074), C(1077), C(1080), C(1082), C(1083), C(1086), C(1089), C(1092),
C(1095), C(1098), C(1101), C(1104), C(1107), C(1110), C(1113), C(1116), C(1118), C(1119), C(1122),
C(1125), C(1128), C(1131), C(1134), C(1137), C(1140), C(1143), C(1146), C(1149), C(1152), C(1154),
C(1155), C(1158), C(1161), C(1164), C(1167), C(1170), C(1173), C(1176), C(1179), C(1182), C(1185),
```

```
C(1188), C(1190), C(1191), C(1194), C(1197), C(1200), C(1203), C(1206), C(1209), C(1212), C(1215), C(1218), C(1221), C(1224), C(1226), C(1227), C(1230), C(1233), C(1236), C(1239), C(1242), C(1245), C(1248), C(1251), C(1254), C(1257), C(1260), C(1262), C(1263), C(1266), C(1269), C(1272), C(1275), C(1278), C(1281), C(1284), C(1287), C(1290), C(1293), C(1296), C(1298), C(1299), C(1302), C(1305), C(1308), C(1311), C(1314), C(1317), C(1320), C(1323), C(1326), C(1329), C(1332), C(1334), C(1335), C(1338), C(1341), C(1344), C(1347), C(1350), C(1353), C(1356), C(1359), C(1362), C(1365), C(1368), C(1370), C(1371), C(1374), C(1377), C(1380), C(1383), C(1386), C(1389), C(1392), C(1392), C(1398), C(1401), C(1404), C(1406), C(1407), C(1410), C(1413), C(1416), C(1419), C(1422), C(1425), C(1428), C(1431), C(1434), C(1437), C(1440), C(1442), C(1443), C(1446), C(1449), C(1452), C(1455), C(1458), C(1461), C(1464), C(1467), C(1470), C(1473), C(1476), C(1478), C(1479), C(1482), C(1485), C(1488), C(1491), C(1494), C(1497), C(1500), C(1503), C(1506), C(1509), C(1512), C(1514), C(1515), C(1518), C(1524), C(1526), C(1554), C(1556), C(1556), C(1560), C(1561), C(1562), C(1563), C(1566), C(1568), C(1572), C(1574), C(1579), C(1585)
```

The vectors P<sub>G</sub> and P<sub>B</sub> of coded and punctured bits is combined with in band bits to vector P<sub>C</sub> as

$P_{C}'(k) = ic(k)$	for $k = 0, 1, 2, 3, 4, 5, 6, 7$
$P_{C}'(k+8) = P_{G}(k)$	for $k = 0, 1,, 447$
$P_{C}'(k+448) = ic(k)$	for k = 8, 9, 10, 11, 12, 13, 14, 15
$P_{C}'(k+16) = P_{G}(k)$	for k = 448, 449,, 895
$P_{C}'(k+896) = ic(k)$	for k = 16, 17, 18, 19, 20, 21, 22, 23
$P_{C}'(k+920) = P_{B}(k)$	for k = 0, 1,, 447

#### **O-TCH/WFS8.85:**

The block of 183 bits  $\{u(0)...u(182)\}$  is encoded with the 1/8 rate convolutional code defined by the following polynomials:

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G5/G7 = 1 + D + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G5/G7 = 1 + D + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G6/G7 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G6/G7 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G7/G7 = 1$$

$$G7/G7 = 1$$

resulting in 1512 coded bits,  $\{C(0)...C(1511)\}\$  defined by:

$$\begin{split} r(k) &= u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-6) \\ C(8k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(8k+1) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(8k+2) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(8k+3) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(8k+4) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(8k+5) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ \end{split}$$

$$C(8k+6) = u(k)$$
  
 $C(8k+7) = u(k)$  for  $k = 0, 1, ..., 182$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(8k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(8k+1) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(8k+2) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(8k+3) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(8k+4) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(8k+5) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(8k+6) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) \\ C(8k+7) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) & \text{for } k = 183, 184, ..., 188 \\ \end{split}$$

The following 896 coded bits are moved to data block P<sub>G</sub>:

```
C(1), C(2), C(6), C(9), C(10), C(14), C(16), C(17), C(18), C(20), C(22), C(24), C(25), C(26), C(29), C(30),
C(32), C(33), C(34), C(36), C(38), C(40), C(41), C(42), C(43), C(44), C(45), C(46), C(48), C(49), C(50),
C(52), C(54), C(56), C(57), C(58), C(60), C(61), C(62), C(64), C(65), C(66), C(67), C(68), C(70), C(72),
C(73), C(74), C(76), C(77), C(78), C(80), C(81), C(82), C(83), C(84), C(86), C(88), C(89), C(90), C(92),
C(93), C(94), C(96), C(97), C(98), C(100), C(102), C(104), C(105), C(106), C(108), C(109), C(110), C(112),
C(113), C(114), C(115), C(116), C(118), C(120), C(121), C(122), C(124), C(125), C(126), C(128), C(129),
C(130), C(131), C(132), C(134), C(136), C(137), C(138), C(140), C(141), C(142), C(144), C(145), C(146),
C(148), C(150), C(152), C(153), C(154), C(156), C(157), C(158), C(160), C(161), C(162), C(163), C(164),
C(166), C(168), C(169), C(170), C(172), C(173), C(174), C(176), C(177), C(178), C(179), C(180), C(182),
C(184), C(185), C(186), C(188), C(189), C(190), C(192), C(193), C(194), C(196), C(198), C(200), C(201),
C(202), C(204), C(205), C(206), C(208), C(209), C(210), C(211), C(212), C(214), C(216), C(217), C(218),
C(220), C(221), C(222), C(224), C(225), C(226), C(227), C(228), C(230), C(232), C(233), C(234), C(236),
C(237), C(238), C(240), C(241), C(242), C(244), C(246), C(248), C(249), C(250), C(252), C(253), C(254),
C(256), C(257), C(258), C(259), C(260), C(262), C(264), C(265), C(266), C(268), C(269), C(270), C(272),
C(273), C(274), C(275), C(276), C(278), C(280), C(281), C(282), C(284), C(285), C(286), C(288), C(289),
C(290), C(292), C(294), C(296), C(297), C(298), C(300), C(301), C(302), C(304), C(305), C(306), C(307),
C(308), C(310), C(312), C(313), C(314), C(316), C(317), C(318), C(320), C(321), C(322), C(323), C(324),
C(326), C(328), C(329), C(330), C(332), C(333), C(334), C(336), C(337), C(338), C(340), C(342), C(344),
C(345), C(346), C(348), C(349), C(350), C(352), C(353), C(354), C(355), C(356), C(358), C(360), C(361),
C(362), C(364), C(365), C(366), C(368), C(369), C(370), C(371), C(372), C(374), C(376), C(377), C(378),
C(380), C(381), C(382), C(384), C(385), C(386), C(388), C(390), C(392), C(393), C(394), C(396), C(397),
C(398), C(400), C(401), C(402), C(403), C(404), C(406), C(408), C(409), C(410), C(412), C(413), C(414),
C(416), C(417), C(418), C(419), C(420), C(422), C(424), C(425), C(426), C(428), C(429), C(430), C(432),
C(433), C(434), C(436), C(438), C(440), C(441), C(442), C(444), C(445), C(446), C(448), C(449), C(450),
C(451), C(452), C(454), C(456), C(457), C(458), C(460), C(461), C(462), C(464), C(465), C(466), C(467),
C(468), C(470), C(472), C(473), C(474), C(476), C(477), C(478), C(480), C(481), C(482), C(484), C(486),
C(488), C(489), C(490), C(492), C(493), C(494), C(496), C(497), C(498), C(499), C(500), C(502), C(504),
C(505), C(506), C(508), C(509), C(510), C(512), C(513), C(514), C(515), C(516), C(518), C(520), C(521),
C(522), C(524), C(525), C(526), C(528), C(529), C(530), C(532), C(534), C(536), C(537), C(538), C(540),
C(541), C(542), C(544), C(545), C(546), C(547), C(548), C(550), C(552), C(553), C(554), C(556), C(557),
C(558), C(560), C(561), C(562), C(563), C(564), C(566), C(568), C(569), C(570), C(572), C(573), C(574),
C(576), C(577), C(578), C(580), C(582), C(584), C(585), C(586), C(588), C(589), C(590), C(592), C(593),
C(594), C(595), C(596), C(598), C(600), C(601), C(602), C(604), C(605), C(606), C(608), C(609), C(610),
C(611), C(612), C(614), C(616), C(617), C(618), C(620), C(621), C(622), C(624), C(625), C(626), C(628),
C(630), C(632), C(633), C(634), C(636), C(637), C(638), C(640), C(641), C(642), C(643), C(643), C(646), C(666), C(66
C(648), C(649), C(650), C(652), C(653), C(654), C(656), C(657), C(658), C(659), C(660), C(662), C(664),
C(665), C(666), C(668), C(669), C(670), C(673), C(674), C(676), C(678), C(681), C(682), C(684), C(686),
C(689), C(690), C(692), C(694), C(697), C(698), C(700), C(702), C(705), C(706), C(708), C(710), C(713),
```

```
C(714), C(716), C(718), C(721), C(722), C(724), C(726), C(729), C(730), C(732), C(734), C(737), C(738),
C(740), C(742), C(745), C(746), C(748), C(750), C(753), C(754), C(756), C(758), C(761), C(762), C(764),
C(766), C(769), C(770), C(772), C(774), C(777), C(778), C(780), C(782), C(785), C(786), C(788), C(790),
C(793), C(794), C(796), C(798), C(801), C(802), C(804), C(806), C(809), C(810), C(812), C(814), C(817),
C(818), C(820), C(822), C(825), C(826), C(828), C(830), C(833), C(834), C(836), C(838), C(841), C(842),
C(844), C(846), C(849), C(850), C(852), C(854), C(857), C(858), C(860), C(862), C(865), C(866), C(868),
C(870), C(873), C(874), C(876), C(878), C(881), C(882), C(884), C(886), C(889), C(890), C(892), C(894),
C(897), C(898), C(900), C(902), C(905), C(906), C(908), C(910), C(913), C(914), C(916), C(918), C(921),
C(922), C(924), C(926), C(929), C(930), C(932), C(934), C(937), C(938), C(940), C(942), C(945), C(946),
C(948), C(950), C(953), C(954), C(956), C(958), C(961), C(962), C(964), C(966), C(969), C(970), C(972),
C(974), C(977), C(978), C(980), C(982), C(985), C(986), C(988), C(990), C(993), C(994), C(996), C(998),
C(1001), C(1002), C(1004), C(1006), C(1009), C(1010), C(1012), C(1014), C(1017), C(1018), C(1020),
C(1022), C(1025), C(1026), C(1028), C(1030), C(1033), C(1034), C(1036), C(1038), C(1041), C(1042),
C(1044), C(1046), C(1049), C(1050), C(1052), C(1054), C(1057), C(1058), C(1060), C(1062), C(1065),
C(1066), C(1068), C(1070), C(1073), C(1074), C(1076), C(1078), C(1081), C(1082), C(1084), C(1086),
C(1089), C(1090), C(1092), C(1094), C(1097), C(1098), C(1100), C(1102), C(1105), C(1106), C(1108),
C(1110), C(1113), C(1114), C(1116), C(1118), C(1121), C(1122), C(1124), C(1126), C(1129), C(1130),
C(1132), C(1134), C(1137), C(1138), C(1140), C(1142), C(1145), C(1146), C(1148), C(1150), C(1153),
C(1154), C(1156), C(1158), C(1161), C(1162), C(1164), C(1166), C(1169), C(1170), C(1172), C(1174),
C(1177), C(1178), C(1180), C(1182), C(1185), C(1186), C(1188), C(1190), C(1193), C(1194), C(1196),
C(1198), C(1201), C(1202), C(1204), C(1206), C(1209), C(1210), C(1212), C(1214), C(1217), C(1218),
C(1220), C(1222), C(1225), C(1226), C(1228), C(1230), C(1233), C(1234), C(1236), C(1238), C(1241),
C(1242), C(1244), C(1246), C(1249), C(1250), C(1252), C(1254), C(1257), C(1258), C(1260), C(1262),
C(1265), C(1266), C(1268), C(1270), C(1273), C(1274), C(1276), C(1278), C(1281), C(1282), C(1284),
C(1286), C(1289), C(1290), C(1292), C(1294), C(1297), C(1298), C(1300), C(1302), C(1305), C(1306),
C(1308), C(1310), C(1313), C(1314), C(1316), C(1318), C(1321), C(1322), C(1324), C(1326), C(1329),
C(1330), C(1332), C(1334), C(1337), C(1338), C(1340), C(1342), C(1345), C(1346), C(1348), C(1350),
C(1353), C(1354), C(1356), C(1358), C(1361), C(1362), C(1364), C(1366), C(1369), C(1370), C(1372),
C(1374), C(1377), C(1378), C(1380), C(1382), C(1385), C(1386), C(1388), C(1390), C(1393), C(1394),
C(1396), C(1398), C(1401), C(1402), C(1404), C(1406), C(1409), C(1410), C(1412), C(1414), C(1417),
C(1418), C(1420), C(1422), C(1425), C(1426), C(1428), C(1430), C(1433), C(1434), C(1436), C(1438),
C(1441), C(1442), C(1446), C(1449), C(1450), C(1454), C(1457), C(1458), C(1462), C(1465), C(1466),
C(1470), C(1473), C(1474), C(1478), C(1481), C(1482), C(1484), C(1486), C(1489), C(1490), C(1492),
C(1494), C(1497), C(1498), C(1502), C(1505), C(1506), C(1510)
```

#### And the following 448 coded bits are moved to data block P<sub>B</sub>:

```
C(51), C(53), C(55), C(59), C(63), C(69), C(71), C(75), C(79), C(85), C(87), C(91), C(95), C(99), C(101),
C(103), C(107), C(111), C(117), C(119), C(123), C(127), C(133), C(135), C(139), C(143), C(147), C(149),
C(151), C(155), C(159), C(165), C(167), C(171), C(175), C(181), C(183), C(187), C(191), C(195), C(197),
C(199), C(203), C(207), C(213), C(215), C(219), C(223), C(229), C(231), C(235), C(239), C(243), C(245),
C(247), C(251), C(255), C(261), C(263), C(267), C(271), C(277), C(279), C(283), C(287), C(291), C(293),
C(295), C(299), C(303), C(309), C(311), C(315), C(319), C(325), C(327), C(331), C(335), C(339), C(341),
C(343), C(347), C(351), C(357), C(359), C(363), C(367), C(373), C(375), C(379), C(383), C(387), C(389),
C(391), C(395), C(399), C(405), C(407), C(411), C(415), C(421), C(423), C(427), C(431), C(435), C(437),
C(439), C(443), C(447), C(453), C(455), C(459), C(463), C(469), C(471), C(475), C(479), C(483), C(485),
C(487), C(491), C(495), C(501), C(503), C(507), C(511), C(517), C(519), C(523), C(527), C(531), C(533),
C(535), C(539), C(543), C(549), C(551), C(555), C(559), C(565), C(567), C(571), C(575), C(579), C(581),
C(583), C(587), C(591), C(597), C(599), C(603), C(607), C(613), C(615), C(619), C(623), C(627), C(629),
C(631), C(635), C(639), C(645), C(647), C(651), C(655), C(661), C(663), C(667), C(671), C(672), C(675),
C(677), C(680), C(683), C(685), C(688), C(691), C(693), C(696), C(699), C(701), C(704), C(707), C(709),
C(712), C(715), C(717), C(720), C(723), C(725), C(728), C(731), C(733), C(736), C(739), C(741), C(744),
C(747), C(749), C(752), C(755), C(757), C(760), C(763), C(765), C(768), C(771), C(773), C(776), C(779),
C(781), C(784), C(787), C(789), C(792), C(795), C(797), C(800), C(803), C(805), C(808), C(811), C(813),
C(816), C(819), C(821), C(824), C(827), C(829), C(832), C(835), C(837), C(840), C(843), C(845), C(848),
C(851), C(853), C(856), C(859), C(861), C(864), C(867), C(869), C(872), C(875), C(877), C(880), C(883),
C(885), C(888), C(891), C(893), C(896), C(899), C(901), C(904), C(907), C(909), C(912), C(915), C(917),
C(920), C(923), C(925), C(928), C(931), C(933), C(936), C(939), C(941), C(944), C(947), C(949), C(952),
C(955), C(957), C(960), C(963), C(965), C(968), C(971), C(973), C(976), C(979), C(981), C(984), C(987),
C(989), C(992), C(995), C(997), C(1000), C(1003), C(1005), C(1008), C(1013), C(1016), C(1019), C(1021),
C(1024), C(1029), C(1032), C(1035), C(1037), C(1040), C(1045), C(1048), C(1051), C(1053), C(1056),
```

```
C(1061), C(1064), C(1067), C(1069), C(1072), C(1077), C(1080), C(1083), C(1085), C(1088), C(1093), C(1096), C(1099), C(1101), C(1104), C(1109), C(1112), C(1115), C(1117), C(1120), C(1125), C(1128), C(1131), C(1133), C(1136), C(1141), C(1144), C(1147), C(1149), C(1152), C(1157), C(1160), C(1163), C(1165), C(1168), C(1173), C(1176), C(1179), C(1181), C(1184), C(1189), C(1192), C(1195), C(1197), C(1200), C(1205), C(1208), C(1211), C(1213), C(1216), C(1221), C(1224), C(1227), C(1229), C(1232), C(1237), C(1240), C(1243), C(1245), C(1248), C(1253), C(1256), C(1259), C(1261), C(1264), C(1269), C(1272), C(1275), C(1277), C(1280), C(1285), C(1288), C(1291), C(1293), C(1296), C(1301), C(1304), C(1307), C(1309), C(1312), C(1317), C(1320), C(1323), C(1325), C(1328), C(1333), C(1336), C(1339), C(1341), C(1344), C(1349), C(1352), C(1355), C(1357), C(1360), C(1365), C(1368), C(1371), C(1373), C(1376), C(1381), C(1384), C(1387), C(1389), C(1392), C(1397), C(1400), C(1403), C(1405), C(1408), C(1413), C(1416), C(1448), C(1451), C(1452), C(1453), C(1456), C(1460), C(1461), C(1464), C(1467), C(1468), C(1469), C(1472), C(1472), C(1476), C(1488)
```

The vectors P<sub>G</sub> and P<sub>B</sub> of coded and punctured bits is combined with in band bits to vector P<sub>C</sub>' as

$P_{C}'(k) = ic(k)$	for $k = 0, 1, 2, 3, 4, 5, 6, 7$
$P_{C}'(k+8) = P_{G}(k)$	for $k = 0, 1,, 447$
$P_{C}'(k+448) = ic(k)$	for k = 8, 9, 10, 11, 12, 13, 14, 15
$P_{C}'(k+16) = P_{G}(k)$	for k = 448, 449,, 895
$P_{C}'(k+896) = ic(k)$	for k = 16, 17, 18, 19, 20, 21, 22, 23
$P_{C}'(k+920) = P_{B}(k)$	for k = 0, 1,, 447

#### **O-TCH/WFS6.60**:

The block of 138 bits  $\{u(0)...u(137)\}$  is encoded with the 1/10 rate convolutional code defined by the following polynomials:

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G5/G7 = 1 + D + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G5/G7 = 1 + D + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G5/G7 = 1 + D + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G6/G7 = 1 + D + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G6/G7 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G6/G7 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G6/G7 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G6/G7 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G7/G7 = 1$$

resulting in 1440 coded bits,  $\{C(0)...C(1439)\}\$  defined by:

$$r(k) = u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-6)$$

$$C(10k) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(10k+1) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(10k+2) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(10k+3) = r(k) + r(k-1) + r(k-4) + r(k-6)$$

$$C(10k+4) = r(k) + r(k-1) + r(k-4) + r(k-6)$$

$$\begin{split} &C(10k+5) = r(k) + r(k-1) + r(k-4) + r(k-6) \\ &C(10k+6) = r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ &C(10k+7) = r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ &C(10k+8) = r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ &C(10k+9) = u(k) \end{split}$$
 for  $k = 0, 1, ..., 182;$   $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(10k) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(10k+1) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(10k+2) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(10k+3) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(10k+4) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(10k+5) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(10k+6) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(10k+7) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(10k+8) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(10k+9) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) & \text{for } k = 183, 184, ..., 188 \end{split}$$

The following 896 coded bits are moved to data block P<sub>G</sub>:

```
C(0), C(6), C(9), C(10), C(16), C(19), C(20), C(23), C(26), C(29), C(30), C(33), C(34), C(36), C(37), C(39),
C(40), C(41), C(42), C(43), C(44), C(46), C(47), C(49), C(50), C(51), C(52), C(53), C(54), C(56), C(57),
C(59), C(60), C(61), C(62), C(63), C(64), C(65), C(66), C(67), C(69), C(70), C(71), C(72), C(73), C(74),
C(76), C(77), C(79), C(80), C(81), C(83), C(84), C(86), C(87), C(89), C(90), C(91), C(93), C(94), C(96),
C(97), C(99), C(100), C(101), C(103), C(104), C(105), C(106), C(107), C(109), C(110), C(111), C(113),
C(114), C(116), C(117), C(119), C(120), C(121), C(123), C(124), C(126), C(127), C(129), C(130), C(131),
C(133), C(134), C(136), C(137), C(139), C(140), C(141), C(143), C(144), C(145), C(146), C(147), C(149),
C(150), C(151), C(153), C(154), C(156), C(157), C(159), C(160), C(161), C(163), C(164), C(166), C(167),
C(169), C(170), C(171), C(173), C(174), C(176), C(177), C(179), C(180), C(181), C(183), C(184), C(186),
C(187), C(189), C(190), C(191), C(193), C(194), C(196), C(197), C(199), C(200), C(201), C(203), C(204),
C(206), C(207), C(209), C(210), C(211), C(213), C(214), C(216), C(217), C(219), C(220), C(221), C(223),
C(224), C(226), C(227), C(229), C(230), C(231), C(233), C(234), C(236), C(237), C(239), C(240), C(241),
C(243), C(244), C(246), C(247), C(249), C(250), C(251), C(253), C(254), C(256), C(257), C(259), C(260),
C(261), C(263), C(264), C(266), C(267), C(269), C(270), C(271), C(273), C(274), C(276), C(277), C(279),
C(280), C(281), C(283), C(284), C(286), C(287), C(289), C(290), C(291), C(293), C(294), C(296), C(297),
C(299), C(300), C(301), C(303), C(304), C(306), C(307), C(309), C(310), C(311), C(313), C(314), C(316),
C(317), C(319), C(320), C(321), C(323), C(324), C(326), C(327), C(329), C(330), C(331), C(333), C(334),
C(336), C(337), C(339), C(340), C(341), C(343), C(344), C(346), C(347), C(349), C(350), C(351), C(353),
C(354), C(356), C(357), C(359), C(360), C(361), C(363), C(364), C(366), C(367), C(369), C(370), C(371),
C(373), C(374), C(376), C(377), C(379), C(380), C(381), C(383), C(384), C(386), C(387), C(389), C(390),
C(391), C(393), C(394), C(396), C(397), C(399), C(400), C(401), C(403), C(404), C(406), C(407), C(409),
C(410), C(411), C(413), C(414), C(416), C(417), C(419), C(420), C(421), C(423), C(424), C(426), C(427),
C(429), C(430), C(431), C(433), C(434), C(436), C(437), C(439), C(440), C(441), C(443), C(444), C(446),
C(447), C(449), C(450), C(451), C(453), C(454), C(456), C(457), C(459), C(460), C(461), C(463), C(464),
C(466), C(467), C(469), C(470), C(471), C(473), C(474), C(476), C(477), C(479), C(480), C(481), C(483),
C(484), C(486), C(487), C(489), C(490), C(491), C(493), C(494), C(496), C(497), C(499), C(500), C(501),
C(503), C(504), C(506), C(507), C(509), C(510), C(511), C(513), C(514), C(516), C(517), C(519), C(520),
C(521), C(523), C(524), C(526), C(527), C(529), C(530), C(531), C(533), C(534), C(536), C(537), C(539),
C(540), C(541), C(543), C(544), C(546), C(547), C(549), C(550), C(551), C(553), C(554), C(556), C(557), C(557), C(568), C(56
C(559), C(560), C(561), C(563), C(564), C(566), C(567), C(569), C(570), C(571), C(573), C(574), C(576),
```

```
C(577), C(579), C(580), C(581), C(583), C(584), C(586), C(587), C(589), C(590), C(591), C(593), C(594),
C(596), C(597), C(599), C(600), C(601), C(603), C(604), C(606), C(607), C(609), C(610), C(611), C(613),
C(614), C(616), C(617), C(619), C(620), C(621), C(623), C(624), C(626), C(627), C(629), C(630), C(631),
C(633), C(634), C(636), C(637), C(639), C(640), C(641), C(643), C(644), C(646), C(647), C(649), C(650),
C(651), C(653), C(654), C(656), C(657), C(659), C(660), C(661), C(663), C(664), C(666), C(667), C(669),
C(670), C(671), C(673), C(674), C(676), C(677), C(679), C(680), C(681), C(683), C(686), C(687), C(689),
C(690), C(691), C(693), C(696), C(697), C(699), C(700), C(701), C(703), C(706), C(707), C(709), C(710),
C(711), C(713), C(716), C(719), C(720), C(721), C(723), C(726), C(727), C(729), C(730), C(731), C(733),
C(736), C(737), C(739), C(740), C(741), C(743), C(746), C(747), C(749), C(750), C(751), C(753), C(756),
C(759), C(760), C(761), C(763), C(766), C(767), C(769), C(770), C(771), C(773), C(776), C(777), C(779),
C(780), C(781), C(783), C(786), C(787), C(789), C(790), C(791), C(793), C(796), C(799), C(800), C(801),
C(803), C(806), C(807), C(809), C(810), C(811), C(813), C(816), C(817), C(819), C(820), C(821), C(823),
C(826), C(827), C(829), C(830), C(831), C(833), C(836), C(839), C(840), C(841), C(843), C(846), C(847),
C(849), C(850), C(851), C(853), C(856), C(857), C(859), C(860), C(861), C(863), C(866), C(867), C(869),
C(870), C(871), C(873), C(876), C(879), C(880), C(881), C(883), C(886), C(887), C(889), C(890), C(891),
C(893), C(896), C(897), C(899), C(900), C(901), C(903), C(906), C(907), C(909), C(910), C(911), C(913),
C(916), C(919), C(920), C(921), C(923), C(926), C(927), C(929), C(930), C(931), C(933), C(936), C(937),
C(939), C(940), C(941), C(943), C(946), C(947), C(949), C(950), C(951), C(953), C(956), C(959), C(960),
C(961), C(963), C(966), C(967), C(969), C(970), C(971), C(973), C(976), C(977), C(979), C(980), C(981),
C(983), C(986), C(987), C(989), C(990), C(991), C(993), C(996), C(999), C(1000), C(1001), C(1003),
C(1006), C(1007), C(1009), C(1010), C(1011), C(1013), C(1016), C(1017), C(1019), C(1020), C(1021),
C(1023), C(1026), C(1027), C(1029), C(1030), C(1031), C(1033), C(1036), C(1039), C(1040), C(1041),
C(1043), C(1046), C(1047), C(1049), C(1050), C(1051), C(1053), C(1056), C(1057), C(1059), C(1060),
C(1061), C(1063), C(1066), C(1067), C(1069), C(1070), C(1071), C(1073), C(1076), C(1079), C(1080),
C(1081), C(1083), C(1086), C(1087), C(1089), C(1090), C(1091), C(1093), C(1096), C(1097), C(1099), C
C(1100), C(1101), C(1103), C(1106), C(1107), C(1109), C(1110), C(1111), C(1113), C(1116), C(1119),
C(1120), C(1121), C(1123), C(1126), C(1127), C(1129), C(1130), C(1131), C(1133), C(1136), C(1137),
C(1139), C(1140), C(1141), C(1143), C(1146), C(1147), C(1149), C(1150), C(1151), C(1153), C(1156),
C(1159), C(1160), C(1161), C(1163), C(1166), C(1167), C(1169), C(1170), C(1171), C(1173), C(1176),
C(1177), C(1179), C(1180), C(1181), C(1183), C(1186), C(1187), C(1189), C(1190), C(1191), C(1193),
C(1196), C(1199), C(1200), C(1201), C(1203), C(1206), C(1207), C(1209), C(1210), C(1211), C(1213),
C(1216), C(1217), C(1219), C(1220), C(1221), C(1223), C(1226), C(1227), C(1229), C(1230), C(1231),
C(1233), C(1236), C(1239), C(1240), C(1241), C(1243), C(1246), C(1247), C(1249), C(1250), C(1251),
C(1253), C(1256), C(1257), C(1259), C(1260), C(1261), C(1263), C(1266), C(1267), C(1269), C(1270),
C(1271), C(1273), C(1276), C(1279), C(1280), C(1281), C(1283), C(1286), C(1289), C(1290), C(1291),
C(1293), C(1296), C(1299), C(1300), C(1301), C(1303), C(1306), C(1309), C(1310), C(1311), C(1313),
C(1316), C(1319), C(1320), C(1321), C(1323), C(1326), C(1329), C(1330), C(1331), C(1333), C(1336),
C(1339), C(1340), C(1341), C(1343), C(1346), C(1349), C(1350), C(1351), C(1353), C(1356), C(1359),
C(1360), C(1361), C(1363), C(1366), C(1369), C(1370), C(1371), C(1373), C(1376), C(1379), C(1380),
C(1381), C(1383), C(1386), C(1389), C(1390), C(1391), C(1393), C(1396), C(1399), C(1400), C(1401),
C(1403), C(1406), C(1409), C(1410), C(1411), C(1413), C(1416), C(1419), C(1420), C(1421), C(1423),
C(1426), C(1429), C(1430), C(1431), C(1433), C(1436), C(1439)
```

#### And the following 448 coded bits are moved to data block P<sub>B</sub>:

```
C(45), C(48), C(55), C(58), C(68), C(75), C(78), C(82), C(85), C(88), C(92), C(95), C(98), C(102), C(108),
C(112), C(115), C(118), C(122), C(125), C(128), C(132), C(135), C(138), C(142), C(148), C(152), C(155),
C(158), C(162), C(165), C(168), C(172), C(175), C(178), C(182), C(185), C(188), C(192), C(195), C(198),
C(202), C(205), C(208), C(212), C(215), C(218), C(222), C(225), C(228), C(232), C(235), C(238), C(242),
C(245), C(248), C(252), C(255), C(258), C(262), C(265), C(268), C(272), C(275), C(278), C(282), C(285),
C(288), C(292), C(295), C(298), C(302), C(305), C(308), C(312), C(315), C(318), C(322), C(325), C(328),
C(332), C(335), C(338), C(342), C(345), C(348), C(352), C(355), C(358), C(362), C(365), C(368), C(372),
C(375), C(378), C(382), C(385), C(388), C(392), C(395), C(398), C(402), C(405), C(408), C(412), C(415),
C(418), C(422), C(425), C(428), C(432), C(435), C(438), C(442), C(445), C(448), C(452), C(455), C(458),
C(462), C(465), C(468), C(472), C(475), C(478), C(482), C(485), C(488), C(492), C(495), C(498), C(502),
C(505), C(508), C(512), C(515), C(518), C(522), C(525), C(528), C(532), C(535), C(538), C(542), C(545),
C(548), C(552), C(555), C(558), C(562), C(565), C(568), C(572), C(575), C(578), C(582), C(585), C(588),
C(592), C(595), C(598), C(602), C(605), C(608), C(612), C(615), C(618), C(622), C(625), C(628), C(632),
C(635), C(638), C(642), C(648), C(652), C(655), C(658), C(668), C(668), C(672), C(675), C(678), C(682),
C(684), C(688), C(692), C(694), C(698), C(702), C(704), C(708), C(712), C(714), C(715), C(717), C(718),
C(722), C(724), C(728), C(732), C(734), C(738), C(742), C(744), C(748), C(752), C(754), C(755), C(757),
```

```
C(758), C(762), C(764), C(768), C(772), C(774), C(778), C(782), C(784), C(788), C(792), C(794), C(795),
C(797), C(798), C(802), C(804), C(808), C(812), C(814), C(818), C(822), C(824), C(828), C(832), C(834),
C(835), C(837), C(838), C(842), C(844), C(848), C(852), C(854), C(858), C(862), C(864), C(868), C(872),
C(874), C(875), C(877), C(878), C(882), C(884), C(888), C(892), C(894), C(898), C(902), C(904), C(908),
C(912), C(914), C(915), C(917), C(918), C(922), C(924), C(928), C(932), C(934), C(938), C(942), C(944),
C(948), C(952), C(954), C(955), C(957), C(958), C(962), C(964), C(968), C(972), C(974), C(978), C(982),
C(984), C(988), C(992), C(994), C(995), C(997), C(998), C(1002), C(1004), C(1008), C(1012), C(1014),
C(1018), C(1022), C(1024), C(1028), C(1032), C(1034), C(1035), C(1037), C(1038), C(1042), C(1044),
C(1048), C(1052), C(1054), C(1058), C(1062), C(1064), C(1068), C(1072), C(1074), C(1075), C(1077),
C(1078), C(1082), C(1084), C(1088), C(1092), C(1094), C(1098), C(1102), C(1104), C(1108), C(1112),
C(1114), C(1115), C(1117), C(1118), C(1122), C(1124), C(1128), C(1132), C(1134), C(1138), C(1142),
C(1144), C(1148), C(1152), C(1154), C(1155), C(1157), C(1158), C(1162), C(1164), C(1168), C(1172),
C(1174), C(1178), C(1182), C(1184), C(1188), C(1192), C(1194), C(1195), C(1197), C(1198), C(1202),
C(1204), C(1208), C(1212), C(1214), C(1218), C(1222), C(1224), C(1228), C(1232), C(1234), C(1235),
C(1237), C(1238), C(1242), C(1244), C(1248), C(1252), C(1254), C(1258), C(1262), C(1264), C(1268),
C(1272), C(1274), C(1275), C(1277), C(1278), C(1282), C(1284), C(1287), C(1288), C(1292), C(1294),
C(1297), C(1298), C(1302), C(1304), C(1307), C(1308), C(1312), C(1314), C(1315), C(1317), C(1318),
C(1322), C(1324), C(1327), C(1328), C(1332), C(1334), C(1337), C(1338), C(1342), C(1344), C(1347),
C(1348), C(1352), C(1354), C(1355), C(1357), C(1358), C(1362), C(1364), C(1367), C(1368), C(1372),
C(1374), C(1377), C(1378), C(1382), C(1384), C(1387), C(1394), C(1397), C(1404), C(1407), C(1414),
C(1417), C(1424), C(1427), C(1434), C(1437)
```

The vectors  $P_G$  and  $P_B$  of coded and punctured bits is combined with in band bits to vector  $P_{C}$  as

$P_{C}'(k) = ic(k)$	for $k = 0, 1, 2, 3, 4, 5, 6, 7$
$P_{C}'(k+8) = P_{G}(k)$	for $k = 0, 1,, 447$
$P_{C}'(k+448) = ic(k)$	for k = 8, 9, 10, 11, 12, 13, 14, 15
$P_{C}'(k+16) = P_{G}(k)$	for k = 448, 449,, 895
$P_{C}'(k+896) = ic(k)$	for k = 16, 17, 18, 19, 20, 21, 22, 23
$P_{C}'(k+920) = P_{B}(k)$	for $k = 0, 1,, 447$

#### 3.16.4.5 Interleaving

Before interleaving the bits  $\{P_C'(0) \dots P_C'(1367)\}$  are converted into 3-bit symbols  $\{c(0) \dots c(455)\}$  according to table 1 in 3GPP TS 45.004, the symbol c(k) consists of  $d_{3k}=P_C'(k)$ ,  $d_{3k+1}=P_C'(k+456)$  and  $d_{3k+2}=P_C'(k+912)$  for  $k=0,1,\dots,456$ . The interleaving is done as specified for the TCH/FS in subclause 3.1.3. The difference is that the interleaving is done by symbols instead of single bits, reusing the existing interleaving tables.

# 3.16.4.6 Mapping on a Burst

The mapping is done as specified for the TCH/FS in subclause 3.1.4 with exception that it is done by symbols instead of single bits.

## 3.16.5 RATSCCH

The RATSCCH message consists of 35 bits. Also delivered are two in-band channels, id0(0,1) and id1(0,1), id0 corresponding to Mode Commands or Mode Requests and id1 to Mode Indication. The general coding is as: the two inband data channels are coded to 16 bits each, a 14-bit CRC is added to the 35 RATSCCH bits which are then coded by a rate ½ RSC coder to 212 bits. Finally a 212 bit identification field is added thereby giving a total size of 456 bits. These 456 bits are then block interleaved in the same way as a normal speech frame.

#### 3.16.5.1 Coding of in-band data

The coding of in-band data is done as specified for the RATSCCH message in TCH/AFS (subclause 3.9.5.1).

# 3.16.5.2 Parity and convolutional encoding for the RATSCCH message

The parity and convolutional encoding for the RATSCCH message are done as specified for the RATSCCH message in TCH/AFS (subclause 3.9.5.2).

#### 3.16.5.3 Identification marker

The identification marker is done as specified for the RATSCCH message in TCH/AFS (subclause 3.9.5.3).

#### 3.16.5.4 Interleaving

Before interleaving the bits are repeated 3 times:

$$c'(3k+2) = c'(3k+1) = c'(3k) = c(k)$$
 for k=0,...,455

The bits  $\{c'(0) \dots c'(1367)\}$  are then converted into 3-bit symbols  $\{C(0) \dots C(455)\}$  according to table 1 in 3GPP TS 45.004, the symbol C(k) depends on c'(3k+2), c'(3k+1) and c'(3k) for  $k=0,1,\dots,455$ .

The interleaving is done as specified for the TCH/AFS (subclause 3.9.5.4). The difference is that the interleaving is done by symbols instead of single bits, reusing the existing interleaving tables.

# 3.16.5.5 Mapping on a Burst

The mapping is done as specified for the TCH/AFS (subclause 3.9.5.5) with exception that it is done by symbols instead of single bits.

# 3.17 Wideband Adaptive multi rate speech channel at 8-PSK half rate (O-TCH/WHS)

This section describes the coding for the different frame formats used for O-TCH/WHS. The formats used are (in the order they are described):

SID\_UPDATE Used to convey comfort noise parameters during DTX

SID\_UPDATE\_INH Used to inhibit the second part of a SID\_UPDATE frame if there is a speech onset

SID\_FIRST\_P1 First part of marker to define end of speech, start of DTX

SID\_FIRST\_P2 Second part of marker to define end of speech, start of DTX

SID\_FIRST\_INH Used to inhibit the second part of a SID\_FIRST\_P1 frame if there is a speech onset

ONSET Used to signal the Codec mode for the first speech frame after DTX

SPEECH Speech frames

RATSCCH\_MARKER Marker to identify RATSCCH frames

RATSCCH\_DATA Frame that conveys the actual RATSCCH message

In this chapter, sub chapters 3.17.1 to 3.17.9 describe the channel coding for the different formats listed above.

Common to all the formats is that in-band information is conveyed, the coding for the in-band channel is described in the table below.

Identifier (defined in 3GPP 45.009)	Received in-band data id(1), id(0)	Encoded in-band data for SID and RATSCCH frames ic(15),, ic(0)	Encoded in-band data for speech frames ic(11),, ic(0)
CODEC_MODE_1	00	0101001100001111	00000000000
CODEC_MODE_2	01	0011111010111000	110110101110
CODEC_MODE_3	10	1000100001100011	101101110101
CODEC_MODE_4	11	1110010111010100	011011011011

# **3.17.1 SID UPDATE**

The speech encoder delivers 35 bits of comfort noise parameters. Also delivered is two in-band channels, id0(0,1) and id1(0,1), id0 corresponding to Mode Commands/Mode Requests and id1 to Mode Indication. The general coding is as: the two in-band data channels are coded to 16 bits each, a 14-bit CRC is added to the 35 CN bits which are then coded by a rate ½ RSC coder to 212 bits. A 212 bit identification field is added thereby giving a total size of 456 bits. Finally each bit is repeated 3 times and then converted into 3-bit symbols giving a total size of 456 symbols. These 456 symbols are block interleaved over 4 bursts.

# 3.17.1.1 Coding of in-band data

The coding of in-band data is done as specified for the SID UPDATE frame in O-TCH/AHS (subclause 3.15.1.1).

# 3.17.1.2 Parity and convolutional encoding for the comfort noise parameters

The parity and convolutional encoding for the comfort noise parameters are done as specified for the SID\_UPDATE frame in O-TCH/AHS (subclause 3.15.1.2).

#### 3.17.1.3 Identification marker

The identification marker is constructed as specified for the SID\_UPDATE frame in O-TCH/AHS (subclause 3.15.1.3).

#### 3.17.1.4 Repetition

The repetition is done as specified for the SID\_UPDATE frame in O-TCH/AHS (subclause 3.15.1.4).

#### 3.17.1.5 Interleaving

The interleaving is done as specified for the SID\_UPDATE frame in O-TCH/AHS (subclause 3.15.1.5).

#### 3.17.1.6 Mapping on a Burst

The mapping is done as specified for the SID\_UPDATE frame in O-TCH/AHS (subclause 3.15.1.6).

# 3.17.2 SID\_UPDATE\_INH

This special frame is used when the first 2 burst of a SID\_UPDATE frame have been transmitted but the second two bursts cannot be transmitted due to a speech frame. The general coding is as: the in-band data (Note that this must be the same Mode Indication bits as id1(0,1) for the SID\_UPDATE frame that is being inhibited) is encoded, a marker that is the opposite of the SID\_UPDATE marker is appended and the data is interleaved in such a way that the odd symbols of two bursts are filled.

## 3.17.2.1 Coding of in-band data

The coding of in-band data is done as specified for the SID\_UPDATE\_INH frame in O-TCH/AHS (subclause 3.15.2.1).

#### 3.17.2.2 Identification marker

The identification marker is constructed as specified for the SID\_UPDATE\_INH frame in O-TCH/AHS (subclause 3.15.2.2).

## 3.17.2.3 Repetition

The repetition is done as specified for the SID UPDATE INH frame in O-TCH/AHS (subclause 3.15.2.3).

#### 3.17.2.4 Interleaving

The interleaving is done as specified for the SID\_UPDATE\_INH frame in O-TCH/AHS (subclause 3.15.2.4).

## 3.17.2.5 Mapping on a Burst

The mapping is done as specified for the SID UPDATE INH frame in O-TCH/AHS (subclause 3.15.2.5).

# 3.17.3 SID FIRST P1

This frame type contains no source data from the speech coder. What is generated is the in-band channel and an identification marker. The in-band data id(0,1) represents Mode Indication or Mode Command/Mode Request depending on the current frame number.

## 3.17.3.1 Coding of in-band data

The coding of in-band data is done as specified for the SID\_FIRST\_P1 frame in O-TCH/AHS (subclause 3.15.3.1).

#### 3.17.3.2 Identification marker

The identification marker is constructed as specified for the SID\_FIRST\_P1 frame in O-TCH/AHS (subclause 3.15.3.2).

#### 3.17.3.3 Repetition

The repetition is done as specified for the SID\_FIRST\_P1 frame in O-TCH/AHS (subclause 3.15.3.3).

## 3.17.3.4 Interleaving

The interleaving is done as specified for the SID\_FIRST\_P1 frame in O-TCH/AHS (subclause 3.15.3.4).

## 3.17.3.5 Mapping on a Burst

The mapping is done as specified for the SID\_FIRST\_P1 frame in O-TCH/AHS (subclause 3.15.3.5).

# 3.17.4 SID FIRST P2

This frame type contains no source data from the speech coder. What is generated is the in-band channel and, derived from that, an identification marker. The in-band data id(0,1) represents Mode Indication or Mode Command/Mode Request depending on the current frame number.

# 3.17.4.1 Coding of in-band data

The coding of in-band data is done as specified for the SID\_FIRST\_P2 frame in O-TCH/AHS (subclause 3.15.4.1).

#### 3.17.4.2 Repetition

The repetition is done as specified for the SID\_FIRST\_P2 frame in O-TCH/AHS (subclause 3.15.4.2).

## 3.17.4.3 Interleaving

The interleaving is done as specified for the SID\_FIRST\_P2 frame in O-TCH/AHS (subclause 3.15.4.3).

# 3.17.4.4 Mapping on a Burst

The mapping is done as specified for the SID\_FIRST\_P2 frame in O-TCH/AHS (subclause 3.15.4.4).

# 3.17.5 SID\_FIRST\_INH

This special frame is used when the first 2 burst of a SID\_FIRST\_P1 frame have been transmitted but the second two bursts cannot be transmitted due to a SPEECH frame. The general coding is as: the in-band data (Note that this must be the same data as for the SID\_FIRST\_P1 frame that is being inhibited) is encoded, a marker that is the opposite of the SID\_FIRST\_P1 marker is appended and the data is interleaved in such a way that the odd symbols of two bursts are filled.

## 3.17.5.1 Coding of in-band data

The coding of the in-band data is done as specified for the SID\_FIRST\_INH frame in O-TCH/AHS (subclause 3.15.5.1).

#### 3.17.5.2 Identification marker

The identification marker is done as specified for the SID\_FIRST\_INH frame in O-TCH/AHS (subclause 3.15.5.2).

## 3.17.5.3 Repetition

The repetition is done as specified for the SID\_FIRST\_INH frame in O-TCH/AHS (subclause 3.15.5.3).

## 3.17.5.4 Interleaving

The interleaving is done as specified for the SID\_FIRST\_INH frame in O-TCH/AHS (subclause 3.15.5.4).

#### 3.17.5.5 Mapping on a Burst

The mapping is done as specified for the SID\_FIRST\_INH frame in O-TCH/AHS (subclause 3.15.5.5).

#### 3.17.6 ONSET

Onset frames are used to preset the interleaver buffer after a period of no speech activity in DTX mode. This frame type contains no source data from the speech coder. What is transmitted is the in-band channel signalling the Mode Indication for the speech frame following the onset marker.

## 3.17.6.1 Coding of in-band data

The coding of in-band data is done as specified for the ONSET frame in O-TCH/AHS (subclause 3.15.6.1).

# 3.17.6.2 Repetition

The repetition is done as specified for the ONSET frame in O-TCH/AHS (subclause 3.15.6.2).

## 3.17.6.3 Interleaving

The interleaving is done as specified for the ONSET frame in O-TCH/AHS (subclause 3.15.6.3).

## 3.17.6.4 Mapping on a Burst

The mapping is done as specified for the ONSET frame in O-TCH/AHS (subclause 3.15.6.4).

# 3.17.7 SPEECH

The speech coder delivers to the channel encoder a sequence of blocks of data. One block of data corresponds to one speech frame and the block length is different in each of the nine channel codec modes. Adjoining each block of data is information of the channel codec mode to use when encoding the block. Also delivered is the in-band data id(0,1) representing Mode Indication or Mode Command/Mode Request depending on the current frame number.

## 3.17.7.1 Coding of the in-band data

The two input in-band bits (id(0,1)) are coded to twelve coded in-band bits (ic(0..11)).

The encoded in-band bits are moved to the coded bits, c, as

$$c(k) = ic(k)$$
 for  $k = 0, 1, ..., 11$ .

# 3.17.7.2 Ordering according to subjective importance

The bits delivered by the speech encoder,  $\{s(1),s(2),...,s(K_s)\}$ , are rearranged according to subjective importance before channel coding. Tables 16 to 18 define the correct rearrangement for the speech codec modes 12.65 kbit/s, 8.85 kbit/s and 6.60 kbit/s, respectively. In the tables speech codec parameters are numbered in the order they are delivered by the corresponding speech encoder according to 3GPP TS 26.190 and the rearranged bits are labelled  $\{d(0),d(1),...,d(K_d-1)\}$ , defined in the order of decreasing importance. Index  $K_d$  refers to the number of bits delivered by the speech encoder, see below:

Codec mode	Number of speech bits delivered per block (K <sub>d</sub> )
O-TCH/WHS12.65	253
O-TCH/WHS8.85	177
O-TCH/WHS6.60	132

The ordering algorithm is in pseudo code as:

for 
$$j = 0$$
 to  $K_{d-1}$   $d(j) := s(table(j)+1)$ ; where  $table(j)$  is read line by line left to right

The rearranged bits are further divided into two different classes to perform unequal error protection for different bits according to subjective importance.

The protection classes are:

- 1a Data protected with the CRC and the convolution code.
- 1b Data protected with the convolution code.

The number of class 1 (sum of class 1a and 1b), class 1a and class 1b bits for each codec mode is shown below:

Codec mode	Number of speech bits delivered per block	Number of class 1 bits per block	Number of Class 1a bits per block	Number of class 1b bits per block
O-TCH/WHS12.65	253	253	72	181
O-TCH/WHS8.85	177	177	64	113
O-TCH/WHS6.60	132	132	54	78

## 3.17.7.3 Parity for speech frames

The basic parameters for each codec mode for the first encoding step are shown below:

Codec mode	Number of class 1 bits (K <sub>d1</sub> )	CRC Protected bits (K <sub>d1a</sub> )	CRC bits	Number of bits after first encoding step (K <sub>u</sub> = K <sub>d</sub> + 6)
O-TCH/WHS12.65	253	72	6	259
O-TCH/WHS8.85	177	64	6	183
O-TCH/WHS6.60	132	54	6	138

A 6-bit CRC is used for error-detection. These parity bits are generated by the cyclic generator polynomial:  $g(D) = D^6 + D^5 + D^3 + D^2 + D^1 + I$  from the first  $K_{d1a}$  bits of class 1, where  $K_{d1a}$  refers to number of bits in protection class 1a as shown above for each codec mode. The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

$$d(0)D(K_{d1a}+5) + d(1)D(K_{d1a}+4) + ... + d(K_{d1a}-1)D(6) + p(0)D(5) + ... + p(4)D + p(5)$$

where p(0), p(1) ... p(5) are the parity bits, when divided by g(D), yields a remainder equal to:

$$1+D+D^2+D^3+D^4+D^5$$
.

The information and parity bits are merged:

 $u(k) = d(k) \qquad \qquad \text{for } k = 0, \, 1, \, ..., \, K_{d1a}\text{-}1$ 

 $u(k) = p(k - K_{d1a}) \hspace{1cm} \text{for } k = K_{d1a}, \, K_{d1a} + 1, \, \dots, \, K_{d1a} + 5$ 

u(k) = d(k-6) for  $k = K_{d1a} + 6, K_{d1a} + 7, ..., K_{u-1}$ 

#### O-TCH/WHS12.65:

u(k) = d(k) for k = 0, 1, ..., 71

u(k) = p(k-72) for k = 72, 73, ..., 77

u(k) = d(k-6) for k = 78, 79, ..., 258

## O-TCH/WHS8.85:

u(k) = d(k) for k = 0, 1, ..., 63

u(k) = p(k-64) for k = 64, 65, ..., 69

u(k) = d(k-6) for k = 70, 71, ..., 182

#### **O-TCH/WHS6.60**:

u(k) = d(k) for k = 0, 1, ..., 53

u(k) = p(k-54) for k = 54, 55, ..., 59

u(k) = d(k-6) for k = 60, 61, ..., 137

#### 3.17.7.4 Convolutional encoder

The bits from the first encoding step (u(k)) are encoded with the recursive systematic convolutional codes as summarised below. The number of output bits after puncturing is 672 for all codec modes.

Codec Mode	Rate	Number of input bits to conv. coder	Number of output bits from conv. Coder	Number Of Punctured bits
O-TCH/WHS12.65	1/3	259	795	123
O-TCH/WHS8.85	1/4	183	756	84
O-TCH/WHS6.60	1/5	138	720	48

Below the coding for each codec mode is specified in detail. The puncturing for each mode is designed to give an even protection of the class 1A bits while the protection within class 1B is not equal to reflect the individual error sensitivity of the class 1B bits.

#### O-TCH/WHS12.65:

The block of 259 bits  $\{u(0)...u(258)\}$  is encoded with the 1/3 rate convolutional code defined by the following polynomials:

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G5/G7 = 1 + D + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G7/G7 = 1$$

resulting in 795 coded bits,  $\{C(0)...C(794)\}\$  defined by:

$$\begin{split} r(k) &= u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-6) \\ C(3k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(3k+1) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(3k+2) &= u(k) \end{split} \qquad \qquad \text{for } k = 0, 1, ..., 258; \qquad r(k) = 0 \text{ for } k < 0.$$

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(3k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(3k+1) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(3k+2) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) \qquad \text{for } k = 259, 260, ..., 264 \end{split}$$

The following 448 coded bits are moved to data block P<sub>G</sub>:

```
C(2), C(5), C(6), C(8), C(9), C(11), C(12), C(14), C(15), C(17), C(18), C(20), C(21), C(23), C(24), C(26),
C(27), C(29), C(30), C(32), C(33), C(35), C(36), C(38), C(39), C(41), C(42), C(44), C(45), C(47), C(48),
C(50), C(51), C(53), C(54), C(56), C(57), C(59), C(60), C(62), C(63), C(65), C(66), C(68), C(69), C(71),
C(72), C(74), C(75), C(77), C(78), C(80), C(81), C(83), C(84), C(86), C(87), C(89), C(90), C(92), C(93),
C(95), C(96), C(98), C(99), C(101), C(102), C(104), C(105), C(107), C(108), C(110), C(111), C(113),
C(114), C(116), C(117), C(119), C(120), C(122), C(123), C(125), C(126), C(128), C(129), C(131), C(132),
C(134), C(135), C(137), C(138), C(140), C(141), C(143), C(144), C(146), C(147), C(149), C(150), C(152),
C(153), C(155), C(156), C(158), C(159), C(161), C(162), C(164), C(165), C(167), C(168), C(170), C(171),
C(173), C(174), C(176), C(177), C(179), C(180), C(182), C(183), C(185), C(186), C(188), C(189), C(191),
C(192), C(194), C(195), C(197), C(198), C(200), C(201), C(203), C(204), C(206), C(207), C(209), C(210), C(21
C(212), C(213), C(215), C(216), C(218), C(219), C(221), C(222), C(224), C(225), C(227), C(228), C(230),
C(231), C(233), C(234), C(236), C(237), C(239), C(240), C(242), C(243), C(245), C(246), C(248), C(249),
C(251), C(252), C(254), C(255), C(257), C(258), C(260), C(261), C(263), C(266), C(267), C(269), C(270),
C(272), C(275), C(276), C(278), C(279), C(281), C(282), C(284), C(287), C(290), C(291), C(293), C(294),
C(296), C(299), C(300), C(302), C(303), C(305), C(306), C(308), C(311), C(314), C(315), C(317), C(318),
C(320), C(323), C(324), C(326), C(327), C(329), C(330), C(332), C(335), C(338), C(339), C(341), C(342),
C(344), C(347), C(348), C(350), C(351), C(353), C(354), C(356), C(359), C(362), C(363), C(365), C(366),
```

```
C(368), C(371), C(372), C(374), C(375), C(377), C(378), C(380), C(383), C(386), C(387), C(389), C(390),
C(392), C(395), C(396), C(398), C(399), C(401), C(402), C(404), C(407), C(410), C(411), C(413), C(414),
C(416), C(419), C(420), C(422), C(423), C(425), C(426), C(428), C(431), C(434), C(435), C(437), C(438),
C(440), C(443), C(444), C(446), C(447), C(449), C(450), C(452), C(455), C(458), C(459), C(461), C(462),
C(464), C(467), C(468), C(470), C(471), C(473), C(474), C(476), C(479), C(482), C(483), C(485), C(486),
C(488), C(491), C(492), C(494), C(495), C(497), C(498), C(500), C(503), C(506), C(507), C(509), C(510),
C(512), C(515), C(516), C(518), C(519), C(521), C(522), C(524), C(527), C(530), C(531), C(533), C(534),
C(536), C(539), C(540), C(542), C(543), C(545), C(546), C(548), C(551), C(554), C(555), C(557), C(558),
C(560), C(563), C(564), C(566), C(567), C(569), C(570), C(572), C(575), C(578), C(579), C(581), C(582),
C(584), C(587), C(588), C(590), C(591), C(593), C(594), C(596), C(599), C(602), C(603), C(605), C(606),
C(608), C(611), C(612), C(614), C(615), C(617), C(618), C(620), C(623), C(626), C(627), C(629), C(630),
C(632), C(635), C(636), C(638), C(639), C(641), C(642), C(644), C(647), C(650), C(651), C(653), C(654),
C(656), C(659), C(660), C(662), C(663), C(665), C(666), C(668), C(671), C(674), C(675), C(677), C(678),
C(680), C(683), C(684), C(686), C(687), C(689), C(690), C(692), C(695), C(698), C(698), C(701), C(702),
C(704), C(707), C(708), C(710), C(711), C(713), C(714), C(716), C(719), C(722), C(723), C(725), C(726),
C(728), C(731), C(734), C(737), C(740), C(743), C(746), C(749), C(752), C(755), C(758), C(761), C(764),
C(767), C(770), C(773), C(776), C(779), C(782), C(785), C(788), C(791), C(794)
```

And the following 224 coded bits are moved to data block P<sub>B</sub>:

```
C(10), C(13), C(16), C(19), C(22), C(25), C(28), C(31), C(34), C(37), C(40), C(43), C(46), C(49), C(52),
C(55), C(58), C(61), C(64), C(67), C(70), C(73), C(76), C(79), C(82), C(85), C(88), C(91), C(94), C(97),
C(100), C(103), C(106), C(109), C(112), C(115), C(118), C(121), C(124), C(127), C(130), C(133), C(136),
C(139), C(142), C(145), C(148), C(151), C(154), C(157), C(160), C(163), C(166), C(169), C(172), C(175),
C(178), C(181), C(184), C(187), C(190), C(193), C(196), C(199), C(202), C(205), C(208), C(211), C(214),
C(217), C(220), C(223), C(226), C(229), C(232), C(235), C(238), C(241), C(244), C(247), C(250), C(253),
C(256), C(259), C(262), C(264), C(265), C(268), C(271), C(273), C(274), C(277), C(280), C(283), C(285),
C(286), C(288), C(289), C(292), C(295), C(297), C(298), C(301), C(304), C(307), C(309), C(310), C(312),
C(313), C(316), C(319), C(321), C(322), C(325), C(333), C(334), C(336), C(337), C(345), C(346), C(357),
C(358), C(360), C(361), C(369), C(370), C(381), C(384), C(385), C(393), C(394), C(405), C(408), C(409),
C(417), C(418), C(429), C(432), C(433), C(441), C(442), C(453), C(456), C(457), C(465), C(466), C(477),
C(480), C(481), C(489), C(490), C(501), C(504), C(505), C(513), C(514), C(525), C(528), C(529), C(537),
C(538), C(549), C(552), C(553), C(561), C(562), C(573), C(576), C(577), C(585), C(586), C(597), C(600),
C(601), C(609), C(610), C(621), C(624), C(625), C(633), C(634), C(645), C(648), C(649), C(657), C(658),
C(669), C(672), C(673), C(681), C(693), C(696), C(697), C(705), C(717), C(720), C(721), C(729), C(730),
C(732), C(735), C(736), C(738), C(741), C(744), C(745), C(747), C(750), C(753), C(754), C(756), C(759),
C(762), C(763), C(765), C(768), C(771), C(772), C(774), C(777), C(780), C(781), C(783), C(786)
```

The vectors P<sub>G</sub> and P<sub>B</sub> of coded and punctured bits is combined with in band bits to vector P<sub>C</sub> as

$P_{C}'(k) = ic(k)$	for $k = 0, 1, 2, 3$
$P_{C}'(k+4) = P_{G}(k)$	for $k = 0, 1,, 223$
$P_{C}'(k+224) = ic(k)$	for $k = 4, 5, 6, 7$
$P_{C}'(k+8) = P_{G}(k)$	for k = 224, 225,, 447
$P_{C}'(k+448) = ic(k)$	for $k = 8, 9, 10, 11$
$P_{C}'(k+460) = P_{B}(k)$	for $k = 0, 1,, 223$

#### **O-TCH/WHS8.85:**

The block of 183 bits  $\{u(0)...u(182)\}$  is encoded with the  $\frac{1}{4}$  rate convolutional code defined by polynomials:

$$G4/G7 = 1 + D^{2} + D^{3} + D^{5} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G5/G7 = 1 + D + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

$$G6/G7 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}/1 + D + D^{2} + D^{3} + D^{6}$$

G7/G7 = 1

resulting in 756 coded bits,  $\{C(0)...C(755)\}\$  defined by:

$$r(k) = u(k) + r(k-1) + r(k-2) + r(k-3) + r(k-6)$$

$$C(4k) = r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6)$$

$$C(4k+1) = r(k) + r(k-1) + r(k-4) + r(k-6)$$

$$C(4k+2) = r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6)$$

$$C(4k+3) = u(k)$$
 for  $k = 0, 1, ..., 182$ ;  $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$\begin{split} r(k) &= 0 \\ C(4k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ C(4k+1) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ C(4k+2) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ C(4k+3) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) & \text{for } k = 183, 184, ..., 188 \end{split}$$

The following 448 coded bits are moved to data block P<sub>G</sub>:

```
C(3), C(7), C(11), C(15), C(16), C(19), C(20), C(23), C(24), C(27), C(28), C(31), C(32), C(33), C(35),
C(36), C(39), C(40), C(41), C(43), C(44), C(47), C(48), C(49), C(51), C(52), C(55), C(56), C(57), C(59),
C(60), C(63), C(64), C(65), C(67), C(68), C(71), C(72), C(73), C(75), C(76), C(79), C(80), C(81), C(83),
C(84), C(87), C(88), C(89), C(91), C(92), C(95), C(96), C(97), C(99), C(100), C(103), C(104), C(105),
C(107), C(108), C(111), C(112), C(113), C(115), C(116), C(119), C(120), C(121), C(123), C(124), C(127),
C(128), C(129), C(131), C(132), C(135), C(136), C(137), C(139), C(140), C(143), C(144), C(145), C(147),
C(148), C(151), C(152), C(153), C(155), C(156), C(159), C(160), C(161), C(163), C(164), C(167), C(168),
C(169), C(171), C(172), C(175), C(176), C(177), C(179), C(180), C(183), C(184), C(185), C(187), C(188),
C(191), C(192), C(193), C(195), C(196), C(199), C(200), C(201), C(203), C(204), C(207), C(208), C(209),
C(211), C(212), C(215), C(216), C(217), C(219), C(220), C(223), C(224), C(225), C(227), C(228), C(231),
C(232), C(233), C(235), C(236), C(239), C(240), C(241), C(243), C(244), C(247), C(248), C(249), C(251),
C(252), C(255), C(256), C(257), C(259), C(260), C(263), C(264), C(265), C(267), C(268), C(271), C(272),
C(273), C(275), C(276), C(279), C(280), C(281), C(283), C(284), C(287), C(288), C(289), C(291), C(292),
C(295), C(296), C(297), C(299), C(300), C(303), C(304), C(305), C(307), C(308), C(311), C(312), C(313),
C(315), C(316), C(319), C(320), C(321), C(323), C(324), C(327), C(328), C(329), C(331), C(332), C(335),
C(336), C(337), C(339), C(340), C(343), C(344), C(345), C(347), C(348), C(351), C(352), C(353), C(355),
C(356), C(359), C(360), C(361), C(363), C(364), C(367), C(368), C(369), C(371), C(372), C(375), C(376),
C(377), C(379), C(380), C(383), C(384), C(385), C(387), C(388), C(391), C(392), C(393), C(395), C(396),
C(399), C(400), C(401), C(403), C(404), C(407), C(408), C(409), C(411), C(412), C(415), C(416), C(417),
C(419), C(420), C(423), C(424), C(425), C(427), C(428), C(431), C(432), C(433), C(435), C(436), C(439),
C(440), C(441), C(443), C(444), C(447), C(448), C(449), C(451), C(452), C(455), C(456), C(457), C(459),
C(460), C(463), C(464), C(465), C(467), C(468), C(471), C(472), C(473), C(475), C(476), C(479), C(480),
C(481), C(483), C(484), C(487), C(488), C(489), C(491), C(492), C(495), C(496), C(497), C(499), C(500),
C(503), C(504), C(505), C(507), C(508), C(511), C(512), C(513), C(515), C(516), C(519), C(520), C(521),
C(523), C(524), C(527), C(528), C(529), C(531), C(532), C(535), C(536), C(537), C(539), C(540), C(543),
C(544), C(545), C(547), C(548), C(551), C(552), C(555), C(556), C(559), C(560), C(561), C(563), C(564),
C(567), C(568), C(569), C(571), C(572), C(575), C(576), C(577), C(579), C(580), C(583), C(584), C(585),
C(587), C(588), C(591), C(592), C(595), C(596), C(599), C(600), C(603), C(604), C(607), C(608), C(609),
C(611), C(612), C(615), C(616), C(617), C(619), C(620), C(623), C(624), C(625), C(627), C(628), C(631),
C(632), C(633), C(635), C(636), C(639), C(640), C(643), C(644), C(647), C(648), C(651), C(652), C(655),
C(656), C(657), C(659), C(660), C(663), C(664), C(665), C(667), C(668), C(671), C(672), C(673), C(675),
C(676), C(679), C(680), C(681), C(683), C(684), C(687), C(688), C(691), C(692), C(695), C(696), C(699),
C(700), C(703), C(704), C(705), C(707), C(708), C(711), C(712), C(713), C(715), C(716), C(719), C(720),
C(723), C(727), C(728), C(731), C(732), C(735), C(736), C(739), C(743), C(747), C(751), C(755)
```

And the following 224 coded bits are moved to data block P<sub>B</sub>:

```
C(4), C(8), C(12), C(13), C(17), C(21), C(22), C(25), C(26), C(29), C(30), C(34), C(37), C(38), C(42),
C(45), C(46), C(50), C(53), C(54), C(58), C(61), C(62), C(66), C(69), C(70), C(74), C(77), C(78), C(82),
C(85), C(86), C(90), C(93), C(94), C(98), C(101), C(102), C(106), C(109), C(110), C(114), C(117), C(118),
C(122), C(125), C(126), C(130), C(133), C(134), C(138), C(141), C(142), C(146), C(149), C(150), C(154),
C(157), C(158), C(162), C(165), C(166), C(170), C(173), C(174), C(178), C(181), C(182), C(186), C(189),
C(190), C(194), C(197), C(198), C(202), C(205), C(206), C(210), C(213), C(214), C(218), C(221), C(222),
C(226), C(229), C(230), C(234), C(237), C(238), C(242), C(245), C(246), C(250), C(253), C(254), C(258),
C(261), C(262), C(266), C(269), C(270), C(274), C(277), C(278), C(282), C(285), C(286), C(290), C(293),
C(294), C(298), C(301), C(302), C(306), C(309), C(310), C(314), C(317), C(318), C(322), C(325), C(326),
C(330), C(333), C(334), C(338), C(341), C(342), C(346), C(349), C(350), C(354), C(357), C(358), C(362),
C(365), C(366), C(370), C(373), C(374), C(378), C(381), C(382), C(386), C(389), C(394), C(397), C(402),
C(405), C(413), C(418), C(421), C(429), C(434), C(437), C(445), C(450), C(453), C(461), C(466), C(469),
C(477), C(482), C(485), C(493), C(498), C(501), C(509), C(514), C(517), C(525), C(530), C(533), C(541),
C(549), C(553), C(557), C(562), C(565), C(570), C(573), C(578), C(581), C(589), C(593), C(597), C(601),
C(605), C(610), C(613), C(618), C(621), C(626), C(629), C(637), C(641), C(645), C(649), C(653), C(658),
C(661), C(666), C(669), C(674), C(677), C(685), C(689), C(693), C(697), C(701), C(706), C(709), C(714),
C(717), C(721), C(722), C(724), C(725), C(729), C(733), C(737), C(740), C(741), C(744)
```

The vectors P<sub>G</sub> and P<sub>B</sub> of coded and punctured bits is combined with in band bits to vector P<sub>C</sub> as

$$\begin{split} &P_{C^{'}}(k)=ic(k) & \text{for } k=0,\,1,\,2,\,3 \\ &P_{C^{'}}(k+4)=P_{G}(k) & \text{for } k=0,\,1,\,...,\,223 \\ &P_{C^{'}}(k+224)=ic(k) & \text{for } k=4,\,5,\,6,\,7 \\ &P_{C^{'}}(k+8)=P_{G}(k) & \text{for } k=224,\,225,\,...,\,447 \\ &P_{C^{'}}(k+448)=ic(k) & \text{for } k=8,\,9,\,10,\,11 \\ &P_{C^{'}}(k+460)=P_{B}(k) & \text{for } k=0,\,1,\,...,\,223 \end{split}$$

### O-TCH/WHS6.60:

The block of 138 bits  $\{u(0)...\ u(137)\}$  is encoded with the 1/5 rate convolutional code defined by the following polynomials:

$$\begin{aligned} G4/G7 &= 1 + D^2 + D^3 + D^5 + D^6/1 + D + D^2 + D^3 + D^6 \\ G4/G7 &= 1 + D^2 + D^3 + D^5 + D^6/1 + D + D^2 + D^3 + D^6 \\ G5/G7 &= 1 + D + D^4 + D^6/1 + D + D^2 + D^3 + D^6 \\ G6/G7 &= 1 + D + D^2 + D^3 + D^4 + D^6/1 + D + D^2 + D^3 + D^6 \\ G7/G7 &= 1 \end{aligned}$$

resulting in 720 coded bits,  $\{C(0)...C(719)\}\$  defined by:

$$\begin{split} r(k) &= u(k) + r(k\text{-}1) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}6) \\ C(5k) &= r(k) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}5) + r(k\text{-}6) \\ C(5k+1) &= r(k) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}5) + r(k\text{-}6) \\ C(5k+2) &= r(k) + r(k\text{-}1) + r(k\text{-}4) + r(k\text{-}6) \\ C(5k+3) &= r(k) + r(k\text{-}1) + r(k\text{-}2) + r(k\text{-}3) + r(k\text{-}4) + r(k\text{-}6) \\ C(5k+4) &= u(k) \end{split}$$
 for  $k = 0, 1, ..., 137;$   $r(k) = 0$  for  $k < 0$ 

and (for termination of the coder):

$$r(k) = 0$$

$$\begin{split} &C(5k) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ &C(5k+1) &= r(k) + r(k-2) + r(k-3) + r(k-5) + r(k-6) \\ &C(5k+2) &= r(k) + r(k-1) + r(k-4) + r(k-6) \\ &C(5k+3) &= r(k) + r(k-1) + r(k-2) + r(k-3) + r(k-4) + r(k-6) \\ &C(5k+4) &= r(k-1) + r(k-2) + r(k-3) + r(k-6) & \text{for } k = 138, 139, ..., 143 \end{split}$$

The following 448 coded bits are moved to data block P<sub>G</sub>:

```
C(4), C(9), C(11), C(12), C(14), C(16), C(17), C(19), C(21), C(22), C(24), C(26), C(27), C(29), C(31),
C(32), C(34), C(36), C(37), C(38), C(39), C(41), C(42), C(44), C(46), C(47), C(48), C(49), C(51), C(52),
C(54), C(56), C(57), C(59), C(60), C(61), C(62), C(63), C(64), C(66), C(67), C(69), C(71), C(72), C(73),
C(74), C(76), C(77), C(79), C(81), C(82), C(84), C(85), C(86), C(87), C(88), C(89), C(91), C(92), C(94),
C(96), C(97), C(98), C(99), C(101), C(102), C(104), C(106), C(107), C(109), C(111), C(112), C(113),
C(114), C(116), C(117), C(119), C(121), C(122), C(123), C(124), C(126), C(127), C(129), C(131), C(132),
C(134), C(135), C(136), C(137), C(138), C(139), C(141), C(142), C(144), C(146), C(147), C(148), C(149),
C(151), C(152), C(154), C(156), C(157), C(159), C(161), C(162), C(163), C(164), C(166), C(167), C(169),
C(171), C(172), C(173), C(174), C(176), C(177), C(179), C(181), C(182), C(184), C(186), C(187), C(188),
C(189), C(191), C(192), C(194), C(196), C(197), C(198), C(199), C(201), C(202), C(204), C(206), C(207),
C(209), C(210), C(211), C(212), C(213), C(214), C(216), C(217), C(219), C(221), C(222), C(223), C(224),
C(226), C(227), C(229), C(231), C(232), C(234), C(236), C(237), C(238), C(239), C(241), C(242), C(244),
C(246), C(247), C(248), C(249), C(251), C(252), C(254), C(256), C(257), C(259), C(261), C(262), C(263),
C(264), C(266), C(267), C(269), C(271), C(272), C(273), C(274), C(276), C(277), C(279), C(281), C(282),
C(284), C(285), C(286), C(287), C(288), C(289), C(291), C(292), C(294), C(296), C(297), C(298), C(299),
C(301), C(302), C(304), C(306), C(307), C(309), C(311), C(312), C(313), C(314), C(316), C(317), C(319),
C(321), C(322), C(323), C(324), C(326), C(327), C(329), C(331), C(332), C(334), C(336), C(337), C(338),
C(339), C(341), C(342), C(344), C(346), C(347), C(349), C(351), C(352), C(354), C(356), C(357), C(359),
C(361), C(362), C(364), C(366), C(367), C(369), C(371), C(372), C(374), C(376), C(377), C(379), C(381),
C(382), C(384), C(386), C(387), C(389), C(391), C(392), C(394), C(396), C(397), C(399), C(401), C(402),
C(404), C(406), C(407), C(409), C(411), C(412), C(414), C(416), C(417), C(419), C(421), C(422), C(424),
C(426), C(427), C(429), C(431), C(432), C(434), C(436), C(437), C(439), C(441), C(442), C(444), C(446),
C(447), C(449), C(451), C(452), C(454), C(456), C(457), C(459), C(461), C(462), C(464), C(466), C(467), C(467), C(468), C(46
C(469), C(471), C(472), C(474), C(476), C(477), C(479), C(481), C(482), C(484), C(486), C(487), C(489),
C(491), C(492), C(494), C(496), C(499), C(501), C(502), C(504), C(506), C(507), C(509), C(511), C(512),
C(514), C(516), C(517), C(519), C(521), C(524), C(526), C(527), C(529), C(531), C(532), C(534), C(536),
C(537), C(539), C(541), C(542), C(544), C(546), C(549), C(551), C(552), C(554), C(556), C(557), C(559),
C(561), C(562), C(564), C(566), C(567), C(569), C(571), C(574), C(576), C(577), C(579), C(581), C(582),
C(584), C(586), C(587), C(589), C(591), C(592), C(594), C(596), C(599), C(601), C(602), C(604), C(606),
C(607), C(609), C(611), C(612), C(614), C(616), C(617), C(619), C(621), C(624), C(626), C(627), C(629),
C(631), C(632), C(634), C(636), C(637), C(639), C(641), C(642), C(644), C(646), C(649), C(651), C(652),
C(654), C(656), C(657), C(659), C(661), C(662), C(664), C(666), C(667), C(669), C(671), C(674), C(676),
C(677), C(679), C(681), C(682), C(684), C(686), C(687), C(689), C(691), C(692), C(694), C(696), C(699),
C(701), C(702), C(704), C(706), C(707), C(709), C(711), C(712), C(714), C(716), C(719)
```

And the following 224 coded bits are moved to data block P<sub>B</sub>:

```
C(18), C(23), C(25), C(28), C(30), C(33), C(35), C(40), C(43), C(45), C(50), C(53), C(55), C(58), C(65), C(68), C(70), C(75), C(78), C(80), C(83), C(90), C(93), C(95), C(100), C(103), C(105), C(108), C(110), C(115), C(118), C(120), C(125), C(128), C(130), C(133), C(140), C(143), C(145), C(150), C(153), C(155), C(158), C(160), C(165), C(168), C(170), C(175), C(178), C(180), C(183), C(185), C(190), C(193), C(195), C(200), C(203), C(205), C(208), C(215), C(218), C(220), C(225), C(228), C(230), C(233), C(235), C(240), C(243), C(245), C(250), C(253), C(255), C(258), C(260), C(265), C(268), C(270), C(275), C(278), C(280), C(283), C(290), C(293), C(295), C(300), C(303), C(305), C(308), C(310), C(315), C(318), C(320), C(325), C(328), C(330), C(333), C(335), C(340), C(343), C(345), C(348), C(350), C(353), C(355), C(358), C(360), C(363), C(365), C(368), C(370), C(373), C(375), C(378), C(380), C(383), C(385), C(388), C(390), C(393), C(395), C(398), C(400), C(403), C(405), C(408), C(410), C(413), C(415), C(418), C(420), C(423), C(425), C(428), C(430), C(433), C(435), C(438), C(440), C(443), C(445), C(448), C(445), C(448), C(450), C(453), C(488), C(490), C(493), C(495), C(497), C(498), C(503), C(505), C(508), C(510), C(513), C(515), C(518), C(522), C(523), C(523), C(523), C(552), C(523), C(552), C(523), C(552), C(523), C(552), C(523), C(552), C(508), C(508), C(510), C(513), C(515), C(518), C(522), C(523), C(523), C(523), C(523), C(523), C(503), C(505), C(508), C(510), C(513), C(515), C(518), C(522), C(523), C(503), C(505), C(508), C(510), C(513), C(515), C(518), C(522), C(523), C(523)
```

C(528), C(530), C(533), C(538), C(540), C(543), C(547), C(548), C(553), C(555), C(558), C(560), C(563), C(565), C(568), C(572), C(573), C(578), C(580), C(583), C(588), C(590), C(593), C(597), C(598), C(603), C(605), C(608), C(610), C(613), C(615), C(618), C(622), C(623), C(628), C(630), C(633), C(638), C(640), C(643), C(647), C(648), C(653), C(658), C(660), C(663), C(668), C(672), C(673), C(678), C(683), C(697)

The vectors P<sub>G</sub> and P<sub>B</sub> of coded and punctured bits is combined with in band bits to vector P<sub>C</sub> as

$P_{C}'(k) = ic(k)$	for $k = 0, 1, 2, 3$
$P_{C}'(k+4) = P_{G}(k)$	for $k = 0, 1,, 223$
$P_{C}'(k+224) = ic(k)$	for $k = 4, 5, 6, 7$
$P_{C}'(k+8) = P_{G}(k)$	for k = 224, 225,, 447
$P_{C}'(k+448) = ic(k)$	for $k = 8, 9, 10, 11$
$P_{C}'(k+460) = P_{B}(k)$	for $k = 0, 1,, 223$

# 3.17.7.5 Interleaving

The interleaving is done as specified for the O-TCH/AHS (subclause 3.15.7.5).

# 3.17.7.6 Mapping on a Burst

The mapping is done as specified for the O-TCH/AHS (subclause 3.15.7.6).

# 3.17.8 RATSCCH MARKER

This frame type contains the in-band channel and an identification marker. The in-band data id(0,1) represents Mode Indication or Mode Command/Mode Request depending on the current frame number.

## 3.17.8.1 Coding of in-band data

The coding of in-band data is done as specified for the RATSCCH\_MARKER frame in O-TCH/AHS (subclause 3.15.8.1).

# 3.17.8.2 Identification marker

The identification marker is done as specified for the RATSCCH\_MARKER frame in O-TCH/AHS (subclause 3.15.8.2).

#### 3.17.8.3 Interleaving

The interleaving is done as specified for the RATSCCH MARKER frame in O-TCH/AHS (subclause 3.15.8.3).

# 3.17.8.4 Mapping on a Burst

The mapping is done as specified for the RATSCCH\_MARKER frame in O-TCH/AHS (subclause 3.15.8.4).

## 3.17.9 RATSCCH DATA

This frame contains the RATSCCH data and an inband channel. The RATSCCH data consists of 35 bits. The in-band data id(0,1) represents Mode Indication or Mode Command/Mode Request depending on the current frame number.

# 3.17.9.1 Coding of in-band data

The coding of in-band data is done as specified for the RATSCCH\_DATA frame in O-TCH/AHS (subclause 3.15.9.1).

# 3.17.9.2 Parity and convolutional encoding for the RATSCCH message

The parity and convolutional encoding for the RATSCCH message are done as specified for the RATSCCH\_DATA frame in O-TCH/AHS (subclause 3.15.9.2).

# 3.17.9.3 Interleaving

The interleaving is done as specified for the RATSCCH DATA frame in O-TCH/AHS (subclause 3.15.9.3).

## 3.17.9.4 Mapping on a Burst

The mapping is done as specified for the RATSCCH\_DATA frame in O-TCH/AHS (subclause 3.15.9.4).

# 4 Control Channels

# 4.1 Slow associated control channel (SACCH)

# 4.1.1 Block constitution

The message delivered to the encoder has a fixed size of 184 information bits  $\{d(0),d(1),...,d(183)\}$ . It is delivered on a burst mode.

# 4.1.2 Block code

a) Parity bits:

The block of 184 information bits is protected by 40 extra bits used for error correction and detection. These bits are added to the 184 bits according to a shortened binary cyclic code (FIRE code) using the generator polynomial:

$$g(D) = (D^{23} + 1)*(D^{17} + D^3 + 1)$$

The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

$$d(0)D^{223} + d(1)D^{222} + ... + d(183)D^{40} + p(0)D^{39} + p(1)D^{38} + ... + p(38)D + p(39)$$

where  $\{p(0), p(1), ..., p(39)\}\$  are the parity bits, when divided by g(D) yields a remainder equal to:

$$1 + D + D^2 + ... + D^{39}$$
.

b) Tail bits

Four tail bits equal to 0 are added to the information and parity bits, the result being a block of 228 bits.

$$u(k) = d(k)$$
 for  $k = 0,1,...,183$  
$$u(k) = p(k-184)$$
 for  $k = 184,185,...,223$  
$$u(k) = 0$$
 for  $k = 224,225,226,227$  (tail bits)

# 4.1.3 Convolutional encoder

This block of 228 bits is encoded with the ½ rate convolutional code (identical to the one used for TCH/FS) defined by the polynomials:

$$G0 = 1 + D^3 + D^4$$
  
 $G1 = 1 + D + D^3 + D^4$ 

This results in a block of 456 coded bits:  $\{c(0),c(1),...,c(455)\}\$  defined by:

$$\begin{split} c(2k) &= u(k) + u(k-3) + u(k-4) \\ c(2k+1) &= u(k) + u(k-1) + u(k-3) + u(k-4) & \text{for } k = 0,1,...,227 \ ; \ u(k) = 0 \ \text{for } k < 0 \end{split}$$

# 4.1.4 Interleaving

The coded bits are reordered and interleaved according to the following rule:

$$i(B,j) = c(n,k) \text{ for } k = 0,1,...,455$$
 
$$n = 0,1,...,N,N+1,...$$
 
$$B = B_0 + 4n + (k \text{ mod } 4)$$
 
$$j = 2((49k) \text{ mod } 57) + ((k \text{ mod } 8) \text{ div } 4)$$

See table 1. The result of the reordering of bits is the same as given for a TCH/FS (subclause 3.1.3) as can be seen from the evaluation of the bit number-index j, distributing the 456 bits over 4 blocks on even numbered bits and 4 blocks on odd numbered bits. The resulting 4 blocks are built by putting blocks with even numbered bits and blocks with odd numbered bits together into one block.

The block of coded data is interleaved "block rectangular" where a new data block starts every 4<sup>th</sup> block and is distributed over 4 blocks.

# 4.1.5 Mapping on a Burst

The mapping is given by the rule:

$$e(B,j) = i(B,j)$$
 and  $e(B,59+j) = i(B,57+j)$  for  $j = 0,1,...,56$ 

and

$$e(B,57) = hl(B)$$
 and  $e(B,58) = hu(B)$ 

The two bits labelled hl(B) and hu(B) on burst number B are flags used for indication of control channel signalling. They are set to "1" for a SACCH.

# 4.2 Fast associated control channel at full rate (FACCH/F)

# 4.2.1 Block constitution

The message delivered to the encoder has a fixed size of 184 information bits. It is delivered on a burst mode.

## 4.2.2 Block code

The block encoding is done as specified for the SACCH in subclause 4.1.2.

## 4.2.3 Convolutional encoder

The convolutional encoding is done as specified for the SACCH in subclause 4.1.3.

# 4.2.4 Interleaving

The interleaving is done as specified for the TCH/FS in subclause 3.1.3.

# 4.2.5 Mapping on a Burst

A FACCH/F frame of 456 coded bits is mapped on 8 consecutive bursts as specified for the TCH/FS in subclause 3.1.4. As a FACCH is transmitted on bits which are stolen in a burst from the traffic channel, the even numbered bits in the first 4 bursts and the odd numbered bits of the last 4 bursts are stolen.

To indicate this to the receiving device the flags hl(B) and hu(B) have to be set according to the following rule:

```
hu(B) = 1 for the first 4 bursts (even numbered bits are stolen);
```

hl(B) = 1 for the last 4 bursts (odd numbered bits are stolen).

The consequences of this bitstealing by a FACCH/F is for a:

- speech channel (TCH/FS) and data channel (TCH/F2.4):

One full frame of data is stolen by the FACCH.

- Data channel (TCH/F14.4):

The bitstealing by a FACCH/F disturbs a maximum of 96 of the 456 coded bits generated from an input data block of 290 bits.

- Data channel (TCH/F9.6):

The bitstealing by a FACCH/F disturbs a maximum of 96 coded bits generated from an input frame of four data blocks. A maximum of 24 of the 114 coded bits resulting from one input data block of 60 bits may be disturbed.

- Data channel (TCH/F4.8):

The bit stealing by FACCH/F disturbs a maximum of 96 coded bits generated from an input frame of two data blocks. A maximum of 48 of the 228 coded bits resulting from one input data block of 60 bits may be disturbed.

NOTE: In the case of consecutive stolen frames, a number of bursts will have both the even and the odd bits stolen and both flags hu(B) and hl(B) must be set to 1.

# 4.3 Fast associated control channel at half rate (FACCH/H)

# 4.3.1 Block constitution

The message delivered to the encoder has a fixed size of 184 information bits. It is delivered on a burst mode.

#### 4.3.2 Block code

The block encoding is done as specified for the SACCH in subclause 4.1.2.

## 4.3.3 Convolutional encoder

The convolutional encoding is done as specified for the SACCH in subclause 4.1.3.

# 4.3.4 Interleaving

The coded bits are reordered and interleaved according to the following rule:

$$i(B,j) = c(n,k) \, for \, k = 0,1,...,455$$
 
$$n = 0,1,...,N,N+1,...$$
 
$$B = B_0 + 4n + (k \, mod \, 8) - 4((k \, mod \, 8) \, div \, 6)$$

$$i = 2((49k) \mod 57) + ((k \mod 8) \operatorname{div} 4)$$

See table 1. The result of the reordering of bits is the same as given for a TCH/FS (subclause 3.1.3) as can be seen from the evaluation of the bit number-index j, distributing the 456 bits over 4 blocks on even numbered bits and 4 blocks on odd numbered bits. The 2 last blocks with even numbered bits and the 2 last blocks with odd numbered bits are put together into 2 full middle blocks.

The block of coded data is interleaved "block diagonal" where a new data block starts every 4<sup>th</sup> block and is distributed over 6 blocks.

# 4.3.5 Mapping on a Burst

A FACCH/H frame of 456 coded bits is mapped on 6 consecutive bursts by the rule:

```
e(B,j) = i(B,j) \qquad \text{and} \qquad e(B,59+j) = i(B,57+j) \qquad \text{for } j=0,1,...,56 and e(B,57) = hl(B) \quad \text{and} \qquad e(B,58) = hu(B)
```

As a FACCH/H is transmitted on bits which are stolen from the traffic channel, the even numbered bits of the first 2 bursts, all bits of the middle 2 bursts and the odd numbered bits of the last 2 bursts are stolen.

To indicate this to the receiving device the flags hl(B) and hu(B) have to be set according to the following rule:

```
hu(B) = 1 for the first 2 bursts (even numbered bits are stolen)

hu(B) = 1 and hl(B) = 1 for the middle 2 bursts (all bits are stolen)

hl(B) = 1 for the last 2 bursts (odd numbered bits are stolen)
```

The consequences of this bitstealing by a FACCH/H is for a:

- speech channel (TCH/HS):

two full consecutive speech frames are stolen by a FACCH/H.

data channel (TCH/H4.8):

The bitstealing by FACCH/H disturbs a maximum of 96 coded bits generated from an input frame of four data blocks. A maximum of 24 out of the 114 coded bits resulting from one input data block of 60 bits may be disturbed.

data channel (TCH/H2.4):

The bitstealing by FACCH/H disturbs a maximum of 96 coded bits generated from an input frame of four data blocks. A maximum of 24 out of the 114 coded bits resulting from one input data block of 36 bits may be disturbed.

NOTE: In the case of consecutive stolen frames, two overlapping bursts will have both the even and the odd numbered bits stolen and both flags hu(B) and hl(B) must be set to 1.

# 4.4 Broadcast control, Paging, Access grant, Notification and Cell broadcast channels (BCCH, PCH, AGCH, NCH, CBCH), CTS Paging and Access grant channels (CTSPCH, CTSAGCH)

The coding scheme used for the broadcast control, paging, access grant, notification and cell broadcast messages is the same as for the SACCH messages, specified in subclause 4.1. In CTS, the coding scheme used for the paging and access grant messages is also the same as for the SACCH messages, specified in subclause 4.1.

# 4.5 Stand-alone dedicated control channel (SDCCH)

The coding scheme used for the dedicated control channel messages is the same as for SACCH messages, specified in subclause 4.1.

# 4.6 Random access channel (RACH)

Three coding schemes are specified for the burst carrying the random access uplink message:

the access burst containing 8 information bits,

the access burst containing 11 information bits and

the access burst containing 30 information bits.

The encoding of the access burst containing 8 information bits is as defined in subclause 4.6.1.

The encoding of the access burst containing 11 information bits is as defined in sub-clause 5.3.2 for the packet random access channel (PRACH, CPRACH and MPRACH).

The encoding of the access burst containing 30 information bits is as defined in sub-clause 4.6.2. It is used for the Multilateration Timing Advance procedure when the Extended Access Burst method is used (see 3GPP TS 43.059).

# 4.6.1 RACH carrying 8 information bits

The encoding of the access burst containing 8 information bits is defined as follows. It contains 8 information bits d(0),d(1),...,d(7).

Six parity bits p(0),p(1),...,p(5) are defined in such a way that in GF(2) the binary polynomial:

$$d(0)D^{13} + ... + d(7)D^6 + p(0)D^5 + ... + p(5)$$
, when divided by  $D^6 + D^5 + D^3 + D^2 + D + 1$  yields a remainder equal to  $D^5 + D^4 + D^3 + D^2 + D + 1$ .

The six bits of the BSIC,  $\{B(0),B(1),...,B(5)\}$ , of the BS to which the Random Access is intended, are added bitwise modulo 2 to the six parity bits,  $\{p(0),p(1),...,p(5)\}$ . This results in six colour bits, C(0) to C(5) defined as C(k) = b(k) + p(k) (k = 0 to 5) where:

b(0) = MSB of PLMN colour code

b(5) = LSB of BS colour code.

This defines  $\{u(0), u(1), ..., u(17)\}$  by:

$$u(k) = d(k)$$
 for  $k = 0,1,...,7$ 

$$u(k) = C(k-8)$$
 for  $k = 8,9,...,13$ 

$$u(k) = 0$$
 for  $k = 14,15,16,17$  (tail bits)

The bits  $\{e(0),e(1),...,e(35)\}$  are obtained by the same convolutional code of rate  $\frac{1}{2}$  as for TCH/FS, defined by the polynomials:

$$G0 = 1 + D^3 + D^4$$

$$G1 = 1 + D + D^3 + D^4$$

and with:

$$e(2k) = u(k) + u(k-3) + u(k-4)$$

$$e(2k+1) = u(k) + u(k-1) + u(k-3) + u(k-4)$$
 for  $k = 0,1,...,17$ ;  $u(k) = 0$  for  $k < 0$ 

# 4.6.2 (EC-)RACH carrying 30 information bits

In case of the Multilateration Timing Advance procedure when the Extended Access Burst method is used, indicated by the network, the mobile station first sends a random access message carrying 11 information bits, using the encoding for the access burst containing 11 bits, as specified in sub-clause 5.3.2, and after access grant by the network sends a subsequent random access message carrying 30 information bits as soon as possible after receiving the access grant while still using the principles for (EC-)RACH transmission opportunity selection described in 3GPP TS 44.018. The encoding of the 30 information bits is as given in the following sub-clauses.

#### 4.6.2.1 Block constitution

The burst carrying the extended random access uplink message for connectionless Multilateration Positioning contains 30 information bits d(0),d(1),...,d(29). The first 11 bits d(0),...,d(10) are encoded in the same way as specified for the access burst containing 11 information bits in sub-clause 5.3.2 resulting in a block of 36 coded bits  $\{e(0), e(1),...,e(35)\}$ . The encoding of the remaining 19 information bits d(11),...,d(29) is specified in the following sub-clauses.

#### 4.6.2.2 Parity bits

Six parity bits p(0),p(1),...,p(5) are defined in such a way that in GF(2) the binary polynomial:

$$d(11)D^{24} + ... + d(29)D^6 + p(0)D^5 + ... + p(5)$$
, when divided by  $D^6 + D^5 + D^3 + D^2 + D + 1$  yields a remainder equal to  $D^5 + D^4 + D^3 + D^2 + D + 1$ .

## 4.6.2.3 Tail bits

Four tail bits are added to information bits and parity bits defining the bit sequence  $\{u(0), u(1), ..., u(28)\}$  as follows:

$$u(k-11) = d(k)$$
 for  $k = 11,12,...,29$   $u(k+19) = p(k)$  for  $k = 0,...,5$   $u(k+25) = 0$  for  $k = 0,1,2,3$  (tail bits)

#### 4.6.2.4 Convolutional code

The coded bits  $\{c(0),c(1),...,c(57)\}$  are obtained by the same convolutional code of rate  $\frac{1}{2}$  as for TCH/FS, defined by the polynomials:

$$G0 = 1 + D^3 + D^4$$
  
 $G1 = 1 + D + D^3 + D^4$ 

and with:

$$\begin{split} c(2k) &= u(k) + u(k-3) + u(k-4) \\ c(2k+1) &= u(k) + u(k-1) + u(k-3) + u(k-4) \qquad \text{ for } k = 0,1,...,28 \ ; \ u(k) = 0 \ \text{for } k < 0 \end{split}$$

The code is punctured in such a way that the following coded bit:

c(57) is not transmitted.

This results in a block of 57 coded bits, {e(36),...,e(92)} with

$$e(k+36) = c(k)$$
 for  $k = 0,1,...,56$ 

# 4.6.2.5 Burst mapping

Both data blocks  $\{e(0),...,e(35)\}$  and  $\{e(36),...,e(92)\}$  are mapped onto the Extended Access burst as specified in 3GPP TS 45.002.

# 4.7 Synchronization channel (SCH), Compact synchronization channel (CSCH), CTS Beacon and Access request channels (CTSBCH-SB, CTSARCH)

The burst carrying the synchronization information on the downlink BCCH, the downlink CPBCCH for Compact, and in CTS the information of the CTSBCH-SB and the access request message of the CTSARCH, has a different structure. It contains 25 information bits  $\{d(0),d(1),...,d(24)\}$ , 10 parity bits  $\{p(0),p(1),...,p(9)\}$  and 4 tail bits. The precise ordering of the information bits is given in 3GPP TS 44.018.

The ten parity bits  $\{p(0),p(1),...,p(9)\}$  are defined in such a way that in GF(2) the binary polynomial:

$$d(0)D^{34} + ... + d(24)D^{10} + p(0)D^9 + ... + p(9), \text{ when divided by:}$$
 
$$D^{10} + D^8 + D^6 + D^5 + D^4 + D^2 + 1, \text{ yields a remainder equal to:}$$
 
$$D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1.$$

Thus the encoded bits  $\{u(0), u(1), \dots, u(38)\}$  are:

$$\begin{array}{ll} u(k) &= d(k) & \text{ for } k = 0,1,...,24 \\ \\ u(k) &= p(k\text{-}25) & \text{ for } k = 25,26,...,34 \\ \\ u(k) &= 0 & \text{ for } k = 35,36,37,38 \text{ (tail bits)} \end{array}$$

The bits  $\{e(0),e(1),...,e(77)\}$  are obtained by the same convolutional code of rate ½ as for TCH/FS, defined by the polynomials:

$$G0 = 1 + D^3 + D^4$$
  
 $G1 = 1 + D + D^3 + D^4$ 

and with:

$$\begin{array}{ll} e(2k) & = u(k) + u(k-3) + u(k-4) \\ \\ e(2k+1) & = u(k) + u(k-1) + u(k-3) + u(k-4) \end{array} \qquad \text{for } k = 0,1,....,38 \ ; \ u(k) = 0 \ \text{for } k < 0 \end{array}$$

# 4.7a Extended Coverage Synchronization channel (EC-SCH)

# 4.7a.1 Block constitution

The burst carrying the synchronization information on the downlink EC-SCH contains 30 information bits  $\{d(0),d(1),...,d(29)\}$ .

# 4.7a.2 Coding

The 30 information bits  $\{d(0),d(1),...,d(29)\}$  are encoded as follows:

a) Parity bits:

Ten parity bits  $\{p(0),p(1),...,p(9)\}$  are defined in such a way that in GF(2) the binary polynomial:

$$d(0)D^{39}$$
 +...+  $d(29)D^{10}$  +  $p(0)D^9$  +...+  $p(9)$ , when divided by:  
 $D^{10}$  +  $D^8$  +  $D^6$  +  $D^5$  +  $D^4$  +  $D^2$  + 1, yields a remainder equal to:  
 $D^9$  +  $D^8$  +  $D^7$  +  $D^6$  +  $D^5$  +  $D^4$  +  $D^3$  +  $D^2$  +  $D$  + 1.

b) Tail bits:

Four tail bits equal to 0 are added to the information and parity bits, the results being a block of 44 bits  $\{u(0),u(1),...,u(43)\}$ :

$$\begin{array}{ll} u(k) &= d(k) & \text{ for } k = 0,1,...,29 \\ \\ u(k) &= p(k\text{--}30) & \text{ for } k = 30,31,...,39 \\ \\ u(k) &= 0 & \text{ for } k = 40,41,42,43 \text{ (tail bits)} \end{array}$$

#### c) Convolutional encoder:

This block of 44 bits  $\{u(0),u(1),...,u(43)\}$  is encoded with the same convolutional code of rate  $\frac{1}{2}$  as for TCH/FS, defined by the polynomials:

$$G0 = 1 + D^3 + D^4$$
  
 $G1 = 1 + D + D^3 + D^4$ 

This results in a block of 88 coded bits  $\{C(0),C(1),...,C(87)\}$  defined by:

$$C(2k) = u(k) + u(k-3) + u(k-4)$$
  
 $C(2k+1) = u(k) + u(k-1) + u(k-3) + u(k-4)$  for  $k = 0,1,...,43$ ;  $u(k) = 0$  for  $k < 0$ 

The block is punctured such that the bits C(k), k=[0,10,19,29,39,48,58,68,77,87] are not transmitted, resulting in a block of 78 encoded bits  $\{e(0),e(1),...,e(77)\}$ .

# 4.7a.3 Blind physical layer transmission

The encoded bits e(j), j=0,...,77 are transmitted 28 times, generating the repeated bursts R(m,B,j)=e(j), m=0,...,27, B=0, j=0,...,77.

# 4.7a.4 Cyclic shift

The repeated bursts are cyclically shifted as:

$$R'(m,B,j) = R(m,B,(j + T2'') \mod 78)$$

where T2'' = (FN div 51) mod 4, i.e. it refers to the specific 51-multiframe FN, in which the burst will be sent in the set of four contiguous 51-multiframes as defined in TS 45.002 [8].

# 4.7a.5 Mapping onto a physical channel

Burst R'(m,B,j), is mapped onto burst B' of the timeslot carrying the EC-SCH, where

$$B' = m$$

NOTE: The burst number B' denotes the relative transmission order of the bursts on the timeslot. The mapping to absolute TDMA frame number is specified in 3GPP TS 45.002 [8].

# 4.8 Access Burst on circuit switched channels other than RACH

The encoding of this burst is as defined in subclause 4.6 for the 8 bits access burst on the random access channel (RACH). The BSIC used shall be the BSIC of the BTS to which the burst is intended.

# 4.9 Access Bursts for uplink access on a channel used for VGCS or VBS

The encoding of this burst is as defined in subclause 4.6 for the 8 bits access burst on the RACH. The BSIC used by the Mobile Station shall be the BSIC indicated by network signalling, or if not thus provided, the last received BSIC on the SCH of the current cell.

# 4.10a Fast associated control channel at ECSD E-TCH/F (E-FACCH/F)

## 4.10a.1 Block constitution

The message delivered to the encoder has a fixed size of 184 information bits. It is delivered on a burst mode.

#### 4.10a.2 Block code

The block encoding is done as specified for the SACCH in subclause 4.1.2.

### 4.10a.3 Convolutional encoder

The convolutional encoding is done as specified for the SACCH in subclause 4.1.3.

# 4.10a.4 Interleaving

The interleaving is done as specified for the SACCH in subclause 4.1.4.

# 4.10a.5 Mapping on a Burst

A E-FACCH/F frame of 456 coded bits is mapped on 4 full consecutive bursts. As a E-FACCH/F is transmitted on bits, which are stolen in a burst from the ECSD traffic channel, the four full bursts are stolen.

The mapping on is given by the rule:

$$e(B,j)=i(B,j)$$
 and  $e(B,59+j)=i(B,57+j)$  for  $j=0,1,...,56$ 

and

$$e(B,57)=hl(B)$$
 and  $e(B,58)=hu(B)$ .

To indicate to the receiving device the flags hl(B) and hu(B) have to be set according to the following rule:

hu(B)=1 and hl(B)=1 for the all 4 bursts (4 full bursts are stolen).

The consequences of this bitstealing by a E-FACCH/F is for a:

- Data channel (E-TCH/F43.2)

The bitstealing by a E-FACCH/F disturbs a maximum of 288 of the 1368 coded bits generated from an input data block of 870 bits.

- Data channel (E-TCH/F32.0)

The bitstealing by a E-FACCH/F disturbs 464 of the 1392 coded bits generated from an input data block of 640 bits.

- Data channel (E-TCH/F28.8)

The bitstealing by a E-FACCH/F disturbs a maximum of 288 of the 1368 coded bits generated from an input data block of 580 bits.

# 4.10b Octal fast associated control channel at half rate (O-FACCH/H)

# 4.10b.1 Block constitution

The message delivered to the encoder has a fixed size of 184 information bits. It is delivered on a burst mode.

## 4.10b.2 Block code

#### a) Parity bits:

The block of 184 information bits is protected by 40 extra bits used for error correction and detection. These bits are added to the 184 bits according to a shortened binary cyclic code (FIRE code) using the generator polynomial:

$$G(D)=(D^{23}+1)(D^{17}+D^3+1)$$

The encoding of the cyclic code is performed in a systematic form, which means that, in GF(2), the polynomial:

$$D(0)D^{223} + d(1)D^{222} + \ldots + d(183)D^{40} + p(0)D^{39} + p(1)D^{38} + \ldots + p(38)D + p(39)$$

where  $\{p(0),p(1),...,p(39)\}$  are the parity bits, when divided by g(D) yields a remainder equal to:

$$1 + D + D^2 + \dots + D^{39}$$
.

#### b) Tail bits

Six tail bits equal to zero are added to the information and parity bits, the result being a block of 230 bits.

$$\begin{array}{ll} u(k)=d(k) & \text{for } k=0,1,...,183 \\ \\ u(k)=p(k\text{-}184) & \text{for } k=184,185,...,223 \\ \\ u(k)=0 & \text{for } k=224,225,226,227,228,229 \text{ (tail bits)} \end{array}$$

## 4.10b.3 Convolutional encoder

This block of 230 bits is encoded with the rate 1/6 convolutional code defined by the polynomials:

$$G4 = 1 + D^{2} + D^{3} + D^{5} + D^{6}$$

$$G4 = 1 + D^{2} + D^{3} + D^{5} + D^{6}$$

$$G5 = 1 + D + D^{4} + D^{6}$$

$$G5 = 1 + D + D^{4} + D^{6}$$

$$G6 = 1 + D + D^{2} + D^{3} + D^{4} + D^{6}$$

$$G7 = 1 + D + D^{2} + D^{3} + D^{6}$$

This results in a block of 1380 encoded bits  $\{C(0),C(1),...C(1379)\}\$  defined by

$$\begin{split} &C(6k) = u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6) \\ &C(6k+1) = u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6) \\ &C(6k+2) = u(k) + u(k-1) + u(k-4) + u(k-6) \\ &C(6k+3) = u(k) + u(k-1) + u(k-4) + u(k-6) \\ &C(6k+4) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-4) + u(k-6) \\ &C(6k+5) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-6) & \text{for } k = 0,1,...,229 \ ; \ u(k) = 0 \ \text{for } k < 0 \end{split}$$

The code is punctured in such a way that the following coded bits:

$$\{C(21+114k) \text{ for } k=0,1,...,11\}$$
 are not transmitted.

The result is a block of 1368 coded bits  $\{c(0),c(1),...,c(1367)\}$ .

# 4.10b.4 Reordering

The coded bits are reordered according to the following rule:

$$r(j) = c(k)$$
, for  $k = 0,1,...,1367$   
 $j = k \text{ div } 36 + 38*(k \text{ mod } 36)$ 

NOTE: The reordering is a simple block interleaver: a 38 rows x 36 columns matrix which is filled in by row and read out by column.

# 4.10b.5 Interleaving

Before interleaving the reordered coded bits  $\{r(0),r(1),...,r(1367)\}$  are converted into 3-bit symbols  $\{Rs(0),Rs(1),...,Rs(455)\}$  according to Table 1 in 3GPP TS 45.004, the symbol Rs(k) depends on r(3k+2), r(3k+1), and r(3k) for k=0,1,...,455. The interleaving is done as specified for the FACCH at half rate in subclause 4.3.4. The difference is that the interleaving is done by symbols instead of single bits, reusing the existing interleaving tables.

# 4.10b.6 Mapping on a burst

As an O-FACCH is transmitted on symbols which are stolen in a burst from the traffic channel, the even numbered symbols in the first 2 bursts, all symbols in the middle 2 bursts, and the odd numbered symbols in the last 2 bursts are stolen.

The mapping is given by the rule:

$$E(B,j) = I(B,j)$$
 and  $E(B,59+j) = I(B,57+j)$  for  $j=0,1,...,56$ 

and

$$E(B,57) = HL(B)$$
 and  $E(B,58) = HU(B)$ .

To indicate the stealing to the receiving device the symbols HL(B) and HU(B) have to be set according to the following rule:

 $HU(B) = \{1,1,1\}$  for the first two bursts (even numbered symbols are stolen)

 $HU(B) = \{1,1,1\}$  and  $HL(B) = \{1,1,1\}$  for the middle two bursts (all symbols are stolen)

 $HL(B) = \{1,1,1\}$  for the last two burts (odd numbered symbols are stolen).

As a consequence, two full consecutive speech frames of an O-TCH/AHS are stolen by an O-FACCH/H.

# 4.10c Octal fast associated control channel at full rate (O-FACCH/F)

# 4.10c.1 Block constitution

The message delivered to the encoder has a fixed size of 184 information bits. It is delivered on a burst mode.

#### 4.10c.2 Block code

The block encoding is done as specified for the O-FACCH/H in subclause 4.10b.2

## 4.10c.3 Convolutional encoder

The convolutional encoding is done as specified for the O-FACCH/H in subclause 4.10b.3.

# 4.10c.4 Reordering

The reordering is done as specified for the O-FACCH/H in subclause 4.10b.4.

# 4.10c.5 Interleaving

Before interleaving the reordered coded bits  $\{r(0),r(1),...,r(1367)\}$  are converted into 3-bit symbols  $\{Rs(0),Rs(1),...,Rs(455)\}$  according to Table 1 in 3GPP TS 45.004, the symbol Rs(k) depends on r(3k+2), r(3k+1), and r(3k) for k=0,1,...,455. The interleaving is done as specified for the FACCH at full rate in subclause 4.2.4. The difference is that the interleaving is done by symbols instead of single bits, reusing the existing interleaving tables.

# 4.10c.6 Mapping on a burst

As an O-FACCH is transmitted on symbols which are stolen in a burst from the traffic channel, the even numbered symbols in the first four bursts and the odd numbered symbols in the last four bursts are stolen.

The mapping is given by the rule:

$$E(B,j) = I(B,j)$$
 and  $E(B,59+j) = I(B,57+j)$  for  $j=0,1,...,56$ 

and

$$E(B,57) = HL(B)$$
 and  $E(B,58) = HU(B)$ .

To indicate the stealing to the receiving device the symbols HL(B) and HU(B) have to be set according to the following rule:

 $HU(B) = \{1,1,1\}$  for the first four bursts (even numbered symbols are stolen)

 $HL(B) = \{1,1,1\}$  for the last four burts (odd numbered symbols are stolen).

As a consequence, one speech frame of an O-TCH/F is stolen by an O-FACCH/F.

# 4.11 Slow associated control channel with embedded enhanced power control (SACCH/TP)

# 4.11.1 Block constitution

The message delivered to the encoder has a fixed size of 184 information bits  $\{d(0),d(1),...,d(183)\}$ . It is delivered on a burst mode.

## 4.11.2 Block code

a) Parity bits:

Eighteen parity bits p(0),p(1),...,p(17) are defined in such a way that in GF(2) the binary polynomial:

$$d(0)D^{201} + ... + d(183)D^{18} + p(0)D^{17} + ... + p(17)$$
, when divided by:

$$D^{18} + D^{17} + D^{14} + D^{13} + D^{11} + D^{10} + D^{8} + D^{7} + D^{6} + D^{3} + D^{2} + 1$$
, yields a remainder equal to:

$$D^{17} + D^{16} + D^{15} + D^{14} + D^{13} + D^{12} + D^{11} + D^{10} + D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D^4$$

b) Tail bits

Six tail bits equal to 0 are added to the information and parity bits, the result being a block of 208 bits.

$$\begin{array}{ll} u(k)=d(k) & \text{for } k=0,1,...,183 \\ \\ u(k)=p(k\text{-}184) & \text{for } k=184,185,...,201 \\ \\ u(k)=0 & \text{for } k=202,203,204,205,206,207 \text{ (tail bits)} \end{array}$$

## 4.11.3 Convolutional encoder

This block of 208 bits is encoded with the ½ rate convolutional code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

This results in a block of 416 coded bits:  $\{c'(0),c'(1),...,c'(415)\}\$  defined by:

$$c'(2k) = u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6)$$

$$c'(2k+1) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-6) \qquad \text{for } k = 0, 1, ..., 207 \; ; \; u(k) = 0 \; \text{for } k < 0$$

# 4.11.4 Dummy bits insertion

Forty dummy bits are first inserted to the coded bits according to the following rule:

- c(k) = c'(k) for k = 0,1,2
- c(k) = c'(k-1) for k = 4,...,31
- c(k) = c'(k-2) for k = 33,...,39
- c(k) = c'(k-3) for k = 41,...,45
- c(k) = c'(k-5) for k = 48,...,67
- c(k) = c'(k-6) for k = 69,...,88
- c(k) = c'(k-7) for k = 90,...,95
- c(k) = c'(k-9) for k = 98,...,102
- c(k) = c'(k-10) for k = 104,...,123
- c(k) = c'(k-12) for k = 126,...,131
- c(k) = c'(k-13) for k = 133,...,145
- c(k) = c'(k-14) for k = 147,...,152
- c(k) = c'(k-16) for k = 155,...,180

for k = 183,...,188

for k = 204,...,209

= c'(k-18)

= c'(k-20)

c(k)

c(k)

- c(k) = c'(k-19) for k = 190,...,202
- c(k) = c'(k-22) for k = 212,...,231
- c(k) = c'(k-23) for k = 233,...,237
- c(k) = c'(k-25) for k = 240,...,245
- c(k) = c'(k-26) for k = 247,...,266
- c(k) = c'(k-27) for k = 268,...,287
- c(k) = c'(k-29) for k = 290,...,294
- c(k) = c'(k-30) for k = 296,...,302
- c(k) = c'(k-31) for k = 304,...,331
- c(k) = c'(k-32) for k = 333,...,344

= c'(k-34)

$$\begin{array}{lll} c(k) & = c'(k\text{-}36) & \text{for } k = 390,...,401 \\ \\ c(k) & = c'(k\text{-}38) & \text{for } k = 404,...,444 \\ \\ c(k) & = c'(k\text{-}40) & \text{for } k = 447,...,455 \\ \\ c(k) & = 0 & \text{for } k = 3, 32, 40, 46, 47, 68, 89, 96, 97, 103, 124, 125, 132, 146, 153, 154, 181, 182, \\ \\ 189, 203, 210, 211, 232, 238, 239, 246, 267, 288, 289, 295, 303, 332, 345, 346, 388, \\ \end{array}$$

# 4.11.5 Interleaving

c(k)

The interleaving is done as specified for the SACCH in subclause 4.1.4.

# 4.11.6 Mapping on a Burst

The mapping is given by the rule:

$$e(B,j) = i(B,j)$$
 and  $e(B,59+j) = i(B,57+j)$  for  $j = 0,1,...,56$ 

for k = 347....387

389, 402, 403, 445, 446

NOTE: The bits e(B,57) and e(B,58) on burst number B do not need to be set as they are used by the EPCCH (see subclause 4.12).

# 4.12 Enhanced power control channel (EPCCH)

## 4.12.1 Block code

The EPCCH message delivered to the encoder on every 120ms, and has a fixed size of 3 information bits  $\{pm(0), pm(1), pm(2)\}$ . The contents of the bits are defined in 3GPP TS 45.008 for both uplink and downlink.

The EPCCH information bits  $\{pm(n,0),pm(n,1),pm(n,2)\}$  are coded into 12 bits pb(B,k), k=0,1,...11 according to the following table (identical to the one used for USF in section 5.1.4.2):

pm(n,0),pm(n,1),pm(n,2)	pb(B,0),, pb(B,11)
000	000 000 000 000
001	000 011 011 101
010	001 101 110 110
011	001 110 101 011
100	110 100 001 011
101	110 111 010 110
110	111 001 111 101
111	111 010 100 000

# 4.12.2 Mapping on a Burst

The EPCCH message is mapped on the SACCH/TP burst.

The mapping is given by the rule:

$$e(B,j) = pb(B,k)$$
 for respectively  $j = 44, 47, 50, 53, 55, 57, 58, 60, 62, 65, 68, 71, and  $k = 0,1, ..., 11$$ 

# 5 Packet Switched Channels

# 5.1 Packet data traffic channel (PDTCH)

Thirteen coding schemes are specified for the packet data traffic channels. For the coding schemes CS-2 to CS-4 and MCS-1 to MCS-4, the first three bits (USF-bits) of the data block are encoded such that the first twelve coded bits are representing the same bit pattern, irrespective of the coding scheme, depending only on the USF-bits. For these coding schemes, the USF-bits can therefore always be decoded from these twelve bits in the same way. It should be noted that the USF precoding is done in the uplink direction for coding schemes CS-2 – CS-4, despite the fact that uplink RLC data block structure (3GPP TS 44.060) does not define USF-field.

For the nine coding schemes MCS-1 to MCS-9, the block structure differs between uplink and downlink since header sizes before coding are not the same.

In BTTI configuration, the RLC/MAC layer delivers to the encoder one data block every 20 ms. In RTTI configuration, the RLC/MAC layer delivers to the encoder one data block every 10 ms or, if BTTI USF mode is used (see 3GPP TS 45.002), the RLC/MAC layer in the downlink may deliver to the encoder two data blocks every 20 ms.

In the downlink direction, if BTTI USF mode is used (see 3GPP TS 45.002), one value of the USF per PDCH is delivered to the encoder every 20 ms; if RTTI USF mode is used (see 3GPP TS 45.002), one value of the USF per corresponding downlink PDCH-pair is delivered to the encoder every 10 ms.

If BTTI USF mode is used when sending downlink data blocks in RTTI configuration, then the USF need not be delivered to the encoder as the first three bits of a data block. In this case, the first three bits of a data block are set to an unspecified value (see 3GPP TS 44.060).

NOTE: How the USFs are delivered to the encoder in this case is implementation dependent.

If BTTI USF mode is used when sending downlink data blocks in RTTI configuration, then both data blocks sent in a 20 ms block period shall be encoded using coding schemes with the same modulation, unless

- Reduced Latency is supported by the MS to which the USF is intended or
- EGPRS2 is supported by the MS to which the USF is intended or
- The USF to be transmitted is set to an unused value (see 3GPP TS 45.002).

If an MS, that neither supports Reduced Latency nor EGPRS2, receives in a 20ms block period data blocks sent using different modulations, the MS shall ignore the USF.

If BTTI USF mode is used when sending downlink data blocks in RTTI configuration and different modulations are used in the two data blocks sent in a 20 ms block period, the USF will be sent with mixed modulation. In this case, the coding of the USF bits sent in the first 10 ms block period is according to the MCS used in that block period, while the coding of the USF bits sent in the second 10 ms block period is according to the MCS used in that block period. The network shall use only MCSs supported by the MS to which the USF is intended.

# 5.1.1 Packet data block type 1 (CS-1)

The coding scheme used for packet data block type 1 is the same as for SACCH as specified in section 4.1.

The flags hl(B) and hu(B) set to "1" identify the coding scheme CS-1.

In RTTI configuration with RTTI USF mode, the bursts with B = 0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B = 1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.

# 5.1.2 Packet data block type 2 (CS-2)

#### 5.1.2.1 Block constitution

The message delivered to the encoder has a fixed size of 271 information bits  $\{d(0),d(1),...,d(270)\}$ . It is delivered on a burst mode.

## 5.1.2.2 Block code

### a) USF precoding:

The first three bits d(0), d(1), d(2) are precoded into six bits u'(0), u'(1),...,u'(5) according to the following table:

d(0),d(1),d(2)	u'(0),u'(1),,u'(5)
000	000 000
001	001 011
010	010 110
011	011 101
100	100 101
101	101 110
110	110 011
111	111 000

#### b) Parity bits:

Sixteen parity bits p(0),p(1),...,p(15) are defined in such a way that in GF(2) the binary polynomial:

$$d(0)D^{286} + ... + d(270)D^{16} + p(0)D^{15} + ... + p(15)$$
, when divided by:

$$D^{16} + D^{12} + D^5 + 1$$
, yields a remainder equal to:

$$D^{15} + D^{14} + D^{13} + D^{12} + D^{11} + D^{10} + D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D^{+1}$$
.

#### c) Tail bits:

Four tail bits equal to 0 are added to the information and parity bits, the result being a block of 294 bits  $\{u(0),u(1),...,u(293)\}$ :

$$u(k) = u'(k)$$
 for  $k = 0,1,...,5$ 

$$u(k) = d(k-3)$$
 for  $k = 6,7,...,273$ 

$$u(k) = p(k-274)$$
 for  $k = 274,275,...,289$ 

$$u(k) = 0$$
 for  $k = 290,291,292,293$  (tail bits)

#### 5.1.2.3 Convolutional encoder

This block of 294 bits  $\{u(0),u(1),...,u(293)\}$  is encoded with the ½ rate convolutional code (identical to the one used for TCH/FS) defined by the polynomials:

$$G0 = 1 + D^3 + D^4$$

$$G1 = 1 + D + D^3 + D^4$$

This results in a block of 588 coded bits:  $\{C(0),C(1),...,C(587)\}\$  defined by:

$$C(2k) = u(k) + u(k-3) + u(k-4)$$

$$C(2k+1) = u(k) + u(k-1) + u(k-3) + u(k-4) \qquad \text{for } k = 0,1,...,293 \ ; \ u(k) = 0 \ \text{for } k < 0 \ \text{for } k = 0,1,...,293 \ ; \ u(k) = 0 \ \text{for } k < 0 \ \text{for } k = 0,1,...,293 \ ; \ u(k) = 0 \ \text{for } k < 0 \ \text{for } k = 0,1,...,293 \ ; \ u(k) = 0 \ \text{for } k < 0 \ \text{for } k = 0,1,...,293 \ ; \ u(k) = 0 \ \text{for } k < 0 \ \text{for } k = 0,1,...,293 \ ; \ u(k) = 0 \ \text{for } k < 0 \ \text{for } k = 0,1,...,293 \ ; \ u(k) = 0 \ \text{for } k < 0 \ \text{for } k = 0,1,...,293 \ ; \ u(k) = 0 \ \text{for } k < 0 \ \text{for } k = 0,1,...,293 \ ; \ u(k) = 0 \ \text{for } k < 0 \$$

The code is punctured in such a way that the following coded bits:

 $\{C(3+4j) \ for \ j=3,4,...,146 \ except \ for \ j=9,21,33,45,57,69,81,93,105,117,129,141\} \ are \ not \ transmitted$ 

The result is a block of 456 coded bits,  $\{c(0),c(1),...,c(455)\}$ .

## 5.1.2.4 Interleaving

The interleaving is done as specified for SACCH in section 4.1.4.

## 5.1.2.5 Mapping on a burst

The mapping is given by the rule:

$$e(B,j) = i(B,j)$$
 and  $e(B,59+j) = i(B,57+j)$  for  $j = 0,1,...,56$ 

and

$$e(B+m,57) = q(2m)$$
 and  $e(B+m,58) = q(2m+1)$  for  $m = 0,1,2,3$ 

where

q(0),q(1),...,q(7) = 1,1,0,0,1,0,0,0 identifies the coding scheme CS-2.

# 5.1.3 Packet data block type 3 (CS-3)

#### 5.1.3.1 Block constitution

The messages delivered to the encoder has a fixed size of 315 information bits  $\{d(0),d(1),...,d(314)\}$ . It is delivered on a burst mode.

#### 5.1.3.2 Block code

a) USF precoding:

The first three bits d(0), d(1), d(2) are precoded into six bits u'(0), u'(1),...,u'(5) as specified for CS-2 in section 5.1.2.2.a).

b) Parity bits:

Sixteen parity bits p(0),p(1),...,p(15) are defined in such a way that in GF(2) the binary polynomial:

$$d(0)D^{330} + ... + d(314)D^{16} + p(0)D^{15} + ... + p(15)$$
, when divided by:

 $D^{16} + D^{12} + D^5 + 1$ , yields a remainder equal to:

$$D^{15} + D^{14} + D^{13} + D^{12} + D^{11} + D^{10} + D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D^{+1}$$
.

c) Tail bits:

Four tail bits equal to 0 are added to the information and parity bits, the result being a block of 338 bits  $\{u(0),u(1),...,u(337)\}$ :

$$u(k) = u'(k)$$
 for  $k = 0,1,...,5$ 

$$u(k) = d(k-3)$$
 for  $k = 6,7,...,317$ 

$$u(k) = p(k-318)$$
 for  $k = 318,319,...,333$ 

$$u(k) = 0$$
 for  $k = 334,335,336,337$  (tail bits)

#### 5.1.3.3 Convolutional encoder

This block of 338 bits  $\{u(0),u(1),...,u(337)\}$  is encoded with the ½ rate convolutional code (identical to the one used for TCH/FS) defined by the polynomials:

$$G0 = 1 + D^3 + D^4$$

$$G1 = 1 + D + D^3 + D^4$$

This results in a block of 676 coded bits:  $\{C(0),C(1),...,C(675)\}\$  defined by:

$$C(2k) = u(k) + u(k-3) + u(k-4)$$

$$C(2k+1) = u(k) + u(k-1) + u(k-3) + u(k-4)$$
 for  $k = 0,1,...,337$ ;  $u(k) = 0$  for  $k < 0$ 

The code is punctured in such a way that the following coded bits:

$$\{C(3+6j) \text{ and } C(5+6j) \text{ for } j = 2,3,...,111\}$$
 are not transmitted

The result is a block of 456 coded bits,  $\{c(0),c(1),...,c(455)\}$ .

## 5.1.3.4 Interleaving

The interleaving is done as specified for SACCH in subclause 4.1.4.

# 5.1.3.5 Mapping on a burst

The mapping is given by the rule:

$$e(B,j) = i(B,j)$$
 and  $e(B,59+j) = i(B,57+j)$  for  $j = 0,1,...,56$ 

and

$$e(B+m,57) = q(2m)$$
 and  $e(B+m,58) = q(2m+1)$  for  $m = 0,1,2,3$ 

where

q(0),q(1),...,q(7) = 0,0,1,0,0,0,0,1 identifies the coding scheme CS-3.

# 5.1.4 Packet data block type 4 (CS-4)

#### 5.1.4.1 Block constitution

The message delivered to the encoder has a fixed size of 431 information bits  $\{d(0),d(1),...,d(430)\}$ . It is delivered on a burst mode.

## 5.1.4.2 Block code

## a) USF precoding:

The first three bits d(0),d(1),d(2) are block coded into twelve bits u'(0),u'(1),...,u'(11) according to the following table:

d(0),d(1),d(2)	u'(0),u'(1),,u'(11)
000	000 000 000 000
001	000 011 011 101
010	001 101 110 110
011	001 110 101 011
100	110 100 001 011
101	110 111 010 110
110	111 001 111 101
111	111 010 100 000

#### b) Parity bits:

Sixteen parity bits p(0),p(1),...,p(15) are defined in such a way that in GF(2) the binary polynomial:

$$d(0)D^{446} + ... + d(430)D^{16} + p(0)D^{15} + ... + p(15)$$
, when divided by:

 $D^{16} + D^{12} + D^5 + 1$ , yields a remainder equal to:

$$D^{15} + D^{14} + D^{13} + D^{12} + D^{11} + D^{10} + D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1.$$

The result is a block of 456 coded bits,  $\{c(0),c(1),...,c(455)\}$ :

$$c(k) = u'(k)$$
 for  $k = 0,1,...,11$ 

$$c(k) = d(k-9)$$
 for  $k = 12,13,...,439$ 

$$c(k) = p(k-440)$$
 for  $k = 440,441,...,455$ 

#### 5.1.4.3 Convolutional encoder

No convolutional coding is done.

# 5.1.4.4 Interleaving

The interleaving is done as specified for SACCH in section 4.1.4.

# 5.1.4.5 Mapping on a burst

The mapping is given by the rule:

$$e(B,j) = i(B,j)$$
 and  $e(B,59+j) = i(B,57+j)$  for  $j = 0,1,...,56$ 

and

$$e(B+m,57) = q(2m)$$
 and  $e(B+m,58) = q(2m+1)$  for  $m = 0,1,2,3$ 

where

q(0),q(1),...,q(7) = 0,0,0,1,0,1,1,0 identifies the coding scheme CS-4.

# 5.1.4a Packet data block type 5a (MCS-0)

## 5.1.4a.1 Downlink (MCS-0 DL)

#### 5.1.4a.1.1 Block constitution

The message delivered to the encoder has a fixed size of 207 information bits  $\{d(0),d(1),...,d(206)\}$ . It is delivered on a burst mode.

The message delivered to the encoder will have a fixed size of 210 information bits  $\{d(0),d(1),...,d(209)\}$ , if an eTFI field is included (see 3GPP TS 44.060).

#### 5.1.4a.1.2 USF precoding

Twelve bits u'(0),u'(1),...,u'(11) are generated as described for MCS-1 DL in subclause 5.1.5.1.2.2.

#### 5.1.4a.1.3 Data coding

a) Parity bits:

Eighteen data parity bits p(0),p(1),...,p(17) are defined in such a way that in GF(2) the binary polynomial:

$$d(31)D^{193} + ... + d(206)D^{18} + p(0)D^{17} + ... + p(17)$$
, when divided by:

$$D^{18} + D^{17} + D^{14} + D^{13} + D^{11} + D^{10} + D^{8} + D^{7} + D^{6} + D^{3} + D^{2} + 1$$
, yields a remainder equal to:

$$D^{17} + D^{16} + D^{15} + D^{14} + D^{13} + D^{12} + D^{11} + D^{10} + D^{9} + D^{8} + D^{7} + D^{6} + D^{5} + D^{4} + D^{3} + D^{2} + D + 1$$

In case an eTFI field (see 3GPP TS 44.060) is included the three bits  $\{d(207),d(208),d(209)\}$  are added bit-wise modulo 2 to the last three parity bits  $\{p(15),p(16),p(17)\}$ . This results in eighteen modified parity bits  $\{p(0),...,p(17)\}$  defined as:

$$pt(k) = p(k)$$
 for k=0,...,14  $pt(k) = d(k+192) + p(k)$  for k=15,...,17

#### b) Tail bits:

Six tail bits equal to 0 are added to the information bits, the result being a block of 182 bits  $\{u(0),u(1),...,u(181)\}$ :

$$u(k) = d(k+31)$$
 for  $k = 0,1,...,175$   $u(k) = 0$  for  $k = 176,177,...,181$  (tail bits)

#### c) Convolutional encoder

This block of 182 bits  $\{u(0),u(1),...,u(181)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^{2} + D^{3} + D^{5} + D^{6}$$

$$G7 = 1 + D + D^{2} + D^{3} + D^{6}$$

$$G5 = 1 + D + D^{4} + D^{6}$$

This results in a block of 546 coded bits:  $\{C(0),C(1),...,C(545)\}\$  defined by:

$$\begin{split} &C(3k) &= u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6) \\ &C(3k+1) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-6) \\ &C(3k+2) = u(k) + u(k-1) + u(k-4) + u(k-6) \quad \text{ for } k = 0,1,...,181; \ u(k) = 0 \text{ for } k < 0 \end{split}$$

The code is punctured in such a way that the following coded bits are not transmitted:

 $\{C(2+3j) \text{ for } j = 0,1,...,181\}$  are not transmitted except  $\{C(k) \text{ for } k = 35, 104, 173, 242, 308, 377, 446, 515}\}$  which are transmitted

The result is a block of 372 coded bits,  $\{dc(0),dc(1),...,dc(371)\}$ .

#### 5.1.4a.1.4 Header coding

The header bits  $\{d(3),d(4),...,d(30)\}$  shall be coded as for Packet data block type 5 (MCS-1) in subclause 5.1.5.1.3.

When applying the coding in subclause 5.1.5.1.3, the operations specific to eTFI inclusion therein shall be ignored.

Before coding  $\{d(8)..d(23)\}$  is replaced by  $\{p(0),...,p(15)\}$  and  $\{d(29),d(30)\}$  is replaced by  $\{p(16),p(17)\}$ , where  $\{p(0),...,p(17)\}$  is defined in 5.1.4a.1.3.

In case an eTFI field is included p(0),...,p(17) is replaced by pt(0),...pt(17) defined in 5.1.4a.1.3.

The result is a block of 68 coded bits,  $\{hc(0),hc(1),...,hc(67)\}$ .

#### 5.1.4a.1.5 Interleaving

The interleaving is done as specified for MCS-1 DL in subclause 5.1.5.1.5.

## 5.1.4a.1.6 Mapping on a burst

The mapping is done as specified for MCS-1 DL in subclause 5.1.5.1.6.2.

# 5.1.5 Packet data block type 5 (MCS-1)

# 5.1.5.1 Downlink (MCS-1 DL)

#### 5.1.5.1.1 Block constitution

The message delivered to the encoder has a fixed size of 209 information bits  $\{d(0),d(1),...,d(208)\}$ . It is delivered on a burst mode.

The message delivered to the encoder will have a fixed size of 212 information bits  $\{d(0),d(1),...,d(211)\}$ , if an eTFI field is included (see 3GPP TS 44.060).

The message delivered to the encoder will have a fixed size of 234 information bits  $\{d(0),d(1),...,d(233)\}$ , if a PAN field is included (see 3GPP TS 44.060).

The message delivered to the encoder will have a fixed size of 237 information bits {d(0),d(1),...,d(236)}, if a PAN field and an eTFI field are included (see 3GPP TS 44.060).

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

#### 5.1.5.1.2 USF precoding

#### 5.1.5.1.2.1 BTTI configuration

The first three bits d(0),d(1),d(2) are block coded into twelve bits u'(0),u'(1),...,u'(11) as for Packet data block type 4 (CS-4) in subclause 5.1.4.2.

#### 5.1.5.1.2.2 RTTI configuration

If the USF is sent in RTTI USF mode (see 3GPP TS 45.002) when data blocks are transmitted in RTTI configuration, then the first three bits d(0),d(1),d(2) are block coded into twelve bits u'(0),u'(1),...,u'(11) as for Packet data block type 4 (CS-4) in subclause 5.1.4.2.

If the USF is sent in BTTI USF mode (see 3GPP TS 45.002) when data blocks are transmitted in RTTI configuration, then the three bits of the USF to be sent on the lower numbered PDCH of a corresponding downlink PDCH-pair are block coded into twelve bits  $u_L(0), u_L(1), ..., u_L(11)$  as for Packet data block type 4 (CS-4) in subclause 5.1.4.2; the three bits of the USF to be sent on the higher numbered PDCH of a corresponding downlink PDCH-pair are block coded into twelve bits  $u_H(0), u_H(1), ..., u_H(11)$  as for Packet data block type 4 (CS-4) in subclause 5.1.4.2.

NOTE: If BTTI USF mode is used when sending data blocks in RTTI configuration, then d(0),d(1),d(2) need not contain a USF; in this case, they are ignored by the encoder. How the USFs are delivered to the encoder in this case is implementation dependent.

If the data block is sent in the first 10ms of a 20ms block period, then

```
\begin{split} u'(i) &= u_L(i) & \text{for } i = 0,\, 4,\, 8; \\ u'(i) &= u_H(i\text{-}1) & \text{for } i = 1,\, 5,\, 9; \\ u'(i) &= u_L(i\text{-}1) & \text{for } i = 2,\, 6,\, 10; \\ u'(i) &= u_H(i\text{-}2) & \text{for } i = 3,\, 7,\, 11. \end{split}
```

If the data block is sent in the second 10ms of a 20ms block period, then

```
u'(i) = u_L(i+2) for i = 0, 4, 8;

u'(i) = u_H(i+1) for i = 1, 5, 9;
```

$$u'(i) = u_L(i+1)$$
 for  $i = 2, 6, 10$ ;

$$u'(i) = u_H(i)$$
 for  $i = 3, 7, 11$ .

NOTE: In case mixed modulation USF is used (see subclause 5.1), the USF bits sent during the other half of the 20 ms block period may be sent with a different modulation. In this case, the half of  $u_L$  and  $u_H$  not sent in the present data block will be discarded.

#### 5.1.5.1.3 Header coding

#### a) Parity bits:

Eight header parity bits p(0),p(1),...,p(7) are defined in such a way that in GF(2) the binary polynomial:

$$d(3)D^{35} + ... + d(30)D^8 + p(0)D^7 + ... + p(7)$$
, when divided by:

$$D^8 + D^6 + D^3 + 1$$
, yields a remainder equal to:

$$D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$$
.

In case an eTFI field (see 3GPP TS 44.060) is included and a PAN field is not the three bits  $\{d(209),d(210),d(211)\}$  are added bit-wise modulo 2 to the last three parity bits  $\{p(5),p(6),p(7)\}$ . This results in the eight modified header parity bits  $\{p(0),...,p(7)\}$  defined as:

$$pt(k) = p(k)$$
 for k=0,...,4

$$pt(k) = d(k+204) + p(k)$$
 for k=5,...,7

In case an eTFI field (see 3GPP TS 44.060) and a PAN field are included the three bits  $\{d(234),d(235),d(236)\}$  are added bit-wise modulo 2 to the last three parity bits  $\{p(5), p(6), p(7)\}$ . This results in the eight modified header parity bits  $\{p(0),...,pt(7)\}$  defined as:

$$pt(k) = p(k)$$
 for k=0,...,4

$$pt(k) = d(k+229) + p(k)$$
 for k=5,...,7

#### b) Tail biting:

The six last header parity bits are added before information and parity bits, the result being a block of 42 bits  $\{u"(-6),...,u"(0),u"(1),...,u"(35)\}$  with six negative indexes. In case an eTFI field is not included this operation is defined as:

$$u''(k-6) = p(k+2)$$
 for  $k = 0,1,...,5$ 

$$u''(k) = d(k+3)$$
 for  $k = 0,1,...,27$ 

$$u''(k) = p(k-28)$$
 for  $k = 28,29,...,35$ 

In case an eTFI field is included p(k) is replaced by pt(k) in the above operation.

#### c) Convolutional encoder

This block of 42 bits  $\{u''(-6),...,u''(0),u''(1),...,u''(35)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 108 coded bits:  $\{C(0),C(1),...,C(107)\}\$  defined by:

$$C(3k) = u''(k) + u''(k-2) + u''(k-3) + u''(k-5) + u''(k-6)$$

$$C(3k+1) = u''(k) + u''(k-1) + u''(k-2) + u''(k-3) + u''(k-6)$$

$$C(3k+2) = u''(k) + u''(k-1) + u''(k-4) + u''(k-6)$$
 for  $k = 0,1,...,35$ 

The code is punctured in such a way that the following coded bits:

$$\{C(2+3j) \text{ for } j = 0,1,...,35\}$$
 as well as  $\{C(k) \text{ for } k = 34,58,82,106\}$  are not transmitted

The result is a block of 68 coded bits,  $\{hc(0),hc(1),...,hc(67)\}$ .

## 5.1.5.1.4 Data coding

#### a) Parity bits:

Twelve data parity bits p(0), p(1), ..., p(11) are defined in such a way that in GF(2) the binary polynomial:

$$d(31)D^{189} + ... + d(208)D^{12} + p(0)D^{11} + ... + p(11)$$
, when divided by:

$$D^{12} + D^{11} + D^{10} + D^8 + D^5 + D^4 + 1$$
, yields a remainder equal to:

$$D^{11} + D^{10} + D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$$
.

#### b) Tail bits:

Six tail bits equal to 0 are added to the information and parity bits, the result being a block of 196 bits  $\{u(0),u(1),...,u(195)\}$ :

$$u(k) = d(k+31)$$
 for  $k = 0,1,...,177$ 

$$u(k) = p(k-178)$$
 for  $k = 178,179,...,189$ 

$$u(k) = 0$$
 for  $k = 190,191,...,195$  (tail bits)

#### c) Convolutional encoder

This block of 196 bits  $\{u(0),u(1),...,u(195)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 588 coded bits:  $\{C(0),C(1),...,C(587)\}\$  defined by:

$$C(3k) = u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6)$$

$$C(3k+1) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-6)$$

$$C(3k+2) = u(k) + u(k-1) + u(k-4) + u(k-6)$$
 for  $k = 0,1,...,195$ ;  $u(k) = 0$  for  $k < 0$ 

The code is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied in such a way that the following coded bits:

P1	$\{C(2+21j), C(5+21j), C(8+21j), C(10+21j), C(11+21j), C(14+21j), C(17+21j), C(20+21j) \text{ for } j=1,2,3,3,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4$
	0,1,,27} are not transmitted except {C(k) for k = 73,136,199,262,325,388,451,514} which are
	transmitted
P2	$\{C(1+21j), C(4+21j), C(7+21j), C(9+21j), C(13+21j), C(15+21j), C(16+21j), C(19+21j) \text{ for } j=0,1,,27\}$
	are not transmitted except {C(k) for k = 78,141,204,267,330,393,456,519} which are transmitted

The result is a block of 372 coded bits, {dc(0),dc(1),...,dc(371)}.

For the FANR procedure, the code is punctured depending on the CPS field and the PANI field as defined in 3GPP TS 44.060. If the PANI field is set to 0, the puncturing is the same as for EGPRS. If the PANI field is set to 1, the puncturing schemes named P1 or P2 are applied in such a way that, in addition to the bits punctured for EGPRS, the following coded bits:

P1	$\{C(3+21j), C(12+21j) \text{ for } j=0,1,,27\}$ are not transmitted except $\{C(k) \text{ for } k=33,96,159,222,369,432,495,558}\}$ which are transmitted
P2	$\{C(6+21j), C(18+21j) \text{ for } j=0,1,,27\}$ are not transmitted except $\{C(k) \text{ for } k=39,102,165,228,375,438,501,564}\}$ which are transmitted

The result is a block of 324 coded bits  $\{pc(0), pc(1), ..., pc(323)\}$ .

#### 5.1.5.1.4a Piggy-backed Ack/Nack coding

The operations in this subclause shall be carried out only if a PAN field is included.

#### a) Parity bits

Ten PAN parity bits p(0),p(1),...,p(9) are defined in such a way that in GF(2) the binary polynomial:

$$d(209)D^{29} + ... + d(228)D^{10} + p(0)D^9 + ... + p(9)$$
, when divided by:

$$D^{10} + D^9 + D^5 + D^4 + D + 1$$
, yields a remainder equal to:

$$D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D^1 + 1.$$

The five bits  $\{d(229),...,d(233)\}$  (TFI value or 00000, see 3GPP TS 44.060) are added bit-wise modulo 2 to the 5 last parity bits  $\{p(5),...,p(9)\}$ . This results in the ten modified PAN parity bits  $\{p(0),...,p(9)\}$  defined as:

$$pt(k) = p(k)$$
 for k=0,...,4  
 $pt(k) = d(k+224) + p(k)$  for k=5,...,9

#### b) Tail biting:

The six last modified PAN parity bits are added before information and modified PAN parity bits, the result being a block of 36 {u''(-6),...,u''(0),u''(1),...,u''(29)} bits with six negative indexes:

$$u''(k-6) = pt(k+4)$$
 for  $k = 0,1,...,5$   
 $u''(k) = d(k+209)$  for  $k = 0,1,...,19$   
 $u''(k) = pt(k-20)$  for  $k = 20,21,...,29$ 

#### c) Convolutional encoder

The block of 36 bits  $\{u''(-6),...,u''(0),u''(1),...,u''(29)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$
$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 90 coded bits  $\{C(0),C(1),...,C(89)\}$  defined by:

$$\begin{split} &C(3k) &= u"(k) + u"(k-2) + u"(k-3) + u"(k-5) + u"(k-6) \\ &C(3k+1) = u"(k) + u"(k-1) + u"(k-2) + u"(k-3) + u"(k-6) \\ &C(3k+2) = u"(k) + u"(k-1) + u"(k-4) + u"(k-6) & \text{for } k = 0,1,...,29 \end{split}$$

The block of 90 coded bits is punctured in such way that the following coded bits:

$$\{C(15k), C(2+15k), C(4+15k), C(6+15k), C(7+15k), C(10+15k), C(13+15k) \text{ for } k=0,1,...,5\}$$
 are not transmitted.

The result is a block of 48 coded bits  $\{ac(0),ac(1),...,ac(47)\}$ .

The data coded bits  $\{pc(0),pc(1),...,pc(323)\}$  are appended to the PAN coded bits by the following rule:

$$dc(k) = ac(k)$$
 for  $k = 0,1,...,47$   
 $dc(k) = pc(k-48)$  for  $k = 48,49,...,371$ 

The result is a block of 372 coded bits  $\{dc(0),dc(1),...,dc(371)\}$ .

## 5.1.5.1.5 Interleaving

The USF, header and data are put together as one entity as described by the following rule:

$$\begin{array}{lll} c(k) = u'(k) & \text{for } k = 0,1,...,11 \\ \\ c(k) = hc(k-12) & \text{for } k = 12,13,...,79 \\ \\ c(k) = dc(k-80) & \text{for } k = 80,81,...,451 \\ \\ c'(n,k) = c(n,k) & \text{for } k = 0,1,...,24 \\ \\ c'(n,k) = c(n,k-1) & \text{for } k = 26,27,...,81 \\ \\ c'(n,k) = c(n,k-2) & \text{for } k = 83,84,...,138 \\ \\ c'(n,k) = c(n,k-3) & \text{for } k = 140,141,...,423 \\ \\ c'(n,k) = c(n,k-4) & \text{for } k = 425,426,...,455 \\ \\ c'(n,25) = q(8) & c'(n,82) = q(9) & c'(n,139) = q(10) & c'(n,424) = q(11) \\ \end{array}$$

c(n,k) are the coded bits and q(8),q(9),...,q(11) = 0,0,0,0 are four extra stealing flags

The resulting block is interleaved according to the following rule:

$$\begin{split} i(B,j) &= c'(n,k) \quad \text{ for } k = 0,1,...,455 \\ n &= 0,1,...,N,N+1,... \\ B &= B_0 + 4n + (k \text{ mod } 4) \\ j &= 2((49k) \text{ mod } 57) + ((k \text{ mod } 8) \text{ div } 4) \end{split}$$

## 5.1.5.1.6 Mapping on a burst

#### 5.1.5.1.6.1 BTTI configuration

The mapping is given by the rule:

$$e(B,j) = i(B,j)$$
 and  $e(B,59+j) = i(B,57+j)$  for  $j = 0,1,...,56$ 

and

$$e(B+m,57) = q(2m)$$
 and  $e(B+m,58) = q(2m+1)$  for  $m = 0,1,2,3$ 

where

$$q(0),q(1),...,q(7) = 0,0,0,1,0,1,1,0.$$

Note: For a standard GPRS MS, bits q(0),...,q(7) indicates that the USF is coded as for CS-4.

#### 5.1.5.1.6.2 RTTI configuration

a) Bit swapping

After the interleaving the following bits are swapped:

If the RTTI radio block is sent in the first 10ms of a 20ms block period:

Swap i(B+1,98) with i(B+1,0)

Swap i(B+1,35) with i(B+1,51)

Swap i(B+1,84) with i(B+1,100)

Swap i(B+2,98) with i(B+2,82)

Swap i(B+2,35) with i(B+2,19)

Swap i(B+2,84) with i(B+2,68)

Swap i(B+3,35) with i(B+3,3)

Swap i(B+3,84) with i(B+3,52)

Swap i(B+3,98) with i(B+3,66)

If the RTTI radio block is sent in the second 10ms of a 20ms block period:

Swap i(B,19) with i(B,51)

Swap i(B,68) with i(B,100)

Swap i(B,82) with i(B,0)

Swap i(B+1,19) with i(B+1,35)

Swap i(B+1,68) with i(B+1,84)

Swap i(B+1,82) with i(B+1,98)

Swap i(B+2,19) with i(B+2,3)

Swap i(B+2,68) with i(B+2,52)

Swap i(B+2,82) with i(B+2,66)

## b) Mapping on bursts

The mapping is given by the rule:

$$e(B,j) = i(B,j)$$
 and  $e(B,59+j) = i(B,57+j)$  for  $j = 0,1,...,56$ 

and

$$e(B+m,57) = q(2m)$$
 and  $e(B+m,58) = q(2m+1)$  for  $m = 0,1,2,3$ 

where q(0),q(1),...,q(7) are set according to the following table, depending on the USF mode (see 3GPP TS 45.002):

	in the first 10ms of a	in the second 10ms of a
	20ms block period	20ms block period
USF sent in BTTI USF mode	0,0,0,0,0,1,0,1	0,1,0,1,1,0,1,0
USF sent in RTTI USF mode	0,0,0,1,0,1,1,0	

### c) Mapping on PDCHs

The bursts with B = 0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B = 1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.

## 5.1.5.2 Uplink (MCS-1 UL)

#### 5.1.5.2.1 Block constitution

The block constitution is the same as for MCS-1 DL specified in subclause 5.1.5.1.1, with the exception that an eTFI field can only be included in combination with a PAN field, see sub-clause 5.1.5.2.3a.

#### 5.1.5.2.2 Header coding

a) Parity bits:

Eight header parity bits p(0),p(1),...,p(7) are defined in such a way that in GF(2) the binary polynomial:

$$d(0)D^{38} + ... + d(30)D^{8} + p(0)D^{7} + ... + p(7)$$
, when divided by:

$$D^8 + D^6 + D^3 + 1$$
, yields a remainder equal to:

$$D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$$
.

b) Tail biting:

The six last header parity bits are added before information and parity bits, the result being a block of 45 bits  $\{u''(-6),...,u''(0),u''(1),...,u''(38)\}$  with six negative indexes:

$$u''(k-6) = p(k+2)$$
 for  $k = 0,1,...,5$ 

$$u''(k) = d(k)$$
 for  $k = 0,1,...,30$ 

$$u''(k) = p(k-31)$$
 for  $k = 31,32,...,38$ 

c) Convolutional encoder

This block of 45 bits  $\{u''(-6),...,u''(0),u''(1),...,u''(38)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 117 coded bits:  $\{C(0),C(1),...,C(116)\}$  defined by:

$$C(3k) = u''(k) + u''(k-2) + u''(k-3) + u''(k-5) + u''(k-6)$$

$$C(3k+1) = u''(k) + u''(k-1) + u''(k-2) + u''(k-3) + u''(k-6)$$

$$C(3k+2) = u''(k) + u''(k-1) + u''(k-4) + u''(k-6)$$
 for  $k = 0,1,...,38$ 

The code is punctured in such a way that the following coded bits:

 $\{C(5+12j),\,C(8+12j),\,C(11+12j),\,for\,j=0,1,...,8\}\,\,as\,\,well\,\,as\,\,\{C(k)\,\,for\,\,k=26,38,50,62,74,86,98,110,113,116\}\,\,are\,\,not\,\,transmitted$ 

The result is a block of 80 coded bits,  $\{hc(0),hc(1),...,hc(79)\}$ .

## 5.1.5.2.3 Data coding

The data coding is the same as for downlink as specified in subclause 5.1.5.1.4.

#### 5.1.5.2.3a Piggy-backed Ack/Nack coding

If a PAN field is included and an eTFI field is not included, the PAN coding is the same as for the downlink as specified in subclause 5.1.5.1.4a.

If a PAN field and an eTFI field are included, the PAN coding is the same as for the downlink as specified in subclause 5.1.5.1.4a, with the addition that the five bits  $\{d(229),d(230),d(231),d(232),d(233)\}$  (TFI value or 00000, see 3GPP TS 44.060) and the three bits  $\{d(234),d(235),d(236)\}$  (eTFI value or 000, see 3GPP TS 44.060) are added bit-wise modulo 2 to the PAN parity bits  $\{p(2),p(3),p(4),p(5),p(6),p(7),p(8),p(9)\}$  resulting in the ten modified PAN parity bits defined as:

$$\begin{aligned} pt(k) &= p(k) & \text{for } k = 0, ..., 1 \\ pt(k) &= d(k + 232) + p(k) & \text{for } k = 2, ..., 4 \\ pt(k) &= d(k + 224) + p(k) & \text{for } k = 5, ..., 9 \end{aligned}$$

## 5.1.5.2.4 Interleaving

The header and data are put together as one entity as described by the following rule:

$$c(k) = hc(k) \qquad \text{for } k = 0,1,...,79$$
 
$$c(k) = dc(k-80)) \qquad \text{for } k = 80,81,...,451$$
 
$$c'(n,k) = c(n,k) \qquad \text{for } k = 0,1,...,24$$
 
$$c'(n,k) = c(n,k-1) \qquad \text{for } k = 26,27,...,81$$
 
$$c'(n,k) = c(n,k-2) \qquad \text{for } k = 83,84,...,138$$
 
$$c'(n,k) = c(n,k-3) \qquad \text{for } k = 140,141,...,423$$
 
$$c'(n,k) = c(n,k-4) \qquad \text{for } k = 425,426,...,455$$
 
$$c'(n,25) = q(8) \qquad c'(n,82) = q(9) \qquad c'(n,139) = q(10) c'(n,424) = q(11)$$
 
$$c(n,k) \text{ are the coded bits and } q(8),q(9),...,q(11) = 0,0,0,0 \text{ are four extra stealing flags}$$

The resulting block is interleaved according to the following rule:

$$\begin{split} i(B,j) &= c'(n,k) \quad \text{ for } k = 0,1,...,455 \\ n &= 0,1,...,N,N+1,... \\ B &= B_0 + 4n + (k \text{ mod } 4) \\ j &= 2((49k) \text{ mod } 57) + ((k \text{ mod } 8) \text{ div } 4) \end{split}$$

## 5.1.5.2.5 Mapping on a burst

In BTTI configuration, the mapping is the same as for MCS-1 DL as specified in subclause 5.1.5.1.6.1.

NOTE: This mapping is also applied in RTTI configuration.

In RTTI configuration, the bursts with B = 0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B = 1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.

# 5.1.6 Packet data block type 6 (MCS-2)

### 5.1.6.1 Downlink (MCS-2 DL)

#### 5.1.6.1.1 Block constitution

The message delivered to the encoder has a fixed size of 257 information bits  $\{d(0),d(1),...,d(256)\}$ . It is delivered on a burst mode.

The message delivered to the encoder will have a fixed size of 260 information bits  $\{d(0),d(1),...,d(259)\}$ , if an eTFI field is included (see 3GPP TS 44.060).

The message delivered to the encoder will have a fixed size of 282 information bits {d(0),d(1),...,d(281)} if a PAN field is included.

The message delivered to the encoder will have a fixed size of 285 information bits  $\{d(0),d(1),...,d(284)\}$ , if a PAN field and an eTFI field are included (see 3GPP TS 44.060).

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

#### 5.1.6.1.2 USF precoding

#### 5.1.6.1.2.1 BTTI configuration

The first three bits d(0),d(1),d(2) are block coded into twelve bits u'(0),u'(1),...,u'(11) as for Packet data block type 4 (CS-4) in subclause 5.1.4.2.

#### 5.1.6.1.2.2 RTTI configuration

Twelve bits u'(0),u'(1),...,u'(11) are generated as described for MCS-1 DL in subclause 5.1.5.1.2.2.

## 5.1.6.1.3 Header coding

A block of 68 coded bits  $\{hc(0),hc(1),...,hc(67)\}\$  is derived from  $\{d(3),d(4),...,d(30)\}\$  as described for MCS-1 DL in subclause 5.1.5.1.3, with the exception that eTFI field bits are now located in  $\{d(257),d(258),d(259)\}\$  if included when PAN is not included, and located in  $\{d(282),d(283),d(284)\}\$  if included in combination with a PAN field.

#### 5.1.6.1.4 Data coding

a) Parity bits:

Twelve data parity bits p(0),p(1),...,p(11) are defined in such a way that in GF(2) the binary polynomial:

$$d(31)D^{237}+...+d(256)D^{12}+p(0)D^{11}+...+p(11), \ when \ divided \ by:$$
 
$$D^{12}+D^{11}+D^{10}+D^8+D^5+D^4+1, \ yields \ a \ remainder \ equal \ to:$$

$$D^{11} + D^{10} + D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$$
.

b) Tail bits:

Six tail bits equal to 0 are added to the information and parity bits, the result being a block of 244 bits  $\{u(0),u(1),...,u(243)\}$ :

$$\begin{array}{ll} u(k) &= d(k+31) & \text{ for } k=0,1,...,225 \\ \\ u(k) &= p(k-226) & \text{ for } k=226,227,...,237 \\ \\ u(k) &= 0 & \text{ for } k=238,239,...,243 \text{ (tail bits)} \end{array}$$

#### c) Convolutional encoder

This block of 244 bits  $\{u(0),u(1),...,u(243)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$
 
$$G7 = 1 + D + D^2 + D^3 + D^6$$
 
$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 732 coded bits:  $\{C(0),C(1),...,C(731)\}\$  defined by:

$$C(3k) = u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6)$$

$$\begin{split} &C(3k+1)=u(k)+u(k-1)+u(k-2)+u(k-3)+u(k-6)\\ &C(3k+2)=u(k)+u(k-1)+u(k-4)+u(k-6) & \text{ for } k=0,1,...,243; \ u(k)=0 \ \text{ for } k<0 \end{split}$$

The code is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied in such a way that the following coded bits:

P1	$\{C(6j), C(1+6j), C(5+6j) \text{ for } j = 0,1,,121\}$ and $\{C(k) \text{ for } k = 57,171,285,399,513,627\}$ are transmitted
P2	$\{C(2+6j), C(3+6j), C(4+6j) \text{ for } j = 0,1,,121\}$ and $\{C(k) \text{ for } k = 108,222,336,450,564,678\}$ are
	transmitted

The result is a block of 372 coded bits,  $\{dc(0),dc(1),...,dc(371)\}$ .

For the FANR procedure, the code is punctured depending on the CPS field and the PANI field as defined in 3GPP TS 44.060. If the PANI field is set to 0, the puncturing is the same as for EGPRS. If the PANI field is set to 1, the puncturing schemes named P1 or P2 are applied in such a way that, in addition to the bits punctured for EGPRS, the following coded bits:

P1	$\{C(18+30j), C(30+30j) \text{ for } j=0,1,,23\}$ are not transmitted
P2	$\{C(9+30j), C(27+30j) \text{ for } j = 0,1,,23\}$ are not transmitted

The result is a block of 324 coded bits  $\{pc(0),pc(1),...,pc(323)\}$ .

## 5.1.6.1.4a Piggy-backed Ack/Nack coding

The operations in this subclause shall be carried out only if a PAN field is included.

A block of 48 coded bits  $\{ac(0),ac(1),...,ac(47)\}\$  is derived from  $\{d(257),d(258),...,d(281)\}\$  as described for MCS-1 DL in subclause 5.1.5.1.4a, with bits  $\{d(209),d(210),...,d(233)\}\$  replaced by bits  $\{d(257),d(258),...,d(281)\}\$ .

The data coded bits  $\{pc(0),pc(1),...,pc(323)\}$  are appended to the PAN coded bits as described for MCS-1 DL in subclause 5.1.5.1.4a. The result is a block of 372 coded bits  $\{dc(0),dc(1),...,dc(371)\}$ .

#### 5.1.6.1.5 Interleaving

The interleaving is done as specified for MCS-1 DL in subclause 5.1.5.1.5.

#### 5.1.6.1.6 Mapping on a burst

The mapping is done as specified for MCS-1 DL in subclause 5.1.5.1.6.

#### 5.1.6.2 Uplink (MCS-2 UL)

# 5.1.6.2.1 Block constitution

The block constitution is the same as for MCS-2 DL specified in subclause 5.1.6.1.1, with the exception that an eTFI field can only be included in combination with a PAN field, see sub-clause 5.1.6.2.3a.

### 5.1.6.2.2 Header coding

A block of 80 coded bits  $\{hc(0),hc(1),...,hc(79)\}$  is derived from  $\{d(0),d(1),...,d(30)\}$  as described for MCS-1 UL in subclause 5.1.5.2.2.

### 5.1.6.2.3 Data coding

The data coding is the same as for downlink as specified in subclause 5.1.6.1.4.

## 5.1.6.2.3a Piggy-backed Ack/Nack coding

If a PAN field is included and an eTFI field is not included, its coding is the same as for downlink as specified in subclause 5.1.6.1.4a.

If a PAN field and an eTFI field are included, the coding of these fields are the same as for the downlink as specified in subclause 5.1.6.1.4a, with the addition that the TFI and eTFI values being added bit-wise modulo 2 to the last eight PAN parity bits as specified in subclause 5.1.5.2.3a.

#### 5.1.6.2.4 Interleaving

The interleaving is the same as for MCS-1 UL as specified in subclause 5.1.5.2.4.

#### 5.1.6.2.5 Mapping on a burst

The mapping is the same as for MCS-1 UL as specified in subclause 5.1.5. 2.5.

# 5.1.7 Packet data block type 7 (MCS-3)

# 5.1.7.1 Downlink (MCS-3 DL)

## 5.1.7.1.1 Block constitution

The message delivered to the encoder has a fixed size of 329 information bits  $\{d(0),d(1),...,d(328)\}$ . It is delivered on a burst mode.

The message delivered to the encoder will have a fixed size of 332 information bits  $\{d(0),d(1),...,d(331)\}$ , if an eTFI field is included (see 3GPP TS 44.060).

The message delivered to the encoder will have a fixed size of 354 information bits {d(0),d(1),...,d(353)} if a PAN field is included.

The message delivered to the encoder will have a fixed size of 357 information bits  $\{d(0),d(1),...,d(356)\}$ , if a PAN field and an eTFI field are included (see 3GPP TS 44.060).

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

#### 5.1.7.1.2 USF precoding

#### 5.1.7.1.2.1 BTTI configuration

The first three bits d(0),d(1),d(2) are block coded into twelve bits u'(0),u'(1),...,u'(11) as for Packet data block type 4 (CS-4) in subclause 5.1.4.2.

#### 5.1.7.1.2.2 RTTI configuration

Twelve bits u'(0),u'(1),...,u'(11) are generated as described for MCS-1 DL in subclause 5.1.5.1.2.2.

#### 5.1.7.1.3 Header coding

A block of 68 coded bits  $\{hc(0),hc(1),...,hc(67)\}$  is derived from  $\{d(3),d(4),...,d(30)\}$  as described for MCS-1 DL in subclause 5.1.5.1.3, with the exception that eTFI field bits are now located in  $\{d(329),d(330),d(331)\}$  if included when PAN is not included, and located in  $\{d(354),d(355),d(356)\}$  if included in combination with a PAN field.

#### 5.1.7.1.4 Data coding

#### a) Parity bits:

Twelve data parity bits p(0),p(1),...,p(11) are defined in such a way that in GF(2) the binary polynomial:

$$d(31)D^{309} + ... + d(328)D^{12} + p(0)D^{11} + ... + p(11)$$
, when divided by:

$$D^{12}+D^{11}+D^{10}+D^8+D^5+D^4+1,$$
 yields a remainder equal to:

$$D^{11} + D^{10} + D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$$

#### b) Tail bits:

Six tail bits equal to 0 are added to the information and parity bits, the result being a block of 316 bits  $\{u(0),u(1),...,u(315)\}$ :

$$\begin{array}{ll} u(k) &= d(k+31) & \text{ for } k=0,1,...,297 \\ \\ u(k) &= p(k-298) & \text{ for } k=298,299,...,309 \\ \\ u(k) &= 0 & \text{ for } k=310,311,...,315 \text{ (tail bits)} \end{array}$$

#### c) Convolutional encoder

This block of 316 bits  $\{u(0),u(1),...,u(315)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^{2} + D^{3} + D^{5} + D^{6}$$

$$G7 = 1 + D + D^{2} + D^{3} + D^{6}$$

$$G5 = 1 + D + D^{4} + D^{6}$$

This results in a block of 948 coded bits:  $\{C(0),C(1),...,C(947)\}\$  defined by:

$$\begin{split} &C(3k) &= u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6) \\ &C(3k+1) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-6) \\ &C(3k+2) = u(k) + u(k-1) + u(k-4) + u(k-6) \quad \text{ for } k = 0,1,...,315; \ u(k) = 0 \text{ for } k < 0 \end{split}$$

The code is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied in such a way that the following coded bits:

P1	$\{C(18j), C(1+18j), C(3+18j), C(6+18j), C(10+18j), C(14+18j), C(17+18j) \text{ for } j=0,1,,51\}$
	and {C(k) for k = 241,475,709, 936,937,939,942,946 } are transmitted
P2	$\{C(2+18j), C(5+18j), C(6+18j), C(7+18j), C(9+18j), C(12+18j), C(16+18j) \text{ for } j=0,1,,51\}$
	and {C(k) for k = 121,355,589, 938,941,942,943,945 } are transmitted
P3	$\{C(18j), C(4+18j), C(8+18j), C(11+18j), C(12+18j), C(13+18j), C(15+18j) \text{ for } j=0,1,,51\}$
	and {C(k) for k = 181,289,523,811, 936,940,944,947 } are transmitted

The result is a block of 372 coded bits, {dc(0),dc(1),...,dc(371)}.

For the FANR procedure, the code is punctured depending on the CPS field and the PANI field as defined in 3GPP TS 44.060. If the PANI field is set to 0, the puncturing is the same as for EGPRS. If the PANI field is set to 1, the puncturing schemes named P1 or P2 are applied in such a way that, in addition to the bits punctured for EGPRS, the following coded bits:

P1	$\{C(18j) \text{ for } j=0,1,,51\}$ are not transmitted except $\{C(k) \text{ for } k=108,342,576,810\}$ which are transmitted
P2	$\{C(6+18j) \text{ for } j=0,1,,51\}$ are not transmitted except $\{C(k) \text{ for } k=186,294,528,762\}$ which are transmitted
P3	$\{C(12+18j) \text{ for } j=0,1,,51\}$ are not transmitted except $\{C(k) \text{ for } k=66,390,642,876\}$ which are transmitted

The result is a block of 324 coded bits  $\{pc(0),pc(1),...,pc(323)\}$ .

## 5.1.7.1.4a Piggy-backed Ack/Nack coding

The operations in this subclause shall be carried out only if a PAN field is included.

A block of 48 coded bits  $\{ac(0),ac(1),...,ac(47)\}\$  is derived from  $\{d(329),d(330),...,d(353)\}\$  as described for MCS-1 DL in subclause 5.1.5.1.4a, with bits  $\{d(209),d(210),...,d(233)\}\$  replaced by bits  $\{d(329),d(330),...,d(353)\}\$ .

The data coded bits  $\{pc(0),pc(1),...,pc(323)\}$  are appended to the PAN coded bits as described for MCS-1 DL in subclause 5.1.5.1.4a. The result is a block of 372 coded bits  $\{dc(0),dc(1),...,dc(371)\}$ .

#### 5.1.7.1.5 Interleaving

The interleaving is done as specified for MCS-1 DL in subclause 5.1.5.1.5.

#### 5.1.7.1.6 Mapping on a burst

The mapping is done as specified for MCS-1 DL in subclause 5.1.5.1.6.

# 5.1.7.2 Uplink (MCS-3 UL)

#### 5.1.7.2.1 Block constitution

The block constitution is the same as for MCS-3 DL specified in subclause 5.1.7.1.1, with the exception that an eTFI field can only be included in combination with a PAN field, see sub-clause 5.1.7.2.3a.

### 5.1.7.2.2 Header coding

A block of 80 coded bits  $\{hc(0),hc(1),...,hc(79)\}$  is derived from  $\{d(0),d(1),...,d(30)\}$  as described for MCS-1 UL in subclause 5.1.5.2.2.

## 5.1.7.2.3 Data coding

The data coding is the same as for downlink as specified in subclause 5.1.7.1.4.

# 5.1.7.2.3a Piggy-backed Ack/Nack coding

If a PAN field is included and an eTFI field is not included, its coding is the same as for downlink as specified in subclause 5.1.7.1.4a.

If a PAN field and an eTFI field are included, the coding of theses fields are the same as for the downlink as specified in subclause 5.1.7.1.4a, with the addition that the TFI and eTFI values being added bit-wise modulo 2 to the last eight PAN parity bits as specified in subclause 5.1.5.2.3a.

#### 5.1.7.2.4 Interleaving

The interleaving is the same as for MCS-1 UL as specified in subclause 5.1.5.2.4.

# 5.1.7.2.5 Mapping on a burst

The mapping is the same as for MCS-1 UL as specified in subclause 5.1.5. 2.5.

# 5.1.8 Packet data block type 8 (MCS-4)

#### 5.1.8.1 Downlink (MCS-4 DL)

#### 5.1.8.1.1 Block constitution

The message delivered to the encoder has a fixed size of 385 information bits  $\{d(0),d(1),...,d(384)\}$ . It is delivered on a burst mode.

The message delivered to the encoder will have a fixed size of 388 information bits  $\{d(0),d(1),...,d(387)\}$ , if an eTFI field is included (see 3GPP TS 44.060).

### 5.1.8.1.2 USF precoding

### 5.1.8.1.2.1 BTTI configuration

The first three bits d(0),d(1),d(2) are block coded into twelve bits u'(0),u'(1),...,u'(11) as for Packet data block type 4 (CS-4) in subclause 5.1.4.2.

#### 5.1.8.1.2.2 RTTI configuration

Twelve bits u'(0),u'(1),...,u'(11) are generated as described for MCS-1 DL in subclause 5.1.5.1.2.2.

### 5.1.8.1.3 Header coding

A block of 68 coded bits  $\{hc(0),hc(1),...,hc(67)\}$  is derived from  $\{d(3),d(4),...,d(30)\}$  as described for MCS-1 DL in subclause 5.1.5.1.3, with the exception that eTFI field bits are now located in  $\{d(385),d(386),d(387)\}$  if included.

#### 5.1.8.1.4 Data coding

a) Parity bits:

Twelve data parity bits p(0),p(1),...,p(11) are defined in such a way that in GF(2) the binary polynomial:

$$d(31)D^{365} + ... + d(384)D^{12} + p(0)D^{11} + ... + p(11)$$
, when divided by:

$$D^{12} + D^{11} + D^{10} + D^8 + D^5 + D^4 + 1$$
, yields a remainder equal to:

$$D^{11} + D^{10} + D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1.$$

b) Tail bits:

Six tail bits equal to 0 are added to the information and parity bits, the result being a block of 372 bits  $\{u(0),u(1),...,u(371)\}$ :

$$u(k) = d(k+31)$$
 for  $k = 0,1,...,353$ 

$$u(k) = p(k-354)$$
 for  $k = 354,355,...,365$ 

$$u(k) = 0$$
 for  $k = 366, 367, ..., 371$  (tail bits)

c) Convolutional encoder

This block of 372 bits  $\{u(0),u(1),...,u(371)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 1116 coded bits:  $\{C(0),C(1),...,C(1115)\}\$  defined by:

$$C(3k) = u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6)$$

$$C(3k+1) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-6)$$

$$C(3k+2) = u(k) + u(k-1) + u(k-4) + u(k-6)$$
 for  $k = 0,1,..., 371$ ;  $u(k) = 0$  for  $k < 0$ 

The code is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied in such a way that the following coded bits:

P1	$\{C(3j) \text{ for } j = 0,1,,371\}$ are transmitted
P2	$\{C(1+3j) \text{ for } j = 0,1,,371\}$ are transmitted
P3	$\{C(2+3j) \text{ for } j = 0,1,,371\}$ are transmitted

The result is a block of 372 coded bits,  $\{dc(0),dc(1),...,dc(371)\}$ .

### 5.1.8.1.5 Interleaving

The interleaving is done as specified for MCS-1 DL in subclause 5.1.5.1.5.

#### 5.1.8.1.6 Mapping on a burst

The mapping is done as specified for MCS-1 DL in subclause 5.1.5.1.6.

## 5.1.8.2 Uplink (MCS-4 UL)

### 5.1.8.2.1 Block constitution

The message delivered to the encoder has a fixed size of 385 information bits  $\{d(0),d(1),...,d(384)\}$ . It is delivered on a burst mode.

## 5.1.8.2.2 Header coding

A block of 80 coded bits  $\{hc(0),hc(1),...,hc(79)\}$  is derived from  $\{d(0),d(1),...,d(30)\}$  as described for MCS-1 UL in subclause 5.1.5.2.2.

#### 5.1.8.2.3 Data coding

The data coding is the same as for downlink as specified in subclause 5.1.8.1.4.

## 5.1.8.2.4 Interleaving

The interleaving is the same as for MCS-1 UL as specified in subclause 5.1.5.2.4.

## 5.1.8.2.5 Mapping on a burst

The mapping is the same as for MCS-1 UL as specified in subclause 5.1.5. 2.5.

# 5.1.9 Packet data block type 9 (MCS-5)

## 5.1.9.1 Downlink (MCS-5 DL)

#### 5.1.9.1.1 Block constitution

The message delivered to the encoder has a fixed size of 478 information bits  $\{d(0),d(1),...,d(477)\}$ . It is delivered on a burst mode.

The message delivered to the encoder will have a fixed size of 481 information bits  $\{d(0),d(1),...,d(480)\}$ , if an eTFI field is included (see 3GPP TS 44.060).

The message delivered to the encoder will have a fixed size of 503 information bits  $\{d(0),d(1),...,d(502)\}$  if a PAN field is included.

The message delivered to the encoder will have a fixed size of 506 information bits  $\{d(0),d(1),...,d(505)\}$ , if a PAN field and an eTFI field are included (see 3GPP TS 44.060).

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

## 5.1.9.1.2 USF precoding

### 5.1.9.1.2.1 BTTI configuration

The first three bits d(0),d(1),d(2) are block coded into 36 bits u'(0),u'(1),...,u'(35) according to the following table:

d(0),d(1),d(2)	u'(0),u'(1),,u'(35)					
	burst 0	burst 1	burst 2	burst 3		
000	00000000	000000000	000000000	000000000		
001	111110000	111100000	111111000	111110001		
010	111001110	111011100	110000110	110001100		
011	100111100	110000011	101110111	001001111		
100	000110011	001011010	100001101	111111110		
101	110101011	000110101	011101011	100101011		
110	001001101	101111111	011010001	001110100		
111	011010111	010101111	000111110	010010011		

#### 5.1.9.1.2.2 RTTI configurations

If the USF is sent in RTTI USF mode (see 3GPP TS 45.002) when data blocks are transmitted in RTTI configuration, then the first three bits d(0),d(1),d(2) are block coded into 36 bits u'(0),u'(1),...,u'(35) as described in subclause 5.1.9.1.2.1.

If the USF is sent in BTTI USF mode (see 3GPP TS 45.002) when data blocks are transmitted in RTTI configuration, then the three bits of the USF to be sent on the lower numbered PDCH of a corresponding downlink PDCH-pair are block coded into 36 bits  $u_L(0)$ ,  $u_L(1)$ ,..., $u_L(35)$  as described in subclause 5.1.9.1.2.1; the three bits of the USF to be sent on the higher numbered PDCH of a corresponding downlink PDCH-pair are block coded into 36 bits  $u_H(0)$ ,  $u_H(1)$ ,..., $u_H(35)$  as described in subclause 5.1.9.1.2.1.

NOTE: If BTTI USF mode is used when sending data blocks in RTTI configuration, then d(0),d(1),d(2) need not contain a USF; in this case, they are ignored by the encoder. How the USFs are delivered to the encoder in this case is implementation dependent.

If the data block is sent in the first 10ms of a 20ms block period, then:

$$u'(j)=u_L(j), j=0...8$$

$$u'(j)=u_H(j-9), j=9...17$$

$$u'(j)=u_L(j-9), j=18...26$$

$$u'(j)=u_H(j-18)$$
  $j=27...35$ 

If the data block is sent in the second 10ms of a 20ms block period, then:

$$u'(j)=u_L(j+18), j=0...8$$

$$u'(j)=u_H(j+9), j=9...17$$

$$u'(j)=u_L(j+9), j=18...26$$

$$u'(j)=u_H(j)$$
  $j=27...35$ 

NOTE: In case mixed modulation USF is used (see subclause 5.1), the USF bits sent during the other half of the 20 ms block period may be sent with a different modulation. In this case, the half of  $u_L$  and  $u_H$  not sent in the present data block will be discarded.

## 5.1.9.1.3 Header coding

a) Parity bits:

Eight header parity bits p(0),p(1),...,p(7) are defined in such a way that in GF(2) the binary polynomial:

$$d(3)D^{32} + ... + d(27)D^8 + p(0)D^7 + ... + p(7)$$
, when divided by:  
 $D^8 + D^6 + D^3 + 1$ , yields a remainder equal to:  
 $D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$ .

In case an eTFI field (see 3GPP TS 44.060) is included and a PAN field is not included the three bits  $\{d(478),d(479),d(480)\}$  are added bit-wise modulo 2 to the last three parity bits  $\{p(5),p(6),p(7)\}$ . In case an eTFI field and a PAN field are included the three bits  $\{d(503),d(504),d(505)\}$  are added bit-wise modulo 2 to the last three parity bits  $\{p(5),p(6),p(7)\}$ . This result in eight modified header parity bits  $\{p(0),...,pt(7)\}$  as specified in subclause 5.1.5.1.3.

#### b) Tail biting:

The six last header parity bits are added before information and parity bits, the result being a block of 39 bits  $\{u"(-6),...,u"(0),u"(1),...,u"(32)\}$  with six negative indexes. In case an eTFI field is not included this operation is defined as:

$$u"(k-6) = p(k+2)$$
 for  $k = 0,1,...,5$   
 $u"(k) = d(k+3)$  for  $k = 0,1,...,24$   
 $u"(k) = p(k-25)$  for  $k = 25,26,...,32$ 

In case an eTFI field is included p(k) is replaced by pt(k) in the above operation.

#### c) Convolutional encoder

This block of 39 bits  $\{u''(-6),...,u''(0),u''(1),...,u''(32)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^{2} + D^{3} + D^{5} + D^{6}$$

$$G7 = 1 + D + D^{2} + D^{3} + D^{6}$$

$$G5 = 1 + D + D^{4} + D^{6}$$

This results in a block of 99 coded bits:  $\{C(0),C(1),...,C(98)\}$  defined by:

$$C(3k) = u''(k) + u''(k-2) + u''(k-3) + u''(k-5) + u''(k-6)$$

$$C(3k+1) = u''(k) + u''(k-1) + u''(k-2) + u''(k-3) + u''(k-6)$$

$$C(3k+2) = u''(k) + u''(k-1) + u''(k-4) + u''(k-6) \quad \text{for } k = 0,1,...,32$$

A spare bit is added at the end of this block:

$$hc(k) = C(k)$$
 for  $k = 0,1,...,98$   
 $hc(99) = C(98)$ 

The result is a block of 100 coded bits,  $\{hc(0),hc(1),...,hc(99)\}$ .

## 5.1.9.1.4 Data coding

### a) Parity bits:

Twelve data parity bits p(0), p(1), ..., p(11) are defined in such a way that in GF(2) the binary polynomial:

$$d(28)D^{461} + ... + d(477)D^{12} + p(0)D^{11} + ... + p(11), \text{ when divided by:}$$
 
$$D^{12} + D^{11} + D^{10} + D^8 + D^5 + D^4 + 1, \text{ yields a remainder equal to:}$$
 
$$D^{11} + D^{10} + D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1.$$

#### b) Tail bits:

Six tail bits equal to 0 are added to the information and parity bits, the result being a block of 468 bits  $\{u(0),u(1),...,u(467)\}$ :

$$\begin{array}{ll} u(k) &= d(k+28) & \text{ for } k=0,1,...,449 \\ \\ u(k) &= p(k-450) & \text{ for } k=450,451,...,461 \\ \\ u(k) &= 0 & \text{ for } k=462,463,...,467 \text{ (tail bits)} \end{array}$$

#### c) Convolutional encoder

This block of 468 bits  $\{u(0),u(1),...,u(467)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^{2} + D^{3} + D^{5} + D^{6}$$

$$G7 = 1 + D + D^{2} + D^{3} + D^{6}$$

$$G5 = 1 + D + D^{4} + D^{6}$$

This results in a block of 1404 coded bits:  $\{C(0),C(1),...,C(1403)\}\$  defined by:

$$\begin{split} &C(3k) &= u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6) \\ &C(3k+1) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-6) \\ &C(3k+2) = u(k) + u(k-1) + u(k-4) + u(k-6) \quad \text{ for } k = 0,1,...,467; \ u(k) = 0 \text{ for } k < 0 \end{split}$$

The code is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied in such a way that the following coded bits:

P1	$\{C(2+9j) \text{ for } j = 0,1,,153\}$ as well as $\{C(1388+3j) \text{ for } j = 0,1,,5\}$ are not transmitted
	except $\{C(k) \text{ for } k = 47,371,695,1019\}$ which are transmitted
P2	$\{C(1+9j) \text{ for } j = 0,1,,153\}$ as well as $\{C(1387+3j) \text{ for } j = 0,1,,5\}$ are not transmitted
	except {C(k) for k = 136,460,784,1108} which are transmitted

The result is a block of 1248 coded bits, {dc(0),dc(1),...,dc(1247)}.

For the FANR procedure, the code is punctured depending on the CPS field and the PANI field as defined in 3GPP TS 44.060. If the PANI field is set to 0, the puncturing is the same as for EGPRS. If the PANI field is set to 1, the puncturing schemes named P1 or P2 are applied in such a way that, in addition to the bits punctured for EGPRS, the following coded bits:

P1	$\{C(4+18j) \text{ for } j=0,1,,76\}$ are not transmitted except $\{C(k) \text{ for } k=526\}$ which is transmitted
P2	$\{C(14+18j) \text{ for } j=0,1,,76\}$ are not transmitted except $\{C(k) \text{ for } k=626\}$ which is transmitted

The result is a block of 1172 coded bits  $\{pc(0),pc(1),...,pc(1171)\}$ .

## 5.1.9.1.4a Piggy-backed Ack/Nack coding

The operations in this subclause shall be carried out only if a PAN field is included.

#### a) Parity bits

Ten PAN parity bits p(0), p(1),...,p(9) are defined in such a way that in GF(2) the binary polynomial:

$$d(478)D^{29} + ... + d(497)D^{10} + p(0)D^9 + ... + p(9)$$
, when divided by:

 $D^{10} + D^9 + D^5 + D^4 + D + 1$ , yields a remainder equal to:

$$D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D^1 + 1$$
.

The five bits  $\{d(498),...,d(502)\}\$  (TFI value or 00000, see 3GPP TS 44.060) are added bit-wise modulo 2 to the 5 last parity bits  $\{p(5),...,p(9)\}\$ . This results in the ten modified PAN parity bits  $\{p(0),...,p(9)\}\$  defined as:

$$pt(k) = p(k)$$
 for k=0,...,4  
 $pt(k) = d(k+493) + p(k)$  for k=5,...,9

#### b) Tail biting:

The six last modified PAN parity bits are added before information and modified PAN parity bits, the result being a block of 36 {u''(-6),...,u''(0),u''(1),...,u''(29)} bits with six negative indexes:

$$u''(k-6) = pt(k+4)$$
 for  $k = 0,1,...,5$   
 $u''(k) = d(k+478)$  for  $k = 0,1,...,19$   
 $u''(k) = pt(k-20)$  for  $k = 20,21,...,29$ 

#### c) Convolutional encoder

The block of 36 bits  $\{u''(-6),...,u''(0),u''(1),...,u''(29)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^{2} + D^{3} + D^{5} + D^{6}$$

$$G7 = 1 + D + D^{2} + D^{3} + D^{6}$$

$$G5 = 1 + D + D^{4} + D^{6}$$

This results in a block of 90 coded bits  $\{C(0),C(1),...,C(89)\}$  defined by:

$$\begin{split} &C(3k) &= u"(k) + u"(k-2) + u"(k-3) + u"(k-5) + u"(k-6) \\ &C(3k+1) = u"(k) + u"(k-1) + u"(k-2) + u"(k-3) + u"(k-6) \\ &C(3k+2) = u"(k) + u"(k-1) + u"(k-4) + u"(k-6) & \text{for } k = 0,1,...,29 \end{split}$$

The block of 90 coded bits is punctured in such way that the following coded bits:

$$\{C(5+6k), C(50+6k) \text{ for } k = 0,1,...,6\}$$
 are not transmitted.

The result is a block of 76 coded bits  $\{ac(0),ac(1),...,ac(75)\}$ .

The data coded bits  $\{pc(0),pc(1),...,pc(1171)\}$  are appended to the PAN coded bits by the following rule:

$$dc(k) = ac(k)$$
 for  $k = 0,1,...,75$   
 $dc(k) = pc(k-76)$  for  $k = 76,49,...,1247$ 

The result is a block of 1248 coded bits  $\{dc(0),dc(1),...,dc(1247)\}$ .

### 5.1.9.1.5 Interleaving

#### a) Header

The 100 coded bits of the header, {hc(0),hc(1),...,hc(99)}, are interleaved according to the following rule:

$$hi(j) = hc(k)$$
 for  $k = 0,1,...,99$   
 $j = 25(k \mod 4) + ((17k) \mod 25)$ 

#### b) Data

There is no closed expression describing the interleaver, but it has been derived taking the following approach:

1. A block interleaver with a 1392 bit block size is defined:

The kth input data bit is mapped to the jth bit of the Bth burst, where

```
\begin{split} k &= 0, \dots, 1391 \\ B &= mod(k, 4) \\ d &= mod(k, 464) \\ j &= 3*(2mod(25d, 58) + div(mod(d, 8), 4) + 2(-1)^B div(d, 232)) + mod(k, 3) \end{split}
```

- 2. The data bit positions being mapped onto header positions in the interleaved block are removed (the header positions are j = 156,157,...,191 when the header is placed next to the training sequence. This leaves 1248 bits in the mapping.
- 3. The bits are renumbered to fill out the gaps both in j and k, without changing the relative order

The resulting interleaver transform the block of 1248 coded bits,  $\{dc(0),dc(1),...,dc(1247)\}$  into a block of 1248 interleaved bits,  $\{di(0),di(1),...,di(1247)\}$ .

$$di(j') = dc(k')$$
 for  $k' = 0,1,...,1247$ 

(An explicit relation between j' and k' is given in table 15)

### 5.1.9.1.6 Mapping on a burst

a) Straightforward Mapping

The mapping is given by the rule:

$$\begin{array}{lll} e(B,j) = di(312B+j) & \text{for } j = 0,1,...,155 \\ e(B,j) = hi(25B+j-156) & \text{for } j = 156,157,...,167 \\ e(B,j) = u'(9B+j-168) & \text{for } j = 168,169,...,173 \\ e(B,j) = q(2B+j-174) & \text{for } j = 174,175 \\ e(B,j) = u'(9B+j-170) & \text{for } j = 176,177,178 \\ e(B,j) = hi(25B+j-167) & \text{for } j = 179,180,...,191 \\ e(B,j) = di(312B+j-36) & \text{for } j = 192,193,...,347 \\ \end{array}$$

where

 $\label{eq:q0} q(0), q(1), ..., q(7) = 0, 0, 0, 0, 0, 0, 0, 0, 0 \ identifies \ the \ coding \ scheme \ MCS-5 \ or \ MCS-6.$ 

b) Bit swapping

After this mapping the following bits are swapped:

Swap 
$$e(B,148)$$
 with  $e(B,167)$ 

Swap e(B,150) with e(B,170)

```
Swap e(B,151) with e(B,173)
Swap e(B,176) with e(B,195)
Swap e(B,179) with e(B,196)
Swap e(B,182) with e(B,198)
Swap e(B,185) with e(B,199)
Swap e(B,188) with e(B,201)
Swap e(B,191) with e(B,202)
Swap e(B,194) with e(B,204).
```

In RTTI configuration, the bursts with B = 0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B = 1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.

## c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

```
For B = 0
    Swap e(B,50) with e(B,49)
    Swap e(B,113) with e(B,112)
    Swap e(B,167) with e(B,154)
    Swap e(B,221) with e(B,220)
    Swap e(B,278) with e(B,277)
    Swap e(B,341) with e(B,340)
 For B = 1
    Swap e(B,8) with e(B,7)
    Swap e(B,59) with e(B,58)
    Swap e(B,71) with e(B,70)
    Swap e(B,116) with e(B,115)
    Swap e(B,173) with e(B,154)
    Swap e(B,182) with e(B,193)
    Swap e(B,236) with e(B,235)
    Swap e(B,299) with e(B,298)
For B = 2
    Swap e(B,17) with e(B,16)
    Swap e(B,74) with e(B,73)
    Swap e(B,137) with e(B,136)
    Swap e(B,257) with e(B,256)
```

Swap e(B,302) with e(B,301)

Swap e(B,314) with e(B,313)

For B = 3

Swap e(B,32) with e(B,31)

Swap e(B,95) with e(B,94)

Swap e(B,152) with e(B,154)

Swap e(B,215) with e(B,214)

Swap e(B,260) with e(B,259)

Swap e(B,272) with e(B,271)

Swap e(B,323) with e(B,322)

# 5.1.9.2 Uplink (MCS-5 UL)

#### 5.1.9.2.1 Block constitution

The message delivered to the encoder has a fixed size of 487 information bits  $\{d(0),d(1),...,d(486)\}$ . It is delivered on a burst mode.

The message delivered to the encoder will have a fixed size of 512 information bits {d(0),d(1),...,d(511)} if a PAN field is included (see 3GPP TS 44.060).

The message delivered to the encoder will have a fixed size of 515 information bits  $\{d(0),d(1),...,d(514)\}$ , if a PAN field and an eTFI field are included (see 3GPP TS 44.060).

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

## 5.1.9.2.2 Header coding

a) Parity bits:

Eight header parity bits p(0),p(1),...,p(7) are defined in such a way that in GF(2) the binary polynomial:

$$d(0)D^{44} + ... + d(36)D^{8} + p(0)D^{7} + ... + p(7)$$
, when divided by:

 $D^8 + D^6 + D^3 + 1$ , yields a remainder equal to:

$$D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$$
.

b) Tail biting:

The six last header parity bits are added before information and parity bits, the result being a block of 51 bits  $\{u"(-6),...,u"(0),u"(1),...,u"(44)\}$  with six negative indexes:

$$u''(k-6) = p(k+2)$$
 for  $k = 0,1,...,5$ 

$$u''(k) = d(k)$$
 for  $k = 0,1,...,36$ 

$$u''(k) = p(k-37)$$
 for  $k = 37,38,...,44$ 

c) Convolutional encoder

This block of 51 bits  $\{u''(-6),...,u''(0),u''(1),...,u''(44)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 135 coded bits:  $\{C(0),C(1),...,C(134)\}\$  defined by:

$$C(3k) = u''(k) + u''(k-2) + u''(k-3) + u''(k-5) + u''(k-6)$$

$$C(3k+1) = u''(k) + u''(k-1) + u''(k-2) + u''(k-3) + u''(k-6)$$

$$C(3k+2) = u''(k) + u''(k-1) + u''(k-4) + u''(k-6)$$
 for  $k = 0,1,...,44$ 

The code is punctured in such a way that the following coded bits:

$$hc(k) = C(k)$$
 for  $k = 0,1,...,134$   
 $hc(135) = C(134)$ 

The result is a block of 136 coded bits,  $\{hc(0),hc(1),...,hc(135)\}$ .

## 5.1.9.2.3 Data coding

The data coding is the same as for downlink as specified in subclause 5.1.9.1.4 where bits  $\{d(28),d(29),...,d(477)\}$  are replaced by bits  $\{d(37),d(38),...,d(486)\}$ .

### 5.1.9.2.3a Piggy-backed Ack/Nack coding

If a PAN field is included and an eTFI field is not included, the coding of the PAN field is the same as for the downlink as specified in subclause 5.1.9.1.4a where bits  $\{d(478), d(479),...,d(502)\}$  are replaced by bits  $\{d(487), d(488),...,d(511)\}$ .

If a PAN field and an eTFI field are included, the PAN coding is the same as for the downlink as specified in sub-clause 5.1.9.1.4a, with the exception that the five bits  $\{d(507),d(508),d(509),d(510),d(511)\}$  (TFI value or 00000, see 3GPP TS 44.060) and the three bits  $\{d(512),d(513),d(514)\}$  (eTFI value or 000, see 3GPP TS 44.060) are added bit-wise modulo 2 to the PAN parity bits  $\{p(2),p(3),p(4),p(5),p(6),p(7),p(8),p(9)\}$  resulting in the ten modified PAN parity bits defined as:

$$pt(k) = p(k)$$
 for k=0,...,1  
 $pt(k) = d(k+510) + p(k)$  for k=2,...,4  
 $pt(k) = d(k+502) + p(k)$  for k=5,...,9

The data coded bits  $\{pc(0),pc(1),...,pc(1171)\}$  are appended to the PAN coded bits as described for the downlink in subclause 5.1.9.1.4a. The result is a block of 1248 coded bits  $\{dc(0),dc(1),...,dc(1247)\}$ .

#### 5.1.9.2.4 Interleaving

a) Header

The 136 coded bits of the header, {hc(0),hc(1),...,hc(135)}, are interleaved according to the following rule:

$$hi(j) = hc(k)$$
 for  $k = 0,1,...,135$  
$$j = 34(k \mod 4) + 2((11k) \mod 17) + [(k \mod 8)/4]$$

b) Data

The data interleaving is the same as for MCS-5 DL as specified in subclause 5.1.9.1.5.

### 5.1.9.2.5 Mapping on a burst

a) Straightforward Mapping

The mapping is given by the rule:

$$e(B,j) = di(312B+j)$$
 for  $j = 0,1,...,155$ 

$$e(B,j) = hi(34B+j-156)$$
 for  $j = 156,157,...,173$ 

$$e(B,j) = q(2B+j-174)$$
 for  $j = 174,175$ 

$$e(B,j) = hi(34B+j-158)$$
 for  $j = 176,177,...,191$ 

$$e(B,j) = di(312B+j-36)$$
 for  $j = 192,193,...,347$ 

where

q(0),q(1),...,q(7) = 0,0,0,0,0,0,0,0,0 identifies the coding scheme MCS-5 or MCS-6.

#### b) Bit swapping

The bit swapping is the same as for MCS-5 DL as specified in subclause 5.1.9.1.6 b).

In RTTI configuration, the bursts with B = 0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B = 1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.

## c) PAN bit swapping

In case a PAN is included in the radio block, additional bits are swapped as specified in subclause 5.1.9.1.6 c).

# 5.1.10 Packet data block type 10 (MCS-6)

## 5.1.10.1 Downlink (MCS-6 DL)

#### 5.1.10.1.1 Block constitution

The message delivered to the encoder has a fixed size of 622 information bits  $\{d(0),d(1),...,d(621)\}$ . It is delivered on a burst mode.

The message delivered to the encoder will have a fixed size of 625 information bits {d(0),d(1),...,d(624)}, if an eTFI field is included (see 3GPP TS 44.060).

The message delivered to the encoder will have a fixed size of 647 information bits {d(0),d(1),...,d(646)} if a PAN field is included.

The message delivered to the encoder will have a fixed size of 650 information bits  $\{d(0),d(1),...,d(649)\}$ , if a PAN field and an eTFI field are included (see 3GPP TS 44.060).

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

## 5.1.10.1.2 USF precoding

#### 5.1.10.1.2.1 BTTI configuration

A block of 36 bits  $\{u'(0), u'(1), ..., u'(35)\}$  is derived from  $\{d(0), d(1), d(2)\}$  as described for MCS-5 DL in subclause 5.1.9.1.2. 1.

## 5.1.10.1.2.2 RTTI configuration

A block of 36 bits {u'(0),u'(1),...,u'(35)} is generated as described for MCS-5 DL in subclause 5.1.9.1.2.2.

#### 5.1.10.1.3 Header coding

A block of 100 coded bits  $\{hc(0),hc(1),...,hc(99)\}$  is derived from  $\{d(3),d(4),...,d(27)\}$  as described for MCS-5 DL in subclause 5.1.9.1.3, with the exception that eTFI field bits are now located in  $\{d(622),d(623),d(624)\}$  if included when PAN is not included, and located in  $\{d(647),d(648),d(649)\}$  if included in combination with a PAN field.

## 5.1.10.1.4 Data coding

#### a) Parity bits:

Twelve data parity bits p(0), p(1), ..., p(11) are defined in such a way that in GF(2) the binary polynomial:

$$d(28)D^{605} + ... + d(621)D^{12} + p(0)D^{11} + ... + p(11), \text{ when divided by:}$$
 
$$D^{12} + D^{11} + D^{10} + D^8 + D^5 + D^4 + 1, \text{ yields a remainder equal to:}$$
 
$$D^{11} + D^{10} + D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1.$$

### b) Tail bits:

Six tail bits equal to 0 are added to the information and parity bits, the result being a block of 612 bits  $\{u(0),u(1),...,u(611)\}$ :

$$\begin{array}{ll} u(k) &= d(k{+}28) & \text{ for } k = 0,1,...,593 \\ \\ u(k) &= p(k{-}594) & \text{ for } k = 594,595,...,605 \\ \\ u(k) &= 0 & \text{ for } k = 606,607,...,611 \text{ (tail bits)} \end{array}$$

#### c) Convolutional encoder

This block of 612 bits  $\{u(0),u(1),...,u(611)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^{2} + D^{3} + D^{5} + D^{6}$$

$$G7 = 1 + D + D^{2} + D^{3} + D^{6}$$

$$G5 = 1 + D + D^{4} + D^{6}$$

This results in a block of 1836 coded bits:  $\{C(0),C(1),...,C(1835)\}\$  defined by:

$$\begin{split} &C(3k) &= u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6) \\ &C(3k+1) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-6) \\ &C(3k+2) = u(k) + u(k-1) + u(k-4) + u(k-6) \quad \text{ for } k = 0,1,...,611; \ u(k) = 0 \text{ for } k < 0 \end{split}$$

The code is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied in such a way that the following coded bits:

P1	$ \{C(2+3j) \text{ for } j=0,1,,611\} \text{ are not transmitted except } \{C(k) \text{ for } k=32,98,164,230,296,428,494,560,} \\ 626,692,824,890,956,1022,1088,1220,1286,1352,1418,1484,1616,1682,1748,1814} \text{ which are transmitted} $
P2	$ \begin{cases} \text{C(1+3j) for } \text{j} = 0,1,,611 \} \text{ are not transmitted except } \{\text{C(k) for k} = 16,82,148,214,280,412,478,544}, \\ 610,676,808,874,940,1006,1072,1204,1270,1336,1402,1468,1600,1666,1732,1798} \text{ which are transmitted} $

The result is a block of 1248 coded bits,  $\{dc(0),dc(1),...,dc(1247)\}$ .

For the FANR procedure, the code is punctured depending on the CPS field and the PANI field as defined in 3GPP TS 44.060. If the PANI field is set to 0, the puncturing is the same as for EGPRS. If the PANI field is set to 1, the puncturing schemes named P1 or P2 are applied in such a way that, in addition to the bits punctured for EGPRS, the following coded bits:

P1	$\{C(6+24j) \text{ for } j=0,1,,75\}$ are not transmitted
P2	$\{C(18+24j) \text{ for } j=0,1,,75\}$ are not transmitted

The result is a block of 1172 coded bits  $\{pc(0),pc(1),...,pc(1171)\}$ .

## 5.1.10.1.4a Piggy-backed Ack/Nack coding

The operations in this subclause shall be carried out only if a PAN field is included.

A block of 76 coded bits  $\{ac(0),ac(1),...,ac(75)\}\$  is derived from  $\{(622),d(623),...,d(646)\}\$  as described for MCS-5 DL in subclause 5.1.9.1.4a, with bits  $\{d(478),d(479),...,d(502)\}\$  replaced by bits  $\{d(622),d(623),...,d(646)\}\$ .

The data coded bits  $\{pc(0),pc(1),...,pc(1171)\}$  are appended to the PAN coded bits as described for MCS-5 DL in subclause 5.1.9.1.4a. The result is a block of 1248 coded bits  $\{dc(0),dc(1),...,dc(1247)\}$ .

## 5.1.10.1.5 Interleaving

The interleaving is done as specified for MCS-5 DL in subclause 5.1.9.1.5.

### 5.1.10.1.6 Mapping on a burst

The mapping is done as specified for MCS-5 DL in subclause 5.1.9.1.6.

## 5.1.10.2 Uplink (MCS-6 UL)

#### 5.1.10.2.1 Block constitution

The message delivered to the encoder has a fixed size of 631 information bits  $\{d(0),d(1),...,d(630)\}$ . It is delivered on a burst mode.

The message delivered to the encoder will have a fixed size of 656 information bits  $\{d(0),d(1),...,d(655)\}\$  if a PAN field is included (see 3GPP TS 44.060).

The message delivered to the encoder will have a fixed size of 659 information bits  $\{d(0),d(1),...,d(658)\}$ , if a PAN field and an eTFI field are included (see 3GPP TS 44.060).

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

### 5.1.10.2.2 Header coding

A block of 136 coded bits  $\{hc(0),hc(1),...,hc(135)\}$  is derived from  $\{d(0),d(1),...,d(36)\}$  as described for MCS-5 UL in subclause 5.1.9.2.2.

### 5.1.10.2.3 Data coding

The data coding is the same as for downlink as specified in subclause 5.1.10.1.4 where bits  $\{d(28),d(29),...,d(621)\}$  are replaced by bits  $\{d(37),d(38),...,d(630)\}$ .

### 5.1.10.2.3a Piggy-backed Ack/Nack coding

If a PAN field is included and an eTFI field is not included, the coding of the PAN field is the same as for the MCS-5 DL as specified in subclause 5.1.9.1.4a where bits {d(478), d(479),...,d(502)} are replaced by bits {d(631),d(632),...,d(655)}.

If a PAN field and an eTFI field are included, the PAN coding is the same as for the downlink as specified in sub-clause 5.1.10.1.4a, with the exception that the five bits  $\{d(651),d(652),d(653),d(654),d(655)\}$  (TFI value or 00000, see 3GPP TS 44.060) and the three bits  $\{d(656),d(657),d(658)\}$  (eTFI value or 000, see 3GPP TS 44.060) are added bit-wise

modulo 2 to the PAN parity bits  $\{p(2),p(3),p(4),p(5),p(6),p(7),p(8),p(9)\}$  resulting in the ten modified PAN parity bits defined as:

$$pt(k) = p(k)$$
 for  $k=0,...,1$   
 $pt(k) = d(k+654) + p(k)$  for  $k=2,...,4$   
 $pt(k) = d(k+646) + p(k)$  for  $k=5,...,9$ 

The data coded bits  $\{pc(0),pc(1),...,pc(1171)\}$  are appended to the PAN coded bits as described for MCS-5 DL in subclause 5.1.9.1.4a. The result is a block of 1248 coded bits  $\{dc(0),dc(1),...,dc(1247)\}$ .

### 5.1.10.2.4 Interleaving

The interleaving is the same as for MCS-5 UL as specified in subclause 5.1.9.2.4.

## 5.1.10.2.5 Mapping on a burst

The mapping is the same as for MCS-5 UL as specified in subclause 5.1.9.2.5.

# 5.1.11 Packet data block type 11 (MCS-7)

## 5.1.11.1 Downlink (MCS-7 DL)

#### 5.1.11.1.1 Block constitution

The message delivered to the encoder has a fixed size of 940 information bits {d(0),d(1),...,d(939)}. It is delivered on a burst mode.

The message delivered to the encoder will have a fixed size of 943 information bits  $\{d(0),d(1),...,d(942)\}$ , if an eTFI field is included (see 3GPP TS 44.060).

The message delivered to the encoder will have a fixed size of 965 information bits  $\{d(0),d(1),...,d(964)\}$  if a PAN field is included.

The message delivered to the encoder will have a fixed size of 968 information bits  $\{d(0),d(1),...,d(967)\}$ , if a PAN field and an eTFI field are included (see 3GPP TS 44.060).

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

## 5.1.11.1.2 USF precoding

## 5.1.11.1.2.1 BTTI configuration

A block of 36 bits  $\{u'(0), u'(1), ..., u'(35)\}$  is derived from  $\{d(0), d(1), d(2)\}$  as described for MCS-5 DL in subclause 5.1.9.1.2.1.

## 5.1.11.1.2.2 RTTI configuration

A block of 36 bits {u'(0),u'(1),...,u'(35)} is generated as described for MCS-5 DL in subclause 5.1.9.1.2.2.

#### 5.1.11.1.3 Header coding

a) Parity bits:

Eight header parity bits p(0),p(1),...,p(7) are defined in such a way that in GF(2) the binary polynomial:

$$d(3)D^{44} + ... + d(39)D^{8} + p(0)D^{7} + ... + p(7)$$
, when divided by:

 $D^8 + D^6 + D^3 + 1$ , yields a remainder equal to:

$$D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$$
.

In case an eTFI field (see 3GPP TS 44.060) is included and a PAN field is not included the three bits  $\{d(940),d(941),d(942)\}$  are added bit-wise modulo 2 to the last three parity bits  $\{p(5),p(6),p(7)\}$ . In case an eTFI field and a PAN field are included the three bits  $\{d(965),d(966),d(967)\}$  are added bit-wise modulo 2 to the last three parity bits  $\{p(5),p(6),p(7)\}$ . This result in eight modified header parity bits  $\{p(0),...,p(7)\}$  as specified in subclause 5.1.5.1.3.

#### b) Tail biting:

The six last header parity bits are added before information and parity bits, the result being a block of 51 bits  $\{u"(-6),...,u"(0),u"(1),...,u"(44)\}$  with six negative indexes. In case an eTFI field is not included this operation is defined as:

$$u''(k-6) = p(k+2)$$
 for  $k = 0,1,...,5$   
 $u''(k) = d(k+3)$  for  $k = 0,1,...,36$   
 $u''(k) = p(k-37)$  for  $k = 37,38,...,44$ 

In case an eTFI field is included p(k) is replaced by pt(k) in the above operation.

### c) Convolutional encoder

This block of 51 bits  $\{u''(-6),...,u''(0),u''(1),...,u''(44)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^{2} + D^{3} + D^{5} + D^{6}$$

$$G7 = 1 + D + D^{2} + D^{3} + D^{6}$$

$$G5 = 1 + D + D^{4} + D^{6}$$

This results in a block of 135 coded bits:  $\{C(0),C(1),...,C(134)\}\$  defined by:

$$\begin{split} &C(3k) &= u"(k) + u"(k-2) + u"(k-3) + u"(k-5) + u"(k-6) \\ &C(3k+1) = u"(k) + u"(k-1) + u"(k-2) + u"(k-3) + u"(k-6) \\ &C(3k+2) = u"(k) + u"(k-1) + u"(k-4) + u"(k-6) & \text{for } k = 0,1,...,44 \end{split}$$

The code is punctured in such a way that the following coded bits:

$$\{C(k) \text{ for } k = 14,23,33,50,59,69,86,95,105,122,131\}$$
 are not transmitted

The result is a block of 124 coded bits,  $\{hc(0),hc(1),...,hc(123)\}$ .

### 5.1.11.1.4 Data coding

- I) First half:
- a) Parity bits:

Twelve data parity bits p(0), p(1), ..., p(11) are defined in such a way that in GF(2) the binary polynomial:

$$d(40)D^{461} + ... + d(489)D^{12} + p(0)D^{11} + ... + p(11)$$
, when divided by:  
 $D^{12} + D^{11} + D^{10} + D^8 + D^5 + D^4 + 1$ , yields a remainder equal to:  
 $D^{11} + D^{10} + D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$ .

#### b) Tail bits:

Six tail bits equal to 0 are added to the information and parity bits, the result being a block of 468 bits  $\{u(0),u(1),...,u(467)\}$ :

$$u(k) = d(k+40)$$
 for  $k = 0,1,...,449$ 

$$u(k) = p(k-450)$$
 for  $k = 450,451,...,461$   
 $u(k) = 0$  for  $k = 462,463,...,467$  (tail bits)

#### c) Convolutional encoder

This block of 468 bits  $\{u(0),u(1),...,u(467)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^{2} + D^{3} + D^{5} + D^{6}$$

$$G7 = 1 + D + D^{2} + D^{3} + D^{6}$$

$$G5 = 1 + D + D^{4} + D^{6}$$

This results in a block of 1404 coded bits:  $\{C(0),C(1),...,C(1403)\}\$  defined by:

$$\begin{split} &C(3k) &= u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6) \\ &C(3k+1) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-6) \\ &C(3k+2) = u(k) + u(k-1) + u(k-4) + u(k-6) \quad \text{ for } k = 0,1,...,467; \ u(k) = 0 \text{ for } k < 0 \end{split}$$

The code is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied in such a way that the following coded bits:

P1	{C(18j), C(1+18j), C(4+18j), C(8+18j), C(11+18j), C(12+18j), C(13+18j), C(15+18j)
	for j = 0,1,,77} are transmitted
	except {C(k) for k = 1,19,37,235,415,595,775,955,1135,1351,1369,1387} which are not transmitted
P2	{C(2+18j), C(3+18j), C(5+18j), C(6+18j), C(10+18j), C(14+18j), C(16+18j), C(17+18j)
	for j = 0,1,,77} are transmitted
	except {C(k) for k = 16,34,52,196,376,556,736,916,1096,1366,1384,1402} which are not transmitted
P3	{C(2+18j), C(5+18j), C(6+18j), C(7+18j), C(9+18j), C(12+18j), C(13+18j), C(16+18j)
	for j = 0,1,,77} are transmitted
	except {C(k) for k = 13,31,49,301,481,661,841,1021,1201,1363,1381,1399} which are not transmitted

The result is a block of 612 coded bits, {c1(0),c1(1),...,c1(611)}.

For the FANR procedure, the code is punctured depending on the CPS field and the PANI field as defined in 3GPP TS 44.060. If the PANI field is set to 0, the puncturing is the same as for EGPRS. If the PANI field is set to 1, the puncturing schemes named P1 or P2 are applied in such a way that, in addition to the bits punctured for EGPRS, the following coded bits:

P1	$\{C(13+36j) \text{ for } j=0,1,,38\}$ are not transmitted except $\{C(k) \text{ for } k=13,49,1381\}$ which are transmitted
P2	$\{C(5+36j) \text{ for } j=0,1,,38\}$ are not transmitted except $\{C(k) \text{ for } k=185,545,1085\}$ which are transmitted
P3	$\{C(6+36j) \text{ for } j=0,1,,38\}$ are not transmitted except $\{C(k) \text{ for } k=294,654,1194\}$ which are transmitted

The result is a block of 576 coded bits  $\{pc1(0),pc1(1),...,pc1(575)\}$ .

### II) Second half:

The same data coding as for first half is proceeded with bits  $\{d(40),d(41),...,d(489)\}$  replaced by bits  $\{d(490),d(491),...,d(939)\}$ . The result is a block of 612 coded bits,  $\{c2(0),c2(1),...,c2(611)\}$ .

If the PANI field is set to 1, additional bits are punctured as for the first half. The result is a block of 576 coded bits  $\{pc2(0),pc2(1),...,pc2(575)\}$ .

## 5.1.11.1.4a Piggy-backed Ack/Nack coding

The operations in this subclause shall be carried out only if a PAN field is included.

a) Parity bits

Ten PAN parity bits p(0), p(1),...,p(9) are defined in such a way that in GF(2) the binary polynomial:

$$d(940)D^{29} + ... + d(959)D^{10} + p(0)D^9 + ... + p(9)$$
, when divided by:

$$D^{10} + D^9 + D^5 + D^4 + D + 1$$
, yields a remainder equal to:

$$D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D^1 + 1$$
.

The five bits  $\{d(960),...,d(964)\}\$  (TFI value or 00000, see 3GPP TS 44.060) are added bit-wise modulo 2 to the 5 last parity bits  $\{p(5),...,p(9)\}$ . This results in the ten modified PAN parity bits  $\{p(0),...,p(9)\}$  defined as:

$$pt(k) = p(k)$$
 for k=0,...,4  
 $pt(k) = d(k+955) + p(k)$  for k=5,...,9

#### b) Tail biting:

The six last modified PAN parity bits are added before information and modified PAN parity bits, the result being a block of 36 {u''(-6),...,u''(0),u''(1),...,u''(29)} bits with six negative indexes:

$$u''(k-6) = pt(k+4)$$
 for  $k = 0,1,...,5$   
 $u''(k) = d(k+940)$  for  $k = 0,1,...,19$   
 $u''(k) = pt(k-20)$  for  $k = 20,21,...,29$ 

## c) Convolutional encoder

The block of 36 bits  $\{u''(-6),...,u''(0),u''(1),...,u''(29)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 90 coded bits  $\{C(0),C(1),...,C(89)\}$  defined by:

$$C(3k) = u''(k) + u''(k-2) + u''(k-3) + u''(k-5) + u''(k-6)$$

$$C(3k+1) = u''(k) + u''(k-1) + u''(k-2) + u''(k-3) + u''(k-6)$$

$$C(3k+2) = u''(k) + u''(k-1) + u''(k-4) + u''(k-6)$$
 for  $k = 0,1,...,29$ 

The block of 90 coded bits is punctured in such way that the following coded bits:

$$\{C(2+15k), C(8+15k), C(14+15k) \text{ for } k = 0,1,...,5\}$$
 are not transmitted.

The result is a block of 72 coded bits  $\{ac(0),ac(1),...,ac(71)\}$ .

The data coded bits  $\{pc1(0),pc1(1),...,pc1(575)\}$  and  $\{pc2(0),pc2(1),...,pc2(575)\}$  are appended to the PAN coded bits by the following rule:

$$c1(k) = ac(k)$$
 for  $k = 0,1,...,71$ 

$$c1(k) = pc1(k-72)$$
 for  $k = 72,73,...,611$ 

$$c2(k) = pc1(k+540)$$
 for  $k = 0,1,...,35$ 

$$c2(k) = pc2(k-36)$$
 for  $k = 36,37,...,611$ 

The result is two blocks of 612 coded bits  $\{c1(0),c1(1),...,c1(611)\}\$  and  $\{c2(0),c2(1),...,c2(611)\}\$ .

### 5.1.11.1.5 Interleaving

a) Header

The 124 coded bits of the header, {hc(0),hc(1),...,hc(123)}, are interleaved according to the following rule:

$$hi(j) = hc(k)$$
 for  $k = 0,1,...,123$   
 $j = 31(k \mod 4) + ((17k) \mod 31)$ 

b) Data

Data are put together as one entity as described by the following rule:

$$dc(k) = c1(k)$$
 for  $k = 0,1,...,611$   
 $dc(k) = c2(k-612)$  for  $k = 612,613,...,1223$ 

The resulting block is interleaved according to the following rule:

$$di(j) = dc(k)$$
 for  $k = 0,1,...,1223$   
 $j = 306(k \text{ mod } 4) + 3((44k) \text{ mod } 102 + (k \text{ div } 4) \text{ mod } 2) + (k + 2 - (k \text{ div } 408)) \text{ mod } 3$ 

## 5.1.11.1.6 Mapping on a burst

a) Straightforward Mapping

The mapping is given by the rule:

$$e(B,j) = di(306B+j) \qquad \text{for } j = 0,1,...,152$$
 
$$e(B,j) = hi(31B+j-153) \qquad \text{for } j = 153,154,...,167$$
 
$$e(B,j) = u'(9B+j-168) \qquad \text{for } j = 168,169,...,173$$
 
$$e(B,j) = q(2B+j-174) \qquad \text{for } j = 174,175$$
 
$$e(B,j) = u'(9B+j-170) \qquad \text{for } j = 176,177,178$$
 
$$e(B,j) = hi(31B+j-164) \qquad \text{for } j = 179,180,...,194$$
 
$$e(B,j) = di(306B+j-42) \qquad \text{for } j = 195,196,...,347$$

where

q(0),q(1),...,q(7) = 1,1,1,0,0,1,1,1 identifies the coding scheme MCS-7, MCS-8 or MCS-9.

b) Bit swapping

The bit swapping is the same as for MCS-5 DL as specified in subclause 5.1.9.1.6 b).

In RTTI configuration, the bursts with B = 0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B = 1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.

c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

For B = 0

```
Swap e(B,2) with e(B,1)
   Swap e(B,59) with e(B,58)
   Swap e(B,110) with e(B,109)
   Swap e(B,209) with e(B,208)
   Swap e(B,260) with e(B,259)
   Swap e(B,317) with e(B,316)
For B = 1
   Swap e(B,23) with e(B,22)
   Swap e(B,74) with e(B,73)
   Swap e(B,131) with e(B,130)
   Swap e(B,314) with e(B,313)
   Swap e(B,224) with e(B,223)
   Swap e(B,281) with e(B,280)
   Swap e(B,191) with e(B,205)
For B = 2
   Swap e(B,38) with e(B,37)
   Swap e(B,95) with e(B,94)
   Swap e(B,146) with e(B,141)
   Swap e(B,227) with e(B,226)
   Swap e(B,278) with e(B,277)
   Swap e(B,335) with e(B,334)
   Swap e(B,176) with e(B,205)
For B = 3
   Swap e(B,2) with e(B,1)
   Swap e(B,59) with e(B,58)
   Swap e(B,92) with e(B,91)
   Swap e(B,149) with e(B,141)
   Swap e(B,242) with e(B,241)
```

## 5.1.11.2 Uplink (MCS-7 UL)

Swap e(B,299) with e(B,298)

### 5.1.11.2.1 Block constitution

The message delivered to the encoder has a fixed size of 946 information bits  $\{d(0),d(1),...,d(945)\}$ . It is delivered on a burst mode.

The message delivered to the encoder will have a fixed size of 971 information bits  $\{d(0),d(1),...,d(970)\}$  if a PAN field is included (see 3GPP TS 44.060).

The message delivered to the encoder will have a fixed size of 974 information bits {d(0),d(1),...,d(973)}, if a PAN field and an eTFI field are included (see 3GPP TS 44.060).

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

### 5.1.11.2.2 Header coding

#### a) Parity bits:

Eight header parity bits p(0),p(1),...,p(7) are defined in such a way that in GF(2) the binary polynomial:

$$d(0)D^{53} + ... + d(45)D^{8} + p(0)D^{7} + ... + p(7)$$
, when divided by:

$$D^8 + D^6 + D^3 + 1$$
, yields a remainder equal to:

$$D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D^{+1}$$
.

### b) Tail biting:

The six last header parity bits are added before information and parity bits, the result being a block of 60 bits  $\{u''(-6),...,u''(0),u''(1),...,u''(53)\}$  with six negative indexes:

$$u"(k-6) = p(k+2)$$
 for  $k = 0,1,...,5$   
 $u"(k) = d(k)$  for  $k = 0,1,...,45$   
 $u"(k) = p(k-46)$  for  $k = 46,47,...,53$ 

#### c) Convolutional encoder

This block of 60 bits  $\{u''(-6),...,u''(0),u''(1),...,u''(53)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 162 coded bits:  $\{C(0),C(1),...,C(161)\}\$  defined by:

$$C(3k) = u''(k) + u''(k-2) + u''(k-3) + u''(k-5) + u''(k-6)$$

$$C(3k+1) = u''(k) + u''(k-1) + u''(k-2) + u''(k-3) + u''(k-6)$$

$$C(3k+2) = u''(k) + u''(k-1) + u''(k-4) + u''(k-6)$$
 for  $k = 0,1,...,53$ 

The code is punctured in such a way that the following coded bits:

$$\{C(k) \text{ for } k = 35,131\}$$
 are not transmitted

The result is a block of 160 coded bits,  $\{hc(0),hc(1),...,hc(159)\}$ .

### 5.1.11.2.3 Data coding

The data coding is the same as for downlink as specified in subclause 5.1.11.1.4 where bits  $\{d(40),d(41),...,d(939)\}$  are replaced by bits  $\{d(46),d(47),...,d(945)\}$ .

### 5.1.11.2.3a Piggy-backed Ack/Nack coding

If a PAN field is included and an eTFI field is not included, the coding of the PAN field is the same as for the downlink as specified in subclause 5.1.11.1.4a where bits  $\{d(940), d(941), ..., d(964)\}$  are replaced by bits  $\{d(946), d(947), ..., d(970)\}$ .

If a PAN field and an eTFI field are included, the PAN coding is the same as for the downlink as specified in sub-clause 5.1.11.1.4a, with the exception that the five bits  $\{d(966),d(967),d(968),d(969),d(970)\}$  (TFI value or 00000, see 3GPP TS 44.060) and the three bits  $\{d(971),d(972),d(973)\}$  (eTFI value or 000, see 3GPP TS 44.060) are added bit-wise modulo 2 to the PAN parity bits  $\{p(2),p(3),p(4),p(5),p(6),p(7),p(8),p(9)\}$  resulting in the ten modified PAN parity bits defined as:

$$pt(k) = p(k)$$
 for k=0,...,1  
 $pt(k) = d(k+969) + p(k)$  for k=2,...,4  
 $pt(k) = d(k+961) + p(k)$  for k=5,...,9

The data coded bits  $\{pc1(0),pc1(1),...,pc1(575)\}$  and  $\{pc2(0),pc2(1),...,pc2(575)\}$  are appended to the PAN coded bits as described for the downlink in subclause 5.1.11.1.4a. The result is two blocks of 612 coded bits  $\{c1(0),c1(1),...,c1(611)\}$  and  $\{c2(0),c2(1),...,c2(611)\}$ .

#### 5.1.11.2.4 Interleaving

a) Header

The 160 coded bits of the header, {hc(0),hc(1),...,hc(159)}, are interleaved according to the following rule:

$$hi(j) = hc(k)$$
 for  $k = 0,1,...,159$   
 $j = 40(k \mod 4) + 2((13(k \operatorname{div} 8)) \mod 20) + ((k \mod 8) \operatorname{div} 4)$ 

b) Data

The data interleaving is the same as for MCS-7 DL as specified in subclause 5.1.11.1.5.

### 5.1.11.2.5 Mapping on a burst

a) Straightforward Mapping

The mapping is given by the rule:

$$\begin{split} e(B,j) &= di(306B+j) & \text{for } j = 0,1,...,152 \\ e(B,j) &= hi(40B+j-153) & \text{for } j = 153,154,...,173 \\ e(B,j) &= q(2B+j-174) & \text{for } j = 174,175 \\ e(B,j) &= hi(40B+j-155) & \text{for } j = 176,177,...,194 \\ e(B,j) &= di(306B+j-42) & \text{for } j = 195,196,...,347 \end{split}$$

where

q(0),q(1),...,q(7) = 1,1,1,0,0,1,1,1 identifies the coding scheme MCS-7, MCS-8 or MCS-9.

b) Bit swapping

The bit swapping is the same as for MCS-5 DL as specified in subclause 5.1.9.1.6 b).

In RTTI configuration, the bursts with B = 0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B = 1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.

c) PAN bit swapping

In case a PAN is included in the radio block, additional bits are swapped as specified in subclause 5.1.11.1.6 c).

# 5.1.12 Packet data block type 12 (MCS-8)

## 5.1.12.1 Downlink (MCS-8 DL)

### 5.1.12.1.1 Block constitution

The message delivered to the encoder has a fixed size of 1132 information bits  $\{d(0),d(1),...,d(1131)\}$ . It is delivered on a burst mode.

The message delivered to the encoder will have a fixed size of 1135 information bits  $\{d(0),d(1),...,d(1134)\}$ , if an eTFI field is included (see 3GPP TS 44.060).

The message delivered to the encoder will have a fixed size of 1157 information bits {d(0),d(1),...,d(1156)} if a PAN field is included.

The message delivered to the encoder will have a fixed size of 1160 information bits  $\{d(0),d(1),...,d(1159)\}$ , if a PAN field and an eTFI field are included (see 3GPP TS 44.060).

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

### 5.1.12.1.2 USF precoding

#### 5.1.12.1.2.1 BTTI configuration

A block of 36 bits  $\{u'(0),u'(1),...,u'(35)\}$  is derived from  $\{d(0),d(1),d(2)\}$  as described for MCS-5 DL in subclause 5.1.9.1.2.1.

### 5.1.12.1.2.2 RTTI configuration

A block of 36 bits {u'(0),u'(1),...,u'(35)} is generated as described for MCS-5 DL in subclause 5.1.9.1.2.2.

### 5.1.12.1.3 Header coding

A block of 124 coded bits  $\{hc(0),hc(1),...,hc(123)\}$  is derived from  $\{d(3),d(4),...,d(39)\}$  as described for MCS-7 DL in subclause 5.1.11.1.3, with the exception that eTFI field bits are now located in  $\{d(1132),d(1133),d(1134)\}$  if included and PAN is not included, and located in  $\{d(1157),d(1158),d(1159)\}$  if included in combination with a PAN field.

### 5.1.12.1.4 Data coding

- I) First half:
- a) Parity bits:

Twelve data parity bits p(0), p(1), ..., p(11) are defined in such a way that in GF(2) the binary polynomial:

$$d(40)D^{557} + ... + d(585)D^{12} + p(0)D^{11} + ... + p(11)$$
, when divided by:

$$D^{12} + D^{11} + D^{10} + D^8 + D^5 + D^4 + 1$$
, yields a remainder equal to:

$$D^{11} + D^{10} + D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$$
.

b) Tail bits:

Six tail bits equal to 0 are added to the information and parity bits, the result being a block of 564 bits  $\{u(0),u(1),...,u(563)\}$ :

$$u(k) = d(k+40)$$
 for  $k = 0,1,...,545$ 

$$u(k) = p(k-546)$$
 for  $k = 546,547,...,557$ 

$$u(k) = 0$$
 for  $k = 558,559,...,563$  (tail bits)

c) Convolutional encoder

This block of 564 bits  $\{u(0),u(1),...,u(563)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 1692 coded bits:  $\{C(0),C(1),...,C(1691)\}\$  defined by:

$$C(3k) = u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6)$$

$$C(3k+1) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-6)$$

$$C(3k+2) = u(k) + u(k-1) + u(k-4) + u(k-6) \quad \text{ for } k = 0,1,...,563; \ u(k) = 0 \text{ for } k < 0$$

The code is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied in such a way that the following coded bits:

P1	{C(36j), C(2+36j), C(5+36j), C(6+36j), C(10+36j), C(13+36j), C(16+36j), C(20+36j), C(23+36j),
	$C(24+36j)$ , $C(27+36j)$ , $C(31+36j)$ , $C(35+36j)$ , for $j = 0,1,,46$ as well as $\{C(845)\}$ are transmitted
P2	{C(1+36j), C(4+36j), C(8+36j), C(11+36j), C(12+36j), C(15+36j), C(17+36j), C(19+36j), C(22+36j),
	$C(25+36j)$ , $C(28+36j)$ , $C(30+36j)$ , $C(33+36j)$ , for $j = 0,1,,46$ as well as $\{C(582)\}$ are transmitted
P3	$\{C(2+36j), C(3+36j), C(7+36j), C(9+36j), C(14+36j), C(17+36j), C(18+36j), C(21+36j), C(26+36j), C$
	$C(27+36j)$ , $C(29+36j)$ , $C(32+36j)$ , $C(34+36j)$ , for $j = 0,1,,46$ as well as $\{C(1156)\}$ are transmitted

The result is a block of 612 coded bits, {c1(0),c1(1),...,c1(611)}.

For the FANR procedure, the code is punctured depending on the CPS field and the PANI field as defined in 3GPP TS 44.060. If the PANI field is set to 0, the puncturing is the same as for EGPRS. If the PANI field is set to 1, the puncturing schemes named P1 or P2 are applied in such a way that, in addition to the bits punctured for EGPRS, the following coded bits:

P1	{C(2+36j) for j = 0,1,,46} are not transmitted except {C(k) for k = 38,182,326,470,614,758,902,1046,1190,1334,1478} which are transmitted
P2	$\{C(17+36j) \text{ for } j=0,1,,46\}$ are not transmitted except $\{C(k) \text{ for } k=89,233,377,521,665,809,953,1097,1241,1385,1529}\}$ which are transmitted
P3	{C(27+36j) for j = 0,1,,46} are not transmitted except {C(k) for k = 135,279,423,567,711,855,999,1143,1287,1431,1575} which are transmitted

The result is a block of 576 coded bits  $\{pc1(0),pc1(1),...,pc1(575)\}$ .

#### II) Second half:

The same data coding as for first half is proceeded with bits  $\{d(40),d(41),...,d(585)\}$  replaced by bits  $\{d(586),d(587),...,d(1131)\}$ . The result is a block of 612 coded bits,  $\{c2(0),c2(1),...,c2(611)\}$ .

If the PANI field is set to 1, additional bits are punctured as for the first half. The result is a block of 576 coded bits {pc2(0),pc2(1),...,pc2(575)}.

### 5.1.12.1.4a Piggy-backed Ack/Nack coding

The operations in this subclause shall be carried out only if a PAN field is included.

## a) Parity bits

Ten PAN parity bits p(0), p(1),...,p(9) are defined in such a way that in GF(2) the binary polynomial:

$$d(1132)D^{29} + ... + d(1151)D^{10} + p(0)D^{9} + ... + p(9)$$
, when divided by:

$$D^{10} + D^9 + D^5 + D^4 + D + 1$$
, yields a remainder equal to:

$$D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D^1 + 1$$
.

The five bits  $\{d(1152),...,d(1156)\}$  (TFI value or 00000, see 3GPP TS 44.060) are added bit-wise modulo 2 to the 5 last parity bits  $\{p(5),...,p(9)\}$ . This results in the ten modified PAN parity bits  $\{p(0),...,p(9)\}$  defined as:

$$pt(k) = p(k)$$
 for k=0,...,4  
 $pt(k) = d(k+1147) + p(k)$  for k=5,...,9

#### b) Tail biting:

The six last modified PAN parity bits are added before information and modified PAN parity bits, the result being a block of 36 {u''(-6),...,u''(0),u''(1),...,u''(29)} bits with six negative indexes:

$$u''(k-6) = pt(k+4)$$
 for  $k = 0,1,...,5$   
 $u''(k) = d(k+1132)$  for  $k = 0,1,...,19$   
 $u''(k) = pt(k-20)$  for  $k = 20,21,...,29$ 

#### c) Convolutional encoder

The block of 36 bits  $\{u''(-6),...,u''(0),u''(1),...,u''(29)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^{2} + D^{3} + D^{5} + D^{6}$$

$$G7 = 1 + D + D^{2} + D^{3} + D^{6}$$

$$G5 = 1 + D + D^{4} + D^{6}$$

This results in a block of 90 coded bits  $\{C(0),C(1),...,C(89)\}$  defined by:

$$C(3k) = u"(k) + u"(k-2) + u"(k-3) + u"(k-5) + u"(k-6)$$

$$C(3k+1) = u"(k) + u"(k-1) + u"(k-2) + u"(k-3) + u"(k-6)$$

$$C(3k+2) = u"(k) + u"(k-1) + u"(k-4) + u"(k-6)$$
 for  $k = 0,1,...,29$ 

The block of 90 coded bits is punctured in such way that the following coded bits:

$$\{C(2+15k), C(8+15k), C(14+15k) \text{ for } k = 0,1,...,5\}$$
 are not transmitted.

The result is a block of 72 coded bits  $\{ac(0),ac(1),...,ac(71)\}.$ 

The data coded bits  $\{pc1(0),pc1(1),...,pc1(575)\}\$  and  $\{pc2(0),pc2(1),...,pc2(575)\}\$  are appended to the PAN coded bits by the following rule:

$$c1(k) = ac(2k) \qquad \qquad \text{for } k = 0,1,...,35$$
 
$$c1(k) = pc1(k-36) \qquad \qquad \text{for } k = 36,37,...,611$$
 
$$c2(k) = ac(2k+1) \qquad \qquad \text{for } k = 0,1,...,35$$
 
$$c2(k) = pc2(k-36) \qquad \qquad \text{for } k = 36,37,...,611$$

The result is two blocks of 612 coded bits  $\{c1(0),c1(1),...,c1(611)\}\$  and  $\{c2(0),c2(1),...,c2(611)\}\$ .

## 5.1.12.1.5 Interleaving

a) Header

The header interleaving is the same as for MCS-7 DL as specified in subclause 5.1.11.1.5.

b) Data

Data are put together as one entity as described by the following rule:

$$dc(k) = c1(k)$$
 for  $k = 0,1,...,611$   
 $dc(k) = c2(k-612)$  for  $k = 612,613,...,1223$ 

The resulting block is interleaved according to the following rule:

$$\begin{aligned} &\text{di}(j) = \text{dc}(k) \quad \text{for } k = 0, 1, ..., 1223 \\ &j = 306(2(k \text{ div } 612) + (k \text{ mod } 2)) + 3((74k) \text{ mod } 102 + (k \text{ div } 2) \text{ mod } 2) + (k + 2 - (k \text{ div } 204)) \text{ mod } 3 \end{aligned}$$

## 5.1.12.1.6 Mapping on a burst

a) Straightforward Mapping

The mapping is the same as for MCS-7 DL as specified in subclause 5.1.11.1.6 a).

b) Bit swapping

The bit swapping is the same as for MCS-7 DL as specified in subclause 5.1.11.1.6 b).

c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

```
For B = 0.2
   Swap e(B,2) with e(B,1)
   Swap e(B,23) with e(B,22)
   Swap e(B,113) with e(B,112)
   Swap e(B,128) with e(B,127)
   Swap e(B,155) with e(B,141)
   Swap e(B,185) with e(B,205)
   Swap e(B,260) with e(B,259)
   Swap e(B,281) with e(B,280)
For B = 1,3
   Swap e(B,59) with e(B,58)
   Swap e(B,74) with e(B,73)
   Swap e(B,176) with e(B,207)
   Swap e(B,206) with e(B,205)
   Swap e(B,227) with e(B,226)
   Swap e(B,317) with e(B,316)
   Swap e(B,332) with e(B,331)
```

## 5.1.12.2 Uplink (MCS-8 UL)

#### 5.1.12.2.1 Block constitution

The message delivered to the encoder has a fixed size of 1138 information bits  $\{d(0),d(1),...,d(1137)\}$ . It is delivered on a burst mode.

The message delivered to the encoder will have a fixed size of 1163 information bits  $\{d(0),d(1),...,d(1162)\}$  if a PAN field is included (see 3GPP TS 44.060).

The message delivered to the encoder will have a fixed size of 1166 information bits  $\{d(0),d(1),...,d(1165)\}$ , if a PAN field and an eTFI field are included (see 3GPP TS 44.060).

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

### 5.1.12.2.2 Header coding

A block of 160 coded bits  $\{hc(0),hc(1),...,hc(159)\}$  is derived from  $\{d(0),d(1),...,d(45)\}$  as described for MCS-7 UL in subclause 5.1.11.2.2.

### 5.1.12.2.3 Data coding

The data coding is the same as for downlink as specified in subclause 5.1.12.1.4 where bits  $\{d(40), d(41), \dots, d(1131)\}$  are replaced by bits  $\{d(46), d(47), \dots, d(1137)\}$ .

## 5.1.12.2.3a Piggy-backed Ack/Nack coding

If a PAN field is included and an eTFI field is not included, the coding of the PAN field is the same as for the MCS-8 DL as specified in subclause 5.1.12.1.4a where bits {d(1132), d(1133),...,d(1156)} are replaced by bits {d(1138),d(1139),...,d(1162)}.

If a PAN field and an eTFI field are included, the PAN coding is the same as for the downlink as specified in sub-clause 5.1.12.1.4a, with the exception that the five bits  $\{d(1158),d(1159),d(1160),d(1161),d(1162)\}$  (TFI value or 00000, see 3GPP TS 44.060) and the three bits  $\{d(1163),d(1164),d(1165)\}$  (eTFI value or 000, see 3GPP TS 44.060) are added bitwise modulo 2 to the PAN parity bits  $\{p(2),p(3),p(4),p(5),p(6),p(7),p(8),p(9)\}$  resulting in the ten modified PAN parity bits defined as:

$$\begin{aligned} pt(k) &= p(k) & \text{for } k = 0, ..., 1 \\ pt(k) &= d(k + 1161) + p(k) & \text{for } k = 2, ..., 4 \\ pt(k) &= d(k + 1153) + p(k) & \text{for } k = 5, ..., 9 \end{aligned}$$

The data coded bits  $\{pc1(0),pc1(1),...,pc1(575)\}$  and  $\{pc2(0),pc2(1),...,pc2(575)\}$  are appended to the PAN coded bits as described for MCS-8 DL in subclause 5.1.12.1.4a. The result is two blocks of 612 coded bits  $\{c1(0),c1(1),...,c1(611)\}$  and  $\{c2(0),c2(1),...,c2(611)\}$ .

### 5.1.12.2.4 Interleaving

a) Header

The header interleaving is the same as for MCS-7 UL as specified in subclause 5.1.11.2.4.

b) Data

The data interleaving is the same as for MCS-8 DL as specified in subclause 5.1.12.1.5.

## 5.1.12.2.5 Mapping on a burst

a) Straightforward mapping

The mapping is the same as for MCS-7 UL as specified in subclause 5.1.11.2.5 a).

b) Bit swapping

The bit swapping is the same as for MCS-7 UL as specified in subclause 5.1.11.2.5 b).

c) PAN bit swapping

In case a PAN is included in the radio block, additional bits are swapped as specified in subclause 5.1.12.1.6 c).

# 5.1.13 Packet data block type 13 (MCS-9)

## 5.1.13.1 Downlink (MCS-9 DL)

#### 5.1.13.1.1 Block constitution

The message delivered to the encoder has a fixed size of 1228 information bits  $\{d(0),d(1),...,d(1227)\}$ . It is delivered on a burst mode.

The message delivered to the encoder will have a fixed size of 1231 information bits  $\{d(0),d(1),...,d(1230)\}$ , if an eTFI field is included (see 3GPP TS 44.060).

## 5.1.13.1.2 USF precoding

### 5.1.13.1.2.1 BTTI configuration

A block of 36 bits  $\{u'(0),u'(1),...,u'(35)\}$  is derived from  $\{d(0),d(1),d(2)\}$  as described for MCS-5 DL in subclause 5.1.9.1.2.1.

#### 5.1.13.1.2.2 RTTI configuration

A block of 36 bits {u'(0),u'(1),...,u'(35)} is generated as described for MCS-5 DL in subclause 5.1.9.1.2.2.

### 5.1.13.1.3 Header coding

A block of 124 coded bits  $\{hc(0),hc(1),...,hc(123)\}$  is derived from  $\{d(3),d(4),...,d(39)\}$  as described for MCS-7 DL in subclause 5.1.11.1.3, with the exception that eTFI field bits are now located in  $\{d(1228),d(1229),d(1230)\}$  if included.

## 5.1.13.1.4 Data coding

- I) First half:
- a) Parity bits:

Twelve data parity bits p(0),p(1),...,p(11) are defined in such a way that in GF(2) the binary polynomial:

$$d(40)D^{605} + ... + d(633)D^{12} + p(0)D^{11} + ... + p(11)$$
, when divided by:  
 $D^{12} + D^{11} + D^{10} + D^8 + D^5 + D^4 + 1$ , yields a remainder equal to:

$$D^{11} + D^{10} + D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$$
.

b) Tail bits:

Six tail bits equal to 0 are added to the information and parity bits, the result being a block of 612 bits  $\{u(0),u(1),...,u(611)\}$ :

$$u(k) = d(k+40)$$
 for  $k = 0,1,...,593$ 

$$u(k) = p(k-594)$$
 for  $k = 594,595,...,605$ 

$$u(k) = 0$$
 for  $k = 606,607,...,611$  (tail bits)

#### c) Convolutional encoder

This block of 612 bits  $\{u(0),u(1),...,u(611)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 1836 coded bits:  $\{C(0),C(1),...,C(1835)\}\$  defined by:

$$C(3k) = u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6)$$

$$C(3k+1) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-6)$$

$$C(3k+2) = u(k) + u(k-1) + u(k-4) + u(k-6)$$
 for  $k = 0,1,...,611$ ;  $u(k) = 0$  for  $k < 0$ 

The code is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied in such a way that the following coded bits:

P1	$\{C(3j) \text{ for } j = 0,1,,611\}$ are transmitted
P2	$\{C(1+3j) \text{ for } j = 0,1,,611\}$ are transmitted
P3	$\{C(2+3j) \text{ for } j = 0,1,,611\}$ are transmitted

The result is a block of 612 coded bits, {c1(0),c1(1),...,c1(611)}.

#### II) Second half:

The same data coding as for first half is proceeded with bits  $\{d(40),d(41),...,d(633)\}$  replaced by bits  $\{d(634),d(635),...,d(1227)\}$ . The result is a block of 612 coded bits,  $\{c2(0),c2(1),...,c2(611)\}$ .

### 5.1.13.1.5 Interleaving

The interleaving is the same as for MCS-8 DL as specified in subclause 5.1.12.1.5.

### 5.1.13.1.6 Mapping on a burst

The mapping is the same as for MCS-7 DL as specified in subclause 5.1.11.1.6.

## 5.1.13.2 Uplink (MCS-9 UL)

#### 5.1.13.2.1 Block constitution

The message delivered to the encoder has a fixed size of 1234 information bits  $\{d(0),d(1),...,d(1233)\}$ . It is delivered on a burst mode.

### 5.1.13.2.2 Header coding

A block of 160 coded bits  $\{hc(0),hc(1),...,hc(159)\}$  is derived from  $\{d(0),d(1),...,d(45)\}$  as described for MCS-7 UL in subclause 5.1.11.2.2.

### 5.1.13.2.3 Data coding

The data coding is the same as for downlink as specified in subclause 5.1.13.1.4 where bits  $\{d(40),d(41),...,d(1227)\}$  are replaced by bits  $\{d(46),d(47),...,d(1233)\}$ .

## 5.1.13.2.4 Interleaving

The interleaving is the same as for MCS-8 UL as specified in subclause 5.1.12.2.4.

## 5.1.13.2.5 Mapping on a burst

The mapping is the same as for MCS-7 UL as specified in subclause 5.1.11.2.5.

# 5.1a Packet data traffic channels (PDTCH) for EGPRS2

For each of EGPRS2-A downlink, EGPRS2-B downlink and EGPRS2-B uplink, eight additional coding schemes are specified for the packet data traffic channels. For EGPRS2-A uplink, five additional coding schemes are specified for the packet data traffic channels.

# 5.1a.1 General descriptions of channel coding functions

#### 5.1a.1.1 Header

a) Parity bits

Given a block of N bits,  $\{h(0),...,h(N-1)\}$ , eight header parity bits  $\{p(0),p(1),...,p(7)\}$  are defined in such a way that in GF(2) the binary polynomial:

$$h(0)D^{8+N-1} + ... + h(N-1)D^8 + p(0)D^7 + ... + p(7)$$
, when divided by:

for k=5,...,7

 $D^8 + D^6 + D^3 + 1$ , yields a remainder equal to:

$$D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$$
.

In case an eTFI field (see 3GPP TS 44.060) is included the three eTFI bits  $\{et(0), et(1), et(2)\}$  are added bit-wise modulo 2 to the last three parity bits  $\{p(5), p(6), p(7)\}$ . This results in the eight modified header parity bits  $\{pt(0), ..., pt(7)\}$  defined as:

$$pt(k) = p(k)$$
 or  $k=0,...,4$ 

The parity bits are added after the block of N bits, the result being a block of N+8 bits,  $\{b(0),...,b(N+7)\}$ , in case an eTFI field is not included, defined as:

$$b(k)$$
 =  $h(k)$  for  $k = 0,1,...,N-1$ 

$$b(k) = p(k-N)$$
 for  $k = N,...,N+7$ 

In case an eTFI field is included, p(k) is replaced by pt(k) in the above operation.

b) Tail-biting convolutional encoder

pt(k) = et(k-5) + p(k)

Given the block of N+8 bits,  $\{b(0),...,b(N+7)\}$ , the six last bits are added before the block of N+8 bits, the result being a block of N+14 bits  $\{c(-6),...,c(0),c(1),...,c(N+7)\}$  with six negative indexes:

$$c(k) = b(N+8+k)$$
 for  $k = -6,...,-1$ 

$$c(k) = b(k)$$
 for  $k = 0,1,...,N+7$ 

This block of N+14 bits  $\{c(-6),...,c(0),c(1),...,c(N+7)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 3(N+8) coded bits  $\{C(0),...,C(3(N+8)-1)\}$  defined by:

$$C(3k) = c(k) + c(k-2) + c(k-3) + c(k-5) + c(k-6)$$

$$C(3k+1) = c(k) + c(k-1) + c(k-2) + c(k-3) + c(k-6)$$

$$C(3k+2) = c(k) + c(k-1) + c(k-4) + c(k-6)$$
 for  $k = 0,1,...,N+7$ 

## 5.1a.1.2 Data encoded with convolutional code

#### a) Parity bits

Given a block of N bits,  $\{i(0),...,i(N-1)\}$ , twelve data parity bits  $\{p(0),p(1),...,p(11)\}$  are defined in such a way that in GF(2) the binary polynomial:

$$i(0)D^{12+N-1} + ... + i(N-1)D^{12} + p(0)D^{11} + ... + p(11)$$
, when divided by:

$$D^{12} + D^{11} + D^{10} + D^8 + D^5 + D^4 + 1$$
, yields a remainder equal to:

$$D^{11} + D^{10} + D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1.$$

The parity bits are added after the block of N bits, the result being a block of N+12 bits,  $\{b(0),...,b(N+11)\}$ , defined as:

$$b(k) = i(k)$$
 for  $k = 0,1,...,N-1$ 

$$b(k) = p(k-N)$$
 for  $k = N,...,N+11$ 

#### b) Convolutional encoding

Given the block of N+12 bits,  $\{b(0),...,b(N+11)\}$ , six tail bits equal to 0 are added to the block of N+12 bits, the result being a block of N+18 bits  $\{c(0),...,c(N+17)\}$ :

$$c(k) = b(k)$$
 for  $k = 0,1,...,N+11$ 

$$c(k) = 0$$
 for  $k = N+12,...,N+17$  (tail bits)

This block of N+18 bits  $\{c(0),...,c(N+17)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 3(N+18) coded bits,  $\{C(0),...,C(3(N+18)-1)\}$ , defined by:

$$C(3k) = c(k) + c(k-2) + c(k-3) + c(k-5) + c(k-6)$$

$$C(3k+1) = c(k) + c(k-1) + c(k-2) + c(k-3) + c(k-6)$$

$$C(3k+2) = c(k) + c(k-1) + c(k-4) + c(k-6)$$
 for  $k = 0,1,...,N+17$ ;  $c(k) = 0$  for  $k < 0$ 

### 5.1a.1.3 Data encoded with turbo code

Given a block of N bits,  $\{i(0),...,i(N-1)\}$ , the following steps are taken:

### 5. 1a.1.3.1 Parity bits

Parity bits are added as defined in subclause 5.1a.1.2, the result being a block of N+12 bits,  $\{b(0),...,b(N+11)\}$ .

### 5.1a.1.3.2 Turbo encoding

The block of K=N+12 bits is encoded with a Turbo code. The input bits to the Turbo coder are defined as:

$$x_i = b(i-1)$$
 for  $i=1,...,K$ 

The output bits from the Turbo coder are defined as:

C(3i-3) = 
$$x_i$$

$$C(3i-2) = z_i$$

C(3i-1) = 
$$z'_i$$
 for i=1,...,K

and

 $C(3K+2i-2) = x_{K+i}$ 

 $C(3K+2i-1) = z_{K+i}$ 

 $C(3K+2i+4) = x'_{K+i}$ 

 $C(3K+2i+5) = z'_{K+i}$  for i=1,2,3

where  $z_i$ ,  $z'_i$  and  $x'_i$  are defined below.

The scheme of Turbo coder is a Parallel Concatenated Convolutional Code (PCCC) with two 8-state constituent encoders and one Turbo code internal interleaver. The coding rate of Turbo coder is 1/3. The structure of Turbo coder is illustrated in figure 2a.

The transfer function of the 8-state constituent code for PCCC is:

$$G(D) = \left[1, \frac{G9(D)}{G8(D)}\right],$$

where

$$G8(D) = 1 + D^2 + D^3$$
,

$$G9(D) = 1 + D + D^3$$
.

The initial value of the shift registers of the 8-state constituent encoders shall be all zeros when starting to encode the input bits.

Output from the Turbo coder is

$$x_1, z_1, z'_1, x_2, z_2, z'_2, \ldots, x_K, z_K, z'_K,$$

where  $x_1, x_2, ..., x_K$  are the bits input to the Turbo coder i.e. both first 8-state constituent encoder and Turbo code internal interleaver, and K is the number of bits, and  $z_1, z_2, ..., z_K$  and  $z'_1, z'_2, ..., z'_K$  are the bits output from first and second 8-state constituent encoders, respectively.

The bits output from Turbo code internal interleaver are denoted by  $x'_1, x'_2, ..., x'_K$ , and these bits are to be input to the second 8-state constituent encoder.

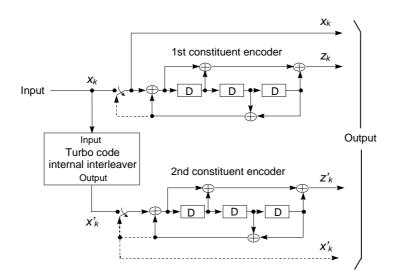


Figure 2a: Structure of rate 1/3 Turbo coder (dotted lines apply for trellis termination only)

#### 5.1a.1.3.3 Trellis termination for Turbo coder

Trellis termination is performed by taking the tail bits from the shift register feedback after all information bits are encoded. Tail bits are padded after the encoding of information bits.

The first three tail bits shall be used to terminate the first constituent encoder (upper switch of figure 2a in lower position) while the second constituent encoder is disabled. The last three tail bits shall be used to terminate the second constituent encoder (lower switch of figure 2a in lower position) while the first constituent encoder is disabled.

The transmitted bits for trellis termination shall then be:

$$X_{K+1}$$
,  $Z_{K+1}$ ,  $X_{K+2}$ ,  $Z_{K+2}$ ,  $X_{K+3}$ ,  $Z_{K+3}$ ,  $X'_{K+1}$ ,  $Z'_{K+1}$ ,  $X'_{K+2}$ ,  $Z'_{K+2}$ ,  $X'_{K+3}$ ,  $Z'_{K+3}$ .

#### 5.1a.1.3.4 Turbo code internal interleaver

The Turbo code internal interleaver consists of bits-input to a rectangular matrix with padding, intra-row and inter-row permutations of the rectangular matrix, and bits-output from the rectangular matrix with pruning. The bits input to the Turbo code internal interleaver are denoted by  $x_1, x_2, x_3, ..., x_K$ , where K is the integer number of the bits.

The following subclause specific symbols are used in subclauses 5.1a.1.3.4.1 to 5.1a.1.3.4.3:

- K Number of bits input to Turbo code internal interleaver
- R Number of rows of rectangular matrix
- C Number of columns of rectangular matrix
- p Prime number
- v Primitive root
- $\langle s(j) \rangle_{i \in \{0,1,\cdots,p-2\}}$  Base sequence for intra-row permutation
- $q_i$  Minimum prime integers
- $r_i$  Permuted prime integers
- $\langle T(i)\rangle_{i\in\{0,1,\cdots,R-1\}}$  Inter-row permutation pattern
- $\left\langle U_{i}(j)\right\rangle_{j\in\{0,1,\cdots,C-1\}}$  Intra-row permutation pattern of *i*-th row
- i Index of row number of rectangular matrix
- j Index of column number of rectangularmatrix
- k Index of bit sequence

## 5.1a.1.3.4.1 Bits-input to rectangular matrix with padding

The bit sequence  $x_1, x_2, x_3, ..., x_K$  input to the Turbo code internal interleaver is written into the rectangular matrix as follows.

(1) Determine the number of rows of the rectangular matrix R, such that:

$$R = \begin{cases} 5, & \text{if } (40 \le K \le 159) \\ 10, & \text{if } ((160 \le K \le 200) \text{ or } (481 \le K \le 530)) \\ 20, & \text{if } (K = \text{any other value}) \end{cases}$$

The rows of rectangular matrix are numbered 0, 1, ..., R-1 from top to bottom.

(2) Determine the prime number to be used in the intra-permutation, p, and the number of columns of rectangular matrix, C, such that:

if 
$$(481 \le K \le 530)$$
 then

$$p = 53$$
 and  $C = p$ .

else

Find minimum prime number p from table 0 such that

$$K \leq R \times (p+1),$$

and determine C such that

$$C = \begin{cases} p-1 & if \quad K \leq R \times (p-1) \\ p & if \quad R \times (p-1) < K \leq R \times p \\ p+1 & if \quad R \times p < K \end{cases}$$

end if

The columns of rectangular matrix are numbered 0, 1, ..., C-1 from left to right.

Table 0: List of prime number *p* and associated primitive root *v* 

р	V	р	V	р	V	р	V	р	V
7	3	47	5	101	2	157	5	223	3
11	2	53	2	103	5	163	2	227	2
13	2	59	2	107	2	167	5	229	6
17	3	61	2	109	6	173	2	233	3
19	2	67	2	113	3	179	2	239	7
23	5	71	7	127	3	181	2	241	7
29	2	73	5	131	2	191	19	251	6
31	3	79	3	137	3	193	5	257	3
37	2	83	2	139	2	197	2		
41	6	89	3	149	2	199	3		
43	3	97	5	151	6	211	2		

(3) Write the input bit sequence  $x_1, x_2, x_3, ..., x_K$  into the  $R \times C$  rectangular matrix row by row starting with bit  $y_1$  in column 0 of row 0:

$$\begin{bmatrix} y_1 & y_2 & y_3 & \cdots & y_C \\ y_{(C+1)} & y_{(C+2)} & y_{(C+3)} & \cdots & y_{2C} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ y_{((R-1)C+1)} & y_{((R-1)C+2)} & y_{((R-1)C+3)} & \cdots & y_{R\times C} \end{bmatrix}$$

where  $y_k = x_k$  for k = 1, 2, ..., K and if  $R \times C > K$ , the dummy bits are padded such that  $y_k = 0$  or 1 for  $k = K + 1, K + 2, ..., R \times C$ . These dummy bits are pruned away from the output of the rectangular matrix after intra-row and interrow permutations.

### 5.1a.1.3.4.2 Intra-row and inter-row permutations

After the bits-input to the  $R \times C$  rectangular matrix, the intra-row and inter-row permutations for the  $R \times C$  rectangular matrix are performed stepwise by using the following algorithm with steps (1) - (6):

- (1) Select a primitive root *v* from table 0 in section 5.1a.1.3.4.1, which is indicated on the right side of the prime number *p*.
- (2) Construct the base sequence  $\langle s(j) \rangle_{j \in \{0,1,\dots,p-2\}}$  for intra-row permutation as:

$$s(j) = (v \times s(j-1)) \mod p$$
,  $j = 1, 2, ..., (p-2)$ , and  $s(0) = 1$ .

- (3) Assign  $q_0 = 1$  to be the first prime integer in the sequence  $\langle q_i \rangle_{i \in \{0,1,\cdots,R-1\}}$ , and determine the prime integer  $q_i$  in the sequence  $\langle q_i \rangle_{i \in \{0,1,\cdots,R-1\}}$  to be a least prime integer such that g.c.d $(q_i, p-1) = 1$ ,  $q_i > 6$ , and  $q_i > q_{(i-1)}$  for each i = 1, 2, ..., R-1. Here g.c.d. is greatest common divisor.
- (4) Permute the sequence  $\langle q_i \rangle_{i \in \{0,1,\cdots,R-1\}}$  to make the sequence  $\langle r_i \rangle_{i \in \{0,1,\cdots,R-1\}}$  such that

$$r_{T(i)} = q_i, i = 0, 1, ..., R-1,$$

where  $\langle T(i)\rangle_{i\in\{0,1,\cdots,R-1\}}$  is the inter-row permutation pattern defined as the one of the four kind of patterns, which are shown in table 0a, depending on the number of input bits K.

Table 0a: Inter-row permutation patterns for Turbo code internal interleaver

Number of input bits  K	Number of rows <i>R</i>	Inter-row permutation patterns < T(0), T(1),, T(R - 1)>
(40 ≤ K ≤ 159)	5	<4, 3, 2, 1, 0>
$(160 \le K \le 200)$ or $(481 \le K \le 530)$	10	<9, 8, 7, 6, 5, 4, 3, 2, 1, 0>
$(2281 \le K \le 2480)$ or $(3161 \le K \le 3210)$	20	<19, 9, 14, 4, 0, 2, 5, 7, 12, 18, 16, 13, 17, 15, 3, 1, 6, 11, 8, 10>
K = any other value	20	<19, 9, 14, 4, 0, 2, 5, 7, 12, 18, 10, 8, 13, 17, 3, 1, 16, 6, 15, 11>

(5) Perform the *i*-th (i = 0, 1, ..., R - 1) intra-row permutation as:

if 
$$(C = p)$$
 then

$$U_i(j) = s((j \times r_i) \mod(p-1)), \quad j = 0, 1, ..., (p-2), \text{ and } U_i(p-1) = 0,$$

where  $U_i(j)$  is the original bit position of *j*-th permuted bit of *i*-th row.

end if

if 
$$(C = p + 1)$$
 then

$$U_i(j) = s((j \times r_i) \mod(p-1)), \quad j = 0, 1, ..., (p-2). \quad U_i(p-1) = 0, \text{ and } U_i(p) = p,$$

where  $U_i(j)$  is the original bit position of j-th permuted bit of i-th row, and

if 
$$(K = R \times C)$$
 then

Exchange  $U_{R-1}(p)$  with  $U_{R-1}(0)$ .

end if

end if

if 
$$(C = p - 1)$$
 then

$$U_i(j) = s((j \times r_i) \mod (p-1)) - 1, \quad j = 0, 1, ..., (p-2),$$

where  $U_i(j)$  is the original bit position of j-th permuted bit of i-th row.

end if

(6) Perform the inter-row permutation for the rectangular matrix based on the pattern  $\langle T(i) \rangle_{i \in \{0,1,\cdots,R-1\}}$ , where T(i) is the original row position of the i-th permuted row.

### 5.1a.1.3.4.3 Bits-output from rectangular matrix with pruning

After intra-row and inter-row permutations, the bits of the permuted rectangular matrix are denoted by  $y_k$ :

$$\begin{bmatrix} y'_1 & y'_{(R+1)} & y'_{(2R+1)} & \cdots y'_{((C-1)R+1)} \\ y'_2 & y'_{(R+2)} & y'_{(2R+2)} & \cdots y'_{((C-1)R+2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y'_R & y'_{2R} & y'_{3R} & \cdots & y'_{C\times R} \end{bmatrix}$$

The output of the Turbo code internal interleaver is the bit sequence read out column by column from the intra-row and inter-row permuted  $R \times C$  rectangular matrix starting with bit  $y'_1$  in row 0 of column 0 and ending with bit  $y'_{CR}$  in row R - 1 of column C - 1. The output is pruned by deleting dummy bits that were padded to the input of the rectangular matrix before intra-row and inter row permutations, i.e. bits  $y'_k$  that corresponds to bits  $y_k$  with k > K are removed from the output. The bits output from Turbo code internal interleaver are denoted by  $x'_1, x'_2, ..., x'_K$ , where  $x'_1$  corresponds to the bit  $y'_k$  with smallest index k after pruning,  $x'_2$  to the bit  $y'_k$  with second smallest index k after pruning, and so on. The number of bits output from Turbo code internal interleaver is K and the total number of pruned bits is:

$$R \times C - K$$
.

### 5.1a.1.3.5 Turbo code puncturing

This section defines the generation of the puncturing sequences for Turbo coded schemes. The process is defined in 6 parts.

Section 5.1a.1.3.5.1 describes the notation used.

Section 5.1a.1.3.5.2 defines setup of the length parameters based on the properties of each Modulation and Coding scheme.

Section 5.1a.1.3.5.3 defines the modification of the parameters to handle the support of a PAN field.

Section 5.1a.1.3.5.4 defines calculation of the loop parameters explicitly used in the puncturing loop.

Section 5.1a.1.3.5.5 defines the puncturing loop operation.

Section 5.1a.1.3.5.6 gives a usage example for DAS-5.

### 5.1a.1.3.5.1 Notation

The following notation is used to denote the variables used in sub-sections of Section 5.1a.1.3.5.

swap fraction of systematic bits not transmitted in P1; defined per DAS/DBS

stream 1 Vector of output bits from Turbo encoder, selected as 1,4,7,...

stream 2 Vector of output bits from Turbo encoder, selected as 2,5,8,...

stream 3 Vector of output bits from Turbo encoder, selected as 3,6,9,...

N number of bits in stream

 $N_{\scriptscriptstyle data}$  number of data bits of each BSN transmitted after puncturing, with no PAN field present

 $N_{data2}$  number of data bits of each BSN transmitted after puncturing, with PAN field present

flip logical Boolean to indicate logical XOR with P1 to map previously transmitted bits

 $P\langle r \rangle$  Puncturing sequence version, r is the version number, 1,2, or 3

 $r_{\text{max}}$  Number of puncturing sequences for a BSN, either 2 or 3

 $X_{i\langle r \rangle,\langle a \rangle}$  Internal variable used for parameter calculations  $\langle r \rangle$  is the PS number (1,2,or3)  $\langle a \rangle$  is type 1,2 or 3

 $N_{t\langle r \rangle,\langle a \rangle}$  Internal variable used for parameter calculations  $\langle r \rangle$  is the PS number (1,2,or3)  $\langle a \rangle$  is type 1,2, or 3

### 5.1a.1.3.5.2 Puncturing Loop parameter setup

This section defines the calculation of the initial parameters for the puncturing sequences. There are 2 types of puncturing for P2; the type to be used is defined per Modulation and Coding scheme in its definition.

Set the parameter values as

$$X_{i11} = N$$
,  $X_{i12} = N$ ,  $X_{i13} = N$ .

$$N_{t1.1} = \min((1 - swap) \cdot N, N_{data}),$$

$$N_{t1,2} = \left| \frac{N_{data} - N_{t1,1}}{2} \right|,$$

$$N_{t1,3} = \left\lceil \frac{N_{data} - N_{t1,1}}{2} \right\rceil$$

$$flip = 0$$
.

### 5.1a.1.3.5.2.2 P2 – second puncturing version – Type 1

Set the parameter values as

$$X_{i2,1} = N_{t1,1}, \ X_{i2,2} = N_{t1,2}, \ X_{i2,3} = N_{t1,3}$$

$$N_{t2,1} = \min\{N_{t1,1}, 2N_{data} - 3N\},\$$

$$N_{t2,2} = \left[ \frac{2N_{data} - 3N - N_{t2,1}}{2} \right],$$

$$N_{t2,3} = \left[ \frac{2N_{data} - 3N - N_{t2,1}}{2} \right]$$

$$flip = 1$$
.

## 5.1a.1.3.5.2.3 P2 – second puncturing version – Type 2

Set the parameter values as

$$X_{i2.1} = N - N_{t1.1}, \ X_{i2.2} = N - N_{t1.2}, \ X_{i2.3} = N - N_{t1.3}$$

$$N_{t2.1} = N - N_{t1.1}$$

$$N_{t2,2} = \left\lceil \frac{N_{data} - N_{t2,1}}{2} \right\rceil,$$

$$N_{t2,3} = \left| \frac{N_{data} - N_{t2,1}}{2} \right|$$

$$flip = 0$$

### 5.1a.1.3.5.2.4 P3 – third puncturing version

Set the parameter values as for P1 with a fixed swap = 0.3 as depicted below.

$$X_{i3,1} = N$$
 ,  $X_{i3,2} = N$  ,  $X_{i3,3} = N$  .

$$N_{t31} = \min((1 - swap) \cdot N, N_{data}),$$

$$N_{t3,2} = \left| \frac{N_{data} - N_{t3,1}}{2} \right|,$$

$$N_{t3,3} = \left[ \frac{N_{data} - N_{t3,1}}{2} \right]$$

$$flip = 0$$
.

## 5.1a.1.3.5.3 PAN Parameters Handling

This section deals with the treatment of the loop to handle the inclusion of the PAN field.

Set 
$$\widetilde{X}_{ir,1} = N_{tr,1}$$
,  $\widetilde{X}_{ir,2} = N_{tr,2}$ ,  $\widetilde{X}_{ir,3} = N_{tr,3}$ 

where r is the puncturing sequence number, 1,2 or 3.

If PAN is not included, then set

$$\tilde{N}_{tr,1} = N_{tr,1}, \ \tilde{N}_{tr,2} = N_{tr,2}, \ \tilde{N}_{tr,3} = N_{tr,3}$$

This has the effect of neutralising the part of the loop that punctures out the extra bits to leave space for the PAN field.

If PAN is included, then set

$$\tilde{N}_{tr,1} = N_{tr,1} - \left( \left( N_{data} - N_{data2} \right) - \left( N_{tr,2} - \tilde{N}_{tr,2} \right) - \left( N_{tr,3} - \tilde{N}_{tr,3} \right) \right),$$

$$\widetilde{N}_{tr,2} = \max(N_{tr,2} - \lfloor (N_{data} - N_{data2})/2 \rfloor, 0),$$

$$\widetilde{N}_{tr,3} = \max(N_{tr,3} - \lceil (N_{data} - N_{data2})/2 \rceil, 0)$$

### 5.1a.1.3.5.4 Puncturing Loop Parameter Calculation

The parameters  $e_{plus}$ ,  $e_{minus}$ ,,  $e_{plus}$  and  $e_{minus}$  are as defined in Table 0a using the parameters calculated in Section 5.1a.1.3.5.3.

	<b>e</b> plus	<b>e</b> minus	e2 <sub>plus</sub>	e2 <sub>minus</sub>
Stream 1	$X_{ir,1}$	$\left X_{ir,1}-N_{tr,1}\right $	${f \widetilde{X}}_{ir,1}$	$\left \widetilde{X}_{ir,1}-\widetilde{N}_{tr,1}\right $
Stream 2	$2 \cdot X_{ir,2}$	$2 \cdot \left  X_{ir,2} - N_{tr,2} \right $	$2\cdot \widetilde{X}_{ir,2}$	$2 \cdot \left  \widetilde{X}_{ir,2} - \widetilde{N}_{tr,2} \right $
Stream 3	$X_{ir,3}$	$\left X_{ir,3}-N_{tr,3}\right $	${\widetilde X}_{ir,3}$	$\left \widetilde{X}_{ir,3}-\widetilde{N}_{tr,3}\right $

**Table 0a: Rate Matching Loop Parameters** 

Also  $e_{ini}$ ,  $e2_{ini}$  are calculated for the stream 1 bits as

$$\begin{split} e_{ini} &= \left\{ \left( X_{ir,1} - \left\lfloor (r-1) \cdot e_{plus} / r_{\max} \right\rfloor - 1 \right) \operatorname{mod} e_{plus} \right\} + 1 \\ e2_{ini} &= \left\{ \left( \widetilde{X}_{ir,1} - \left\lfloor (r-1) \cdot e2_{plus} / r_{\max} \right\rfloor - 1 \right) \operatorname{mod} e2_{plus} \right\} + 1 \end{split}$$

NOTE: Given two integer numbers a and b, a mod b is defined as a – floor(a/b)\*b if b is inequal to 0, where floor(a/b) rounds a/b to the nearest integer towards minus infinity. By convention, a mod 0 is a.

Similarly, the values of  $e_{ini}$ ,  $e_{ini}^2$  are calculated for stream 2 and stream 3 bits.

For P3 puncturing scheme, the values of  $e_{ini}$  and  $e_{ini}$  shall be calculated with 'r' equal to 1 for all the streams.

### 5.1a.1.3.5.5 Puncturing Loop

This section describes the puncturing loop. The operation of the loop is based on the parameter setup and calculation described in Section 5.1a.1.3.5.3. In order to generate a puncturing sequence  $P\langle r \rangle$ , the puncturing loop is run for each stream 1,2 and 3 using the parameters calculated in 5.1a.1.3.5.4.

Regardless of the presence of the PAN field, the P1 puncturing vector without PAN field is used explicitly in the generation of P2, both Types 1 and 2. For this purpose, the generated puncturing vector of P1 is denoted as variable  $T_m$ ,

 $m=1,\ldots,N_{sys}$ . A logical value of  $T_m=1$  indicates the bit is not punctured, and 0 if punctured. For puncturing versions P1 and P3 (if relevant),  $T_m=0$  for all values of m.

```
e = e_{ini};
e2 = e2_{ini};
m=1;
while (m \le N)
  if xor(T(m), ~flip)
    e = e - e_minus;
    if (e <= 0)
      puncture the bit
      e = e + e_plus;
    else
      e2 = e2 - e2_{minus};
      if (e2 <= 0)
         puncture the bit
         e2 = e2 + e2_plus;
      endif,
    endif,
  else
    if (~flip)
      puncture the bit
    endif
  endif,
  m = m +
end while
```

### 5.1a.1.3.5.6 Usage Example, 2 puncturing versions

This section gives a usage example for DAS-5, using the parameters defined for DAS-5 in Section 5.1a.16.4.

The parameter values used for rate matching are swap=0.05,  $N_{sys}=466$ ,  $N_{data}=1248$  and  $N_{data2}=1172$ . Using these parameters, we calculate the following parameters using the equations in section 5.1a.1.3.5.2.1:

$$X_{i1.1} = X_{i1.2} = X_{i1.3} = 466,$$

$$N_{t,1} = \min((1-5/100) \cdot 466), 1248 = 443,$$

$$N_{t1,2} = \left| \frac{1248 - 443}{2} \right| = 402,$$

$$N_{t1,3} = \left\lceil \frac{1248 - 443}{2} \right\rceil = 403,$$

$$flip = 0$$
.

Assuming PAN field is not included, then

$$\widetilde{X}_{ir,1} = 443$$
,  $\widetilde{X}_{ir,2} = 402$ ,  $\widetilde{X}_{ir,3} = 403$ ,  $\widetilde{N}_{t,1} = 443$   $\widetilde{N}_{t,2} = 402$ ,  $\widetilde{N}_{t,3} = 403$ .

Placing these values in Table 0a gives values as follows:

[e\_ini\_s,e\_plus\_s,e\_minus\_s] = [466,466,23] [e\_ini\_p1,e\_plus\_p1,e\_minus\_p1] = [466,932,128]

 $[e_ini_p2,e_plus_p2,e_minus_p2] = [466,466,63]$ 

 $[e2\_ini\_s,e2\_plus\_s,e2\_minus\_s] = [443,443,0]$ 

 $[e2\_ini\_p1,e2\_plus\_p1,e2\_minus\_p1] = [402,804,0]$ 

 $[e2\_ini\_p2,e2\_plus\_p2,e2\_minus\_p2] = [403,403,0]$ 

For P1, flip=0 and T(m)=0 for every m. So, the first ten puncture pattern bits for each stream are

For stream 1 bits: 1 1 1 1 1 1 1 1 1 1

For stream 2 bits: 1 1 1 0 1 1 1 1 1 1

For stream 3 bits: 1 1 1 1 1 1 1 0 1 1

For the P2, Type 1 in this case as  $r_{\text{max}} = 2$ , the parameters are calculated in a similar manner using the equation in section 5.1a.1.3.5.2.2:

 $[e_ini_s,e_plus_s,e_minus_s] = [222,443,0]$ 

 $[e_{ini}p1,e_{plus}p1,e_{minus}p1] = [804,804,148]$ 

 $[e_ini_p2,e_plus_p2,e_minus_p2] = [202,403,76]$ 

 $[e2\_ini\_s,e2\_plus\_s,e2\_minus\_s] = [222,443,0]$ 

 $[e2\_ini\_p1,e2\_plus\_p1,e2\_minus\_p1] = [656,656,0]$ 

 $[e2\_ini\_p2,e2\_plus\_p2,e2\_minus\_p2] = [164,327,0]$ 

For P2, flip=1, and T(m) vector is the output from P1. Therefore the first ten puncture pattern bits are

For stream 1 bits: 1 1 1 1 1 1 1 1 1 1

For stream 2 bits: 1 1 1 1 1 1 0 1 1 1

For stream 3 bits: 1 1 0 1 1 1 1 1 0 1

### 5.1a.1.3.5.7 Usage Example, 3 puncturing versions

This section gives a usage example for DAS-12, using the parameters defined for DAS-12 in Section 5.1a.23.4.

### 5.1a.1.3.5.7.1 Without PAN

The parameter values used for rate matching are swap=0.0,  $N_{sys}=67$ ,  $N_{data}=700$  and  $N_{data2}=674$ . Using these parameters, we calculate the following parameters using the equations in section 5.1a.1.3.5.2.1:

$$X_{i11} = X_{i12} = X_{i13} = 674$$
,

$$N_{t11} = \min([(1-0/100)\cdot 674],700) = 674$$

$$N_{t1,2} = \left| \frac{700 - 674}{2} \right| = 13,$$

$$N_{t1,3} = \left\lceil \frac{700 - 674}{2} \right\rceil = 13,$$

$$flip = 0$$
.

Assuming PAN field is not included, then

$$\widetilde{X}_{ir,1} = 674$$
,  $\widetilde{X}_{ir,2} = 13$ ,  $\widetilde{X}_{ir,3} = 13$ ,  $\widetilde{N}_{t,1} = 674$   $\widetilde{N}_{t,2} = 13$ ,  $\widetilde{N}_{t,3} = 13$ .

Placing these values in Table 0a gives values as follows:

$$[e_{ini}p2,e_{plus}p2,e_{minus}p2] = [674, 674,661]$$

$$[e2\_ini\_s,e2\_plus\_s,e2\_minus\_s] = [674, 674,0]$$

$$[e2\_ini\_p1,e2\_plus\_p1,e2\_minus\_p1] = [13,26,0]$$

$$[e2\_ini\_p2,e2\_plus\_p2,e2\_minus\_p2] = [13,13,0]$$

For P1, flip=0 and T(m)=0 for every m. So, the first ten puncture pattern bits for each stream are

For stream 1 bits: 1 1 1 1 1 1 1 1 1 1

For stream 2 bits: 0 0 0 0 0 0 0 0 0 0

For stream 3 bits: 1 0 0 0 0 0 0 0 0 0

For the P2, Type 2 is used in this case as  $r_{\rm max}=3$  . The parameters are calculated in a similar manner using the equation in section 5.1a.1.3.5.2.3

 $[e\_ini\_s,e\_plus\_s,e\_minus\_s] = [0,0,0]$ 

 $[e_{ini}p1,e_{plus}p1,e_{minus}p1] = [221,1322,622]$ 

 $[e_ini_p2,e_plus_p2,e_minus_p2] = [441,661,311]$ 

 $[e2\_ini\_s,e2\_plus\_s,e2\_minus\_s] = [0,0,0]$ 

 $[e2\_ini\_p1,e2\_plus\_p1,e2\_minus\_p1] = [117,700,0]$ 

 $[e2\_ini\_p2,e2\_plus\_p2,e2\_minus\_p2] = [234,350,0]$ 

For P2, flip=0, and T(m) vector is the output from P1. Therefore the first ten puncture pattern bits are

For stream 1 bits: 0 0 0 0 0 0 0 0 0 0

For the P3, the parameters are calculated in a similar manner using the equation in section 5.1a.1.3.5.2.4 (with a fixed swap = 0.3), where the 'r' value in calculating  $e_{ini}$  and  $e2_{ini}$  in subsection 5.1a.1.3.5.4 is equal 1.

```
[e_ini_s,e_plus_s,e_minus_s] = [674, 674,202]

[e_ini_p1,e_plus_p1,e_minus_p1] = [674,1348,1120]

[e_ini_p2,e_plus_p2,e_minus_p2] = [674, 674,560]

[e2_ini_s,e2_plus_s,e2_minus_s] = [472,472,0]

[e2_ini_p1,e2_plus_p1,e2_minus_p1] = [114,228,0]

[e2_ini_p2,e2_plus_p2,e2_minus_p2] = [114, 114,0]
```

For P3, flip=0, and T(m)=0 for every m. swap is 30%. Therefore, the first ten puncture pattern bits are

For stream 1 bits: 1 1 1 0 1 1 0 1 1 1

### 5.1a.1.3.5.7.2 With PAN

In case of PAN inclusion, all the parameters are the same except those that specified below:

```
[e2_ini_s,e2_plus_s,e2_minus_s] = [674, 674,0]
[e2_ini_p1,e2_plus_p1,e2_minus_p1] = [13,26,26]
[e2_ini_p2,e2_plus_p2,e2_minus_p2] = [13,13,13]
```

For P1, flip=0 and T(m)=0 for every m. So, the first ten puncture pattern bits for each stream are

For the P2, Type 2 is used in this case as  $r_{\text{max}} = 3$ . The parameters are calculated in a similar manner using the equation in section 5.1a.1.3.5.2.3

```
[e2_ini_s,e2_plus_s,e2_minus_s] = [0,0,0]
[e2_ini_p1,e2_plus_p1,e2_minus_p1] = [117,700,26]
[e2_ini_p2,e2_plus_p2,e2_minus_p2] = [234,350,13]
```

For P2, flip=0, and T(m) vector is the output from P1 without PAN. Therefore the first ten puncture pattern bits are

For the P3, the parameters are calculated in a similar manner using the equation in section 5.1a.1.3.5.2.1, where the 'r' value in calculating  $e_{ini}$  and  $e_{ini}$  in subsection 5.1a.1.3.5.4 is equal to 1.

```
[e2_ini_s,e2_plus_s,e2_minus_s] = [472,472,0]
[e2_ini_p1,e2_plus_p1,e2_minus_p1] = [114,228,26]
[e2_ini_p2,e2_plus_p2,e2_minus_p2] = [114,114,13]
```

For P3, flip=0, and T(m)=0 for every m. swap is 30%. Therefore, the first ten puncture pattern bits are

For stream 1 bits: 1 1 1 0 1 1 0 1 1 1

For stream 2 bits: 0 0 1 0 0 0 0 0 1 0

For stream 3 bits: 1 0 0 0 0 1 0 0 0 0

### 5.1a.1.4 PAN

### a) Parity bits

Given a block of 25 bits,  $\{pn(0),...,pn(24)\}$ , ten PAN parity bits  $\{p(0),p(1),...,p(9)\}$  are defined in such a way that in GF(2) the binary polynomial:

$$pn(0)D^{29} + ... + pn(19)D^{10} + p(0)D^{9} + ... + p(9)$$
, when divided by:

$$D^{10} + D^9 + D^5 + D^4 + D + 1$$
, yields a remainder equal to:

$$D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$$
.

The five last bits of the PAN,  $\{pn(20),...,pn(24)\}$ , are added bit-wise modulo 2 to the 5 last parity bits  $\{p(5),...,p(9)\}$ . In case an eTFI field is included, the three eTFI bits  $\{et(0),et(1),et(2)\}$  are added bit-wise modulo 2 to the parity bits  $\{p(2),p(3),p(4)\}$ .

The modified parity bits are added after the block of 20 bits, the result being a block of 30 bits,  $\{b(0),...,b(29)\}$ , incase the eTFI field is included, defined as:

$$b(k) = pn(k)$$
 for  $k = 0,...,19$ 

$$b(k) = p(k-20)$$
 for  $k = 20,...,21$ 

$$b(k) = p(k-20) + et(k-22)$$
 for  $k = 22,...,24$ 

$$b(k) = p(k-20) + pn(k-5)$$
 for  $k = 25,...,29$ 

And incase the eTFI field is not included, defined as:

$$b(k) = pn(k)$$
 for  $k = 0,...,19$ 

$$b(k) = p(k-20)$$
 for  $k = 20,...,24b(k) = p(k-20) + pn(k-5)$  for  $k = 25,...,29$ 

# b) Tail-biting convolutional encoder

The six last bits are added before the block of 30 bits, the result being a block of 36 bits  $\{c(-6),...,c(0),c(1),...,c(29)\}$  with six negative indices:

$$c(k) = b(30+k)$$
 for  $k = -6,...,-1$ 

$$c(k) = b(k)$$
 for  $k = 0,1,...,29$ 

This block of 36 bits is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 90 coded bits  $\{C(0),...,C(89)\}$  defined by:

$$C(3k) = c(k) + c(k-2) + c(k-3) + c(k-5) + c(k-6)$$

$$C(3k+1) = c(k) + c(k-1) + c(k-2) + c(k-3) + c(k-6)$$

$$C(3k+2) = c(k) + c(k-1) + c(k-4) + c(k-6)$$
 for  $k = 0,1,...,29$ 

# 5.1a.2 General descriptions of interleaving functions

## 5.1a.2.1 Interleaver type 1

Given a block of  $N_C$  bits,  $\{c(0),...,c(N_{C}-1)\}$  and the parameter a, interleaving is performed according to the following rule:

$$i(j) = c(k)$$
 for  $k = 0,1,..., N_C-1$   
 $j = N_C B/4 + (((k \text{ div } 4) + (N_C \text{ div } 16)B)a \text{ mod } N_C/4)$   
 $B = 2(k \text{ mod } 2) + (k \text{ mod } 4) \text{ div } 2$ 

This results in a block of  $N_C$  bits  $\{i(0),...,i(N_C-1)\}$ .

### 5.1a.2.2 Interleaver type 2

Given a block of  $N_C$  bits,  $\{c(0),...,c(N_C-1)\}$  and the parameter a, interleaving is performed according to the following rule:

$$i(j) = c(k)$$
 for  $k = 0,1,..., N_{C}-1$   
 $j = ka \mod N_{C}$ 

This results in a block of  $N_C$  bits  $\{i(0),...,i(N_C-1)\}$ .

# 5.1a.3 Packet data block type 14 (UAS-7)

### 5.1a.3.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 941 information bits  $\{d(0),d(1),...,d(940)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 966 information bits  $\{d(0),d(1),...,d(965)\}$ . If the message delivered to the encoder includes a PAN and an eTFI, it has a fixed size of 969 information bits  $\{d(0),d(1),...,d(968)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$h(k) = d(k)$$
 for  $k = 0,...,40$   
 $i1(k-41) = d(k)$  for  $k = 41,...,490$   
 $i2(k-491) = d(k)$  for  $k = 491,...,940$ 

And if a PAN is included:

$$pn(k-941) = d(k)$$
 for  $k = 941,...,965$ 

And if a PAN and an eTFI are included:

$$et(k-966) = d(k)$$
 for  $k = 966,...,968$ 

## 5.1a.3.2 Header coding

The header  $\{h(0),...,h(40)\}$  is coded as defined in subclause 5.1a.1.1, with N=41, resulting in a block of 147 bits,  $\{C(0),...,C(146)\}$ .

The code is punctured in such a way that the following coded bits:

```
\{C(k) \text{ for } k = 0, 14, 27, 41, 54, 67, 81, 94, 107, 121 and 134\} are not transmitted
```

This results in a block of 136 bits,  $\{hc(0),...,hc(135)\}$ .

### 5.1a.3.3 Data coding

Each data part,  $\{i1(0),...,i1(449)\}$  and  $\{i2(0),...,i2(449)\}$ , is coded as defined in subclause 5.1a.1.2, with N=450, resulting in two coded blocks of 1404 bits,  $\{C1(0),...,C1(1403)\}$  and  $\{C2(0),...,C2(1403)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied in such a way that the following coded bits are punctured:

	Always punctured	Punctured only if a PAN is included
P1	C(33*k+j) for k=0,,41, j=2, 5, 8, 10, 11, 13, 17,	C(33*k+10) for k=4, 13, 22, 31 and 40; and
	19, 23, 25, 29, 31 and 32; and	C(33*k) for k=0, 1, 2, 3, 4, 6, 7, 8, 9, 11, 12, 13,
	C(33*42+j) for j=2, 5, 8, 10, 11, 13 and 17,	14, 16, 17, 18, 19, 21, 22, 23, 24, 25, 27, 28, 29,
	except C(33*k+10) for k=4, 13, 22, 31 and 40 that	30, 32, 33, 34, 35, 37, 38, 39 and 40.
	are not punctured.	
P2	C(33*k+j) for k=0,,41, j=1, 3, 6, 9, 12, 14, 15,	C(33*k+24) for k=1, 10, 19, 28 and 37; and
	20, 22, 24, 26, 27 and 30; and	C(33*k+4) for k=0, 1, 2, 4, 5, 6, 7, 9, 10, 11, 12,
	C(33*42+j) for j=1, 3, 6, 9, 12, 14 and 15,	14, 15, 16, 17, 19, 20, 21, 22, 23, 25, 26, 27, 28,
	except C(33*k+24) for k=1, 10, 19, 28 and 37 that	30, 31, 32, 33, 35, 36, 37, 38, 40 and 41.
	are not punctured.	

If a PAN is not included, the result is two blocks of 856 bits,  $\{c1(0),...,c1(855)\}$  and  $\{c2(0),...,c2(855)\}$ .

If a PAN is included, the result is two blocks of 817 bits,  $\{c1(0),...,c1(816)\}$  and  $\{c2(0),...,c2(816)\}$ .

NOTE: C1 and c1 correspond to i1, and C2 and c2 to i2.

## 5.1a.3.4 PAN coding

The PAN  $\{pn(0),...,pn(24)\}$ , if included, is coded as defined in subclause 5.1a.1.4, resulting in a block of 90 bits,  $\{C(0),...,C(89)\}$ .

The code is punctured in such a way that the following coded bits:

 $\{C(6*k+5), C(6*k+50) \text{ for } k=0,1,...6\}$  are not transmitted; except C(23), C(68) which are transmitted.

This results in a block of 78 bits,  $\{ac(0),...,ac(77)\}$ .

## 5.1a.3.5 Interleaving

### a) Header

The header,  $\{hc(0),...,hc(135)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C=136$  and a=23, resulting in a block of 136 bits,  $\{hi(0),...,hi(135)\}$ .

### b) Data and PAN

If a PAN is not included, data are put together as one entity as described by the following rule:

$$dc(k) = c1(k)$$
 for  $k = 0,...,855$   
 $dc(k) = c2(k-856)$  for  $k = 856,...,1711$ 

If a PAN is included, data and PAN are put together as one entity as described by the following rule:

$$dc(k) = ac(k)$$
 for  $k = 0,...,77$   
 $dc(k) = c1(k-78)$  for  $k = 78,...,894$   
 $dc(k) = c2(k-895)$  for  $k = 895,...,1711$ 

The block  $\{dc(0),...,dc(1711)\}$  is interleaved as defined in subclause 5.1a.2.1, with  $N_C$ =1712 and a=187, resulting in a block of 1712 bits,  $\{di(0),...,di(1711)\}$ .

# 5.1a.3.6 Mapping on a burst

## a) Straightforward mapping

The mapping is given by the rule:

For B=0,1,2,3, let

$$e(B,j) = di(428B+j)$$
 for  $j = 0,...,215$ 

$$e(B,j) = hi(34B+j-216)$$
 for  $j = 216,...,231$ 

$$e(B,j) = q(2B+j-232)$$
 for  $j = 232,233$ 

$$e(B,j) = hi(34B+j-218)$$
 for  $j = 234,...,251$ 

$$e(B,j) = di(428B+j-36)$$
 for  $j = 252,...,463$ 

where

q(0),q(1),...,q(7) = 0,0,0,0,0,0,0,0,0 identifies the coding scheme UAS-7, UAS-8 or UAS-9.

### b) Bit swapping

After this mapping the following bits are swapped:

For 
$$B = 0,1,2,3$$
,

Swap e(B,200+k) with e(B,218+k) for k=0, 1, 4, 5, 8, 9, 12, 13, 34, 35, 38, 39, 42, 43, 46, 47, 50 and 51.

### c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

For B = 0

Swap e(B,50) with e(B,40)

Swap e(B,79) with e(B,160)

Swap e(B,158) with e(B,81)

Swap e(B,183) with e(B,21)

Swap e(B,187) with e(B,177)

Swap e(B,230) with e(B,120)

Swap e(B,302) with e(B,405)

Swap e(B,327) with e(B,305)

Swap e(B,406) with e(B,280)

Swap e(B,410) with e(B,444)

Swap e(B,435) with e(B,368)

For B = 1

Swap e(B,26) with e(B,177)

Swap e(B,51) with e(B,81)

Swap e(B,130) with e(B,40)

Swap e(B,159) with e(B,120)

```
Swap e(B,274) with e(B,368)
Swap e(B,299) with e(B,280)
Swap e(B,303) with e(B,444)
```

Swap e(B,231) with e(B,160)

Swap e(B,382) with e(B,405)

Swap e(B,407) with e(B,305)

#### For B = 2

Swap e(B,23) with e(B,40)

Swap e(B,106) with e(B,120)

Swap e(B,131) with e(B,81)

Swap e(B,210) with e(B,0)

Swap e(B,214) with e(B,177)

Swap e(B,275) with e(B,405)

Swap e(B,354) with e(B,305)

Swap e(B,379) with e(B,280)

Swap e(B,383) with e(B,444)

Swap e(B,462) with e(B,368)

### For B = 3

Swap e(B,78) with e(B,120)

Swap e(B,103) with e(B,40)

Swap e(B,107) with e(B,177)

Swap e(B,186) with e(B,160)

Swap e(B,211) with e(B,81)

Swap e(B,326) with e(B,280)

Swap e(B,330) with e(B,444)

Swap e(B,355) with e(B,368)

Swap e(B,434) with e(B,305)

Swap e(B,463) with e(B,405)

# 5.1a.4 Packet data block type 15 (UAS-8)

# 5.1a.4.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 1069 information bits  $\{d(0),d(1),...,d(1068)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 1094 information bits  $\{d(0),d(1),...,d(1093)\}$ . If the message delivered to the encoder includes a PAN and an eTFI, it has a fixed size of 1097 information bits  $\{d(0),d(1),...,d(1096)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$h(k) = d(k)$$
 for  $k = 0,...,40$ 

$$i1(k-41) = d(k)$$
 for  $k = 41,...,554$ 

$$i2(k-555) = d(k)$$
 for  $k = 555,...,1068$ 

And if a PAN is included:

$$pn(k-1069) = d(k)$$
 for  $k = 1069,...,1093$ 

And if a PAN and an eTFI are included:

$$et(k-1094) = d(k)$$
 for  $k = 1094,...,1096$ 

# 5.1a.4.2 Header coding

The header coding is the same as for UAS-7 as specified in subclause 5.1a.3.2.

## 5.1a.4.3 Data coding

Each data part,  $\{i1(0),...,i1(513)\}$  and  $\{i2(0),...,i2(513)\}$ , is coded as defined in subclause 5.1a.1.2, with N=514, resulting in two coded blocks of 1596 bits,  $\{C1(0),...,C1(1595)\}$  and  $\{C2(0),...,C2(1595)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied in such a way that the following coded bits are punctured:

	Always punctured	Punctured only if a PAN is included
P1	C(15*k+j) for k=0,,105, j=2, 4, 8, 9, 11, 12 and	C(15*k+9) for k=13, 40, 67 and 94; and
	13; and	C(15*k+5) for k=0, 3, 6, 9, 12, 15, 18, 21, 24, 27,
	C(15*106+j) for j=2 and 4,	30, 33, 36, 39, 42, 45, 48, 51, 54, 57, 60, 63, 66,
	except C(15*k+9) for k=13, 40, 67 and 94 that are	69, 72, 76, 79, 82, 85, 88, 91, 94, 97, 100 and
	not punctured.	103
P2	C(15*k+j) for k=0,,105, j=0, 1, 3, 6, 7, 10 and	C(15*k+1) for k=8, 31, 54, 77 and 100; and
	14; and	C(15*k+13) for k=1, 4, 7, 10, 13, 16, 19, 22, 25,
	C(15*106+j) for j=0,1 and 3,	28, 31, 34, 38, 41, 44, 47, 50, 53, 56, 59, 62, 65,
	except C(15*k+1) for k=8, 31, 54, 77 and 100 that	68, 71, 74, 77, 80, 83, 86, 89, 92, 95, 98 and
	are not punctured.	101

If a PAN is not included, the result is two blocks of 856 bits,  $\{c1(0),...,c1(855)\}$  and  $\{c2(0),...,c2(855)\}$ .

If a PAN is included, the result is two blocks of 817 bits,  $\{c1(0),...,c1(816)\}$  and  $\{c2(0),...,c2(816)\}$ .

NOTE: C1 and c1 correspond to i1, and C2 and c2 to i2.

## 5.1a.4.4 PAN coding

The PAN coding is the same as for UAS-7 as specified in subclause 5.1a.3.4.

## 5.1a.4.5 Interleaving

The interleaving is the same as for UAS-7 as specified in subclause 5.1a.3.5.

## 5.1a.4.6 Mapping on a burst

The mapping is the same as for UAS-7 as specified in subclause 5.1a.3.6.

# 5.1a.5 Packet data block type 16 (UAS-9)

### 5.1a.5.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 1229 information bits  $\{d(0),d(1),...,d(1228)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 1254 information bits  $\{d(0),d(1),...,d(1253)\}$ . If the message delivered to the encoder includes a PAN and an eTFI, it has a fixed size of 1257 information bits  $\{d(0),d(1),...,d(1256)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

h(k) = d(k) for k = 0,...,40

i1(k-41) = d(k) for k = 41,...,634

i2(k-635) = d(k) for k = 635,...,1228

And if a PAN is included:

pn(k-1229) = d(k) for k = 1229,...,1253

And if a PAN and an eTFI are included:

$$et(k-1254) = d(k)$$
 for  $k = 1254,...,1256$ 

# 5.1a.5.2 Header coding

The header coding is the same as for UAS-7 as specified in subclause 5.1a.3.2.

## 5.1a.5.3 Data coding

Each data part,  $\{i1(0),...,i1(593)\}$  and  $\{i2(0),...,i2(593)\}$ , is coded as defined in subclause 5.1a.1.2, with N=594, resulting in two coded blocks of 1836 bits,  $\{C1(0),...,C1(1835)\}$  and  $\{C2(0),...,C2(1835)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied in such a way that the following coded bits are punctured:

	Always punctured	Punctured only if a PAN is included
P1	C(15*k+j) for k=0,,121, j=1, 2, 3, 5, 8, 9, 10 and	C(15*k+13) for k=0, 3, 6, 9, 12, 15, 18, 21, 25,
	14; and	28, 31, 34, 37, 40, 43, 47, 50, 53, 56, 59, 62, 65,
	C(15*122+j) for j=1, 2, 3 and 5	69, 72, 75, 78, 81, 84, 87, 91, 94, 97, 100, 103,
		106, 109, 112, 116 and 119
P2	C(15*k+j) for k=0,,121, j=0, 4, 5, 6, 7, 11, 12	C(15*k+9) for k=1, 4, 7, 10, 13, 16, 19, 23, 26,
	and 13; and	29, 32, 35, 38, 41, 44, 48, 51, 54, 57, 60, 63, 66,
	C(15*122+j) for j=0, 4 and 5; and	70, 73, 76, 79, 82, 85, 88, 92, 95, 98, 101, 104,
	C(15*121+9)	107, 110, 114, 117 and 120
P3	C(15*k+j) for k=0,,121, j=2, 3, 4, 8, 10, 11, 12	C(15*k+6) for k=2, 5, 8, 11, 14, 17, 20, 24, 27,
	and 14; and	30, 33, 36, 39, 42, 46, 49, 52, 55, 58, 61, 64, 68,
	C(15*122+j) for j=2, 3 and 4; and	71, 74, 77, 80, 83, 86, 89, 93, 96, 99, 102, 105,
	C(6)	108, 111, 115, 118 and 121

If a PAN is not included, the result is two blocks of 856 bits,  $\{c1(0),...,c1(855)\}$  and  $\{c2(0),...,c2(855)\}$ .

If a PAN is included, the result is two blocks of 817 bits,  $\{c1(0),...,c1(816)\}$  and  $\{c2(0),...,c2(816)\}$ .

NOTE: C1 and c1 correspond to i1, and C2 and c2 to i2.

## 5.1a.5.4 PAN coding

The PAN coding is the same as for UAS-7 as specified in subclause 5.1a.3.4.

# 5.1a.5.5 Interleaving

The interleaving is the same as for UAS-7 as specified in subclause 5.1a.3.5.

# 5.1a.5.6 Mapping on a burst

The mapping is the same as for UAS-7 as specified in subclause 5.1a.3.6.

# 5.1a.6 Packet data block type 17 (UAS-10)

### 5.1a.6.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 1402 information bits  $\{d(0),d(1),...,d(1401)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 1427 information bits  $\{d(0),d(1),...,d(1426)\}$ . If the message delivered to the encoder includes a PAN and an eTFI, it has a fixed size of 1430 information bits  $\{d(0),d(1),...,d(1429)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$\begin{array}{ll} h(k) = d(k) & \text{for } k = 0,...,51 \\ \\ i1(k-52) = d(k) & \text{for } k = 52,...,501 \\ \\ i2(k-502) = d(k) & \text{for } k = 502,...,951 \\ \\ i3(k-952) = d(k) & \text{for } k = 952,...,1401 \end{array}$$

And if a PAN is included:

$$pn(k-1402) = d(k)$$
 for  $k = 1402,...,1426$ 

And if a PAN and an eTFI are included:

$$et(k-1427) = d(k)$$
 for  $k = 1427,...,1429$ 

## 5.1a.6.2 Header coding

The header  $\{h(0),...,h(51)\}$  is coded as defined in subclause 5.1a.1.1, with N=52, resulting in a block of 180 bits,  $\{C(0),...,C(179)\}$ .

The code is punctured in such a way that the following coded bits:

```
{ C(15*k+12) for k=0,...,11} are not transmitted
```

This results in a block of 168 bits, {hc(0),...,hc(167)}.

## 5.1a.6.3 Data coding

Each data part,  $\{i1(0),...,i1(449)\}$ ,  $\{i2(0),...,i2(449)\}$  and  $\{i3(0),...,i3(449)\}$ , is coded as defined in subclause 5.1a.1.2, with N=450, resulting in three coded blocks of 1404 bits,  $\{C1(0),...,C1(1403)\}$ ,  $\{C2(0),...,C2(1403)\}$  and  $\{C3(0),...,C3(1403)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied in such a way that the following coded bits are punctured:

	Always punctured	Punctured only if a PAN is included
P1	C(18*k+j) for k=0,,77, j=2, 4, 5, 7, 8, 9, 11, 12, 13, 15 and 16, except C(18*k+7) for k=2, 8, 13, 19, 24, 30, 35, 41, 46, 52, 57, 63, 68 and 74 that are not punctured.	C(18*k+7) for k=2, 8, 13, 19, 24, 30, 35, 41, 46, 52, 57, 63, 68 and 74; and C(18*k) for k=0, 6, 13, 19, 26, 32, 39, 45, 52, 58, 65 and 71
P2	C(18*k+j) for k=0,,77, j=0, 1, 3, 4, 6, 8, 10, 11, 13, 14 and 17, except C(18*k+3) for k=4, 10, 15, 21, 26, 32, 37, 43, 48, 54, 59, 65, 70 and 76 that are not punctured.	C(18*k+3) for k=4, 10, 15, 21, 26, 32, 37, 43, 48, 54, 59, 65, 70 and 76; and C(18*k+15) for k=2, 8, 15, 21, 28, 34, 41, 47, 54, 60, 67 and 73
P3	C(18*k+j) for k=0,,77, j=1, 2, 3, 5, 6, 7, 9, 10, 14, 16 and 17, except C(18*k+1) for k=0, 6, 11, 17, 22, 28, 33, 39, 44, 50, 55, 61, 66 and 72 that are not punctured.	C(18*k+1) for k=0, 6, 11, 17, 22, 28, 33, 39, 44, 50, 55, 61, 66 and 72; and C(18*k+12) for k=4, 10, 17, 23, 30, 36, 43, 49, 56, 62, 69 and 75

If a PAN is not included, the result is three blocks of 560 bits,  $\{c1(0),...,c1(559)\}$ ,  $\{c2(0),...,c2(559)\}$  and  $\{c3(0),...,c3(559)\}$ , where c1 corresponds to i1, c2 to i2 and c3 to i3.

If a PAN is included, the result is three blocks of 534 bits,  $\{c1(0),...,c1(533)\}$ ,  $\{c2(0),...,c2(533)\}$  and  $\{c3(0),...,c3(533)\}$ , where c1 corresponds to i1, c2 to i2 and c3 to i3.

NOTE: C1 and c1 correspond to i1, C2 and c2 to i2, and C3 and c3 to i3.

# 5.1a.6.4 PAN coding

The PAN coding is the same as for UAS-7 as specified in subclause 5.1a.3.4.

## 5.1a.6.5 Interleaving

### a) Header

The header,  $\{hc(0),...,hc(167)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C=168$  and a=25, resulting in a block of 168 bits,  $\{hi(0),...,hi(167)\}$ .

### b) Data and PAN

If a PAN is not included, data are put together as one entity as described by the following rule:

$$dc(k) = c1(k)$$
 for  $k = 0,...,559$   
 $dc(k) = c2(k-560)$  for  $k = 560,...,1119$   
 $dc(k) = c3(k-1120)$  for  $k = 1120,...,1679$ 

If a PAN is included, data and PAN are put together as one entity as described by the following rule:

$$\begin{aligned} dc(k) &= ac(k) & \text{for } k = 0,...,77 \\ dc(k) &= c1(k-78) & \text{for } k = 78,...,611 \\ dc(k) &= c2(k-612) & \text{for } k = 612,...,1145 \\ dc(k) &= c3(k-1146) & \text{for } k = 1146,...,1679 \end{aligned}$$

The block  $\{dc(0),...,dc(1679)\}$  is interleaved as defined in subclause 5.1a.2.1, with  $N_C=1680$  and a=173, resulting in a block of 1680 bits,  $\{di(0),...,di(1679)\}$ .

## 5.1a.6.6 Mapping on a burst

### a) Straightforward Mapping

The mapping is given by the rule:

$$e(B,j) = di(420B+j)$$
 for  $j = 0,...,211$ 

$$e(B,j) = hi(42B+j-212)$$
 for  $j = 212,...,231$ 

$$e(B,j) = q(2B+j-232)$$
 for  $j = 232,233$ 

$$e(B,j) = hi(42B+j-214)$$
 for  $j = 234,...,255$ 

$$e(B,j) = di(420B+j-44)$$
 for  $j = 256,...,463$ 

where

q(0),q(1),...,q(7) = 1,1,1,1,1,1,1,1 identifies the coding scheme UAS-10 or UAS-11.

### b) Bit swapping

After this mapping the following bits are swapped:

For 
$$B = 0,1,2,3$$
,

Swap e(B,192+k) with e(B,214+k) for k=0, 1, 4, 5, 8, 9, 12, 13, 16, 17, 42, 43, 46, 47, 50, 51, 54, 55, 58, 59, 62 and 63.

### c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

#### For B = 0

Swap e(B,50) with e(B,120)

Swap e(B,75) with e(B,81)

Swap e(B,99) with e(B,177)

Swap e(B,174) with e(B,40)

Swap e(B,198) with e(B,160)

Swap e(B,267) with e(B,368)

Swap e(B,366) with e(B,305)

Swap e(B,390) with e(B,444)

Swap e(B,391) with e(B,280)

Swap e(B,415) with e(B,405)

### For B = 1

Swap e(B,7) with e(B,0)

Swap e(B,31) with e(B,177)

Swap e(B,106) with e(B,21)

Swap e(B,130) with e(B,120)

Swap e(B,155) with e(B,40)

Swap e(B,226) with e(B,160)

Swap e(B,251) with e(B,368)

Swap e(B,298) with e(B,280)

```
Swap e(B,322) with e(B,444)
         Swap e(B,323) with e(B,321)
         Swap e(B,347) with e(B,405)
         Swap e(B,446) with e(B,305)
For B = 2
         Swap e(B,38) with e(B,40)
         Swap e(B,62) with e(B,160)
         Swap e(B,87) with e(B,120)
         Swap e(B,186) with e(B,81)
         Swap e(B,210) with e(B,177)
         Swap e(B,211) with e(B,0)
         Swap e(B,279) with e(B,405)
         Swap e(B,378) with e(B,368)
         Swap e(B,403) with e(B,305)
         Swap e(B,427) with e(B,444)
For B = 3
         Swap e(B,19) with e(B,160)
         Swap e(B,118) with e(B,81)
         Swap e(B,143) with e(B,40)
         Swap e(B,167) with e(B,177)
         Swap e(B,214) with e(B,120)
         Swap e(B,239) with e(B,280)
         Swap e(B,310) with e(B,368)
         Swap e(B,335) with e(B,305)
         Swap e(B,359) with e(B,444)
         Swap e(B,434) with e(B,321)
```

# 5.1a.7 Packet data block type 18 (UAS-11)

Swap e(B,458) with e(B,405)

### 5.1a.7.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 1594 information bits  $\{d(0),d(1),...,d(1593)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 1619 information bits  $\{d(0),d(1),...,d(1618)\}$ . If the message delivered to the encoder includes a PAN and an eTFI, it has a fixed size of 1622 information bits  $\{d(0),d(1),...,d(1621)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$h(k) = d(k)$$
 for  $k = 0,...,51$ 

$$i1(k-52) = d(k)$$
 for  $k = 52,...,565$ 

$$i2(k-566) = d(k)$$
 for  $k = 566,...,1079$ 

$$i3(k-1080) = d(k)$$
 for  $k = 1080,...,1593$ 

And if a PAN is included:

$$pn(k-1594) = d(k)$$
 for  $k = 1594,...,1618$ 

And if a PAN and an eTFI are included:

$$et(k-1619) = d(k)$$
 for  $k = 1619,...,1621$ 

# 5.1a.7.2 Header coding

The header coding is the same as for UAS-10 as specified in subclause 5.1a.6.2.

## 5.1a.7.3 Data coding

Each data part,  $\{i1(0),...,i1(513)\}$ ,  $\{i2(0),...,i2(513)\}$  and  $\{i3(0),...,i3(513)\}$ , is coded as defined in subclause 5.1a.1.2, with N=514, resulting in three coded blocks of 1596 bits,  $\{C1(0),...,C1(1595)\}$ ,  $\{C2(0),...,C2(1595)\}$  and  $\{C3(0),...,C3(1595)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied in such a way that the following coded bits are punctured:

	Always punctured	Punctured only if a PAN is included
P1	C(57*k+j) for k=0,,27, j=0, 2, 3, 5, 6, 8, 9, 11, 12, 14, 15, 17, 19, 20, 22, 23, 25, 26, 28, 29, 31, 32, 34, 35, 37, 39, 40, 42, 43, 45, 46, 48, 49, 51, 52, 54 and 55	C(57*k+38) for k=0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25 and 26
P2	C(57*k+j) for k=0,,27, j=1, 3, 4, 6, 7, 9, 10, 12, 13, 15, 16, 18, 20, 21, 23, 24, 26, 27, 29, 30, 32, 33, 35, 36, 38, 40, 41, 43, 44, 46, 47, 49, 50, 52, 53, 55 and 56	C(57*k+2) for k=0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 24, 25, 26 and 27
P3	C(57*k+j) for k=0,,27, j=0, 2, 4, 5, 7, 8, 10, 11, 13, 14, 16, 17, 19, 21, 22, 24, 25, 27, 28, 30, 31, 33, 34, 36, 37, 39, 41, 42, 44, 45, 47, 48, 50, 51, 53, 54 and 56	C(57*k+18) for k=0, 1, 2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 19, 20, 21, 22, 23, 24, 25, 26 and 27

If a PAN is not included, the result is three blocks of 560 bits,  $\{c1(0),...,c1(559)\}$ ,  $\{c2(0),...,c2(559)\}$  and  $\{c3(0),...,c3(559)\}$ .

If a PAN is included, the result is three blocks of 534 bits,  $\{c1(0),...,c1(533)\}$ ,  $\{c2(0),...,c2(533)\}$  and  $\{c3(0),...,c3(533)\}$ .

NOTE: C1 and c1 correspond to i1, C2 and c2 to i2, and C3 and c3 to i3.

### 5.1a.7.4 PAN coding

The PAN coding is the same as for UAS-7 as specified in subclause 5.1a.3.4.

### 5.1a.7.5 Interleaving

a) Header

The interleaving is the same as for UAS-10 as specified in subclause 5.1a.6.5.

#### b) Data and PAN

If a PAN is not included, the following definitions apply:

$$dc1(k) = c1(k)$$
 for  $k = 0,...,559$ 

$$dc2(k) = c2(k)$$
 for  $k = 0,...,559$ 

$$dc3(k) = c3(k)$$
 for  $k = 0,...,559$ 

If a PAN is included, the following definitions apply:

$$dc1(k) = ac(3k)$$
 for  $k = 0,...,25$ 

$$dc1(k) = c1(k-26)$$
 for  $k = 26,...,559$ 

$$dc2(k) = ac(3k+1)$$
 for  $k = 0,...,25$ 

$$dc2(k) = c2(k-26)$$
 for  $k = 26,...,559$ 

$$dc3(k) = ac(3k+2)$$
 for  $k = 0,...,25$ 

$$dc3(k) = c3(k-26)$$
 for  $k = 26,...,559$ 

The three blocks  $\{dc1(0),...,dc1(559)\}$ ,  $\{dc2(0),...,dc2(559)\}$  and  $\{dc3(0),...,dc3(559)\}$  are separately interleaved as defined in subclause 5.1a.2.2, with N<sub>C</sub>=560 and a=359, resulting in the three blocks  $\{di1(0),...,di1(559)\}$ ,  $\{di2(0),...,di2(559)\}$  and  $\{di3(0),...,di3(559)\}$ , where di1 corresponds to dc1, di2 to dc2 and di3 to dc3.

The blocks are put together as one entity as described by the following rule:

$$di(k) = di1(k)$$
 for  $k = 0,...,559$ 

$$di(k) = di2(k-560)$$
 for  $k = 560,...,1119$ 

$$di(k) = di3(k-1120)$$
 for  $k = 1120,...,1679$ 

# 5.1a.7.6 Mapping on a burst

### a) Straightforward mapping

The mapping is the same as for UAS-10 as specified in subclause 5.1a.6.6 a).

## b) Bit swapping

The bit swapping is the same as for UAS-10 as specified in subclause 5.1a.6.6. b)

### c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

For 
$$B = 0$$

Swap e(B,15) with e(B,40)

Swap e(B,58) with e(B,81)

Swap e(B,115) with e(B,160)

Swap e(B,158) with e(B,177)

Swap e(B,187) with e(B,120)

Swap e(B,238) with e(B,280)

Swap e(B,274) with e(B,405)

Swap e(B,303) with e(B,305)

Swap e(B,346) with e(B,368)

Swap e(B,403) with e(B,444)

### For B = 1

Swap e(B,11) with e(B,160)

Swap e(B,54) with e(B,177)

Swap e(B,83) with e(B,81)

Swap e(B,126) with e(B,120)

Swap e(B,155) with e(B,21)

Swap e(B,198) with e(B,0)

Swap e(B,234) with e(B,368)

Swap e(B,299) with e(B,405)

Swap e(B,342) with e(B,444)

Swap e(B,371) with e(B,280)

Swap e(B,414) with e(B,305)

Swap e(B,443) with e(B,321)

### For B = 2

Swap e(B,22) with e(B,81)

Swap e(B,79) with e(B,177)

Swap e(B,151) with e(B,120)

Swap e(B,194) with e(B,160)

Swap e(B,267) with e(B,405)

Swap e(B,310) with e(B,444)

Swap e(B,339) with e(B,280)

Swap e(B,382) with e(B,305)

Swap e(B,439) with e(B,368)

### For B = 3

Swap e(B,18) with e(B,177)

Swap e(B,47) with e(B,120)

Swap e(B,90) with e(B,160)

Swap e(B,119) with e(B,0)

Swap e(B,162) with e(B,40)

Swap e(B,227) with e(B,81)

Swap e(B,263) with e(B,444)

Swap e(B,335) with e(B,368)

Swap e(B,378) with e(B,405)

Swap e(B,407) with e(B,280)

Swap e(B,450) with e(B,305)

# 5.1a.8 Packet data block type 19 (UBS-5)

### 5.1a.8.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 479 information bits  $\{d(0),d(1),...,d(478)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 504 information bits  $\{d(0),d(1),...,d(503)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$h(k) = d(k)$$
 for  $k = 0,...,28$   
 $i(k-29) = d(k)$  for  $k = 29,...,478$ 

And if a PAN is included:

$$pn(k-479) = d(k)$$
 for  $k = 479,...,503$ 

# 5.1a.8.2 Header coding

The header  $\{h(0),...,h(28)\}$  is coded as defined in subclause 5.1a.1.1, with N=29, resulting in a block of 111 bits,  $\{C(0),...,C(110)\}$ .

The code is punctured in such a way that the following coded bits:

{ 
$$C(37*k+36)$$
 for  $k=0,...,2$ } are not transmitted

This results in a block of 108 bits, {hc(0),...,hc(107)}.

### 5.1a.8.3 Data coding

The data,  $\{i(0),...,i(449)\}$ , is coded as defined in subclause 5.1a.1.2, with N=450, resulting in a coded block of 1404 bits,  $\{C(0),...,C(1403)\}$ .

The coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied in such a way that the following coded bits are punctured:

	Always punctured	Punctured only if a PAN is included
P1	C(27*k+j) for k=0,,51, j=5, 8, 11, 14, 17, 20, 23 and 26	C(27*k) for k=0,,51; and C(27*k+10) for k=0, 1, 3, 5, 7, 9, 11, 13, 14, 16, 18, 20, 22, 24, 26, 27, 29, 31, 33, 35, 37, 39, 40, 42, 44, 46, 48 and 50
P2	C(27*k+j) for k=0,,51, j=2, 4, 6, 12, 13, 18, 22 and 24	C(27*k+16) for k=0,,51; and C(27*k+9) for k=0, 2, 4, 6, 8, 10, 12, 13, 15, 17, 19, 21, 23, 25, 26, 28, 30, 32, 34, 36, 38, 39, 41, 43, 45, 47, 49 and 51

If a PAN is not included, the result is a block of 988 bits,  $\{c(0),...,c(987)\}$ .

If a PAN is included, the result is a block of 908 bits,  $\{c(0),...,c(907)\}$ .

# 5.1a.8.4 PAN coding

The PAN  $\{pn(0),...,pn(24)\}$ , if included, is coded as defined in subclause 5.1a.1.4, resulting in a block of 90 bits,  $\{C(0),...,C(89)\}$ .

The code is punctured in such a way that the following coded bits:

$$\{C(11+k), C(17+k), C(23+k), C(32+k), C(41+k) \text{ for } k = 0, 45\}$$
 are not transmitted

This results in a block of 80 bits,  $\{ac(0),...,ac(79)\}$ .

### 5.1a.8.5 Interleaving

a) Header

The header,  $\{hc(0),...,hc(107)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C=108$  and a=23, resulting in a block of 108 bits,  $\{hi(0),...,hi(107)\}$ .

b) Data and PAN

If a PAN is not included, the following rule applies:

$$dc(k) = c(k)$$
 for  $k = 0,...,987$ 

If a PAN is included, data and PAN are put together as one entity as described by the following rule:

$$dc(k) = ac(k)$$
 for  $k = 0,...,79$   
 $dc(k) = c(k-80)$  for  $k = 80,...,987$ 

The block  $\{dc(0),...,dc(987)\}$  is interleaved as defined in subclause 5.1a.2.1, with  $N_C$ =988 and a=108, resulting in a block of 988 bits,  $\{di(0),...,di(987)\}$ .

## 5.1a.8.6 Mapping on a burst

The mapping is given by the rule:

For B=0,1,2,3, let

$$\begin{split} &e(B,j)=di(247B+j) & \text{for } j=0,...,123 \\ &e(B,j)=hi(27B+j-124) & \text{for } j=124,...,137 \\ &e(B,j)=q(2B+j-138) & \text{for } j=138,139 \\ &e(B,j)=hi(27B+j-126) & \text{for } j=140,...,152 \\ &e(B,j)=di(247B+j-29) & \text{for } j=153,...,275 \end{split}$$

where

q(0),q(1),...,q(7) = 0,0,0,0,0,0,0,0,0 identifies the coding scheme UBS-5 or UBS-6.

# 5.1a.9 Packet data block type 20 (UBS-6)

### 5.1a.9.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 623 information bits  $\{d(0),d(1),...,d(622)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 648 information bits  $\{d(0),d(1),...,d(647)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$h(k) = d(k)$$
 for  $k = 0,...,28$   
 $i(k-29) = d(k)$  for  $k = 29,...,622$ 

And if a PAN is included:

$$pn(k-623) = d(k)$$
 for  $k = 622,...,647$ 

## 5.1a.9.2 Header coding

The header coding is the same as for UBS-5 as specified in subclause 5.1a.8.2.

### 5.1a.9.3 Data coding

The data,  $\{i(0),...,i(593)\}$ , is coded as defined in subclause 5.1a.1.2, with N=594, resulting in a coded block of 1836 bits,  $\{C(0),...,C(1835)\}$ .

The coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied in such a way that the following coded bits are punctured:

	Always punctured	Punctured only if a PAN is included
P1	C(15*k+j) for k=0,,121, j=2, 4, 8, 9, 11, 12 and 13; and C(15*122+j) for j=2 and 4, except C(15*k+9) for k=9, 24, 39, 54, 69, 84, 99 and 114 that are not punctured	C(15*k+9) for k=9, 24, 39, 54, 69, 84, 99 and 114; and C(15*k+5) for k=0, 1, 3, 5, 6, 8, 10, 11, 13, 15, 17, 18, 20, 22, 23, 25, 27, 28, 30, 32, 34, 35, 37, 39, 40, 42, 44, 45, 47, 49, 51, 52, 54, 56, 57, 59, 61, 62, 64, 66, 68, 69, 71, 73, 74, 76, 78, 79, 81, 83, 85, 86, 88, 90, 91, 93, 95, 96, 98, 100, 102, 103, 105, 107, 108, 110, 112, 113, 115, 117, 119 and 120
P2	C(15*k+j) for k=0,,121, j=0, 1, 3, 6, 7, 10 and 14; and C(15*122+j) for j=0, 1 and 3, except C(15*k+1) for k=2, 17, 32, 47, 62, 77, 92, 107 and 122 that are not punctured	C(15*k+1) for k=2, 17, 32, 47, 62, 77, 92, 107 and 122; and C(15*k+13) for k=0, 2, 4, 5, 7, 9, 11, 12, 14, 16, 17, 19, 21, 22, 24, 26, 28, 29, 31, 33, 34, 36, 38, 39, 41, 43, 45, 46, 48, 50, 51, 53, 55, 56, 58, 60, 62, 63, 65, 67, 68, 70, 72, 73, 75, 77, 79, 80, 82, 84, 85, 87, 89, 90, 92, 94, 96, 97, 99, 101, 102, 104, 106, 107, 109, 111, 113, 114, 116, 118 and 119

If a PAN is not included, the result is a block of 988 bits,  $\{c(0),...,c(987)\}$ .

If a PAN is included, the result is a block of 908 bits,  $\{c(0),...,c(907)\}$ .

## 5.1a.9.4 PAN coding

The PAN coding is the same as for UBS-5 as specified in subclause 5.1a.8.4.

## 5.1a.9.5 Interleaving

The interleaving is the same as for UBS-5 as specified in subclause 5.1a.8.5.

## 5.1a.9.6 Mapping on a burst

The mapping is the same as for UBS-5 as specified in subclause 5.1a.8.6.

# 5.1a.10 Packet data block type 21 (UBS-7)

## 5.1a.10.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 940 information bits  $\{d(0),d(1),...,d(939)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 965 information bits  $\{d(0),d(1),...,d(964)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$h(k) = d(k)$$
 for  $k = 0,...,39$ 

$$i1(k-40) = d(k)$$
 for  $k = 40,...,489$ 

$$i2(k-490) = d(k)$$
 for  $k = 490,...,939$ 

And if a PAN is included:

$$pn(k-940) = d(k)$$
 for  $k = 940,...,959+l_t$ 

## 5.1a.10.2 Header coding

The header  $\{h(0),...,h(39)\}$  is coded as defined in subclause 5.1a.1.1, with N=40, resulting in a block of 144 bits,  $\{C(0),...,C(143)\}$ .

No puncturing is applied. The coded header is defined as:

$$hc(k) = C(k)$$
 for  $k = 0,...,143$ .

## 5.1a.10.3 Data coding

Each data part,  $\{i1(0),...,i1(449)\}$  and  $\{i2(0),...,i2(449)\}$ , is coded as defined in subclause 5.1a.1.2, with N=450, resulting in two coded blocks of 1404 bits,  $\{C1(0),...,C1(1403)\}$  and  $\{C2(0),...,C2(1403)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied in such a way that the following coded bits are punctured:

	Always punctured	Punctured only if a PAN is included
P1	C(33*k+j) for k=0,,41, j=4, 8, 10, 14, 20, 23, 25,	C(33*k+20) for k=6, 12, 18, 24, 30 and 36; and
	29 and 30; and	C(33*k+18) for k=0, 1, 2, 3, 5, 6, 7, 8, 10, 11, 12,
	C(33*42+j) for j=4, 8, 10 and 14,	13, 15, 16, 17, 18, 20, 21, 22, 23, 25, 26, 27, 28,
	except C(33*k+20) for k=6, 12, 18, 24, 30 and 36	30, 31, 32, 33, 35, 36, 37, 38, 40 and 41
	that are not punctured	
P2	C(33*k+j) for k=0,,41, j=2, 5, 7, 12, 17, 19, 24,	C(33*k+26) for k=3, 9, 15, 21, 27, 33 and 39;
	26 and 28; and	and
	C(33*42+j) for j=2, 5, 7, 12 and 17,	C(33*k+13) for k=0, 1, 3, 4, 5, 6, 8, 9, 10, 11, 13,
	except C(33*k+26) for k=3, 9, 15, 21, 27, 33	14, 15, 16, 18, 19, 20, 21, 23, 24, 25, 26, 28, 29,
	and 39 that are not punctured	30, 31, 33, 34, 35, 36, 38, 39 and 40

If a PAN is not included, the result is two blocks of 1028 bits,  $\{c1(0),...,c1(1027)\}$  and  $\{c2(0),...,c2(1027)\}$ .

If a PAN is included, the result is two blocks of 988 bits,  $\{c1(0),...,c1(987)\}$  and  $\{c2(0),...,c2(987)\}$ .

NOTE: C1 and c1 correspond to i1, and C2 and c2 to i2.

## 5.1a.10.4 PAN coding

The PAN coding is the same as for UBS-5 as specified in subclause 5.1a.8.4.

### 5.1a.10.5 Interleaving

a) Header

The header,  $\{hc(0),...,hc(143)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C=144$  and a=29, resulting in a block of 144 bits,  $\{hi(0),...,hi(143)\}$ .

b) Data and PAN

If a PAN is not included, data are put together as one entity as described by the following rule:

$$dc(k) = c1(k)$$
 for  $k = 0,...,1027$ 

$$dc(k) = c2(k-1028)$$
 for  $k = 1028,...,2055$ 

If a PAN is included, data and PAN are put together as one entity as described by the following rule:

$$\begin{aligned} dc(k) &= ac(k) & \text{for } k = 0,...,79 \\ dc(k) &= c1(k-80) & \text{for } k = 80,...,1067 \\ dc(k) &= c2(k-1068) & \text{for } k = 1068,...,2055 \end{aligned}$$

The block  $\{dc(0),...,dc(2055)\}$  is interleaved as defined in subclause 5.1a.2.1, with  $N_C$ =2056 and a=403, resulting in a block of 2056 bits,  $\{di(0),...,di(2055)\}$ .

## 5.1a.10.6 Mapping on a burst

### a) Straightforward Mapping

The mapping is given by the rule:

For B=0,1,2,3, let

$$\begin{split} e(B,j) &= di(514B+j) & \text{for } j = 0,...,257 \\ e(B,j) &= hi(36B+j-258) & \text{for } j = 258,...,275 \\ e(B,j) &= q(2B+j-276) & \text{for } j = 276,277 \\ e(B,j) &= hi(36B+j-260) & \text{for } j = 278,...,295 \\ e(B,j) &= di(514B+j-38) & \text{for } j = 296,...,551 \end{split}$$

where

q(0),q(1),...,q(7) = 0,0,0,0,0,0,0,0,0 identifies the coding scheme UBS-7 or UBS-8.

### b) Bit swapping

After this mapping the following bits are swapped:

For 
$$B = 0,1,2,3$$
,

Swap e(B,240+k) with e(B,258+k) for k=0, 1, 4, 5, 8, 9, 12, 13, 16, 17, 38, 39, 42, 43, 46, 47, 50, 51, 54 and 55.

### c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

For B = 0

Swap e(B,58) with e(B,81)

Swap e(B,70) with e(B,177)

Swap e(B,99) with e(B,120)

Swap e(B,210) with e(B,201)

Swap e(B,251) with e(B,160)

Swap e(B,318) with e(B,321)

Swap e(B,330) with e(B,444)

Swap e(B,359) with e(B,500)

Swap e(B,470) with e(B,368)

Swap e(B,499) with e(B,340)

Swap e(B,511) with e(B,405)

### For B = 1

Swap e(B,102) with e(B,201)

Swap e(B,131) with e(B,81)

Swap e(B,143) with e(B,160)

Swap e(B,242) with e(B,120)

Swap e(B,254) with e(B,177)

Swap e(B,362) with e(B,368)

Swap e(B,391) with e(B,520)

Swap e(B,403) with e(B,405)

Swap e(B,502) with e(B,340)

Swap e(B,514) with e(B,444)

Swap e(B,543) with e(B,500)

### For B = 2

Swap e(B,23) with e(B,0)

Swap e(B,35) with e(B,201)

Swap e(B,134) with e(B,40)

Swap e(B,146) with e(B,160)

Swap e(B,175) with e(B,120)

Swap e(B,263) with e(B,81)

Swap e(B,275) with e(B,177)

Swap e(B,394) with e(B,500)

Swap e(B,406) with e(B,444)

Swap e(B,435) with e(B,368)

Swap e(B,546) with e(B,405)

### For B = 3

Swap e(B,26) with e(B,40)

Swap e(B,38) with e(B,177)

Swap e(B,67) with e(B,120)

Swap e(B,178) with e(B,201)

Swap e(B,207) with e(B,0)

Swap e(B,219) with e(B,160)

Swap e(B,266) with e(B,81)

Swap e(B,327) with e(B,500)

Swap e(B,438) with e(B,405)Swap e(B,467) with e(B,321)

Swap e(B,479) with e(B,444)

# 5.1a.11 Packet data block type 22 (UBS-8)

### 5.1a.11.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 1228 information bits  $\{d(0),d(1),...,d(1227)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 1253 information bits  $\{d(0),d(1),...,d(1252)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

h(k) = d(k) for k = 0,...,39 i1(k-40) = d(k) for k = 40,...,633i2(k-634) = d(k) for k = 634,...,1227

And if a PAN is included:

$$pn(k-1228) = d(k)$$
 for  $k = 1228,...,1252$ 

# 5.1a.11.2 Header coding

The header coding is the same as for UBS-7 as specified in subclause 5.1a.10.2.

# 5.1a.11.3 Data coding

Each data part,  $\{i1(0),...,i1(593)\}$  and  $\{i2(0),...,i2(593)\}$ , is coded as defined in subclause 5.1a.1.2, with N=594, resulting in two coded blocks of 1836 bits,  $\{C1(0),...,C1(1835)\}$  and  $\{C2(0),...,C2(1835)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied in such a way that the following coded bits are punctured:

	Always punctured	Punctured only if a PAN is included
P1	C(18*k+j) for k=0,,101, j=2, 4, 8, 9, 10, 14, 16	C(18*k+17) for k=4, 17, 30, 43, 56, 69, 82 and
	and 17,	95; and
	except C(18*k+17) for k=4, 17, 30, 43, 56, 69, 82	C(18*k+3) for k=0, 3, 6, 9, 12, 15, 19, 22, 25, 28,
	and 95 that are not punctured	31, 35, 38, 41, 44, 47, 51, 54, 57, 60, 63, 66, 70,
		73, 76, 79, 82, 86, 89, 92, 95 and 98
P2	C(18*k+j) for k=0,,101, j=0, 1, 5, 6, 7, 11, 12	C(18*k+7) for k=8, 21, 34, 47, 60, 73, 86 and 99;
	and 13,	and
	except C(18*k+7) for k=8, 21, 34, 47, 60, 73, 86	C(18*k+15) for k=1, 4, 7, 11, 14, 17, 20, 23, 27,
	and 99 that are not punctured	30, 33, 36, 39, 43, 46, 49, 52, 55, 58, 62, 65, 68,
		71, 74, 78, 81, 84, 87, 90, 94, 97 and 100

If a PAN is not included, the result is two blocks of 1028 bits,  $\{c1(0),...,c1(1027)\}$  and  $\{c2(0),...,c2(1027)\}$ .

If a PAN is included, the result is two blocks of 988 bits,  $\{c1(0),...,c1(987)\}$  and  $\{c2(0),...,c2(987)\}$ .

NOTE: C1 and c1 correspond to i1, and C2 and c2 to i2.

# 5.1a.11.4 PAN coding

The PAN coding is the same as for UBS-5 as specified in subclause 5.1a.8.4.

# 5.1a.11.5 Interleaving

The interleaving is the same as for UBS-7 as specified in subclause 5.1a.10.5.

# 5.1a.11.6 Mapping on a burst

The mapping is the same as for UBS-7 as specified in subclause 5.1a.10.6.

# 5.1a.12 Packet data block type 23 (UBS-9)

### 5.1a.12.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 1402 information bits  $\{d(0),d(1),...,d(1401)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 1427 information bits  $\{d(0),d(1),...,d(1426)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$h(k) = d(k)$$
 for  $k = 0,...,51$   
 $i1(k-52) = d(k)$  for  $k = 52,...,501$   
 $i2(k-502) = d(k)$  for  $k = 502,...,951$   
 $i3(k-952) = d(k)$  for  $k = 952,...,1401$ 

And if a PAN is included:

$$pn(k-1402) = d(k)$$
 for  $k = 1402,...,1426$ 

### 5.1a.12.2 Header coding

The header  $\{h(0),...,h(51)\}$  is coded as defined in subclause 5.1a.1.1, with N=52, resulting in a block of 180 bits,  $\{C(0),...,C(179)\}$ .

Some bits of the code are repeated in the following way:

```
hc(k) = C(k) for k = 0, 1, ... 179 and hc(180) = C(8), hc(181) = C(53), hc(182) = C(98), hc(183) = C(143)
```

This results in a block of 184 bits, {hc(0),...,hc(183)}.

# 5.1a.12.3 Data coding

Each data part,  $\{i1(0),...,i1(449)\}$ ,  $\{i2(0),...,i2(449)\}$  and  $\{i3(0),...,i3(449)\}$ , is coded as defined in subclause 5.1a.1.2, with N=450, resulting in three coded blocks of 1404 bits,  $\{C1(0),...,C1(1403)\}$ ,  $\{C2(0),...,C2(1403)\}$  and  $\{C3(0),...,C3(1403)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied in such a way that the following coded bits are punctured:

	Always punctured	Punctured only if a PAN is included
P1	C(21*k+j) for k=0,,65, j=2, 4, 5, 7, 8, 10, 13, 14, 16, 18 and 20; and C(21*66+j) for j=2, 4, 5, 7, 8, 10, 13, 14 and 16, except C(21*k+7) for k=16, 33 and 50 that are not punctured	C(21*k+7) for k=16, 33 and 50; and C(21*k+9) for k=0, 2, 5, 8, 10, 13, 16, 18, 21, 24, 26, 29, 32, 34, 37, 40, 42, 45, 48, 50, 53, 56, 58, 61 and 64
P2	C(21*k+j) for k=0,,65, j=0, 1, 3, 6, 9, 11, 12, 14, 15, 17 and 19; and C(21*66+j) for j=0, 1, 3, 6, 9, 11, 12, 14, 15 and 17, except C(21*k+17) for k=4, 21, 38 and 55 that are not punctured	C(21*k+17) for k=4, 21, 38 and 55; and C(21*k+20) for k=0, 3, 6, 8, 11, 14, 16, 19, 22, 24, 27, 30, 32, 35, 38, 41, 43, 46, 49, 51, 54, 57, 59 and 62
P3	C(21*k+j) for k=0,,65, j=0, 2, 5, 7, 8, 10, 11, 13, 16, 17 and 19; and C(21*66+j) for j=0, 2, 5, 7, 8, 10, 11, 13, 16 and 17, except C(21*k+10) for k=10, 27, 44 and 61 that are not punctured	C(21*k+10) for k=10, 27, 44 and 61; and C(21*k+12) for k=1, 4, 7, 9, 12, 15, 17, 20, 23, 25, 28, 31, 33, 36, 39, 41, 44, 47, 49, 52, 55, 57, 60 and 63

If a PAN is not included, the result is three blocks of 672 bits,  $\{c1(0),...,c1(671)\}$ ,  $\{c2(0),...,c2(671)\}$  and  $\{c3(0),...,c3(671)\}$ .

If a PAN is included, the result is three blocks of 644 bits,  $\{c1(0),...,c1(643)\}$ ,  $\{c2(0),...,c2(643)\}$  and  $\{c3(0),...,c3(643)\}$ .

NOTE: C1 and c1 correspond to i1, C2 and c2 to i2, and C3 and c3 to i3.

## 5.1a.12.4 PAN coding

The PAN  $\{pn(0),...,pn(24)\}$ , if included, is coded as defined in subclause 5.1a.1.4, resulting in a block of 90 bits,  $\{C(0),...,C(89)\}$ .

The code is punctured in such a way that the following coded bits:

$$\{C(15*k+5) \text{ for } k = 0,1,...5\}$$
 are not transmitted

This results in a block of 84 bits,  $\{ac(0),...,ac(83)\}$ .

## 5.1a.12.5 Interleaving

### a) Header

The header,  $\{hc(0),...,hc(183)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C=184$  and a=33, resulting in a block of 184 bits,  $\{hi(0),...,hi(183)\}$ .

### b) Data and PAN

If a PAN is not included, data are put together as one entity as described by the following rule:

$$dc(k) = c1(k)$$
 for  $k = 0,...,671$   
 $dc(k) = c2(k-672)$  for  $k = 672,...,1343$   
 $dc(k) = c3(k-1344)$  for  $k = 1344,...,2015$ 

If a PAN is included, data and PAN are put together as one entity as described by the following rule:

$$dc(k) = ac(k)$$
 for  $k = 0,...,83$   
 $dc(k) = c1(k-84)$  for  $k = 84,....,727$   
 $dc(k) = c2(k-728)$  for  $k = 728,....,1371$   
 $dc(k) = c3(k-1372)$  for  $k = 1372,....,2015$ 

The block  $\{dc(0),...,dc(2015)\}$  is interleaved as defined in subclause 5.1a.2.1, with  $N_C$ =2016 and a=229, resulting in a block of 2016 bits,  $\{di(0),...,di(2015)\}$ .

# 5.1a.12.6 Mapping on a burst

### a) Straightforward Mapping

The mapping is given by the rule:

For B=0,1,2,3, let

$$\begin{split} e(B,j) &= \text{di}(504\text{B}+j) & \text{for } j = 0,...,251 \\ e(B,j) &= \text{hi}(46\text{B}+j-252) & \text{for } j = 252,...,275 \\ e(B,j) &= \text{q}(2\text{B}+j-276) & \text{for } j = 276,277 \\ e(B,j) &= \text{hi}(46\text{B}+j-254) & \text{for } j = 278,...,299 \\ e(B,j) &= \text{di}(504\text{B}+j-48) & \text{for } j = 300,...,551 \end{split}$$

where

q(0),q(1),...,q(7) = 1,1,1,1,1,1,1,1 identifies the coding scheme UBS-9.

### b) Bit swapping

After this mapping the following bits are swapped:

For 
$$B = 0,1,2,3$$
,

Swap e(B,228+k) with e(B,254+k) for k=0, 1, 4, 5, 8, 9, 12, 13, 16, 17, 20 and 21.

Swap e(B,278+k) with e(B,300+k) for k=0, 1, 4, 5, 8, 9, 12, 13, 16, 17, 20 and 21.

### c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

For B = 0

Swap e(B,90) with e(B,40)

Swap e(B,91) with e(B,201)

Swap e(B,182) with e(B,81)

Swap e(B,183) with e(B,160)

Swap e(B,254) with e(B,120)

Swap e(B,255) with e(B,177)

Swap e(B,322) with e(B,500)

Swap e(B,367) with e(B,520)

Swap e(B,414) with e(B,405)

Swap e(B,459) with e(B,340)

Swap e(B,506) with e(B,444)

Swap e(B,551) with e(B,329)

For B = 1

Swap e(B,34) with e(B,160)

Swap e(B,79) with e(B,120)

Swap e(B,126) with e(B,177)

Swap e(B,170) with e(B,81)

Swap e(B,171) with e(B,201)

Swap e(B,310) with e(B,329)

Swap e(B,311) with e(B,405)

Swap e(B,402) with e(B,500)

Swap e(B,403) with e(B,444)

Swap e(B,494) with e(B,368)

Swap e(B,539) with e(B,340)

### For B = 2

Swap e(B,22) with e(B,120)

Swap e(B,67) with e(B,0)

Swap e(B,114) with e(B,201)

Swap e(B,159) with e(B,40)

Swap e(B,206) with e(B,177)

Swap e(B,251) with e(B,81)

Swap e(B,278) with e(B,444)

Swap e(B,390) with e(B,329)

Swap e(B,391) with e(B,368)

Swap e(B,482) with e(B,500)

Swap e(B,483) with e(B,405)

### For B = 3

Swap e(B,10) with e(B,40)

Swap e(B,11) with e(B,160)

Swap e(B,102) with e(B,81)

Swap e(B,103) with e(B,177)

Swap e(B,194) with e(B,120)

Swap e(B,239) with e(B,0)

Swap e(B,266) with e(B,201)

Swap e(B,334) with e(B,405)

Swap e(B,379) with e(B,500)

Swap e(B,426) with e(B,444)

Swap e(B,470) with e(B,329)

Swap e(B,471) with e(B,368)

# 5.1a.13 Packet data block type 24 (UBS-10)

### 5.1a.13.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 1834 information bits  $\{d(0),d(1),...,d(1833)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 1859 information bits  $\{d(0),d(1),...,d(1858)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$h(k) = d(k)$$
 for  $k = 0,...,51$   
 $i1(k-52) = d(k)$  for  $k = 52,...,645$   
 $i2(k-646) = d(k)$  for  $k = 646,...,1239$   
 $i3(k-1240) = d(k)$  for  $k = 1240,...,1833$ 

And if a PAN is included:

$$pn(k-1834) = d(k)$$
 for  $k = 1834,...,1858$ 

## 5.1a.13.2 Header coding

The header  $\{h(0),...,h(51)\}$  is coded as defined in subclause 5.1a.1.1, with N=52, resulting in a block of 180 bits,  $\{C(0),...,C(179)\}$ .

The code is punctured in such a way that the following coded bits:

```
{ C(15*k+14) for k=0,...,11} are not transmitted.
```

This results in a block of 168 bits,  $\{hc(0),...,hc(167)\}$ .

## 5.1a.13.3 Data coding

Each data part,  $\{i1(0),...,i1(593)\}$ ,  $\{i2(0),...,i2(593)\}$  and  $\{i3(0),...,i3(593)\}$ , is coded as defined in subclause 5.1a.1.2, with N=594, resulting in three coded blocks of 1836 bits,  $\{C1(0),...,C1(1835)\}$ ,  $\{C2(0),...,C2(1835)\}$  and  $\{C3(0),...,C3(1835)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied in such a way that the following coded bits are punctured:

	Always punctured	Punctured only if a PAN is included
P1	C(15*k+j) for k=0,,121, j=2, 4, 5, 6, 8, 11, 12 and 13; and C(15*122+j) for j=2, 4 and 5, except C(15*k+8) for k=10, 51 and 92 that are not punctured	C(15*k+8) for k=10, 51 and 92; and C(15*k+1) for k=0, 4, 9, 14, 19, 24, 29, 34, 39, 44, 48, 53, 58, 63, 68, 73, 78, 83, 88, 93, 97, 102, 107, 112 and 117
P2	C(15*k+j) for k=0,,121, j=0, 1, 3, 7, 8, 9, 10 and 14; and C(15*122+j) for j=0, 1 and 3, except C(15*k) for k=20, 61 and 102 that are not punctured	C(15*k) for k=20, 61 and 102; and C(15*k+12) for k=1, 6, 11, 16, 21, 26, 31, 35, 40, 45, 50, 55, 60, 65, 70, 75, 79, 84, 89, 94, 99, 104, 109, 114 and 119
P3	C(15*k+j) for k=0,,121, j=0, 2, 5, 6, 7, 11, 13 and 14; and C(15*122+j) for j=0, 2 and 5, except C(15*k+2) for k=30, 71 and 112 that are not punctured	C(15*k+2) for k=30, 71 and 112; and C(15*k+9) for k=3, 8, 13, 17, 22, 27, 32, 37, 42, 47, 52, 57, 62, 66, 71, 76, 81, 86, 91, 96, 101, 106, 110, 115 and 120

If a PAN is not included, the result is three blocks of 860 bits,  $\{c1(0),...,c1(859)\}$ ,  $\{c2(0),...,c2(859)\}$  and  $\{c3(0),...,c3(859)\}$ .

If a PAN is included, the result is three blocks of 832 bits,  $\{c1(0),...,c1(831)\}$ ,  $\{c2(0),...,c2(831)\}$  and  $\{c3(0),...,c3(831)\}$ .

NOTE: C1 and c1 correspond to i1, C2 and c2 to i2, and C3 and c3 to i3.

# 5.1a.13.4 PAN coding

The PAN coding is the same as for UBS-9 as specified in subclause 5.1a.12.4.

## 5.1a.13.5 Interleaving

### a) Header

The header,  $\{hc(0),...,hc(167)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C=168$  and a=29, resulting in a block of 168 bits,  $\{hi(0),...,hi(167)\}$ .

### b) Data and PAN

If a PAN is not included, data are put together as one entity as described by the following rule:

$$dc(k) = c1(k)$$
 for  $k = 0,...,859$   
 $dc(k) = c2(k-860)$  for  $k = 860,...,1719$   
 $dc(k) = c3(k-1720)$  for  $k = 1720,...,2579$ 

If a PAN is included, data and PAN are put together as one entity as described by the following rule:

$$dc(k) = ac(k) \qquad \qquad \text{for } k = 0, ..., 83$$
 
$$dc(k) = c1(k-84) \qquad \text{for } k = 84, ..., 915$$
 
$$dc(k) = c2(k-916) \qquad \text{for } k = 916, ..., 1747$$
 
$$dc(k) = c3(k-1748) \qquad \text{for } k = 1748, ..., 2579$$

The block  $\{dc(0),...,dc(2579)\}$  is interleaved as defined in subclause 5.1a.2.1, with  $N_C$ =2580 and a=179, resulting in a block of 2580 bits,  $\{di(0),...,di(2579)\}$ .

## 5.1a.13.6 Mapping on a burst

### a) Straightforward Mapping

The mapping is given by the rule:

For B=0,1,2,3, let

$$e(B,j) = di(645B+j)$$
 for  $j = 0,...,324$ 

$$e(B,j) = hi(42B+j-325)$$
 for  $j = 325,...,344$ 

$$e(B,j) = q(3B+j-345)$$
 for  $j = 345$ 

$$e(B,j) = hi(42B+j-326)$$
 for  $j = 346$ 

$$e(B,j) = q(3B+j-346)$$
 for  $j = 347,348$ 

$$e(B,j) = hi(42B+j-328)$$
 for  $j = 349,...,369$ 

$$e(B,j) = di(645B+j-45)$$
 for  $j = 370,...,689$ 

where

q(0),q(1),...,q(11) = 0,0,0,0,0,0,0,0,0,0,0,0,0 identifies the coding scheme UBS-10.

# b) Bit swapping

After this mapping the following bits are swapped:

For 
$$B = 0,1,2,3$$
,

Swap e(B,305+k) with e(B,326+k) for k=0, 3, 5, 8, 10, 13, 15 and 18.

Swap e(B,295+k) with e(B,327+k) for k=0 and 5

Swap e(B,298+k) with e(B,337+k) for k=0 and 5

Swap e(B,370+k) with e(B,346+k) for k=0, 3, 5, 8, 10, 13, 15, 18, 20 and 23.

Swap e(B,395+k) with e(B,362+k) for k=0 and 5.

Swap e(B,398+k) with e(B,352+k) for k=0 and 5.

## c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

### For B = 0

Swap e(B,34) with e(B,170)

Swap e(B,71) with e(B,98)

Swap e(B,142) with e(B,110)

Swap e(B,176) with e(B,200)

Swap e(B,179) with e(B,80)

Swap e(B,284) with e(B,188)

Swap e(B,321) with e(B,158)

Swap e(B,357) with e(B,410)

Swap e(B,367) with e(B,688)

Swap e(B,437) with e(B,503)

Swap e(B,474) with e(B,480)

Swap e(B,582) with e(B,433)

Swap e(B,616) with e(B,530)

Swap e(B,687) with e(B,493)

### For B = 1

Swap e(B,7) with e(B,158)

Swap e(B,44) with e(B,98)

Swap e(B,149) with e(B,200)

Swap e(B,152) with e(B,80)

Swap e(B,186) with e(B,170)

Swap e(B,257) with e(B,188)

Swap e(B,294) with e(B,110)

Swap e(B,376) with e(B,480)

Swap e(B,447) with e(B,503)

Swap e(B,481) with e(B,688)

Swap e(B,484) with e(B,433)

Swap e(B,589) with e(B,493)

Swap e(B,626) with e(B,530)

### For B = 2

Swap e(B,17) with e(B,98)

Swap e(B,51) with e(B,188)

Swap e(B,159) with e(B,170)

Swap e(B,196) with e(B,110)

Swap e(B,267) with e(B,158)

Swap e(B,304) with e(B,80)

Swap e(B,359) with e(B,530)

Swap e(B,454) with e(B,688)

Swap e(B,457) with e(B,410)

Swap e(B,491) with e(B,503)

Swap e(B,562) with e(B,493)

Swap e(B,599) with e(B,480)

Swap e(B,636) with e(B,433)

### For B = 3

Swap e(B,24) with e(B,103)

Swap e(B,27) with e(B,80)

Swap e(B,61) with e(B,188)

Swap e(B,132) with e(B,200)

Swap e(B,169) with e(B,170)

Swap e(B,206) with e(B,110)

Swap e(B,277) with e(B,158)

Swap e(B,311) with e(B,0)

Swap e(B,369) with e(B,480)

Swap e(B,427) with e(B,493)

Swap e(B,464) with e(B,530)

Swap e(B,501) with e(B,433)

Swap e(B,572) with e(B,503)

Swap e(B,609) with e(B,410)

# 5.1a.14 Packet data block type 25 (UBS-11)

### 5.1a.14.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 2248 information bits  $\{d(0),d(1),...,d(2247)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 2273 information bits  $\{d(0),d(1),...,d(2272)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$h(k) = d(k)$$
 for  $k = 0,...,63$ 

i1(k-64) = d(k) for k = 64,...,609

i2(k-610) = d(k) for k = 610,...,1155

i3(k-1156) = d(k) for k = 1156,...,1701

i4(k-1702) = d(k) for k = 1702,...,2247

And if a PAN is included:

$$pn(k-2248) = d(k)$$
 for  $k = 2248,...,2272$ 

### 5.1a.14.2 Header coding

The header  $\{h(0),...,h(63)\}$  is coded as defined in subclause 5.1a.1.1, with N=64, resulting in a block of 216 bits,  $\{C(0),...,C(215)\}$ .

The code is punctured in such a way that the following coded bits:

{ C(27\*k+24) for k=0,...,7} are not transmitted

This results in a block of 208 bits, {hc(0),...,hc(207)}.

## 5.1a.14.3 Data coding

Each data part,  $\{i1(0),...,i1(545)\}$ ,  $\{i2(0),...,i2(545)\}$ ,  $\{i3(0),...,i3(545)\}$  and  $\{i4(0),...,i4(545)\}$ , is coded as defined in subclause 5.1a.1.2, with N=546, resulting in four coded blocks of 1692 bits,  $\{C1(0),...,C1(1691)\}$ ,  $\{C2(0),...,C2(1691)\}$ ,  $\{C3(0),...,C3(1691)\}$  and  $\{C4(0),...,C4(1691)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied in such a way that the following coded bits are punctured:

	Always punctured	Punctured only if a PAN is included
P1	C(24*k+j) for k=0,,69, j=2, 4, 5, 6, 8, 10, 11, 13,	C(24*k) for k=0, 3, 7, 10, 14, 17, 21, 24, 28, 31,
	14, 15, 16, 18, 19, 21 and 22; and	35, 38, 42, 45, 49, 52, 56, 59, 63 and 66
	C(24*70+j) for j=2, 4, 5, 6, 8, 10 and 11	
P2	C(24*k+j) for k=0,,69, j=0, 1, 3, 4, 6, 7, 9, 11,	C(24*35+3); and
	12, 13, 15, 17, 18, 20 and 23; and	C(24*k+22) for k=2, 5, 9, 12, 16, 19, 23, 27, 30,
	C(24*70+j) for j=0, 1, 3, 4, 6, 7, 9 and 11,	34, 37, 41, 44, 48, 51, 55, 58, 62 and 65
	except C(24*35+3) that is not punctured	
P3	C(24*k+j) for k=0,,69, j=1, 2, 3, 5, 7, 8, 10, 12,	C(24*k+9) for k=1, 4, 8, 11, 15, 18, 22, 25, 29,
	14, 16, 17, 19, 20, 21 and 22; and	32, 36, 39, 43, 47, 50, 54, 57, 61, 64 and 68
	C(24*70+j) for j=1, 2, 3, 5, 7, 8 and 10	

If a PAN is not included, the result is four blocks of 635 bits,  $\{c1(0),...,c1(634)\}$ ,  $\{c2(0),...,c2(634)\}$ ,  $\{c3(0),...,c3(634)\}$  and  $\{c4(0),...,c4(634)\}$ .

If a PAN is included, the result is four blocks of 615 bits,  $\{c1(0),...,c1(614)\}$ ,  $\{c2(0),...,c2(614)\}$ ,  $\{c3(0),...,c3(614)\}$  and  $\{c4(0),...,c4(614)\}$ .

NOTE: C1 and c1 correspond to i1, C2 and c2 to i2, C3 and c3 to i3, and C4 and c4 to i4.

# 5.1a.14.4 PAN coding

The PAN coding is the same as for UBS-5 as specified in subclause 5.1a.8.4.

## 5.1a.14.5 Interleaving

### a) Header

The header,  $\{hc(0),...,hc(207)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C$ =208 and a=49, resulting in a block of 208 bits,  $\{hi(0),...,hi(207)\}$ .

### b) Data and PAN

If a PAN is not included, the following definitions apply:

dc1(k) = c1(k)	for $k = 0,,634$
dc2(k) = c2(k)	for $k = 0,,634$
dc3(k) = c3(k)	for $k = 0,,634$
dc4(k) = c4(k)	for $k = 0,,634$

If a PAN is included, the following definitions apply:

$$\begin{aligned} &dc1(k) = ac(4k) & & for \ k = 0,...,19 \\ &dc1(k) = c1(k-20) & for \ k = 20,...,634 \\ &dc2(k) = ac(4k+1) & for \ k = 0,...,19 \\ &dc2(k) = c2(k-20) & for \ k = 20,...,634 \\ &dc3(k) = ac(4k+2) & for \ k = 0,...,19 \\ &dc3(k) = c3(k-20) & for \ k = 20,...,634 \\ &dc4(k) = ac(4k+3) & for \ k = 0,...,19 \\ &dc4(k) = c4(k-20) & for \ k = 20,...,634 \end{aligned}$$

The four blocks  $\{dc1(0),...,dc1(634)\}$ ,  $\{dc2(0),...,dc2(634)\}$ ,  $\{dc3(0),...,dc3(634)\}$  and  $\{dc4(0),...,dc4(634)\}$  are separately interleaved as defined in subclause 5.1a.2.2, with  $N_C=635$  and a=177, resulting in the four blocks

 $\{di1(0),...,di1(634)\}, \{di2(0),...,di2(634)\}, \{di3(0),...,di3(634)\}$  and  $\{di4(0),...,di4(634)\},$  where di1 corresponds to dc1, di2 to dc2, di3 to dc3 and di4 to dc4.

The blocks are put together as one entity as described by the following rule:

for k = 1905,...,2539

$$\begin{aligned} &\text{di}(k) = \text{di}1(k) & \text{for } k = 0,...,634 \\ &\text{di}(k) = \text{di}2(k\text{-}635) & \text{for } k = 635,...,1269 \\ &\text{di}(k) = \text{di}3(k\text{-}1270) & \text{for } k = 1270,...,1904 \end{aligned}$$

# 5.1a.14.6 Mapping on a burst

di(k) = di4(k-1905)

Straightforward Mapping

The mapping is given by the rule:

For B=0,1,2,3, let

$$\begin{split} e(B,j) &= \text{di}(635B+j) & \text{for } j = 0,...,319 \\ e(B,j) &= \text{hi}(52B+j-320) & \text{for } j = 320,...,344 \\ e(B,j) &= q(3B+j-345) & \text{for } j = 345 \\ e(B,j) &= \text{hi}(52B+j-321) & \text{for } j = 346 \\ e(B,j) &= q(3B+j-346) & \text{for } j = 347,348 \\ e(B,j) &= \text{hi}(52B+j-323) & \text{for } j = 349,...,374 \\ e(B,j) &= \text{di}(635B+j-55) & \text{for } j = 375,...,689 \\ \end{split}$$

where

q(0),q(1),...,q(11) = 1,1,1,1,1,1,1,1,1,1,1,1,1,1 identifies the coding scheme UBS-11 or UBS-12.

### b) Bit swapping

After this mapping the following bits are swapped:

For 
$$B = 0,1,2,3$$
,

Swap e(B,295+k) with e(B,321+k) for k=0, 3, 5, 8, 10, 13, 15, 18, 20 and 23.

Swap e(B,280+k) with e(B,322+k) for k=0, 5 and 10.

Swap e(B,288+k) with e(B,337+k) for k=0 and 5.

Swap e(B,375+k) with e(B,346+k) for k=0, 3, 5, 8, 10, 13, 15, 18, 20, 23, 25 and 28.

Swap e(B,405+k) with e(B,362+k) for k=0, 5 and 10.

Swap e(B,408+k) with e(B,352+k) for k=0 and 5.

#### c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

For B = 0, 1, 2 and 3

Swap e(B,11) with e(B,200)

Swap e(B,42) with e(B,110)

Swap e(B,146) with e(B,98)

Swap e(B,177) with e(B,80)

Swap e(B,219) with e(B,158)

Swap e(B,292) with e(B,170)

Swap e(B,349) with e(B,530)

Swap e(B,409) with e(B,420)

Swap e(B,451) with e(B,493)

Swap e(B,482) with e(B,480)

Swap e(B,524) with e(B,688)

Swap e(B,586) with e(B,433)

Swap e(B,659) with e(B,503)

# 5.1a.15 Packet data block type 26 (UBS-12)

### 5.1a.15.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 2440 information bits  $\{d(0),d(1),...,d(2439)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 2465 information bits  $\{d(0),d(1),...,d(2464)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$h(k) = d(k)$$
 for  $k = 0,...,63$   
 $i1(k-64) = d(k)$  for  $k = 64,...,657$   
 $i2(k-658) = d(k)$  for  $k = 658,...,1251$   
 $i3(k-1252) = d(k)$  for  $k = 1252...,1845$   
 $i4(k-1846) = d(k)$  for  $k = 1846,...,2439$ 

And if a PAN is included:

$$pn(k-2440) = d(k)$$
 for  $k = 2440,...,2464$ 

#### 5.1a.15.2 Header coding

The header coding is the same as for UBS-11 as specified in subclause 5.1a.14.2.

### 5.1a.15.3 Data coding

Each data part,  $\{i1(0),...,i1(593)\}$ ,  $\{i2(0),...,i2(593)\}$ ,  $\{i3(0),...,i3(593)\}$  and  $\{i4(0),...,i4(593)\}$ , is coded as defined in subclause 5.1a.1.2, with N=594, resulting in four coded blocks of 1836 bits,  $\{C1(0),...,C1(1835)\}$ ,  $\{C2(0),...,C2(1835)\}$ ,  $\{C3(0),...,C3(1835)\}$  and  $\{C4(0),...,C4(1835)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied in such a way that the following coded bits are punctured:

	Transmitted bits when PAN not included	Transmitted bits when PAN is included
P1	C(399*k1+j1) for $k1=0,3$ , $j1=2,5,8,11,14$ ,	C(399*k2+j3) for $k2=0,3, j3=2, 5, 8, 11, 14,$
	17, 20, 23, 26, 28, 31, 34, 37, 40, 43, 46, 49, 52,	17, 20, 23, 26, 28, 31, 34, 37, 40, 43, 46, 49, 52,
	54, 57, 60, 63, 66, 69, 72, 75, 78, 80, 83, 86, 89,	54, 57, 60, 63, 66, 69, 72, 75, 78, 83, 86, 89, 92,
	92, 95, 98, 101, 104, 106, 109, 112, 115, 118, 121,	95, 98, 101, 104, 106, 109, 112, 115, 118, 121,
	124, 127, 130, 133, 135, 138, 141, 144, 147, 150,	124, 127, 130, 133, 135, 138, 141, 144, 147,
	153, 156, 159, 161, 164, 167, 170, 173, 176, 179,	150, 153, 156, 159, 164, 167, 170, 173, 176,
	182, 185, 187, 190, 193, 196, 199, 202, 205, 208,	179, 182, 185, 187, 190, 193, 196, 199, 202,
	211, 213, 216, 219, 222, 225, 228, 231, 234, 237,	205, 208, 211, 213, 216, 219, 222, 225, 228,
	239, 242, 245, 248, 251, 254, 257, 260, 263, 266,	231, 234, 237, 242, 245, 248, 251, 254, 257,
	268, 271, 274, 277, 280, 283, 286, 289, 292, 294,	260, 263, 266, 268, 271, 274, 277, 280, 283,
	297, 300, 303, 306, 309, 312, 315, 318, 320, 323,	286, 289, 292, 294, 297, 300, 303, 306, 309,
	326, 329, 332, 335, 338, 341, 344, 346, 349, 352,	312, 315, 318, 323, 326, 329, 332, 335, 338,
	355, 358, 361, 364, 367, 370, 372, 375, 378, 381,	341, 344, 346, 349, 352, 355, 358, 361, 364,
	384, 387, 390, 393, 396; and	367, 370, 372, 375, 378, 381, 384, 387, 390,
	C(399*4+j2) for j2=0, 2, 5, 8, 11, 14, 17, 20, 23,	393, 396 ; and
	26, 28, 31, 34, 37, 40, 43, 46, 49, 52, 54, 57, 60,	C(399*4+j4) for j4=0, 2, 5, 8, 11, 14, 17, 20,
	63, 66, 69, 72, 75, 78, 80, 83, 86, 89, 92, 95, 98,	23, 26, 28, 31, 34, 37, 40, 43, 46, 49, 52, 54, 57,
	101, 104, 106, 109, 112, 115, 118, 121, 124, 127,	60, 63, 66, 69, 72, 75, 78, 83, 86, 89, 92, 95, 98,
	130, 133, 135, 138, 141, 144, 147, 150, 153, 156,	101, 104, 106, 109, 112, 115, 118, 121, 124,
	159, 161, 164, 167, 170, 173, 176, 179, 182, 185,	127, 130, 133, 135, 138, 141, 144, 147, 150,
	187, 190, 193, 196, 199, 202, 205, 208, 211, 213,	153, 156, 159, 164, 167, 170, 173, 176, 179,
	216, 219, 222, 225, 228, 231, 234, 237, 239; and	182, 185, 187, 190, 193, 196, 199, 202, 205,
		208, 211, 213, 216, 219, 222, 225, 228, 231,
	C(399), C(798) and C(1197)	234, 237; and
		C(798) and C(1197)
P2	C(399*k1+(j1-1)), C(399*4+(j2-1)), C(398),	C(399*k2+(j3-1)) and $C(399*4+(j4-1))$ and
	C(797) and C(1196) where k1, j1 and j2 are as	C(797) and C(1196)where k2, j3 and j4 are as
	defined above for P1	defined above for P1
P3	C(399*k1+(j1-2)), C(399*4+(j2-2)), C(397),	C(399*k2+(j3-2)) and $C(399*4+(j4-2))$ and
	C(796) and C(1195) where k1, j1 and j2 are as	C(796) and C(1195)where k2, j3 and j4 are as
	defined above for P1	defined above for P1

If a PAN is not included, the result is four blocks of 635 bits,  $\{c1(0),...,c1(634)\}$ ,  $\{c2(0),...,c2(634)\}$ ,  $\{c3(0),...,c3(634)\}$  and  $\{c4(0),...,c4(634)\}$ .

If a PAN is included, the result is four blocks of 615 bits,  $\{c1(0),...,c1(614)\}$ ,  $\{c2(0),...,c2(614)\}$ ,  $\{c3(0),...,c3(614)\}$  and  $\{c4(0),...,c4(614)\}$ .

NOTE: C1 and c1 correspond to i1, C2 and c2 to i2, C3 and c3 to i3, and C4 and c4 to i4.

# 5.1a.15.4 PAN coding

The PAN coding is the same as for UBS-5 as specified in subclause 5.1a.8.4.

# 5.1a.15.5 Interleaving

The interleaving is the same as for UBS-11 as specified in subclause 5.1a.14.5.

# 5.1a.15.6 Mapping on a burst

The mapping is the same as for UBS-11 as specified in subclause 5.1a.14.6.

# 5.1a.16 Packet data block type 27 (DAS-5)

#### 5.1a.16.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 478 information bits  $\{d(0),d(1),...,d(477)\}$ . If the message delivered to the encoder does not include a PAN but includes an eTFI, it has a fixed size of 481 information bits  $\{d(0),d(1),...,d(480)\}$ . If the message delivered to the encoder includes a PAN but does not include an eTFI, it has a fixed size of 503 information bits  $\{d(0),d(1),...,d(502)\}$ . If the message delivered to the encoder includes a PAN and an eTFI, it has a fixed size of 506 information bits  $\{d(0),d(1),...,d(505)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

### 5.1a.16.2 USF coding

A block of 36 bits {u'(0),u'(1),...,u'(35)} is derived as described for MCS-5 DL in subclause 5.1.9.1.2.

# 5.1a.16.3 Header coding

A block of 100 coded bits  $\{hc(0),hc(1),...,hc(99)\}$  is derived from  $\{d(3),d(4),...,d(27)\}$  as described for MCS-5 DL in subclause 5.1.9.1.3.

In case an eTFI field is included and a PAN field is not included the three bits  $\{d(478), d(479), d(480)\}$  are added bitwise modulo 2 to the last three parity bits  $\{p(5), p(6), p(7)\}$  as specified in clause 5.1.9.1.3. In case an eTFI field and a PAN field are included the three bits  $\{d(503), d(504), d(505)\}$  are added bit-wise modulo 2 to the last three parity bits  $\{p(5), p(6), p(7)\}$  as specified in clause 5.1.9.1.3.

# 5.1a.16.4 Data coding

The data, defined as

$$i(k) = d(k+28)$$
 for  $k = 0,...,449$ 

is coded as defined in subclause 5.1a.1.3, with N=450, resulting in a coded block of 1398 bits, {C(0),...,C(1397)}.

The coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied.

The parameter values used for rate matching are: swap=0.05, N=466,  $N_{data}=1248$  and  $N_{data2}=1172$ .

P1 puncturing is generated according to 5.1a.1.3.5

P2 (Type 1) puncturing is generated according to 5.1a.1.3.5.

If a PAN is not included, the result is a block of 1248 bits,  $\{c(0),...,c(1247)\}$ .

If a PAN is included, the result is a block of 1172 bits,  $\{c(0),...,c(1171)\}$ .

# 5.1a.16.5 PAN coding

The PAN, if included, is defined as

$$pn(i) = d(478+i)$$
 for  $i=0,...,24$ .

The PAN coding is the same as for MCS-5 DL as specified in subclause 5.1.9.1.4a.

## 5.1a.16.6 Interleaving

a) Header

The header interleaving is the same as for MCS-5 DL as specified in subclause 5.1.9.1.5, resulting in a block of 100 bits, {hi(0),...,hi(99)}.

#### b) Data and PAN

If a PAN is not included, the following rule applies:

$$dc(k) = c(k)$$
 for  $k = 0,...,1247$ 

If a PAN is included, data and PAN are put together as one entity as described by the following rule:

$$dc(k) = ac(k)$$
 for  $k = 0,..., 75$ 

$$dc(k) = c(k-76)$$
 for  $k = 76,...,1247$ 

The block  $\{dc(0),...,dc(1247)\}$  is interleaved as for MCS-5 DL as specified in subclause 5.1.9.1.5, resulting in a block of 1248 bits,  $\{di(0),...,di(1247)\}$ .

## 5.1a.16.7 Mapping on a burst

The mapping is the same as for MCS-5 DL as specified in subclause 5.1.9.1.6.

NOTE: In this case, the stealing flags q(0),q(1),...,q(7) = 0,0,0,0,0,0,0,0 identify the coding scheme DAS-5, DAS-6 or DAS-7.

# 5.1a.17 Packet data block type 28 (DAS-6)

#### 5.1a.17.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 574 information bits  $\{d(0),d(1),...,d(573)\}$ . If the message delivered to the encoder does not include a PAN but includes an eTFI, it has a fixed size of 577 information bits  $\{d(0),d(1),...,d(576)\}$ . If the message delivered to the encoder includes a PAN but does not include an eTFI, it has a fixed size of 599 information bits  $\{d(0),d(1),...,d(598)\}$ . If the message delivered to the encoder includes a PAN and an eTFI, it has a fixed size of 602 information bits  $\{d(0),d(1),...,d(601)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

#### 5.1a.17.2 USF coding

A block of 36 bits {u'(0),u'(1),...,u'(35)} is derived as described for MCS-5 DL in subclause 5.1.9.1.2.

#### 5.1a.17.3 Header coding

A block of 100 coded bits  $\{hc(0),hc(1),...,hc(99)\}$  is derived from  $\{d(3),d(4),...,d(27)\}$  as described for MCS-5 DL in subclause 5.1.9.1.3.

In case an eTFI field is included and a PAN field is not included the three bits  $\{d(574),d(575),d(576)\}$  are added bitwise modulo 2 to the last three parity bits  $\{p(5),p(6),p(7)\}$  as specified in clause 5.1.9.1.3. In case an eTFI field and a PAN field are included the three bits  $\{d(599),d(600),d(601)\}$  are added bit-wise modulo 2 to the last three parity bits  $\{p(5),p(6),p(7)\}$  as specified in clause 5.1.9.1.3.

# 5.1a.17.4 Data coding

The data, defined as

$$i(k) = d(k+28)$$
 for  $k = 0,...,545$ 

is coded as defined in subclause 5.1a.1.3, with N=546, resulting in a coded block of 1686 bits, {C(0),...,C(1685)}.

The coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied.

The parameter values used for rate matching are: swap=0, N=562,  $N_{data}=1248$  and  $N_{data2}=1172$ .

P1 puncturing is generated according to 5.1a.1.3.5

P2 (Type 1) puncturing is generated according to 5.1a.1.3.5.

If a PAN is not included, the result is a block of 1248 bits,  $\{c(0),...,c(1247)\}$ .

If a PAN is included, the result is a block of 1172 bits,  $\{c(0),...,c(1171)\}$ .

### 5.1a.17.5 PAN coding

The PAN, if included, is defined as

$$pn(i) = d(574+i)$$
 for  $i=0,...,24$ .

The PAN coding is the same as for MCS-5 DL as specified in subclause 5.1.9.1.4a.

# 5.1a.17.6 Interleaving

The interleaving is the same as for DAS-5 as specified in subclause 5.1a.16.6.

# 5.1a.17.7 Mapping on a burst

The mapping is the same as for MCS-5 DL as specified in subclause 5.1.9.1.6.

NOTE: In this case, the stealing flags q(0),q(1),...,q(7) = 0,0,0,0,0,0,0,0 identify the coding scheme DAS-5, DAS-6 or DAS-7.

# 5.1a.18 Packet data block type 29 (DAS-7)

#### 5.1a.18.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 686 information bits  $\{d(0),d(1),...,d(685)\}$ . If the message delivered to the encoder does not include a PAN but includes an eTFI, it has a fixed size of 689 information bits  $\{d(0),d(1),...,d(688)\}$ . If the message delivered to the encoder includes a PAN but does not include an eTFI, it has a fixed size of 711 information bits  $\{d(0),d(1),...,d(710)\}$ . If the message delivered to the encoder includes a PAN and an eTFI, it has a fixed size of 714 information bits  $\{d(0),d(1),...,d(713)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

# 5.1a.18.2 USF coding

A block of 36 bits {u'(0),u'(1),...,u'(35)} is derived as described for MCS-5 DL in subclause 5.1.9.1.2.

# 5.1a.18.3 Header coding

A block of 100 coded bits  $\{hc(0),hc(1),...,hc(99)\}$  is derived from  $\{d(3),d(4),...,d(27)\}$  as described for MCS-5 DL in subclause 5.1.9.1.3.

In case an eTFI field is included and a PAN field is not included the three bits  $\{d(686),d(687),d(688)\}$  are added bitwise modulo 2 to the last three parity bits  $\{p(5),p(6),p(7)\}$  as specified in clause 5.1.9.1.3. In case an eTFI field and a PAN field are included the three bits  $\{d(711),d(712),d(713)\}$  are added bit-wise modulo 2 to the last three parity bits  $\{p(5),p(6),p(7)\}$  as specified in clause 5.1.9.1.3.

### 5.1a.18.4 Data coding

The data, defined as

$$i(k) = d(k+28)$$
 for  $k = 0,....657$ 

is coded as defined in subclause 5.1a.1.3, with N=658, resulting in a coded block of 2022 bits, {C(0),...,C(2021)}.

The coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied.

The parameter values used for rate matching are: swap=0, N=674,  $N_{data}=1248$  and  $N_{data}=1172$ .

P1 puncturing is generated according to 5.1a.1.3.5

P2 (Type 1) puncturing is generated according to 5.1a.1.3.5.

If a PAN is not included, the result is a block of 1248 bits,  $\{c(0),...,c(1247)\}$ .

If a PAN is included, the result is a block of 1172 bits,  $\{c(0),...,c(1171)\}$ .

### 5.1a.18.5 PAN coding

The PAN, if included, is defined as

$$pn(i) = d(686+i)$$
 for  $i=0,...,24$ .

The PAN coding is the same as for MCS-5 DL as specified in subclause 5.1.9.1.4a.

# 5.1a.18.6 Interleaving

The interleaving is the same as for DAS-5 as specified in subclause 5.1a.16.6.

# 5.1a.18.7 Mapping on a burst

The mapping is the same as for MCS-5 DL as specified in subclause 5.1.9.1.6.

NOTE: In this case, the stealing flags q(0),q(1),...,q(7) = 0,0,0,0,0,0,0,0 identify the coding scheme DAS-5, DAS-6 or DAS-7.

# 5.1a.19 Packet data block type 30 (DAS-8)

#### 5.1a.19.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 941 information bits  $\{d(0),d(1),...,d(940)\}$ . If the message delivered to the encoder does not include a PAN but includes an eTFI, it has a fixed size of 944 information bits  $\{d(0),d(1),...,d(943)\}$ . If the message delivered to the encoder includes a PAN but does not include an eTFI, it has a fixed size of 966 information bits  $\{d(0),d(1),...,d(965)\}$ . If the message delivered to the encoder includes a PAN and an eTFI, it has a fixed size of 969 information bits  $\{d(0),d(1),...,d(968)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$u(k) = d(k)$$
 for  $k = 0,...,2$   
 $h(k-3) = d(k)$  for  $k = 3,...,40$   
 $i1(k-41) = d(k)$  for  $k = 41,...,490$   
 $i2(k-491) = d(k)$  for  $k = 491,...,940$ 

And if an eTFI is included:

$$et(k-941) = d(k)$$
 for  $k = 941,...,943$ 

And if a PAN is included:

$$pn(k-941) = d(k)$$
 for  $k = 941,...,965$ 

And if a PAN and an eTFI are included:

$$et(k-966) = d(k)$$
 for  $k = 966,...,968$ 

# 5.1a.19.2 USF coding

#### 5.1a.19.2.1 BTTI configuration

The USF bits  $\{u(0),u(1),u(2)\}$  are block coded into 48 bits u'(0),u'(1),...,u'(47) according to the following table:

u(0),u(1),u(2)	u'(0),u'(1),,u'(47)			
	burst 0	burst 1	burst 2	burst 3
000	011111111111	111111111011	001101110111	111110111011
001	001100110111	001101110111	001100110111	001100110111
010	011101111111	001100110111	111111111011	001101111111
011	111110110011	101100110011	011101111111	101110110011
100	001101110111	011111111111	101110110011	111111111011
101	101100110011	101110111011	011111111111	011111111111
110	111111111011	111110110011	101100110011	011101110111
111	101110111011	011101111111	111110111011	101100110011

# 5.1a.19.2.2 RTTI configurations

If the USF is sent in RTTI USF mode (see 3GPP TS 45.002) when data blocks are transmitted in RTTI configuration, then the USF bits  $\{u(0),u(1),u(2)\}$  are block coded into 48 bits u'(0),u'(1),...,u'(47) as described in subclause 5.1a.19.2.1.

If the USF is sent in BTTI USF mode (see 3GPP TS 45.002) when data blocks are transmitted in RTTI configuration, then the three bits of the USF to be sent on the lower numbered PDCH of a corresponding downlink PDCH-pair are block coded into 48 bits  $u_L(0), u_L(1), ..., u_L(47)$  as described in subclause 5.1a.19.2.1; the three bits of the USF to be sent on the higher numbered PDCH of a corresponding downlink PDCH-pair are block coded into 48 bits  $u_H(0), u_H(1), ..., u_H(47)$  as described in subclause 5.1a.19.2.1.

NOTE: If BTTI USF mode is used when sending data blocks in RTTI configuration, then u(0),u(1),u(2) need not contain a USF; in this case, they are ignored by the encoder. How the USFs are delivered to the encoder in this case is implementation dependent.

If the data block is sent in the first 10ms of a 20ms block period, then:

$$u'(j)=u_L(j), j=0...11$$

$$u'(j)=u_H(j-12), j=12...23$$

$$u'(j)=u_L(j-12), j=24...35$$

$$u'(j)=u_H(j-24)$$
  $j=36...47$ 

If the data block is sent in the second 10ms of a 20ms block period, then:

$$u'(j)=u_L(j+24), j=0...11$$

$$u'(j)=u_H(j+12), j=12...23$$

$$u'(j)=u_L(j+12), j=24...35$$

$$u'(j)=u_H(j)$$
  $j=36...47$ 

NOTE: In case mixed modulation USF is used (see subclause 5.1), the USF bits sent during the other half of the 20 ms block period may be sent with a different modulation. In this case, the half of u<sub>L</sub> and u<sub>H</sub> not sent in the present data block will be discarded.

### 5.1a.19.3 Header coding

The header  $\{h(0),...,h(37)\}$  is coded as defined in subclause 5.1a.1.1, with N=38, resulting in a block of 138 bits,  $\{C(0),...,C(137)\}$ .

The code is punctured in such a way that the following coded bits:

 $\{C(k) \text{ for } k = 8 \text{ and } 77\}$  are not transmitted

This results in a block of 136 bits,  $\{hc(0),...,hc(135)\}$ .

# 5.1a.19.4 Data coding

Each data part,  $\{i1(0),...,i1(449)\}$  and  $\{i2(0),...,i2(449)\}$ , is coded as defined in subclause 5.1a.1.3, with N=450, resulting in two coded blocks of 1398 bits,  $\{C1(0),...,C1(1397)\}$  and  $\{C2(0),...,C2(1397)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied.

The parameter values used for rate matching are: swap=0, N=466,  $N_{data}=832$  and  $N_{data2}=793$ .

P1 puncturing is generated according to 5.1a.1.3.5

P2 (Type 1) puncturing is generated according to 5.1a.1.3.5.

If a PAN is not included, the result is two blocks of 832 bits,  $\{c1(0),...,c1(831)\}$  and  $\{c2(0),...,c2(831)\}$ .

If a PAN is included, the result is two blocks of 793 bits,  $\{c1(0),...,c1(792)\}$  and  $\{c2(0),...,c2(792)\}$ .

NOTE: C1 and c1 correspond to i1, and C2 and c2 to i2.

## 5.1a.19.5 PAN coding

The PAN coding is the same as for UAS-7 as specified in subclause 5.1a.3.4.

### 5.1a.19.6 Interleaving

a) Header

The header,  $\{hc(0),...,hc(135)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C=136$  and a=23, resulting in a block of 136 bits,  $\{hi(0),...,hi(135)\}$ .

b) Data and PAN

If a PAN is not included, data are put together as one entity as described by the following rule:

$$dc(k) = c1(k)$$
 for  $k = 0,...,831$   
 $dc(k) = c2(k-832)$  for  $k = 832,...,1663$ 

If a PAN is included, data and PAN are put together as one entity as described by the following rule:

$$dc(k) = ac(k)$$
 for  $k = 0,...,77$   
 $dc(k) = c1(k-78)$  for  $k = 78,...,870$   
 $dc(k) = c2(k-871)$  for  $k = 871,...,1663$ 

The block  $\{dc(0),...,dc(1663)\}$  is interleaved as defined in subclause 5.1a.2.1, with  $N_C=1664$  and a=199, resulting in a block of 1664 bits,  $\{di(0),...,di(1663)\}$ .

# 5.1a.19.7 Mapping on a burst

#### a) Straightforward mapping

The mapping is given by the rule:

For B=0,1,2,3, let

$$e(B,j) = di(416B+j)$$
 for  $j = 0,...,207$ 

$$e(B,j) = hi(34B+j-208)$$
 for  $j = 208,...,227$ 

$$e(B,j) = q(2B+j-228)$$
 for  $j = 228,229$ 

$$e(B,j) = hi(34B+j-210)$$
 for  $j = 230,...,231$ 

$$e(B,j) = u'(12B+j-232)$$
 for  $j = 232,...,243$ 

$$e(B,j) = hi(34B+j-222)$$
 for  $j = 244,...,255$ 

$$e(B,j) = di(416B+j-48)$$
 for  $j = 256,...,463$ 

where

q(0),q(1),...,q(7) = 0,0,0,0,0,0,0,0,0 identifies the coding scheme DAS-8 or DAS-9.

#### b) Bit swapping

After this mapping the following bits are swapped:

For 
$$B = 0,1,2,3$$
,

In RTTI configuration, the bursts with B=0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B=1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.

### c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

For B = 0

Swap 
$$e(B,55)$$
 with  $e(B,40)$ 

Swap 
$$e(B,91)$$
 with  $e(B,81)$ 

Swap 
$$e(B,127)$$
 with  $e(B,120)$ 

Swap 
$$e(B,302)$$
 with  $e(B,268)$ 

Swap 
$$e(B,338)$$
 with  $e(B,305)$ 

Swap 
$$e(B,374)$$
 with  $e(B,368)$ 

Swap 
$$e(B,410)$$
 with  $e(B,405)$ 

Swap 
$$e(B,446)$$
 with  $e(B,444)$ 

For 
$$B = 1$$

- Swap e(B,23) with e(B,120)
- Swap e(B,59) with e(B,160)
- Swap e(B,95) with e(B,177)
- Swap e(B,150) with e(B,12)
- Swap e(B,186) with e(B,81)
- Swap e(B,230) with e(B,181)
- Swap e(B,415) with e(B,268)
- Swap e(B,451) with e(B,305)
- Swap e(B,270) with e(B,368)
- Swap e(B,306) with e(B,405)
- Swap e(B,342) with e(B,444)

#### For B = 2

- Swap e(B,46) with e(B,40)
- Swap e(B,82) with e(B,81)
- Swap e(B,118) with e(B,120)
- Swap e(B,154) with e(B,160)
- Swap e(B,190) with e(B,177)
- Swap e(B,311) with e(B,268)
- Swap e(B,347) with e(B,305)
- Swap e(B,383) with e(B,368)
- Swap e(B,419) with e(B,405)
- Swap e(B,455) with e(B,444)

#### For B = 3

- Swap e(B,14) with e(B,120)
- Swap e(B,50) with e(B,160)
- Swap e(B,86) with e(B,180)
- Swap e(B,159) with e(B,40)
- Swap e(B,195) with e(B,81)
- Swap e(B,406) with e(B,268)
- Swap e(B,442) with e(B,305)
- Swap e(B,279) with e(B,368)
- Swap e(B,315) with e(B,405)
- Swap e(B,351) with e(B,444)

# 5.1a.20 Packet data block type 31 (DAS-9)

for k = 587,...,1132

#### 5.1a.20.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 1133 information bits  $\{d(0),d(1),...,d(1132)\}$ . If the message delivered to the encoder does not include a PAN but includes an eTFI, it has a fixed size of 1136 information bits  $\{d(0),d(1),...,d(1135)\}$ . If the message delivered to the encoder includes a PAN but does not include an eTFI, it has a fixed size of 1158 information bits  $\{d(0),d(1),...,d(1157)\}$ . If the message delivered to the encoder includes a PAN and an eTFI, it has a fixed size of 1161 information bits  $\{d(0),d(1),...,d(1160)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$u(k) = d(k)$$
 for  $k = 0,...,2$   
 $h(k-3) = d(k)$  for  $k = 3,...,40$   
 $i1(k-41) = d(k)$  for  $k = 41,...,586$ 

And if an eTFI is included:

i2(k-587) = d(k)

$$et(k-1133) = d(k)$$
 for  $k = 1133,...,1135$ 

And if a PAN is included:

$$pn(k-1133) = d(k)$$
 for  $k = 1133,...,1157$ 

And if a PAN and an eTFI are included:

$$et(k-1158) = d(k)$$
 for  $k = 1158,...,1160$ 

# 5.1a.20.2 USF coding

The USF coding is the same as for DAS-8 as specified in subclause 5.1a.19.2.

#### 5.1a.20.3 Header coding

The header coding is the same as for DAS-8 as specified in subclause 5.1a.19.3.

# 5.1a.20.4 Data coding

Each data part,  $\{i1(0),...,i1(545)\}$  and  $\{i2(0),...,i2(545)\}$ , is coded as defined in subclause 5.1a.1.3, with N=546, resulting in two coded blocks of 1686 bits,  $\{C1(0),...,C1(1685)\}$  and  $\{C2(0),...,C2(1685)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied.

The parameter values used for rate matching are: swap=0, N=562,  $N_{data}=832$  and  $N_{data2}=793$ .

P1 puncturing is generated according to 5.1a.1.3.5

P2 (Type 2) puncturing is generated according to 5.1a.1.3.5.

P3 puncturing is generated according to 5.1a.1.3.5.

If a PAN is not included, the result is two blocks of 832 bits,  $\{c1(0),...,c1(831)\}$  and  $\{c2(0),...,c2(831)\}$ .

If a PAN is included, the result is two blocks of 793 bits,  $\{c1(0),...,c1(792)\}\$  and  $\{c2(0),...,c2(792)\}\$ .

NOTE: C1 and c1 correspond to i1, and C2 and c2 to i2.

# 5.1a.20.5 PAN coding

The PAN coding is the same as for UAS-7 as specified in subclause 5.1a.3.4.

# 5.1a.20.6 Interleaving

The interleaving is the same as for DAS-8 as specified in subclause 5.1a.19.6.

# 5.1a.20.7 Mapping on a burst

The mapping is the same as for DAS-8 as specified in subclause 5.1a.19.7.

# 5.1a.21 Packet data block type 32 (DAS-10)

### 5.1a.21.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 1355 information bits  $\{d(0),d(1),...,d(1354)\}$ . If the message delivered to the encoder does not include a PAN but includes an eTFI, it has a fixed size of 1358 information bits  $\{d(0),d(1),...,d(1357)\}$ . If the message delivered to the encoder includes a PAN but does not include an eTFI, it has a fixed size of 1380 information bits  $\{d(0),d(1),...,d(1379)\}$ . If the message delivered to the encoder includes a PAN and an eTFI, it has a fixed size of 1383 information bits  $\{d(0),d(1),...,d(1382)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$u(k) = d(k)$$
 for  $k = 0,...,2$   
 $h(k-3) = d(k)$  for  $k = 3,...,38$   
 $i1(k-39) = d(k)$  for  $k = 39,...,696$ 

$$i2(k-697) = d(k)$$
 for  $k = 697,...,1354$ 

And if an eTFI is included:

$$et(k-1355) = d(k)$$
 for  $k = 1355,...,1357$ 

And if a PAN is included:

$$pn(k-1355) = d(k)$$
 for  $k = 1355,...,1379$ 

And if a PAN and an eTFI are included:

$$et(k-1380) = d(k)$$
 for  $k = 1380,...,1382$ 

# 5.1a.21.2 USF coding

# 5.1a.21.2.1 BTTI configuration

The USF bits  $\{u(0),u(1),u(2)\}$  are block coded into 60 bits u'(0),u'(1),...,u'(59) according to the following table:

u(0),u(1),u(2)	u'(0),u'(1),,u'(59)			
	burst 0	burst 1	burst 2	burst 3
000	0011000000000000	000000000010100	100100011000110	000001010010100
001	100101001000110	100100011000110	100101001000110	100101001000110
010	001100011000000	100101001000110	000000000010100	100100011000000
011	000001010010010	101001001010010	001100011000000	101001010010010
100	100100011000110	0011000000000000	101001010010010	000000000010100
101	000000000010100	000001010010010	101001001010000	001100011000110
110	101001001010010	101001010010100	0011000000000000	0011000000000000
111	101001010010100	00110001100000	000001010010101	101001001010010

# 5.1a.21.2.2 RTTI configurations

If the USF is sent in RTTI USF mode (see 3GPP TS 45.002) when data blocks are transmitted in RTTI configuration, then the USF bits  $\{u(0),u(1),u(2)\}$  are block coded into 60 bits u'(0),u'(1),...,u'(59) as described in subclause 5.1a.21.2.1.

If the USF is sent in BTTI USF mode (see 3GPP TS 45.002) when data blocks are transmitted in RTTI configuration, then the three bits of the USF to be sent on the lower numbered PDCH of a corresponding downlink PDCH-pair are block coded into 60 bits  $u_L(0)$ , $u_L(1)$ ,..., $u_L(59)$  as described in subclause 5.1a.21.2.1; the three bits of the USF to be sent on the higher numbered PDCH of a corresponding downlink PDCH-pair are block coded into 60 bits  $u_H(0)$ , $u_H(1)$ ,..., $u_H(59)$  as described in subclause 5.1a.21.2.1.

NOTE: If BTTI USF mode is used when sending data blocks in RTTI configuration, then u(0),u(1),u(2) need not contain a USF; in this case, they are ignored by the encoder. How the USFs are delivered to the encoder in this case is implementation dependent.

If the data block is sent in the first 10ms of a 20ms block period, then:

$$u'(j)=u_L(j),$$
  $j=0...14$   $u'(j)=u_H(j-15),$   $j=15...29$   $u'(j)=u_L(j-15),$   $j=30...44$   $u'(j)=u_H(j-30)$   $j=45...59$ 

If the data block is sent in the second 10ms of a 20ms block period, then:

$$u'(j)=u_L(j+30), \quad j=0...14$$
  
 $u'(j)=u_H(j+15), \quad j=15...29$   
 $u'(j)=u_L(j+15), \quad j=30...44$   
 $u'(j)=u_H(j) \qquad j=45...59$ 

NOTE: In case mixed modulation USF is used (see subclause 5.1), the USF bits sent during the other half of the 20 ms block period may be sent with a different modulation. In this case, the half of  $u_L$  and  $u_H$  not sent in the present data block will be discarded.

### 5.1a.21.3 Header coding

The header  $\{h(0),...,h(35)\}$  is coded as defined in subclause 5.1a.1.1, with N=36, resulting in a block of 132 bits,  $\{C(0),...,C(131)\}$ .

The coded header is defined as:

$$hc(k) = C(k)$$
 for  $k = 0,...,131$ 

### 5.1a.21.4 Data coding

Each data part,  $\{i1(0),...,i1(657)\}$  and  $\{i2(0),...,i2(657)\}$ , is coded as defined in subclause 5.1a.1.3, with N=658, resulting in two coded blocks of 2022 bits,  $\{C1(0),...,C1(2021)\}$  and  $\{C2(0),...,C2(2021)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied.

The parameter values used for rate matching are: swap=0.15, N=674,  $N_{data}=1060$  and  $N_{data}=1021$ .

P1 puncturing is generated according to 5.1a.1.3.5

P2 (Type 1) puncturing is generated according to 5.1a.1.3.5

If a PAN is not included, the result is two blocks of 1060 bits,  $\{c1(0),...,c1(1059)\}$  and  $\{c2(0),...,c2(1059)\}$ .

If a PAN is included, the result is two blocks of 1021 bits, {c1(0),...,c1(1020)} and {c2(0),...,c2(1020)}.

NOTE: C1 and c1 correspond to i1, and C2 and c2 to i2.

### 5.1a.21.5 PAN coding

The PAN coding is the same as for UAS-7 as specified in subclause 5.1a.3.4.

## 5.1a.21.6 Interleaving

a) Header

The header,  $\{hc(0),...,hc(131)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C$ =132 and a=7, resulting in a block of 132 bits,  $\{hi(0),...,hi(131)\}$ .

b) Data and PAN

If a PAN is not included, data are put together as one entity as described by the following rule:

$$dc(k) = c1(k)$$
 for  $k = 0,...,1059$   
 $dc(k) = c2(k-1060)$  for  $k = 1060,...,2119$ 

If a PAN is included, data and PAN are put together as one entity as described by the following rule:

$$dc(k) = ac(k)$$
 for  $k = 0,...,77$   
 $dc(k) = c1(k-78)$  for  $k = 78,...,1098$   
 $dc(k) = c2(k-1099)$  for  $k = 1099,...,2119$ 

The block  $\{dc(0),...,dc(2119)\}$  is interleaved as defined in subclause 5.1a.2.1, with  $N_C$ =2120 and a=301, resulting in a block of 2120 bits,  $\{di(0),...,di(2119)\}$ .

# 5.1a.21.7 Mapping on a burst

a) Straightforward mapping

The mapping is given by the rule:

For B=0,1,2,3, let 
$$e(B,j) = di(530B+j) \qquad \text{for } j = 0,...,264$$

$$e(B,j) = hi(33B+j-265)$$
 for  $j = 265,...,284$ 

$$e(B,j) = q(2B+j-285)$$
 for  $j = 285$ 

$$e(B,j) = hi(33B+j-266)$$
 for  $j = 286,...,287$ 

$$e(B,j) = q(2B+j-287)$$
 for  $j = 288$ 

$$e(B,j) = hi(33B+j-267)$$
 for  $j = 289$ 

$$e(B,j) = u'(15B+j-290)$$
 for  $j = 290,...,304$ 

$$e(B,j) = hi(33B+j-282)$$
 for  $j = 305,...,314$ 

$$e(B,j) = di(530B+j-50)$$
 for  $j = 315,...,579$ 

where

q(0),q(1),...,q(7) = 0,0,0,0,0,0,0,0,0 identifies the coding scheme DAS-10.

# b) Bit swapping

After this mapping the following bits are swapped:

For 
$$B = 0,1,2,3$$
,

Swap e(B,240+k) with e(B,266+k) for k=0, 3, 5, 8, 10, 13, 15, 18, 20, 23.

Swap e(B,225+k) with e(B,267+k) for k=0, 5, 10.

Swap e(B,233+k) with e(B,282+k) for k=0, 5.

Swap e(B,315+k) with e(B,306+k) for k=0, 3, 5, 8.

Swap e(B,328) with e(B,312), Swap e(B,325) with e(B,307).

In RTTI configuration, the bursts with B=0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B=1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.

### c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

For B = 0

Swap e(B,46) with e(B,188)

Swap e(B,59) with e(B,158)

Swap e(B,72) with e(B,80)

Swap e(B,131) with e(B,170)

Swap e(B,144) with e(B,98)

Swap e(B,216) with e(B,110)

Swap e(B,307) with e(B,385)

Swap e(B,351) with e(B,330)

Swap e(B,397) with e(B,480)

Swap e(B,469) with e(B,503)

Swap e(B,482) with e(B,400)

- Swap e(B,554) with e(B,433)
- Swap e(B,567) with e(B,363)
- For B = 1
  - Swap e(B,41) with e(B,110)
  - Swap e(B,54) with e(B,80)
  - Swap e(B,126) with e(B,98)
  - Swap e(B,257) with e(B,158)
  - Swap e(B,311) with e(B,450)
  - Swap e(B,379) with e(B,530)
  - Swap e(B,392) with e(B,433)
  - Swap e(B,464) with e(B,480)
  - Swap e(B,477) with e(B,363)
  - Swap e(B,536) with e(B,503)
  - Swap e(B,549) with e(B,400)
  - Swap e(B,562) with e(B,330)

#### For B = 2

- Swap e(B,36) with e(B,80)
- Swap e(B,82) with e(B,170)
- Swap e(B,167) with e(B,110)
- Swap e(B,239) with e(B,158)
- Swap e(B,252) with e(B,98)
- Swap e(B,306) with e(B,450)
- Swap e(B,361) with e(B,530)
- Swap e(B,374) with e(B,480)
- Swap e(B,387) with e(B,363)
- Swap e(B,446) with e(B,503)
- Swap e(B,459) with e(B,400)
- Swap e(B,531) with e(B,433)
- Swap e(B,544) with e(B,330)

#### For B = 3

- Swap e(B,64) with e(B,200)
- Swap e(B,77) with e(B,158)
- Swap e(B,149) with e(B,170)
- Swap e(B,162) with e(B,98)
- Swap e(B,221) with e(B,188)

Swap e(B,234) with e(B,110)

Swap e(B,247) with e(B,80)

Swap e(B,356) with e(B,433)

Swap e(B,369) with e(B,363)

Swap e(B,441) with e(B,400)

Swap e(B,526) with e(B,330)

Swap e(B,572) with e(B,480)

# 5.1a.22 Packet data block type 33 (DAS-11)

#### 5.1a.22.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 1691 information bits  $\{d(0),d(1),...,d(1690)\}$ . If the message delivered to the encoder does not include a PAN but includes an eTFI, it has a fixed size of 1694 information bits  $\{d(0),d(1),...,d(1693)\}$ . If the message delivered to the encoder includes a PAN but does not include an eTFI, it has a fixed size of 1716 information bits  $\{d(0),d(1),...,d(1715)\}$ . If the message delivered to the encoder includes a PAN and an eTFI, it has a fixed size of 1719 information bits  $\{d(0),d(1),...,d(1718)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$u(k) = d(k)$$
 for  $k = 0,...,2$ 

h(k-3) = d(k) for k = 3,...,52

i1(k-53) = d(k) for k = 53,...,598

i2(k-599) = d(k) for k = 599,...,1144

i3(k-1145) = d(k) for k = 1145,...,1690

And if an eTFI is included:

$$et(k-1691) = d(k)$$
 for  $k = 1691,...,1693$ 

And if a PAN is included:

$$pn(k-1691) = d(k)$$
 for  $k = 1691,...,1715$ 

And if a PAN and an eTFI are included:

$$et(k-1716) = d(k)$$
 for  $k = 1716,...,1718$ 

# 5.1a.22.2 USF coding

The USF coding is the same as for DAS-10 as specified in subclause 5.1a.21.2.

### 5.1a.22.3 Header coding

The header  $\{h(0),...,h(49)\}$  is coded as defined in subclause 5.1a.1.1, with N=50, resulting in a block of 174 bits,  $\{C(0),...,C(173)\}$ .

The code is punctured in such a way that the following coded bits:

$$\{C(15*k+j) \text{ for } k = 0,...,10, j = 11 \text{ and } 14\}$$
 are not transmitted

This results in a block of 152 bits,  $\{hc(0),...,hc(151)\}$ .

# 5.1a.22.4 Data coding

Each data part,  $\{i1(0),...,i1(545)\}$ ,  $\{i2(0),...,i2(545)\}$  and  $\{i3(0),...,i3(545)\}$ , is coded as defined in subclause 5.1a.1.3, with N=546, resulting in three coded blocks of 1686 bits,  $\{C1(0),...,C1(1685)\}$ ,  $\{C2(0),...,C2(1685)\}$  and  $\{C3(0),...,C3(1685)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied.

The parameter values used for rate matching are: swap=0, N=562,  $N_{data}=700$  and  $N_{data2}=674$ .

P1 puncturing is generated according to 5.1a.1.3.5

P2 (Type 2) puncturing is generated according to 5.1a.1.3.5.

P3 puncturing is generated according to 5.1a.1.3.5.

If a PAN is not included, the result is three blocks of 700 bits,  $\{c1(0),...,c1(699)\}$ ,  $\{c2(0),...,c2(699)\}$  and  $\{c3(0),...,c3(699)\}$ .

If a PAN is included, the result is three blocks of 674 bits,  $\{c1(0),...,c1(673)\}$ ,  $\{c2(0),...,c2(673)\}$  and  $\{c3(0),...,c3(673)\}$ .

NOTE: C1 and c1 correspond to i1, C2 and c2 to i2 and C3 and c3 to i3.

# 5.1a.22.5 PAN coding

The PAN coding is the same as for UAS-7 as specified in subclause 5.1a.3.4.

# 5.1a.22.6 Interleaving

a) Header

The header,  $\{hc(0),...,hc(151)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C=152$  and a=3, resulting in a block of 152 bits,  $\{hi(0),...,hi(151)\}$ .

b) Data and PAN

If a PAN is not included, data are put together as one entity as described by the following rule:

$$dc(k) = c1(k)$$
 for  $k = 0,...,699$   
 $dc(k) = c2(k-700)$  for  $k = 700,...,1399$   
 $dc(k) = c3(k-1400)$  for  $k = 1400,...,2099$ 

If a PAN is included, data and PAN are put together as one entity as described by the following rule:

$$\begin{aligned} &dc(k) = ac(k) & for \ k = 0,...,77 \\ &dc(k) = c1(k-78) & for \ k = 78,...,751 \\ &dc(k) = c2(k-752) & for \ k = 752,...,1425 \\ &dc(k) = c3(k-1426) & for \ k = 1426,...,2099 \end{aligned}$$

The block  $\{dc(0),...,dc(2099)\}$  is interleaved as defined in subclause 5.1a.2.1, with  $N_C$ =2100 and a=47, resulting in a block of 2100 bits,  $\{di(0),...,di(2099)\}$ .

# 5.1a.22.7 Mapping on a burst

### a) Straightforward mapping

The mapping is given by the rule:

For B=0,1,2,3, let

$$e(B,j) = di(525B+j)$$
 for  $j = 0,...,259$ 

$$e(B,j) = hi(38B+j-260)$$
 for  $j = 260,...,284$ 

$$e(B,j) = q(2B+j-285)$$
 for  $j = 285$ 

$$e(B,j) = hi(38B+j-261)$$
 for  $j = 286,...,287$ 

$$e(B,j) = q(2B+j-287)$$
 for  $j = 288$ 

$$e(B,j) = hi(38B+j-262)$$
 for  $j = 289$ 

$$e(B,j) = u'(15B+j-290)$$
 for  $j = 290,...,304$ 

$$e(B,j) = hi(38B+j-277)$$
 for  $j = 305,...,314$ 

$$e(B,j) = di(525B+j-55)$$
 for  $j = 315,...,579$ 

where

$$q(0),q(1),...,q(7) = 1,1,1,1,1,1,1,1,1$$
 identifies the coding scheme DAS-11 or DAS-12.

#### b) Bit swapping

After this mapping the following bits are swapped:

For 
$$B = 0,1,2,3$$
,

Swap e(B,215+k) with e(B,262+k) for k=0, 5, 10.

Swap e(B,218+k) with e(B,277+k) for k=0, 5, 10.

Swap e(B,315+k) with e(B,306+k) for k=0, 3, 5, 8.

Swap e(B,325) with e(B,312), Swap e(B,328) with e(B,307).

In RTTI configuration, the bursts with B = 0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B = 1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.

# c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

For B = 0

Swap e(B,39) with e(B,158)

Swap e(B,86) with e(B,170)

Swap e(B,94) with e(B,98)

Swap e(B,141) with e(B,110)

Swap e(B,266) with e(B,200)

Swap e(B,329) with e(B,480)

Swap e(B,376) with e(B,503)

Swap e(B,384) with e(B,363)

Swap e(B,431) with e(B,400)

#### For B = 1

Swap e(B,84) with e(B,170)

Swap e(B,131) with e(B,188)

Swap e(B,139) with e(B,98)

Swap e(B,186) with e(B,110)

Swap e(B,264) with e(B,200)

Swap e(B,429) with e(B,480)

Swap e(B,476) with e(B,503)

Swap e(B,484) with e(B,363)

Swap e(B,531) with e(B,400)

#### For B = 2

Swap e(B,184) with e(B,110)

Swap e(B,231) with e(B,158)

Swap e(B,239) with e(B,80)

Swap e(B,341) with e(B,330)

Swap e(B,474) with e(B,503)

Swap e(B,482) with e(B,363)

Swap e(B,521) with e(B,530)

Swap e(B,529) with e(B,400)

Swap e(B,576) with e(B,433)

# For B = 3

Swap e(B,41) with e(B,110)

Swap e(B,96) with e(B,80)

Swap e(B,229) with e(B,170)

Swap e(B,331) with e(B,573)

Swap e(B,339) with e(B,330)

Swap e(B,386) with e(B,363)

Swap e(B,574) with e(B,530)

# 5.1a.23 Packet data block type 34 (DAS-12)

#### 5.1a.23.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 2027 information bits  $\{d(0),d(1),...,d(2026)\}$ . If the message delivered to the encoder does not include a PAN but includes an eTFI, it has a fixed size of 2030 information bits  $\{d(0),d(1),...,d(2029)\}$ . If the message delivered to the encoder includes a PAN but does not include an eTFI, it has a fixed size of 2052 information bits  $\{d(0),d(1),...,d(2051)\}$ . If the message delivered to the encoder includes a PAN and an eTFI, it has a fixed size of 2055 information bits  $\{d(0),d(1),...,d(2054)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$u(k) = d(k)$$
 for  $k = 0,...,2$   
 $h(k-3) = d(k)$  for  $k = 3,...,52$   
 $i1(k-53) = d(k)$  for  $k = 53,...,710$   
 $i2(k-711) = d(k)$  for  $k = 711,...,1368$   
 $i3(k-1369) = d(k)$  for  $k = 1369,...,2026$ 

And if an eTFI is included:

$$et(k-2027) = d(k)$$
 for  $k = 2027,...,2029$ 

And if a PAN is included:

$$pn(k-2027) = d(k)$$
 for  $k = 2027,...,2051$ 

And if a PAN and an eTFI are included:

$$et(k-2052) = d(k)$$
 for  $k = 2052,...,2054$ 

# 5.1a.23.2 USF coding

The USF coding is the same as for DAS-10 as specified in subclause 5.1a.21.2.

# 5.1a.23.3 Header coding

The header coding is the same as for DAS-11 as specified in subclause 5.1a.22.3.

### 5.1a.23.4 Data coding

Each data part,  $\{i1(0),...,i1(657)\}$ ,  $\{i2(0),...,i2(657)\}$  and  $\{i3(0),...,i3(657)\}$ , is coded as defined in subclause 5.1a.1.3, with N=658, resulting in three coded blocks of 2022 bits,  $\{C1(0),...,C1(2021)\}$ ,  $\{C2(0),...,C2(2021)\}$  and  $\{C3(0),...,C3(2021)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied.

The parameter values used for rate matching are: swap=0, N=674,  $N_{data}=700$  and  $N_{data2}=674$ .

P1 puncturing is generated according to 5.1a.1.3.5

P2 (Type 2) puncturing is generated according to 5.1a.1.3.5.

P3 puncturing is generated according to 5.1a.1.3.5.

If a PAN is not included, the result is three blocks of 700 bits,  $\{c1(0),...,c1(699)\}$ ,  $\{c2(0),...,c2(699)\}$  and  $\{c3(0),...,c3(699)\}$ .

If a PAN is included, the result is three blocks of 674 bits,  $\{c1(0),...,c1(673)\}$ ,  $\{c2(0),...,c2(673)\}$  and  $\{c3(0),...,c3(673)\}$ .

NOTE: C1 and c1 correspond to i1, C2 and c2 to i2 and C3 and c3 to i3...

# 5.1a.23.5 PAN coding

The PAN coding is the same as for UAS-7 as specified in subclause 5.1a.3.4.

# 5.1a.23.6 Interleaving

a) Header

The header,  $\{hc(0),...,hc(151)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C=152$  and a=3, resulting in a block of 152 bits,  $\{hi(0),...,hi(151)\}$ .

b) Data and PAN

If a PAN is not included, the following definitions apply:

$$dc1(k) = c1(k)$$
 for  $k = 0,...,699$   
 $dc2(k) = c2(k)$  for  $k = 0,...,699$   
 $dc3(k) = c3(k)$  for  $k = 0,...,699$ 

If a PAN is included, the following definitions apply:

$$\begin{aligned} &\text{dc1}(k) = \text{ac}(3k) & \text{for } k = 0,...,25 \\ &\text{dc1}(k) = \text{c1}(k\text{-}26) & \text{for } k = 26,...,699 \\ &\text{dc2}(k) = \text{ac}(3k\text{+}1) & \text{for } k = 0,...,25 \\ &\text{dc2}(k) = \text{c2}(k\text{-}26) & \text{for } k = 26,...,699 \\ &\text{dc3}(k) = \text{ac}(3k\text{+}2) & \text{for } k = 0,...,25 \\ &\text{dc3}(k) = \text{c3}(k\text{-}26) & \text{for } k = 26,...,699 \end{aligned}$$

The three blocks  $\{dc1(0),...,dc1(699)\}$ ,  $\{dc2(0),...,dc2(699)\}$  and  $\{dc3(0),...,dc3(699)\}$  are separately interleaved as defined in subclause 5.1a.2.2, with N<sub>C</sub>=700 and a=129, resulting in the three blocks  $\{di1(0),...,di1(699)\}$ ,  $\{di2(0),...,di2(699)\}$  and  $\{di3(0),...,di3(699)\}$ , where di1 corresponds to dc1, di2 to dc2 and di3 to dc3.

The blocks are put together as one entity as described by the following rule:

$$\begin{aligned} &\text{di}(k) = \text{di}1(k) & &\text{for } k = 0,...,699 \\ &\text{di}(k) = \text{di}2(k\text{-}700) & &\text{for } k = 700,...,1399 \\ &\text{di}(k) = \text{di}3(k\text{-}1400) & &\text{for } k = 1400,...,2099 \end{aligned}$$

### 5.1a.23.7 Mapping on a burst

a) Straightforward mapping

The mapping is the same as for DAS-11 as specified in subclause 5.1a.22.7 a).

b) Bit swapping

The bit swapping is the same as for DAS-11 as specified in subclause 5.1a.22.7 b)

In RTTI configuration, the bursts with B = 0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B = 1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.

#### c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

# For B = 0

Swap e(B,19) with e(B,158)

Swap e(B,74) with e(B,110)

Swap e(B,129) with e(B,80)

Swap e(B,167) with e(B,188)

Swap e(B,289) with e(B,98)

Swap e(B,332) with e(B,503)

Swap e(B,351) with e(B,350)

Swap e(B,406) with e(B,493)

Swap e(B,461) with e(B,530)

Swap e(B,516) with e(B,433)

Swap e(B,571) with e(B,363)

#### For B = 1

Swap e(B,84) with e(B,98)

Swap e(B,139) with e(B,80)

Swap e(B,194) with e(B,158)

Swap e(B,249) with e(B,110)

Swap e(B,314) with e(B,480)

Swap e(B,359) with e(B,330)

Swap e(B,452) with e(B,530)

Swap e(B,526) with e(B,493)

#### For B = 2

Swap e(B,1) with e(B,188)

Swap e(B,56) with e(B,170)

Swap e(B,111) with e(B,110)

Swap e(B,166) with e(B,98)

Swap e(B,259) with e(B,200)

Swap e(B,271) with e(B,158)

Swap e(B,369) with e(B,330)

Swap e(B,424) with e(B,433)

Swap e(B,479) with e(B,400)

Swap e(B,534) with e(B,363)

Swap e(B,572) with e(B,480)

For B = 3

Swap e(B,121) with e(B,200)

Swap e(B,157) with e(B,98)

Swap e(B,176) with e(B,188)

Swap e(B,231) with e(B,158)

Swap e(B,281) with e(B,0)

Swap e(B,341) with e(B,363)

Swap e(B,396) with e(B,330)

Swap e(B,489) with e(B,433)

Swap e(B,544) with e(B,400)

# 5.1a.24 Packet data block type 35 (DBS-5)

### 5.1a.24.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 480 information bits  $\{d(0),d(1),...,d(479)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 505 information bits  $\{d(0),d(1),...,d(504)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$u(k) = d(k)$$
 for  $k = 0,...,2$ 

$$h(k-3) = d(k)$$
 for  $k = 3,...,29$ 

$$i(k-30) = d(k)$$
 for  $k = 30,...,479$ 

And if a PAN is included:

$$pn(k-480) = d(k)$$
 for  $k = 480,...,504$ 

# 5.1a.24.2 USF coding

# 5.1a.24.2.1 BTTI configuration

The USF bits  $\{u(0),u(1),u(2)\}$  are block coded into 32 bits u'(0),u'(1),...,u'(31) according to the following table:

u(0),u(1),u(2)	u'(0),u'(1),,u'(31)			
	burst 0	burst 1	burst 2	burst 3
000	00110011	00110011	00110011	00110011
001	11001100	00110011	00111100	11000011
010	11000011	11000011	11000011	11000011
011	00111100	11000011	11001100	00110011
100	11000011	00111100	00110011	11001100
101	00111100	00111100	00111100	00111100
110	00110011	11001100	11000011	00111100
111	11001100	11001100	11001100	11001100

# 5.1a.24.2.2 RTTI configurations

If the USF is sent in RTTI USF mode (see 3GPP TS 45.002) when data blocks are transmitted in RTTI configuration, then the USF bits  $\{u(0),u(1),u(2)\}$  are block coded into 32 bits u'(0),u'(1),...,u'(31) as described in subclause 5.1a.24.2.1.

If the USF is sent in BTTI USF mode (see 3GPP TS 45.002) when data blocks are transmitted in RTTI configuration, then the three bits of the USF to be sent on the lower numbered PDCH of a corresponding downlink PDCH-pair are block coded into 32 bits  $u_L(0), u_L(1), ..., u_L(31)$  as described in subclause 5.1a.24.2.1; the three bits of the USF to be sent on the higher numbered PDCH of a corresponding downlink PDCH-pair are block coded into 32 bits  $u_H(0), u_H(1), ..., u_H(31)$  as described in subclause 5.1a.24.2.1.

NOTE: If BTTI USF mode is used when sending data blocks in RTTI configuration, then u(0),u(1),u(2) need not contain a USF; in this case, they are ignored by the encoder. How the USFs are delivered to the encoder in this case is implementation dependent.

If the data block is sent in the first 10ms of a 20ms block period, then:

$$u'(j)=u_L(j),$$
  $j=0...7$   
 $u'(j)=u_H(j-8),$   $j=8...15$   
 $u'(j)=u_L(j-8),$   $j=16...23$   
 $u'(j)=u_H(j-16)$   $j=24...31$ 

If the data block is sent in the second 10ms of a 20ms block period, then:

$$u'(j)=u_L(j+16), \quad j=0...7$$
  
 $u'(j)=u_H(j+8), \quad j=8...15$   
 $u'(j)=u_L(j+8), \quad j=16...23$   
 $u'(j)=u_H(j) \quad j=24...31$ 

NOTE: In case mixed modulation USF is used (see subclause 5.1), the USF bits sent during the other half of the 20 ms block period may be sent with a different modulation. In this case, the half of  $u_L$  and  $u_H$  not sent in the present data block will be discarded.

### 5.1a.24.3 Header coding

The header  $\{h(0),...,h(26)\}$  is coded as defined in subclause 5.1a.1.1, with N=27, resulting in a block of 105 bits,  $\{C(0),...,C(104)\}$ .

Some bits of the code are repeated in the following way:

$$hc(k) = C(k)$$
 for  $k = 0, 1, ... 104$  and

$$hc(105) = C(0), hc(106) = C(33), hc(107) = C(70)$$

This results in a block of 108 bits,  $\{hc(0),...,hc(107)\}$ .

# 5.1a.24.4 Data coding

The data part,  $\{i(0),...,i(449)\}$ , is coded as defined in subclause 5.1a.1.3, with N=450, resulting in a coded block of 1398 bits,  $\{C(0),...,C(1397)\}$ .

The coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied.

The parameter values used for rate matching are: swap=0.05, N=466,  $N_{data}=956$  and  $N_{data2}=876$ .

P1 puncturing is generated according to 5.1a.1.3.5

P2 (Type 1) puncturing is generated according to 5.1a.1.3.5.

If a PAN is not included, the result is a block of 956 bits,  $\{c(0),...,c(955)\}$ .

If a PAN is included, the result is a block of 876 bits,  $\{c(0),...,c(875)\}$ .

### 5.1a.24.5 PAN coding

The PAN coding is the same as for UBS-5 as specified in subclause 5.1a.8.4.

#### 5.1a.24.6 Interleaving

a) Header

The header,  $\{hc(0),...,hc(107)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C=108$  and a=10, resulting in a block of 108 bits,  $\{hi(0),...,hi(107)\}$ .

b) Data and PAN

If a PAN is not included, the following rule applies:

$$dc(k) = c(k)$$
 for  $k = 0,...,955$ 

If a PAN is included, data and PAN are put together as one entity as described by the following rule:

$$dc(k) = ac(k)$$
 for  $k = 0,...,79$ 

$$dc(k) = c(k-80)$$
 for  $k = 80,...,955$ 

The block  $\{dc(0),...,dc(955)\}$  is interleaved as defined in subclause 5.1a.2.1, with  $N_C$ =956 and a=173, resulting in a block of 956 bits,  $\{di(0),...,di(955)\}$ .

## 5.1a.24.7 Mapping on a burst

The mapping is given by the rule:

$$e(B,j) = di(239B+j)$$
 for  $j = 0,...,119$ 

$$e(B,j) = hi(27B+j-120)$$
 for  $j = 120,...,135$ 

$$e(B,j) = q(2B+j-136)$$
 for  $j = 136,137$ 

$$e(B,j) = u'(8B+j-138)$$
 for  $j = 138,...,145$   
 $e(B,j) = hi(27B+j-130)$  for  $j = 146,...,156$   
 $e(B,j) = di(239B+j-37)$  for  $j = 157,...,275$ 

where

q(0),q(1),...,q(7) = 0,0,0,0,0,0,0,0,0 identifies the coding scheme DBS-5 or DBS-6.

In RTTI configuration, the bursts with B = 0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B = 1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.

# 5.1a.25 Packet data block type 36 (DBS-6)

#### 5.1a.25.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 624 information bits  $\{d(0),d(1),...,d(623)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 649 information bits  $\{d(0),d(1),...,d(648)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$u(k) = d(k)$$
 for  $k = 0,...,2$   
 $h(k-3) = d(k)$  for  $k = 3,...,29$   
 $i(k-30) = d(k)$  for  $k = 30,...,623$ 

And if a PAN is included:

$$pn(k-624) = d(k)$$
 for  $k = 624,...,648$ 

#### 5.1a.25.2 USF coding

The USF coding is the same as for DBS-5 as specified in subclause 5.1a.24.2.

### 5.1a.25.3 Header coding

The header coding is the same as for DBS-5 as specified in subclause 5.1a.24.3.

### 5.1a.25.4 Data coding

The data part,  $\{i(0),...,i(593)\}$ , is coded as defined in subclause 5.1a.1.3, with N=594, resulting in a coded block of 1830 bits,  $\{C(0),...,C(1829)\}$ .

The coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied.

The parameter values used for rate matching are: swap=0.15, N=610,  $N_{data}=956$  and  $N_{data}=876$ .

P1 puncturing is generated according to 5.1a.1.3.5

P2 (Type 1) puncturing is generated according to 5.1a.1.3.5.

If a PAN is not included, the result is a block of 956 bits,  $\{c(0),...,c(955)\}$ .

If a PAN is included, the result is a block of 876 bits,  $\{c(0),...,c(875)\}$ .

# 5.1a.25.5 PAN coding

The PAN coding is the same as for UBS-5 as specified in subclause 5.1a.8.4.

# 5.1a.25.6 Interleaving

The interleaving is the same as for DBS-5 as specified in subclause 5.1a.24.6.

# 5.1a.25.7 Mapping on a burst

The mapping is the same as for DBS-5 as specified in subclause 5.1a.24.7.

# 5.1a.26 Packet data block type 37 (DBS-7)

## 5.1a.26.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 941 information bits  $\{d(0),d(1),...,d(940)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 966 information bits  $\{d(0),d(1),...,d(965)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$u(k) = d(k)$$
 for  $k = 0,...,2$   
 $h(k-3) = d(k)$  for  $k = 3,...,40$   
 $i1(k-41) = d(k)$  for  $k = 41,...,490$   
 $i2(k-491) = d(k)$  for  $k = 491,...,940$ 

And if a PAN is included:

$$pn(k-941) = d(k)$$
 for  $k = 941,...,965$ 

### 5.1a.26.2 USF coding

# 5.1a.26.2.1 BTTI configuration

The USF bits  $\{u(0), u(1), u(2)\}$  are block coded into 64 bits u'(0), u'(1), ..., u'(63) according to the following table:

u(0),u(1),u(2)	u'(0),u'(1),,u'(63)			
	burst 0	burst 1	burst 2	burst 3
000	0011001100110011	0011001100110011	0011001100110011	0011001100110011
001	11111111111111111	0011001100110011	0011001111111111	1111111100110011
010	1111111100110011	1111111100110011	1111111100110011	1111111100110011
011	0011001111111111	1111111100110011	11111111111111111	0011001100110011
100	1111111100110011	0011001111111111	00110011 00110011	11111111111111111
101	0011001111111111	0011001111111111	0011001111111111	0011001111111111
110	0011001100110011	11111111111111111	1111111100110011	0011001111111111
111	11111111111111111	11111111111111111	11111111111111111	11111111111111111

# 5.1a.26.2.2 RTTI configurations

If the USF is sent in RTTI USF mode (see 3GPP TS 45.002) when data blocks are transmitted in RTTI configuration, then the USF bits  $\{u(0),u(1),u(2)\}$  are block coded into 64 bits u'(0),u'(1),...,u'(63) as described in subclause 5.1a.26.2.1.

If the USF is sent in BTTI USF mode (see 3GPP TS 45.002) when data blocks are transmitted in RTTI configuration, then the three bits of the USF to be sent on the lower numbered PDCH of a corresponding downlink PDCH-pair are block coded into 64 bits  $u_L(0)$ ,  $u_L(1)$ ,..., $u_L(63)$  as described in subclause 5.1a.26.2.1; the three bits of the USF to be sent on the higher numbered PDCH of a corresponding downlink PDCH-pair are block coded into 64 bits  $u_H(0)$ ,  $u_H(1)$ ,..., $u_H(63)$  as described in subclause 5.1a.26.2.1.

NOTE: If BTTI USF mode is used when sending data blocks in RTTI configuration, then u(0),u(1),u(2) need not contain a USF; in this case, they are ignored by the encoder. How the USFs are delivered to the encoder in this case is implementation dependent.

If the data block is sent in the first 10ms of a 20ms block period, then:

```
u'(j)=u_L(j), j=0...15

u'(j)=u_H(j-16), j=16...31

u'(j)=u_L(j-16), j=32...47

u'(j)=u_H(j-32) j=48...63
```

If the data block is sent in the second 10ms of a 20ms block period, then:

```
u'(j)=u_L(j+32), \quad j=0...15

u'(j)=u_H(j+16), \quad j=16...31

u'(j)=u_L(j+16), \quad j=32...47

u'(j)=u_H(j) \qquad j=48...63
```

NOTE: In case mixed modulation USF is used (see subclause 5.1), the USF bits sent during the other half of the 20 ms block period may be sent with a different modulation. In this case, the half of u<sub>L</sub> and u<sub>H</sub> not sent in the present data block will be discarded.

## 5.1a.26.3 Header coding

The header  $\{h(0),...,h(37)\}$  is coded as defined in subclause 5.1a.1.1, with N=38, resulting in a block of 138 bits,  $\{C(0),...,C(137)\}$ .

Some bits of the code are repeated in the following way:

```
hc(k) = C(k) for k = 0, 1, 2, ... 137 and hc(138) = C(0), hc(139) = C(22), hc(140) = C(49), hc(141) = C(69), hc(142) = C(91), hc(143) = C(118).
```

The result is a block of 144 coded bits,  $\{hc(0),...,hc(143)\}$ .

#### 5.1a.26.4 Data coding

Each data part,  $\{i1(0),...,i1(449)\}$  and  $\{i2(0),...,i2(449)\}$ , is coded as defined in subclause 5.1a.1.3, with N=450, resulting in two coded blocks of 1398 bits,  $\{C1(0),...,C1(1397)\}$  and  $\{C2(0),...,C2(1397)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied.

The parameter values used for rate matching are: swap=0, N=466,  $N_{data}=996$  and  $N_{data2}=956$ .

P1 puncturing is generated according to 5.1a.1.3.5

P2 (Type 1) puncturing is generated according to 5.1a.1.3.5.

If a PAN is not included, the result is two blocks of 996 bits, {c1(0),...,c1(995)} and {c2(0),...,c2(995)}.

If a PAN is included, the result is two blocks of 956 bits,  $\{c1(0),...,c1(955)\}\$  and  $\{c2(0),...,c2(955)\}\$ .

NOTE: C1 and c1 correspond to i1, and C2 and c2 to i2.

# 5.1a.26.5 PAN coding

The PAN coding is the same as for UBS-5 as specified in subclause 5.1a.8.4.

# 5.1a.26.6 Interleaving

a) Header

The header,  $\{hc(0),...,hc(143)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C=144$  and a=5, resulting in a block of 144 bits,  $\{hi(0),...,hi(143)\}$ .

b) Data and PAN

If a PAN is not included, data are put together as one entity as described by the following rule:

$$dc(k) = c1(k)$$
 for  $k = 0,...,995$   
 $dc(k) = c2(k-996)$  for  $k = 996,...,1991$ 

If a PAN is included, data and PAN are put together as one entity as described by the following rule:

$$dc(k) = ac(k)$$
 for  $k = 0,...,79$   
 $dc(k) = c1(k-80)$  for  $k = 80,...,1035$   
 $dc(k) = c2(k-1036)$  for  $k = 1036,...,1991$ 

The block  $\{dc(0),...,dc(1991)\}$  is interleaved as defined in subclause 5.1a.2.1, with  $N_C=1992$  and a=325, resulting in a block of 1992 bits,  $\{di(0),...,di(1991)\}$ .

# 5.1a.26.7 Mapping on a burst

a) Straightforward mapping

The mapping is given by the rule:

$$e(B,j) = di(498B+j) \qquad \text{for } j = 0,...,248$$

$$e(B,j) = hi(36B+j-249) \qquad \text{for } j = 249,...,271$$

$$e(B,j) = q(2B+j-272) \qquad \text{for } j = 272,273$$

$$e(B,j) = hi(36B+j-251) \qquad \text{for } j = 274,...,275$$

$$e(B,j) = u'(16B+j-276) \qquad \text{for } j = 276,...,291$$

$$e(B,j) = hi(36B+j-267) \qquad \text{for } j = 292,...,302$$

$$e(B,j) = di(498B+j-54) \qquad \text{for } j = 303,...,551$$

where

q(0),q(1),...,q(7) = 0,0,0,0,0,0,0,0,0 identifies the coding scheme DBS-7 or DBS-8.

b) Bit swapping

After this mapping the following bits are swapped:

For 
$$B = 0,1,2,3$$
,

Swap e(B,224+k) with e(B,250+k) for k=0, 4, 8, 12, 16, 20, 24.

Swap e(B,221+k) with e(B,251+k) for k=0, 4, 8, 12, 16, 20, 24.

Swap e(B,304+k) with e(B,294+k) for k=0, 1, 4, 5, 8.

In RTTI configuration, the bursts with B = 0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B = 1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.

#### c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

For B = 0

Swap e(B,47) with e(B,120)

Swap e(B,110) with e(B,160)

Swap e(B,131) with e(B,177)

Swap e(B,199) with e(B,81)

Swap e(B,271) with e(B,201)

Swap e(B,358) with e(B,368)

Swap e(B,379) with e(B,444)

Swap e(B,426) with e(B,340)

Swap e(B,447) with e(B,329)

Swap e(B,510) with e(B,500)

Swap e(B,531) with e(B,405)

For B = 1

Swap e(B,30) with e(B,40)

Swap e(B,51) with e(B,120)

Swap e(B,114) with e(B,177)

Swap e(B,182) with e(B,0)

Swap e(B,203) with e(B,81)

Swap e(B,250) with e(B,201)

Swap e(B,275) with e(B,160)

Swap e(B,430) with e(B,368)

Swap e(B,451) with e(B,405)

Swap e(B,514) with e(B,444)

For B = 2

Swap e(B,34) with e(B,120)

Swap e(B,55) with e(B,160)

Swap e(B,123) with e(B,40)

Swap e(B,186) with e(B,81)

Swap e(B,207) with e(B,201)

Swap e(B,254) with e(B,177)

Swap e(B,303) with e(B,444)

Swap e(B,350) with e(B,340)

Swap e(B,371) with e(B,500)

Swap e(B,434) with e(B,368)

Swap e(B,455) with e(B,405)

Swap e(B,523) with e(B,329)

#### For B = 3

Swap e(B,38) with e(B,160)

Swap e(B,106) with e(B,81)

Swap e(B,127) with e(B,120)

Swap e(B,190) with e(B,201)

Swap e(B,211) with e(B,177)

Swap e(B,302) with e(B,340)

Swap e(B,354) with e(B,500)

Swap e(B,375) with e(B,405)

Swap e(B,438) with e(B,444)

Swap e(B,506) with e(B,329)

Swap e(B,527) with e(B,368)

# 5.1a.27 Packet data block type 38 (DBS-8)

### 5.1a.27.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 1229 information bits  $\{d(0),d(1),...,d(1228)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 1254 information bits  $\{d(0),d(1),...,d(1253)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$u(k) = d(k)$$
 for  $k = 0,...,2$ 

$$h(k-3) = d(k)$$
 for  $k = 3,...,40$ 

$$i1(k-41) = d(k)$$
 for  $k = 41,...,634$ 

$$i2(k-635) = d(k)$$
 for  $k = 635,...,1228$ 

And if a PAN is included:

$$pn(k-1229) = d(k)$$
 for  $k = 1229,...,1253$ 

# 5.1a.27.2 USF coding

The USF coding is the same as for DBS-7 as specified in subclause 5.1a.26.2.

# 5.1a.27.3 Header coding

The header coding is the same as for DBS-7 as specified in subclause 5.1a.26.3.

### 5.1a.27.4 Data coding

Each data part,  $\{i1(0),...,i1(593)\}$  and  $\{i2(0),...,i2(593)\}$ , is coded as defined in subclause 5.1a.1.3, with N=594, resulting in two coded blocks of 1830 bits,  $\{C1(0),...,C1(1829)\}$  and  $\{C2(0),...,C2(1829)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Two puncturing schemes named P1 or P2 are applied.

The parameter values used for rate matching are: swap=0.10, N=610,  $N_{data}=996$  and  $N_{data2}=956$ .

P1 puncturing is generated according to 5.1a.1.3.5

P2 (Type 1) puncturing is generated according to 5.1a.1.3.5.

If a PAN is not included, the result is two blocks of 996 bits,  $\{c1(0),...,c1(995)\}$  and  $\{c2(0),...,c2(995)\}$ .

If a PAN is included, the result is two blocks of 956 bits, {c1(0),...,c1(955)} and {c2(0),...,c2(955)}.

NOTE: C1 and c1 correspond to i1, and C2 and c2 to i2.

# 5.1a.27.5 PAN coding

The PAN coding is the same as for UBS-5 as specified in subclause 5.1a.8.4.

### 5.1a.27.6 Interleaving

The interleaving is the same as for DBS-7 as specified in subclause 5.1a.26.6.

### 5.1a.27.7 Mapping on a burst

The mapping is the same as for DBS-7 as specified in subclause 5.1a.26.7.

# 5.1a.28 Packet data block type 39 (DBS-9)

# 5.1a.28.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 1403 information bits  $\{d(0),d(1),...,d(1402)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 1428 information bits  $\{d(0),d(1),...,d(1427)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$u(k) = d(k)$$
 for  $k = 0,...,2$   
 $h(k-3) = d(k)$  for  $k = 3,...,52$   
 $i1(k-53) = d(k)$  for  $k = 53,...,502$   
 $i2(k-503) = d(k)$  for  $k = 503,...,952$   
 $i3(k-953) = d(k)$  for  $k = 953,...,1402$ 

And if a PAN is included:

$$pn(k-1403) = d(k)$$
 for  $k = 1403,...,1427$ 

### 5.1a.28.2 USF coding

The USF coding is the same as for DBS-7 as specified in subclause 5.1a.26.2.

# 5.1a.28.3 Header coding

The header  $\{h(0),...,h(49)\}$  is coded as defined in subclause 5.1a.1.1, with N=50, resulting in a block of 174 bits,  $\{C(0),...,C(173)\}$ .

The code is punctured in such a way that the following coded bits:

$$\{C(87*k+j) \text{ for } k=0 \text{ and } 1, j=44, 65 \text{ and } 86\}$$
 are not transmitted

This results in a block of 168 bits,  $\{hc(0),...,hc(167)\}$ .

# 5.1a.28.4 Data coding

Each data part,  $\{i1(0),...,i1(449)\}$ ,  $\{i2(0),...,i2(449)\}$  and  $\{i3(0),...,i3(449)\}$ , is coded as defined in subclause 5.1a.1.3, with N=450, resulting in three coded blocks of 1398 bits,  $\{C1(0),...,C1(1397)\}$ ,  $\{C2(0),...,C2(1397)\}$  and  $\{C3(0),...,C3(1397)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied.

The parameter values used for rate matching are: swap=0, N=466,  $N_{data}=656$  and  $N_{data}=630$ .

P1 puncturing is generated according to 5.1a.1.3.5

P2 (Type 2) puncturing is generated according to 5.1a.1.3.5.

P3 puncturing is generated according to 5.1a.1.3.5.

If a PAN is not included, the result is three blocks of 656 bits,  $\{c1(0),...,c1(655)\}$ ,  $\{c2(0),...,c2(655)\}$  and  $\{c3(0),...,c3(655)\}$ .

If a PAN is included, the result is three blocks of 630 bits,  $\{c1(0),...,c1(629)\}$ ,  $\{c2(0),...,c2(629)\}$  and  $\{c3(0),...,c3(629)\}$ .

NOTE: C1 and c1 correspond to i1, C2 and c2 to i2 and C3 and c3 to i3...

# 5.1a.28.5 PAN coding

The PAN coding is the same as for UAS-7 as specified in subclause 5.1a.3.4.

# 5.1a.28.6 Interleaving

a) Header

The header,  $\{hc(0),...,hc(167)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C=168$  and a=17, resulting in a block of 168 bits,  $\{hi(0),...,hi(167)\}$ .

b) Data and PAN

If a PAN is not included, data are put together as one entity as described by the following rule:

$$dc(k) = c1(k)$$
 for  $k = 0,...,655$ 

$$dc(k) = c2(k-656)$$
 for  $k = 656,...,1311$ 

$$dc(k) = c3(k-1312)$$
 for  $k = 1312,...,1967$ 

If a PAN is included, data and PAN are put together as one entity as described by the following rule:

$$\begin{aligned} dc(k) &= ac(k) & \text{for } k = 0,...,77 \\ dc(k) &= c1(k-78) & \text{for } k = 78,...,707 \\ dc(k) &= c2(k-708) & \text{for } k = 708,...,1337 \\ dc(k) &= c3(k-1338) & \text{for } k = 1338,...,1967 \end{aligned}$$

The block  $\{dc(0),...,dc(1967)\}$  is interleaved as defined in subclause 5.1a.2.1, with  $N_C$ =1968 and a=283, resulting in a block of 1968 bits,  $\{di(0),...,di(1967)\}$ .

# 5.1a.28.7 Mapping on a burst

a) Straightforward mapping

The mapping is given by the rule:

$$\begin{split} e(B,j) &= \text{di}(492\text{B}+j) & \text{for } j = 0,...,245 \\ e(B,j) &= \text{hi}(42\text{B}+j\text{-}246) & \text{for } j = 246,...,271 \\ e(B,j) &= q(2\text{B}+j\text{-}272) & \text{for } j = 272,273 \\ e(B,j) &= \text{hi}(42\text{B}+j\text{-}248) & \text{for } j = 274,...,275 \\ e(B,j) &= \text{u}'(16\text{B}+j\text{-}276) & \text{for } j = 276,...,291 \\ e(B,j) &= \text{hi}(42\text{B}+j\text{-}264) & \text{for } j = 292,...,305 \\ e(B,j) &= \text{di}(492\text{B}+j\text{-}60) & \text{for } j = 306,...,551 \\ \end{split}$$

where

$$q(0),q(1),...,q(7) = 1,1,1,1,1,1,1,1$$
 identifies the coding scheme DBS-9.

b) Bit swapping

After this mapping the following bits are swapped:

For 
$$B = 0,1,2,3$$
,   
 Swap  $e(B,216+k)$  with  $e(B,246+k)$  for  $k=0,1,4,5,8,9,12,13,16,17,20,21,24,25,28,29$ .   
 Swap  $e(B,308+k)$  with  $e(B,294+k)$  for  $k=0,1,4,5,8,9$ .

In RTTI configuration, the bursts with B = 0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B = 1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.

c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

Swap e(B,174) with e(B,40)

Swap e(B,222) with e(B,160)

Swap e(B,235) with e(B,120)

Swap e(B,343) with e(B,444)

Swap e(B,430) with e(B,368)

Swap e(B,443) with e(B,500)

Swap e(B,491) with e(B,405)

#### For B = 1

Swap e(B,38) with e(B,120)

Swap e(B,51) with e(B,40)

Swap e(B,99) with e(B,201)

Swap e(B,186) with e(B,81)

Swap e(B,234) with e(B,177)

Swap e(B,307) with e(B,500)

Swap e(B,442) with e(B,405)

Swap e(B,455) with e(B,329)

Swap e(B,503) with e(B,444)

#### For B = 2

Swap e(B,50) with e(B,160)

Swap e(B,63) with e(B,120)

Swap e(B,111) with e(B,177)

Swap e(B,198) with e(B,201)

Swap e(B,211) with e(B,81)

Swap e(B,306) with e(B,444)

Swap e(B,319) with e(B,368)

Swap e(B,406) with e(B,329)

Swap e(B,454) with e(B,405)

Swap e(B,467) with e(B,500)

# For B = 3

Swap e(B,14) with e(B,40)

Swap e(B,62) with e(B,160)

Swap e(B,75) with e(B,120)

Swap e(B,123) with e(B,177)

Swap e(B,210) with e(B,201)

Swap e(B,223) with e(B,81)

Swap e(B,331) with e(B,368)

Swap e(B,418) with e(B,329)

Swap e(B,466) with e(B,444)

Swap e(B,479) with e(B,500)

# 5.1a.29 Packet data block type 40 (DBS-10)

#### 5.1a.29.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 1835 information bits  $\{d(0),d(1),...,d(1834)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 1860 information bits  $\{d(0),d(1),...,d(1859)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$u(k) = d(k)$$
 for  $k = 0,...,2$ 

$$h(k-3) = d(k)$$
 for  $k = 3,...,52$ 

$$i1(k-53) = d(k)$$
 for  $k = 53,...,646$ 

$$i2(k-647) = d(k)$$
 for  $k = 647,...,1240$ 

$$i3(k-1241) = d(k)$$
 for  $k = 1241,...,1834$ 

And if a PAN is included:

$$pn(k-1835) = d(k)$$
 for  $k = 1835,...,1859$ 

# 5.1a.29.2 USF coding

# 5.1a.29.2.1 BTTI configuration

The USF bits  $\{u(0),u(1),u(2)\}$  are block coded into 80 bits u'(0),u'(1),...,u'(79) according to the following table:

u(0),u(1),u(2)		u'(0),u'(1),,u'(79)		
	burst 0	burst 1	burst 2	burst 3
000	000000000	0000000000	000000000	000000000
	000000000	000000000	000000000	000000000
001	1001010010	0000000000	000000000	1001010010
	1001010010	000000000	1001010010	000000000
010	1001010010	1001010010	1001010010	1001010010
	000000000	0000000000	000000000	000000000
011	000000000	1001010010	1001010010	000000000
	1001010010	0000000000	1001010010	000000000
100	1001010010	0000000000	000000000	1001010010
	000000000	1001010010	000000000	1001010010
101	000000000	0000000000	000000000	000000000
	1001010010	1001010010	1001010010	1001010010
110	000000000	1001010010	1001010010	000000000
	000000000	1001010010	000000000	1001010010
111	1001010010	1001010010	1001010010	1001010010
	1001010010	1001010010	1001010010	1001010010

## 5.1a.29.2.2 RTTI configurations

If the USF is sent in RTTI USF mode (see 3GPP TS 45.002) when data blocks are transmitted in RTTI configuration, then the USF bits  $\{u(0),u(1),u(2)\}$  are block coded into 80 bits u'(0),u'(1),...,u'(79) as described in subclause 5.1a.29.2.1.

If the USF is sent in BTTI USF mode (see 3GPP TS 45.002) when data blocks are transmitted in RTTI configuration, then the three bits of the USF to be sent on the lower numbered PDCH of a corresponding downlink PDCH-pair are block coded into 80 bits  $u_L(0)$ , $u_L(1)$ ,..., $u_L(79)$  as described in subclause 5.1a.29.2.1; the three bits of the USF to be sent on the higher numbered PDCH of a corresponding downlink PDCH-pair are block coded into 80 bits  $u_H(0)$ , $u_H(1)$ ,..., $u_H(79)$  as described in subclause 5.1a.29.2.1.

NOTE: If BTTI USF mode is used when sending data blocks in RTTI configuration, then u(0),u(1),u(2) need not contain a USF; in this case, they are ignored by the encoder. How the USFs are delivered to the encoder in this case is implementation dependent.

If the data block is sent in the first 10ms of a 20ms block period, then:

$$u'(j)=u_L(j),$$
  $j=0...19$   
 $u'(j)=u_H(j-20),$   $j=20...39$   
 $u'(j)=u_L(j-20),$   $j=40...59$   
 $u'(j)=u_H(j-40)$   $j=60...79$ 

If the data block is sent in the second 10ms of a 20ms block period, then:

$$u'(j)=u_L(j+40), \quad j=0...19$$
  
 $u'(j)=u_H(j+20), \quad j=20...39$   
 $u'(j)=u_L(j+20), \quad j=40...59$   
 $u'(j)=u_H(j) \qquad j=60...79$ 

NOTE: In case mixed modulation USF is used (see subclause 5.1), the USF bits sent during the other half of the 20ms block period may be sent with a different modulation. In this case, the half of  $u_L$  and  $u_H$  not sent in the present data block will be discarded.

# 5.1a.29.3 Header coding

The header coding is the same as for DBS-9 as specified in subclause 5.1a.28.3.

#### 5.1a.29.4 Data coding

Each data part,  $\{i1(0),...,i1(593)\}$ ,  $\{i2(0),...,i2(593)\}$  and  $\{i3(0),...,i3(593)\}$ , is coded as defined in subclause 5.1a.1.3, with N=594, resulting in three coded blocks of 1830 bits,  $\{C1(0),...,C1(1829)\}$ ,  $\{C2(0),...,C2(1829)\}$  and  $\{C3(0),...,C3(1829)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied.

The parameter values used for rate matching are: swap=0, N=610,  $N_{data}=833$  and  $N_{data2}=807$ .

P1 puncturing is generated according to 5.1a.1.3.5

P2 (Type 2) puncturing is generated according to 5.1a.1.3.5.

P3 puncturing is generated according to 5.1a.1.3.5.

If a PAN is not included, the result is three blocks of 833 bits,  $\{c1(0),...,c1(832)\}$ ,  $\{c2(0),...,c2(832)\}$  and  $\{c3(0),...,c3(832)\}$ .

If a PAN is included, the result is three blocks of 807 bits,  $\{c1(0),...,c1(806)\}$ ,  $\{c2(0),...,c2(806)\}$  and  $\{c3(0),...,c3(806)\}$ .

NOTE: C1 and c1 correspond to i1, C2 and c2 to i2 and C3 and c3 to i3...

# 5.1a.29.5 PAN coding

The PAN coding is the same as for UAS-7 as specified in subclause 5.1a.3.4.

# 5.1a.29.6 Interleaving

#### a) Header

The header,  $\{hc(0),...,hc(167)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C=168$  and a=19, resulting in a block of 168 bits,  $\{hi(0),...,hi(167)\}$ .

#### b) Data and PAN

If a PAN is not included, data are put together as one entity as described by the following rule:

$$dc(k) = c1(k) \qquad \qquad \text{for } k = 0,...,832$$
 
$$dc(k) = c2(k-833) \qquad \text{for } k = 833,...,1665$$
 
$$dc(k) = c3(k-1666) \qquad \text{for } k = 1666,...,2498$$
 
$$dc(k) = 0 \qquad \qquad \text{for } k = 2499$$

If a PAN is included, data and PAN are put together as one entity as described by the following rule:

$$\begin{aligned} &dc(k) = ac(k) & for \ k = 0,...,77 \\ &dc(k) = c1(k-78) & for \ k = 78,...,884 \\ &dc(k) = c2(k-885) & for \ k = 885,...,1691 \\ &dc(k) = c3(k-1692) & for \ k = 1692,...,2498 \\ &dc(k) = 0 & for \ k = 2499 \end{aligned}$$

The block  $\{dc(0),...,dc(2499)\}$  is interleaved as defined in subclause 5.1a.2.1, with  $N_C$ =2500 and a=323, resulting in a block of 2500 bits,  $\{di(0),...,di(2499)\}$ .

# 5.1a.29.7 Mapping on a burst

# a) Straightforward mapping

The mapping is given by the rule:

$$\begin{split} e(B,j) &= di(625B+j) & \text{for } j = 0,...,312 \\ e(B,j) &= hi(42B+j-313) & \text{for } j = 313,...,339 \\ e(B,j) &= q(3B+j-340) & \text{for } j = 340 \\ e(B,j) &= hi(42B+j-314) & \text{for } j = 341 \\ e(B,j) &= q(3B+j-341) & \text{for } j = 342,...,343 \\ e(B,j) &= hi(42B+j-316) & \text{for } j = 344 \\ e(B,j) &= u'(20B+j-345) & \text{for } j = 345,...,364 \end{split}$$

$$e(B,j) = hi(42B+j-336)$$
 for  $j = 365,...,377$ 

$$e(B,j) = di(625B+j-65)$$
 for  $j = 378,...,689$ 

where

q(0),q(1),...,q(11) = 0,0,0,0,0,0,0,0,0,0,0,0 identifies the coding scheme DBS-10.

#### b) Bit swapping

After this mapping the following bits are swapped:

For 
$$B = 0,1,2,3$$
,

Swap e(B,285+k) with e(B,316+k) for  $k=0,\,5,\,10,\,15,\,20,\,25.$ 

Swap e(B,278+k) with e(B,314+k) for k=0, 5, 10, 15, 20, 25, 30.

Swap e(B,270+k) with e(B,317+k) for k=0,5,10.

Swap e(B,268+k) with e(B,332+k) for k=0, 5.

Swap e(B,380+k) with e(B,366+k) for k=0, 5, 10.

Swap e(B,378+k) with e(B,369+k) for k=0, 5.

Swap e(B,388+k) with e(B,367+k) for k=0, 5.

Swap e(B,395) with e(B,377)

In RTTI configuration, the bursts with B = 0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B = 1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.

#### c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

#### For B = 0

Swap e(B,21) with e(B,80)

Swap e(B,42) with e(B,98)

Swap e(B,84) with e(B,110)

Swap e(B,126) with e(B,158)

Swap e(B,147) with e(B,170)

Swap e(B,189) with e(B,188)

Swap e(B,367) with e(B,410)

Swap e(B,409) with e(B,433)

Swap e(B,451) with e(B,480)

Swap e(B,472) with e(B,503)

Swap e(B,514) with e(B,530)

Swap e(B,556) with e(B,688)

Swap e(B,577) with e(B,640)

For B = 1

Swap e(B,86) with e(B,80)

Swap e(B,107) with e(B,98)

Swap e(B, 149) with e(B, 110)

Swap e(B,191) with e(B,158)

Swap e(B,212) with e(B,188)

Swap e(B,254) with e(B,200)

Swap e(B,474) with e(B,410)

Swap e(B,516) with e(B,433)

Swap e(B,537) with e(B,480)

Swap e(B,579) with e(B,503)

Swap e(B,621) with e(B,530)

Swap e(B,642) with e(B,493)

#### For B = 2

Swap e(B,17) with e(B,200)

Swap e(B,151) with e(B,80)

Swap e(B,172) with e(B,98)

Swap e(B,214) with e(B,110)

Swap e(B,256) with e(B,158)

Swap e(B,277) with e(B,170)

Swap e(B,334) with e(B,188)

Swap e(B,384) with e(B,493)

Swap e(B,539) with e(B,410)

Swap e(B,581) with e(B,433)

Swap e(B,602) with e(B,480) Swap e(B,644) with e(B,503)

Swap e(B,686) with e(B,530)

#### For B = 3

Swap e(B,19) with e(B,158)

Swap e(B,61) with e(B,170)

Swap e(B,82) with e(B,188)

Swap e(B,237) with e(B,80)

Swap e(B,279) with e(B,98)

Swap e(B,331) with e(B,110) Swap e(B,386) with e(B,503)

Swap e(B,407) with e(B,530)

Swap e(B,449) with e(B,493) Swap e(B,604) with e(B,410) Swap e(B,646) with e(B,433)

Swap e(B,667) with e(B,480)

# 5.1a.30 Packet data block type 41 (DBS-11)

#### 5.1a.30.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 2249 information bits  $\{d(0),d(1),...,d(2248)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 2274 information bits  $\{d(0),d(1),...,d(2273)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$\begin{array}{ll} u(k)=d(k) & \text{for } k=0,...,2 \\ \\ h(k-3)=d(k) & \text{for } k=3,...,64 \\ \\ i1(k-65)=d(k) & \text{for } k=65,...,610 \\ \\ i2(k-611)=d(k) & \text{for } k=611,...,1156 \\ \\ i3(k-1157)=d(k) & \text{for } k=1157,...,1702 \\ \\ i4(k-1703)=d(k) & \text{for } k=1703,...,2248 \end{array}$$

And if a PAN is included:

$$pn(k-2249) = d(k)$$
 for  $k = 2249,...,2273$ 

#### 5.1a.30.2 USF coding

The USF coding is the same as for DBS-10 as specified in subclause 5.1a.29.2.

#### 5.1a.30.3 Header coding

The header  $\{h(0),...,h(61)\}$  is coded as defined in subclause 5.1a.1.1, with N=62, resulting in a block of 210 bits,  $\{C(0),...,C(209)\}$ .

The code is punctured in such a way that the following coded bits:

```
\{C(30*k+j) \text{ for } k = 0,...,6, j = 17, 20 \text{ and } 28, \text{ and } C(10)\}\ are not transmitted
```

This results in a block of 188 bits,  $\{hc(0),...,hc(187)\}$ .

# 5.1a.30.4 Data coding

Each data part,  $\{i1(0),...,i1(545)\}$ ,  $\{i2(0),...,i2(545)\}$ ,  $\{i3(0),...,i3(545)\}$  and  $\{i4(0),...,i4(545)\}$ , is coded as defined in subclause 5.1a.1.3, with N=546, resulting in four coded blocks of 1686 bits,  $\{C1(0),...,C1(1685)\}$ ,  $\{C2(0),...,C2(1685)\}$ ,  $\{C3(0),...,C3(1685)\}$  and  $\{C4(0),...,C4(1685)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied.

The parameter values used for rate matching are: swap=0, N=562,  $N_{data}=620$  and  $N_{data}=600$ .

P1 puncturing is generated according to 5.1a.1.3.5

P2 (Type 2) puncturing is generated according to 5.1a.1.3.5.

P3 puncturing is generated according to 5.1a.1.3.5.

If a PAN is not included, the result is four blocks of 620 bits,  $\{c1(0),...,c1(619)\}$ ,  $\{c2(0),...,c2(619)\}$ ,  $\{c3(0),...,c3(619)\}$  and  $\{c4(0),...,c4(619)\}$ .

If a PAN is included, the result is four blocks of 600 bits,  $\{c1(0),...,c1(599)\}$ ,  $\{c2(0),...,c2(599)\}$ ,  $\{c3(0),...,c3(599)\}$  and  $\{c4(0),...,c4(599)\}$ .

NOTE: C1 and c1 correspond to i1, C2 and c2 to i2, C3 and c3 to i3, and C4 and c4 to i4.

# 5.1a.30.5 PAN coding

The PAN coding is the same as for UBS-5 as specified in subclause 5.1a.8.4.

# 5.1a.30.6 Interleaving

a) Header

The header,  $\{hc(0),...,hc(187)\}$ , is interleaved as defined in subclause 5.1a.2.1, with  $N_C=188$  and a=3, resulting in a block of 188 bits,  $\{hi(0),...,hi(187)\}$ .

b) Data and PAN

If a PAN is not included, the following definitions apply:

$$dc1(k) = c1(k) \qquad for k = 0,...,619$$
 
$$dc2(k) = c2(k) \qquad for k = 0,...,619$$
 
$$dc3(k) = c3(k) \qquad for k = 0,...,619$$
 
$$dc4(k) = c4(k) \qquad for k = 0,...,619$$

If a PAN is included, the following definitions apply:

$$\begin{aligned} &\text{dc1}(k) = \text{ac}(4k) & \text{for } k = 0,...,19 \\ &\text{dc1}(k) = \text{c1}(k\text{-}20) & \text{for } k = 20,...,619 \\ &\text{dc2}(k) = \text{ac}(4k+1) & \text{for } k = 0,...,19 \\ &\text{dc2}(k) = \text{c2}(k\text{-}20) & \text{for } k = 20,...,619 \\ &\text{dc3}(k) = \text{ac}(4k+2) & \text{for } k = 0,...,19 \\ &\text{dc3}(k) = \text{c3}(k\text{-}20) & \text{for } k = 20,...,619 \\ &\text{dc4}(k) = \text{ac}(4k+3) & \text{for } k = 0,...,19 \\ &\text{dc4}(k) = \text{c4}(k\text{-}20) & \text{for } k = 20,...,619 \end{aligned}$$

The four blocks  $\{dc1(0),...,dc1(619)\}$ ,  $\{dc2(0),...,dc2(619)\}$ ,  $\{dc3(0),...,dc3(619)\}$  and  $\{dc4(0),...,dc4(619)\}$  are separately interleaved as defined in subclause 5.1a.2.2, with  $N_C$ =620 and a=141, resulting in the four blocks  $\{di1(0),...,di1(619)\}$ ,  $\{di2(0),...,di2(619)\}$ ,  $\{di3(0),...,di3(619)\}$  and  $\{di4(0),...,di4(619)\}$ , where di1 corresponds to dc1, di2 to dc2, di3 to dc3 and di4 to dc4.

The blocks are put together as one entity as described by the following rule:

$$\begin{aligned} &\text{di}(k) = \text{di}1(k) & &\text{for } k = 0,...,619 \\ &\text{di}(k) = \text{di}2(k\text{-}620) & &\text{for } k = 620,...,1239 \\ &\text{di}(k) = \text{di}3(k\text{-}1240) & &\text{for } k = 1240,...,1859 \end{aligned}$$

$$di(k) = di4(k-1860)$$
 for  $k = 1860,...,2479$ 

# 5.1a.30.7 Mapping on a burst

a) Straightforward mapping

The mapping is given by the rule:

For B=0,1,2,3, let

$$e(B,j) = di(620B+j)$$
 for  $j = 0,...,309$ 

$$e(B,j) = hi(47B+j-310)$$
 for  $j = 310,...,339$ 

$$e(B,j) = q(3B+j-340)$$
 for  $j = 340$ 

$$e(B,j) = hi(47B+j-311)$$
 for  $j = 341$ 

$$e(B,j) = q(3B+j-341)$$
 for  $j = 342,...,343$ 

$$e(B,j) = hi(47B+j-313)$$
 for  $j = 344$ 

$$e(B,j) = u'(20B+j-345)$$
 for  $j = 345,...,364$ 

$$e(B,j) = hi(47B+j-333)$$
 for  $j = 365,...,379$ 

$$e(B,j) = di(620B+j-70)$$
 for  $j = 380,...,689$ 

where

$$q(0),q(1),...,q(11) = 1,1,1,1,1,1,1,1,1,1,1,1,1,1$$
 identifies the coding scheme DBS-11 or DBS-12.

b) Bit swapping

After this mapping the following bits are swapped:

For 
$$B = 0,1,2,3$$
,

Swap e(B,275+k) with e(B,311+k) for k=0, 3, 5, 8, 10, 13, 15, 18, 20, 23, 25, 28, 30, 33.

Swap e(B,260+k) with e(B,312+k) for k=0, 5, 10.

Swap e(B,263+k) with e(B,327+k) for k=0, 5, 10.

Swap e(B,380+k) with e(B,366+k) for k=0, 3, 5, 8, 10, 13.

Swap e(B,395+k) with e(B,372+k) for k=0, 5.

Swap e(B,398) with e(B,367).

- In RTTI configuration, the bursts with B = 0.2 shall be mapped on the PDCH having the lower timeslot number, whereas the bursts with B = 1.3 shall be mapped on the PDCH having the higher timeslot number, see 3GPP TS 45.002.
- c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

For 
$$B = 0, 1, 2$$
 and 3

Swap e(B,29) with e(B,110)

Swap e(B,114) with e(B,158)

Swap e(B,141) with e(B,80)

Swap e(B,199) with e(B,188)

Swap e(B,226) with e(B,98)

Swap e(B,381) with e(B,433)

Swap e(B,466) with e(B,480)

Swap e(B,634) with e(B,410)

# 5.1a.31 Packet data block type 42 (DBS-12)

#### 5.1a.31.1 Block constitution

If the message delivered to the encoder does not include a PAN, it has a fixed size of 2441 information bits  $\{d(0),d(1),...,d(2440)\}$ . If the message delivered to the encoder includes a PAN, it has a fixed size of 2466 information bits  $\{d(0),d(1),...,d(2465)\}$ .

NOTE: The presence of the PAN is indicated by the PANI field in the header (see 3GPP TS 44.060).

The message is separated into the following parts:

$$u(k) = d(k)$$
 for  $k = 0,...,2$   
 $h(k-3) = d(k)$  for  $k = 3,...,64$   
 $i1(k-65) = d(k)$  for  $k = 65,...,658$   
 $i2(k-659) = d(k)$  for  $k = 659,...,1252$   
 $i3(k-1253) = d(k)$  for  $k = 1253,...,1846$   
 $i4(k-1847) = d(k)$  for  $k = 1847,...,2440$ 

And if a PAN is included:

$$pn(k-2441) = d(k)$$
 for  $k = 2441,...,2465$ 

#### 5.1a.31.2 USF coding

The USF coding is the same as for DBS-10 as specified in subclause 5.1a.29.2.

# 5.1a.31.3 Header coding

The header coding is the same as for DBS-11 as specified in subclause 5.1a.30.3.

# 5.1a.31.4 Data coding

Each data part,  $\{i1(0),...,i1(593)\}$ ,  $\{i2(0),...,i2(593)\}$ ,  $\{i3(0),...,i3(593)\}$  and  $\{i4(0),...,i4(593)\}$ , is coded as defined in subclause 5.1a.1.3, with N=594, resulting in four coded blocks of 1830 bits,  $\{C1(0),...,C1(1829)\}$ ,  $\{C2(0),...,C2(1829)\}$ ,  $\{C3(0),...,C3(1829)\}$  and  $\{C4(0),...,C4(1829)\}$ .

Each coded block is punctured depending on the value of the CPS field as defined in 3GPP TS 44.060. Three puncturing schemes named P1, P2 or P3 are applied.

The parameter values used for rate matching are: swap=0, N=610,  $N_{data}=620$  and  $N_{data2}=606$ .

P1 puncturing is generated according to 5.1a.1.3.5 using the parameters above if PAN is not included. If PAN is included in P1, then for generation of P1, N = 606 is used.

NOTE: When PAN is included, P1 calculated as above results in transmission of all systematic bits.

P2 (Type 2) puncturing is generated according to 5.1a.1.3.5.

P3 puncturing is generated according to 5.1a.1.3.5.

NOTE: P2 and P3 are always generated using N = 610 irrespective of whether or not P1 is with or without PAN.

If a PAN is not included, the result is four blocks of 620 bits,  $\{c1(0),...,c1(619)\}$ ,  $\{c2(0),...,c2(619)\}$ ,  $\{c3(0),...,c3(619)\}$  and  $\{c4(0),...,c4(619)\}$ .

If a PAN is included, the result is four blocks of 606 bits,  $\{c1(0),...,c1(605)\}$ ,  $\{c2(0),...,c2(605)\}$ ,  $\{c3(0),...,c3(605)\}$  and  $\{c4(0),...,c4(605)\}$ .

NOTE: C1 and c1 correspond to i1, C2 and c2 to i2, C3 and c3 to i3, and C4 and c4 to i4.

# 5.1a.31.5 PAN coding

The PAN  $\{pn(0),...,pn(24)\}$ , if included, is coded as defined in subclause 5.1a.1.4, resulting in a block of 90 bits,  $\{C(0),...,C(89)\}$ .

The code is punctured in such a way that the following coded bits:

 $\{C(15*k), C(15*k+2), C(15*k+4), C(15*k+7), C(15*k+10), C(15*k+13) \text{ for } k=0,1,...5\}$  are not transmitted; except C(15), C(45) which are transmitted.

This results in a block of 56 bits,  $\{ac(0),...,ac(55)\}$ .

## 5.1a.31.6 Interleaving

a) Header

The header interleaving is the same as for DBS-11 as specified in subclause 5.1a.30.6.

b) Data and PAN

If a PAN is not included, the following definitions apply:

$$\begin{split} dc1(k) &= c1(k) & \text{for } k = 0,...,619 \\ dc2(k) &= c2(k) & \text{for } k = 0,...,619 \\ dc3(k) &= c3(k) & \text{for } k = 0,...,619 \\ dc4(k) &= c4(k) & \text{for } k = 0,...,619 \end{split}$$

If a PAN is included, the following definitions apply:

$$\begin{array}{lll} dc1(k) = ac(4k) & \text{for } k = 0, \dots, 13 \\ \\ dc1(k) = c1(k-14) & \text{for } k = 14, \dots, 619 \\ \\ dc2(k) = ac(4k+1) & \text{for } k = 0, \dots, 13 \\ \\ dc2(k) = c2(k-14) & \text{for } k = 14, \dots, 619 \\ \\ dc3(k) = ac(4k+2) & \text{for } k = 0, \dots, 13 \\ \\ dc3(k) = c3(k-14) & \text{for } k = 14, \dots, 619 \\ \\ dc4(k) = ac(4k+3) & \text{for } k = 0, \dots, 13 \\ \\ dc4(k) = c4(k-14) & \text{for } k = 14, \dots, 619 \\ \end{array}$$

The four blocks  $\{dc1(0),...,dc1(619)\}$ ,  $\{dc2(0),...,dc2(619)\}$ ,  $\{dc3(0),...,dc3(619)\}$  and  $\{dc4(0),...,dc4(619)\}$  are separately interleaved as defined in subclause 5.1a.2.2, with  $N_C=620$  and a=141, resulting in the four blocks  $\{di1(0),...,di1(619)\}$ ,  $\{di2(0),...,di2(619)\}$ ,  $\{di3(0),...,di3(619)\}$  and  $\{di4(0),...,di4(619)\}$ , where di1 corresponds to dc1, di2 to dc2, di3 to dc3 and di4 to dc4.

The blocks are put together as one entity as described by the following rule:

$$di(k) = di1(k)$$
 for  $k = 0,...,619$ 

$$di(k) = di2(k-620)$$
 for  $k = 620,...,1239$   
 $di(k) = di3(k-1240)$  for  $k = 1240,...,1859$   
 $di(k) = di4(k-1860)$  for  $k = 1860,...,2479$ 

# 5.1a.31.7 Mapping on a burst

a) Straightforward mapping

The mapping is the same as for DBS-11 as specified in subclause 5.1a.30.7 a).

b) Bit swapping

The bit swapping is the same as for DBS-11 as specified in subclause 5.1a.30.7.b).

c) PAN bit swapping

In case a PAN is included in the radio block, the following additional bits are swapped after the bit swapping in b):

For B = 0, 1, 2 and 3

Swap e(B,29) with e(B,158)

Swap e(B,141) with e(B,80)

Swap e(B,226) with e(B,110)

Swap e(B,282) with e(B,98)

Swap e(B,381) with e(B,480)

Swap e(B,437) with e(B,433)

Swap e(B,522) with e(B,503)

Swap e(B,634) with e(B,410)

# 5.1b Extended Coverage Packet data traffic channel (EC-PDTCH)

# 5.1b.1 General

Twelve coding schemes are specified for the Extended Coverage Packet data traffic channels. The coding schemes MCS-1 to MCS-9 are identical to those defined for PDTCH in subclause 5.1 and apply for downlink and uplink coverage class CC1. The coding schemes MCS-1/16, MCS-1/8 and MCS-1/4 for downlink and uplink coverage classes CC4, CC3 and CC2, respectively, are based on MCS-1 defined in subclause 5.1. The coding scheme MCS-1/48 for uplink coverage class CC5 is specified in subclause 5.1b.4.

The following additions apply:

- Blind physical layer transmissions are employed.
- For downlink, multiple sequences of USF bits are delivered to the encoder. Each precoded USF sequence is mapped onto one instance of the repeated block.

The following restrictions apply:

- Reduced Transmission Time Interval (RTTI) shall not be used on EC-PDTCH.
- A PAN field shall not be included in the message delivered to the encoder.

- For data coding, a reduced set of puncturing schemes is used on an EC-PDTCH, compared to a PDTCH. Only the puncturing schemes listed in table 0 shall be used.

Table 0: Allowed puncturing schemes on EC-PDTCH

MCS	Allowed puncturing schemes in subclause 5.1	
MCS-1/16		
MCS-1/8		
MCS-1/4	P1	
MCS-1		
MCS-2		
MCS-3	P1 or P2	
MCS-4	FIUIP2	
Other MCSs	No restriction	

## 5.1b.2 Downlink MCS-1/M

#### 5.1b.2.1 Block constitution

The message delivered to the encoder has a fixed size of 206 information bits  $\{a(0),a(1),...,a(205)\}$ . In addition, M sequences of USF bits  $\{u(m,0),u(m,1),u(m,2)\}$ , m=0,...,M-1, are delivered to the encoder, where M=16, 8 and 4 for MCS-1/16, MCS-1/8 and MCS-1/4, respectively.

# 5.1b.2.2 Encoding and blind physical layer transmission

For each of the M blind physical layer transmissions, the block is encoded as specified for MCS-1 in subclause 5.1.5.1. The message  $\{d(0),d(1),...,d(208)\}$  delivered to the encoder (see subclause 5.1.5.1.1) for blind physical layer transmission k is defined as

d(k)=u(m,k) for k=0,1,2;d(k)=a(k-3) for k=3,...,208.

Denote the resulting bursts from encoding of blind physical layer transmissions m as R(m,B,j) = e(B,j) for m=0,...,M-1, B=0,...,3 and j=0,...,115, where e(B,j) is defined in subclause 5.1.5.1.6.1.

# 5.1b.2.3 Mapping onto PDCHs

Burst R(m,B,j), j=0,...,115, is mapped onto burst B' of PDCH number pn, where

- for EC channels with 4 PDCHs assigned

B' = B + 4(m div 4); pn = (m mod 4).

- for EC channels with 2 PDCHs assigned

B' = B + 4(m div 2);pn = (m mod 2).

NOTE 1: The burst number B' denotes the relative transmission order of the bursts on a PDCH. The mapping to absolute TDMA frame number is specified in 3GPP TS 45.002 [8] and 3GPP TS 44.060 [6].

NOTE 2: pn denotes the relative PDCH number within the set of PDCHs on which the EC-PDTCH is mapped, where PDCHs are numbered from lower to higher timeslot number. The absolute timeslot number depends on the assignment, see 3GPP TS 44.060 [6].

# 5.1b.3 Uplink MCS-1/M

#### 5.1b.3.1 Block constitution

The message delivered to the encoder has a fixed size of 209 information bits  $\{a(0), a(1), ..., a(208)\}$ .

# 5.1b.3.2 Encoding and blind physical layer transmission

For each of the M blind physical layer transmissions the block is encoded as specified for MCS-1 in subclause 5.1.5.2. The message  $\{d(0),d(1),...,d(208)\}$  delivered to the encoder (see subclause 5.1.5.1.1) for blind physical layer transmission m is defined as

$$d(k)=a(k)$$
 for  $k=0,...,208$ .

Denote the resulting bursts from encoding of blind physical layer transmission m as R(m,B,j) = e(B,j) for m=0,...,M-1, B=0,...,3 and j=0,...,115, where e(B,j) is defined in subclause 5.1.5.1.6.1.

# 5.1b.3.3 Mapping onto PDCHs

The mapping is done as specified for MCS-1/M DL in subclause 5.1b.2.3, except when 2 PDCHs are assigned. In this case, burst R(m,B,j), j=0,...,115, is mapped onto burst B' of PDCH number pn, where:

$$B' = B(M \text{ div } 2) + (m \text{ div } 2);$$
 $pn = (m \text{ mod } 2).$ 

# 5.1b.4 Uplink MCS-1'/48

## 5.1b.4.1 Block constitution

The message delivered to the encoder has a fixed size of 194 information bits  $\{d(0),d(1),...,d(193)\}$ . It is delivered on a burst mode.

#### 5.1b.4.2 Header coding

a) Parity bits:

Eight header parity bits p(0),p(1),...,p(7) are defined in such a way that in GF(2) the binary polynomial:

$$d(0)D^{23} + ... + d(15)D^{8} + p(0)D^{7} + ... + p(7)$$
, when divided by:

 $D^8 + D^6 + D^3 + 1$ , yields a remainder equal to:

$$D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$$
.

b) Tail biting:

The six last header parity bits are added before information and parity bits, the result being a block of 30 bits  $\{u''(-6),...,u''(0),u''(1),...,u''(23)\}$  with six negative indexes:

$$\begin{array}{ll} u"(k\text{-}6) & = p(k+2) & \quad \text{for } k = 0,1,...,5 \\ \\ u"(k) & = d(k) & \quad \text{for } k = 0,1,...,15 \end{array}$$

$$u''(k) = p(k-16)$$
 for  $k = 16,17,...,23$ 

c) Convolutional encoder

This block of 30 bits  $\{u''(-6),...,u''(0),u''(1),...,u''(23)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 72 coded bits:  $\{C(0),C(1),...,C(71)\}\$  defined by:

$$C(3k) = u''(k) + u''(k-2) + u''(k-3) + u''(k-5) + u''(k-6)$$

$$C(3k+1) = u''(k) + u''(k-1) + u''(k-2) + u''(k-3) + u''(k-6)$$

$$C(3k+2) = u''(k) + u''(k-1) + u''(k-4) + u''(k-6)$$
 for  $k = 0,1,...,23$ 

The code is punctured in such a way that the following coded bits:

$$C(2+12j)$$
,  $C(5+12j)$ ,  $C(8+12j)$ ,  $C(11+12j)$  for  $j = 0,1,...,5$  are not transmitted

The result is a block of 48 coded bits,  $\{hc(0),hc(1),...,hc(47)\}$ .

# 5.1b.4.3 Data coding

a) Parity bits:

Twelve data parity bits p(0), p(1), ..., p(11) are defined in such a way that in GF(2) the binary polynomial:

$$d(16)D^{189} + ... + d(193)D^{12} + p(0)D^{11} + ... + p(11)$$
, when divided by:

$$D^{12} + D^{11} + D^{10} + D^8 + D^5 + D^4 + 1$$
, yields a remainder equal to:

$$D^{11} + D^{10} + D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$$
.

b) Tail bits:

Six tail bits equal to 0 are added to the information and parity bits, the result being a block of 196 bits  $\{u(0),u(1),...,u(195)\}$ :

$$u(k) = d(k+16)$$
 for  $k = 0,1,...,177$ 

$$u(k) = p(k-178)$$
 for  $k = 178, 179, ..., 189$ 

$$u(k) = 0$$
 for  $k = 190,191,...,195$  (tail bits)

c) Convolutional encoder

This block of 196 bits  $\{u(0),u(1),...,u(195)\}$  is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of 588 coded bits:  $\{C(0),C(1),...,C(587)\}\$  defined by:

$$C(3k) = u(k) + u(k-2) + u(k-3) + u(k-5) + u(k-6)$$

$$C(3k+1) = u(k) + u(k-1) + u(k-2) + u(k-3) + u(k-6)$$

$$C(3k+2) = u(k) + u(k-1) + u(k-4) + u(k-6) \qquad \text{for } k = 0, 1, ..., 195; \ u(k) = 0 \ \text{for } k < 0$$

The code is punctured using always P1 as defined below.

P1	$\{C(2+21j), C(5+21j), C(8+21j), C(10+21j), C(11+21j), C(14+21j), C(17+21j), C(20+21j) \text{ for } j=1,2,3,3,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4$
	$0,1,,27$ } are not transmitted except {C(k) for k = 73,136,199,262,325,388,451,514} which are
	transmitted

The result is a block of 372 coded bits,  $\{dc(0),dc(1),...,dc(371)\}$ .

# 5.1b.4.4 Interleaving

The header and data are put together as one entity as described by the following rule:

$$c(k) = hc(k) \qquad \text{for } k = 0,1,...,47$$
 
$$c(k) = dc(k-48)) \qquad \text{for } k = 48,49,...,419$$
 
$$c'(n,k) = c(n,k) \qquad \text{for } k = 0,1,...,24$$
 
$$c'(n,k) = c(n,k-1) \qquad \text{for } k = 26,27,...,81$$
 
$$c'(n,k) = c(n,k-2) \qquad \text{for } k = 83,84,...,138$$
 
$$c'(n,k) = c(n,k-3) \qquad \text{for } k = 140,141,...,400$$
 
$$c'(n,k) = c(n,k-4) \qquad \text{for } k = 402,403,...,423$$
 
$$c'(n,25) = q(10) \qquad c'(n,82) = q(11) \qquad c'(n,139) = q(12) \qquad c'(n,401) = q(13)$$

c(n,k) are the coded bits and q(10),q(11),q(12),q(13) = 0,0,0,0 are four extra stealing flags

The resulting block of 424 bits is interleaved according to the following rule:

$$i(B,j) = c'(n,k)$$
 for  $k = 0,1,...,423$  
$$n = 0,1,...,N,N+1,...$$
 
$$B = B_0 + 4n + (k \text{ mod } 4)$$
 
$$j = 2((49k) \text{ mod } 53) + ((k \text{ mod } 8) \text{ div } 4)$$

# 5.1b.4.5 Mapping on a burst

The mapping is given by the rule:

$$e(B,j) = i(B,j)$$
 and  $e(B,63+j) = i(B,53+j)$  for  $j = 0,1,...,52$ 

and

$$e(B+m,53+i) = q(i)$$
 and  $e(B+58+i) = q(i+5)$  for  $m = 0,1,2,3$  and  $i = 0,1,2,...,4$ 

where

$$q(0),q(1),..,q(9) = 0,0,0,0,0,0,0,0,0,0$$

# 5.1b.4.6 Blind physical layer transmission

The resulting bursts from encoding of blind physical layer transmission m are denoted as R(m,B,j) = e(B,j) for m=0,...,47, B=0,...,3 and j=0,...,115, where e(B,j) is defined in subclause 5.1b.4.5.

# 5.1b.4.7 Mapping onto PDCHs

Burst R(m,B,j), j=0,...,115, is mapped onto burst B' of PDCH number pn, where

- for EC channels with 4 PDCHs assigned

```
B' = B + 4*(m \ div \ 4); pn = (m \ mod \ 4). - for EC \ channels \ with \ 2 \ PDCHs \ assigned B' = B(M \ div \ 2) + (m \ div \ 2); pn = (m \ mod \ 2).
```

NOTE 1: The burst number B' denotes the relative transmission order of the bursts on a PDCH. The mapping to absolute TDMA frame number is specified in 3GPP TS 45.002 [8] and 3GPP TS 44.060 [6].

NOTE 2: pn denotes the relative PDCH number within the set of PDCHs on which the EC-PDTCH is mapped, where PDCHs are numbered from lower to higher timeslot number. The absolute timeslot number depends on the assignment, see 3GPP TS 44.060 [6].

# 5.2 Packet control channels (PACCH, PBCCH, PAGCH, PPCH, PTCCH, CPBCCH, CPAGCH and CPPCH)

The coding scheme used for PACCH, PBCCH, PAGCH, PPCH, downlink PTCCH, CPBCCH, CPAGCH and CPPCH is the same as for CS-1 as specified in subclause 5.1.1.

In RTTI configuration, the channel coding for the downlink PACCH may also be the one defined for MCS-0 in subclause 5.1.4a.

In a DLMC configuration (see 3GPP TS 44.060), the coding scheme for the uplink PACCH may also be the one defined for CS-3 in subclause 5.1.3.

The coding scheme used for uplink PTCCH is the same as for PRACH as specified in subclause 5.3.

In the case of a downlink PACCH including an eTFI field (see 3GPP TS 44.060) the message delivered to the encoder will have a fixed size of 187 information bits  $\{d(0),d(1),...,d(186)\}$ . The first 184 bits will be protected by 40 parity bits as specified in subclause 4.1.2 a). Bit-wise modulo 2 addition is performed using a subset of the parity bits. Which parity bits are used is dependent on the state of the three bits  $\{d(184),d(185),d(186)\}$ . These three bits represent the assigned eTFI value and are converted to their equivalent decimal representation, et  $\{0,1,2...7\}$ .  $d\{186\}$  represents the least significant bit.

This results in the forty modified parity bits  $\{pt(0),...,pt(39)\}$  defined as:

```
pt(k*10+et) = 1 + p(k*10+et) \qquad \text{for } k=0,1,2 pt(32+et) = 1 + p(32+et) pt(k) = p(k) \qquad \text{for the remaining 36 parity bits}
```

Four tail bits equal to 0 are then added to the information and modified parity bits as specified in subclause 4.1.2 b with p(k) replaced by p(k). Convolutional encoding, Interleaving and Mapping on a Burst then follows the process outlined in subclauses 4.1.3, 4.1.4 and 4.1.5 respectively.

# 5.2a Extended Coverage broadcast control channel (EC-BCCH)

# 5.2a.1 Block constitution

The message delivered to the encoder has a fixed size of 184 information bits {d(0),d(1),...,d(183)}.

# 5.2a.2 Coding

The message is coded as for SACCH as specified in subclause 4.1, resulting in the bursts e(B,j), B=0,1,2,3 and j=0,1,...,115.

# 5.2a.3 Blind physical layer transmission

Each burst is transmitted 16 times. Denote the resulting bursts of transmission m as R(m,B,j) = e(B,j) for m=0,...,15, B=0,1,2,3 and j=0,...,115.

# 5.2a.4 Mapping onto a physical channel

Burst R(m,B,j), j=0...,115, is mapped onto burst B' of the timeslot carrying the EC-BCCH, where

$$B' = B + 4m$$

NOTE: The burst number B' denotes the relative transmission order of the bursts on the timeslot. The mapping to absolute TDMA frame number is specified in 3GPP TS 45.002 [8].

# 5.2b Extended coverage common control channel downlink (EC-CCH/D/M), Extended coverage paging indication channel (EC-PICH) and Extended coverage packet associated control channel (EC-PACCH/M)

# 5.2b.1 Block constitution

The message delivered to the encoder has a fixed size of  $K_d$  information bits  $\{d(0), d(1), ..., d(K_d-1)\}$ , where:

- EC-CCCH/D, EC-PICH: K<sub>d</sub>=88
- EC-PACCH/D: K<sub>d</sub>=80
- EC-PACCH/U/M: K<sub>d</sub>=64 for M=1,4,8,16
- EC-PACCH/U/M: K<sub>d=</sub>56 for M=48

In addition, for EC-PACCH/D, up to M sequences of USF bits  $\{u(m,0),u(m,1),u(m,2)\}$ , m=0,...,M-1, are delivered to the encoder, where M=16, 8, 4 or 1 depending on the number of blind physical layer transmissions.

NOTE: USF bits need not be delivered for those blocks that are transmitted on timeslots no GPRS/EGPRS MS is monitoring for USF.

# 5.2b.2 Data coding

a) Parity bits:

Eighteen data parity bits p(0),p(1),...,p(17) are defined in such a way that in GF(2) the binary polynomial:

$$d(0)D^{K}{}_{d}{}^{+18-1} + ... + d(K_{d}{}^{-1})D^{18} + p(0)D^{17} + ... + p(17), \text{ when divided by:}$$
 
$$D^{18} + D^{17} + D^{14} + D^{13} + D^{11} + D^{10} + D^{8} + D^{7} + D^{6} + D^{3} + D^{2} + 1, \text{ yields a remainder equal to:}$$
 
$$D^{17} + D^{16} + D^{15} + D^{14} + D^{13} + D^{12} + D^{11} + D^{10} + D^{9} + D^{8} + D^{7} + D^{6} + D^{5} + D^{4} + D^{3} + D^{2} + D + 1.$$

The parity bits are added after the block of  $K_d$  bits, the result being a block of  $K_{d+18}$  bits,  $\{b(0),...,b(K_{d+18-1})\}$ , defined as:

$$b(k) = d(k) \quad \text{for } k = 0,1,..., K_d-1$$
 
$$b(k) = p(k-K_d) \text{ for } k = K_d,..., K_d+18-1$$

#### b) Tail-biting convolutional encoder

The six last bits are added before the block of  $K_d+18$  bits, the result being a block of  $K_d+24$  bits  $\{c(-6),...,c(0),c(1),...,c(K_d+18-1)\}$  with six negative indices:

$$c(k) = b(K_d+18+k)$$
 for  $k = -6,...,-1$ 

$$c(k) = b(k)$$
 for  $k = 0,1,..., K_d+18-1$ 

This block of K<sub>d</sub>+24 bits is encoded with the 1/3 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

This results in a block of  $(K_d+18)*3$  coded bits  $\{C(0),...,C((K_d+18)*3-1)\}$  defined by:

$$C(3k) = c(k) + c(k-2) + c(k-3) + c(k-5) + c(k-6)$$

$$C(3k+1) = c(k) + c(k-1) + c(k-2) + c(k-3) + c(k-6)$$

$$C(3k+2) = c(k) + c(k-1) + c(k-4) + c(k-6)$$
 for  $k = 0,1,..., K_d+18-1$ 

The code is punctured in such a way that the following coded bits are not transmitted:

EC-CCCH/D, EC-PICH	C(floor(n*318/202)) for n=0,,201
EC-PACCH/U (M=1,4,8,16)	C(floor(n*246/130)) for n=0,,129
EC-PACCH/D	C(floor(n*294/180)) for n=0,,179
EC-PACCH/U	C(floor(n*222/116)) for n=0,,115
(M=48)	

The result is a block of  $K_{pc}$  coded bits {pc(0),pc(1),...,pc( $K_{pc}$ -1)}, where  $K_{pc}$ =114 for EC-PACCH/D and  $K_{pc}$ =116 for EC-CCCH/D, EC-PICH and EC-PACCH/U/1, EC-PACCH/U/4, EC-PACCH/U/8, EC-PACCH/U/16 and  $K_{pc}$ =106 for EC-PACCH/U/48.

# 5.2b.3 USF coding (only for EC-PACCH/D)

For each of the up to M delivered USF sequences, the USF bits are block coded as for Packet data block type 4 (CS-4) in subclause 5.1.4.2. The input d(0),d(1),d(2) in subclause 5.1.4.2 shall be  $\{u(m,0),u(m,1),u(m,2)\}$  for USF number m.

Denote the resulting twelve encoded bits of USF number m as {u'(m,0),u'(m,1),...,u'(m,11)}.

# 5.2b.4 Mapping on a Burst

## 5.2b.4.1 EC-CCCH/D/M and EC-PICH

The mapping is given by the rule:

$$e(B, j) = pc(j)$$
 for  $j = 0,1,...,115$  and  $B=0,1$ 

#### 5.2b.4.2 EC-PACCH/D/M

The mapping is given by the rule:

$$e(B,j) = pc(j)$$
 and  $e(B,59+j) = pc(57+j)$  for  $j = 0,1,...,56$  and  $B=0,1,2,3$ 

and

$$e(B,57) = q(2B)$$
 and  $e(B,58) = q(2B+1)$  for  $B = 0,1,2,3$ 

where

$$q(0),q(1),...,q(7) = 0,0,0,1,0,1,1,0.$$

NOTE: For a standard GPRS MS, bits q(0),...,q(7) indicates that the USF is coded as for CS-4.

## 5.2b.4.3 EC-PACCH/U/M

The mapping is given by the following rule for M = 1,4,8 or 16:

$$e(B, j) = pc(j)$$
 for  $j = 0,1,...,115$  and  $B=0,1,2,3$ 

The mapping is given by the following rule for M = 48:

$$e(B,j) = pc(j)$$
 and  $e(B,63+j) = pc(53+j)$  for  $j = 0,1,...,52$  and  $B=0,1,2,3$ 

and

$$e(B,53+i) = q(i)$$
 for  $i = 0,1,...,9$ 

where

$$q(0),q(1),...,q(9) = 0,0,0,0,0,0,0,0,0,0,0$$

# 5.2b.5 Blind physical layer transmission

Each burst is transmitted M times, where M is the number of blind physical layer transmissions. For EC-PACCH/D, M=1, 4, 8 or 16. For EC-PACCH/U, M=1, 4, 8, 16 or 48. For EC-CCCH/D, M=1, 8, 16 or 32. For EC-PICH, M=1, i.e. no blind physical layer transmission is applied.

Denote the resulting bursts of transmission m as R(m,B,j) = e(B,j) for m=0,...,M-1, B=0,1 (EC-CCCH) or B=0,1,2,3 (EC-PACCH), and j=0,...,115 or m=0, B=0,1 (EC-PICH).

# 5.2b.6 Bit stealing for USF (only for EC-PACCH/D)

For those m for which a USF has been delivered to the encoder, replace bits in the repeated bursts with encoded USF bits as follows:

for m=0,...,M-1

$$R(m,0,0) = u'(m,0)$$

$$R(m,0,51) = u'(m,4)$$

$$R(m,0,102) = u'(m,8)$$

$$R(m,1,100) = u'(m,1)$$

$$R(m,1,35) = u'(m,5)$$

$$R(m,1,86) = u'(m,9)$$

$$R(m,2,84) = u'(m,2)$$

$$R(m,2,19) = u'(m,6)$$

$$R(m,2,70) = u'(m,10)$$

$$R(m,3,68) = u'(m,3)$$

$$R(m,3,3) = u'(m,7)$$

$$R(m,3,52) = u'(m,11)$$

# 5.2b.7 Mapping onto physical channels

#### 5.2b.7.1 EC-PACCH

The mapping onto PDCHs is done as specified for downlink MCS-1/M in subclause 5.1b.2.3 for M=1,4,8,16. The mapping onto PDCHs is done as specified for uplink MCS-1'/48 in subclause 5.1b.4.3.

# 5.2b.7.2 EC-CCCH/D and EC-PICH

Burst R(m,B,j), B=0,1, j=0...,115, is mapped onto burst B' of the timeslot carrying the EC-CCCH/D or EC-PICH, where

$$B' = B + 2m$$
, for m=0,...,M-1 (EC-CCCH/D) and m=0 (EC-PICH)

NOTE: The burst number B' denotes the relative transmission order of the bursts on the timeslot. The mapping to absolute TDMA frame number is specified in 3GPP TS 45.002 [8].

# 5.3 Packet random access channel (PRACH, CPRACH and MPRACH)

Two coding schemes are specified for access bursts on the packet switched channels. The packet access burst containing 8 information bits and the extended packet access burst containing 11 information bits. Only the 11 information bits access burst may be transmitted on the CPRACH.

## 5.3.1 Packet Access Burst

The encoding of this burst is as defined in section 4.6 for the random access channel (RACH). The BSIC used shall be the BSIC of the BTS to which the burst is intended.

# 5.3.2 Extended Packet Access Burst

# 5.3.2.1 Block constitution

The burst carrying the extended packet random access uplink message contains 11 information bits d(0), d(1), ..., d(10).

# 5.3.2.2 Parity bits

Six parity bits p(0), p(1), ..., p(5) are defined in such a way that in GF(2) the binary polynomial:

$$d(0)D^{16} + ... + d(10)D^{6} + p(0)D^{5} + ... + p(5)$$
, when divided by  $D^{6} + D^{5} + D^{3} + D^{2} + D + 1$  yields a remainder equal to  $D^{5} + D^{4} + D^{3} + D^{2} + D + 1$ .

# 5.3.2.3 Addition of BSIC

An MS that has not enabled PEO or EC operation shall use the six bit BSIC of the BTS to which the access burst is intended. The six bits of the BSIC,  $\{b(0),b(1),...,b(5)\}$ , are added bitwise modulo 2 to the six parity bits,  $\{p(0),p(1),...,p(5)\}$ .

This results in six colour bits, C(0) to C(5) defined as C(k) = b(k) + p(k) (k = 0 to 5) where:

b(0) = MSB of PLMN colour code

b(5) = LSB of BS colour code.

This defines  $\{u(0), u(1), ..., u(20)\}$  by:

$$u(k) = d(k)$$
 for  $k = 0,1,...,10$ 

$$u(k) = C(k-11)$$
 for  $k = 11,12,...,16$ 

$$u(k) = 0$$
 for  $k = 17,18,19,20$  (tail bits)

An MS that has enabled PEO or EC operation shall use the nine bit BSIC of the BTS to which the access burst is intended . The first six bits of the BSIC,  $\{b(0),b(1),...,b(5)\}$ , are added bitwise modulo 2 to the six parity bits while the last three bits of the BSIC  $\{b(6),b(7),b(8)\}$  are added bitwise modulo 2 to the three last information bits d(8), d(9) and d(10). The bitwise modulo 2 operation results in nine colour bits, C(0) to C(8), defined as C(k) = b(k) + p(k) (for k = 0 to 5) and C(k) = b(k) + d(k+2) (for k = 6 to 8) where:

- b(0) = MSB of PLMN colour code
- b(8) = LSB of Radio frequency colour code.

This defines  $\{u(0), u(1), ..., u(20)\}$  by:

$$\begin{array}{ll} u(k) = d(k) & \text{for } k = 0,1,...,7 \\ \\ u(k) = C(k\text{-}2) & \text{for } k = 8,9,10 \\ \\ u(k) = C(k\text{-}11) & \text{for } k = 11,12,...,16 \\ \\ u(k) = 0 & \text{for } k = 17,18,19,20 \text{ (tail bits)} \end{array}$$

#### 5.3.2.4 Convolutional code

The coded bits  $\{c(0),c(1),...,c(41)\}$  are obtained by the same convolutional code of rate ½ as for TCH/FS, defined by the polynomials:

$$G0 = 1 + D^3 + D^4$$
  
 $G1 = 1 + D + D^3 + D^4$ 

and with:

$$\begin{split} c(2k) &= u(k) + u(k-3) + u(k-4) \\ c(2k+1) &= u(k) + u(k-1) + u(k-3) + u(k-4) \qquad \text{ for } k = 0,1,...,20 \ ; \ u(k) = 0 \ \text{for } k < 0 \end{split}$$

The code is punctured in such a way that the following coded bits:

$$c(0)$$
,  $c(2)$ ,  $c(5)$ ,  $c(37)$ ,  $c(39)$ ,  $c(41)$  are not transmitted.

This results in a block of 36 coded bits,  $\{e(0), e(1), \dots, e(35)\}$ .

# 5.3a Extended coverage random access channel (EC-RACH/M)

#### 5.3a.0 General

Channel encoding for the random access message in EC operation containing 11 information bits is depicted in the following sub-clauses 5.3a.1 to 5.3a.4, except for EC-RACH/66 (for ESAB format) and EC-RACH/132 (for EDAB format) the channel encoding are specified in clause 5.3a.6 and 5.3a.7. Sub-clause 5.3a.5 deals with the channel encoding for the random access message in EC operation containing 30 information bits.

# 5.3a.1 Block constitution

The message delivered to the encoder has a fixed size of 11 information bits  $\{d(0),d(1),...,d(10)\}$ .

# 5.3a.2 Coding

The message is encoded as for the 11 information bit access defined in subclause 5.3.2, resulting in a block of 36 coded bits  $\{e(0), e(1), ..., e(35)\}$ .

# 5.3a.3 Blind physical layer transmissions

The block is transmitted M times, where M=1, 4, 16 or 48 for 1 TS EC-RACH and M=4, 16 or 48 for 2 TS EC-RACH, resulting in the repeated bursts R(m,B,j)=e(j) for m=0,...,M-1, B=0 and j=0,...,35.

# 5.3a.4 Mapping onto physical channel(s)

#### 5.3a.4.1 1 TS EC-RACH

Burst R(m,B,j), B=0, j=0...,35, is mapped onto burst B' of the timeslot carrying the EC-RACH, where

B' = m

NOTE: The burst number B' denotes the relative transmission order of the bursts on the timeslot. The mapping to absolute TDMA frame number is specified in 3GPP TS 45.002 [8].

#### 5.3a.4.2 2 TS EC-RACH

Burst R(m,B,j), B=0, j=0...,35, is mapped onto burst B' of timeslot number tn of the timeslots carrying the EC-RACH, where

 $B' = m \operatorname{div} 2$ 

 $tn = m \mod 2$ 

NOTE 1: The burst number B' denotes the relative transmission order of the bursts on the timeslot. The mapping to absolute TDMA frame number is specified in 3GPP TS 45.002 [8].

NOTE 2: tn denotes the relative timeslot number within the timeslot pair on which the EC-RACH is mapped. The absolute timeslot number depends on the configuration, see 3GPP TS 44.018 [4].

# 5.3a.5 EC-RACH carrying 30 information bits

In case of performing the Multilateration Timing Advance procedure using the Extended Access Burst method in EC operation, i.e. the mobile station has selected Coverage Class 1 for the uplink and the downlink, the MS sends a random access message carrying 11 information bits using the encoding according to the above sub-clauses, and after access grant by the network sends a subsequent random access message carrying 30 information bits. The encoding of the 30 information bits follows the one for (EC-)RACH specified in subclause 4.6.2.

# 5.3a.6 Extended coverage random access channel (EC-RACH/66)

# 5.3a.6.0 General

If ESAB format (see 3GPP TS 45.002) is used for EC-RACH in CC5 operation the channel coding, blind physical layer transmissions and mapping onto physical channels are depicted in the subclause 5.3a.6.

Channel encoding, blind physical layer transmissions and mapping onto physical channels for the random access channel EC-RACH/66 used for CC5 in EC operation is depicted in the following sub-clauses 5.3a.6.1 to 5.3a.6.6.

#### 5.3a.6.1 Block constitution

The message delivered to the encoder has a fixed size of 11 information bits  $\{d(0),d(1),...,d(10)\}$ .

#### 5.3a.6.2 Parity bits

Six parity bits are added as defined in subclause 5.3.2.2

## 5.3a.6.3 Addition of BSIC

The first six bits of the BSIC,  $\{b(0),b(1),...,b(5)\}$ , are added bitwise modulo 2 to the six parity bits while the last three bits of the BSIC  $\{b(6),b(7),b(8)\}$  are added bitwise modulo 2 to the three last information bits d(8), d(9) and d(10). The bitwise modulo 2 operation results in nine colour bits, C(0) to C(8), defined as C(k) = b(k) + p(k) (for k = 0 to 5) and C(k) = b(k) + d(k+2) (for k = 6 to 8) where:

- b(0) = MSB of PLMN colour code
- b(8) = LSB of Radio frequency colour code.

This defines a block of 17 bits  $\{u(0),u(1),...,u(16)\}$  by:

$$u(k) = d(k)$$
 for  $k = 0,1,...,7$ 

$$u(k) = C(k-2)$$
 for  $k = 8,9,10$ 

$$u(k) = C(k-11)$$
 for  $k = 11,12,...,16$ 

# 5.3a.6.4 Tail biting Convolutional code

The six last bits are added before the block of 17 bits, the result being a block of 23 bits  $\{c(-6),...,c(0),c(1),...,c(17-1)\}$  with six negative indices:

$$c(k) = u(23+k)$$
 for  $k = -6,...,-1$ 

$$c(k) = u(k)$$
 for  $k = 0, 1,...,17-1$ 

The block of 23 bits is encoded with 1/6 rate convolutional mother code defined by the polynomials:

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G4 = 1 + D^2 + D^3 + D^5 + D^6$$

$$G7 = 1 + D + D^2 + D^3 + D^6$$

$$G5 = 1 + D + D^4 + D^6$$

$$G6 = 1 + D + D^2 + D^3 + D^4 + D^6$$

$$G6 = 1 + D + D^2 + D^3 + D^4 + D^6$$

This results in a block of 17\*6 coded bits  $\{e(0), e(1), \dots, e(101)\}$  defined by:

$$e(6k) = c(k) + c(k-2) + c(k-3) + c(k-5) + c(k-6)$$

$$e(6k+1) = c(k) + c(k-2) + c(k-3) + c(k-5) + c(k-6)$$

$$e(6k+2) = c(k) + c(k-1) + c(k-2) + c(k-3) + c(k-6)$$

$$e(6k+3) = c(k) + c(k-1) + c(k-4) + c(k-6)$$

$$e(6k+4) = c(k) + c(k-1) + c(k-2) + c(k-3) + c(k-4) + c(k-6)$$

$$e(6k+5) = c(k) + c(k-1) + c(k-2) + c(k-3) + c(k-4) + c(k-6)$$
 for  $k = 0,1,...,16$ 

# 5.3a.6.5 Blind physical layer transmissions

The block is transmitted M times, where M=1, 4, 16 or 48 for 1 TS EC-RACH, M=4, 16 or 48 for 2 TS EC-RACH, resulting in the repeated bursts R(m,B,j)=e(j) for m=0,...,M-1, B=0 and j=0,...,35, and M=66 for EC-RACH CC5 using ESAB burst, resulting in the repeated ESAB bursts R(m,B,j)=e(j) for m=0,...,65, B=0 and j=0,...,101.

# 5.3a.6.6 Mapping onto physical channel(s)

For EC-RACH CC5/66, burst R(m,B,j), B=0, j=0...,101, is mapped onto ESAB burst B'' on timeslot pair tn' carrying the EC-RACH CC5, where

$$B'' = m$$

NOTE 3: The burst number B" denotes the relative transmission order of the ESAB bursts on 2 timeslots. The mapping to absolute TDMA frame number is specified in 3GPP TS 45.002 [8].

NOTE 4: tn' denotes the relative timeslot pair number on which the EC-RACH CC5 is mapped. The absolute timeslot pair depends on the configuration, see 3GPP TS 44.018 [4].

# 5.3a.7 Extended coverage random access channel (EC-RACH/132)

#### 5.3a.7.0 General

If EDAB format (see 3GPP TS 45.002) is used for EC-RACH in CC5 operation the channel coding, blind physical layer transmissions and mapping onto physical channels are depicted in the subclause 5.3a.7.

Channel encoding, blind physical layer transmissions and mapping onto physical channels for the random access channel EC-RACH/132 used for CC5 in EC operation is depicted in the following sub-clauses 5.3a.6.1 to 5.3a.6.6.

#### 5.3a.7.1 Block constitution

The message delivered to the encoder has a fixed size of 11 information bits  $\{d(0),d(1),...,d(10)\}$ .

# 5.3a.7.2 Parity bits

Six parity bits are added as defined in subclause 5.3.2.2

# 5.3a.7.3 Addition of BSIC

9 bits of BSIC information is added over 6 parity bits and 3 data bits as defined in subclause 5.3.2.3.

# 5.3a.7.4 Convolutional code

The coded bits  $\{c(0),c(1),...,c(41)\}$  are obtained by the same convolutional code of rate ½ as for TCH/FS, defined by the polynomials:

$$G0 = 1 + D^3 + D^4$$

$$G1 = 1 + D + D^3 + D^4$$

and with:

$$c(2k) = u(k) + u(k-3) + u(k-4)$$

$$c(2k+1) = u(k) + u(k-1) + u(k-3) + u(k-4)$$
 for  $k = 0,1,...,20$ ;  $u(k) = 0$  for  $k < 0$ 

The code is punctured in such a way that the following coded bits:

$$[c(0), c(3), c(7), c(12), c(14), c(17), c(24), c(27), c(31), c(34), c(39), c(41)]$$
 are not transmitted.

This results in a block of 30 coded bits,  $\{e(0), e(1), \dots, e(29)\}$ .

# 5.3a.7.5 Blind physical layer transmissions

The block of encoded data is transmitted 264 times using the Segmented AB for the 2 TS EC-RACH format (see 3GPP TS 45.002[8]). The first burst with normal burst size carries 3 instances of the encoded data block and the second burst with access burst size carries one instance of the encoded data block. The bursts are transmitted in a timeslot pair which is repeated over 66 TDMA frames forming 132 bursts in total.

# 5.3a.7.6 Mapping onto physical channel(s)

The data block {e(0),...,e(29)} is mapped onto the Segmented AB, repeated over 66 TDMA frames in total, as specified in 3GPP TS 45.002[8].

For tn = TN mod 2 = 0, the data block  $\{e(0), e(1), \dots, e(29)\}$  is mapped 3 times on the burst with normal burst size.

For tn = TN mod 2 = 1, the data block  $\{e(0), e(1), \dots, e(29)\}$  is mapped once on the burst with access burst size.

NOTE: tn denotes the relative timeslot number within the timeslot pair on which the EC-RACH is mapped. The absolute timeslot number depends on the configuration, see 3GPP TS 44.018 [4].

# 5.4 Access Burst on packet switched channels other than PRACH, CPRACH and MPRACH

The encoding of this burst is as defined in subclause 5.3 for the packet random access channel (PRACH). The BSIC used shall be the BSIC of the BTS to which the burst is intended.

An MS in EC operation sending a block on EC-PACCH/U as four access bursts shall use blind physical layer transmissions of each access burst according to the assigned UL Coverage Class. The blind physical layer transmissions and mapping onto physical channels shall be as described for EC-PACCH/U in subclause 5.2b.

Table 1: Reordering and partitioning of a coded block of 456 bits into 8 sub-blocks

k mod 8=	0	1	2	3
j=0	k=0	57	114	171
2	64	121	178	235
4	128	185	242	299
6	192	249	306	363
8	256	313	370	427
10	320	377	434	35
	384	441	42	99
	448	49	106	163
	56	113	170	227
	120	177	234	291
20	184	241	298	355
20	248	305	362	419
	312	369	426	27
	376	433	34	91
	440	41	98	155
30	48	105	162	219
30	112	169	226	283
				1
	176	233	290	347
	240	297	354	411
40	304	361	418	19
40	368	425	26	83
	432	33	90	147
	40	97	154	211
	104	161	218	275
	168	225	282	339
50	232	289	346	403
	296	353	410	11
	360	417	18	75
	424	25	82	139
	32	89	146	203
60	96	153	210	267
	160	217	274	331
	224	281	338	395
	288	345	402	3
	352	409	10	67
70	416	17	74	131
	24	81	138	195
	88	145	202	259
	152	209	266	323
	216	273	330	387
80	280	337	394	451
	344	401	2	59
	408	9	66	123
	16	73	130	187
	80	137	194	251
90	144	201	258	315
55	208	265	322	379
	272	329	386	443
	336	393	450	51
400	400	1	58	115
100	8	65	122	179
	72	129	186	243
	136	193	250	307
	200	257	314	371
	264	321	378	435
110	328	385	442	43
112	392	449	50	107

k mod 8=	4	5	6	7
j=1	228	285	342	399
3	292	349	406	7
5	356	413	14	71
7	420	21	78	135
9	28	85	142	199
11	92	149	206	263
	156	213	270	327
	220	277	334	391
	284	341	398	455
	348	405	6	63
21	412	13	70	127
	20	77	134	191
	84	141	198	255
	148	205	262	319
	212	269	326	383
31	276	333	390	447
	340	397	454	55
	404	5	62	119
	12	69	126	183
	76	133	190	247
41	140	197	254	311
	204	261	318	375
	268	325	382	439
	332	389	446	47
	396	453	54	111
51	4	61	118	175
	68	125	182	239
	132	189	246	303
	196	253	310	367
	260	317	374	431
61	324	381	438	39
	388	445	46	103
	452	53	110	167
	60	117	174	231
	124	181	238	295
71	188	245	302	359
	252	309	366	423
	316	373	430	31
	380	437	38	95 150
04	444	45	102	159
81	52 116	109	166	223
	116 180	173 237	230 294	287 351
	244	301	358	415
	308	365	422	23
91	372	429	30	23 87
31	436	37	94	151
	44	101	158	215
	108	165	222	279
	172	229	286	343
101	236	293	350	407
	300	357	414	15
	364	421	22	79
	428	29	86	143
	36	93	150	207
111	100	157	214	271
113	164	221	278	335
				550

Table 2: Subjective importance of encoded bits for the full rate speech TCH (Parameter names and bit indices refer to 3GPP TS 46.010)

Importance class	Parameter name	Parameter number	Bit index	Label	Class
1	Log area ratio 1	1	5	d0	
	block amplitude	12,29,46,63	5	d1, d2, d3, d4	
	Log area ratio 1	1	4		
2	Log area ratio 2	2	5		
	Log area ratio 3	3	4		
	Log area ratio 1	1	3		
	Log area ratio 2 Log area ratio 3	2 3	4 3		
	Log area ratio 4	4	4		
3	LPT lag	9,26,43,60	6		1
J	block amplitude	12,29,46,63	4		with
	Log area ratio 2,5,6	2,5,6	3		parity
	LPT lag	9,26,43,60	5		check
	LPT lag	9,26,43,60	4		
	LPT lag	9,26,43,60	3		
	LPT lag	9,26,43,60	2		
	block amplitude	12,29,46,63	3		
	Log area ratio 1	1	2		
	Log area ratio 4	4	3		
	Log area ratio 7	7	2	140 140	
4	LPT lag	9,26,43,60	1	d48, d49	
	Log area ratio 5,6	5,6	2	d50	
	LPT gain	10,27,44,61	1		
	LPT lag Grid position	9,26,43,60 11,28,45,62	0		
	Log area ratio 1	1	1		
	Log area ratio 2,3,8,4	2,3,8,4	2		
	Log area ratio 5,7	5,7	1 1		
	LPT gain	10,27,44,61	0		
	block amplitude	12,29,46,63	2		1
	RPE pulses	1325	2		without
	RPE pulses	3042	2		parity
5	RPE pulses	4759	2		check
	RPE pulses	6476	2		
	Grid position	11,28,45,62	0		
	block amplitude	12,29,46,63	1		
	RPE pulses	1325	1 1		
	RPE pulses RPE pulses	3042 4759			
	RPE pulses	6467		d181	
	RPE pulses	6876	1	d182	
	Log area ratio 1	1	0	u102	
	Log area ratio 2,3,6	2,3,6	1		
	Log area ratio 7	7	Ö		
	Log area ratio 8	8	1		
	Log area ratio 8,3	8,3	0		
6	Log area ratio 4	4	1		2
	Log area ratio 4,5	4,5	0		
	block amplitude	12,29,46,63	0		
	RPE pulses	1325	0		
	RPE pulses	3042	0		
	RPE pulses	4759	0		
	RPE pulses	6476	0	طمحم	
	Log area ratio 2,6	2,6	0	d259	

Table 3a: Subjective importance of encoded bits for the half rate speech TCH for unvoiced speech frames (Parameter names and bit indices refer to 3GPP TS 46.020)

Parameter	Bit	Label	Class
name	index		
R0	1	d0	
LPC 3	7	d1	
GSP 0-1	2	d2	
GSP 0-2	2	d3	
GSP 0-3	2	d4	
GSP 0-4	2	d5	
LPC 1	0	d6	
LPC 2	51	d7d11	
LPC 3	61	d12	
Code 1-2	01	u 12	
Code 1-2 Code 2-2	60		
			4
Code 1-3	60		1
Code 2-3	63		
LPC3	0		without
R0	0		parity
INT-LPC	0		check
Code 1-2	16		
Code 2-1	06		
Code 1-1	06		
GSP 0-4	0		
GSP 0-3	0		
GSP 0-2	0		
GSP 0-1	0		
LPC 2	0		
GSP 0-4	1		
GSP 0-3	1		
GSP 0-2	1		
GSP 0-1	1		
LPC 1	14	d72	
LPC 1	5	d73	
GSP 0-4	3		
GSP 0-3	3		
GSP 0-2	3		
GSP 0-1	3		
LPC2	68		1
GSP 0-4	4		
GSP 0-3	4		with
GSP 0-2	4		parity
GSP 0-1	4		check
LPC 1	69		
R0	2		
LPC 1	10		
R0	3,4		
Mode	0,1	d94	
Code 2-4	06	d95	
Code 1-4	06	-	2
Code 2-3	02	d111	

Table 3b: Subjective importance of encoded bits for the half rate speech TCH for voiced speech frames (Parameter names and bit indices refer to 3GPP TS 46.020)

Parameter	Bit index	Label	Class
name			
name   LPC 1   LPC 2   GSP 0-1   GSP 0-2   GSP 0-3   GSP 0-4   GSP 0-2   GSP 0-3   GSP 0-4   GSP 0-2   GSP 0-3   GSP 0-4   Code 2   Code 2   Code 2   Code 2   Code 2   Code 2   GSP 0-3   GSP 0-4   GSP 0-2   GSP 0-3   GSP 0-4   GSP 0-2   GSP 0-3   GSP 0-4   INT-LPC	2,1 64 4 4 4 4 3 3 3 3 3 2 2 2 2 80 8 4,3 1 1 1 0 0 0 0 0 0 0 1 1 1 1 24 2 5,6 3 0 7 0 3 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	d0, d1 d2	1 without parity check

Parameter name	Bit index	Label	Class
rafailletei flaille	Dit illuex	Labei	Class
LAG 3	3	d73	
LAG 2	3		
LAG 1	3,4		1
LPC 2	7,8		
LPC 1	36		with
R0	2		parity
LAG 1	57		check
LPC 1	710		
R0	3,4		
Mode	0,1	d94	
Code 4	08	d95	2
Code 3	07	d111	

Table 4: Reordering and partitioning of a coded block of 228 bits into 4 sub-blocks for TCH/HS

b=	0	1
j=0 2 4 6 8 10	k=0 38 76 114 152 190 18 56	150 188 226 14 52 90 128 166
20	94 132 170 208 8	204 32 70 108 146
30	46 84 122 160 198 28	184 222 10 48 86 124
40	66 104 142 180 218	162 200 30 68 106
50	4 42 80 118 156	144 182 220 6 44
60	194 22 60 98 136	82 120 158 196 24
70	174 212 12 50 88	62 100 138 176 214
80	126 164 202 34 72	2 40 78 116 154
90	110 148 186 224 16	192 26 64 102 140
100	54 92 130 168 206	178 216 20 58 96
110 112	36 74 112	134 172 210

b=	2	3
j=1 3 5 7 9 11	k=1 39 77 115 153 191	151 189 227 15 53 91 129
21	57 95 133 171 209 9	167 205 33 71 109 147 185
31	85 123 161 199	223 11 49 87 125
41	29 67 105 143 181 219	125 163 201 31 69 107
51	5 43 81 119	145 183 221 7
61	157 195 23 61 99	45 83 121 159 197
71	137 175 213 13 51	25 63 101 139 177
81	89 127 165 203 35	215 3 41 79 117
91	73 111 149 187 225	155 193 27 65 103
101	17 55 93 131 169 207	141 179 217 21 59 97
111 113	37 75 113	135 173 211

Table 5: Enhanced Full rate Source Encoder output parameters in order of occurrence and bit allocation within the speech frame of 244 bits/20 ms(Parameter names and bit indices refer to 3GPP TS 46.060)

Bits (MSB-LSB)	Description				
s1 - s7	index of 1st LSF submatrix				
s8 - s15	index of 2 <sup>nd</sup> LSF submatrix				
s16 - s23	index of 3 <sup>rd</sup> LSF submatrix				
s24	sign of 3 <sup>rd</sup> LSF submatrix				
s25 - s32	index of 4 <sup>th</sup> LSF submatrix				
s33 - s38	index of 5 <sup>th</sup> LSF submatrix				
subframe 1					
s39 - s47	adaptive codebook index				
s48 - s51	adaptive codebook gain				
s52	sign information for 1st and 6th pulses				
s53 - s55	position of 1 <sup>st</sup> pulse				
s56	sign information for 2 <sup>nd</sup> and 7 <sup>th</sup> pulses				
s57 - s59	position of 2 <sup>nd</sup> pulse				
s60	sign information for 3 <sup>rd</sup> and 8 <sup>th</sup> pulses				
s61 - s63	position of 3 <sup>rd</sup> pulse				
s64	sign information for 4 <sup>th</sup> and 9 <sup>th</sup> pulses				
s65 - s67	position of 4 <sup>th</sup> pulse				
s68	sign information for 5th and 10th pulses				
s69 - s71	position of 5 <sup>th</sup> pulse				
s72 - s74	position of 6 <sup>th</sup> pulse				
s75 - s77	position of 7 <sup>th</sup> pulse				
s78 - s80	position of 8 <sup>th</sup> pulse				
s81 - s83	position of 9 <sup>th</sup> pulse				
s84 - s86	position of 10 <sup>th</sup> pulse				
s87 - s91	fixed codebook gain				
subframe 2					
s92 - s97	adaptive codebook index (relative)				
s98 - s141	same description as s48 - s91				
	subframe 3				
s142 - s194	same description as s39 - s91				
	subframe 4				
s195 - s244	same description as s92 - s141				

Table 6: Ordering of enhanced full rate speech parameters for the channel encoder (subjective importance of encoded bits) (after preliminary channel coding)

(Parameter names refers to 3GPP TS 46.060)

Description	Bits (Table 5)	Bit index within parameter
CLASS 1a: 50 bits (protected by 3 bit TCH-FS CRC)		
LTP-LAG 1	w39 - w44	b8, b7, b6, b5, b4, b3
LTP-LAG 3	w146 - w151	b8, b7, b6, b5, b4, b3
LTP-LAG 2	w94 - w95	b5, b4
LTP-LAG 4	w201 - w202	b5, b4
LTP-GAIN 1	n48	b3
FCB-GAIN 1	w89	b4
LTP-GAIN 2	w100	b3
FCB-GAIN 2	w141	b4
LTP-LAG 1	w45	b2
LTP-LAG 3	w152	b2
LTP-LAG 2	w96	b3
LTP-LAG 4	w203	b3
LPC 1	w2 - w3	b5, b4
LPC 2	w8	b7
LPC 2	w10	b5
LPC 3	w18 - w19	b6, b5
LPC 3	w24	b0
LTP-LAG 1	w46 - w47	b1, b0
LTP-LAG 3	w153 - w154	b1, b0
LTP-LAG 2	w133 W134	b2
LTP-LAG 4	w204	b2
LPC 1	w4 - w5	b3, b2
LPC 2	w11 - w12	b4, b3
LPC 3	w16	b8
LPC 2	w9	b6
LPC 1	w6 - w7	b1, b0
LPC 2	w3 w7	b2
LPC 3	w17	b7
LPC 3	w20	b/
LTP-LAG 2	w98	b1
LTP-LAG 2	w205	b1
CLASS 1b: 132 bits (protected)	WZOS	D1
LPC 1	w1	b6
LPC 2	w14 - w15	b1, b0
LPC 3	w14 - w15 w21	b1, b0
LPC 3 LPC 4	w25 - w26	b7, b6
LPC 4 LPC 4		b7, b6
LPC 4 LTP-GAIN 3	w28	
LTP-GAIN 3 LTP-GAIN 4	w155 w207	<u>b3</u> b3
FCB-GAIN 3		b3 
	w196	
FCB-GAIN 4	w248	b4
FCB-GAIN 1	w90	b3
FCB-GAIN 2	w142	b3
FCB-GAIN 3	w197	<u>b3</u>
FCB-GAIN 4	w249	b3
CRC-POLY	w253 - w260	b7, b6, b5, b4, b3, b2, b1, b0
LTP-GAIN 1	w49	b2

Table 6 (continued): Ordering of enhanced full rate speech parameters for the channel encoder (subjective importance of encoded bits) (after preliminary channel coding)

(Parameter names refers to 3GPP TS 46.060)

Description	Bits (Table 5)	Bit index within parameter
LTP-GAIN 2	w101	b2
LTP-GAIN 3	w156	b2
LTP-GAIN 4	w208	b2
LPC 3	w22 - w23	b2, b1
LPC 4	w27	b5
LPC 4	w29	b3
PULSE 1_1	w52	b3
PULSE 1_2	w56	b3
PULSE 1_3	w60	b3
PULSE 1_4	w64	b3
PULSE 1_5	w68	b3
PULSE 2_1	w104	b3
PULSE 2_2	w108	b3
PULSE 2_3	w112	b3
PULSE 2_4	w116	b3
PULSE 2_5	w120	b3
PULSE 3_1	w159	b3
PULSE 3_2	w163	b3
PULSE 3_3	w167	b3
PULSE 3_4	w171	b3
PULSE 3_5	w175	b3
PULSE 4_1	w211	b3
PULSE 4_2	w215	b3
PULSE 4_3	w219	b3
PULSE 4_4	w223	b3
PULSE 4_5	w227	b3
FCB-GAIN 1	w91	b2
FCB-GAIN 2	w143	b2
FCB-GAIN 3	w198	b2
FCB-GAIN 4	w250	b2
LTP-GAIN 1	w50	b1
LTP-GAIN 2	w102	b1
LTP-GAIN 3	w157	b1
LTP-GAIN 4	w209	b1
LPC 4	w30 - w32	b2, b1, b0
LPC 5	w33 - w36	b5, b4, b3, b2
LTP-LAG 2	w99	b0
LTP-LAG 4	w206	b0
PULSE 1_1	w53	b2
PULSE 1_2	w57	b2

Table 6 (continued): Ordering of enhanced full rate speech parameters for the channel encoder (subjective importance of encoded bits) (after preliminary channel coding)

(Parameter names refers to 3GPP TS 46.060)

Description	Bits (Table 5)	Bit index within parameter
PULSE 1_3	w61	b2
PULSE 1_4	w65	b2
PULSE 1_5	w69	b2
PULSE 2_1	w105	b2
PULSE 2_2	w109	b2
PULSE 2_3	w113	b2
PULSE 2_4	w117	b2
PULSE 2_5	w121	b2
PULSE 3_1	w160	b2
PULSE 3_2	w164	b2
PULSE 3_3	w168	b2
PULSE 3_4	w172	b2
PULSE 3_5	w176	b2
PULSE 4_1	w212	<u>b2</u>
PULSE 4_2	w216	b2
PULSE 4_3	w220	b2
PULSE 4_4	w224	b2
PULSE 4_5	w228	b2
PULSE 1_1	w54	b1
PULSE 1_2	w58	<u>b1</u>
PULSE 1_3	w62	b1
PULSE 1_4	w66	b1
PULSE 2_1	w106	b1
PULSE 2_2 PULSE 2_3	w110 w114	<u>b1</u> b1
PULSE 2_3 PULSE 2_4	w114 w118	b1
PULSE 2_4 PULSE 3_1	w116	b1
PULSE 3_1	w165	b1
PULSE 3_3	w169	b1
PULSE 3_4	w173	b1
PULSE 4_1	w213	b1
PULSE 4_3	w221	b1
PULSE 4_4	w225	b1
FCB-GAIN 1	w92	b1
FCB-GAIN 2	w144	b1
FCB-GAIN 3	s199	b1
FCB-GAIN 4	w251	b1
LTP-GAIN 1	w51	b0
LTP-GAIN 2	w103	b0
LTP-GAIN 3	w158	b0
LTP-GAIN 4	w210	b0
FCB-GAIN 1	w93	b0
FCB-GAIN 2	w145	b0
FCB-GAIN 3	w200	b0
FCB-GAIN 4	w252	b0
PULSE 1_1	w55	b0
PULSE 1_2	w59	b0
PULSE 1_3	w63	b0
PULSE 1_4	w67	b0
PULSE 2_1	w107	b0
PULSE 2_2	w111	b0
PULSE 2_3	w115	b0
PULSE 2_4	w119	b0
PULSE 3_1	w162	b0
PULSE 3_2	w166	b0
PULSE 3_3	w170	b0

Table 6 (continued): Ordering of enhanced full rate speech parameters for the channel encoder (subjective importance of encoded bits) (after preliminary channel coding)

(Parameter names refers to 3GPP TS 46.060)

Description	Bits (Table 5)	Bit index within parameter
PULSE 3_4	w174	b0
PULSE 4_1	w214	b0
PULSE 4_3	w222	b0
PULSE 4_4	w226	b0
LPC 5	w37 - w38	b1, b0
CLASS 2: 78 bits (unprotected)		
PULSE 1_5	w70	b1
PULSE 1_5	w72 - w73	b1, b1
PULSE 2_5	w122	b1
PULSE 2_5	w124 - s125	b1, b1
PULSE 3_5	w177	b1
PULSE 3_5	w179 - w180	b1, b1
PULSE 4_5	w229	<u>b1</u>
PULSE 4_5	w231 - w232	b1, b1
PULSE 4_2	w217 - w218	b1, b0
PULSE 1_5	w71	<u>b0</u>
PULSE 2_5	w123	b0
PULSE 3_5	w178	b0
PULSE 4_5	w230	<u>b0</u>
PULSE 1_6	w74	b2
PULSE 1_7	w77	b2
PULSE 1_8	w80	b2
PULSE 1_9	w83	b2
PULSE 1_10	w86	b2
PULSE 2_6	w126	b2
PULSE 2_7	w129	b2
PULSE 2_8	w132	b2
PULSE 2_9	w135	b2
PULSE 2_10	w138 w181	b2 b2
PULSE 3_6		
PULSE 3_7	w184 w187	b2 b2
PULSE 3_8 PULSE 3_9	w190	b2
PULSE 3_9 PULSE 3_10	w190 w193	b2
PULSE 4_6	w233	b2
PULSE 4_7	w236	b2
PULSE 4 8	w239	b2
PULSE 4_9	w242	b2
PULSE 4_10	w245	b2
PULSE 1_6	w75	b1
PULSE 1_7	w78	b1
PULSE 1_8	w81	b1
PULSE 1_9	w84	b1
PULSE 1_10	w87	b1
PULSE 2_6	w127	b1
PULSE 2_7	w130	b1
PULSE 2_8	w133	b1
PULSE 2_9	w136	b1
PULSE 2_10	w139	b1
PULSE 3_6	w182	b1
PULSE 3_7	w185	b1
PULSE 3_8	w188	b1
PULSE 3_9	w191	b1
PULSE 3_10	w194	b1
		b1
		b1
PULSE 4_6 PULSE 4_7	w234 w237 continued)	b1

Table 6 (concluded): Ordering of enhanced full rate speech parameters for the channel encoder (subjective importance of encoded bits) (after preliminary channel coding)

(Parameter names refers to 3GPP TS 46.060)

Description	Bits (Table 5)	Bit index within parameter
PULSE 4_8	w240	b1
PULSE 4_9	w243	b1
PULSE 4_10	w246	b1
PULSE 1_6	w76	b0
PULSE 1_7	w79	b0
PULSE 1_8	w82	b0
PULSE 1_9	w85	b0
PULSE 1_10	w88	b0
PULSE 2_6	w128	b0
PULSE 2_7	w131	b0
PULSE 2_8	w134	b0
PULSE 2_9	w137	b0
PULSE 2_10	w140	b0
PULSE 3_6	w183	b0
PULSE 3_7	w186	b0
PULSE 3_8	w189	b0
PULSE 3_9	w192	b0
PULSE 3_10	w195	b0
PULSE 4_6	w235	b0
PULSE 4_7	w238	b0
PULSE 4_8	w241	b0
PULSE 4_9	w244	b0
PULSE 4_10	w247	b0

Table 7: Sorting of the speech encoded bits for TCH/AFS12.2

0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	23	15	16	17	18
19	20	21	22	24	25	26	27	28	38
141	39	142	40	143	41	144	42	145	43
146	44	147	45	148	46	149	47	97	150
200	48	98	151	201	49	99	152	202	86
136	189	239	87	137	190	240	88	138	191
241	91	194	92	195	93	196	94	197	95
198	29	30	31	32	33	34	35	50	100
153	203	89	139	192	242	51	101	154	204
55	105	158	208	90	140	193	243	59	109
162	212	63	113	166	216	67	117	170	220
36	37	54	53	52	58	57	56	62	61
60	66	65	64	70	69	68	104	103	102
108	107	106	112	111	110	116	115	114	120
119	118	157	156	155	161	160	159	165	164
163	169	168	167	173	172	171	207	206	205
211	210	209	215	214	213	219	218	217	223
222	221	73	72	71	76	75	74	79	78
77	82	81	80	85	84	83	123	122	121
126	125	124	129	128	127	132	131	130	135
134	133	176	175	174	179	178	177	182	181
180	185	184	183	188	187	186	226	225	224
229	228	227	232	231	230	235	234	233	238
237	236	96	199						

Table 8: Sorting of the speech encoded bits for TCH/AFS10.2

7	6	5	4	3	2	1	0	16	15
14	13	12	11	10	9	8	26	27	28
29	30	31	115	116	117	118	119	120	72
73	161	162	65	68	69	108	111	112	154
157	158	197	200	201	32	33	121	122	74
75	163	164	66	109	155	198	19	23	21
22	18	17	20	24	25	37	36	35	34
80	79	78	77	126	125	124	123	169	168
167	166	70	67	71	113	110	114	159	156
160	202	199	203	76	165	81	82	92	91
93	83	95	85	84	94	101	102	96	104
86	103	87	97	127	128	138	137	139	129
141	131	130	140	147	148	142	150	132	149
133	143	170	171	181	180	182	172	184	174
173	183	190	191	185	193	175	192	176	186
38	39	49	48	50	40	52	42	41	51
58	59	53	61	43	60	44	54	194	179
189	196	177	195	178	187	188	151	136	146
153	134	152	135	144	145	105	90	100	107
88	106	89	98	99	62	47	57	64	45
63	46	55	56						

Table 9: Sorting of the speech encoded bits for TCH/AFS7.95 and TCH/AHS7.95

8	7	6	5	4	3	2	14	16	9
10	12	13	15	11	17	20	22	24	23
19	18	21	56	88	122	154	57	89	123
155	58	90	124	156	52	84	118	150	53
85	119	151	27	93	28	94	29	95	30
96	31	97	61	127	62	128	63	129	59
91	125	157	32	98	64	130	1	0	25
26	33	99	34	100	65	131	66	132	54
86	120	152	60	92	126	158	55	87	121
153	117	116	115	46	78	112	144	43	75
109	141	40	72	106	138	36	68	102	134
114	149	148	147	146	83	82	81	80	51
50	49	48	47	45	44	42	39	35	79
77	76	74	71	67	113	111	110	108	105
101	145	143	142	140	137	133	41	73	107
139	37	69	103	135	38	70	104	136	

Table 10: Sorting of the speech encoded bits for TCH/AFS7.4 and TCH/AHS7.4

0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	26	87	27
88	28	89	29	90	30	91	51	80	112
141	52	81	113	142	54	83	115	144	55
84	116	145	58	119	59	120	21	22	23
17	18	19	31	60	92	121	56	85	117
146	20	24	25	50	79	111	140	57	86
118	147	49	78	110	139	48	77	53	82
114	143	109	138	47	76	108	137	32	33
61	62	93	94	122	123	41	42	43	44
45	46	70	71	72	73	74	75	102	103
104	105	106	107	131	132	133	134	135	136
34	63	95	124	35	64	96	125	36	65
97	126	37	66	98	127	38	67	99	128
39	68	100	129	40	69	101	130		

Table 11: Sorting of the speech encoded bits for TCH/AFS6.7 and TCH/AHS6.7

0	1	4	3	5	6	13	7	2	8
9	11	15	12	14	10	28	82	29	83
27	81	26	80	30	84	16	55	109	56
110	31	85	57	111	48	73	102	127	32
86	51	76	105	130	52	77	106	131	58
112	33	87	19	23	53	78	107	132	21
22	18	17	20	24	25	50	75	104	129
47	72	101	126	54	79	108	133	46	71
100	125	128	103	74	49	45	70	99	124
42	67	96	121	39	64	93	118	38	63
92	117	35	60	89	114	34	59	88	113
44	69	98	123	43	68	97	122	41	66
95	120	40	65	94	119	37	62	91	116
36	61	90	115						

Table 12: Sorting of the speech encoded bits for TCH/AFS5.9 and TCH/AHS5.9

0	1	4	5	3	6	7	2	13	15
8	9	11	12	14	10	16	28	74	29
75	27	73	26	72	30	76	51	97	50
71	96	117	31	77	52	98	49	70	95
116	53	99	32	78	33	79	48	69	94
115	47	68	93	114	46	67	92	113	19
21	23	22	18	17	20	24	111	43	89
110	64	65	44	90	25	45	66	91	112
54	100	40	61	86	107	39	60	85	106
36	57	82	103	35	56	81	102	34	55
80	101	42	63	88	109	41	62	87	108
38	59	84	105	37	58	83	104		

Table 13: Sorting of the speech encoded bits for TCH/AFS5.15 and TCH/AHS5.15

7	6	5	4	3	2	1	0	15	14
13	12	11	10	9	8	23	24	25	26
27	46	65	84	45	44	43	64	63	62
83	82	81	102	101	100	42	61	80	99
28	47	66	85	18	41	60	79	98	29
48	67	17	20	22	40	59	78	97	21
30	49	68	86	19	16	87	39	38	58
57	77	35	54	73	92	76	96	95	36
55	74	93	32	51	33	52	70	71	89
90	31	50	69	88	37	56	75	94	34
53	72	91							

Table 14: Sorting of the speech encoded bits for TCH/AFS4.75 and TCH/AHS4.75

0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	23	24	25	26
27	28	48	49	61	62	82	83	47	46
45	44	81	80	79	78	17	18	20	22
77	76	75	74	29	30	43	42	41	40
38	39	16	19	21	50	51	59	60	63
64	72	73	84	85	93	94	32	33	35
36	53	54	56	57	66	67	69	70	87
88	90	91	34	55	68	89	37	58	71
92	31	52	65	86					

Table 15: Interleaving table for MCS5 and MCS6:

m\n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	463	890	1038	220	371	795	946	582	733	1160	63	490	641	277	428
1	852	1003	185	333	1223	120	547	698	1122	28	915	1066	242	390	817	968
2	610	761	1185	85	512	660	305	453	880	1031	204	355	782	1242	148	575
3	723	1150	50	474	625	1088	267	418	845	993	169	320	1207	113	537	688
4	1115	12	902	1050	232	383	807	958	594	745	1172	75	502	653	289	440
5	864	1015	197	345	1235	132	559	710	1134	40	927	1078	254	402	829	980
6	159	622	773	1197	97	524	672	1099	5	465	892	1043	216	367	794	942
7	587	735	1162	62	486	637	279	430	857	1005	181	332	1219	125	549	700
8	1127	24	914	1062	244	395	819	970	606	757	1184	87	514	665	301	452
9	876	1027	209	357	784	1247	144	571	722	1146	52	479	627	1090	266	414
10	841	992	171	322	1209	109	536	684	1111	17	904	1055	228	379	806	954
11	599	747	1174	74	498	649	291	442	869	1017	193	344	1231	137	561	712
12	1139	36	926	1074	256	407	831	982	158	618	769	1196	99	526	677	1101
13	7	458	894	1033	227	363	802	941	577	740	1152	70	485	645	284	420
14	859	998	189	328	1215	127	542	702	1117	35	922	1061	246	385	824	960
15	605	765	1180	92	504	667	309	448	887	1023	211	350	786	1237	155	567
16	730	1145	54	469	632	1080	274	413	849	988	176	312	1202	117	532	695
17	1107	19	906	1045	239	375	814	953	589	752	1164	82	497	657	296	432
18	871	1010	201	340	1227	139	554	714	1129	47	934	1073	258	397	836	972
19	166	617	777	1192	104	516	679	1094	9	460	899	1035	223	362	798	937
20	579	742	1157	66	481	644	286	425	861	1000	188	324	1214	129	544	707
21	1119	31	918	1057	251	387	826	965	601	764	1176	94	509	669	308	444
22	883	1022	213	352	791	1239	151	566	726	1141	59	471	634	1085	270	409
23	848	984	178	317	1204	116	528	691	1106	21	911	1047	235	374	810	949
24	591	754	1169	78	493	656	298	437	873	1012	200	336	1226	141	556	719
25	1131	43	930	1069	263	399	838	977	162	613	776	1188	106	521	681	1096
26	2	462	889	1040	219	370	797	945	584	732	1159	65	489	640	276	427
27	854	1002	184	335	1222	122	546	697	1124	27	917	1065	241	392	816	967
28	609	760	1187	84	511	662	304	455	879	1030	206	354	781	1244	147	574
29	725	1149	49	476	624	1087	269	417	844	995	168	319	1206	112	539	687
30	1114	14	901	1052	231	382	809	957	596	744	1171	77	501	652	288	439
31	866	1014	196	347	1234	134	558	709	1136	39	929	1077	253	404	828	979
32	161	621	772	1199	96	523	674	1098	4	467	891	1042	218	366	793	944
33	586	737	1161	61	488	636	281	429	856	1007	180	331	1218	124	551	699
34	1126	26	913	1064	243	394	821	969	608	756	1183	89	513	664	300	451
35	878	1026	208	359	783	1246	146	570	721	1148	51	478	629	1089	265	416
36																
	840	991	173	321	1211	108	535	686	1110	16	903	1054	230	378	805	956
37	840 598	991 749	173 1173	321 73	1211 500	108 648	535 293	686 441	1110 868	16 1019	903 192	1054 343	230 1230	378 136	805 563	956 711
	598	749	1173	321 73 1076	500	108 648 406	293			1019		1054 343 1195		136	805 563 676	711
37				73		648		441	868		192	343	1230		563	
37 38	598 1138	749 38	1173 925	73 1076	500 255	648 406	293 833	441 981	868 157	1019 620	192 768	343 1195	1230 101	136 525	563 676	711 1103
37 38 39	598 1138 6	749 38 457	1173 925 896 191	73 1076 1032	500 255 226 1217	648 406 365	293 833 801	981 940	868 157 576	1019 620 739	192 768 1154 921	343 1195 69 1060	1230 101 484	136 525 647 384	563 676 283	711 1103 422 962
37 38 39 40	598 1138 6 858	749 38 457 997	1173 925 896	73 1076 1032 327	500 255 226	648 406 365 126	293 833 801 541	981 940 704	868 157 576 1116	1019 620 739 34	192 768 1154	343 1195 69	1230 101 484 248	136 525 647	563 676 283 823	711 1103 422
37 38 39 40 41	598 1138 6 858 604 729	749 38 457 997 767	1173 925 896 191 1179 56	73 1076 1032 327 91 468	500 255 226 1217 506 631	648 406 365 126 666	293 833 801 541 311 273	981 940 704 447 412	868 157 576 1116 886 851	1019 620 739 34 1025 987	192 768 1154 921 210 175	343 1195 69 1060 349 314	1230 101 484 248 788 1201	136 525 647 384 1236 119	563 676 283 823 154 531	711 1103 422 962 569 694
37 38 39 40 41 42	598 1138 6 858 604	749 38 457 997 767 1144	1173 925 896 191 1179	73 1076 1032 327 91	500 255 226 1217 506	648 406 365 126 666 1082	293 833 801 541 311	981 940 704 447	868 157 576 1116 886	1019 620 739 34 1025	192 768 1154 921 210	343 1195 69 1060 349	1230 101 484 248 788	136 525 647 384 1236	563 676 283 823 154	711 1103 422 962 569
37 38 39 40 41 42 43	598 1138 6 858 604 729 1109	749 38 457 997 767 1144 18	1173 925 896 191 1179 56 908	73 1076 1032 327 91 468 1044	500 255 226 1217 506 631 238	648 406 365 126 666 1082 377	293 833 801 541 311 273 813	441 981 940 704 447 412 952	868 157 576 1116 886 851 588	1019 620 739 34 1025 987 751	192 768 1154 921 210 175 1166	343 1195 69 1060 349 314 81	1230 101 484 248 788 1201 496	136 525 647 384 1236 119 659	563 676 283 823 154 531 295	711 1103 422 962 569 694 434
37 38 39 40 41 42 43 44	598 1138 6 858 604 729 1109 870	749 38 457 997 767 1144 18 1009	1173 925 896 191 1179 56 908 203	73 1076 1032 327 91 468 1044 339	500 255 226 1217 506 631 238 1229	648 406 365 126 666 1082 377 138	293 833 801 541 311 273 813 553	441 981 940 704 447 412 952 716	868 157 576 1116 886 851 588 1128	1019 620 739 34 1025 987 751 46	192 768 1154 921 210 175 1166 933	343 1195 69 1060 349 314 81 1072	1230 101 484 248 788 1201 496 260	136 525 647 384 1236 119 659 396	563 676 283 823 154 531 295 835	711 1103 422 962 569 694 434 974
37 38 39 40 41 42 43 44 45	598 1138 6 858 604 729 1109 870 165	749 38 457 997 767 1144 18 1009 616	1173 925 896 191 1179 56 908 203 779	73 1076 1032 327 91 468 1044 339 1191	500 255 226 1217 506 631 238 1229 103	648 406 365 126 666 1082 377 138 518	293 833 801 541 311 273 813 553 678	981 940 704 447 412 952 716 1093	868 157 576 1116 886 851 588 1128	1019 620 739 34 1025 987 751 46 459	192 768 1154 921 210 175 1166 933 898	343 1195 69 1060 349 314 81 1072 1037	1230 101 484 248 788 1201 496 260 222	136 525 647 384 1236 119 659 396 361	563 676 283 823 154 531 295 835 800	711 1103 422 962 569 694 434 974 936
37 38 39 40 41 42 43 44 45 46	598 1138 6 858 604 729 1109 870 165 581	749 38 457 997 767 1144 18 1009 616 741	1173 925 896 191 1179 56 908 203 779 1156	73 1076 1032 327 91 468 1044 339 1191 68	500 255 226 1217 506 631 238 1229 103 480	648 406 365 126 666 1082 377 138 518 643	293 833 801 541 311 273 813 553 678 285	441 981 940 704 447 412 952 716 1093 424	868 157 576 1116 886 851 588 1128 11	1019 620 739 34 1025 987 751 46 459 999	192 768 1154 921 210 175 1166 933 898 187	343 1195 69 1060 349 314 81 1072 1037 326	1230 101 484 248 788 1201 496 260 222 1213	136 525 647 384 1236 119 659 396 361 131	563 676 283 823 154 531 295 835 800 543	711 1103 422 962 569 694 434 974 936 706
37 38 39 40 41 42 43 44 45 46 47	598 1138 6 858 604 729 1109 870 165 581 1121	749 38 457 997 767 1144 18 1009 616 741 30	1173 925 896 191 1179 56 908 203 779 1156 920	73 1076 1032 327 91 468 1044 339 1191 68 1056	500 255 226 1217 506 631 238 1229 103 480 250	648 406 365 126 666 1082 377 138 518 643 389	293 833 801 541 311 273 813 553 678 285 825	441 981 940 704 447 412 952 716 1093 424 964	868 157 576 1116 886 851 588 1128 11 863 600	1019 620 739 34 1025 987 751 46 459 999 763	192 768 1154 921 210 175 1166 933 898 187 1178	343 1195 69 1060 349 314 81 1072 1037 326 93	1230 101 484 248 788 1201 496 260 222 1213 508	136 525 647 384 1236 119 659 396 361 131 671	563 676 283 823 154 531 295 835 800 543 307	711 1103 422 962 569 694 434 974 936 706 446
37 38 39 40 41 42 43 44 45 46 47 48	598 1138 6 858 604 729 1109 870 165 581 1121 882	749 38 457 997 767 1144 18 1009 616 741 30 1021	1173 925 896 191 1179 56 908 203 779 1156 920 215	73 1076 1032 327 91 468 1044 339 1191 68 1056 351	500 255 226 1217 506 631 238 1229 103 480 250 790	648 406 365 126 666 1082 377 138 518 643 389 1241	293 833 801 541 311 273 813 553 678 285 825 150	981 940 704 447 412 952 716 1093 424 964 565	868 157 576 1116 886 851 588 1128 11 863 600 728	1019 620 739 34 1025 987 751 46 459 999 763 1140	192 768 1154 921 210 175 1166 933 898 187 1178 58	343 1195 69 1060 349 314 81 1072 1037 326 93 473	1230 101 484 248 788 1201 496 260 222 1213 508 633	136 525 647 384 1236 119 659 396 361 131 671 1084	563 676 283 823 154 531 295 835 800 543 307 272	711 1103 422 962 569 694 434 974 936 706 446 408
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51	598 1138 6 858 604 729 1109 870 165 581 1121 882	749 38 457 997 767 1144 18 1009 616 741 30 1021 986	1173 925 896 191 1179 56 908 203 779 1156 920 215	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316	500 255 226 1217 506 631 238 1229 103 480 250 790 1203	648 406 365 126 666 1082 377 138 518 643 389 1241 115	293 833 801 541 311 273 813 553 678 285 825 150	941 981 940 704 447 412 952 716 1093 424 964 565 690	868 157 576 1116 886 851 588 1128 11 863 600 728 1105	1019 620 739 34 1025 987 751 46 459 999 763 1140 23	192 768 1154 921 210 175 1166 933 898 187 1178 58	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049	1230 101 484 248 788 1201 496 260 222 1213 508 633 234	136 525 647 384 1236 119 659 396 361 131 671 1084 373	563 676 283 823 154 531 295 835 800 543 307 272 812	711 1103 422 962 569 694 434 974 936 706 446 408
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796	941 981 940 704 447 412 952 716 1093 424 964 565 690 436	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875	1019 620 739 34 1025 987 751 46 459 999 763 1140 23	192 768 1154 921 210 175 1166 933 898 187 1178 58 910	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639	563 676 283 823 154 531 295 835 800 543 307 272 812	711 1103 422 962 569 694 434 974 936 706 446 408 948 718
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888 183	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1221	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796 548	941 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391	563 676 283 823 154 531 295 835 800 543 307 272 812 555 683 278 818	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796	941 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875 164 583	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639	563 676 283 823 154 531 295 835 800 543 307 272 812 555 683 278	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611 724	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004 759	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888 183	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334 86	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1221 510 626	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121 661 1086	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796 548	941 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696 454 419	868 157 576 11116 886 851 588 1128 11 863 600 728 1105 875 164 583 1123 881	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734 29 1029	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916 205 170	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64 1067 356 318	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240 780 1208	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391 1243 111	563 676 283 823 154 531 295 835 800 543 307 272 812 555 683 278 818 149	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426 966 573 689
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611 724 1113	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004 759	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888 183 1186	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334 86 475	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1221 510 626 233	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121 661 1086 381	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796 548 303 268 808	941 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696 454 419 959	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875 164 583 1123 881	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734 29 1029 994 746	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916 205 170 1170	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64 1067 356 318 76	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240 780 1208 503	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391 1243 111 651	563 676 283 823 154 531 295 835 800 543 307 272 812 555 683 278 818	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426 966 573 689
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611 724 1113 865	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004 759 1151 13 1016	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888 183 1186 48	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334 86	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1221 510 626 233 1233	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121 661 1086 381 133	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796 548 303 268 808	941 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696 454 419 959 708	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875 164 583 1123 881 843 595 1135	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734 29 1029	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916 205 170 1170 928	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64 1067 356 318	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240 780 1208 503 252	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391 1243 111	563 676 283 823 154 531 295 835 800 543 307 272 812 555 683 278 818 149 538 290 830	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426 966 573 689 438
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611 724 1113 865 160	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004 759 1151 13 1016 623	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888 183 1186 48 900 195 771	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334 86 475	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1221 510 626 233 1233 98	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121 661 1086 381 133 522	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796 548 303 268 808	941 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696 454 419 959	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875 164 583 1123 881 843 595	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734 29 1029 994 746	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916 205 170 1170 928 893	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64 1067 356 318 76 1079	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240 780 1208 503	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391 1243 111 651 403 368	563 676 283 823 154 531 295 835 800 543 307 272 812 555 683 278 818 149 538 290 830 792	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426 966 573 689 438 978
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611 724 1113 865 160 585	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004 759 1151 13 1016 623 736	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888 183 1186 48 900 195 771 1163	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334 86 475 1051 346 1198	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1221 510 626 233 1233 98 487	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121 661 1086 381 133 522 638	293 833 801 541 311 273 813 553 678 285 825 150 530 297 796 548 303 268 808 560 673 280	441 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696 454 419 959 708 1100 431	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875 164 583 1123 881 843 595 1135 3 855	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734 29 1029 994 746 41 466 1006	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916 205 170 1170 928 893 182	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64 1067 356 318 76 1079 1041 330	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240 780 1208 503 252 217 1220	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391 1243 111 651 403 368 123	563 676 283 823 154 531 295 835 800 543 307 272 812 555 683 278 818 149 538 290 830 792	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426 966 573 689 438 978 943 701
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611 724 1113 865 160 585 1125	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004 759 1151 13 1016 623 736 25	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888 183 1186 48 900 195 771 1163 912	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334 86 475 1051 346 1198 60	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1221 510 626 233 1233 98 487 245	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121 661 1086 381 133 522 638 393	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796 548 303 268 808 560 673 280	941 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696 454 419 959 708 1100 431 971	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875 164 583 1123 881 843 595 1135 3 855 607	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734 29 1029 994 746 41 466 1006 758	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916 205 170 1170 928 893 182 1182	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64 1067 356 318 76 1079 1041 330 88	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240 780 1208 503 252 217 1220 515	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391 1243 111 651 403 368 123 663	563 676 283 823 154 531 295 835 800 543 307 272 812 555 683 278 818 149 538 290 830 792 550 302	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426 573 689 438 978 943 701
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611 724 1113 865 160 585	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004 759 1151 13 1016 623 736	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 938 888 183 1186 48 900 195 771 1163 912 207	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334 86 475 1051 346 1198 60 1063 358	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1221 510 626 233 1233 98 487 245 785	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121 661 1086 381 133 522 638	293 833 801 541 311 273 813 553 678 285 825 150 530 297 796 548 303 268 808 560 673 280	441 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696 454 419 959 708 1100 431	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875 164 583 1123 881 843 595 1135 3 855	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734 29 1029 994 746 41 466 1006 758 1147	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916 205 170 1170 928 893 182 1182 53	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64 1067 356 318 76 1079 1041 330 88 477	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240 780 1208 503 252 217 1220 515 628	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391 1243 111 651 403 368 123	563 676 283 823 154 531 295 835 800 543 307 272 812 555 683 278 818 149 538 290 830 792	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426 573 689 438 978 943 701 450 415
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611 724 1113 865 160 585 1125 877	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004 759 1151 13 1016 623 736 25 1028 990	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888 183 1186 48 900 195 771 1163 912 207 172	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334 475 1051 346 1198 60 1063 358 323	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1221 510 626 233 1233 98 487 245 785 1210	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121 661 1086 381 133 522 638 393 1245 110	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796 548 303 268 808 560 673 280 820 145	941 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696 454 419 959 708 1100 431 971 572 685	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875 164 583 1123 881 843 595 1135 3 855 607 720 1112	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734 29 1029 994 746 41 466 1006 758 1147	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916 205 170 1170 928 893 182 1182 53 905	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64 1067 356 318 76 1079 1041 330 88 477 1053	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240 780 1208 503 252 217 1220 515 628 229	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391 1243 111 651 403 368 123 663 1091 380	563 676 283 823 154 531 295 835 800 543 307 272 812 555 683 278 818 149 538 290 830 792 550 302 264	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426 957 689 438 978 943 701 450 415
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611 724 1113 865 160 585 1125 877 842 597	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004 759 1151 13 1016 623 736 25 1028 990 748	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888 183 1186 48 900 195 771 1163 912 207 172 1175	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334 475 1051 346 1198 60 1063 358 323 72	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1221 510 626 233 1233 98 487 245 785 1210 499	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121 661 1086 381 133 522 638 393 1245 110 650	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796 548 303 268 808 560 673 280 820 145 534	941 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696 4419 959 708 1100 431 971 572 685 443	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875 164 583 1123 881 843 595 1135 3 855 607 720 1112 867	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734 29 1029 994 746 41 466 1006 758 1147 15	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916 205 170 1170 928 893 182 1182 53 905 194	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64 1067 356 318 76 1079 1041 330 88 477 1053 342	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240 780 1208 503 252 217 1220 515 628 229 1232	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391 1243 111 651 403 368 123 663 1091 380 135	563 676 283 823 154 531 295 835 800 543 307 272 812 558 683 278 818 149 538 290 830 792 550 302 264 804	711 1103 422 962 569 694 434 974 936 706 446 408 948 1095 426 567 689 438 978 943 701 450 415 955 713
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611 724 1113 865 160 585 1125 877 842 597 1137	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004 759 1151 13 1016 623 736 25 1028 990 748 37	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888 183 1186 48 900 195 771 1163 912 207 172 1175 924	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334 86 475 1051 346 1198 60 1063 358 323 72 1075	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1221 510 626 233 1233 98 487 245 785 1210 499 257	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121 661 1086 381 133 522 638 393 1245 110 650 405	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796 548 303 268 808 560 673 280 820 145 534	941 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696 459 419 959 708 1100 431 971 572 685 443 983	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875 164 583 1123 881 123 881 595 1135 3 855 607 720 1112 867 156	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734 29 1029 994 746 41 466 1006 758 1147 15 1018 619	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916 205 170 1170 928 893 182 1182 53 905 194 770	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64 1067 356 318 76 1079 1041 330 88 477 1053 342 1194	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240 780 1208 503 252 217 1220 515 628 229 1232 100	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391 1243 111 651 403 368 123 663 1091 380 135 527	563 676 283 823 154 531 295 835 800 543 307 272 812 555 683 278 818 149 538 290 830 792 550 302 264 804 562 675	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426 966 573 689 438 978 943 701 450 415 955 713 1102
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611 724 1113 865 160 585 1125 877 842 597 1137 8	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004 759 1151 13 1016 623 736 25 1028 990 748	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888 183 1186 48 900 195 771 1163 912 207 172 1175	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334 86 475 1051 346 1198 60 1063 358 323 72 1075 1034	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1221 510 626 233 1233 98 487 245 785 1210 499 257 225	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121 661 1086 381 133 522 638 393 1245 110 650 405 364	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796 548 303 268 808 560 673 280 820 145 534	941 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696 459 419 959 708 1100 431 971 572 685 443 983 939	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875 164 583 1123 881 843 595 1135 3 855 607 720 1112 867 156 578	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734 29 1029 994 746 41 466 1006 758 1147 15 1018 619 738	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916 205 170 1170 928 893 182 1182 53 905 194 770 1153	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64 1067 356 318 76 1079 1041 330 88 477 1053 342 1194 71	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240 780 1208 503 252 217 1220 515 628 229 1232 100 483	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391 1243 111 651 403 368 123 663 1091 380 135 527 646	563 676 283 823 154 531 295 835 800 543 307 272 812 558 683 278 818 149 538 290 830 792 550 302 264 804	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426 966 573 689 438 978 943 701 450 415 955 713 1102
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611 724 1113 865 160 585 1125 877 842 597 1137 8	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004 759 1151 13 1016 623 736 25 1028 990 748 37 456 996	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888 183 1186 48 900 195 771 1163 912 207 172 1175 924 895 190	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334 86 475 1051 346 1198 60 1063 358 323 72 1075 1034 329	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1221 510 626 233 1233 98 487 245 785 1210 499 257 225 1216	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121 661 1086 381 133 522 638 393 1245 110 650 405 364 128	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796 548 303 268 808 560 673 280 820 145 534 292 832 803 540	941 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696 454 419 959 708 1100 431 971 572 685 443 983 939 703	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875 164 583 1123 881 123 881 595 1135 3 855 607 720 1112 867 156	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734 29 1029 994 746 41 466 1006 758 1147 15 1018 619 738	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916 205 170 1170 1170 928 893 182 1182 53 905 194 770 1153 923	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64 1067 356 318 76 1079 1041 330 88 477 1053 342 1194 71 1059	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240 1208 503 252 217 1220 515 628 229 1232 100 483 247	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391 1243 111 651 403 368 123 663 1091 380 135 527 646 386	563 676 283 823 154 531 295 835 800 543 307 272 812 555 683 278 818 149 538 290 830 792 550 302 264 804 562 675 282	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426 966 573 689 438 978 943 701 450 415 955 713 1102 421
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611 724 1113 865 160 585 1125 877 842 597 1137 8 860 603	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004 759 1151 13 1016 623 736 25 1028 990 748 37 456 996 766	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888 183 1186 48 900 195 771 1163 912 207 172 1175 924 895 190 1181	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334 86 475 1051 346 1198 60 1063 358 323 72 1075 1034	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1211 510 626 233 1233 98 487 245 785 1210 499 257 225 1216 505	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121 661 1086 381 133 522 638 393 1245 110 650 405 364 128 668	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796 548 303 268 808 560 673 280 820 145 534 292 832 803 540 310	941 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696 4419 959 708 1100 431 971 572 685 443 983 939 703 449	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875 164 583 1123 881 843 595 1135 3 855 607 720 1112 867 156 578 1118 885	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734 29 994 746 41 466 1006 758 1147 15 1018 619 738 33 1024	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916 205 170 1170 1170 928 893 182 1182 53 905 194 770 1153 923 212	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64 1067 356 318 76 1079 1041 330 88 477 1053 342 1194 71 1059 348	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240 780 1208 503 252 217 1220 515 628 229 1232 100 483 247 787	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391 1243 111 651 403 368 123 663 1091 380 135 527 646 386 1238	563 676 283 823 154 531 295 835 800 543 307 272 812 555 683 278 818 149 538 290 830 792 550 302 264 804 562 675 282 822 153	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426 966 573 689 943 701 450 415 955 713 1102 421 961 568
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611 724 1113 865 160 585 1125 877 842 597 1137 8	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004 759 1151 13 1016 623 736 25 1028 990 748 37 456 996	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888 183 1186 48 900 195 771 1163 912 207 172 1175 924 895 190	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334 86 475 1051 346 1198 60 1063 358 323 72 1075 1034 329	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1221 510 626 233 1233 98 487 245 785 1210 499 257 225 1216	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121 661 1086 381 133 522 638 393 1245 110 650 405 364 128	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796 548 303 268 808 560 673 280 820 145 534 292 832 803 540	941 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696 454 419 959 708 1100 431 971 572 685 443 983 939 703	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875 164 583 1123 881 123 881 135 607 720 1112 867 156 578 1118	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734 29 1029 994 746 41 466 1006 758 1147 15 1018 619 738	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916 205 170 1170 1170 928 893 182 1182 53 905 194 770 1153 923	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64 1067 356 318 76 1079 1041 330 88 477 1053 342 1194 71 1059	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240 1208 503 252 217 1220 515 628 229 1232 100 483 247	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391 1243 111 651 403 368 123 663 1091 380 135 527 646 386	563 676 283 823 154 531 295 835 800 543 307 272 812 555 683 278 818 149 538 290 830 792 550 302 264 804 562 675 282	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426 966 573 689 438 978 943 701 450 415 955 713 1102 421
37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67	598 1138 6 858 604 729 1109 870 165 581 1121 882 847 593 1133 1 853 611 724 1113 865 160 585 1125 877 842 597 1137 8 860 603	749 38 457 997 767 1144 18 1009 616 741 30 1021 986 753 42 464 1004 759 1151 13 1016 623 736 25 1028 990 748 37 456 996 766	1173 925 896 191 1179 56 908 203 779 1156 920 215 177 1168 932 888 183 1186 48 900 195 771 1163 912 207 172 1175 924 895 190 1181	73 1076 1032 327 91 468 1044 339 1191 68 1056 351 316 80 1068 1039 334 86 475 1051 346 1198 60 1063 358 323 72 1075 1034 329 90	500 255 226 1217 506 631 238 1229 103 480 250 790 1203 492 262 221 1211 510 626 233 1233 98 487 245 785 1210 499 257 225 1216 505	648 406 365 126 666 1082 377 138 518 643 389 1241 115 655 401 369 121 661 1086 381 133 522 638 393 1245 110 650 405 364 128 668	293 833 801 541 311 273 813 553 678 285 825 150 530 297 837 796 548 303 268 808 560 673 280 820 145 534 292 832 803 540 310	941 981 940 704 447 412 952 716 1093 424 964 565 690 436 976 947 696 4419 959 708 1100 431 971 572 685 443 983 939 703 449	868 157 576 1116 886 851 588 1128 11 863 600 728 1105 875 164 583 1123 881 843 595 1135 3 855 607 720 1112 867 156 578 1118 885	1019 620 739 34 1025 987 751 46 459 999 763 1140 23 1011 612 734 29 994 746 41 466 1006 758 1147 15 1018 619 738 33 1024	192 768 1154 921 210 175 1166 933 898 187 1178 58 910 199 775 1158 916 205 170 1170 1170 928 893 182 1182 53 905 194 770 1153 923 212	343 1195 69 1060 349 314 81 1072 1037 326 93 473 1049 338 1190 64 1067 356 318 76 1079 1041 330 88 477 1053 342 1194 71 1059 348	1230 101 484 248 788 1201 496 260 222 1213 508 633 234 1225 105 491 240 780 1208 503 252 217 1220 515 628 229 1232 100 483 247 787	136 525 647 384 1236 119 659 396 361 131 671 1084 373 143 520 639 391 1243 111 651 403 368 123 663 1091 380 135 527 646 386 1238	563 676 283 823 154 531 295 835 800 543 307 272 812 555 683 278 818 149 538 290 830 792 550 302 264 804 562 675 282 822 153	711 1103 422 962 569 694 434 974 936 706 446 408 948 718 1095 426 966 573 689 438 943 701 450 415 955 713 1102 421 961 568

m\n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
70	872	1008	202	341	1228	140	552	715	1130	45	935	1071	259	398	834	973
71	167	615	778	1193	102	517	680	1092	10	461	897	1036	224	360	799	938
72	580	743	1155	67	482	642	287	423	862	1001	186	325	1212	130	545	705
73	1120	32	919	1058	249	388	827	963	602	762	1177	95	507	670	306	445
74	884	1020	214	353	789	1240	152	564	727	1142	57	472	635	1083	271	410
75	846	985	179	315	1205	114	529	692	1104	22	909	1048	236	372	811	950
76	592	755	1167	79	494	654	299	435	874	1013	198	337	1224	142	557	717
77	1132	44	931	1070	261	400	839	975	163	614	774	1189	107	519	682	1097

This table describes the interleaving applied to MCS-5 and MCS-6 di(j') = dc(k') for k' = 0,1,...,1223

k' = 16\*m + n

The value of j' for a given k is in the cell located in the row m and in the column n.

Table 16: Sorting of the speech encoded bits for TCH/WFS12.65, O-TCH/WFS12.65 and O-TCH/WHS12.65

0	4	6	93	143	196	246	7	5	3
47	48	49	50	51	150	151	152	153	154
94	144	197	247	99	149	202	252	96	146
199	249	97	147	200	250	100	203	98	148
201	251	95	145	198	248	52	2	1	101
204	155	19	21	12	17	18	20	16	25
13	10	14	24	23	22	26	8	15	53
156	31	102	205	9	33	11	103	206	54
157	28	27	104	207	34	35	29	46	32
30	55	158	37	36	39	38	40	105	208
41	42	43	44	45	56	106	159	209	57
66	75	84	107	116	125	134	160	169	178
187	210	219	228	237	58	108	161	211	62
112	165	215	67	117	170	220	71	121	174
224	76	126	179	229	80	130	183	233	85
135	188	238	89	139	192	242	59	109	162
212	63	113	166	216	68	118	171	221	72
122	175	225	77	127	180	230	81	131	184
234	86	136	189	239	90	140	193	243	60
110	163	213	64	114	167	217	69	119	172
222	73	123	176	226	78	128	181	231	82
132	185	235	87	137	190	240	91	141	194
244	61	111	164	214	65	115	168	218	70
120	173	223	74	124	177	227	79	129	182
232	83	133	186	236	88	138	191	241	92
142	195	245							

Table 17: Sorting of the speech encoded bits for TCH/WFS8.85, O-TCH/WFS8.85 and O-TCH/WHS8.85

0	4	6	7	5	3	47	48	49	112
113	114	75	106	140	171	80	111	145	176
77	108	142	173	78	109	143	174	79	110
144	175	76	107	141	172	50	115	51	2
1	81	116	146	19	21	12	17	18	20
16	25	13	10	14	24	23	22	26	8
15	52	117	31	82	147	9	33	11	83
148	53	118	28	27	84	149	34	35	29
46	32	30	54	119	37	36	39	38	40
85	150	41	42	43	44	45	55	60	65
70	86	91	96	101	120	125	130	135	151
156	161	166	56	87	121	152	61	92	126
157	66	97	131	162	71	102	136	167	57
88	122	153	62	93	127	158	67	98	132
163	72	103	137	168	58	89	123	154	63
94	128	159	68	99	133	164	73	104	138
169	59	90	124	155	64	95	129	160	69
100	134	165	74	105	139	170			

Table 18: Sorting of the speech encoded bits for TCH/WFS6.60, O-TCH/WFS6.60 and O-TCH/WHS6.60

0	5	6	7	61	84	107	130	62	85
8	4	37	38	39	40	58	81	104	127
60	83	106	129	108	131	128	41	42	80
126	1	3	57	103	82	105	59	2	63
109	110	86	19	22	23	64	87	18	20
21	17	13	88	43	89	65	111	14	24
25	26	27	28	15	16	44	90	66	112
9	11	10	12	67	113	29	30	31	32
34	33	35	36	45	51	68	74	91	97
114	120	46	69	92	115	52	75	98	121
47	70	93	116	53	76	99	122	48	71
94	117	54	77	100	123	49	72	95	118
55	78	101	124	50	73	96	119	56	79
102	125								

Table 19: Sorting of the speech encoded bits for O-TCH/WFS15.85

0	4	6	109	175	244	310	7	5	3
47	48	49	50	51	182	183	184	185	186
110	176	245	311	115	181	250	316	112	178
247	313	113	179	248	314	116	251	114	180
249	315	111	177	246	312	52	2	1	117
252	187	19	21	12	17	18	20	16	25
13	10	14	24	23	22	26	8	15	53
188	31	118	253	9	33	11	119	254	54
189	28	27	120	255	34	35	29	46	32
30	55	190	37	36	39	38	40	121	256
41	42	43	44	45	56	122	191	257	63
129	198	264	76	142	211	277	89	155	224
290	102	168	237	303	57	123	192	258	70
136	205	271	83	149	218	284	96	162	231
297	62	128	197	263	75	141	210	276	88
154	223	289	101	167	236	302	58	124	193
259	71	137	206	272	84	150	219	285	97
163	232	298	59	125	194	260	64	130	199
265	67	133	202	268	72	138	207	273	77
143	212	278	80	146	215	281	85	151	220
286	90	156	225	291	93	159	228	294	98
164	233	299	103	169	238	304	106	172	241
307	60	126	195	261	65	131	200	266	68
134	203	269	73	139	208	274	78	144	213
279	81	147	216	282	86	152	221	287	91
157	226	292	94	160	229	295	99	165	234
300	104	170	239	305	107	173	242	308	61
127	196	262	66	132	201	267	69	135	204
270	74	140	209	275	79	145	214	280	82
148	217	283	87	153	222	288	92	158	227
293	95	161	230	296	100	166	235	301	105
171	240	306	108	174	243	309			

Table 20: Sorting of the speech encoded bits for O-TCH/WFS23.85

0         4         6         145         251         360         466         7         5         3           47         48         49         50         51         262         263         264         265         266           363         469         149         255         364         470         156         371         150         256           365         471         147         253         362         468         52         2         1         157           372         267         19         21         12         17         18         20         16         25           13         10         14         24         23         22         26         8         15         53           268         31         152         153         154         155         258         259         260         261           367         368         369         370         473         474         475         476         158         373           9         33         11         159         374         54         269         28         27         160           375					0=4		100			
146										
363										
365										
372         267         19         21         12         17         18         20         16         25           13         10         14         24         23         22         26         8         15         53           268         31         152         153         154         155         258         259         260         261           367         368         369         370         473         474         475         476         158         373           9         33         11         159         374         54         269         28         27         160           375         34         35         29         46         32         30         55         270         37           36         39         38         40         161         376         41         42         43         44           45         56         162         271         377         185         196         174         79         57           411         90         163         305         389         378         283         68         187         400           294										
13										
268         31         152         153         154         155         258         259         260         261           367         368         369         370         473         474         475         476         158         373           9         33         11         159         374         54         269         28         27         160           375         34         35         29         46         32         30         55         270         37           36         39         38         40         161         376         41         42         43         44           45         56         162         271         377         195         196         174         79         57           411         90         163         305         389         378         283         68         187         400           294         198         307         92         70         186         413         176         59         91           58         412         380         165         81         164         272         175         80         401 <td< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></td<>										
367         368         369         370         473         474         475         476         158         373           9         33         11         159         374         54         269         28         27         160           375         34         35         29         46         32         30         55         270         37           36         39         38         40         161         376         41         42         43         44           45         56         162         271         377         185         196         174         79         57           411         90         163         305         389         378         283         68         187         400           294         198         307         92         70         186         413         176         59         91           58         412         380         165         81         166         413         176         59         91           402         390         391         197         306         69         274         273         379         285										
9         33         11         159         374         54         269         28         27         160           375         34         35         29         46         32         30         55         270         37           36         39         38         40         161         376         41         42         43         44           45         56         162         271         377         185         196         174         79         57           411         90         163         305         389         378         283         68         187         400           294         198         307         92         70         186         413         176         59         91           58         412         380         165         81         164         272         175         80         401           402         390         391         197         306         69         274         273         379         286           296         284         295         188         60         199         82         93         71         381           414										
375         34         35         29         46         32         30         55         270         37           36         39         38         40         161         376         41         42         43         44           45         56         162         271         377         185         196         174         79         57           411         90         163         305         389         378         283         68         187         400           294         198         307         92         70         186         413         176         59         91           58         412         380         165         81         164         272         175         80         401           402         390         391         197         306         69         274         273         379         285           296         284         295         188         60         199         82         93         71         381           414         177         166         456         308         403         98         76         286         61 <td< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></td<>										
36										
45         56         162         271         377         185         196         174         79         57           411         90         163         305         389         378         283         68         187         400           294         198         307         92         70         186         413         176         59         91           58         412         380         165         81         164         272         175         80         401           402         390         391         197         306         69         274         273         379         285           296         284         295         188         60         199         82         93         71         381           414         177         166         456         308         403         98         76         286         61           275         386         135         423         171         102         392         204         87         182           65         94         208         124         72         350         193         313         393         408 <trr< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></trr<>										
411         90         163         305         389         378         283         68         187         400           294         198         307         92         70         186         413         176         59         91           58         412         380         165         81         164         272         175         80         401           402         390         391         197         306         69         274         273         379         285           296         284         295         188         60         199         82         93         71         381           414         177         166         456         308         403         98         76         286         61           275         386         135         423         171         102         392         204         87         182           65         94         208         124         72         350         193         313         393         408           445         309         230         419         297         241         113         219         189         128							41			
294         198         307         92         70         186         413         176         59         91           58         412         380         165         81         164         272         175         80         401           402         390         391         197         306         69         274         273         379         285           296         284         295         188         60         199         82         93         71         381           414         177         166         456         308         403         98         76         286         61           275         386         135         423         171         102         392         204         87         182           65         94         208         124         72         350         193         313         393         408           445         309         230         419         297         241         113         219         189         128           317         415         116         328         200         339         382         434         178         64	45	56	162	271	377	185	196	174	79	57
58         412         380         165         81         164         272         175         80         401           402         390         391         197         306         69         274         273         379         285           296         284         295         188         60         199         82         93         71         381           414         177         166         456         308         403         98         76         286         61           275         386         135         423         171         102         392         204         87         182           65         94         208         124         72         350         193         313         393         408           445         309         230         419         297         241         113         219         189         128           317         415         116         328         200         339         382         434         178         64           404         83         437         223         134         192         444         112         439         139	411	90	163	305	389	378	283	68	187	400
402         390         391         197         306         69         274         273         379         285           296         284         295         188         60         199         82         93         71         381           414         177         166         456         308         403         98         76         286         61           275         386         135         423         171         102         392         204         87         182           65         94         208         124         72         350         193         313         393         408           445         309         230         419         297         241         113         219         189         128           317         415         116         328         200         339         382         434         178         64           404         83         437         223         134         192         444         112         439         139           287         167         448         212         459         222         240         233         97         302 <td>294</td> <td>198</td> <td>307</td> <td>92</td> <td>70</td> <td>186</td> <td>413</td> <td>176</td> <td>59</td> <td>91</td>	294	198	307	92	70	186	413	176	59	91
296         284         295         188         60         199         82         93         71         381           414         177         166         456         308         403         98         76         286         61           275         386         135         423         171         102         392         204         87         182           65         94         208         124         72         350         193         313         393         408           445         309         230         419         297         241         113         219         189         128           317         415         116         328         200         339         382         434         178         64           404         83         437         223         134         192         444         112         439         139           287         167         448         212         459         222         240         233         97         302           397         234         170         276         181         455         229         438         101         280 <td>58</td> <td>412</td> <td>380</td> <td>165</td> <td>81</td> <td>164</td> <td>272</td> <td>175</td> <td>80</td> <td>401</td>	58	412	380	165	81	164	272	175	80	401
414         177         166         456         308         403         98         76         286         61           275         386         135         423         171         102         392         204         87         182           65         94         208         124         72         350         193         313         393         408           445         309         230         419         297         241         113         219         189         128           317         415         116         328         200         339         382         434         178         64           404         83         437         223         134         192         444         112         439         139           287         167         448         212         459         222         240         233         97         302           397         234         170         276         181         455         229         438         101         280           138         127         298         117         355         203         426         95         140         244	402	390	391	197	306	69	274	273	379	285
275         386         135         423         171         102         392         204         87         182           65         94         208         124         72         350         193         313         393         408           445         309         230         419         297         241         113         219         189         128           317         415         116         328         200         339         382         434         178         64           404         83         437         223         134         192         444         112         439         139           287         167         448         212         459         222         240         233         97         302           397         234         170         276         181         455         229         438         101         280           138         127         298         117         355         203         426         95         140         244           422         407         213         129         291         354         105         245         449         86	296	284	295	188	60	199	82	93	71	381
65         94         208         124         72         350         193         313         393         408           445         309         230         419         297         241         113         219         189         128           317         415         116         328         200         339         382         434         178         64           404         83         437         223         134         192         444         112         439         139           287         167         448         212         459         222         240         233         97         302           397         234         170         276         181         455         229         438         101         280           138         127         298         117         355         203         426         95         140         244           422         407         213         129         291         354         105         245         449         86           316         460         207         353         190         107         224         427         342         327	414	177	166	456	308	403	98	76	286	61
445         309         230         419         297         241         113         219         189         128           317         415         116         328         200         339         382         434         178         64           404         83         437         223         134         192         444         112         439         139           287         167         448         212         459         222         240         233         97         302           397         234         170         276         181         455         229         438         101         280           138         127         298         117         355         203         426         95         140         244           422         407         213         129         291         354         105         245         449         86           316         460         207         353         190         107         224         427         342         327           106         321         118         123         73         211         433         218         396         385	275	386	135	423	171	102	392	204	87	182
317         415         116         328         200         339         382         434         178         64           404         83         437         223         134         192         444         112         439         139           287         167         448         212         459         222         240         233         97         302           397         234         170         276         181         455         229         438         101         280           138         127         298         117         355         203         426         95         140         244           422         407         213         129         291         354         105         245         449         86           316         460         207         353         190         107         224         427         342         327           106         321         118         123         73         211         433         218         396         385           450         62         383         349         75         461         172         331         168         246	65	94	208	124	72	350	193	313	393	408
404         83         437         223         134         192         444         112         439         139           287         167         448         212         459         222         240         233         97         302           397         234         170         276         181         455         229         438         101         280           138         127         298         117         355         203         426         95         140         244           422         407         213         129         291         354         105         245         449         86           316         460         207         353         190         107         224         427         342         327           106         321         118         123         73         211         433         218         396         385           450         62         383         349         75         461         172         331         168         246           428         332         312         201         343         416         279         63         195         333	445	309	230	419	297	241	113	219	189	128
287         167         448         212         459         222         240         233         97         302           397         234         170         276         181         455         229         438         101         280           138         127         298         117         355         203         426         95         140         244           422         407         213         129         291         354         105         245         449         86           316         460         207         353         190         107         224         427         342         327           106         321         118         123         73         211         433         218         396         385           450         62         383         349         75         461         172         331         168         246           428         332         312         201         343         416         279         63         195         333           96         173         235         288         320         191         418         84         205         100	317	415	116	328	200	339	382	434	178	64
397         234         170         276         181         455         229         438         101         280           138         127         298         117         355         203         426         95         140         244           422         407         213         129         291         354         105         245         449         86           316         460         207         353         190         107         224         427         342         327           106         321         118         123         73         211         433         218         396         385           450         62         383         349         75         461         172         331         168         246           428         332         312         201         343         416         279         63         195         333           96         173         235         288         320         191         418         84         205         100           67         394         179         344         206         338         277         405         388         395	404	83	437	223	134	192	444	112	439	139
138         127         298         117         355         203         426         95         140         244           422         407         213         129         291         354         105         245         449         86           316         460         207         353         190         107         224         427         342         327           106         321         118         123         73         211         433         218         396         385           450         62         383         349         75         461         172         331         168         246           428         332         312         201         343         416         279         63         195         333           96         173         235         288         320         191         418         84         205         100           67         394         179         344         206         338         277         405         388         395           301         315         421         183         293         322         310         384         410         194	287	167	448	212	459	222	240	233	97	302
422         407         213         129         291         354         105         245         449         86           316         460         207         353         190         107         224         427         342         327           106         321         118         123         73         211         433         218         396         385           450         62         383         349         75         461         172         331         168         246           428         332         312         201         343         416         279         63         195         333           96         173         235         288         320         191         418         84         205         100           67         394         179         344         206         338         277         405         388         395           301         315         421         183         293         322         310         384         410         194           184         89         99         103         236         78         88         77         136         399 </td <td>397</td> <td>234</td> <td>170</td> <td>276</td> <td>181</td> <td>455</td> <td>229</td> <td>438</td> <td>101</td> <td>280</td>	397	234	170	276	181	455	229	438	101	280
316         460         207         353         190         107         224         427         342         327           106         321         118         123         73         211         433         218         396         385           450         62         383         349         75         461         172         331         168         246           428         332         312         201         343         416         279         63         195         333           96         173         235         288         320         191         418         84         205         100           67         394         179         344         206         338         277         405         388         395           301         315         421         183         293         322         310         384         410         194           184         89         99         103         236         78         88         77         136         399           169         202         406         125         180         440         74         387         242         231 </td <td>138</td> <td>127</td> <td>298</td> <td>117</td> <td>355</td> <td>203</td> <td>426</td> <td>95</td> <td>140</td> <td>244</td>	138	127	298	117	355	203	426	95	140	244
106         321         118         123         73         211         433         218         396         385           450         62         383         349         75         461         172         331         168         246           428         332         312         201         343         416         279         63         195         333           96         173         235         288         320         191         418         84         205         100           67         394         179         344         206         338         277         405         388         395           301         315         421         183         293         322         310         384         410         194           184         89         99         103         236         78         88         77         136         399           169         202         406         125         180         440         74         387         242         231           66         281         290         141         314         424         114         85         130         356 <td>422</td> <td>407</td> <td>213</td> <td>129</td> <td>291</td> <td>354</td> <td>105</td> <td>245</td> <td>449</td> <td>86</td>	422	407	213	129	291	354	105	245	449	86
450         62         383         349         75         461         172         331         168         246           428         332         312         201         343         416         279         63         195         333           96         173         235         288         320         191         418         84         205         100           67         394         179         344         206         338         277         405         388         395           301         315         421         183         293         322         310         384         410         194           184         89         99         103         236         78         88         77         136         399           169         202         406         125         180         440         74         387         242         231           66         281         290         141         314         424         114         85         130         356           119         299         304         398         237         409         311         417         292         457 <td>316</td> <td>460</td> <td>207</td> <td>353</td> <td>190</td> <td>107</td> <td>224</td> <td>427</td> <td>342</td> <td>327</td>	316	460	207	353	190	107	224	427	342	327
428         332         312         201         343         416         279         63         195         333           96         173         235         288         320         191         418         84         205         100           67         394         179         344         206         338         277         405         388         395           301         315         421         183         293         322         310         384         410         194           184         89         99         103         236         78         88         77         136         399           169         202         406         125         180         440         74         387         242         231           66         281         290         141         314         424         114         85         130         356           119         299         304         398         237         409         311         417         292         457           435         225         214         209         462         108         282         446         220         351     <	106	321	118	123	73	211	433	218	396	385
96         173         235         288         320         191         418         84         205         100           67         394         179         344         206         338         277         405         388         395           301         315         421         183         293         322         310         384         410         194           184         89         99         103         236         78         88         77         136         399           169         202         406         125         180         440         74         387         242         231           66         281         290         141         314         424         114         85         130         356           119         299         304         398         237         409         311         417         292         457           435         225         214         209         462         108         282         446         220         351           345         142         247         329         420         463         318         300         120         109	450	62	383	349	75	461	172	331	168	246
67         394         179         344         206         338         277         405         388         395           301         315         421         183         293         322         310         384         410         194           184         89         99         103         236         78         88         77         136         399           169         202         406         125         180         440         74         387         242         231           66         281         290         141         314         424         114         85         130         356           119         299         304         398         237         409         311         417         292         457           435         225         214         209         462         108         282         446         220         351           345         142         247         329         420         463         318         300         120         109           289         451         278         441         340         303         430         215         323         226	428	332	312	201	343	416	279	63	195	333
301         315         421         183         293         322         310         384         410         194           184         89         99         103         236         78         88         77         136         399           169         202         406         125         180         440         74         387         242         231           66         281         290         141         314         424         114         85         130         356           119         299         304         398         237         409         311         417         292         457           435         225         214         209         462         108         282         446         220         351           345         142         247         329         420         463         318         300         120         109           289         451         278         441         340         303         430         215         323         226           334         131         442         248         335         357         429         324         143         346	96	173	235	288	320	191	418	84	205	100
184         89         99         103         236         78         88         77         136         399           169         202         406         125         180         440         74         387         242         231           66         281         290         141         314         424         114         85         130         356           119         299         304         398         237         409         311         417         292         457           435         225         214         209         462         108         282         446         220         351           345         142         247         329         420         463         318         300         120         109           289         451         278         441         340         303         430         215         323         226           334         131         442         248         335         357         429         324         143         346           452         238         110         216         464         249         121         431         358         227	67	394	179	344	206	338	277	405	388	395
169         202         406         125         180         440         74         387         242         231           66         281         290         141         314         424         114         85         130         356           119         299         304         398         237         409         311         417         292         457           435         225         214         209         462         108         282         446         220         351           345         142         247         329         420         463         318         300         120         109           289         451         278         441         340         303         430         215         323         226           334         131         442         248         335         357         429         324         143         346           452         238         110         216         464         249         121         431         358         227           132         453         336         425         325         347         126         104         137         458 </td <td>301</td> <td>315</td> <td>421</td> <td>183</td> <td>293</td> <td>322</td> <td>310</td> <td>384</td> <td>410</td> <td>194</td>	301	315	421	183	293	322	310	384	410	194
66         281         290         141         314         424         114         85         130         356           119         299         304         398         237         409         311         417         292         457           435         225         214         209         462         108         282         446         220         351           345         142         247         329         420         463         318         300         120         109           289         451         278         441         340         303         430         215         323         226           334         131         442         248         335         357         429         324         143         346           452         238         110         216         464         249         121         431         358         227           132         453         336         425         325         347         126         104         137         458           352         243         447         115         341         210         330         221         232         436<	184	89	99	103	236	78	88	77	136	399
119         299         304         398         237         409         311         417         292         457           435         225         214         209         462         108         282         446         220         351           345         142         247         329         420         463         318         300         120         109           289         451         278         441         340         303         430         215         323         226           334         131         442         248         335         357         429         324         143         346           452         238         110         216         464         249         121         431         358         227           132         453         336         425         325         347         126         104         137         458           352         243         447         115         341         210         330         221         232         436	169	202	406	125	180	440	74	387	242	231
435         225         214         209         462         108         282         446         220         351           345         142         247         329         420         463         318         300         120         109           289         451         278         441         340         303         430         215         323         226           334         131         442         248         335         357         429         324         143         346           452         238         110         216         464         249         121         431         358         227           132         453         336         425         325         347         126         104         137         458           352         243         447         115         341         210         330         221         232         436	66	281	290	141	314	424	114	85	130	356
345         142         247         329         420         463         318         300         120         109           289         451         278         441         340         303         430         215         323         226           334         131         442         248         335         357         429         324         143         346           452         238         110         216         464         249         121         431         358         227           132         453         336         425         325         347         126         104         137         458           352         243         447         115         341         210         330         221         232         436	119	299	304	398	237	409	311	417	292	457
289         451         278         441         340         303         430         215         323         226           334         131         442         248         335         357         429         324         143         346           452         238         110         216         464         249         121         431         358         227           132         453         336         425         325         347         126         104         137         458           352         243         447         115         341         210         330         221         232         436	435	225	214	209	462	108	282	446	220	351
334     131     442     248     335     357     429     324     143     346       452     238     110     216     464     249     121     431     358     227       132     453     336     425     325     347     126     104     137     458       352     243     447     115     341     210     330     221     232     436	345	142	247	329	420	463	318	300	120	109
452     238     110     216     464     249     121     431     358     227       132     453     336     425     325     347     126     104     137     458       352     243     447     115     341     210     330     221     232     436	289	451	278	441	340	303	430	215	323	226
132     453     336     425     325     347     126     104     137     458       352     243     447     115     341     210     330     221     232     436	334	131	442	248	335	357	429	324	143	346
352         243         447         115         341         210         330         221         232         436	452	238	110	216	464	249	121	431	358	227
	132	453	336	425	325	347	126	104	137	458
465 319 359 111 454 228 217 122 443 348	352	243	447	115	341	210	330	221	232	436
	465	319	359	111	454	228	217	122	443	348
239 250 133 144 432 337 326	239	250	133	144	432	337	326			

#### 6 Flexible Layer One

#### 6.1 General

Data stream from higher layers (transport blocks) is encoded to offer transport services over the radio transmission link. The coding/multiplexing unit of FLO is a combination of error detection, forward error correction, rate matching, multiplexing, interleaving and burst mapping onto basic physical subchannel.

#### 6.2 Transport channel coding/multiplexing

On transport channels, data arrives to the coding/multiplexing unit in form of transport blocks (TB) once every transmission time interval (TTI).

The following coding/multiplexing steps can be identified:

- add CRC to each transport block (see subclause 6.2.1);
- channel coding (see subclause 6.2.2);
- rate matching (see subclause 6.2.3);
- multiplexing of transport channels (see subclause 6.2.4);
- TFCI encoding (see subclause 6.2.5);
- (downlink only) mapping of in-band signalling bits (see subclause 6.2.6);
- radio packet mapping (see subclause 6.2.7);
- interleaving (see subclause 6.2.8);
- mapping on a burst (see subclause 6.2.9).

The coding/multiplexing steps are shown in figure 3 below.

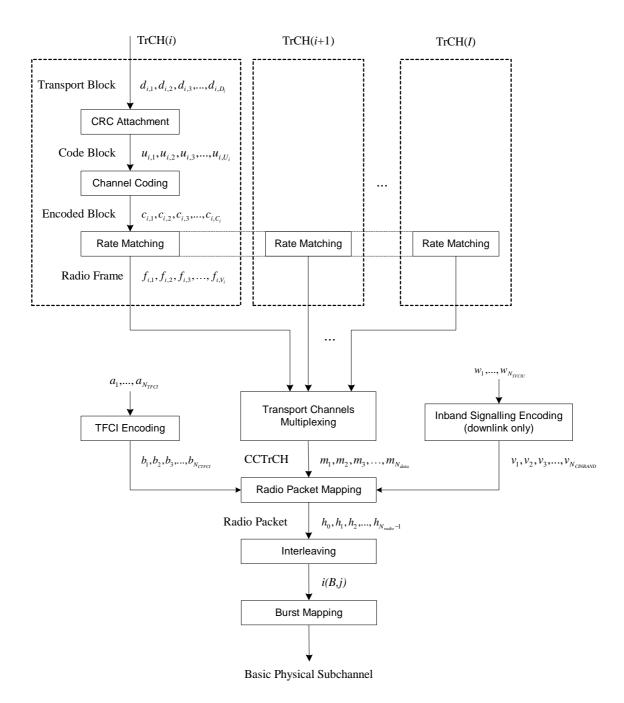


Figure 3: Transport channel coding/multiplexing

#### 6.2.1 CRC Attachment

Error detection is provided on transport blocks through a Cyclic Redundancy Check (CRC). The size of the CRC to be used is 18, 12, 6 or 0 bits and it is configured by higher layers for each TrCH.

Transport blocks are delivered to the CRC attachment block. They are denoted by  $d_{i,1}, d_{i,2}, d_{i,3}, ..., d_{i,D_i}$  where i is the TrCH number and  $D_i$  is the number of bits in the transport block.

The whole transport block is used to calculate the CRC parity bits. The parity bits are generated by one of the following cyclic generator polynomials:

- $g_{CRC18}(D) = D^{18} + D^{17} + D^{14} + D^{13} + D^{11} + D^{10} + D^{8} + D^{7} + D^{6} + D^{3} + D^{2} + 1$  same as for SACCH/TP
- $g_{CRC12}(D) = D^{12} + D^{11} + D^{10} + D^8 + D^5 + D^4 + 1$  same as for MCS-1

- 
$$g_{CRC6}(D) = D^6 + D^5 + D^3 + D^2 + D^1 + 1$$
 same as for TCH/AFS

Denote the parity bits by  $p_{i,1}$ ,  $p_{i,2}$ ,  $p_{i,3}$ ,...,  $p_{i,L_i}$ .  $L_i$  is the number of parity bits (size of the CRC) and can take the values 18, 12, 6, or 0.

The encoding is performed in a systematic form, which means that in GF(2), the polynomial:

$$d_{i,1}D^{D_i+L_i-1}+d_{i,2}D^{D_i+L_i-2}+\ldots+d_{i,D_i}D^{L_i}+p_{i,1}D^{L_i-1}+p_{i,2}D^{L_i-2}+\ldots+p_{i,L_i-1}D^1+p_{i,L_i-1}D$$

- when divided by g<sub>CRC18</sub> yields a remainder equal to:

$$1 + D + D^2 + D^3 + D^4 + D^5 + D^6 + D^7 + D^8 + D^9 + D^{10} + D^{11} + D^{12} + D^{13} + D^{14} + D^{15} + D^{16} + D^{17}$$

- when divided by g<sub>CRC12</sub> yields a remainder equal to:

$$1 + D + D^2 + D^3 + D^4 + D^5 + D^6 + D^7 + D^8 + D^9 + D^{10} + D^{11} + D^{12}$$

- when divided by g<sub>CRC6</sub> yields a remainder equal to:

$$1 + D + D^2 + D^3 + D^4 + D^5$$

The result of CRC attachment is a code block of  $U_i$  bits  $u_{i,1}, u_{i,2}, u_{i,3}, ..., u_{i,U_i}$  where:

$$\begin{split} U_i &= D_i + L_i \\ u_{i,k} &= d_{i,k} & \text{for } k = 1,2,3,...,D_i \\ \\ u_{i,k} &= p_{i,(L_i + l - (k - D_i))} & \text{for } k = D_i + 1,\, D_i + 2,...,\, D_i + L_i \end{split}$$

If no transport blocks are input to the CRC calculation, no CRC attachment shall be done.

#### 6.2.2 Channel Coding

Code blocks are delivered to the channel coding block. They are denoted by  $u_{i,1}, u_{i,2}, u_{i,3}, ..., u_{i,U_i}$  where i is the TrCH number and  $U_i$  is the number of bits in the code block. After channel coding the bits are denoted by  $c_{i,1}, c_{i,2}, c_{i,3}, ..., c_{i,C_i}$  (encoded blocks) where  $C_i$  is the number of encoded bits.

Before convolutional coding 6 tail bits with binary value 0 are added to the end of the code block:

$$u_{i,1}, u_{i,2}, u_{i,3}, ..., u_{i,U_i}, 0,0,0,0,0,0,0$$

The block is then encoded with the same 1/3 rate convolutional code as for MCS-1, defined by the following polynomials:

$$G4 = 1 + D^{2} + D^{3} + D^{5} + D^{6}$$

$$G7 = 1 + D + D^{2} + D^{3} + D^{6}$$

$$G5 = 1 + D + D^{4} + D^{6}$$

resulting in an encoded block of  $C_i$  bits {  $c_{i,1}, c_{i,2}, c_{i,3}, ..., c_{i,C_i}$  } with:

$$C_{i} = 3 \times (U_{i} + 6)$$

$$c_{i,(3k+1)} = u_{i,(k+1)} + u_{i,(k-1)} + u_{i,(k-2)} + u_{i,(k-4)} + u_{i,(k-5)};$$

$$c_{i,(3k+2)} = u_{i,(k+1)} + u_{i,k} + u_{i,(k-1)} + u_{i,(k-2)} + u_{i,(k-5)};$$

$$c_{i,(3k+3)} = u_{i,(k+1)} + u_{i,k} + u_{i,(k-3)} + u_{i,(k-5)}$$
 for  $k = 0,1,..., U_i + 5$  and  $u_{i,k} = 0$  for  $k < 1$ .

#### 6.2.3 Rate Matching

Rate matching means that bits of an encoded block on a transport channel are repeated or punctured. The number of bits on a transport channel can vary between different transmission time intervals. When the number of bits between different transmission time intervals is changed, bits are repeated or punctured to ensure that the total bit rate after TrCH multiplexing is identical to the total channel bit rate of the allocated dedicated basic physical subchannel.

Higher layers assign a rate-matching attribute to each transport channel. The rate matching attribute is used to calculate the number of bits to be repeated or punctured.

The input bit sequences before rate matching (encoded blocks) are denoted by  $c_{i,1}, c_{i,2}, c_{i,3}, ..., c_{i,C_i}$  where i is the TrCH number and  $C_i$  is the number of bits. Only one radio frame per TrCH is delivered to the rate matching block.

Notation used:

 $\lfloor x \rfloor$  Round x towards  $-\infty$ , i.e. integer such that  $x-1 < \lfloor x \rfloor \le x$ .

|x| Absolute value of x.

I Number of TrCHs in the coded composite transport channel (CCTrCH).

 $N_{\it data}$  Total number of bits that are available in a radio packet for the CCTrCH.

 $N_{i,j}$  Number of bits in an encoded block before rate matching on TrCH i with transport format combination j.

 $\Delta N_{i,j}$  If positive,  $\Delta N_{i,j}$  denotes the number of bits that have to be repeated in an encoded block on TrCH i with transport format combination j in order to produce a radio frame.

If negative,  $\Delta N_{i,j}$  denotes the number of bits that have to be punctured in an encoded block on TrCH i with transport format combination j in order to produce a radio frame.

If null, no bits have to be punctured nor repeated, i.e. the rate matching is transparent and the content of the radio frame is identical to the content of the encoded block on TrCH i with transport format combination j.

RM; Semi-static rate matching attribute for transport channel i.

 $e_{ini}$  Initial value of variable e in the rate matching pattern determination algorithm.

 $e_{plus}$  Increment of variable e in the rate matching pattern determination algorithm.

 $e_{minus}$  Decrement value of variable e in the rate matching pattern determination algorithm.

 $Z_{i,i}$  Intermediate calculation variable.

R Redundancy pattern index used for the transmission of signalling transport blocks on half rate channels (see subclause 6.2.10). In all other cases R = 0.

For each radio packet using transport format combination j, the number of bits to be repeated or punctured  $\Delta N_{i,j}$  within one encoded block for each TrCH i is calculated with the following equations:

$$Z_{0,i} = 0$$

$$Z_{i,j} = \left| \frac{\left( \sum_{m=1}^{i} RM_m \times N_{m,j} \right) \times N_{data}}{\sum_{m=1}^{I} RM_m \times N_{m,j}} \right| \quad \text{for all } i = 1 \dots I$$

$$\Delta N_{i,j} = Z_{i,j} - Z_{i-1,j} - N_{i,j} \ \text{ for all } i = 1 \, \dots \, I$$

For the calculation of the rate matching pattern of each TrCH i the following relations are defined:

$$e_{plus} = 2 \times N_{i,i}$$

$$e_{minus} = 2 \times |\Delta N_{i,j}|$$

if 
$$\Delta N_{i,i} < 0$$

if 
$$\frac{e_{plus}}{e_{\min us}} \ge 2$$

$$d = \frac{e_{plus}}{e_{\min us}} = \frac{N_{i,j}}{\left|\Delta N_{i,j}\right|}$$

-- average distance between punctured bits

$$e_{ini} = 1 + (R \mod d) \times e_{\min us}$$

else

$$d = \frac{e_{plus}}{e_{plus} - e_{\min us}} = \frac{N_{i,j}}{N_{i,j} - \left| \Delta N_{i,j} \right|} \quad \text{-- average distance between transmitted bits}$$

$$e_{ini} = 1 + (R \mod d) \times (e_{plus} - e_{\min us})$$

end if

else  $e_{ini} = 1$ 

end if.

The rate matching rule is as follows:

if  $\Delta N_{i,j} < 0$  -- puncturing is to be performed

 $e = e_{ini}$  -- initial error between current and desired puncturing ratio

m = 1 -- index of current bit

do while  $m \le N_{i,j}$  -- for each bit of the encoded block of TrCHi

 $e = e - e_{\text{minus}}$  -- update error

if  $e \le 0$  then -- check if bit number m should be punctured

puncture bit  $b_{im}$  -- bit is punctured

 $e = e + e_{plus}$  -- update error

end if

$$m = m + 1 \qquad -- \text{ next bit}$$
 end do 
$$\text{else if } \Delta N_{i,j} > 0 \qquad -- \text{ repetition is to be performed}$$
 
$$e = e_{ini} \qquad -- \text{ initial error between current and desired puncturing ratio}$$
 
$$m = 1 \qquad -- \text{ index of current bit}$$
 do while  $m \leq N_{i,j} \qquad -- \text{ for each bit of the encoded block of TrCH}i$  
$$e = e - e_{\text{minus}} \qquad -- \text{ update error}$$
 do while  $e \leq 0 \qquad -- \text{ check if bit number } m \text{ should be repeated}$  repeat bit  $b_{i,m} \qquad -- \text{ repeat bit}$  
$$e = e + e_{plus} \qquad -- \text{ update error}$$
 end do 
$$m = m + 1 \qquad -- \text{ next bit}$$
 end do 
$$else \qquad -- \Delta N_{i,j} = 0$$
 do nothing 
$$-- \text{ no repetition nor puncturing}$$
 end if.

For each TrCH i, the bit sequences output from the rate matching are denoted  $f_{i,1}, f_{i,2}, f_{i,3}, \ldots, f_{i,V_i}$ , where i is the TrCH number and  $V_i$  is the number of bits in the radio frame of TrCH i ( $V_i = N_{i,j} + \Delta N_{i,j}$ ).

#### 6.2.4 Transport Channel multiplexing

For every transmitted radio packet, one radio frame from each active TrCH is delivered to the TrCH multiplexing. These radio frames are serially multiplexed into a coded composite transport channel (CCTrCH).

The input bit sequences to the TrCH multiplexing are denoted by  $f_{i,1}, f_{i,2}, f_{i,3}, \ldots, f_{i,V_i}$  where i is the TrCH number and  $V_i$  is the number of bits in the radio frame of TrCH i. If TrCH $_i$  is inactive,  $V_i = 0$ . The number of TrCHs is denoted by I. The bits output from TrCH multiplexing are denoted  $m_1, m_2, m_3, \ldots, m_{N_{data}}$  where  $N_{data}$  is the total number of bits that are available in a radio packet for the CCTrCH, i.e.  $N_{data} = \sum_i V_i$ .

The TrCH multiplexing is defined by the following relations:

$$\begin{split} m_k &= f_{1,k} & \text{for } k = 1,2,...,V_1 \\ m_k &= f_{2,(k-V_1)} & \text{for } k = V_1+1,V_1+2,...,\ V_1+V_2 \\ & ... \\ m_k &= f_{I,(k-(V_1+V_2+...+V_{I-1}))} & \text{for } k = (V_1+V_2+...+V_{I-1})+1,\ (V_1+V_2+...+V_{I-1})+2,...,\ (V_1+V_2+...+V_{I-1})+V_1 \\ \text{OTE:} & \text{when } I = 1, \text{ the TrCH multiplexing block is transparent for the only radio frame of the only transport} \end{split}$$

channel i and consequently the output bit sequence is identical to the input one.

#### 6.2.5 TFCI Encoding

The TFCI informs the receiver about the transport format combination of the CCTrCH. As soon as the TFCI is detected, the transport format combination, and hence the transport formats of the individual transport channels are known. The size and values of the TFCI to be used on basic physical subchannels are configured by higher layers. The value of the TFCI can vary between different transmission time intervals. The size of the TFCI can only be changed through higher layer signalling.

The TFCI bit sequence is denoted by  $a_1,...,a_{N_{TFCI}}$  with  $N_{TFCI} \in \{1,2,3,4,5\}$ .

The TFCI information bits are first block coded. The coded TFCI bit sequence is denoted by  $b_1, b_2, b_3, ..., b_{N_{CTFCI}}$  with  $N_{CTFCI} \in \{4,8,12,14,16,18,24,28,32,36,48,56,72\}$ . The block coding is done according to the following rules:

On GMSK full rate channels and 8PSK half rate channels, the coding of the TFCI shall be as follows:

- 1 bit TFCI shall be encoded to 8 bits according to Table 25;
- 2 bits TFCI shall be encoded to 16 bits according to Table 24;
- 3 bits TFCI shall be encoded to 24 bits according to Table 23;
- 4 bits TFCI shall be encoded to 28 bits according to Table 22;
- 5 bits TFCI shall be encoded to 36 bits according to Table 21.

On 8PSK full rate channels, the coding of the TFCI shall be obtained by repetition of the coding defined for GMSK full rate channels:

- 1 bit TFCI shall be encoded to 16 bits (concatenation of two identical coded sequences of 8 bits);
- 2 bits TFCI shall be encoded to 32 bits (concatenation of two identical coded sequences of 16 bits);
- 3 bits TFCI shall be encoded to 48 bits (concatenation of two identical coded sequences of 24 bits);
- 4 bits TFCI shall be encoded to 56 bits (concatenation of two identical coded sequences of 28 bits);
- 5 bits TFCI shall be encoded to 72 bits (concatenation of two identical coded sequences of 36 bits).

On GMSK half rate channels, the coding of the TFCI shall be obtained by using only the middle segment of the coding defined for GMSK full rate channels:

- 1 bit TFCI shall be encoded to 4 bits;
- 2 bits TFCI shall be encoded to 8 bits;
- 3 bits TFCI shall be encoded to 12 bits;
- 4 bits TFCI shall be encoded to 14 bits;
- 5 bits TFCI shall be encoded to 18 bits.

#### 6.2.6 In-band signalling encoding

The in-band signalling bits are transmitted in the downlink direction only. The information contained in these bits is a TFCI sequence  $w_1, w_2, ..., w_{N_{TFCIU}}$ . The number of in-band signalling bits in each radio packet,  $N_{TFCIU}$ , is equal to the size of the uplink TFCI. The number of coded in-band signalling bits in each radio packet,  $N_{CINBAND}$ , is equal to the size of the coded uplink TFCI (see subclause 6.2.5).

The coded in-band signalling bit sequence is denoted  $v_1, v_2, ..., v_{N_{CINRAND}}$ .

#### 6.2.7 Radio packet mapping

The input data bit sequence is denoted by  $m_1, m_2, m_3, \ldots, m_{N_{data}}$  where  $N_{data}$  is the total number of bits that are available in a radio packet for the CCTrCH. After mapping on a radio packet the bits are denoted by  $h_0, h_1, h_2, \ldots, h_{N_{radio}-1}$  where  $N_{radio}$  is the total number of bits that are available in a radio packet:

On GMSK full rate channels,  $N_{radio} = 464$ ;

On GMSK half rate channels,  $N_{radio} = 232$ ;

On 8PSK full rate channels,  $N_{radio} = 1392$ ;

On 8PSK half rate channels,  $N_{radio} = 696$ .

The result of the radio packet mapping is a radio packet of  $N_{radio}$  bits {  $h_0, h_1, h_2, ..., h_{N_{-n}-1}$  } where:

- in the uplink:

$$N_{radio} = N_{data} + N_{CTFCI}$$
 
$$h_k = b_{k+1} \qquad \text{for } k = 0,1,2,...,N_{CTFCI} - 1$$
 
$$h_k = m_{k-N_{CTFCI}+1} \qquad \text{for } k = N_{CTFCI},N_{CTFCI}+1,...,N_{radio} - 1$$

- in the downlink:

$$\begin{split} N_{radio} &= N_{data} + N_{CTFCI} + N_{CINBAND} \\ h_k &= b_{k+1} & \text{for } k = 0,1,2,...,N_{CTFCI} - 1 \\ h_k &= v_{k-N_{CTFCI}+1} & \text{for } k = N_{CTFCI}, N_{CTFCI}+1, \dots, N_{CTFCI} + N_{CINBAND} - 1 \\ h_k &= m_{k-N_{CTFCI}-N_{CINBAND}+1} & \text{for } k = N_{CTFCI} + N_{CINBAND}, N_{CTFCI} + N_{CINBAND} + 1, \dots, N_{radio} - 1 \end{split}$$

#### 6.2.8 Interleaving

The interleaving type (block rectangular, block diagonal) and interleaving depth are configured by higher layers. The input bit sequence to the interleaving is denoted by  $h_0, h_1, h_2, ..., h_{N_{radio}-1}$  where  $N_{radio}$  is the total number of bits that are available in a radio packet. Interleaved bits are noted  $i(B,j_k)$  where B denotes the burst number and  $j_k$  the position of the bit within the burst.

The interleaving for the n<sup>th</sup> radio packet is based on the following equations:

$$i(B,j_k) = h_k$$
 for  $k = 0,1,2,...N_{radio}-1$  
$$n = 0,1,...,N,N+1,...$$
 
$$B = B_0 + 4n + k \bmod D$$
 for block diagonal interleaving:

$$a = GCD\left(D, \frac{N_{radio}}{D}\right)$$
if  $a > 1$  then  $s = \inf\left[\frac{k}{N_{radio}/a}\right]$ 

else s = 0

$$j_k = \frac{D}{M} \times \left[ \left( 49 \times (k+s) \right) \mod \frac{J}{D/M} \right] + \inf \left[ \frac{k \mod D}{M} \right]$$

for block rectangular interleaving:

$$a = GCD\left(2D, \frac{N_{radio}}{2D}\right)$$

if 
$$a > 1$$
 then  $s = \inf \left[ \frac{k}{N_{radio} / a} \right]$ 

else s = 0

$$j_k = \frac{2D}{M} \times \left[ \left( 49 \times (k+s) \right) \mod \frac{J}{2D/M} \right] + \operatorname{int} \left[ \frac{k \mod 2D}{M} \right]$$

where:

 $j_k$  is the position of the bit k within the burst B;

D is the interleaving depth in bursts;

*J* is the burst size in bits  $(J = N_{radio} / M)$ ;

M is the size of the radio packet in bursts (M = 4 for full rate channels, M = 2 for half rate channels);

GCD(m,n) is the greatest common divisor of m and n.

On 8PSK channels, bit swapping for the coded bits of the TFCI is performed:

```
cpt = 0 \qquad \qquad \text{-- counter of the swapped bits} for k = 0,1,2,3,...,N_{CTFCI} - 1 if (j_k + 1) \mod 3 = 0 -- the coded bit is to be mapped on a weak bit of the 8PSK symbol cpt = cpt + 1 -- increment the counter of swapped bits if (cpt \mod 2 = 0) Swap bit h_k with bit h_{k+80} else Swap bit h_k with bit h_{N_{radio}-80+k} end if end if
```

The value of  $N_{radio}$  is specified in subclause 6.2.7. On GMSK channels J = 116, whereas on 8PSK channels J = 348.

For diagonal interleaving over 40 ms (used on full rate channels), D = 8. The result of the interleaving is then a distribution of the reordered bits over 8 bursts, using the even numbered position of the first 4 bursts and the odd positions of the last 4 bursts.

For diagonal interleaving over 4 bursts (used on half rate channels), D = 4. The result of the interleaving is then a distribution of the reordered bits over 4 bursts, using the even numbered position of the first 2 bursts and the odd positions of the last 2 bursts.

For diagonal interleaving over 60 ms (used on full rate channels), D = 12. The result of the interleaving is then a distribution of the reordered bits over 12 bursts, allocating one third of the bits to each of three consecutive radio packets.

For diagonal interleaving over 60 ms (used on half rate channels), D = 6. The result of the interleaving is then a distribution of the reordered bits over 6 bursts, allocating one third of the bits to each of three consecutive radio packets.

For block rectangular interleaving over 20 ms (used on full rate channels), D = 4. The result of the interleaving is the distribution of the reordered bits over  $2 \times D = 8$  blocks as in the case of block diagonal interleaving over 8 bursts, and the first 4 blocks are combined with the last 4 blocks: block 0 is combined with block 4, block 1 is combined with block 5, block 2 is combined with block 6, block 3 is combined with block 7.

Block diagonal interleaving over 60 ms shall be used for 8PSK modulation only.

#### 6.2.9 Mapping on a Burst

The mapping is given by the rule:

e(B,j) = i(B,j) for j = 0,1,...,115 on GMSK channels;

for j = 0,1,...,347 on 8PSK channels.

NOTE: No stealing flags are used with FLO.

Table 21: Block Code for 5 bits TFCI

TFCI	Coded TFCI
0,0,0,0,0	1,
0,0,0,0,1	1,1,1,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1
0,0,0,1,0	1,1,1,0,1,0,1,1,0,0,1,1,0,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,0,0,1,1,0,0,1
0,0,0,1,1	1,1,1,0,0,0,0,1,1,0,0,1,1,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0
0,0,1,0,0	1,1,0,1,1,0,0,0,1,1,1,1,0,1,0,0,0,0,1,1,1,1,0,0,0,0,1,1,1,0,0,0,0,1,1,1
0,0,1,0,1	1,1,0,1,0,0,1,0,0,1,0,1,1,1,1,0,1,0,0,1,0,1,1,0,1,0,0,1,0,1,0,1,0,1,0
0,0,1,1,0	1,1,0,0,1,1,0,0,0,0,1,1,1,1,1,1,0,0,0,0
0,0,1,1,1	1,1,0,0,0,1,1,0,1,0,0,1,0,1,0,1,1,0,1,0
0,1,0,0,0	1,0,1,1,1,0,0,0,0,0,0,1,0,1,1,1,1,1,1,1
0,1,0,0,1	1,0,1,1,0,0,1,0,1,0,1,0,0,0,0,1,0,1,0,1
0,1,0,1,0	1,0,1,0,1,1,0,0,1,1,0,0,0,0,0,1,1,0,0,1,0,0,0,1,1,0,0,1,1,1,0,0,1,1,0
0,1,0,1,1	1,0,1,0,0,1,1,0,0,1,1,0,1,0,1,0,0,1,1,0,0,0,1,1,0,0,0,1,1,0,0,1,1
0,1,1,0,0	1,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,1,1,1,1,1,1,1,1,0,0,0
0,1,1,0,1	1,0,0,1,0,1,0,1,1,0,1,0,1,0,1,0,0,1,0,0,1,0,1,0,0,1,0,0,1,0,0,1,0,1
0,1,1,1,0	1,0,0,0,1,0,1,1,1,1,0,0,1,0,1,1,0,0,0,0
0,1,1,1,1	1,0,0,0,0,0,1,0,1,1,0,0,0,0,1,1,0,1,0,0,0,0,1,1,0,1,0,0,1,0,0,1,0,1,1
1,0,0,0,0	0,1,1,1,1,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1
1,0,0,0,1	0,1,1,1,0,0,1,0,1,0,1,0,1,1,0,1,0,1,0,1
1,0,0,1,0	0,1,1,0,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1
1,0,0,1,1	0,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,0,1,1,0,0,1,1,1,0,0,1,1,0,0
1,0,1,0,0	0,1,0,1,1,1,1,0,0,0,0,1,1,0,0,0,0,1,1,1,0,1,1,1,1,0,0,0,0,0,0,0,0,1,1,1
1,0,1,0,1	0,1,0,1,0,1,0,1,1,0,1,0,0,1,1,0,1,0,0,1,0,0,1,0,1,1,0,1,1,0,1,0,0,1,0
1,0,1,1,0	0,1,0,0,1,0,1,1,1,1,0,0,0,1,1,1,0,0,0,0
1,0,1,1,1	0,1,0,0,0,0,1,0,1,1,0,1,1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,0,1,1,0,1,0,0
1,1,0,0,0	0,0,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1
1,1,0,0,1	0,0,1,1,0,1,0,1,0,1,0,1,1,0,0,1,0,1,0,1
1,1,0,1,0	0,0,1,0,1,0,1,1,0,0,1,1,1,0,0,0,1,1,1,1
1,1,0,1,1	0,0,1,0,0,0,0,1,1,0,0,1,0,0,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,1
1,1,1,0,0	0,0,0,1,1,0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,0,0,0
1,1,1,0,1	0,0,0,1,0,0,1,0,0,1,0,1,0,1,0,1,0,1,0,1
1,1,1,1,0	0,0,0,0,1,1,0,0,0,0,1,1,0,0,1,1,0,0,0,1,1,0,0,1,1,1,1,0,0,0,1,1,1,1,0
1,1,1,1,1	0,0,0,0,1,1,0,1,0,0,1,1,0,0,1,1,0,1,0,0,1,1,0,0,1,0,1,1,1,0,0,1,0,1,1

Table 22: Block Code for 4 bits TFCI

TFCI	Coded TFCI
0,0,0,0	1,
0,0,0,1	1,1,1,1,1,0,1,0,0,1,0,0,0,0,0,1,1,1,1,1
0,0,1,0	1,1,1,0,0,1,0,1,0,0,1,1,0,0,0,1,1,1,0,0,0,0,1,1,1,1,1,0,0
0,0,1,1	1,1,1,0,0,0,0,0,1,0,0,0,1,1,1,1,1,1,0,0,0,0,0,0,0,0,1,1
0,1,0,0	1,0,0,1,1,1,0,1,1,0,0,0,1,0,0,1,0,0,1,1,0,0,1,1,0,0,1,1
0,1,0,1	1,0,0,1,0,0,1,1,0,0,1,0,0,1,1,1,0,0,1,1,0,0,0,0,1,1,0,0
0,1,1,0	1,0,0,0,1,0,0,0,1,1,1,1,0,1,0,1,0,0,0,0
0,1,1,1	1,0,0,0,0,1,1,0,0,1,0,1,1,0,1,1,0,0,0,0
1,0,0,0	0,1,0,1,1,1,0,0,0,0,0,1,0,1,1,0,1,0,1,0
1,0,0,1	0,1,0,1,0,0,1,0,1,0,1,1,1,0,0,0,1,0,1,0
1,0,1,0	0,1,0,0,1,0,0,1,0,1,1,0,1,0,1,0,1,0,0,1,0,1,1,0,1,0,0,1
1,0,1,1	0,1,0,0,0,1,1,1,1,1,0,0,0,1,0,0,1,0,0,1,0,1,0,1,0,1,1,0
1,1,0,0	0,0,1,1,0,1,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0,1,1,0,0,1,1,0
1,1,0,1	0,0,1,1,0,0,0,1,0,1,0,1,1,1,0,0,0,1,1,0,0,1,0,1,1,0,0,1
1,1,1,0	0,0,1,0,1,1,1,0,0,0,1,0,1,1,0,0,0,1,0,1
1,1,1,1	0,0,1,0,1,0,1,1,1,0,0,1,0,0,1,0,0,1,0,1

Table 23: Block Code for 3 bits TFCI

TFCI	Coded TFCI
0,0,0	1,
0,0,1	1,1,1,0,0,0,0,1,1,1,0,0,0,0,1,1,1,0,0,0,0,1,1,1
0,1,0	1,0,0,1,1,0,0,1,0,0,1,1,0,0,1,0,0,1,1,0,0,1,0,0
0,1,1	1,0,0,0,0,1,1,1,0,0,0,0,1,1,1,0,0,0,0,1,1,1,0,0
1,0,0	0,1,0,1,0,1,0,0,1,0,1,0,1,0,0,1,0,1,0,1
1,0,1	0,1,0,0,1,0,1,0,1,0,0,1,0,1,0,1,0,0,1,0,1,0,1,0
1,1,0	0,0,1,1,0,0,1,0,0,1,1,0,0,1,0,0,1,1,0,0,1,0,0,1
1,1,1	0,0,1,0,1,1,0,0,0,1,0,1,1,0,0,0,1,0,1,1,0,0,0,1

Table 24: Block Code for 2 bits TFCI

TFCI	Coded TFCI
0,0	1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
0,1	1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1
1,0	0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0
1,1	0.0,1,0,0,1,0,0,1,0,0,1,0,0,1,0

Table 25: Block Code for 1 bit TFCI

TFCI	Coded TFCI
0	1,1,1,1,1,1,1,1
1	0,0,0,0,0,0,0,0

#### 6.2.10 Signalling on Half Rate Channels

When a signalling transport block is sent on half rate channels, the value of the redundancy pattern index (*R*) to be used in rate matching is given by the TDMA frame number (see 3GPP TS 45.002) of the first burst carrying coded bits of the corresponding radio packet, and table 26 below.

NOTE: A radio packet containing a signalling transport block is indicated by TFCI=0.

Table 26: R and TDMA frame number modulo 26

TDMA frame number	R
0, 1, 2, 3	0
4, 5, 6, 7	1
8, 9, 10, 11	0
13, 14, 15, 16	1
17, 18, 19, 20	0
21, 22, 23, 24	1

NOTE: For a given signalling transport block, this could result in R=1 being used in rate matching during the first transmission and R=0 during the subsequent retransmission.

### Annex A (informative): Summary of Channel Types

TCH/EFS: enhanced full rate speech traffic channel

TCH/FS: full rate speech traffic channel

TCH/HS: half rate speech traffic channel

TCH/AFS: adaptive multirate full rate speech traffic channel

TCH/AFS12.2 adaptive multirate full rate speech, 12.2 kbit/s

TCH/AFS10.2 adaptive multirate full rate speech, 10.2 kbit/s

TCH/AFS7.95 adaptive multirate full rate speech, 7.95 kbit/s

TCH/AFS7.4 adaptive multirate full rate speech, 7.5 kbit/s

TCH/AFS6.7 adaptive multirate full rate speech, 6.7 kbit/s

TCH/AFS5.9 adaptive multirate full rate speech, 5.9 kbit/s

TCH/AFS5.15 adaptive multirate full rate speech, 5.15 kbit/s

TCH/AFS4.75 adaptive multirate full rate speech, 4.75 kbit/s

TCH/AHS: adaptive multirate half rate speech traffic channel

TCH/AHS7.95 adaptive multirate half rate speech, 7.95 kbit/s

TCH/AHS7.4 adaptive multirate half rate speech, 7.5 kbit/s

TCH/AHS6.7 adaptive multirate half rate speech, 6.7 kbit/s

TCH/AHS5.9 adaptive multirate half rate speech, 5.9 kbit/s

TCH/AHS5.15 adaptive multirate half rate speech, 5.15 kbit/s

TCH/AHS4.75 adaptive multirate half rate speech, 4.75 kbit/s

O-TCH/AHS: adaptive multirate half rate 8PSK speech traffic channel

O-TCH/AHS12.2 adaptive multirate half rate 8PSK speech, 12.2 kbit/s

O-TCH/AHS10.2 adaptive multirate half rate 8PSK speech, 10.2 kbit/s

O-TCH/AHS7.95 adaptive multirate half rate 8PSK speech, 7.95 kbit/s

O-TCH/AHS7.4 adaptive multirate half rate 8PSK speech, 7.5 kbit/s

O-TCH/AHS6.7 adaptive multirate half rate 8PSK speech, 6.7 kbit/s

O-TCH/AHS5.9 adaptive multirate half rate 8PSK speech, 5.9 kbit/s

O-TCH/AHS5.15 adaptive multirate half rate 8PSK speech, 5.15 kbit/s

O-TCH/AHS4.75 adaptive multirate half rate 8PSK speech, 4.75 kbit/s

TCH/WFS wideband adaptive multirate full rate speech traffic channels

TCH/WFS12.65 wideband adaptive multirate full rate speech, 12.65 kbit/s

TCH/WFS8.85 wideband adaptive multirate full rate speech, 8.85 kbit/s

TCH/WFS6.60 wideband adaptive multirate full rate speech, 6.60 kbit/s

O-TCH/WFS: adaptive multirate full rate 8PSK wideband speech traffic channel

O-TCH/WFS23.85 adaptive multirate full rate 8PSK wideband speech, 23.85 kbit/s

O-TCH/WFS15.85 adaptive multirate full rate 8PSK wideband speech, 15.85 kbit/s

O-TCH/WFS12.65 adaptive multirate full rate 8PSK wideband speech, 12.65 kbit/s

O-TCH/WFS8.85 adaptive multirate full rate 8PSK wideband speech, 8.85 kbit/s

O-TCH/WFS6.6 adaptive multirate full rate 8PSK wideband speech, 6.6 kbit/s

O-TCH/WHS: adaptive multirate half rate 8PSK wideband speech traffic channel

O-TCH/WHS12.65 adaptive multirate half rate 8PSK wideband speech, 12.65 kbit/s

O-TCH/WHS8.85 adaptive multirate half rate 8PSK wideband speech, 8.85 kbit/s

O-TCH/WHS6.6 adaptive multirate half rate 8PSK wideband speech, 6.6 kbit/s

E-TCH/F43.2: 43.2 kbit/s full rate data traffic channel

E-TCH/F32.0: 32.0 kbit/s full rate data traffic channel

E-TCH/F28.8: 28.8 kbit/s full rate data traffic channel

TCH/F14.4 14.4 kbit/s full rate data traffic channel

TCH/F9.6: 9.6 kbit/s full rate data traffic channel

TCH/F4.8: 4.8 kbit/s full rate data traffic channel

TCH/H4.8: 4.8 kbit/s half rate data traffic channel

TCH/F2.4: 2.4 kbit/s full rate data traffic channel

TCH/H2.4: 2.4 kbit/s half rate data traffic channel

SACCH: slow associated control channel

FACCH/F: fast associated control channel at full rate

FACCH/H: fast associated control channel at half rate

E-FACCH/F: enhanced circuit switched fast associated control channel at full rate

O-FACCH/H octal fast associated control channel at half rate

EPCCH: Enhanced power control channel

SDCCH: stand-alone dedicated control channel

BCCH: broadcast control channel

PCH: paging channel

AGCH access grant channel

RACH: random access channel

SCH: synchronization channel

CBCH: cell broadcast channel

CTSBCH-SB: CTS beacon channel (synchronisation burst)

CTSPCH: CTS paging channel

CTSARCH: CTS access request channel

CTSAGCH: CTS access grant channel

PDTCH packet data traffic channel

PACCH packet associated control channel

PBCCH packet broadcast control channel

PAGCH packet access grant channel

PPCH packet paging channel

PTCCH packet timing advance control channel

PRACH packet random access channel

CFCCH Compact Frequency Correction Channel

CPAGCH Compact Packet Access Grant Channel

CPBCCH Compact Packet Broadcast Control Channel

CPCCCH Compact Packet Common Control Channel

CPPCH Compact Packet Paging Channel

CPRACH Compact Packet Random Access Channel

CSCH Compact Synchronization Channel

MPRACH MBMS Packet Random Access Channel

EC-SCH Extended Coverage synchronization channel

EC-BCCH Extended Coverage broadcast control channel

EC-PCH Extended Coverage paging channel

EC-AGCH Extended Coverage access grant channel

EC-RACH Extended Coverage random access channel

EC-PDTCH Extended Coverage packet data traffic channel

EC-PACCH Extended Coverage packet associated control channel

 $G9 = 1 + D + D^3$ 

# Annex B (informative): Summary of Polynomials Used for Convolutional Codes and

Turbo Codes	
$G0 = 1 + D^3 + D^4$	TCH/FS, TCH/EFS, TCH/AFS, TCH/WFS, TCH/AHS, TCH/F14.4, TCH/F9.6, TCH/H4.8, SDCCH, BCCH, PCH, SACCH, FACCH, E-FACCH, AGCH, RACH, SCH, CSCH, EC-SCH, CTSBCH-SB, CTSPCH, CTSARCH, CTSAGCH, PDTCH (CS-1, CS-2, CS3, CS-4), PACCH, PBCCH, PAGCH, PPCH, PTCCH, PRACH, CPBCCH, CPAGCH, CPPCH, MPRACH, EC-RACH (M=1, 4, 16, 48), EC-BCCH
$G1 = 1 + D + D^3 + D^4$	TCH/FS, TCH/EFS, TCH/AFS, TCH/WFS, TCH/AHS, TCH/F14.4, TCH/F9.6, TCH/H4.8, SACCH, FACCH, E-FACCH, SDCCH, BCCH,PCH, AGCH, RACH, SCH, CSCH, EC-SCH, TCH/F4.8, TCH/F2.4, TCH/H2.4,PDTCH(CS-1, CS-2, CS-3, CS-4), PACCH, PBCCH, PAGCH, PPCH, PTCCH, PRACH, CPBCCH, CPAGCH, CPPCH, MPRACH, EC-RACH (M=1, 4, 16, 48), EC-BCCH
$G2 = 1 + D^2 + D^4$	TCH/AFS, TCH/WFS, TCH/F4.8, TCH/F2.4, TCH/H2.4
$G3 = 1 + D + D^2 + D^3 + D^4$	TCH/AFS, TCH/WFS, TCH/F4.8, TCH/F2.4, TCH/H2.4
$G4 = 1 + D^2 + D^3 + D^5 + D^6$	TCH/HS, TCH/AFS, TCH/AHS, O-TCH/AHS, O-TCH/WFS, O-TCH/WHS, E-TCH/F43.2, E-TCH/F32.0, E-TCH/F28.8, PDTCH(MCS-1, MCS-2, MCS-3, MCS-4, MCS-5, MCS-6, MCS-7, MCS-8, MCS-9, UAS-7, UAS-8, UAS-9, UAS-10, UAS-11, UBS-5, UBS-6, UBS-7, UBS-8, UBS-9, UBS-10, UBS-11, UBS-12), SACCH/TP, O-FACCH/H, O-FACCH/F, EC-CCCH/D, EC-PACCH, EC-RACH (M=66)
$G5 = 1 + D + D^4 + D^6$	TCH/HS, TCH/AFS, TCH/AHS, O-TCH/AHS, O-TCH/WFS, O-TCH/WHS, E-TCH/F32.0, PDTCH(MCS-1, MCS-2, MCS-3, MCS-4, MCS-5, MCS-6, MCS-7, MCS-8, MCS-9, UAS-7, UAS-8, UAS-9, UAS-10, UAS-11, UBS-5, UBS-6, UBS-7, UBS-8, UBS-9, UBS-10, UBS-11, UBS-12), O-FACCH/H, O-FACCH/F, EC-CCCH/D, EC-PACCH, EC-RACH (M=66)
$G6 = 1 + D + D^2 + D^3 + D^4 + D^6$	TCH/HS, TCH/AFS, TCH/AHS, O-TCH/AHS, O-TCH/WFS, O-TCH/WHS, O-FACCH/H, O-FACCH/F, EC-RACH (M=66)
$G7 = 1 + D + D^2 + D^3 + D^6$	O-TCH/AHS, O-TCH/WFS, O-TCH/WHS, E-TCH/F43.2, E-TCH/F32.0, E-TCH/F28.8, PDTCH(MCS-1, MCS-2, MCS-3, MCS-4, MCS-5, MCS-6, MCS-7, MCS-8, MCS-9, UAS-7, UAS-8, UAS-9, UAS-10, UAS-11, UBS-5, UBS-6, UBS-7, UBS-8, UBS-9, UBS-10, UBS-11, UBS-12), SACCH/TP, O-FACCH/H, O-FACCH/F, EC-CCCH/D, EC-PACCH, EC-RACH (M=66)
$G8 = 1 + D^2 + D^3$	PDTCH(DAS-5, DAS-6, DAS-7, DAS-8, DAS-9, DAS-10, DAS-11, DAS-12, DBS-5, DBS-6, DBS-7, DBS-8, DBS-9, DBS-10, DBS-11, DBS-12)

PDTCH(DAS-5, DAS-6, DAS-7, DAS-8, DAS-9, DAS-10, DAS-11, DAS-12,

DBS-5, DBS-6, DBS-7, DBS-8, DBS-9, DBS-10, DBS-11, DBS-12)

## Annex C (informative): Change history

SPEC	SMG#	CR	PHASE	VERS	NEW_VERS	SUBJECT
05.03	s25	A015	R97	6.0.0	6.1.0	14.4kbps Data Service
05.03	s27		R97	6.1.0	6.1.2	Change of status to EN
05.03	s28	A017	R97	6.1.2	6.2.0	Clarification on the definition of USF precoding
05.03	s28	A016	R98	6.2.0	7.0.0	Introduction of CTS in 05.03
05.03	s28		R98	7.0.0	7.0.1	Correction to Figure 1
05.03	s29	A021	R98	7.0.1	7.1.0	Introduction of AMR
05.03	s29	A022	R99	7.1.0	8.0.0	Introduction of ECSD/EDGE
05.03	s30	A023	R99	8.0.0	8.1.0	Introduction of Fast power Control for ECSD in 05.03
05.03	s30	A025	R99	8.0.0	8.1.0	EGPRS Channel Coding
05.03	s30	A026	R99	8.0.0	8.1.0	AMR Channel Coding
05.03	s30	A027	R99	8.0.0	8.1.0	EDGE Compact logical channels
05.03	s30	A029	R99	8.0.0	8.1.0	Correction of several small bugs in the AMR section / Optimization of the transmission of the in-band parameter Mode Indication
05.03	s30	A030	R99	8.0.0	8.1.0	E-FACCH/F interleaving
05.03	s30	A032	R99	8.0.0	8.1.0	Introduction of RATSCCH for AMR
05.03	s30b	A033	R99	8.1.0	8.2.0	Correction of EGPRS channel coding
05.03	s31	A035	R99	8.2.0	8.3.0	Correction concerning SID_FIRST and clarification concerning bit order of codec mode code words
05.03	s31	A036	R99	8.2.0	8.3.0	Editorial correction for ECSD channel coding
05.03	s31	A037	R99	8.2.0	8.3.0	Correction for EGPRS Channel Coding
05.03	S31b	A039	R99	8.3.0	8.4.0	Fast inband signalling: E-IACCH
05.03	S32	A040	R99	8.4.0	8.5.0	Clarification of stealing bits for MCS-1 to 4
05.03	S32	A041	R99	8.4.0	8.5.0	Correction to the interleaving formula of MCS-8 case
						GERAN#2 November 2000
05.03	G02	A043	R99	8.5.0	8.6.0	Correction of errors in coding schemes

Date	TSG #	TSG Doc.	CR	Rev	Change history   Subject/Comment	Old	New
2001-01	03	GP-010261	A045	Rev	CR 05.03-A045 Editorial Correction to SACCH Block	8.6.0	4.0.0
2001-01	03	GP-010261	A045		Coding	0.0.0	4.0.0
2001-01	03	GP-010242	A046		CR 05.03-A046 Channel coding for TCH/WFS	4.0.0	5.0.0
2001-06	05	GP-011412	004	1	Introduction of EPC and SACCH/TP 5		5.1.0
2001-06	05	GP-011264	005	'	Channel coding of AMR-NB codec on O-TCH/H 5		5.1.0
2001-08	06	GP-011919	006	1	Editorial changes due to the introduction of O-TCH/AHS 5.		5.2.0
2001-08	06	GP-011778	007	<u> </u>			5.2.0
2001-08	06	GP-011779	008		AMR signaling frames for O-TCH/AHS	5.1.0 5.1.0	5.2.0
2001-11	07	GP-012771	010	1	Correction of references to relevant 3GPP TSs	5.2.0	5.3.0
2001-11	07	GP-012650	012	1	Update of channel coding and interleaving organization	5.2.0	5.3.0
2001-11	07	GP-012758	014		Correction of interleaving index	5.2.0	5.3.0
2002-02	08	GP-020055	015		Correction to channel coding for TCH/WFS	5.3.0	5.4.0
2002-04	09	GP-021203	018	1	Cleaning & Updates	5.4.0	5.5.0
2002-04	09	GP-021169	020	1	Alignment of number of codecs for WB-AMR to proposed		5.5.0
				Ī	set		
2002-06	10	GP-022025	016	3	Channel coding for AMR-WB on O-TCH	5.5.0	5.6.0
2002-06	10	GP-021435	021		Corrections and clean up	5.5.0	5.6.0
2002-06	10	GP-021761	022		Update of channel coding and interleaving organisation for AMR-WB 8-PSK	5.5.0	5.6.0
2003-04	14	GP-030758	024		Padding for MCS-8 Retransmissions	5.6.0	5.7.0
2003-06	15	GP-031426	025		Correction of SACCH/TP	5.7.0	5.8.0
2003-06	15	GP-031543	027		Correction of reordering of bits for O-FACCH/H	5.7.0	5.8.0
2003-08	16	GP-031995	028		Correction of parameters for TCH/FS	5.8.0	6.0.0
2003-11	17	GP-032767	026	3	Coding/Multiplexing unit for the Flexible Layer One	6.0.0	6.1.0
2003-11	17	GP-032458	029		Figure 1e	6.0.0	6.1.0
2003-11	17	GP-032616	030		11 information bits access burst on RACH	6.0.0	6.1.0
2004-02	18	GP-040167	031		Correction of encoded in-band data bits	6.1.0	6.2.0
2004-04	19	GP-041165	033	1	One TFC for signalling on HR channels	6.2.0	6.3.0
2004-06	20	GP-041666	034	1	Signalling for Uplink TFC selection for FLO	6.3.0	6.4.0
2004-06	20	GP-041369	035	•	Corrections for FLO	6.3.0	6.4.0
2004-06	20	GP-041554	036		Small editorial correction to F.32 Channel Coding for ECSD 6.		6.4.0
2004-08	21	GP-041935	037		Addition of RATSCCH for TCH/WFS	6.4.0	6.5.0
2004-11	22	GP-042471	042		Introduction of MBMS	6.5.0	6.6.0
2004-11	22	GP-042824	045	1	FLO-compatible quick fix for VT over GERAN	6.5.0	6.6.0
2004-11	22	GP-042786	046		Removal of PTM-M	6.5.0	6.6.0
2005-01	23	GP-050485	0047	1	Inclusion of 60ms interleaving for FLO	6.6.0	6.7.0
2005-01	23	GP-050040	0050		Correction to E-FACCH/F for E-TCH/F32.0	6.6.0	6.7.0
2005-01	23	GP-050490	0053		Interleaving for E-TCH/F32.0	6.6.0	6.7.0
2005-09	26	GP-051984	0055		Correction to stealing flags for SACCH/TP	6.7.0	6.8.0
2006-01	28	GP-060014	0060		Correction to the text in SACCH/TP Convolutional code	6.8.0	6.9.0
2006-04	29	GP-060922	0063	1	Correction of confusing text	6.9.0	7.0.0
2007-02	33	GP-070366	0067	1	Correction to the channel coding of the synchronization channel	7.0.0	7.1.0
2007-08	35	GP-071549	0068	4	Introduction of Reduced TTI	7.1.0	7.2.0
2007-08	35	GP-071500	0069	2	Introduction of Fast Ack/Nack Reporting	7.1.0	7.2.0
2007-08	35	GP-071543	0070	2	Introduction of channel coding for RED HOT and HUGE	7.1.0	7.2.0
2007-11	36	GP-071966	0072	1	FANR instead of RL and miscellaneous corrections on Reduced Latency	7.2.0	7.3.0
2007-11	36	GP-071676	0073		Deletion of RL-EGPRS in EGPRS2	7.2.0	7.3.0
2007-11	36	GP-071689	0074		Puncturing patterns for EGPRS PAN	7.2.0	7.3.0
2007-11	36	GP-071691	0075		Correction to stealing flag sequences for RTTI configurations	7.2.0	7.3.0
2007-11	36	GP-071963	0076	1	Corrections to PAN	7.2.0	7.3.0
2007-11	36	GP-071963 GP-071858	0078	1	Puncturing patterns for EGPRS2 PAN	7.2.0	7.3.0
<u></u>	36	GP-071958	0080	1	Channel coding for EGPRS2	7.2.0	7.3.0
2007 11	36		0081	2	Introduction of EGPRS-2 (RED HOT rate matching)	7.2.0	7.3.0
		GP-072029			USF coding for EGPRS2	7.2.0	
2007-11					IUSE GOUNG IOLEGERS/	17.Z.U	7.3.0
2007-11 2007-11	36	GP-071745	0082	1			
2007-11 2007-11 2007-11	36 36	GP-071964	0084	1	Bit swapping for EGPRS PAN	7.2.0	7.3.0
2007-11 2007-11 2007-11 2007-11 2007-11	36			1			

2008-02	37	GP-080362	0088	1	LATRED and EGPRS2	7.3.0	7.4.0
2008-02	37	GP-080115	0089		Corrections for LATRED and HUGE	7.3.0	7.4.0
2008-02	37	GP-080128	0090		Corrections for LATRED and EGPRS2	7.3.0	7.4.0
2008-02	37	GP-080133			Modified puncturing for UBS-12		7.4.0
2008-02	37	GP-080135	0092		Modified USF coding for EGPRS2-B	7.3.0 7.3.0	7.4.0
2008-02	37	GP-080175	0094		Corrections in EGPRS2 (RED HOT rate matching)	7.3.0	7.4.0
2008-05	38	GP-080665	0096		Correction to burst mapping for HUGE A	7.4.0	7.5.0
2008-05	38	GP-080760	0097	1	Miscellaneous corrections to EGPRS2	7.4.0	7.5.0
2008-08	39	GP-081314	0102	1	Clarification to EGPRS2	7.5.0	7.6.0
2008-08	39	GP-081124	0103		Correction to MCS-8 PAN	7.5.0	7.6.0
2008-08	39	GP-081125	0104		Correction to PAN bit swapping for MCS-5/6	7.5.0	7.6.0
2008-08	39	GP-081313	0105	1	EGPRS2 corrections	7.5.0	7.6.0
2008-11	40	GP-081586	0099	2	Mixed modulation USF	7.6.0	7.7.0
2008-11	40	GP-081462	0108		Correction linked to the mapping on a burst for UBS-10 to UBS-12	7.6.0	7.7.0
2008-11	40	GP-081483	0109		Correction to speech channel at full rate	7.6.0	7.7.0
2008-11	40	GP-081599	0110		Addition of PAN bit swapping for MCS-5-6 UL	7.6.0	7.7.0
2008-12	40				Version for Release 8	7.7.0	8.0.0
2009-02	41	GP-090080	0112		Corrections of RED HOT rate matching	8.0.0	8.1.0
2009-02	41	GP-090156	0114		Correction to DBS-10, DBS-11 and DBS-12 mapping on	8.0.0	8.1.0
					a burst		
2009-02	41	GP-090413	0115	1	Correction to DAS11, DAS12, DBS9, and DBS10	8.0.0	8.1.0
					punctured block count and DBS10 RTTI USF bit count		
2009-02	41	GP-090415	0116	1	PAN bit swapping order clarification	8.0.0	8.1.0
2009-05	42	GP-090988	0118	2	New USF code words for EGPRS2-A	8.1.0	8.2.0
2009-05	42	GP-090661	0122		Clarification on puncturing loop parameters calculation	8.1.0	8.2.0
2009-05	42	GP-090663	0124		Miscellaneous corrections for EGPRS2	8.1.0	8.2.0
2009-05	42	GP-090955	0125	1	Corrections to EGPRS2 DL puncturing and UBS-6 coding	8.1.0	8.2.0
2009-09	43	GP-091138	0128		Various minor corrections on EGPRS2	8.2.0	8.3.0
2009-09	43	GP-091143	0130		Correction of UBS-12 puncturing with PAN included	8.2.0	8.3.0
2009-12	44				Version for Release 9	8.3.0	9.0.0
2011-03	49				Version for Release 10	9.0.0	10.0.0
2012-09	55				Version for Release 11	10.0.0	11.0.0
2012-11	56	GP-121314	0131		TC RT: Uplink Reply Procedure	11.0.0	11.1.0
2013-11	60	GP-131125	0132	4	Extended TFI Addressing space for DLMC	11.1.0	12.0.0
2014-05	62	GP-140347	0133	2	DLMC – Introduction of CS-3 for PDAN	12.0.0	12.1.0
2014-05	62	GP-140388	0134	1	DLMC - Correction of eTFI usage in PACCH	12.0.0	12.1.0
2015-11	68	GP-151180	0135	2	Introduction of EC-EGPRS	12.1.0	13.0.0
2016-02	69	GP-160209	0136	3	Corrections for EC-EGPRS	13.0.0	13.1.0

Change history							
Date	Meeting	TDoc	CR	Rev	Cat	Subject/Comment	New version
2016-05	RP-70	GP-160467	0139	3	В	Introduction of Radio Frequency Colour Code, clarifications and miscellaneous corrections to EC-GSM-IoT (including name change)	13.2.0
2016-09	RP-73	RP-161392	0141	2	F	Miscellaneous corrections for EC-GSM-IoT	13.3.0
2016-12	RP-74	RP-162066	0142	-	F	Introduction of Alternative Mappings for Higher Coverage Classes with 2 PDCHs	14.0.0
2017-03	RP-75	RP-170061	0143	1	В	Introduction of compact burst mapping	14.1.0
2017-03	RP-75	RP-170060	0144	2	В	Introduction of Extended Access Burst for connectionless Multilateration Positioning	14.1.0
2017-06	RP-76	RP-170924	0145	5	В	Introduction of uplink coverage class CC5 for UL MCL improvement	14.2.0
2017-06	RP-76	RP-170928	0148	1	Α	Correction of EC-PACCH mapping onto PDCHs	14.2.0
2018-06	RP-80					Version for Release 15	15.0.0
2018-09	RP-81	RP-181593	0149	-	В	Introduction of EC-PICH	15.1.0
2019-03	RP-83	RP-190061	0152	-	Α	Correction to EC-SCH cyclic shift operation	15.2.0
2020-07	RP-88e	-	-	-	-	Upgrade to Rel-16 version without technical change	16.0.0

### History

Document history						
V16.0.0	September 2020	Publication				