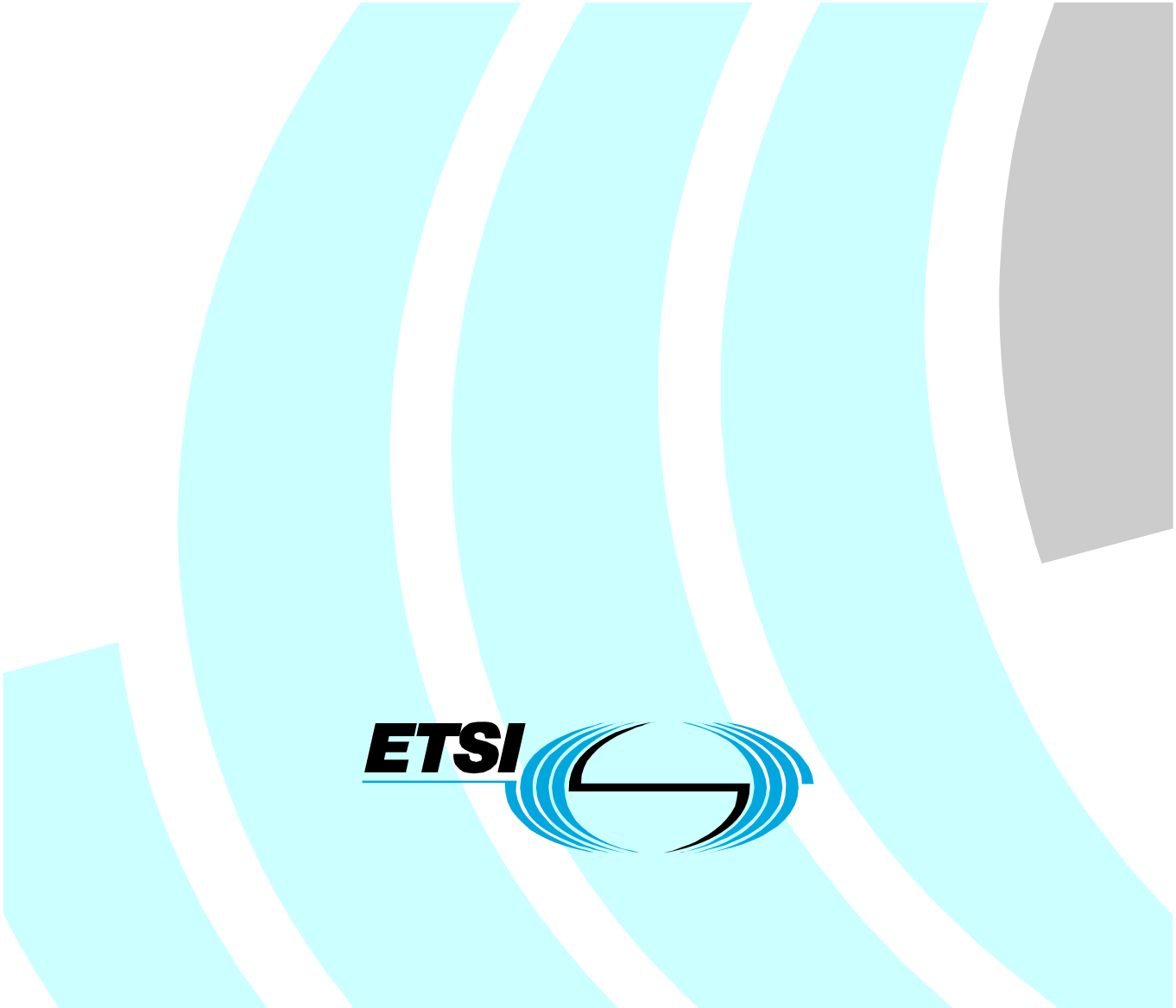
ETSI TS 101 376-5-3 V2.3.1 (2008-07)

Technical Specification

GEO-Mobile Radio Interface Specifications (Release 2);
General Packet Radio Service;
Part 5: Radio interface physical layer specifications;
Sub-part 3: Channel Coding;
GMPRS-1 05.003



Reference
RTS/SES-00303-5-3

Keywords
coding, GMPRS, GMR, GPRS, GSM, GSO, interface, MES, mobile, MSS, radio, satellite, S-PCN

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

Contents

Intellectual Property Rights

IPRs essential or potentially essential to the present document 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 (http://webapp.etsi.org/IPR/home.asp).

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.

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Satellite Earth Stations and Systems (SES).

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

Version 2.m.n

where:

- the third digit (n) is incremented when editorial only changes have been incorporated in the specification;
- the second digit (m) is incremented for all other types of changes, i.e. technical enhancements, corrections, updates, etc.

The present document is part 5, sub-part 3 of a multi-part deliverable c covering the GEO-Mobile Radio Interface Specifications (Release 2); General Packet Radio Service, as identified below:

Part 1: "General specifications";

Part 2: "Service specifications";

Part 3: "Network specifications";

Part 4: "Radio interface protocol specifications";

Part 5: "Radio interface physical layer specifications":

Sub-part 1: "Physical Layer on the Radio Path: General Description";

Sub-part 2: "Multiplexing and Multiple Access; Stage 2 Service Description";

Sub-part 3: "Channel Coding";

Sub-part 4: "Modulation";

Sub-part 5: "Radio Transmission and Reception";

Sub-part 6: "Radio Subsystem Link Control";

Sub-part 7: "Radio Subsystem Synchronization";

Part 6: "Speech coding specifications";

Part 7: "Terminal adaptor specifications".

Introduction

GMR stands for GEO (Geostationary Earth Orbit) Mobile Radio interface, which is used for mobile satellite services (MSS) utilizing geostationary satellite(s). GMR is derived from the terrestrial digital cellular standard GSM and supports access to GSM core networks.

The present document is part of the GMR Release 2 specifications. Release 2 specifications are identified in the title and can also be identified by the version number:

- Release 1 specifications have a GMR-1 prefix in the title and a version number starting with "1" (V1.x.x.).
- Release 2 specifications have a GMPRS-1 prefix in the title and a version number starting with "2" (V2.x.x.).

The GMR release 1 specifications introduce the GEO-Mobile Radio interface specifications for circuit mode mobile satellite services (MSS) utilizing geostationary satellite(s). GMR release 1 is derived from the terrestrial digital cellular standard GSM (phase 2) and it supports access to GSM core networks.

The GMR release 2 specifications add packet mode services to GMR release 1. The GMR release 2 specifications introduce the GEO-Mobile Packet Radio Service (GMPRS). GMPRS is derived from the terrestrial digital cellular standard GPRS (included in GSM Phase 2+) and it supports access to GSM/GPRS core networks.

Due to the differences between terrestrial and satellite channels, some modifications to the GSM standard are necessary. Some GSM specifications are directly applicable, whereas others are applicable with modifications. Similarly, some GSM specifications do not apply, while some GMR specifications have no corresponding GSM specification.

Since GMR is derived from GSM, the organization of the GMR specifications closely follows that of GSM. The GMR numbers have been designed to correspond to the GSM numbering system. All GMR specifications are allocated a unique GMR number. This GMR number has a different prefix for Release 2 specifications as follows:

- Release 1: GMR-n xx.zyy.
- Release 2: GMPRS-n xx.zyy.

where:

- xx.0yy (z = 0) is used for GMR specifications that have a corresponding GSM specification. In this case, the numbers xx and yy correspond to the GSM numbering scheme.
- xx.2yy (z = 2) is used for GMR specifications that do not correspond to a GSM specification. In this case, only the number xx corresponds to the GSM numbering scheme and the number yy is allocated by GMR.
- n denotes the first (n = 1) or second (n = 2) family of GMR specifications.

A GMR system is defined by the combination of a family of GMR specifications and GSM specifications as follows:

- If a GMR specification exists it takes precedence over the corresponding GSM specification (if any). This precedence rule applies to any references in the corresponding GSM specifications.

NOTE: Any references to GSM specifications within the GMR specifications are not subject to this precedence rule. For example, a GMR specification may contain specific references to the corresponding GSM specification.

- If a GMR specification does not exist, the corresponding GSM specification may or may not apply. The applicability of the GSM specifications is defined in GMPRS-1 01.201 [2].

1 Scope

The present document specifies the data blocks given to the encryption unit and the mapping onto the free bits of a burst. It includes the specifications for encoding, reordering, interleaving, and detailed mapping onto the burst. It does not specify the channel decoding method. The definition is given for each kind of logical channel, starting with the data provided to the channel encoder by the speech coder, the data terminal equipment, or the controller of the Mobile Earth Station (MES).

2 References

References are either specific (identified by date of publication and/or edition number or version number) or non-specific.

- For a specific reference, subsequent revisions do not apply.
- Non-specific reference may be made only to a complete document or a part thereof and only in the following cases:
  - if it is accepted that it will be possible to use all future changes of the referenced document for the purposes of the referring document;
  - for informative references.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

For online referenced documents, information sufficient to identify and locate the source shall be provided. Preferably, the primary source of the referenced document should be cited, in order to ensure traceability. Furthermore, the reference should, as far as possible, remain valid for the expected life of the document. The reference shall include the method of access to the referenced document and the full network address, with the same punctuation and use of upper case and lower case letters.

NOTE: While any hyperlinks included in this clause were valid at the time of publication ETSI cannot guarantee their long term validity.

2.1 Normative references

The following referenced documents are indispensable for the application of the present document. For dated references, only the edition cited applies. For non-specific references, the latest edition of the referenced document (including any amendments) applies.

[1] ETSI TS 101 376-1-1: "GEO-Mobile Radio Interface Specifications (Release 2); General Packet Radio Service; Part 1: General specifications; Sub-part 1: Abbreviations and acronyms; GMPRS-1 01.004".

[2] ETSI TS 101 376-1-2: "GEO-Mobile Radio Interface Specifications (Release 2); General Packet Radio Service; Part 1: General specifications; Sub-part 2: Introduction to the GMR-1 family; GMPRS-1 01.201".

[3] ETSI TS 101 376-5-3 (V1.2.1): "GEO-Mobile Radio Interface Specifications; Part 5: Radio interface physical layer specifications; Sub-part 3: Channel Coding; GMR-1 05.003".

NOTE: This is a reference to a GMR-1 Release 1 specification. See the introduction for more details.

[4] ETSI TS 101 376-4-8: "GEO-Mobile Radio Interface Specifications (Release 2); General Packet Radio Service; Part 4: Radio interface protocol specifications; Sub-part 8: Mobile Radio Interface Layer 3 Specifications; GMPRS-1 04.008".

[5] ETSI TS 101 376-4-12: "GEO-Mobile Radio Interface Specifications (Release 2); General Packet Radio Service; Part 4: Radio interface protocol specifications; Sub-part 12: Mobile Earth Station (MES) - Base Station System (BSS) interface; Radio Link Control/Medium Access Control (RLC/MAC) protocol GMPRS-1 04.060".

2.2 Informative references

The following referenced documents are not essential to the use of the present document but they assist the user with regard to a particular subject area**.** For non-specific references, the latest version of the referenced document (including any amendments) applies.

Not applicable.

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in GMPRS-1 01.201 [2] apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in GMPRS-1 01.004 [1] apply.

4 General

4.1 General organization

Same as clause 4.1 in GMR-1 05.003 [3].

4.2 Naming convention

Same as clause 4.2 in GMR-1 05.003 [3].

Table 4.1: Void

4.3 Parity checking

Same as clause 4.3 in GMR-1 05.003 [3], except table 4.2.

Table 4.2 indicates the CRC polynomials used in GMR-1 channels.

Table 4.2: CRC polynomials used in GMR-1

Channel	$g_8(D)$	$g_{12}(D)$	$g_{16}(D)$
BCCH			X
PCH			X
AGCH			X
RACH	X	X	
CBCH			X
SDCCH			X
SACCH			X
FACCH3			X
FACCH6			X
FACCH9			X
TACCH			X
GBCH			X
PDCH	X		X
PRACH	X		
Downlink PDCH (5,12) Extended PUI	X		

4.4 Convolutional coding

4.4.1 Convolutional encoding (all channels except TCH3)

Same as clause 4.4.1 in GMR-1 05.003 [3] with the following additions.

4.4.1.1 Rate 1/2 convolutional code

Same as clause 4.4.1.1 in GMR-1 05.003 [3].

4.4.1.2 Rate 1/4 convolutional code

Same as clause 4.4.1.2 in GMR-1 05.003 [3].

4.4.1.3 Rate 1/3 convolutional code

Same as clause 4.4.1.3 in GMR-1 05.003 [3].

4.4.1.4 Rate 1/5 convolutional code

Same as clause 4.4.1.4 in GMR-1 05.003 [3].

4.4.1.5 Rate 1/2 convolutional code (constraint length 9)

The Rate 1/2 convolutional code of constraint length 9 is defined by the following generator polynomials:

$$g_0(D) = 1 + D^2 + D^3 + D^4 + D^8;$$

$$g_1(D) = 1 + D + D^2 + D^3 + D^5 + D^7 + D^8.$$

The input data block {u(0), u(1), ..., u(K-1)} to be encoded is first extended with tail bits so that u(k) = 0 for k =K, K+1,…, K+7. The coded bits are then defined by the following set of linear equations:

For k = 0, 1, …, K+7

$c(2k) = u(k) \oplus u(k-2) \oplus u(k-3)) \oplus u(k-4) \oplus u(k-8)$

$c(2k+1) = u(k) \oplus u(k-1) \oplus u(k-2) \oplus u(k-3)) \oplus u(k-5) \oplus u(k-7) \oplus u(k-8)$

This results in a block of coded bits *{c(0), c(1), c(2), ..., c(2K+15)}*.

4.4.2 Convolutional encoding for TCH3

Same as clause 4.4.2 in GMR-1 05.003 [3].

4.4.3 Viterbi decoder for TCH3

Same as clause 4.4.3 in GMR-1 05.003 [3].

4.4.4 Convolutional encoding for Extended PUI

Same as clause 4.4.2 in GMR-1 05.003 [3], except that a rate 1/4 convolutional code of constraint length 6 is used. The code is defined by the following generator polynomials:

$$g_0(D) = 1 + D^2 + D^5$$

$$g_1(D) = 1 + D^2 + D^3 + D^5$$

$$g_2(D) = 1 + D + D^3 + D^4 + D^5$$

$$g_3(D) = 1 + D + D^2 + D^3 + D^4 + D^5$$

The encoder is initialized with bits {u(K-1), u(K-2), …, u(K-5)} from the input data block {u(0), u(1), …, u(K-1)} to be encoded; bit u(K-1) is placed in the register D1 and bit u(K-6) is placed in the register D5. The coded bits are then defined by the following set of linear equation:

For k = 0, 1, …, K-1

$$c(4k) = u(k) \oplus u(k-2) \oplus u(k-5)$$

$$c(4k+1) = u(k) \oplus u(k-2) \oplus u(k-3) \oplus u(k-5)$$

$$c(4k+2) = u(k) \oplus u(k-1) \oplus u(k-3) \oplus u(k-4) \oplus u(k-5)$$

$$c(4k+3) = u(k) \oplus u(k-1) \oplus u(k-2) \oplus u(k-3) \oplus u(k-4) \oplus u(k-5)$$

This results in a block of coded bits {c(0), c(1), …c(4K-1)}.

4.5 Puncturing and repetition

The number of available free bits on a burst may not equal the number of coded bits output by the convolutional encoder. In this case, selected coded bits are either punctured (not processed for transmission) or repeated (transmitted twice) as needed to match the coded output to the available payload. The coded bits to be punctured and/or repeated are specified by channel-dependent puncturing and repetition masks. These masks take the form of an $n \times L$ integer array, in which the i^{th} row applies to the coded bits produced by the $g_i(D)$ generator polynomial, $i = 0, \ldots, n\text{-}1$, and each entry specifies the number of times that the corresponding coded bit is to be transmitted. The parameter L denotes the period of the pattern. If the period is less than the total number of encoder input (information plus tail bits), the mask is reapplied on a periodic basis. If the number of encoder input is not divisible by L, the mask applies on all the encoder input and stops at the end of the encoder input. In some instances, prefix and suffix masks are applied at the beginning and end of the burst, respectively, to facilitate the rate matching.

The puncturing and repetition masks used in GMR-1 for the Rate 1/2, Rate 1/3, and Rate 1/5 convolutional code (K = 5) are listed in tables 4.3, 4.4 and 4.5 respectively. The puncturing masks used for Rate 1/2 convolutional code with constraint length K = 7 are listed in table 4.6. The puncturing masks used for Rate 1/2 convolutional code with constraint length K = 9 are listed in table 4.7. The puncturing masks used for Rate 1/4 convolutional code with constraint length K = 6 are listed in table 4.7a. The identifier P(r;L) denotes the preferred puncturing mask used for circuit-switched services in which r coded bits are punctured every L input bits to the convolutional encoder. The time-reversed version of this mask is denoted by P*(r;L).

The puncturing/repetition masks used for convolutionally-encoded packet-switched services are formed as concatenations of the five basic masks denoted A, B, C, D and E in table 4.3. This allows complex puncturing and repetition patterns to be specified by short mathematical descriptions. For example, the composite mask $P = A(BC)^{10}D$ denotes the puncture pattern in which mask A is applied at the beginning of the burst; the combination of mask B followed by mask C is applied 10 times during the middle of the burst; and, finally, mask D is applied at the end of the burst.

Table 4.3: GMR-1 puncturing and repetition masks for the rate 1/2 convolutional code (K = 5)

Identifier	Mask	Remark
P(2;3)	$\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 3/4.
P(2;5)	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 5/8.
P*(2;5)	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix}$	Time-reversal of the puncturing mask P(2;5).
P(3;11)	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 11/19.
P(4;12)	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 12/20.
P*(4;12)	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$	Time-reversal of the puncturing mask P(4;12).
P(1;2)	$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 2/3.
A	$\begin{bmatrix} 1111 \\ 1111 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 1/2 (no puncturing).
B	$\begin{bmatrix} 1111 \\ 0111 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 4/7.
C	$\begin{bmatrix} 1111 \\ 0101 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 2/3.
D	$\begin{bmatrix} 0101 \\ 1011 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 4/5.
E	$\begin{bmatrix} 1111 \\ 2111 \end{bmatrix}$	Repetition mask that, if applied repetitively, produces effective code rate = 4/9.

Table 4.4: GMR-1 puncturing masks for the rate 1/3 convolutional code (K = 5)

Identifier	Mask	Remark
P(1;6)	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$	Puncturing mask that, if applied repetitively for NT6 burst punctures 24 bits giving an effective code rate = 0,3523.
P(2;5)	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 5/13.
P(1;5)	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 5/14.

Identifier	Mask	Remark
P*(1;5)	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$	Time-reversal of the puncturing mask P(1;5).

Table 4.5: GMR-1 puncturing masks for the rate 1/5 convolutional code (K = 5)

Identifier	Mask	Remark
P(2;3)	$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 3/13.
P(5;3)	$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 3/10.
P*(5;3)	$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$	Time-reversal of the puncturing mask P(5;3).

Table 4.6: GMR-1 puncturing masks for the rate 1/2 convolutional code (K = 7)

Identifier	Mask	Remark
P(2;3)	$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 3/4.
P(4;10)	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 5/8.
P(5;12)	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 12/19.
P(1;16)	$\begin{bmatrix} 1 & 1 & \dots & \dots & 1 \\ 0 & 1 & \dots & \dots & 1 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 16/31.
P(1;48)	$\begin{bmatrix} 1 & 1 & \dots & \dots & 1 \\ 0 & 1 & \dots & \dots & 1 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 48/95.
P(1;84)	$\begin{bmatrix} 1 & 1 & \dots & \dots & 1 \\ 0 & 1 & \dots & \dots & 1 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 84/167.
P(1;152)	$\begin{bmatrix} 1 & 1 & \dots & \dots & 1 \\ 0 & 1 & \dots & \dots & 1 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 152/303.

Table 4.7: GMR-1 puncturing masks for the rate 1/2 convolutional code (K = 9)

Identifier	Mask	Remark
P(1;3)	$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 3/5.

Identifier	Mask	Remark
P(4;7)	$\begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 7/10.
P(3;4)	$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 4/5.

Table 4.7a: GMR-1 puncturing masks for the rate 1/4 convolutional code (K = 6)

Identifier	Mask	Remark
P(4;5)	$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix}$	Puncturing mask that, if applied repetitively, produces effective code rate = 5/16

4.6 Golay encoding

Same as clause 4.6 in GMR-1 05.003 [3].

4.7 Reed-Solomon encoding

Same as clause 4.7 in GMR-1 05.003 [3].

4.7.1 Encoder

Same as clause 4.7.1 in GMR-1 05.003 [3].

4.7.2 Galois field arithmetics

Same as clause 4.7.2 in GMR-1 05.003 [3].

4.7.3 Encoder feedback register operation

Same as clause 4.7.3 in GMR-1 05.003 [3].

4.8 Interleaving

Intraburst and interburst interleaving schemes are based on block interleaving methods with pseudorandom permutations and are channel dependent.

4.8.1 Intraburst interleaving

Intraburst interleaving is performed by mapping the block of the coded bits {c(0), c(1), …, c(M-1)} into a N × 8 matrix by rows, interchanging the columns using the pseudorandom permutation factor of 5, and reading out blocks of data by columns. Matrix dimension N is channel dependent and $N = \lceil M/8 \rceil$.

When columns are interchanged the index of the matrix element (i, j) changes to (i, j_p), where $j_p = (j \times 5) \bmod 8$.

When the data is read out by columns, note there may be only N-1 elements in certain columns.

4.8.2 Interburst interleaving

Same as clause 4.8.2 in GMR-1 05.003 [3].

4.9 Scrambling

Same as clause 4.9 in GMR-1 05.003 [3].

4.10 LDPC Codes

4.10.1 General Operations

LDPC encoder systematically encodes an input block of size k_{ldpc}, $\boldsymbol{i} = (i_0, i_1, ..., i_{k_{ldpc}-1})$ onto a codeword of size n_{ldpc}, $\boldsymbol{c} = (i_0, i_1, ..., i_{k_{ldpc}-1}, p_0, p_1, ...p_{n_{ldpc}-k_{ldpc}-1})$ In order to match the exact burst structure shortening, repeating and/or puncturing can be applied. The number of shortened, repeated or punctured bits will be denoted by *XS*, *XR* and *XP*, respectively. Each procedure is described as follows.

Shortening

Set the first *XS* bits in the input block to 0 before encoding. Omit these bits from the resulting codeword before transmission. The transmission of the codeword starts in the given order from i_{XS} and ends with $p_{n_{ldpc}-k_{ldpc}-1}$.

Repeating

Repeat the transmission of the first *XR* transmitted bits, i.e. transmit:

$$i_{XS}, i_{XS}, i_{XS+1}, i_{XS+1}, ..., i_{XS+XR-1}, i_{XS+XR-1}, i_{XS+XR}, i_{XS+XR+1}, ..., p_{n_{ldpc}-k_{ldpc}-1}$$

Puncturing

For PNB2(5,3) rate 9/10, do not transmit the following *XP* systematic bits:

$$i_{k_{ldpc}-4XP+3}, i_{k_{ldpc}-4(XP-1)+3}, i_{k_{ldpc}-4(XP-2)+3}, ..., i_{k_{ldpc}-1}$$

For PNB2(5,12) rate 9/10, do not transmit the following *XP* systematic bits:

$$i_{k_{ldpc}-4XP}, i_{k_{ldpc}-4(XP-1)}, i_{k_{ldpc}-4(XP-2)}, ..., i_{k_{ldpc}-4}$$

For all other code rates, do not transmit the following *XP* parity bits:

$$p_0, p_4, p_8, ..., p_{4(XP-1)}$$

As a result, the effective rate of the LDPC code is $R_{eff} = \dfrac{k_{ldpc} - XS}{n_{ldpc} - XS + XR - XP}$. The block sizes and modulation types of the LDPC codes for PNB2(5,12) and PNB2(5,3) described in the present document are shown in tables 4.8 and 4.10, respectively.

Table 4.8: PNB2(5,12) LDPC Code Block Sizes

Code	Modulation	k_{ldpc}	n_{ldpc}	XS	XR	XP (Notes 1 and 2)	R_{eff}
2/3 C1	π/4 QPSK	2 960	4 440	0	0	0 or 96	0,667 or 0,681
2/3 C2	16APSK	5 920	8 880	0	0	0 or 192	0,667 or 0,681
3/4	32APSK	8 352	11 136	40	4	240	0,765
4/5 C1	π/4 QPSK	3 552	4 440	0	0	0 or 96	0,800 or 0,818
4/5 C2	16APSK	7 104	8 880	0	0	0 or 192	0,800 or 0,818
4/5 C3	32APSK	8 880	11 100	0	0	240	0,818
9/10 C1	π/4 QPSK	4 032	4 480	40	0	0 or 96	0,899 or 0,919
9/10 C2	16APSK	7 992	8 880	0	0	0	0,900
1/2	π/4 QPSK	2 232	4 464	24	0	0 or 96	0,497 or 0,508
NOTE 1: XP = 0 means return link only and no puncturing. NOTE 2: XP = 96, 192 or 240 means forward link only and puncturing.							

4.10.2 LDPC Encoding

The task of the encoder is to determine $n_{ldpc} - k_{ldpc}$ parity bits $(p_0, p_1, ..., p_{n_{ldpc}-k_{ldpc}-1})$ for every block of k_{ldpc} input bits, $(i_0, i_1, ..., i_{k_{ldpc}-1})$. The procedure is follows:

- Initialize $p_0 = p_1 = p_2 = ... = p_{n_{ldpc}-k_{ldpc}-1} = 0$.
- Accumulate the first input bit, i_0, at parity bit addresses specified in the first row of tables A.1 through A.9, and tables A.10 through A.18.

EXAMPLE: For code 2/3 C1 (see table A.1):

$$p_{1025} = p_{1025} \oplus i_0$$

$$p_{1277} = p_{1277} \oplus i_0$$

$$p_{1247} = p_{1247} \oplus i_0$$

$$p_{1429} = p_{1429} \oplus i_0$$

$$p_{838} = p_{838} \oplus i_0$$

$$p_{836} = p_{836} \oplus i_0$$

$$p_{93} = p_{93} \oplus i_0$$

$$p_{104} = p_{104} \oplus i_0$$

(All additions are modulo 2.)

- For the next M-1 input bits, $i_m, m = 1,2,...,M-1$ accumulate i_m at parity bit addresses $\{x + m \bmod M \times q\} \bmod (n_{ldpc} - k_{ldpc})$ where x denotes the address of the parity bit accumulator corresponding to the first bit i_0, and M and q are code dependent constants specified in tables 4.9 and 4.11. Continuing with the example, $M = 74, q = 20$ for code 2/3 C1. So for example for input bit i_1, the following operations are performed:

$$p_{1045} = p_{1045} \oplus i_1$$

$$p_{1297} = p_{1297} \oplus i_1$$

$$p_{1267} = p_{1267} \oplus i_1$$

$$p_{1449} = p_{1449} \oplus i_1$$

$$p_{858} = p_{858} \oplus i_1$$

$$p_{856} = p_{856} \oplus i_1$$

$$p_{113} = p_{113} \oplus i_1$$

$$p_{124} = p_{124} \oplus i_1$$

- For the (M+1)st input bit i_M, the addresses of the parity bit accumulators are given in the second row of the tables A.1 through A.9, and Tables A.10 through A.18. In a similar manner the addresses of the parity bit accumulators for the following M-1 input bits $i_m, m = M+1, M+2, ..., 2M-1$ are obtained using the formula $\{x + m \bmod M \times q\} \bmod (n_{ldpc} - k_{ldpc})$ where x denotes the address of the parity bit accumulator corresponding to the input bit i_M, i.e. the entries in the second row of the tables A.1 through A.9, and tables A.10 through A.18.
- In a similar manner, for every group of M new input bits, a new row from tables A.1 through A.9 and tables A.10 through A.18 are used to find the addresses of the parity bit accumulators.

After all of the input bits are exhausted, the final parity bits are obtained as follows:

- Sequentially perform the following operations starting with $i = 1$:

$$p_i = p_i \oplus p_{i-1}, \quad i = 1, 2, ..., n_{ldpc} - k_{ldpc} - 1$$

- Final content of p_i, $i = 0, 1, .., n_{ldpc} - k_{ldpc} - 1$ is equal to the parity bit p_i.

Table 4.9 M and q Values for PNB2(5,12) LDPC Codes

Code	M	q
2/3 C1	74	20
2/3 C2	74	40
3/4	87	32
4/5 C1	74	12
4/5 C2	74	24
4/5 C3	74	30
9/10 C1	64	7
9/10 C2	74	12
1/2	62	36

Parameters of the PNB2(5,3) LDPC codes are given in tables 4.10, 4.11 and A.10 through table A.18.

Table 4.10: PNB2(5,3) LDPC Code Block Sizes

Code	Modulation	k_{ldpc}	n_{ldpc}	XS	XR	XP	R_{eff}
2/3 C1	π/4 QPSK	640	960	8	6	0	0,6597
2/3 C2	16APSK	1 272	1 908	0	8	0	0,6638
3/4	32APSK	1 800	2 400	8	3	0	0,7482
4/5 C1	π/4 QPSK	760	950	0	8	0	0,7933
4/5 C2	16APSK	1 536	1 920	8	4	0	0,7974
4/5 C3	32APSK	1 920	2 400	8	3	0	0,7983
9/10 C1	π/4 QPSK	864	960	8	6	0	0,8935
9/10 C2	16APSK	1 728	1 920	8	4	0	0,8977
1/2	π/4 QPSK	488	976	0	0	18	0,5093

Table 4.11 M and q Values for PNB2(5,3) LDPC Codes

Code	M	q
2/3 C1	32	10
2/3 C2	53	12
3/4	50	12
4/5 C1	19	10
4/5 C2	32	12
4/5 C3	48	10
9/10 C1	16	6
9/10 C2	32	6
1/2	61	8

4.10.3 Bit Interleaver

For 16-APSK, and 32-APSK modulation formats, the output of the LDPC encoder is bit interleaved using a block interleaver. Data is serially written into the interleaver column-wise (from the top to the bottom), and serially read out row-wise. For PNB2(5,12) codes, the block interleaver has 2 220 rows when $XP = 0$ and 2 172 rows when $XP \neq 0$. For PNB2(5,3) codes, the block interleaver has 479 rows. For all the codes, the block interleaver has 4 columns for 16-APSK modulation and 5 columns for 32-APSK modulation.

For code 2/3 C2, bits 4i+1, 4i+3, 4i and 4i+2 of the interleaver output determine the i^{th} 16APSK symbol.

For code 4/5 C2, bits 4i+2, 4i+3, 4i and 4i+1 of the interleaver output determine the i^{th} 16APSK symbol.

For code 9/10 C2, bits 4i, 4i+2, 4i+1 and 4i+3 of the interleaver output determine the i^{th} 16APSK symbol.

For codes 3/4 and 4/5 C3, bits 5i+4, 5i+2, 5i+3, 5i+1 and 5i of the interleaver output determine the i^{th} 32APSK symbol.

For π/4 QPSK modulation, there is no interleaving. Bits 2i and 2i+1 of the LDPC encoder determines the i^{th} π/4 QPSK symbol.

5 Traffic channels

5.1 Traffic channel-3 (TCH3)

Same as clause 5.1 in GMR-1 05.003 [3].

5.1.1 Channel coding

Same as clause 5.1.1 in GMR-1 05.003 [3].

5.1.2 Interleaving

Same as clause 5.1.2 in GMR-1 05.003 [3].

5.1.3 Scrambling, multiplexing, and encryption

Same as clause 5.1.3 in GMR-1 05.003 [3].

5.2 Traffic channel-6 (TCH6)

5.2.1 Channel coding

5.2.1.1 Coding for 2,4 kbps fax

Same as clause 5.2.1.1 in GMR-1 05.003 [3].

5.2.1.2 Coding for 2,4 kbps data

Same as clause 5.2.1.2 in GMR-1 05.003 [3].

5.2.1.3 Coding for 4,8 kbps fax/data

Same as clause 5.2.1.2 in GMR-1 05.003 [3].

5.2.2 Interleaving

Same as clause 5.2.2 in GMR-1 05.003 [3].

5.2.3 Scrambling, multiplexing, and encryption

Same as clause 5.2.3 in GMR-1 05.003 [3].

5.3 Traffic channel-9 (TCH9)

5.3.1 Channel coding

5.3.1.1 Coding for 2,4 kbps fax

Same as clause 5.3.1.1 in GMR-1 05.003 [3].

5.3.1.2 Coding for 4,8 kbps fax

Same as clause 5.3.1.2 in GMR-1 05.003 [3].

5.3.1.3 Coding for 9,6 kbps fax/data

Same as clause 5.3.1.3 in GMR-1 05.003 [3].

5.3.2 Interleaving

Same as clause 5.3.2 in GMR-1 05.003 [3].

5.3.3 Scrambling, multiplexing, and encryption

Same as clause 5.3.3 in GMR-1 05.003 [3].

6 Control channels

6.1 Broadcast Control CHannel (BCCH)

Same as clause 6.1 in GMR-1 05.003 [3].

6.1.1 Channel coding

Same as clause 6.1.1 in GMR-1 05.003 [3].

6.1.2 Interleaving

Same as clause 6.1.2 in GMR-1 05.003 [3].

6.1.3 Scrambling and multiplexing

Same as clause 6.1.3 in GMR-1 05.003 [3].

6.2 Paging CHannel (PCH)

Same as clause 6.2 in GMR-1 05.003 [3].

6.2.1 Channel coding

Same as clause 6.2.1 in GMR-1 05.003 [3].

6.2.2 Interleaving

Same as clause 6.2.2 in GMR-1 05.003 [3].

6.2.3 Scrambling and multiplexing

Same as clause 6.2.3 in GMR-1 05.003 [3].

6.3 Access Grant CHannel (AGCH)

Same as clause 6.3 in GMR-1 05.003 [3].

6.3.1 Channel coding

Same as clause 6.3.1 in GMR-1 05.003 [3].

6.3.2 Interleaving

Same as clause 6.3.2 in GMR-1 05.003 [3].

6.4 Broadcast Alerting CHannel (BACH)

Same as clause 6.4 in GMR-1 05.003 [3].

6.4.1 Channel coding

Same as clause 6.4.1 in GMR-1 05.003 [3].

6.5 Random Access CHannel (RACH)

Same as clause 6.5 in GMR-1 05.003 [3].

6.5.1 Channel coding

Same as clause 6.5.1 in GMR-1 05.003 [3].

6.5.2 Interleaving

Same as clause 6.5.2 in GMR-1 05.003 [3].

6.5.3 Scrambling and multiplexing

Same as clause 6.5.3 in GMR-1 05.003 [3].

6.6 Cell Broadcast CHannel (CBCH)

Same as clause 6.6 in GMR-1 05.003 [3].

6.6.1 Channel coding

Same as clause 6.6.1 in GMR-1 05.003 [3].

6.6.2 Interleaving

Same as clause 6.6.2 in GMR-1 05.003 [3].

6.7 Standalone Dedicated Control CHannel (SDCCH)

Same as clause 6.7 in GMR-1 05.003 [3].

6.7.1 Channel coding

Same as clause 6.7.1 in GMR-1 05.003 [3].

6.7.2 Interleaving

Same as clause 6.7.2 in GMR-1 05.003 [3].

6.7.3 Scrambling, multiplexing, and encryption

Same as clause 6.7.3 in GMR-1 05.003 [3].

6.8 Slow Associated Control CHannel (SACCH)

Same as clause 6.8 in GMR-1 05.003 [3].

6.8.1 Channel coding

Same as clause 6.8.1 in GMR-1 05.003 [3].

6.8.2 Interleaving

Same as clause 6.8.2 in GMR-1 05.003 [3].

6.9 Fast Associated Control CHannel-3 (FACCH3)

Same as clause 6.9 in GMR-1 05.003 [3].

6.9.1 Channel coding

Same as clause 6.9.1 in GMR-1 05.003 [3].

6.9.2 Interleaving

Same as clause 6.9.2 in GMR-1 05.003 [3].

6.9.3 Scrambling, multiplexing, and encryption

Same as clause 6.9.3 in GMR-1 05.003 [3].

6.10 Fast Associated Control CHannel-6 (FACCH6)

Same as clause 6.10 in GMR-1 05.003 [3].

6.10.1 Channel coding

Same as clause 6.10.1 in GMR-1 05.003 [3].

6.10.2 Interleaving

Same as clause 6.10.2 in GMR-1 05.003 [3].

6.10.3 Scrambling, multiplexing, and encryption

Same as clause 6.10.3 in GMR-1 05.003 [3].

6.11 Fast Associated Control CHannel-9 (FACCH9)

Same as clause 6.11 in GMR-1 05.003 [3].

6.11.1 Channel coding

Same as clause 6.11.1 in GMR-1 05.003 [3].

6.11.2 Interleaving

Same as clause 6.11.2 in GMR-1 05.003 [3].

6.11.3 Scrambling, multiplexing, and encryption

Same as clause 6.11.3 in GMR-1 05.003 [3].

6.12 Terminal-to-terminal Associated Control CHannel (TACCH)

Same as clause 6.12 in GMR-1 05.003 [3].

6.12.1 TACCH channel coding

Same as clause 6.12.1 in GMR-1 05.003 [3].

6.12.2 Interleaving

Same as clause 6.12.2 in GMR-1 05.003 [3].

6.12.3 Scrambling and multiplexing

Same as clause 6.12.3 in GMR-1 05.003 [3].

6.12.4 PHYsical (PHY) header for TACCH

Same as clause 6.12.4 in GMR-1 05.003 [3].

6.13 GPS Broadcast CHannel (GBCH)

Same as clause 6.13 in GMR-1 05.003 [3].

6.13.1 Channel coding

Same as clause 6.13.1 in GMR-1 05.003 [3].

6.13.2 Interleaving

Same as clause 6.13.2 in GMR-1 05.003 [3].

6.13.3 Scrambling and multiplexing

Same as clause 6.13.3 in GMR-1 05.003 [3].

7 Logical channel multiplexing

7.1 SACCH multiplexing

Same as clause 7.1 in GMR-1 05.003 [3].

7.2 Status field

Same as clause 7.2 in GMR-1 05.003 [3].

7.2.1 Power control field

Same as clause 7.2.1 in GMR-1 05.003 [3].

7.2.2 Comfort noise field

Same as clause 7.2.2 in GMR-1 05.003 [3] except deleting the last sentence.

7.3 Status field with NTN bursts

7.3.1 Status field with NT6 and NT9 bursts

Same as clause 7.3.1 in GMR-1 05.003 [3].

7.3.2 Status field with NT3 bursts

7.3.2.1 Status field with NT3 bursts for encoded speech

Same as clause 7.3.2.1 in GMR-1 05.003 [3].

7.3.2.2 Status field with NT3 bursts for FACCH

Same as clause 7.3.2.2 in GMR-1 05.003 [3].

7.3.3 Status field with Keep-Alive Bursts (KAB)

Same as clause 7.3.3 in GMR-1 05.003 [3].

8 Encryption

Same as clause 8 in GMR-1 05.003 [3].

9 Packet Switched Channels

9.1 Packet Data Traffic Channels

Several types of bursts are used by the various logical channels in the downlink and uplink Packet Data CHannels (PDCHs). The Packet Normal Bursts, PNB(m,n), are characterized by their transmission rate in multiples, m, of the basic transmission rate of 23,4 ksps, and their duration in multiples, n, of the timeslot of 40/24 ms. For example, PNB(4,3) is a burst transmitted at m = 4 times the basic rate (93,6 ksps transmission rate occupying a nominal bandwidth of 125 kHz) with a nominal duration n = 3 timeslots or 5 ms.

Bursts transmitted over the downlink and uplink PDCHs consist of two parts:

- PUblic Information (PUI).
- PRivate Information (PRI).

PRI is also termed payload. The PRI comprises N message bits d_0 to $d_{N-1,}$ as defined in clause 10.1 in GMPRS-1 04.060 [5]. PUI is also called burst header, or physical layer header (hereafter referred to as PUI). The coding for the PUI and the various burst formats are described in the following clauses.

9.1.1 PUblic Information (PUI)

For all packet data bursts, the PUI comprises 12 information bits b_0 to b_{11}, as defined in GMPRS-1 04.060 [5]. These 12 bits are encoded via the extended Golay (24,12) code as described in clause 4.6, where $u(i) = b_i$, i = 0,…, 11. The output from the Golay encoder is a block of 24 bits, {c(0),c(1), …, c(23)}. After Golay encoding, each encoded bit is repeated once to form the 48-bit coded PUI as {c(0), c(1), …, c(23), c(0), c(1), …, c(23)}. Note there are no interleaving, scrambling and encryption on PUI data.

9.1.1a Extended PUI

Forward link PNB2(5,12) -20 ms bursts contain an extended PUI. The 22 information bits of the extended PUI and the 12 bits of the PUI are encoded together using an 8-bit CRC, which is described in GMR-1 05.003 [3]. These 8 CRC bits and the 22 extended PUI bits form a block of length 30 bits, b_0 to b_{29}. These 30 bits are encoded using a rate 1/4 tail biting convolutional code, described in clause 4.4.4, that is then punctured to rate 5/16 (the puncturing pattern is given in table 4-8). The output is a block of 96 bits {c(0), c(1), …, c(95)}.

9.1.2 Void

9.1.3 Packet Normal Burst PNB(4,3)

The information rate supported by the PNB(4,3) depends on the channel coding used. Channel coding rates of approximately 1/2, 5/8, and 3/4 are defined for the PNB(4,3). Independent of channel coding rate, the PNB(4,3) provides a common payload for 792 coded bits.

9.1.3.1 Rate 1/2 convolutional coding

A 16-bit CRC is applied to the 376 message bits d_0 to d_{375} as specified in clause 4.3, where $d(i) = d_i$, i = 0, …, 375. An 8-bit masking function is applied to the block of 392 CRC-protected bits {u(0),…,u(391)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 16 CRC bits of the message with an XOR operation to give us {u'(0),…, u'(391)}, where u′(i) = u(i), i = 0,…, 383 and u′(i + 384) = u(i + 384) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 392 bits {u′(0),…, u′(391)} is delivered to the encoder.

The 392 bits {u′(0),…, u′(391)} are encoded via the rate 1/2 convolutional code. The code is defined by the generator polynomial specified in clause 4.4.2, where the constraint length K = 7 is used. Eight tail bits are used in order to accommodate K = 9 in the future for the possible enhancement. The encoding results in a block of 800 coded bits {b(0),…, b(799)}.

Puncturing is performed using the puncture mask P(1;48), as listed in table 4.6. After that, one bit in {b} (with index 769) needs to be depunctured. This action turns out that 8 bits in {b} with index 96k + 1 (k = 0,…, 7) are punctured out. The result is a block of 792 coded bits {c(0),…, c(791)}.

9.1.3.2 Rate 5/8 convolutional coding

A 16-bit CRC is applied to the 472 message bits d_0 to d_{471} as specified in clause 4.3, where $d(i) = d_i$, i = 0, …, 471. An 8-bit masking function is applied to the block of 488 CRC-protected bits {u(0),…,u(487)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 16 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(487)}, where u′(i) = u(i), i = 0, …, 479 and u′(i + 480) = u(i + 480) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 488 bits {u′(0),…, u′(487)} is delivered to the encoder.

The 488 bits {u′(0),…, u′(487)} are encoded via the rate 1/2 convolutional code. The code is defined by the generator polynomial specified in clause 4.4.2, where the constraint length K = 7 is used. Eight tail bits are used in order to accommodate K = 9 in the future for the possible enhancement. The encoding results in a block of 992 coded bits {b(0),…, b(991)}.

Puncturing is performed using the puncture mask P(4;10) as listed in table 4.6. Besides, one more bit in {b} (with index 991) needs to be punctured. This action turns out that 200 bits in {b} with index 20k + 1, 20k + 5, 20k + 9 (k = 0,…, 49), 20k + 17 (k = 0,…, 48) and 991 are punctured out. The result is a block of 792 coded bits {c(0),…, c(791)}.

9.1.3.3 Rate 3/4 convolutional coding

A 16-bit CRC is applied to the 568 message bits d_0 to d_{567} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 567. An 8-bit masking function is applied to the block of 584 CRC-protected bits {u(0),…, u(583)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 16 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(583)}, where u′(i) = u(i), i = 0,…, 575 and u′(i + 576) = u(i + 576) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 584 bits {u′(0),…, u′(583)} is delivered to the encoder.

The 584 bits {u′(0),…, u′(583)} are encoded via the rate 1/2 convolutional code. The code is defined by the generator polynomial specified in clause 4.4.2, where the constraint length K = 7 is used. Eight tail bits are used in order to accommodate K = 9 in the future for the possible enhancement. The encoding results in a block of 1 184 coded bits {b(0),…, b(1 183)}.

Puncturing is performed using the puncture mask P(2;3) as listed in table 4.6. After that, two bits in {b}with index 1 179 and 1 180 are depunctured. This action turns out that 392 bits in {b} with index 6k + 3 and 6k + 4 (k = 0,…, 195) are punctured out. The result is a block of 792 coded bits {c(0),…, c(791)}.

9.1.3.4 Interleaving

Intraburst interleaving is performed as described in clause 4.8.1.

9.1.3.5 Scrambling, multiplexing, and encryption

Scrambling is performed as described in clause 4.9. No encryption is performed.

9.1.4 Packet Normal Burst PNB(5,3)

The information rate supported by the PNB(5,3) depends on the channel coding used. Channel coding rates of approximately 1/2, 5/8 and 3/4 are defined for the PNB(5,3). Independent of channel coding rate, the PNB(5,3) provides a common payload for 1 002 coded bits.

9.1.4.1 Rate 1/2 convolutional coding

A 16-bit CRC is applied to the 480 message bits d_0 to d_{479} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 479. An 8-bit masking function is applied to the block of 496 CRC-protected bits {u(0),…, u(495)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 16 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(495)}, where u′(i) = u(i), i = 0, …, 487 and u′(i + 488) = u(i + 488) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 496 bits {u′(0),…, u′(495)} is delivered to the encoder.

The 496 bits {u′(0),…, u′(495)} are encoded via the rate 1/2 convolutional code. The code is defined by the generator polynomial specified in clause 4.4.2, where the constraint length K = 7 is used. Eight tail bits are used in order to accommodate K = 9 in the future for the possible enhancement. The encoding results in a block of 1 008 coded bits {b(0),…, b(1 007)}.

Puncturing is performed using the puncture mask P(1;84) as listed in table 4.6. This action turns out that 6 bits in {b} with index 168k + 1 (k = 0,…, 5) are punctured out. The result is a block of 1 002 coded bits {c(0),…, c(1 001)}.

9.1.4.2 Rate 5/8 convolutional coding

A 16-bit CRC is applied to the 608 message bits d_0 to d_{607} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 607. An 8-bit masking function is applied to the block of 624 CRC-protected bits {u(0),…,u(623)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 16 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(623)}, where u′(i) = u(i), i = 0,…, 615 and u′(i + 616) = u(i + 616) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 624 bits {u′(0),…, u′(623)} is delivered to the encoder.

The 624 bits {u′(0),…, u′(623)} are encoded via the rate 1/2 convolutional code. The code is defined by the generator polynomial specified in clause 4.4.2, where the constraint length K = 7 is used. Eight tail bits are used in order to accommodate K = 9 in the future for the possible enhancement. The encoding results in a block of 1 264 coded bits {b(0),…, b(1 263)}.

Puncturing is performed using the puncture mask P(5;12) as listed in table 4.6. After that, one bit in {b} with index 1 263 needs to be depunctured. This action turns out that 262 bits in {b} with index 24k + 3, 24k + 7 (k = 0,…, 52) and 24k + 15, 24k + 19, 24k + 23 (k = 0,…, 51) are punctured out. The result is a block of 1 002 coded bits {c(0),…, c(1 001)}.

9.1.4.3 Rate 3/4 convolutional coding

A 16-bit CRC is applied to the 728 message bits d_0 to d_{727} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 727. An 8-bit masking function is applied to the block of 744 CRC-protected bits {u(0),…, u(743)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 16 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(743)}, where u′(i) = u(i), i = 0,…, 735 and u′(i + 736) = u(i + 736) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 744 bits {u′(0),…, u′(743)} is delivered to the encoder.

The 744 bits {u′(0),…, u′(743)} are encoded via the rate 1/2 convolutional code. The code is defined by the generator polynomial specified in clause 4.4.2, where the constraint length K = 7 is used. Eight tail bits are used in order to accommodate K = 9 in the future for the possible enhancement. The encoding results in a block of 1 504 coded bits {b(0),…, b(1 503)}.

Puncturing is performed using the puncture mask P(2;3) as listed in table 4.6. Besides, one more bit in {b} with index 1 495 is punctured. This action turns out that 502 bits in {b} with index 6k + 3 (k = 0,…, 250), 6k + 4 (k = 0,…, 249) and 1 495 are punctured out. The result is a block of 1 002 coded bits {c(0),…, c(1 001)}.

9.1.4.4 Interleaving

Intraburst interleaving is performed as described in clause 4.8.1.

9.1.4.5 Scrambling, multiplexing, and encryption

Scrambling is performed as described in clause 4.9. No encryption is performed.

9.1.5 Void

9.1.6 Packet Normal Burst PNB(1,6)

The information rate supported by the PNB(1,6) depends on the channel coding used. Channel coding rates of approximately 6/10, 7/10 and 4/5 are defined for the PNB(1,6). Independent of channel coding rate, the PNB(1,6) provides a common payload for 366 coded bits.

9.1.6.1 Rate 3/5 convolutional coding

A 16-bit CRC is applied to the 192 message bits d_0 to d_{191} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 191. An 8-bit masking function is applied to the block of 208 CRC-protected bits {u(0),…,u(207)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 16 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(207)}, where u′(i) = u(i), i = 0,…, 199 and u′(i + 200) = u(i + 200) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 208 bits {u′(0),…, u′(207)} is delivered to the encoder.

The 208 bits {u′(0),…, u′(207)} are encoded via the rate 1/2 convolutional code. The code is defined by the generator polynomial specified in clause 4.4.1, where the constraint length K = 9 is used. Eight tail bits are added. The encoding results in a block of 432 coded bits {b(0),…, b(431)}.

Puncturing is performed using the puncture mask P(1;3), as listed in table 4.7. This action turns out that 72 bits in {b} with index 6k + 1 (k = 0,…, 71) are punctured out. After applying P(1:3), these bits will be depunctured: 37, 109, 181, 253, 325, 397. The result is a block of 366 coded bits {c(0),…,c(365)}.

9.1.6.2 Rate 7/10 convolutional coding

A 16-bit CRC is applied to the 232 message bits d_0 to d_{231}as specified in clause 4.3, where d(i) = d_i, i = 0, …, 231. An 8-bit masking function is applied to the block of 248 CRC-protected bits {u(0),…, u(247)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 16 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(247)}, where u′(i) = u(i), i = 0, …, 239 and u′(i + 239) = u(i + 239) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 248 bits {u′(0),…, u′(247)} is delivered to the encoder.

The 248 bits {u′(0),…, u′(247)} are encoded via the rate 1/2 convolutional code. The code is defined by the generator polynomial specified in clause 4.4.1, where the constraint length K = 9 is used. Eight tail bits are added. The encoding results in a block of 512 coded bits {b(0),…, b(511)}.

Puncturing is performed using the puncture mask P(4;7) as listed in table 4.7. This action turns out that 146 bits in {b} with index 14k, 14k + 5, (k = 0,…,36) and 14k + 9, 14k + 13 (k = 0,…, 35) are punctured out. The result is a block of 366 coded bits {c(0),…,c(365)}.

9.1.6.3 Rate 4/5 convolutional coding

A 16-bit CRC is applied to the 272 message bits d_0 to d_{271} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 271. An 8-bit masking function is applied to the block of 288 CRC-protected bits {u(0),…, u(287)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 16 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(287)}, where u′(i) = u(i), i = 0,…, 279 and u′(i + 280) = u(i + 270) ⊕ m(i), i = 0,…,7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 288 bits {u′(0),…, u′(287)} is delivered to the encoder.

The 288 bits {u′(0),…, u′(287)} are encoded via the rate 1/2 convolutional code. The code is defined by the generator polynomial specified in clause 4.4.1, where the constraint length K = 9 is used. Eight tail bits are added. The encoding results in a block of 592 coded bits {b(0),…, b(591)}.

Puncturing is performed using the puncture mask P(3;4) as listed in table 4.7. This action turns out that 222 bits in {b} with index 8k + 3, 8k + 4, 8k + 7 (k = 0,…, 73) are punctured out. After applying P(3:4), these bits also will be punctured: 0, 8, 576, 584. The result is a block of 366 coded bits {c(0),…, c(365)}.

9.1.6.4 Interleaving

Intraburst interleaving is performed as described in clause 4.8.1.

9.1.6.5 Scrambling, multiplexing, and encryption

Scrambling is performed as described in clause 4.9. No encryption is performed.

9.1.7 Packet Normal Burst PNB(2,6)

The information rate supported by the PNB(2,6) depends on the channel coding used. Channel coding rates of approximately 3/5, 7/10 and 4/5 are defined for the PNB(2,6). Independent of channel coding rate, the PNB(2,6) provides a common payload for 810 coded bits.

9.1.7.1 Rate 3/5 convolutional coding

A 16-bit CRC is applied to the 456 message bits d_0 to d_{455} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 455. An 8-bit masking function is applied to the block of 472 CRC-protected bits {u(0),…, u(471)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 16 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(471)}, where u′(i) = u(i), i = 0,…, 463 and u′(i + 464) = u(i + 464) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 472 bits {u′(0),…, u′(471)} is delivered to the encoder.

The 472 bits {u′(0),…, u′(471)} are encoded via the rate 1/2 convolutional code. The code is defined by the generator polynomial specified in clause 4.4.1, where the constraint length K = 9 is used. Eight tail bits are added. The encoding results in a block of 960 coded bits {b(0),…, b(959)}.

Puncturing is performed using the puncture mask P(1;3), as listed in table 4.7. This action turns out that 160 bits in {b} with index 6k + 1 (k = 0,…,159) are punctured out. After applying P(1:3) these bits will be depunctured: 73, 163, 253, 343, 433, 523, 613, 703, 793, 883. The result is a block of 810 coded bits {c(0),…, c(809)}.

9.1.7.2 Rate 7/10 convolutional coding

A 16-bit CRC is applied to the 544 message bits d_0 to d_{543} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 543. An 8-bit masking function is applied to the block of 560 CRC-protected bits {u(0),…,u(559)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 16 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(559)}, where u′(i) = u(i), i = 0, …, 551 and u′(i + 552) = u(i + 552) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 560 bits {u′(0),…, u′(559)} is delivered to the encoder.

The 560 bits {u′(0),…, u′(559)} are encoded via the rate 1/2 convolutional code. The code is defined by the generator polynomial specified in clause 4.4.1, where the constraint length K = 9 is used. Eight tail bits are added. The encoding results in a block of 1 136 coded bits {b(0),…, b(1 135)}.

Puncturing is performed using the puncture mask P(4;7) as listed in table 4.7. This action turns out that 325 bits in {b} with index 14k, (k = 0,…, 81) and 14k + 5, 14k + 9, 14k + 13 (k = 0,…, 80). After applying P(4;7), bit with index 2 will be punctured out. The result is a block of 810 coded bits {c(0),…, c(809)}.

9.1.7.3 Rate 4/5 convolutional coding

A 16-bit CRC is applied to the 624 message bits d_0 to d_{623} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 623. An 8-bit masking function is applied to the block of 640 CRC-protected bits {u(0),…, u(639)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 16 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(639)}, where u′(i) = u(i), i = 0,…, 631 and u′(i + 632) = u(i + 632) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 640 bits {u′(0),…, u′(639)} is delivered to the encoder.

The 640 bits {u′(0),…, u′(639)} are encoded via the rate 1/2 convolutional code. The code is defined by the generator polynomial specified in clause 4.4.1, where the constraint length K = 9 is used. Eight tail bits are added. The encoding results in a block of 1 296 coded bits {b(0),…, b(1 295)}.

Puncturing is performed using the puncture mask P(3;4) as listed in table 4.7. This action turns out that 486 bits in {b} with index 8k + 3, 8k + 4, 8k + 7 (k = 0,…, 161) are punctured out. The result is a block of 810 coded bits {c(0),…, c(809)}.

9.1.7.4 Interleaving

Intraburst interleaving is performed as described in clause 4.8.1.

9.1.7.5 Scrambling, multiplexing, and encryption

Scrambling is performed as described in clause 4.9. No encryption is performed.

9.1.8 Packet Normal Burst PNB2(5,12)

The information rate supported by the PNB2(5,12) depends on the modulation and the channel coding used. Modulations include 32APSK, 16APSK, and π/4QPSK. Approximate channel coding rates include 1/2, 2/3, 4/5, and 9/10. Punctured LDPC is used in the forward direction only to accommodate the Extended PUI.

9.1.8.1 π/4QPSK Rate 0,497 LDPC Coding

π/4QPSK Rate 0,497 is used in the return direction only.

A 8-bit CRC is applied to the 2 200 message bits d_0 to d_{2199} as specified in clause 4.3, where $d(i) = d_i$, i = 0, …, 2 199. An 8-bit masking function is applied to the block of 2 208 CRC-protected bits {u(0),…, u(2 207)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(2 207)}, where u′(i) = u(i), i = 0,…, 2 199 and u′(i + 2 200) = u(i + 2 200) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 2 208 bits {u′(0),…, u′(2 207)} is delivered to the encoder.

The 2 208 bits {u′(0),…, u′(2 207)} are encoded via the rate 0,497 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 4 440 coded bits {b(0),…, b(4 439)}.

9.1.8.2 π/4QPSK Rate 0,508 LDPC Coding

π/4QPSK Rate 0,508 is used in the forward direction only.

A 8-bit CRC is applied to the 2 200 message bits d_0 to d_{2199} as specified in clause 4.3, where $d(i) = d_i$, i = 0, …, 2 199. An 8-bit masking function is applied to the block of 2 208 CRC-protected bits {u(0),…, u(2 207)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(2 207)}, where u′(i) = u(i), i = 0,…, 2 199 and u′(i + 2 200) = u(i + 2 200) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 2 208 bits {u′(0),…, u′(2 207)} is delivered to the encoder.

The 2 208 bits {u′(0),…, u′(2 207)} are encoded via the rate 0,508 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 4 344 coded bits {b(0),…, b(4 343)}.

9.1.8.3 π/4QPSK Rate 0,667 LDPC Coding

π/4QPSK Rate 0,667 is used in the return direction only.

A 8-bit CRC is applied to the 2 952 message bits d_0 to d_{2951} as specified in clause 4.3, where $d(i) = d_i$, i = 0, …, 2 951. An 8-bit masking function is applied to the block of 2 960 CRC-protected bits {u(0),…, u(2 959)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(2 959)}, where u′(i) = u(i), i = 0,…, 2 951 and u′(i + 2 952) = u(i + 2 952) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 2 960 bits {u′(0),…, u′(2 959)} is delivered to the encoder.

The 2 960 bits {u′(0),…, u′(2 959)} are encoded via the rate 0,667 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 4 440 coded bits {b(0),…, b(4 439)}.

9.1.8.4 π/4QPSK Rate 0,681 LDPC Coding

π/4QPSK Rate 0,681 is used in the forward direction only.

A 8-bit CRC is applied to the 2 952 message bits d_0 to d_{2951} as specified in clause 4.3, where $d(i) = d_i$, i = 0, …, 2 951. An 8-bit masking function is applied to the block of 2 960 CRC-protected bits {u(0),…, u(2 959)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(2 959)}, where u′(i) = u(i), i = 0,…, 2 951 and u′(i + 2 952) = u(i + 2 952) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 2 960 bits {u′(0),…, u′(2 959)} is delivered to the encoder.

The 2 960 bits {u′(0),…, u′(2 959)} are encoded via the rate 0,681 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 4 344 coded bits {b(0),…, b(4 343)}.

9.1.8.5 π/4QPSK Rate 0,800 LDPC Coding

π/4QPSK Rate 0,800 is used in the return direction only.

A 8-bit CRC is applied to the 3 544 message bits d_0 to d_{3543} as specified in clause 4.3, where $d(i) = d_i$, i = 0, …, 3 543. An 8-bit masking function is applied to the block of 3 552 CRC-protected bits {u(0),…, u(3 551)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(3 552)}, where u′(i) = u(i), i = 0,…, 3 543 and u′(i + 3 544) = u(i + 3 544) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 3 552 bits {u′(0),…, u′(3 551)} is delivered to the encoder.

The 3 552 bits {u′(0),…, u′(3 551)} are encoded via the rate 0,800 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 4 440 coded bits {b(0),…, b(4 439)}.

9.1.8.6 π/4QPSK Rate 0,818 LDPC Coding

π/4QPSK Rate 0,818 is used in the forward direction only.

A 8-bit CRC is applied to the 3 544 message bits d_0 to d_{3543} as specified in clause 4.3, where $d(i) = d_i$, i = 0, …, 3 543. An 8-bit masking function is applied to the block of 3 552 CRC-protected bits {u(0),…, u(3 551)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(3 552)}, where u′(i) = u(i), i = 0,…, 3 543 and u′(i + 3 544) = u(i + 3 544) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 3 552 bits {u′(0),…, u′(3 551)} is delivered to the encoder.

The 3 552 bits {u′(0),…, u′(3 551)} are encoded via the rate 0,818 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 4 344 coded bits {b(0),…, b(4 343)}.

9.1.8.7 π/4QPSK Rate 0,899 LDPC Coding

π/4QPSK Rate 0,899 is used in the return direction only.

A 8-bit CRC is applied to the 3 984 message bits d_0 to d_{3983} as specified in clause 4.3, where $d(i) = d_i$, i = 0, …, 3 983. An 8-bit masking function is applied to the block of 3 992 CRC-protected bits {u(0),…, u(3 991)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(3 991)}, where u′(i) = u(i), i = 0,…, 3 983 and u′(i + 3 984) = u(i + 3 984) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 3 992 bits {u′(0),…, u′(3 551)} is delivered to the encoder.

The 3 992 bits {u′(0),…, u′(3 991)} are encoded via the rate 0,899 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 4 440 coded bits {b(0),…, b(4 439)}.

9.1.8.8 π/4QPSK Rate 0,919 LDPC Coding

π/4QPSK Rate 0,919 is used in the forward direction only.

A 8-bit CRC is applied to the 3 984 message bits d_0 to d_{3983} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 3 983. An 8-bit masking function is applied to the block of 3 992 CRC-protected bits {u(0),…, u(3 991)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(3 991)}, where u′(i) = u(i), i = 0,…, 3 983 and u′(i + 3 984) = u(i + 3 984) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 3 992 bits {u′(0),…, u′(3 551)} is delivered to the encoder.

The 3 992 bits {u′(0),…, u′(3 991)} are encoded via the rate 0,919 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 4 344 coded bits {b(0),…, b(4 343)}.

9.1.8.9 16APSK Rate 0,667 LDPC Coding

16 APSK Rate 0,667 is used in the return direction only.

A 8-bit CRC is applied to the 5 912 message bits d_0 to d_{5911} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 5 911. An 8-bit masking function is applied to the block of 5 920 CRC-protected bits {u(0),…, u(5 919)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(5 919)}, where u′(i) = u(i), i = 0,…, 5 911 and u′(i + 5 912) = u(i + 5 912) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 5 920 bits {u′(0),…, u′(5 919)} is delivered to the encoder.

The 5 920 bits {u′(0),…, u′(5 919)} are encoded via the rate 0,667 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 8 880 coded bits {b(0),…, b(8 879)}.

9.1.8.10 16APSK Rate 0,681 LDPC Coding

16 APSK Rate 0,681 is used in the forward direction only.

A 8-bit CRC is applied to the 5 912 message bits d_0 to d_{5911} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 5 911. An 8-bit masking function is applied to the block of 5 920 CRC-protected bits {u(0),…, u(5 919)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(5 919)}, where u′(i) = u(i), i = 0,…, 5 911 and u′(i + 5 912) = u(i + 5 912) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 5 920 bits {u′(0),…, u′(5 919)} is delivered to the encoder.

The 5 920 bits {u′(0),…, u′(5 919)} are encoded via the rate 0,681 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 8 688 coded bits {b(0),…, b(8 687)}.

9.1.8.11 16APSK Rate 0,800 LDPC Coding

16 APSK Rate 0,800 is used in the return direction only.

A 8-bit CRC is applied to the 7 096 message bits d_0 to d_{7095} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 7 095. An 8-bit masking function is applied to the block of 7 104 CRC-protected bits {u(0),…, u(7 103)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(7 103)}, where u′(i) = u(i), i = 0,…, 7 095 and u′(i + 7 096) = u(i + 7 096) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 7 104 bits {u′(0),…, u′(7 103)} is delivered to the encoder.

The 7 104 bits {u′(0),…, u′(7 103)} are encoded via the rate 0,800 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 8 880 coded bits {b(0),…, b(8 879)}.

9.1.8.12 16APSK Rate 0,818 LDPC Coding

16 APSK Rate 0,818 is used in the forward direction only.

A 8-bit CRC is applied to the 7 096 message bits d_0 to d_{7095} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 7 095. An 8-bit masking function is applied to the block of 7 104 CRC-protected bits {u(0),…, u(7 103)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(7 103)}, where u′(i) = u(i), i = 0,…, 7 095 and u′(i + 7 096) = u(i + 7 096) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 7 104 bits {u′(0),…, u′(7 103)} is delivered to the encoder.

The 7 104 bits {u′(0),…, u′(7 103)} are encoded via the rate 0,818 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 8 688 coded bits {b(0),…, b(8 687)}.

9.1.8.13 16APSK Rate 0,900 LDPC Coding

16 APSK Rate 0,900 is used in the return direction only.

A 8-bit CRC is applied to the 7 984 message bits d_0 to d_{7983} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 7 983. An 8-bit masking function is applied to the block of 7 992 CRC-protected bits {u(0),…, u(7 991)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(7 991)}, where u′(i) = u(i), i = 0,…, 7 983 and u′(i + 7 984) = u(i + 7 984) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 7 992 bits {u′(0),…, u′(7 991)} is delivered to the encoder.

The 7 992 bits {u′(0),…, u′(7 991)} are encoded via the rate 0,900 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 8 880 coded bits {b(0),…, b(8 879)}.

9.1.8.14 32APSK Rate 0,765 LDPC Coding

32 APSK Rate 0,765 is used in the forward direction only.

A 8-bit CRC is applied to the 8 304 message bits d_0 to d_{8303} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 8 303. An 8-bit masking function is applied to the block of 8 312 CRC-protected bits {u(0),…, u(8 311)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(8 311)}, where u′(i) = u(i), i = 0,…, 8 303 and u′(i + 8 304) = u(i + 8 304) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 8 312 bits {u′(0),…, u′(8 311)} is delivered to the encoder.

The 8 312 bits {u′(0),…, u′(8 311)} are encoded via the rate 0,765 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 10 860 coded bits {b(0),…, b(10 859)}.

9.1.8.15 32APSK Rate 0,818 LDPC Coding

32 APSK Rate 0,818 is used in the forward direction only.

A 8-bit CRC is applied to the 8 872 message bits d_0 to d_{8871} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 8 871. An 8-bit masking function is applied to the block of 8 880 CRC-protected bits {u(0),…, u(8 879)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(8 879)}, where u′(i) = u(i), i = 0,…, 8 871 and u′(i + 8 872) = u(i + 8 872) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 8 880 bits {u′(0),…, u′(8 879)} is delivered to the encoder.

The 8 880 bits {u′(0),…, u′(8 879)} are encoded via the rate 0,818 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 10 860 coded bits {b(0),…, b(10 859)}.

9.1.8.16 Interleaving

Interleaving is performed for 16APSK and 32APSK as described in clause 4.10.3. For π/4 QPSK modulation, there is no interleaving.

9.1.8.17 Scrambling, multiplexing, and encryption

Scrambling is performed as described in clause 4.9. No encryption is performed.

9.1.9 LDPC Coded Packet Normal Burst PNB2(5,3)

The information rate supported by the LDPC coded PNB2(5,3) depends on the modulation and the channel coding used. Modulatios include 32APSK, 16APSK, and π/4QPSK. Approximate channel coding rates include 1/2, 2/3, 3/4, 4/5 and 9/10.

9.1.9.1 π/4QPSK Rate 0,509 LDPC Coding

A 8-bit CRC is applied to the 480 message bits d_0 to d_{479} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 479. An 8-bit masking function is applied to the block of 488 CRC-protected bits {u(0),…, u(487)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(487)}, where u′(i) = u(i), i = 0,…, 479 and u′(i + 480) = u(i + 480) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 488 bits {u′(0),…, u′(487)} is delivered to the encoder.

The 856 bits {u′(0),…, u′(487)} are encoded via the rate 0,509 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 958 coded bits {b(0),…, b(957)}.

9.1.9.2 π/4QPSK Rate 0,660 LDPC Coding

A 8-bit CRC is applied to the 624 message bits d_0 to d_{623} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 623. An 8-bit masking function is applied to the block of 632 CRC-protected bits {u(0),…, u(631)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(631)}, where u′(i) = u(i), i = 0,…, 623 and u′(i + 624) = u(i + 624) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 632 bits {u′(0),…, u′(631)} is delivered to the encoder.

The 632 bits {u′(0),…, u′(631)} are encoded via the rate 0,660 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 958 coded bits {b(0),…, b(957)}.

9.1.9.3 π/4QPSK Rate 0,793 LDPC Coding

A 8-bit CRC is applied to the 752 message bits d_0 to d_{751} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 751. An 8-bit masking function is applied to the block of 760 CRC-protected bits {u(0),…, u(759)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(759)}, where u′(i) = u(i), i = 0,…, 751 and u′(i + 752) = u(i + 752) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 760 bits {u′(0),…, u′(759)} is delivered to the encoder.

The 760 bits {u′(0),…, u′(759)} are encoded via the rate 0,793 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 958 coded bits {b(0),…, b(957)}.

9.1.9.4 π/4QPSK Rate 0,894 LDPC Coding

A 8-bit CRC is applied to the 848 message bits d_0 to d_{847} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 847. An 8-bit masking function is applied to the block of 856 CRC-protected bits {u(0),…, u(855)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(855)}, where u′(i) = u(i), i = 0,…, 847 and u′(i + 848) = u(i + 848) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 856 bits {u′(0),…, u′(855)} is delivered to the encoder.

The 856 bits {u′(0),…, u′(855)} are encoded via the rate 0,894 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 958 coded bits {b(0),…, b(957)}.

9.1.9.5 16APSK Rate 0,664 LDPC Coding

A 8-bit CRC is applied to the 1 264 message bits d_0 to d_{1263} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 1 263. An 8-bit masking function is applied to the block of 1 272 CRC-protected bits {u(0),…, u(1 271)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(1 271)}, where u′(i) = u(i), i = 0,…, 1 263 and u′(i + 1 264) = u(i + 1 264) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 1 272 bits {u′(0),…, u′(1 271)} is delivered to the encoder.

The 1 272 bits {u′(0),…, u′(1 271)} are encoded via the rate 0,664 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 1 916 coded bits {b(0),…, b(1 915)}.

9.1.9.6 16APSK Rate 0,797 LDPC Coding

A 8-bit CRC is applied to the 1 520 message bits d_0 to d_{1519} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 1 519. An 8-bit masking function is applied to the block of 1 528 CRC-protected bits {u(0),…, u(1 527)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(1 527)}, where u′(i) = u(i), i = 0,…, 1 519 and u′(i + 1 520) = u(i + 1 520) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 1 528 bits {u′(0),…, u′(1 527)} is delivered to the encoder.

The 1 528 bits {u′(0),…, u′(1 527)} are encoded via the rate 0,797 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 1 916 coded bits {b(0),…, b(1 915)}.

9.1.9.7 16APSK Rate 0,898 LDPC Coding

A 8-bit CRC is applied to the 1 712 message bits d_0 to d_{1711} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 1 711. An 8-bit masking function is applied to the block of 1 720 CRC-protected bits {u(0),…, u(1 719)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(1 720)}, where u′(i) = u(i), i = 0,…, 1 711 and u′(i + 1 712) = u(i + 1 712) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 1 720 bits {u′(0),…, u′(1 719)} is delivered to the encoder.

The 1 720 bits {u′(0),…, u′(1 719)} are encoded via the rate 0,898 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 1 916 coded bits {b(0),…, b(1 915)}.

9.1.9.8 32APSK Rate 0,748 LDPC Coding

A 8-bit CRC is applied to the 1 784 message bits d_0 to d_{1783} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 1 783. An 8-bit masking function is applied to the block of 1 792 CRC-protected bits {u(0),…, u(1 791)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(1 792)}, where u′(i) = u(i), i = 0,…, 1 783 and u′(i + 1 784) = u(i + 1 784) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 1 792 bits {u′(0),…, u′(1 791)} is delivered to the encoder.

The 1 792 bits {u′(0),…, u′(1 791)} are encoded via the rate 0,748 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 2 395 coded bits {b(0),…, b(2 394)}.

9.1.9.9 32APSK Rate 0,798 LDPC Coding

A 8-bit CRC is applied to the 1 904 message bits d_0 to d_{1903} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 1 903. An 8-bit masking function is applied to the block of 1 912 CRC-protected bits {u(0),…, u(1 911)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(1 912)}, where u′(i) = u(i), i = 0,…, 1 903 and u′(i + 1 904) = u(i + 1 904) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 1 912 bits {u′(0),…, u′(1 911)} is delivered to the encoder.

The 1 912 bits {u′(0),…, u′(1 911)} are encoded via the rate 0,798 LDPC code. The code is defined by the generator polynomial specified in clause 4.10. The encoding results in a block of 2 395 coded bits {b(0),…, b(2 394)}.

9.1.9.10 Interleaving

Interleaving is performed for 16APSK and 32APSK as described in clause 4.10.3. For π/4 QPSK modulation, there is no interleaving.

9.1.9.11 Scrambling, multiplexing, and encryption

Scrambling is performed as described in clause 4.9. No encryption is performed.

9.2 Packet Access Burst (PAB)

9.2.1 Channel coding

An 8-bit CRC is applied to 64 message bits d_0 to d_{63} as specified in clause 4.3, where d(i) = d_i, i = 0, …, 63. An 8-bit masking function is applied to the block of 72 CRC-protected bits {u(0),…,u(71)} in the following manner. The mask, {m(0), m(1), …, m(7)}, is the string RACH_SB_Mask broadcast in system information as specified in GMPRS-1 04.008 [4]. Here m(0) is the MSB bit and m(7) is the LSB bit of the mask. The 8-bit mask is applied to 8 CRC bits of the message with an XOR operation to give us {u′(0),…, u′(71)}, where u′(i) = u(i), i = 0,…, 63 and u′(i + 64) = u(i + 64) ⊕ m(i), i = 0,…, 7, where ⊕ denotes modulo 2 addition (XOR). The resultant block of 72 bits {u′(0),…, u′(71)} is delivered to the encoder.

The 72 bits {u′(0),…, u′(71)} are encoded via the rate 1/2 convolutional code. The code is defined by the generator polynomial specified in clause 4.4.2, where the constraint length K = 7 is used. Eight tail bits are used in order to accommodate K = 9 in the future for the possible enhancement. The encoding results in a block of 160 coded bits {b(0),…, b(159)}.

Puncturing is performed using mask P(2;3) shown in table 4.6. Besides, one more bit in {b} (with index 0) needs to be punctured. This turns out that 54 bits in {b} with index 6k + 3 (k = 0,…, 26), 6k + 4 (k = 0,…, 25) and 0 are punctured. The result is a block of 106 coded bits {c(0),…, c(105)}. The coding rate after puncturing is approximately 3/4.

9.2.2 Interleaving

Intraburst interleaving is performed as described in clause 4.8.1.

9.2.3 Scrambling and multiplexing

No scrambling is performed.

Annex A (normative): LDPC address parity bit accumulators

The Addresses of the Parity Bit Accumulators for the various LDCP codes are defined in tables A.1 to A.18 inclusive.

Table A.1: Address of Parity Bit Accumulators (2/3 C1)

1025 1277 1247 1429 838 836 93 104	642 184 203 1200	548 475 378
557 714 1464 700 681 862 253 672	114 153 1282 408	1029 995 668
516 143 108 810 245 319 411 1238	446 129 399 24	314 963 231
228 399 1182 732 54 1421 9 1098	74 491 1045 319	1245 397 38
564 1218 1334 332 525 47 366 1460	255 731 1080 952	224 692 823
63 286 1110 711 1375 1356 893 14	201 1345 292 353	647 996 220
1055 345 1460 1172 842 9 984 228	10 793 240	1170 702 1456
641 883 226 587 1210 691 575 497	683 1378 407	186 979 87
1279 207 700 1171 776 1198 1423 748	442 641 406	5 439 401
1141 1442 686 889 1430 773 1115 77	433 692 674	75 288 65
916 983 1249 1037	102 804 241	671 1197 640
426 730 101 67	1110 1309 1414	306 991 1157
175 1057 76 1227	756 999 192	
1128 1218 410 1239	109 93 1264	

Table A.2: Address of Parity Bit Accumulators (2/3 C2)

1837 1341 591 2640 790 1906 747 274	2896 1548 902 2596	1572 2609 1377
1802 2917 168 2393 742 1200 740 934	552 937 993 9	1434 2165 2187
631 449 2534 301 927 2643 2750 1516	1504 1382 2483 1035	1015 1107 1121
528 5 2460 1815 305 51 2343 1317	2038 2803 1185 868	2781 1440 2493
720 1043 2685 568 305 2869 2873 2471	866 1809 2284 2458	212 352 1621
2241 1860 484 1835 61 2757 2728 1737	1354 1546 2213 614	1788 598 1569
1979 736 1429 1845 1001 2730 1662 1217	930 2738 2463 1973	2525 2459 2486
366 2115 2753 1371 2932 1454 1546 2187	1433 23 787 670	2484 2138 2742
325 706 2062 411 716 2801 339 1700	1732 2246 2702 1071	1115 2517 1199
423 613 1830 97 412 1044 2125 2864	2286 558 2790 63	2783 558 410
2242 806 1367 1449 570 1293 1215 696	789 212 2796 234	2911 2113 1617
1218 1104 1988 2632 234 438 2079 1920	275 1513 196 754	1428 376 1795
1258 2668 1752 278 2719 613 2149 2002	419 1990 2755 1478	2527 576 1546
255 2621 2125 1834 1327 1860 1056 2306	1239 358 926	1465 607 1690
1704 1058 1331 895 1639 1206 587 1635	328 481 260	1378 1150 1512
145 877 2099 1682 1998 1175 2360 111	1700 1654 1001	2903 2790 2475
1151 2178 2825 1240 663 1002 2941 2239	2309 2571 537	1244 1609 2227
1483 1884 1329 290 2652 2308 2032 2596	1323 2202 510	2691 2308 2752
2599 1180 1907 1371 1237 2561 2768 734	1400 1701 1664	342 1934 2573
763 1764 1126 887 2529 730 2372 2213	2136 951 2843	444 1208 624
442 2377 1781 1694	1852 196 250	2222 911 933
307 265 2079 444	2807 2466 439	864 568 1876
1607 1330 2536 361	1389 2282 1331	334 406 899
1256 2015 2728 1069	1589 1215 1419	2434 473 1178
1592 1339 2749 1641	620 2906 2015	917 1794 2080
2144 2348 1251 247	2953 483 2242	965 25 2677
1259 584 1432 1537	825 1036 1703	

Table A.3: Address of Parity Bit Accumulators (3/4)

92 324 1289 826 1704 135 1214	21 578 702	1952 1432 2186
1114 650 1104 2460 514 2561 1083	558 1480 339	479 2779 433
560 2737 227 316 2247 91 1085	1538 2363 867	2372 1325 2249
2316 411 1040 1916 965 726 1845	538 1007 2533	601 353 2720
2725 2667 1493 41 638 2360 230	873 1776 1423	430 1576 2238
2489 1147 909 726 93 2753 682	999 721 2626	1308 954 2465
1077 2199 1748 1538 894 902 781	1422 1706 826	243 645 67
932 244 941 1214 1243 922 1541	1955 401 937	424 1634 1601
1768 374 1307 2175 1277 1401 610	1242 2509 158	677 1222 1714
2456 141 1334 1810 2751 1896 1116	2207 1095 1925	1116 168 2534
2263 1525 1158 1261 158 1323 1128	672 2313 1055	2105 1877 1628
2057 414 130 1831 1283 556 2164	2165 2099 1443	131 263 2509
2527 1236 1089 974 998 1484 837	2413 2399 2407	401 2348 1494
1762 1403 1899 2158 1394 2005 835	339 2145 1335	2136 2500 2367
2609 173 1409 2133 156 1835 2767	488 305 2095	1327 689 2042
2207 1230 116 541 1835 1737 842	409 1057 1628	1331 1917 1924
352 367 2451 2711 811 822 734	1834 2128 86	573 1447 2712
768 1427 173 409 1119 168 2476	2486 216 1322	2775 627 2127
185 523 2685 652 1696 727 454	89 1202 78	1369 2599 1591
2081 276 979 1655 1384 389 137	1047 1213 769	2676 863 137
284 2432 344 1045 2152 671 1292	299 1189 2137	422 873 1391
701 612 1518 2726 1231 2740 2402	1348 1195 76	2659 1376 194
721 1959 2134 218 901 439 2450	1726 1060 311	1200 2203 684
170 1795 1777 1090 1350 1815 2207	134 2068 2475	302 2551 1197
2121 2144 720 2762 2296 428 473	440 1428 2325	2621 1418 918
2234 1457 1668 1097 1802 2547 289	796 1524 2736	2106 886 1132
2510 1743 96 164 1177 2471 912	1644 228 854	2704 2228 1653
688 673 1011 2094 2009 2610 1391	1528 2674 2780	1140 240 1694
1082 627 291 1231 2289 2416 2016	1133 187 992	763 53 1002
850 2424 292 2198 458 2254 1981	283 768 2086	331 2302 1090
451 967 1746 856 559 613 314	1309 744 1260	971 2450 1550
2211 228 1735 305 1362 1491 696	838 690 2077	1829 1515 1234

Table A.4: Address of Parity Bit Accumulators (4/5 C1)

685 507 62 594 275 103	32 687 26 365 346 456	656 523 459
418 171 164 777 2 313	466 473 443 441 822 794	623 4 128
423 485 181 199 562 575	423 453 642 332 694 780	504 1 304
163 846 737 778 264 836	832 139 123 473 528 253	240 577 319
258 442 391 662 819 264	196 835 635 142 590 741	248 843 146
234 81 187 736 720 338	733 844 115 71 857 42	756 294 856
533 60 218 861 455 7	217 268 667 704 11 378	744 445 319
628 759 851 21 301 666	853 206 340 343 560 431	430 205 284
736 368 806 550 777 702	447 501 700	518 445 140
772 377 764 507 394 517	178 14 750	294 307 477
120 28 317 80 59 469	680 390 809	79 537 412
420 844 689 428 357 467	138 623 369	796 130 207
735 754 237 342 871 49	361 686 707	378 131 732
514 797 227 369 302 276	730 531 141	761 22 387
108 793 428 375 354 448	305 552 478	437 239 266
876 554 293 9 315 188	686 773 187	281 333 467

Table A.5: Address of Parity Bit Accumulators (4/5 C2)

615 1616 81 1048 141 476	17 1512 908 1510 1082 922	1402 862 379
561 818 563 197 1341 878	1150 852 127 933 1291 1309	1069 329 1373
1509 1723 1000 988 103 123	158 1361 1212 1696 634 1350	662 241 1071
286 395 1001 1072 99 1704	29 755 516 800 1127 1521	731 992 1320
1682 1648 1121 7 60 1381	233 755 1336 23 1683 1238	920 1229 578
957 1366 92 962 417 595	409 1357 1629 244 1494 367	300 638 1163
573 665 652 840 1754 843	687 1386 1083 157 513 124	1179 210 636
1567 611 784 1677 134 1250	1579 144 687 132 779 621	1464 650 639
1208 263 277 1002 1370 1137	1301 695 760 1699 1184 1595	1232 1221 907
1000 11 1045 1363 830 1421	414 603 1562 834 695 1119	1692 1027 1489
1286 102 764 196 610 180	605 157 1288 1623 1345 1173	1197 1508 104
574 1091 572 499 425 4	1263 1329 1349 1320 670 573	335 1423 1062
625 910 1210 1039 47 1620	1370 1251 1140 140 441 271	962 490 1252
1313 1349 804 1583 409 984	594 2 670 684 1737 959	1207 1275 1036
709 1329 1124 1205 22 689	1518 1088 232 378 620 1165	611 1490 504
1143 898 336 275 1141 200	194 1734 368 16 1482 1532	104 681 698
528 1044 1175 3 777 368	607 1314 1708	637 563 163
900 1498 1558 1679 1454 507	126 921 1037	409 974 593
1421 1282 354 1542 1639 1345	963 545 846	1752 1589 1547
1033 1602 1555 1632 423 1036	1353 34 812	1421 286 1162
390 1719 440 1027 455 710	448 685 995	784 1338 342
510 1650 1276 548 1573 922	551 44 1327	743 165 1745
433 894 567 1194 1472 1375	1028 1136 934	1372 1061 1618
529 1155 1421 30 1287 90	1725 226 1727	406 282 230
1446 55 557 680 1430 1745	402 412 679	396 1761 1720
801 1522 1135 673 1622 454	1706 1288 204	305 1358 799
679 1411 1211 134 1732 838	1608 1743 25	188 1584 444
1155 24 1775 1724 364 1493	409 907 280	1607 399 1563
1763 355 1035 793 634 1194	571 502 13	1734 459 574
1444 778 1129 548 1176 785	1059 1149 582	496 1257 985
24 1485 931 1418 58 628	518 1053 153	925 1359 1436
399 1616 1289 1190 733 169	28 1025 1103	613 1263 378

Table A.6: Address of Parity Bit Accumulators (4/5 C3)

1664 2044 112 1053 482 1419	1733 785 944 1502 457 819	382 836 995
613 198 1630 1342 56 1024	558 881 826 764 1243 67	2168 21 1514
1306 572 691 1127 1650 2044	541 1975 1968 1090 68 1447	1066 1684 1747
983 1550 1244 2178 617 1756	1675 356 729 259 923 1341	197 1433 1368
455 1920 92 2201 1417 975	838 1111 981 1155 1744 1134	1473 1235 600
390 251 1987 114 56 2219	1349 55 1708 247 43 1736	871 1708 1085
1080 187 688 743 1006 859	84 2072 1253 934 2020 1536	569 259 1581
2090 1738 652 325 1199 591	2024 1481 2186 1802 627 714	983 347 1192
1477 656 973 1858 820 698	660 72 1063 1565 115 1151	829 1409 368
1574 275 2104 2077 679 121	30 800 1484 103 1476 415	1170 1189 1257
216 1131 569 485 1713 1141	1190 1267 63 1028 1086 914	1864 1688 1763
1017 322 2099 565 1486 2162	317 1302 18 1204 2002 1019	1166 672 188
2046 150 520 284 1916 512	1449 1837 1681 739 385 569	416 1358 582
576 531 1687 842 1411 1395	403 135 755 1859 682 1607	355 1799 632
1361 1041 663 2164 496 258	1092 993 658 77 905 409	1145 1234 1423
1992 569 221 796 758 48	1849 502 1026 220 1061 898	303 1671 1771
790 1109 440 1827 1878 2033	2175 1488 27 869 1410 1607	276 789 472
1667 162 1711 1664 2155 1582	1323 1909 1949 1084 1810 878	1473 893 41
315 2084 1941 799 1788 1723	1923 1403 1047 1788 1732 784	1535 2066 863
1797 1501 1508 180 2170 762	1443 39 1720 1157 1403 1167	1681 517 880
219 1641 887 1318 553 6	339 982 1423	1605 1321 1393
1418 503 1914 942 1515 568	1572 1586 1328	406 1978 1105
519 319 2194 1183 1854 1703	440 555 403	1379 2004 1024
804 1815 430 1525 1193 791	198 1716 983	1809 1071 45
1370 908 1974 275 1121 2169	326 1926 1405	1302 1095 1998
2094 553 1388 1197 1176 1035	1518 162 250	959 1630 2016
522 2006 1945 6 1385 9	2110 66 624	13 990 1211
2103 25 2049 139 1307 1047	2090 137 1687	207 1726 49
1323 1910 737 1016 322 2195	1721 1922 1105	2087 1147 1939
273 792 1309 1370 1167 1498	932 1186 2139	1323 1491 572
474 867 2161 828 2126 1778	80 674 2009	1884 1022 306
2116 584 688 1061 1695 1422	1269 1651 616	237 617 166
186 275 2078 2179 1373 1190	373 464 1108	677 950 1248
826 1011 894 762 849 1940	847 954 881	264 770 1732
1200 2066 106 471 1485 20	2021 688 490	761 1507 1128
1779 1292 1623 1171 1106 1573	465 414 332	704 1498 695
392 705 347 1910 1288 417	471 315 453	1220 900 357
184 1926 112 1781 840 1041	1572 904 925	652 669 709
1486 1831 1902 245 1521 1330	1593 721 567	1034 595 268
504 976 330 398 2012 1312	1350 574 2127	150 1450 1994

Table A.7: Address of Parity Bit Accumulators (9/10 C1)

410 348 269 135 440	390 367 154 302 312	54 63 74
26 28 107 242 388	210 396 197 213 355	327 85 72
197 146 404 42 172	247 90 369 354 225	191 127 125
379 90 357 268 445	184 167 403 196 78	228 346 30
1 297 152 335 392	275 381 41 189 221	266 383 244
120 100 262 369 308	233 290 117 104 190	311 60 29
281 44 234 256 447	210 331 325 173 293	42 297 412
317 397 286 283 172	126 297 25	157 146 253
366 64 297 91 334	219 439 364	217 16 197
303 192 134 406 186	134 136 68	312 289 363
167 430 420 327 289	67 366 147	248 256 48
225 312 230 234 441	307 29 170	362 4 205
64 305 369 27 168	20 397 281	21 26 73
372 331 319 404 132	136 207 223	217 15 341
229 30 293 129 256	283 5 408	220 191 235
191 195 266 145 274	127 383 333	293 82 196
170 210 248 302 200	209 390 57	256 296 329
182 254 139 290 110	14 291 366	382 345 442
309 431 138 0 55	259 330 235	229 24 274
232 339 53 216 266	129 104 114	426 87 162
8 247 297 221 355	0 164 314	28 40 48

Table A.8: Address of Parity Bit Accumulators (9/10 C2)

879 372 573 557 788	434 695 354 37 556	582 575 362
827 438 506 308 796	242 34 83 504 673	297 216 294
33 730 353 155 291	82 858 807 744 767	796 843 729
188 857 362 690 755	197 480 111 535 410	597 239 303
657 303 448 406 67	768 799 8 594 789	566 642 377
27 412 69 156 90	337 419 368 594 161	212 787 758
423 665 728 184 261	772 877 142 779 171	571 219 779
253 496 719 91 830	388 218 725 384 115	64 92 823
181 392 583 468 518	9 398 80 738 365	65 548 49
721 574 683 403 246	201 848 457 292 519	238 876 341
756 733 850 210 77	259 633 106 8 880	877 10 720
252 649 470 751 214	411 161 667 513 814	397 736 22
182 297 780 442 119	786 371 829	533 205 495
403 46 873 522 396	636 399 500	763 248 531
545 710 387 706 779	466 651 84	642 475 192
21 242 103 507 577	674 18 826	75 869 560
723 501 840 112 865	738 852 296	586 362 654
170 279 715 56 793	773 335 598	415 484 245
715 765 648 445 695	697 140 604	444 124 442
580 192 722 631 483	399 253 31	372 261 364
449 354 644 85 772	21 317 67	285 848 514
185 462 200 748 58	508 741 451	887 493 462
701 438 80 323 394	278 160 821	14 263 865
772 29 78 548 611	662 885 287	38 717 887
40 343 337 432 330	809 322 163	743 74 197
232 393 363 829 839	181 355 568	705 650 239
132 78 165 482 596	835 402 564	407 484 705
386 78 859 621 99	303 886 297	835 58 762
141 327 673 65 23	299 645 80	207 402 247
524 46 87 11 738	796 685 435	185 397 260
838 811 362 681 402	95 472 284	589 108 789
590 283 457 532 339	574 320 270	682 124 163
293 804 646 878 385	686 167 1	737 482 716
701 692 467 528 607	798 48 638	346 759 882
400 221 560 22 312	809 552 21	432 529 651
796 845 740 370 155	26 711 137	384 496 824

Table A.9: Address of Parity Bit Accumulators (1/2)

779 2064 86 1321 176 606	753 1384 714 640 530 6	217 1678 1435 1662
698 1012 736 260 1688 1101	2206 1079 1911 221 134 1237	1311 208 58 919
1904 2153 1248 1231 119 185	1306 71 2035 887 630 2063	592 1577 1389 187
393 521 1286 1357 149 2124	2222 1765 1424 1091 1443 1782	647 1389 700 1651
2128 2077 1430 10 90 1747	33 1238 721 796 672 216	676 1852 142 1287
1220 1725 138 1227 518 749	1154 117 1705 1858 1719 496	1879 1931 1196 1671
751 853 835 1044 3 1049	1271 1876 2087 932	548 851 1395 454
19 1311 2226 740 133 1559	60 5 1133 44	424 1101 1620 1463
508 816 416 218 2085 2098	1745 2167 1009 1350	1917 1489 978 1730
1524 323 344 1251 1731 1454	433 1889 1197 394	1546 758 1272 1075
1284 17 1316 1721 1066 1772	660 1134 182 121	837 1474 216 1338
1605 118 966 258 771 235	279 1615 101 1849	585 660 1419 322

Table A.10: Address of Parity Bit Accumulators (2/3 C1 PNB2(5,3))

52 226 253 98 124 301 55 277	134 256 221 29	254 118 172
217 42 121 130 278 179 106 255	212 57 79 116	3 180 59
30 21 272 183 84 25 126 107	250 42 175 187	77 46 263
190 273 134 8 49 315 122 286	140 75 222	101 215 143
70 161 13 64 47 158 9 55	104 251 148	227 49 75
270 28 221 113	314 29 51	142 216 167
219 8 203 44	206 90 118	

Table A.11: Address of Parity Bit Accumulators (2/3 C2 PNB2(5,3))

322 371 222 566 392 628 121	29 56 393 442 79 546 613	569 362 200
465 389 218 375 508 307 54	36 557 20 441 418 155 435	615 450 83
591 234 553 403 557 364 278	48 173 284 273 610 599 133	434 412 22
308 489 426 194 228 603 535	600 425 210 152 561 226 203	127 529 9
219 308 516 400 482 295 191	51 321 167	346 19 280
424 581 517 371 603 93 612	338 60 183	405 425 438
190 266 467 493 340 355 123	240 205 274	475 421 208
346 487 228 366 337 376 434	500 119 288	341 390 248

Table A.12: Address of Parity Bit Accumulators (3/4 PNB2(5,3))

381 533 8 84 355 316	352 407 495 596	526 321 15
354 82 16 284 485 165	503 570 149 49	179 165 80
487 282 51 124 584 245	538 299 81 212	351 35 66
254 445 213 303 280 46	286 33 594 348	84 410 388
476 118 9 448 302 451	483 274 479 520	290 480 213
58 41 88 375 587 325	322 483 269 264	188 302 387
108 45 463 207 445 296	435 525 14 175	223 372 334
359 257 234 298 96 469	73 590 89 48	553 209 166
422 575 447 185 346 549	144 2 510 31	172 332 221
540 110 570 443 481 80	208 188 393 121	455 498 445
228 290 462 403 191 219	271 209 128 422	67 234 85
564 505 122 270 403 335	265 52 582 427	484 89 55

Table A.13: Address of Parity Bit Accumulators (4/5 C1 PNB2(5,3))

84 66 59 10 3 51	188 171 54 60 175 22	77 60 156
147 166 111 84 99 13	102 64 158 111 16 167	52 17 58
151 183 125 72 149 30	91 145 180 23 96 58	134 73 127
77 170 128 103 134 155	2 89 98 35 41 86	72 118 189
91 115 8 76 30 123	172 27 189 155 153 144	167 109 125
152 53 99 41 188 156	130 102 76 177 18 109	70 26 133
2 107 70 125 18 156	73 89 51	118 143 107
82 64 87 129 108 35	161 75 79	90 116 182
102 14 97 1 79 166	4 106 170	116 22 140
90 82 74 105 37 148	50 129 155	91 88 154
147 43 70 29 56 61	43 28 41	31 95 154
84 119 180 168 185 163	154 173 112	161 105 59
44 22 137 129 135 183	136 104 102	
161 130 73 84 56 117	117 178 155	

Table A.14: Address of Parity Bit Accumulators (4/5 C2 PNB2(5,3))

334 381 224 218 27 37	270 65 170 265 120 279	29 26 226
339 137 176 370 31 277	241 206 33 279 178 7	115 217 143
224 5 234 297 187 314	123 167 98 10 354 356	272 347 361
283 27 178 50 137 42	113 349 23 256 308 158	300 199 310
276 18 181 154 203 269	321 312 176 88 182 34	74 65 234
316 219 276 103 1 305	307 261 185 279 374 49	140 252 364
274 343 14 42 219 335	216 64 319 56 333 101	49 285 223
4 200 13 341 242 263	360 195 220 103 56 273	375 112 229
300 208 321 373 202 306	379 312 270	267 348 142
264 100 69 83 254 56	338 189 270	346 55 158
348 52 333 59 224 291	375 297 83	340 363 185
144 136 318 307 177 383	180 171 242	70 9 114
164 216 61 114 244 131	248 165 136	237 71 121
47 178 43 285 65 174	63 316 178	323 62 18
143 238 109 276 366 271	234 256 281	260 144 377
34 315 174 143 148 365	293 229 224	79 128 119

Table A.15: Address of Parity Bit Accumulators (4/5 C3 PNB2(5,3))

465 293 22 339 176 200	233 394 366 261 168 15	101 107 372
111 127 192 300 243 169	310 211 338 62 265 474	21 104 347
352 431 448 209 133 105	310 159 305 478 303 444	434 322 458
172 169 103 76 108 290	380 77 289 321 83 92	378 431 336
44 10 442 386 41 398	40 145 476 377 279 288	48 302 271
284 57 116 458 442 25	470 252 435 226 87 439	380 292 449
64 387 370 131 106 443	303 300 286	374 337 365
4 425 67 421 360 126	446 375 240	199 475 433
224 295 257 188 119 181	171 314 98	300 329 277
333 384 295 477 188 339	135 468 109	235 427 381
271 44 416 332 293 208	4 129 200	43 396 174
409 54 52 456 127 143	329 463 458	10 143 306
21 308 87 273 384 370	242 477 215	
37 15 201 362 129 406	323 426 252	

Table A.16: Address of Parity Bit Accumulators (9/10 C1 PNB2(5,3))

61 46 53 86	57 8 13	66 91 10
67 2 52 72	45 74 91	18 40 69
63 17 10 43	25 21 46	29 84 32
84 3 40 91	24 89 58	49 5 28
60 20 3 65	78 95 74	62 21 89
24 56 87 35	0 10 71	85 32 45
31 3 35 72	69 53 80	4 21 29
11 84 1 70	79 38 82	7 83 93
56 85 28 66	88 5 79	60 68 76
2 75 61 30	30 34 8	94 61 74
74 21 10 59	54 39 7	60 1 81
20 15 22 11	24 81 59	48 86 5
86 13 51 64	21 70 43	19 26 65
39 48 95 16	23 76 79	29 27 61
71 78 52 57	5 68 75	11 24 13
67 32 28 83	30 32 22	70 39 42
72 43 92 5	0 43 77	38 22 63
90 67 56 93	48 2 21	84 68 82

Table A.17: Address of Parity Bit Accumulators (9/10 C2 PNB2(5,3))

92 61 29 147 100	155 15 78 184 37	134 115 120
120 105 26 37 65	14 147 167 103 178	79 21 38
0 118 147 91 158	138 2 69 53 31	6 145 77
30 166 119 93 139	144 62 160 71 157	148 53 66
6 176 172 59 121	48 2 141 178 79	177 184 107
0 92 165 22 17	36 188 105 130 11	116 123 190
75 55 182 113 18	62 124 177	127 8 24
154 104 156 191 99	49 153 46	182 138 16
115 58 63 110 77	137 36 88	184 121 96
48 127 4 71 68	107 151 102	171 179 136
18 175 99 4 179	158 9 179	99 41 49
18 37 50 27 160	54 97 188	98 159 125
162 37 3 158 35	159 60 23	94 18 149
124 45 134 131 84	33 61 143	128 6 67
175 136 15 83 102	154 111 109	90 111 97
151 8 64 6 191	32 52 6	185 69 20
1 146 75 184 72	146 103 112	148 169 56
67 32 189 148 179	90 158 47	129 190 83

Table A.18: Address of Parity Bit Accumulators (1/2 PNB2(5,3))

432 467 351 286 130 273	354 153 420 229
146 216 275 428 465 478	247 256 484 226
395 238 272 245 68 354	157 342 63 475
133 303 179 465 460 352	281 29 422 167

Annex B (informative): Bibliography

GMPRS-1 05.002 (ETSI TS 101 376-5-2): "GEO-Mobile Radio Interface Specifications (Release 2); General Packet Radio Service; Part 5: Radio interface physical layer specifications; Sub-part 2: Multiplexing and Multiple Access; Stage 2 Service Description; GMPRS-1 05.002".

GMPRS-1 05.004 (ETSI TS 101 376-5-4): "GEO-Mobile Radio Interface Specifications (Release 2); General Packet Radio Service Part 5: Radio interface physical layer specifications; Sub-part 4: Modulation; GMPRS-1 05.004".

GMPRS-1 05.008 (ETSI TS 101 376-5-6): "GEO-Mobile Radio Interface Specifications (Release 2); General Packet Radio Service Part 5: Radio interface physical layer specifications; Sub-part 6: Radio Subsystem Link Control; GMPRS-1 05.008".

GMPRS-1 05.010 (ETSI TS 101 376-5-7): "GEO-Mobile Radio Interface Specifications (Release 2); General Packet Radio Service Part 5: Radio interface physical layer specifications; Sub-part 7: Radio Subsystem Synchronization; GMPRS-1 05.010".

History

Document history		
V2.1.1	March 2003	Publication
V2.2.1	March 2005	Publication
V2.3.1	July 2008	Publication