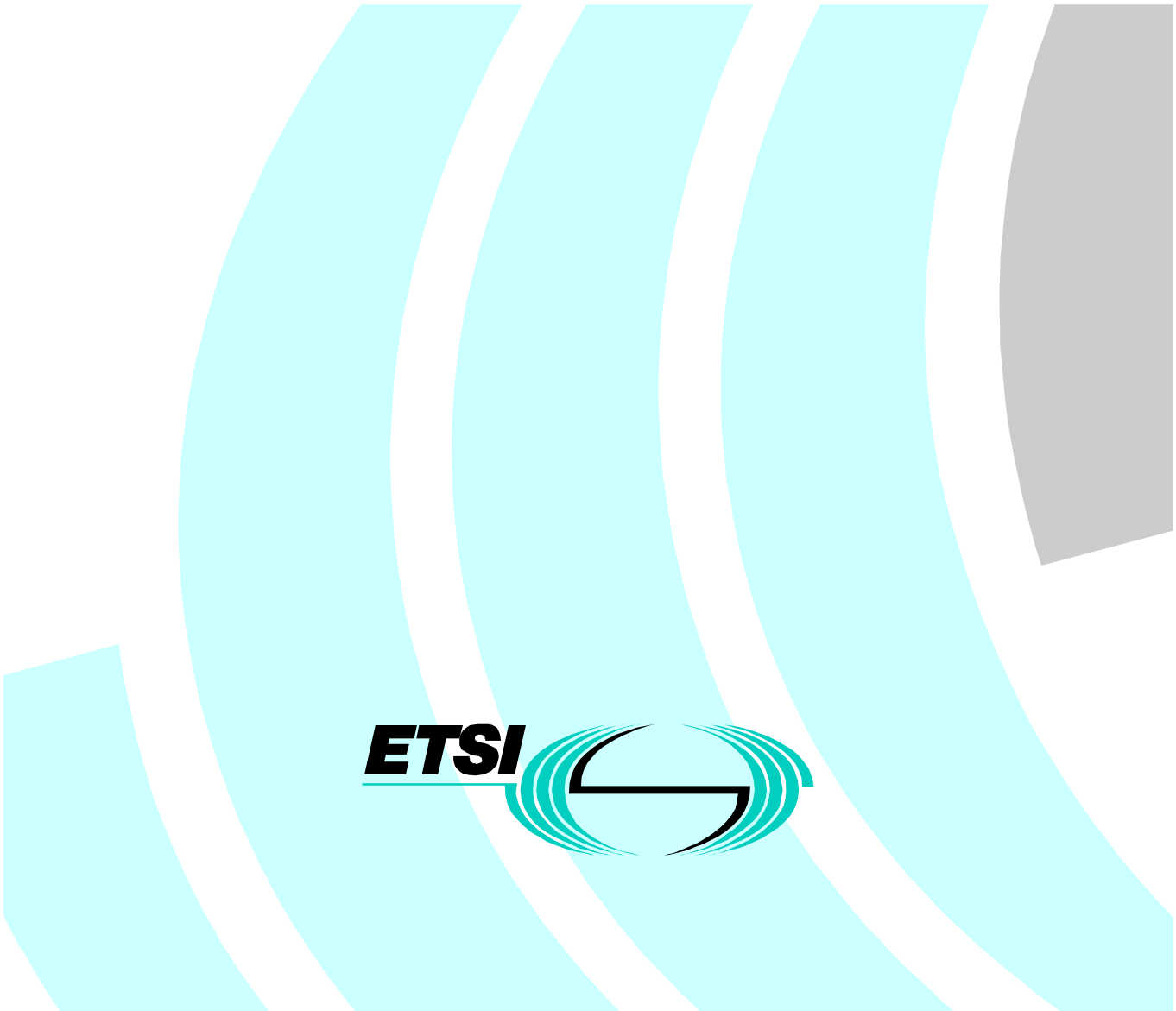
ETSI TS 101 377-5-3 V1.1.1 (2001-03)

Technical Specification

GEO-Mobile Radio Interface Specifications;
Part 5: Radio interface physical layer specifications;
Sub-part 3: Channel Coding;
GMR-2 05.003



Reference
DTS/SES-002-05003

Keywords
coding, GMR, 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

The information pertaining to essential IPRs 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://www.etsi.org/ipr).

The attention of ETSI has been drawn to the Intellectual Property Rights (IPRs) listed below which are, or may be, or may become, Essential to the present document. The IPR owner has undertaken to grant irrevocable licences, on fair, reasonable and non-discriminatory terms and conditions under these IPRs pursuant to the ETSI IPR Policy. Further details pertaining to these IPRs can be obtained directly from the IPR owner.

The present IPR information has been submitted to ETSI and 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.

IPRs:

Project	Company	Title	Country of Origin	Patent n°	Countries Applicable
TS 101 377 V1.1.1	Digital Voice Systems Inc		US	US 5,715,365	US
TS 101 377 V1.1.1	Digital Voice Systems Inc		US	US 5,754,974	US
TS 101 377 V1.1.1	Digital Voice Systems Inc		US	US 5,226,084	US
TS 101 377 V1.1.1	Digital Voice Systems Inc		US	US 5,701,390	US
TS 101 377 V1.1.1	Digital Voice Systems Inc		US	US 5,826,222	US

IPR Owner: Digital Voice Systems Inc
One Van de Graaff Drive Burlington,
MA 01803
USA

Contact: John C. Hardwick
Tel.: +1 781 270 1030
Fax: +1 781 270 0166

Project	Company	Title	Country of Origin	Patent n°	Countries Applicable
TS 101 377 V1.1.1	Ericsson Mobile Communication	Improvements in, or in relation to, equalisers	GB	GB 2 215 567	GB
TS 101 377 V1.1.1	Ericsson Mobile Communication	Power Booster	GB	GB 2 251 768	GB
TS 101 377 V1.1.1	Ericsson Mobile Communication	Receiver Gain	GB	GB 2 233 846	GB
TS 101 377 V1.1.1	Ericsson Mobile Communication	Transmitter Power Control for Radio Telephone System	GB	GB 2 233 517	GB

IPR Owner: Ericsson Mobile Communications (UK) Limited
The Keytech Centre, Ashwood Way
Basingstoke
Hampshire RG23 8BG
United Kingdom

Contact: John Watson
Tel.: +44 1256 864 821

Project	Company	Title	Country of Origin	Patent n°	Countries Applicable
TS 101 377 V1.1.1	Hughes Network Systems		US	Pending	US

IPR Owner: Hughes Network Systems
11717 Exploration Lane
Germantown, Maryland 20876
USA

Contact: John T. Whelan
Tel: +1 301 428 7172
Fax: +1 301 428 2802

Project	Company	Title	Country of Origin	Patent n°	Countries Applicable
TS 101 377 V1.1.1	Lockheed Martin Global Telecommunic. Inc	2.4-to-3 KBPS Rate Adaptation Apparatus for Use in Narrowband Data and Facsimile Communication Systems	US	US 6,108,348	US
TS 101 377 V1.1.1	Lockheed Martin Global Telecommunic. Inc	Cellular Spacecraft TDMA Communications System with Call Interrupt Coding System for Maximizing Traffic ThroughputCellular Spacecraft TDMA Communications System with Call Interrupt Coding System for Maximizing Traffic Throughput	US	US 5,717,686	US
TS 101 377 V1.1.1	Lockheed Martin Global Telecommunic. Inc	Enhanced Access Burst for Random Access Channels in TDMA Mobile Satellite System	US	US 5,875,182	
TS 101 377 V1.1.1	Lockheed Martin Global Telecommunic. Inc	Spacecraft Cellular Communication System	US	US 5,974,314	US
TS 101 377 V1.1.1	Lockheed Martin Global Telecommunic. Inc	Spacecraft Cellular Communication System	US	US 5,974,315	US
TS 101 377 V1.1.1	Lockheed Martin Global Telecommunic. Inc	Spacecraft Cellular Communication System with Mutual Offset High-argin Forward Control Signals	US	US 6,072,985	US
TS 101 377 V1.1.1	Lockheed Martin Global Telecommunic. Inc	Spacecraft Cellular Communication System with Spot Beam Pairing for Reduced Updates	US	US 6,118,998	US

IPR Owner: Lockheed Martin Global Telecommunications, Inc.
900 Forge Road
Norristown, PA. 19403
USA

Contact: R.F. Franciose
Tel.: +1 610 354 2535
Fax: +1 610 354 7244

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 1.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 covering the GEO-Mobile Radio Interface Specifications, 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; GMR-2 05.001";

Sub-part 2: "Multiplexing and Multiple Access on the Radio Path; GMR-2 05.002";

Sub-part 3: "Channel Coding; GMR-2 05.003";

Sub-part 4: "Modulation; GMR-2 05.004";

Sub-part 5: "Radio Transmission and Reception; GMR-2 05.005";

Sub-part 6: "Radio Subsystem Link Control; GMR-2 05.008";

Sub-part 7: "Radio Subsystem Synchronization; GMR-2 05.010";

Part 6: "Speech coding 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.

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 as follows:

GMR-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 GMR-n 01.201.

1 Scope

A reference configuration of the transmission chain is shown in figure A.1 of GMR-2 05.001 [4]. According to this reference configuration, the present document specifies the data blocks given to the encryption unit.

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

The definition is given for each kind of logical channel, starting from the data provided to the channel encoder by the speech coder, the data terminal equipment, or the controller of the MES. The definitions of the logical channel types used in the present document are given in clause 5 of GMR-2 05.002 [5], a summary of which is contained in Annex A of the present document.

2 References

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

- References are either specific (identified by date of publication and/or edition number or version number) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies.

[1] GMR-2 01.004 (ETSI TS 101 377-1-1): "GEO-Mobile Radio Interface Specifications; Part 1: General specifications; Sub-part 1: Abbreviations and Acronyms; GMR-2 01.004".

[2] GMR-2 04.008 (ETSI TS 101 377-4-7): "GEO-Mobile Radio Interface Specifications; Part 4: Radio interface protocol specifications; Sub-part 7: Mobile radio interface Layer 3 Specifications; GMR-2 04.008".

[3] GMR-2 04.021 (ETSI TS 101 377-4-10): "GEO-Mobile Radio Interface Specifications; Part 4: Radio interface protocol specifications; Sub-part 10: Rate Adaptation on the Mobile earth Station (MES)- Gateway System Interface.; GMR-2 04.021".

[4] GMR-2 05.001 (ETSI TS 101 377-5-1): "GEO-Mobile Radio Interface Specifications; Part 5: Radio interface physical layer specifications; Sub-part 1: Physical Layer on the Radio Path; GMR-2 05.001".

[5] GMR-2 05.002 (ETSI TS 101 377-5-2): "GEO-Mobile Radio Interface Specifications; Part 5: Radio interface physical layer specifications; Sub-part 2: Multiplexing and Multiple Access on the Radio Path; GMR-2 05.002".

[6] GSM 05.03 (ETSI ETS 300 575): "Digital cellular telecommunications system (Phase 2); Channel coding (GSM 05.03 version 4.5.1)".

3 Abbreviations

For the purposes of the present document, the abbreviations given in GMR-2 01.004 [1] apply.

4 General

4.1 General organization

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

Each channel generally uses the following sequence and order of operations:

a) the information bits are coded with a systematic block code, building words of information + parity bits;

b) the information + parity bits are encoded with a convolutional code, (punctured or unpunctured), building the coded bits;

c) reordering and interleaving the coded bits gives the interleaved bits.

All these operations are made block by block, the size of which depends on the channel. Figure 4.2.1 gives a diagram showing the general structure of the channel coding.

A block of 480 or 960 coded bits is the basic structure of many of the control channels and the data/fax channels. In the case of control channels, it carries one message. In the case of data/fax, it carries one block of data.

Some channel types do not fit in the general organization. These include the S-RACH, the S-HPACH, the S-HBCCH, and the S-SCH.

4.2 Naming convention

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

a) General naming
"k" and "j" for numbering of bits in data blocks and bursts.
"Kx" gives the number of bits in one block, where "x" refers to the data type
"n" is used for numbering of delivered data blocks where:
"N" marks a certain data block
"B" is used for numbering of bursts or blocks where:
"B0" marks the first burst or block carrying bits from the data block with n = 0 (first data block in the transmission)

b) Data delivered to the encoding unit (interface 1 in figure 4.2.1):
d (n, k) or d (k) for k = 0, 1, ..., Kd - 1

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

c) Data after the first encoding step (block code, cyclic code; interface 2 in figure 4.2.1):
u (n, k) or u (k) for k = 0, 1, ..., Ku - 1

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

d) Data after the second encoding step (convolutional code; interface 3 in figure 4.2.1):
c(n, k) or c(k) for k = 0, 1, ..., Kc - 1

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

e) Interleaved data:
i(B, k) for k = 0, 1, ..., Ki - 1

B = Bo, Bo + 1, ...

f) Bits in one burst (interface 4 in figure 4.2.1):
e(B, k) for k = 0, 1, ..., 118, 119

B = Bo, Bo + 1, ...

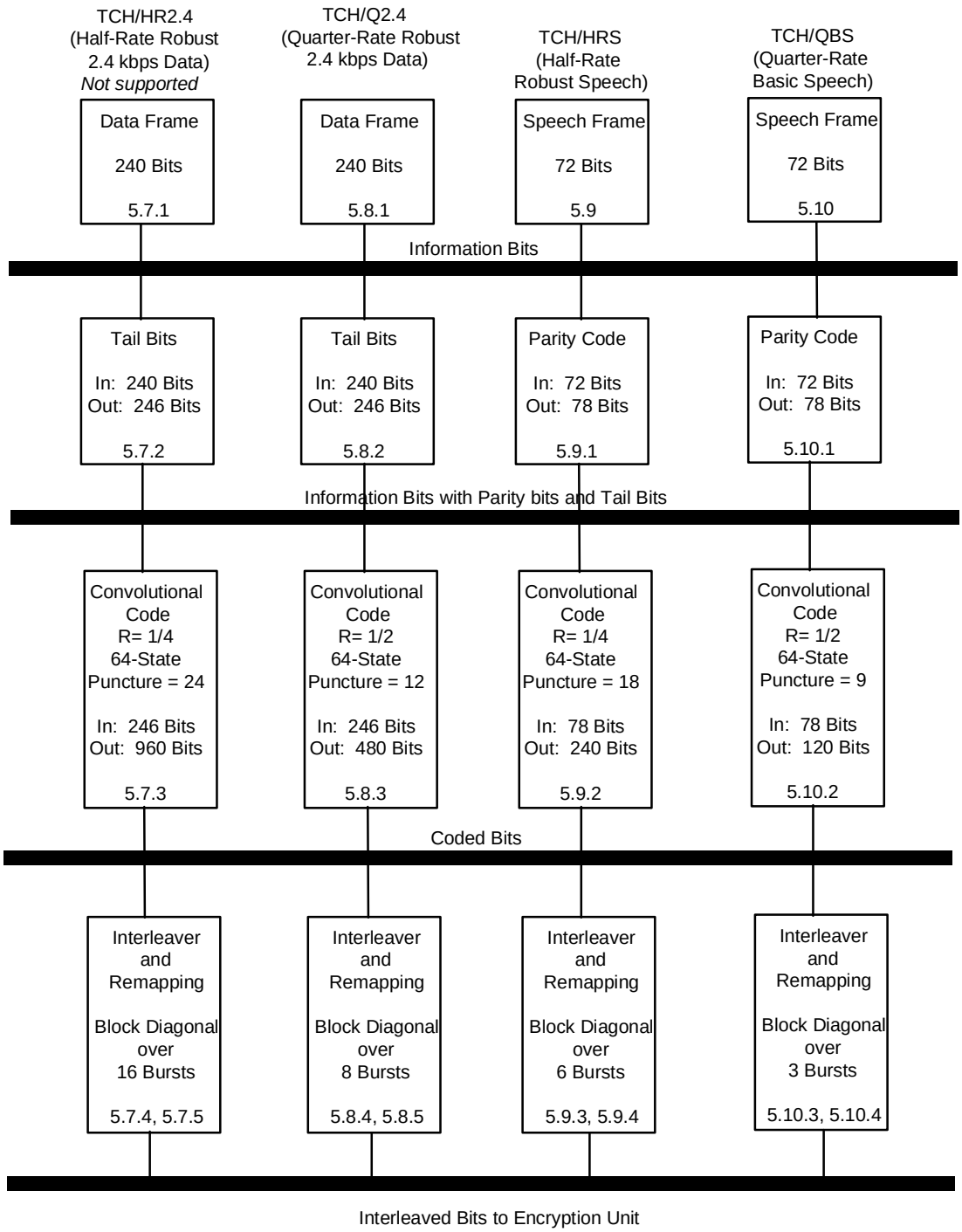


Figure 4.2.1: Channel Coding and Interleaving Organization

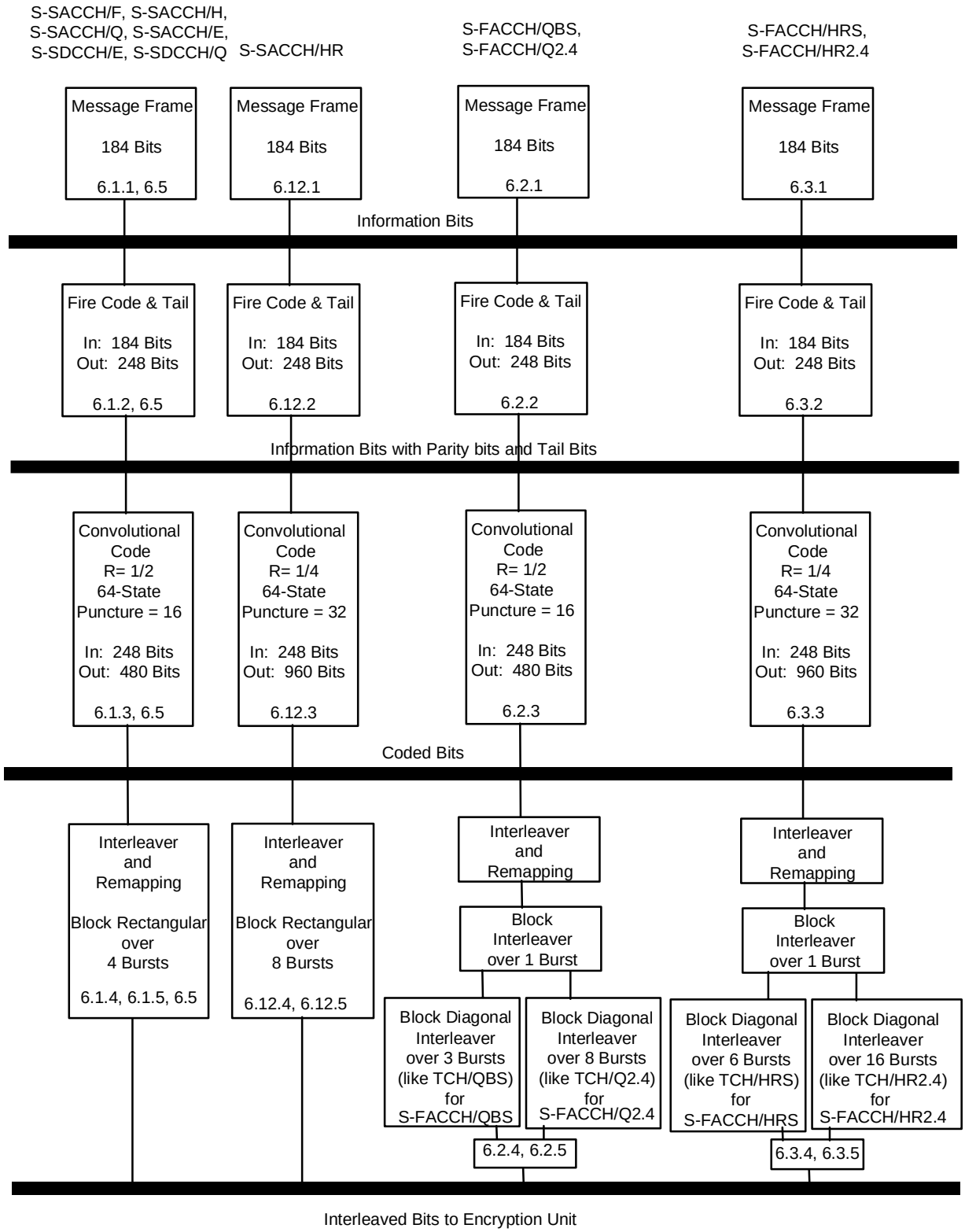


Figure 4.2.1 (continued): Channel Coding and Interleaving Organization

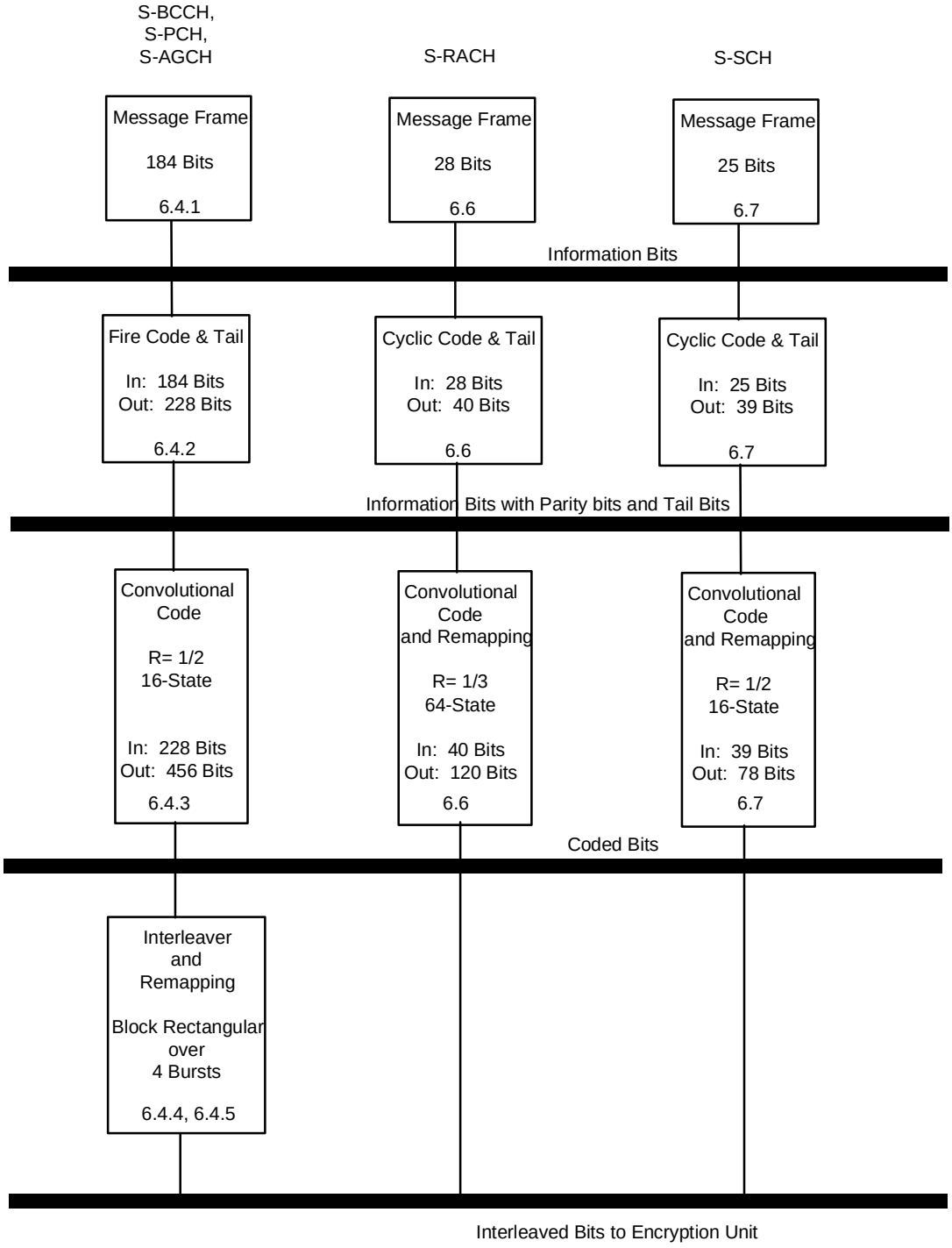


Figure 4.2.1 (continued): Channel Coding and Interleaving Organization

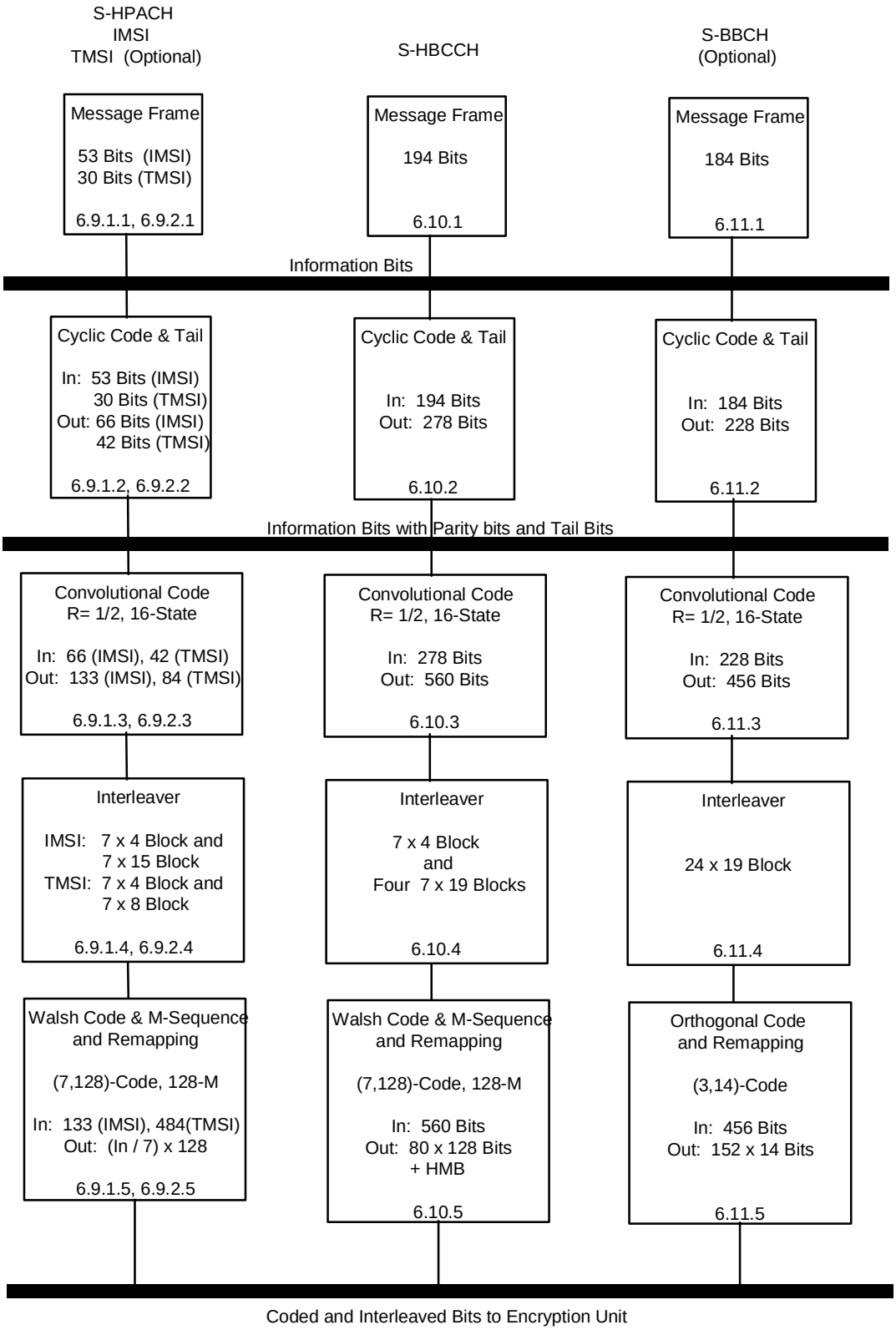


Figure 4.2.1 (continued): Channel Coding and Interleaving Organization

5 Traffic channels

Two kinds of traffic channel are considered: speech and data. Both of them use the same general structure (see figure 4.2.1). A piece of information can be stolen by the S-FACCH. The structure of the burst carrying the traffic channel is detailed in clause 7.2 of GMR-2 05.002 [5].

5.1 Speech Channel at Full Rate (TCH/FS and TCH/EFS)

Reserved for future use, not currently supported in the GMR-2 system.

5.2 Enhanced Speech Channel at Half-Rate (S-TCH/EHS)

Reserved for future use, not currently supported in the GMR-2 system.

5.3 Data Channel at Full-Rate, 12,0 kbit/s Radio Interface Rate (9,6 kbit/s Services (S-TCH/F9.6))

Reserved for future use, not currently supported in the GMR-2 system.

5.4 Data Channel at Full Rate, 6 kbit/s Radio Interface Rate (4,8 kbit/s Services (S-TCH/F4.8))

Reserved for future use, not currently supported in the GMR-2 system.

5.5 Data Channel at Half-Rate, 6,0 kbit/s Radio Interface Rate (4,8 kbit/s Services (S-TCH/H4.8))

Reserved for future use, not currently supported in the GMR-2 system.

5.6 Data Channel at Full-Rate, 3,0 kbit/s Radio Interface Rate (2,4 kbit/s Services (S-TCH/FR2.4))

Reserved for future use, not currently supported in the GMR-2 system.

5.7 Data Channel at Half-Rate, 3,0 kbit/s Radio Interface Rate (2,4 kbit/s Services (S-TCH/HR2.4))

The definition of a 3,0 kbit/s radio interface rate data flow for data services is given in GMR-2 04.021 [3].

5.7.1 Interface with User Unit

The user unit delivers to the encoder a bit stream organized in blocks of 30 information bits (data frames) every 10 ms. Eight such blocks are dealt with together in the coding process {d(0), . . ., d(239)}.

5.7.2 Block Code

The block of 8 × 30 or 240 information bits is not encoded, but only increased with 6 tail bits, each equal to 0, at the end of the block, as follows:

$u(k) = d(k)$ for k = 0, . . ., 239

$u(k) = 0$ for k = 240, 241, 242, 243, 244, 245

5.7.3 Convolutional Encoder

The resulting block of 246 bits {u(0), . . ., u(245)} is encoded with a rate 1/4, 64-state convolutional code defined by the following polynomials:

$G0 = 1+ D^2 + D^3 + D^4 + D^6$

$G1 = 1+ D^2 + D^3 + D^5 + D^6$

$G2 = 1+ D + D^4 + D^5 + D^6$

$G3 = 1+ D + D^2 + D^3 + D^6$

The result is a block of 984 coded bits {ca(0), ca(1), ..., ca(983)} with:

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

$ca(4k + 1) = u(k) + u(k - 2) + u(k - 3) + u(k - 5) + u(k - 6)$

$ca(4k + 2) = u(k) + u(k - 1) + u(k - 4) + u(k - 5) + u(k - 6)$

$ca(4k + 3) = u(k) + u(k - 1) + u(k - 2) + u(k - 3) + u(k - 6)$ for k=0, ..., 245

where u(k) = 0 for k < 0, so that the initial state of the encoder is zero for each sequence.

The code is punctured in such a way that the following 24 coded bits:

ca(0), ca(1), ca(82), ca(83), ca(164), ca(165),

ca(246), ca(247), ca(328), ca(329), ca(410), ca(411),

ca(492), ca(493), ca(574), ca(575), ca(656), ca(657),

ca(738), ca(739), ca(820), ca(821), ca(902), and ca(903),

are not transmitted.

The result is a block of 960 coded bits, {c(0),. . ., c(959)}.

5.7.4 Interleaving

The 960 coded bits are interleaved in accordance with the following procedures.

c(k) is split into two 480 bit blocks, ce(k) and co(k), composed of the even and odd bits of c(k):

$ce(k) = c(2k)$

$co(k) = c(2k + 1)$ for k = 0,. . ., 479

Interleaved bits ie(k) and io(k) are generated as follows. The ce(k) and co(k) are independently interleaved with an 8-slot (a slot corresponds to a subgroup of 120 bits) block diagonal interleaver.

Interleaved bits ie(k) are generated with the following procedures.

Consecutive input blocks of 480 bits of ce(k) are interleaved with an 8-slot (where a slot corresponds to a subgroup of 120 bits) block diagonal interleaver, as follows:

Subgroups of 120 bits from consecutive blocks of the 480-bit ce(k) blocks are formed, where SubGroup -6 through 0 may represent 120 coded bits from the previous data blocks or 7 bursts of initial "0" bits to start the interleaving process, SubGroup 1 is the first 120 coded bits of the current data block, SubGroup 2 is the second 120 bits of the current data block SubGroup 3 is the third 120 bits of the current data block, SubGroup 4 is the last 120 bits of the current data block, and so on for subsequent data blocks:

.

.

.

SubGroup -6: ce-6(0) ce-6(1) ce-6(2) ce-6(3) . . . ce-6(119)

.

.

.

SubGroup 0: ce0(0) ce0(1) ce0(2) ce0(3) ce0(4) ce0(5) ce0(6) ce0(7) . . . ce0(119)

SubGroup 1: ce1(0) ce1(1) ce1(2) ce1(3) ce1(4) ce1(5) ce1(6) ce1(7) . . . ce1(119)

SubGroup 2: ce2(0) ce2(1) ce2(2) ce2(3) ce2(4) ce2(5) ce2(6) ce2(7) . . . ce2(119)

SubGroup 3: ce3(0) ce3(1) ce3(2) ce3(3) ce3(4) ce3(5) ce3(6) ce3(7) . . . ce3(119)

SubGroup 4: ce4(0) ce4(1) ce4(2) ce4(3) ce4(4) ce4(5) ce4(6) ce2(7) . . . ce4(119)

.

.

.

SubGroup 8: ce8(0) ce8(1) ce8(2) ce8(3) ce8(4) ce8(5) ce8(6) . . . ce8(119)

.

.

.

From these subgroups of ce(k), new 120-bit subgroups are formed with 8-subgroup diagonal interleaving as follows:

SubGroup 1': ce-6(0) ce-5(1) ce-4(2) ... ce0(6) ce1(7) ce-6(8) ce-5(9) ce-4(10) . . . ce1(119)

SubGroup 2': ce-5(0) ce-4(1) ce-3(2) ... ce1(6) ce2(7) ce-5(8) ce-4(9) ce-3(10) . . . ce2(119)

SubGroup 3': ce-4(0) ce-3(1) ce-2(2) ... ce2(6) ce3(7) ce-4(8) ce-3(9) ce-2(10) . . . ce3(119)

SubGroup 4': ce-3(0) ce-2(1) ce-1(2) ... ce3(6) ce4(7) ce-3(8) ce-2(9) ce-1(10) . . . ce4(119)

SubGroup 5': ce-2(0) ce-1(1) ce0(2) ... ce4(6) ce5(7) ce-2(8) ce-1(9) ce0(10) . . . ce5(119)

SubGroup 6': ce-1(0) ce0(1) ce1(2) ... ce5(6) ce6(7) ce-1(8) ce0(9) ce1(10) . . . ce6(119)

SubGroup 7': ce0(0) ce1(1) ce2(2) ... ce6(6) ce7(7) ce0(8) ce1(9) ce2(10) . . . ce7(119)

SubGroup 8': ce1(0) ce2(1) ce3(2) ... ce7(6) ce8(7) ce1(8) ce2(9) ce3(10) . . . ce8(119)

SubGroup 9': ce2(0) ce3(1) ce4(2) ... ce8(6) ce9(7) ce2(8) ce3(9) ce4(10) . . . ce9(119)

SubGroup 10': ce3(0) ce4(1) ce5(2) ... ce9(6) ce10(7) ce3(8) ce4(9) ce5(10) . . . ce10(119)

SubGroup 11': ce4(0) ce5(1) ce6(2) ... ce10(6) ce11(7) ce4(8) ce5(9) ce6(10) . . . ce11(119)

.

.

.

and so on. Each of these diagonally interleaved subgroups is further interleaved with a block interleaver with 12 rows and 10 columns. The 120 bits of each 120-bit diagonally-interleaved subgroup are read into the interleaver row by row and are read out column by column to generate bursts of ie'(j,k) for k=0, . . ., 119, where j refers to the subgroup number. For example, the SubGroup 7' is written into a 12-row by 10-column matrix row-by-row, as follows:

ce0(0) ce1(1) ce2(2) ce3(3) ce4(4) ce5(5) ce6(6) ce7(7) ce0(8) ce1(9)

ce2(10) ce3(11) ce4(12) ce5(13) ce6(14) ce7(15) ce0(16) ce1(17) ce2(18) ce3(19)

.

.

.

ce4(100) ce5(101) ce6(102) ce7(103) ce0(104) ce1(105) ce2(106) ce3(107) ce4(108) ce5(109)

ce6(110) ce7(111) ce0(112) ce1(113) ce2(114) ce3(115) ce4(116) ce5(117) ce6(118) ce7(119)

The resulting 120 block-interleaved bits from SubGroup 7' are read out column-by-column, as follows:

ce0(0), ce2(10), ce4(20), . . ., ce4(100), ce6(110), ce1(1), ce3(11), . . ., ce3(99), ce5(109), ce7(119)

Interleaved bits io(k) are generated with the following procedures.

Consecutive input blocks of 480 bits of co(k) are interleaved with an 8-slot (where a slot corresponds to a subgroup of 120 bits) block diagonal interleaver, as follows:

Subgroups of 120 bits from consecutive blocks of the 480-bit co(k) blocks are formed, where SubGroup -6 through 0 may represent 120 coded bits from the previous data blocks or 7 bursts of initial "0" bits to start the interleaving process, SubGroup 1 is the first 120 coded bits of the current data block, SubGroup 2 is the second 120 bits of the current data block SubGroup 3 is the third 120 bits of the current data block, SubGroup 4 is the last 120 bits of the current data block, and so on for subsequent data blocks:

.

.

.

SubGroup -6: co-6(0) co-6(1) co-6(2) co-6(3) . . . co-6(119)

.

.

.

SubGroup 0: co0(0) co0(1) co0(2) co0(3) co0(4) co0(5) co0(6) co0(7) . . . co0(119)

SubGroup 1: co1(0) co1(1) co1(2) co1(3) co1(4) co1(5) co1(6) co1(7) . . . co1(119)

SubGroup 2: co2(0) co2(1) co2(2) co2(3) co2(4) co2(5) co2(6) co2(7) . . . co2(119)

SubGroup 3: co3(0) co3(1) co3(2) co3(3) co3(4) co3(5) co3(6) co3(7) . . . co3(119)

SubGroup 4: co4(0) co4(1) co4(2) co4(3) co4(4) co4(5) co4(6) co2(7) . . . co4(119)

.

.

.

SubGroup 8: co8(0) co8(1) co8(2) co8(3) co8(4) co8(5) co8(6) . . . co8(119)

.

.

.

From these subgroups of co(k), new 120-bit subgroups are formed with 8-subgroup diagonal interleaving as follows:

.

.

.

SubGroup 1': co-6(0) co-5(1) co-4(2) ... co0(6) co1(7) co-6(8) co-5(9) co-4(10) . . . co1(119)

SubGroup 2': co-5(0) co-4(1) co-3(2) ... co1(6) co2(7) co-5(8) co-4(9) co-3(10) . . . co2(119)

SubGroup 3': co-4(0) co-3(1) co-2(2) ... co2(6) co3(7) co-4(8) co-3(9) co-2(10) . . . co3(119)

SubGroup 4': co-3(0) co-2(1) co-1(2) ... co3(6) co4(7) co-3(8) co-2(9) co-1(10) . . . co4(119)

SubGroup 5': co-2(0) co-1(1) co0(2) ... co4(6) co5(7) co-2(8) co-1(9) co0(10) . . . co5(119)

SubGroup 6': co-1(0) co0(1) co1(2) ... co5(6) co6(7) co-1(8) co0(9) co1(10) . . . co6(119)

SubGroup 7': co0(0) co1(1) co2(2) ... co6(6) co7(7) co0(8) co1(9) co2(10) . . . co7(119)

SubGroup 8': co1(0) co2(1) co3(2) ... co7(6) co8(7) co1(8) co2(9) co3(10) . . . co8(119)

SubGroup 9': co2(0) co3(1) co4(2) ... co8(6) co9(7) co2(8) co3(9) co4(10) . . . co9(119)

SubGroup 10': co3(0) co4(1) co5(2) ... co9(6) co10(7) co3(8) co4(9) co5(10) . . . co10(119)

SubGroup 11': co4(0) co5(1) co6(2) ... co10(6) co11(7) co4(8) co5(9) co6(10) . . . co11(119)

.

.

.

and so on. Each of these diagonally interleaved subgroups is further interleaved with a block interleaver with 12 rows and 10 columns. The 120 bits of each 120-bit diagonally-interleaved subgroup are read into the interleaver row by row and are read out column by column to generate bursts of io'(j, k) for k = 0, . . ., 119, where j refers to the subgroup number. For example, the SubGroup 7' is written into a 12-row by 10-column matrix row-by-row, as follows:

co0(0) co1(1) co2(2) co3(3) co4(4) co5(5) co6(6) co7(7) co0(8) co1(9)

co2(10) co3(11) co4(12) co5(13) co6(14) co7(15) co0(16) co1(17) co2(18) co3(19)

.

.

co4(100) co5(101) co6(102) co7(103) co0(104) co1(105) co2(106) co3(107) co4(108) co5(109)

co6(110) co7(111) co0(112) co1(113) co2(114) co3(115) co4(116) co5(117) co6(118) co7(119).

The resulting 120 block-interleaved bits from SubGroup 7' are read out column-by-column, as follows:

co0(0), co2(10), co4(20), . . ., co4(100), co6(110), co1(1), co3(11), . . ., co3(99), co5(109), co7(119)

The resulting interleaved bits are re-combined to produce i(k), as follows:

i(k) = ie'(1,k), k = 0, ..., 119

i(k + 120) = io'(1,k), k = 0, ..., 119

i(k + 240) = ie'(2,k), k = 0, ..., 119

i(k + 360) = io'(2,k), k = 0, ..., 119

i(k + 480) = ie'(3,k), k = 0, ..., 119

i(k + 600) = io'(3,k), k = 0, ..., 119

i(k + 720) = ie'(4,k), k = 0, ..., 119

i(k + 840) = io'(4,k), k = 0, ..., 119

i(k + 960) = ie'(5,k), k = 0, ..., 119

i(k + 1080) = io'(5,k), k = 0, ..., 119

.

and so on.

5.7.5 Mapping on a burst

The block-diagonally interleaved 120-bit subgroups comprising i(k) are sequentially mapped onto consecutive bursts of a half-rate channel. The first burst should correspond to the 120 interleaved bits of Subroup 1' for ie(k), the second burst to Subgroup 1' of io(k), and so on. In this way, the interleaved bits for ie(k) are sequentially mapped onto the even consecutive bursts of a half-rate channel, with 120 bits per burst; and the interleaved bits for io(k) are sequentially mapped onto the odd consecutive bursts. Note that 22 bursts are needed to transmit an entire data block.

5.8 Data Channel at Quarter-Rate, 3,0 kbit/s Radio Interface Rate (2,4 kbit/s Services (S-TCH/Q2.4))

The definition of a 3,0 kbit/s radio interface rate data flow for data services is given in GMR-2 04.021 [3].

5.8.1 Interface with user unit

The user unit delivers to the encoder a bit stream organized in blocks of 30 information bits (data frames) every 10 ms. Eight such blocks are dealt with together in the coding process {d(0), . . ., d(239)}.

5.8.2 Block code

The block of 8 × 30 or 240 information bits is not encoded, but only increased with 6 tail bits, each equal to 0, at the end of the block, as follows:

u(k) = d(k) for k = 0, . . ., 239

u(k) = 0 for k = 240, 241, 242, 243, 244, 245.

5.8.3 Convolutional encoder

The resulting block of 246 bits {u(0), . . ., u(245)} is encoded with a rate 1/2, 64-state convolutional code defined by the following polynomials:

$G0 = 1+ D^2 + D^3 + D^5 + D^6$

$G1 = 1+ D + D^2 + D^3 + D^6$

The result is a block of 492 coded bits {ca(0), ca(1), ..., ca(491)} with:

ca(2k) = u(k) + u(k - 2) + u(k - 3) + u(k - 5) + u(k - 6)

ca(2k + 1) = u(k) + u(k - 1) + u(k - 2) + u(k - 3) + u(k - 6) for k = 0, ..., 245

where u(k) = 0 for k < 0, so that the initial state of the encoder is zero for each sequence.

The code is punctured in such a way that the following 12 coded bits:

ca(0), ca(41), ca(82), ca(123), ca(164), ca(205), ca(246), ca(287)

ca(328), ca (369), ca(410), and ca(451)

are not transmitted.

The result is a block of 480 coded bits, {c(0),. . ., c(479)}.

5.8.4 Interleaving

The 480 coded bits are interleaved in accordance with the following procedures.

Consecutive input blocks of 480 bits of c(k) are interleaved with an 8-slot (where a slot corresponds to a subgroup of 120 bits) block diagonal interleaver, as follows:

Subgroups of 120 bits from consecutive blocks of the 480-bit c(k) blocks are formed, where SubGroup -6 through 0 may represent 120 coded bits from the previous data blocks or 7 bursts of initial "0" bits to start the interleaving process, SubGroup 1 is the first 120 coded bits of the current data block, SubGroup 2 is the second 120 bits of the current data block SubGroup 3 is the third 120 bits of the current data block, SubGroup 4 is the last 120 bits of the current data block, and so on for subsequent data blocks:

.

SubGroup -6: c - 6(0) c - 6(1) c - 6(2) c - 6(3) . . . c - 6(119)

.

.

.

SubGroup 0: c0(0) c0(1) c0(2) c0(3) c0(4) c0(5) c0(6) c0(7) . . . c0(119)

SubGroup 1: c1(0) c1(1) c1(2) c1(3) c1(4) c1(5) c1(6) c1(7) . . . c1(119)

SubGroup 2: c2(0) c2(1) c2(2) c2(3) c2(4) c2(5) c2(6) c2(7) . . . c2(119)

SubGroup 3: c3(0) c3(1) c3(2) c3(3) c3(4) c3(5) c3(6) c3(7) . . . c3(119)

SubGroup 4: c4(0) c4(1) c4(2) c4(3) c4(4) c4(5) c4(6) c2(7) . . . c4(119)

.

.

.

SubGroup 8: c8(0) c8(1) c8(2) c8(3) c8(4) c8(5) c8(6) . . . c8(119)

.

.

.

From these subgroups of c(k), new 120-bit subgroups are formed with 8-subgroup diagonal interleaving as follows:

.

.

.

SubGroup 1': c-6(0) c-5(1) c-4(2) ... c0(6) c1(7) c - 6(8) c-5(9) c-4(10) . . . c1(119)

SubGroup 2': c-5(0) c-4(1) c-3(2) ... c1(6) c2(7) c - 5(8) c-4(9) c-3(10) . . . c2(119)

SubGroup 3': c-4(0) c-3(1) c-2(2) ... c2(6) c3(7) c - 4(8) c-3(9) c-2(10) . . . c3(119)

SubGroup 4': c-3(0) c-2(1) c-1(2) ... c3(6) c4(7) c - 3(8) c-2(9) c-1(10) . . . c4(119)

SubGroup 5': c-2(0) c-1(1) c0(2) ... c4(6) c5(7) c - 2(8) c-1(9) c0(10) . . . c5(119)

SubGroup 6': c-1(0) c0(1) c1(2) ... c5(6) c6(7) c - 1(8) c0(9) c1(10) . . . c6(119)

SubGroup 7': c0(0) c1(1) c2(2) ... c6(6) c7(7) c0(8) c1(9) c2(10) . . . c7(119)

SubGroup 8': c1(0) c2(1) c3(2) ... c7(6) c8(7) c1(8) c2(9) c3(10) . . . c8(119)

SubGroup 9': c2(0) c3(1) c4(2) ... c8(6) c9(7) c2(8) c3(9) c4(10) . . . c9(119)

SubGroup 10': c3(0) c4(1) c5(2) ... c9(6) c10(7) c3(8) c4(9) c5(10) . . . c10(119)

SubGroup 11': c4(0) c5(1) c6(2) ... c10(6) c11(7) c4(8) c5(9) c6(10) . . . c11(119)

.

.

and so on. Each of these diagonally interleaved subgroups is further interleaved with a block interleaver with 12 rows and 10 columns. The 120 bits of each 120-bit diagonally-interleaved SubGroup are read into the interleaver row by row and are read out column by column to generate bursts of i'(j,k) for k=0, . . ., 119, where j refers to the SubGroup number. For example, the SubGroup 7' is written into a 12-row by 10-column matrix row-by-row, as follows:

c0(0) c1(1) c2(2) c3(3) c4(4) c5(5) c6(6) c7(7) c0(8) c1(9)

c2(10) c3(11) c4(12) c5(13) c6(14) c7(15) c0(16) c1(17) c2(18) c3(19)

.

.

.

c4(100) c5(101) c6(102) c7(103) c0(104) c1(105) c2(106) c3(107) c4(108) c5(109)

c6(110) c7(111) c0(112) c1(113) c2(114) c3(115) c4(116) c5(117) c6(118) c7(119).

The resulting 120 block-interleaved bits from SubGroup 7' are read out column-by-column, as follows:

c0(0), c2(10), c4(20), . . ., c4(100), c6(110), c1(1), c3(11), . . ., c3(99), c5(109), c7(119)

So

i(k) = i'(1,k), k = 0, ..., 119

i(k + 120) = i'(2,k), k = 0, ..., 119

i(k + 240) = i'(3,k), k = 0, ..., 119

.

.

.

and so on.

Notice that a 7-slot delay exists in the diagonally interleaved subgroups (i.e., all the c1(k) bits are not received until SubGroup 8'; all of the bits from c1(k) through c4(k), which represents the "current data block" are not received until SubGroup 11'.

5.8.5 Mapping on a Burst

The diagonally interleaved bursts, i(k), are sequentially mapped onto consecutive bursts of a quarter-rate channel, with 120 bits per burst. The first burst should correspond to the 120 interleaved bits of SubGroup 1' (i.e.,. it contains coded information from the previous 2 data blocks and current data block). Note that 11 slots are needed to transmit an interleaved data block.

5.9 Robust speech channel at half-rate (S-TCH/HRS)

The vocoder delivers to the channel encoder a sequence of blocks of data. In the case of a half-rate robust speech TCH, one block of data corresponds to one 20 ms speech frame. Each block contains 72 bits {d(0), . . ., d(71)}. The first 12 {d(0), . . ., d(11)} are class I bits. The next 33 {d(12), . . ., d(44)} are class II bits. The last 27 {d(45), . . ., d(71)} are class III bits.

The bits delivered by the speech coder must be accordingly arranged before channel coding may be performed.

5.9.1 Parity and tailing for a speech frame

Parity Bits
A parity code is applied to the class I bits. Six parity bits are defined in such a way that in GF(2), the binary polynomial $d(0)D^{17} + \ldots + d(11)D^6 + p(0)\,D^5 + \ldots + p(5)$, when divided by $D^6 + D^5 + D^3 + D^2 + 1$ yields a remainder equal to $D^5 + D^4 + D^3 + D^2 + D + 1$. This description of the remainder implies that a one's complement notation shall be used in representing the parity bits.

a) Tail Bits and Reordering
No tail bits are applied.
The information and parity bits are reordered to define 78 bits, {u(0), . . ., u(77)} in the following way:

u(k) = d(k), k = 0, . . ., 39

u(k + 40) = p(k), k = 0, 1, 2, 3, 4, 5

u(k + 46) = d(k + 40), k=0, . . ., 31

5.9.1 Void

5.9.2 Convolutional encoder

Bits {u(0), . . ., u(50)} are encoded with a rate 1/4, 64-state convolutional code defined by the polynomials:

$G0 = 1 + D^2 + D^3 + D^4 + D^6$

$G1 = 1 + D^2 + D^3 + D^5 + D^6$

$G2 = 1 + D + D^4 + D^5 + D^6$

$G3 = 1 + D + D^2 + D^3 + D^6$

The coded bits are then defined by:

ca(4k) = u(k) + u(k - 2) + u(k - 3) + u(k - 4) + u(k - 6)

ca(4k + 1) = u(k) + u(k - 2) + u(k - 3) + u(k - 5) + u(k - 6)

ca(4k + 2) = u(k) + u(k - 1) + u(k - 4) + u(k - 5) + u(k - 6)

ca(4k + 3) = u(k) + u(k - 1) + u(k - 2) + u(k - 3) + u(k-6) for k=0, ..., 50

where u(k) = 0 for k < 0, so that the initial state of the encoder is zero for each sequence.

Bits {u(51), . . ., u(77)} are repeated such that:

ca(2k + 204) = u(k + 51), k = 0, . . . , 26

ca(2k + 205) = u(k + 51), k = 0, . . . , 26

The resulting 258 bits {ca(0), . . ., ca(257)} are punctured in such a way that the following 18 bits are not transmitted:

ca(0), ca(1), ca(20), ca(21), ca(40), ca(41), ca(60), ca(61), ca(80), ca(81)

ca(100), ca(101), ca(120), ca(121), ca(140), ca(141), ca(160), and ca(161)

The result is a block of 240 coded bits {c(0), . . ., c(239)}.

5.9.3 Interleaving

The 240 coded bits are interleaved in accordance with the following procedure.

c(k) is split into two 120 bit blocks, ce(k) and co(k), composed of the even and odd bits of c(k):

ce(k) = c(2k)

co(k) = c(2k + 1) for k=0,. . ., 119

Interleaved bits ie(k) and io(k) are generated as follows. The ce(k) and co(k) are independently interleaved with a 3-slot (a slot corresponds to a subgroup of 120 bits) block diagonal interleaver:

Interleaved bits ie(k) are generated as follows. Three subgroups of 120 bits are formed with ce(k) from three speech blocks, where SubGroups 0 and 1 each represent 120 coded bits from previous voice frames or 120 initial "0" bits to start the interleaving process; SubGroup 2 represents 120 coded bits from the current speech frame; SubGroup 2 represents 120 coded bits from the next speech frame, and so on:

SubGroup 0: ce0(0) ce0(1) ce0(2) ce0(3) ce0(4) . . . ce0(118) ce0(119)

SubGroup 1: ce1(0) ce1(1) ce1(2) ce1(3) ce1(4) . . . ce1(118) ce1(119)

SubGroup 2: ce2(0) ce2(1) ce2(2) ce2(3) ce2(4) . . . ce2(118) ce2(119)

SubGroup 3: ce3(0) ce3(1) ce3(2) ce3(3) ce3(4) . . . ce3(118) ce3(119)

.

.

.

Note that because there are three subgroups of 120 bits from ce(k), the interleaving actually spans three coded voice frames. From these subgroups of 120 bits, new bursts are formed with 3-subgroup diagonal interleaving as follows:

SubGroup 1': ce0(0) ce1(1) ce2(2) ce0(3) ce1(4) ce2(5) . . . ce0(117) ce1(118) ce2(119)

SubGroup 2': ce1(0) ce2(1) ce3(2) ce1(3) ce2(4) ce3(5) . . . ce1(117) ce2(118) ce3(119)

SubGroup 3': ce2(0) ce3(1) ce4(2) ce2(3) ce3(4) ce4(5) . . . ce2(117) ce3(118) ce4(119)

SubGroup 4': ce3(0) ce4(1) ce5(2) ce3(3) ce4(4) ce5(5) . . . ce3(117) ce4(118) ce5(119)

.

.

.

and so on. Each of these diagonally interleaved subgroups is further interleaved with a block interleaver with 12 rows and 10 columns. The 120 bits of each 120-bit diagonally-interleaved subgroup are read into the interleaver row by row and are read out column by column-to generate a bursts of ie(j,k) for k=0, ..., 119, where j refers to the subgroup number. For example, the SubGroup 1' is written into a 12-row by 10-column matrix row-by-row, as follows:

ce0(0) ce1(1) ce2(2) ce0(3) ce1(4) ce2(5) ce0(6) ce1(7) ce2(8) ce0(9)

ce1(10) ce2(11) ce0(12) ce1(13) ce2(14) ce0(15) ce1(16) ce2(17) ce0(18) ce1(19)

.

.

.

ce1(100) ce2(101) ce0(102) ce1(103) ce2(104) ce0(105) ce1(106) ce2(107) ce0(108) ce1(109)

ce2(110) ce0(111) ce1(112) ce2(113) ce0(114) ce1(115) ce2(116) ce0(117) ce1(118) ce2(119)

The resulting 120 block-interleaved bits from SubGroup 1' are read out column-by-column, as follows:

ce0(0), ce1(10), ce2(20), . . ., ce0(90), ce1(100), ce2(110), ce1(1), ce2(11), . . ., ce0(99), ce1(109), ce2(119)

Interleaved bits io(k) are similarly generated as follows. Three subgroups of 120 bits are formed with co(k) from three speech blocks, where SubGroups 0 and 1 each represent 120 coded bits from previous voice frame or 120 initial "0" bits to start the interleaving process; subgroup 2 represents 120 coded bits from the current speech frame; subgroup 3 represents 120 coded bits from the next speech frame, and so on:

SubGroup 0: co0(0) co0(1) co0(2) co0(3) co0(4) . . . co0(118) co0(119)

SubGroup 1: co1(0) co1(1) co1(2) co1(3) co1(4) . . . co1(118) co1(119)

SubGroup 2: co2(0) co2(1) co2(2) co2(3) co2(4) . . . co2(118) co2(119)

SubGroup 3: co3(0) co3(1) co3(2) co3(3) co3(4) . . . co3(118) co3(119)

.

.

.

Note that because there are three subgroups of 120 bits from co(k), the interleaving actually spans three coded voice frames. From these subgroups of 120 bits, new bursts are formed with 3-subgroup diagonal interleaving as follows:

SubGroup 1': co0(0) co1(1) co2(2) co0(3) co1(4) co2(5) . . . co0(117) co1(118) co2(119)

SubGroup 2': co1(0) co2(1) co3(2) co1(3) co2(4) co3(5) . . . co1(117) co2(118) co3(119)

SubGroup 3': co2(0) co3(1) co4(2) co2(3) co3(4) co4(5) . . . co2(117) co3(118) co4(119)

SubGroup 4': co3(0) co4(1) co5(2) co3(3) co4(4) co5(5) . . . co3(117) co4(118) co5(119)

.

.

.

and so on. Each of these diagonally interleaved subgroups is further interleaved with a block interleaver with 12 rows and 10 columns. The 120 bits of each 120-bit diagonally-interleaved subgroup are read into the interleaver row by row and are read out column by column to generate a bursts of io(j,k) for k=0, . . ., 119, where j refers to the subgroup number. For example, the SubGroup 1' is written into a 12-row by 10-column matrix row-by-row, as follows:

co0(0) co1(1) co2(2) co0(3) co1(4) co2(5) co0(6) co1(7) co2(8) co0(9)

co1(10) co2(11) co0(12) co1(13) co2(14) co0(15) co1(16) co2(17) co0(18) co1(19)

.

.

.

co1(100) co2(101) co0(102) co1(103) co2(104) co0(105) co1(106) co2(107) co0(108) co1(109)

co2(110) co0(111) co1(112) co2(113) co0(114) co1(115) co2(116) co0(117) co1(118) co2(119)

The resulting 120 block-interleaved bits from SubGroup 1' are read out column-by-column, as follows:

co0(0), co1(10), co2(20), . . ., co0(90), co1(100), co2(110), co1(1), co2(11), . . ., co0(99), co1(109), co2(119)

The resulting interleaved bits are re-combined to produce i(k), as follows:

i(k) = ie(1,k), k = 0, ..., 119

i(k + 120) = io(1,k), k = 0, ..., 119

i(k + 240) = ie(2,k), k = 0, ..., 119

i(k + 360) = io(2,k), k = 0, ..., 119

i(k + 480) = ie(3,k), k = 0, ..., 119

i(k + 600) = io(3,k), k = 0, ..., 119

.

.

.

and so on.

5.9.4 Mapping on a burst

The block-diagonally interleaved 120-bit subgroups comprising i(k) are sequentially mapped onto consecutive bursts of a half-rate channel. The first burst should correspond to the 120 interleaved bits of subgroup 1' for ie(k), the second burst to SubGroup 1' of io(k), and so on. In this way, the interleaved bits for ie(k) are sequentially mapped onto the even consecutive bursts of a half-rate channel, with 120 bits per burst; and the interleaved bits for io(k) are sequentially mapped onto the odd consecutive bursts. Note that 6 bursts are needed to transmit an entire voice frame.

5.10 Basic speech channel at quarter-rate (S-TCH/QBS)

The vocoder delivers to the channel encoder a sequence of blocks of data. In the case of a quarter-rate basic speech TCH, one block of data corresponds to one 20 ms speech frame. Each block contains 72 bits {d(0), . . ., d(71)}. The first 12 {d(0), . . ., d(11)} are class I bits. The next 33 {d(12), . . ., d(44)} are class II bits. The last 27 {d(45), . . ., d(71)} are class III bits.

The bits delivered by the speech coder must be accordingly arranged before channel coding may be performed.

5.10.1 Parity and tailing for a speech sub-block

a) Parity Bits
A parity code is applied to the class I bits. Six parity bits are defined in such a way that in GF(2), the binary polynomial $d(0)D^{17} + \ldots + d(11)D^6 + p(0)\,D^5 + \ldots + p(5)$, when divided by $D^6 + D^5 + D^3 + D^2 + 1$ yields a remainder equal to $D^5 + D^4 + D^3 + D^2 + D + 1$. This description of the remainder implies that a one's complement notation shall be used in representing the parity bits.

b) Tail Bits and Reordering
No tail bits are applied.
The information and parity bits are reordered to define 78 bits, {u(0), . . ., u(77)} in the following way:

$u(k) = d(k), \quad k = 0, \ldots, 39$

$u(k + 40) = p(k), \quad k = 0, 1, 2, 3, 4, 5$

$u(k + 46) = d(k + 40), \quad k = 0, \ldots, 31$

5.10.2 Convolutional encoder

Bits {u(0), . . ., u(50)} are encoded with a rate 1/2, 64-state convolutional code defined by the polynomials:

$G0 = 1 + D^2 + D^3 + D^5 + D^6$

$G1 = 1 + D + D^2 + D^3 + D^6$

The coded bits are then defined by:

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

$ca(2k + 1) = u(k) + u(k - 1) + u(k - 2) + u(k - 3) + u(k - 6) \quad \text{for } k = 0,..., 50$

where $u(k) = 0$ for $k < 0$, so that the initial state of the encoder is zero for each sequence.

Bits {u(51), . . ., u(77)} are simply appended such that:

$ca(k + 102) = u(k + 51), \quad k = 0,. . ., 26$

The resulting 129 bits {ca(0), . . ., ca(128)} are punctured in such a way that the following 9 bits are not transmitted:

ca(0), ca(10), ca(20), ca(30), ca(40), ca(50), ca(60), ca(70), ca(80)

The result is a block of 120 coded bits {c(0), . . ., c(119)}

5.10.3 Interleaving

The 120 coded bits are interleaved with a 3-slot (a slot corresponds to a subgroup of 120 bits) block diagonal interleaver, in accordance with the following procedure.

Three subgroups of 120 bits are formed with c(k) from three speech blocks, where SubGroups 0 and 1 each represent 120 coded bits from the previous voice frame or 120 initial "0" bits to start the interleaving process; subgroup 2 represents 120 coded bits from the current speech frame; subgroup 3 represents 120 coded bits from the next speech frame, and so on:

SubGroup 0: c0(0) c0(1) c0(2) c0(3) c0(4) . . . c0(118) c0(119)

SubGroup 1: c1(0) c1(1) c1(2) c1(3) c1(4) . . . c1(118) c1(119)

SubGroup 2: c2(0) c2(1) c2(2) c2(3) c2(4) . . . c2(118) c2(119)

SubGroup 3: c3(0) c3(1) c3(2) c3(3) c3(4) . . . c3(118) c3(119)

.

.

.

Note that because there are three subgroups of 120 bits from c(k), the interleaving actually spans three coded voice frames. From these subgroups of 120 bits, new bursts are formed with 3-subgroup diagonal interleaving as follows:

SubGroup 1': c0(0) c1(1) c2(2) c0(3) c1(4) c2(5) . . . c0(117) c1(118) c2(119)

SubGroup 2': c1(0) c2(1) c3(2) c1(3) c2(4) c3(5) . . . c1(117) c2(118) c3(119)

SubGroup 3': c2(0) c3(1) c4(2) c2(3) c3(4) c4(5) . . . c2(117) c3(118) c4(119)

.

.

and so on. Each of these diagonally interleaved subgroups is further interleaved with a block interleaver with 12 rows and 10 columns. The 120 bits of each 120-bit diagonally-interleaved subgroup are read into the interleaver row by row and are read out column by column to generate bursts of i'(j,k) for k=0, . . ., 119, where j refers to the subgroup number. For example, the SubGroup 1' is written into a 12-row by 10-column matrix row-by-row, as follows:

c0(0) c1(1) c2(2) c0(3) c1(4) c2(5) c0(6) c1(7) c2(8) c0(9)

c1(10) c2(11) c0(12) c1(13) c2(14) c0(15) c1(16) c2(17) c0(18) c1(19)

.

.

.

c1(100) c2(101) c0(102) c1(103) c2(104) c0(105) c1(106) c2(107) c0(108) c1(109)

c2(110) c0(111) c1(112) c2(113) c0(114) c1(115) c2(116) c0(117) c1(118) c2(119).

The resulting 120 block-interleaved bits from SubGroup 1' are read out column-by-column, as follows:

c0(0), c1(10), c2(20), . . ., c0(90), c1(100), c2(110), c1(1), c2(11), . . ., c0(99), c1(109), c2(119).

So:

i(k) = i'(1,k), k = 0,...,119

i(k + 120) = i'(2,k), k = 0,...,119

i(k + 240) = i'(3,k), k = 0,...,119

.

.

.

and so on.

5.10.4 Mapping on a burst

The block-diagonally interleaved 120-bit subgroups comprising i(k) are sequentially mapped onto consecutive bursts of a quarter-rate channel. The first burst should correspond to the 120 interleaved bits of SubGroup 1', the second burst to SubGroup 2', and so on. Note that 3 bursts are needed to transmit an entire voice frame.

5.11 Low rate speech channel at eighth-rate (S-TCH/ELS)

Reserved for future use, not currently supported in the GMR-2 system.

6 Control Channels

6.1 Slow Associated Control Channels

Satellite Slow, S-TCH/F Associated, Control Channel (S-SACCH/TF)

Satellite Slow, S-TCH/H Associated, Control Channel (S-SACCH/TH)

Satellite Slow, S-TCH/Q Associated, Control Channel (S-SACCH/TQ)

Satellite Slow, S-TCH/E Associated, Control Channel (S-SACCH/TE)

Satellite Slow, S-SDCCH/E Associated, Control Channel (S-SACCH/CE)

Satellite Slow, S-SDCCH/Q Associated, Control Channel (S-SACCH/CQ)

The S-SACCH/THR and S-SACCH/CHR channels are described in clause 6.12 of the present document.

6.1.1 Block constitution

The message delivered to the encoder has a fixed size of 184 information bits {d(0),...,d(183)}.

6.1.2 Block Code

The block of 184 information bits is encoded, using a shortened binary cyclic code (Fire code), with the following generator polynomial:

$$g(D) = (D^{23} + 1) \times (D^{17} + D^3 + 1)$$

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

$$u(0)D^{223} + u(1)D^{222} + \ldots + u(222)D + u(223)$$

where:

$u(k) = d(k)$ for $k = 0,\ldots, 183$ are information bits

and:

$u(k + 184) = p(k)$ for $k = 0,\ldots, 39$ are parity bits,

when divided by g(D) yields a remainder equal to $1 + D + D^2 + \ldots + D^{39}$. This description of the remainder implies that a one's complement notation shall be used in representing the Fire code parity bits.

The result is a block of 224 bits which are split into four 56-bit subblocks u1(k), u2(k), u3(k), and u4(k) with six tail bits added to each, as follows:

u1(k) = u(k) for k = 0, . . ., 55

u1(k) = 0 for k = 56, . . ., 61

u2(k) = u(k + 56) for k = 0, . . ., 55

u2(k) = 0 for k = 56, . . ., 61

u3(k) = u(k + 112) for k = 0, . . ., 55

u3(k) = 0 for k = 56, . . ., 61

and:

u4(k) = u(k + 168) for k = 0, . . ., 55

u4(k) = 0 for k = 56,. . . , 61

Note that each subblock contains 56 bits of the original 224 bit sequence and 6 tail bits.

6.1.3 Convolutional encoder

Each of the resulting subblocks of 62 bits is encoded with a rate 1/2, 64-state convolutional code defined by the following polynomials:

$G0 = 1 + D^2 + D^3 + D^5 + D^6$

$G1 = 1 + D + D^2 + D^3 + D^6$

The result is four subblocks of 124 coded bits {c1'(0), . . . , c1'(123)}, {c2'(0), . . . , c2'(123)}, {c3'(0), . . . , c3'(123)}, and {c4'(0), . . . , c4'(123)}, where

c1a(2k) = u1(k) + u1(k - 2) + u1(k - 3) + u1(k - 5) + u1(k - 6)

c1a(2k + 1) = u1(k) + u1(k - 1) + u1(k - 2) + u1(k - 3) + u1(k - 6) for k = 0, ..., 61,

c2a(2k) = u2(k) + u2(k - 2) + u2(k - 3) + u2(k - 5) + u2(k - 6)

c2a(2k + 1) = u2(k) + u2(k - 1) + u2(k - 2) + u2(k - 3) + u2(k - 6) for k = 0, ..., 61,

c3a(2k) = u3(k) + u3(k - 2) + u3(k - 3) + u3(k - 5) + u3(k - 6)

c3a(2k + 1) = u3(k) + u3(k - 1) + u3(k - 2) + u3(k - 3) + u3(k - 6) for k = 0, ..., 61,

and:

c4a(2k) = u4(k) + u4(k - 2) + u4(k - 3) + u4(k - 5) + u4(k - 6)

c4a(2k + 1) = u4(k) + u4(k - 1) + u4(k - 2) + u4(k - 3) + u4(k - 6) for k = 0, ..., 61.

In the definition of c1a(k), c2a(k), c3a(k), and c4a(k), note that u1(k), u2(k), u3(k), and u4(k) are assumed to equal 0 for $k < 0$, so that the initial states of the encoders are zero for each sequence.

For each of these four subblocks, the code is punctured in such a way that the following 4 bits are not transmitted:

cia(0), cia(30), cia(60), and cia(90) for i = 1, 2, 3, 4

The remaining four subblocks which contain 480 coded bits are denoted by {c1(0), . . . , c1(119)}, {c2(0), . . . , c2(119)}, {c3(0), . . . , c3(119)}, and {c4(0), . . . , c4(119)}.

6.1.4 Interleaving

The 480 coded bits are interleaved in accordance with the following procedures.

From the four subgroups of 120 bits represented by ci(k), i = 1, 2, 3, 4 new bursts are formed with 4-subgroup diagonal interleaving, as follows:

SubGroup 1': c1(0) c2(1) c3(2) c4(3) c1(4) c2(5) c3(6) c4(7) . . . c1(116) c2(117) c3(118) c4(119)

SubGroup 2': c2(0) c3(1) c4(2) c1(3) c2(4) c3(5) c4(6) c1(7) . . . c2(116) c3(117) c4(118) c1(119)

SubGroup 3': c3(0) c4(1) c1(2) c2(3) c3(4) c4(5) c1(6) c2(7) . . . c3(116) c4(117) c1(118) c2(119)

SubGroup 4': c4(0) c1(1) c2(2) c3(3) c4(4) c1(5) c2(6) c3(7) . . . c4(116) c1(117) c2(118) c3(119)

Each of these diagonally interleaved subgroups is further interleaved with a block interleaver with 12 rows and 10 columns. The 120 bits of each 120-bit diagonally-interleaved subgroup are read into the interleaver row-by-row and are read out column-by-column to generate bursts ij(k) for k = 0, ..., 119, where j = 1, 2, 3, 4 refers to the subgroup number. For example, the SubGroup 1' is written into a 12-row by 10-column matrix row-by-row, as follows:

c1(0) c2(1) c3(2) c4(3) c1(4) c2(5) c3(6) c4(7) c1(8) c2(9)

c3(10) c4(11) c1(12) c2(13) c3(14) c4(15) c1(16) c2(17) c3(18) c4(19)

c1(20) c2(21) c3(22) c4(23) c1(24) c2(25) c3(26) c4(27) c1(28) c2(29)

.

.

.

c3(110) c4(111) c1(112) c2(113) c3(114) c4(115) c1(116) c2(117) c3(118) c4(119)

The resulting 120 block-interleaved bits from SubGroup 1' are read out column-by-column, as follows:

c1(0), c3(10), c1(20), c3(30), ..., c1(100), c3(110), c2(1), c4(11), c2(21), c4(31), ..., c2(109), c4(119)

The resulting 480 interleaved bits recombined to produce i(j,k), as follows:

i(0,k) = i1(k) for k = 0, . . ., 119

i(1,k) = i2(k) for k = 0, . . ., 119

i(2,k) = i3(k) for k = 0, . . ., 119

i(3,k) = i4(k) for k = 0, . . ., 119

6.1.5 Mapping on a burst

The resulting 480 interleaved bits for each message are consecutively mapped onto four bursts of the corresponding-rate SACCH channel as described in table 9.0-4 of GMR-2 05.002 [5]. The mapping is given by the rule:

e(B, j) = i(B, j) for j = 0, ..., 119 with B = 0, 1, 2, 3

6.2 Fast Associated Control Channels

Satellite Fast S-TCH/HES Associated Control Channel (S-FACCH/HES)

Satellite Fast S-TCH/QBS Associated Control Channel (S-FACCH/QBS)

Satellite Fast S-TCH/ELS Associated Control Channel (S-FACCH/ELS)

Satellite Fast S-TCH/H4.8 Associated Control Channel (S-FACCH/H4.8)

Satellite Fast S-TCH/Q2.4 Associated Control Channel (S-FACCH/Q2.4)

Satellite Fast S-TCH/F9.6 Associated Control Channel (S-FACCH/F9.6)

The S-FACCH/HRS and S-FACCH/HR2.4 channels are described in clause 6.3 of the present document.

6.2.1 Block constitution

The message delivered to the encoder has a fixed size of 184 information bits {d(0),...,d(183)}.

6.2.2 Block code

The block of 184 information bits is encoded, using a shortened binary cyclic code (Fire code), with the following generator polynomial:

$$g(D) = (D^{23} + 1) \times (D^{17} + D^3 + 1)$$

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

$$u(0)D^{223} + u(1)D^{222} + \ldots + u(222)D + u(223),$$

where:

$u(k) = d(k)$ for $k = 0, \ldots, 183$ are information bits

and:

$u(k + 184) = p(k)$ for $k = 0, \ldots, 39$ are parity bits,

when divided by g(D) yields a remainder equal to $1 + D + D^2 + \ldots + D^{39}$. This description of the remainder implies that a one's complement notation shall be used in representing the Fire code parity bits.

The result is a block of 224 bits which are split into four 56-bit subblocks u1(k), u2(k), u3(k), and u4(k) with six tail bits added to each, as follows:

$u1(k) = u(k)$ for $k = 0, \ldots, 55$

$u1(k) = 0$ for $k = 56, \ldots, 61$

$u2(k) = u(k + 56)$ for $k = 0, \ldots, 55$

$u2(k) = 0$ for $k = 56, \ldots, 61$

$u3(k) = u(k + 112)$ for $k = 0, \ldots, 55$

$u3(k) = 0$ for $k = 56, \ldots, 61$

and:

$u4(k) = u(k + 168)$ for $k = 0, \ldots, 55$

$u4(k) = 0$ for $k = 56, \ldots, 61$

Note that each subblock contains 56 bits of the original 224 bit sequence and 6 tail bits.

6.2.3 Convolutional Encoder

Each of the resulting subblocks of 62 bits is encoded with a rate 1/2, 64-state convolutional code defined by the following polynomials:

$G0 = 1 + D^2 + D^3 + D^5 + D^6$

$G1 = 1 + D + D^2 + D^3 + D^6$

The result is four subblocks of 124 coded bits {c1'(0), . . . , c1'(123)}, {c2'(0), . . . , c2'(123)}, {c3'(0), . . . , c3'(123)}, and {c4'(0), . . . , c4'(123)}, where:

c1a(2k) = u1(k) + u1(k - 2) + u1(k - 3) + u1(k - 5) + u1(k - 6)

c1a(2k + 1) = u1(k) + u1(k - 1) + u1(k - 2) + u1(k - 3) + u1(k - 6) for k = 0, ..., 61

c2a(2k) = u2(k) + u2(k - 2) + u2(k - 3) + u2(k - 5) + u2(k - 6)

c2a(2k + 1) = u2(k) + u2(k - 1) + u2(k - 2) + u2(k - 3) + u2(k - 6) for k = 0, ..., 61

c3a(2k) = u3(k) + u3(k - 2) + u3(k - 3) + u3(k - 5) + u3(k - 6)

c3a(2k + 1) = u3(k) + u4(k - 1) + u3(k - 2) + u3(k - 3) + u3(k - 6) for k = 0, ..., 61

and:

c4a(2k) = u4(k) + u4(k - 2) + u4(k - 3) + u4(k - 5) + u4(k - 6)

c4a(2k + 1) = u4(k) + u4(k - 1) + u4(k - 2) + u4(k - 3) + u4(k - 6) for k = 0, ..., 61

In the definition of c1a(k), c2a(k), c3a(k), and c4a(k), note that u1(k), u2(k), u3(k), and u4(k) are assumed to equal 0 for $k < 0$, so that the initial states of the encoders are zero for each sequence.

For each of these four subblocks, the code is punctured in such a way that the following 4 bits are not transmitted:

cia(0), cia(30), cia(60), and cia(90) for i = 1, 2, 3, 4

The remaining four subblocks which contain 480 coded bits are denoted by

{c1(0), . . . , c1(119)}, {c2(0), . c2(119)}, {c3(0), . . . , c3(119)}, and {c4(0), . . . , c4(119)}

6.2.4 Interleaving

The 480 coded bits are interleaved in accordance with the following procedures.

Each of the four 120-bit subblocks represented by c1(k), c2(k), c3(k), and c4(k) is interleaved with a block interleaver with 12 rows and 10 columns. The procedure is to read the 120 bits of each subblock into the interleaver row by row and to read out the 120 bits for each interleaved subblock column by column. This results in four 120-bit individually interleaved subblocks i1(k), i2(k), i3(k), and i4(k). For example, c1(k) is written into a 12-row by 10-column matrix row-by-row, as follows:

c1(0) c1(1) c1(2) c1(3) c1(4) c1(5) c1(6) c1(7) c1(8) c1(9)

c1(10) c1(11) c1(12) c1(13) c1(14) c1(15) c1(16) c1(17) c1(18) c1(19)

.

.

.

c1(100) c1(101) c1(102) c1(103) c1(104) c1(105) c1(106) c1(107) c1(108) c1(109)

c1(110) c1(111) c1(112) c1(113) c1(114) c1(115) c1(116) c1(117) c1(118) c1(119)

The resulting 120 block-interleaved bits from c1(k) are read out column-by-column, as follows:

c1(0), c1(10), c1(20), c(30), . . ., c1(100), c1(110), c1(1), c1(11), c1(21), . . ., c1(109), c1(119)

The resulting 480 interleaved bits recombined to produce i(j,k), as follows:

i(0,k) = i1(k) for k = 0, . . ., 119

i(1,k) = i2(k) for k = 0, . . ., 119

i(2,k) = i3(k) for k = 0, . . ., 119

i(3,k) = i4(k) for k = 0, . . ., 119.

At this point for S-FACCH/QBS, the 480 bits of interleaved FACCH data, i(j,k) are inserted into the quarter-rate basic voice interleaver (at the beginning of a speech block) and further interleaved in accordance with clause 5.10.3.

For S-FACCH/Q2.4, the 480 bits of interleaved FACCH data, i(j,k) are inserted into the quarter-rate 2.4 kbps data/fax interleaver (at the beginning of a data block) and further interleaved in accordance with clause 5.8.4.

Additional interleaving for S-FACCH/HES, S-FACCH/ELS, S-FACCH/F9.6, and S-FACCH/H4.8 is reserved.

6.2.5 Mapping on a burst

An S-FACCH/QBS frame is mapped on consecutive bursts as specified for the S-TCH quarter rate basic voice channel as described in clause 5.10.4. Note that 4 slots of speech data are replaced with FACCH/QBS data per clause 6.2.4; however, 6 slots are needed to transmit an interleaved FACCH/Q message due to the additional "voice" interleaving.

An S-FACCH/Q2.4 frame is mapped on consecutive bursts as specified for the S-TCH quarter-rate data/fax channel as described in clause 5.8.4. Note that 4 slots of TCH/Q2.4 data are replaced with FACCH/Q2.4 data per clause 4.2.4; however, 11 slots are needed to transmit an interleaved FACCH/Q2.4 message due to the additional "data/fax" interleaving.

The mappings for S-FACCH/HES, S-FACCH/ELS, S-FACCH/H4.8, and S-FACCH/F9.6 are reserved.

6.3 Robust Fast Associated Control Channels

Satellite Fast S-TCH/HRS Associated Control Channel (S-FACCH/HRS)

Satellite Fast S-TCH/H2.4 Associated Control Channel (S-FACCH/HR2.4)

6.3.1 Block constitution

The message delivered to the encoder has a fixed size of 184 information bits {d(0), ..., d(183)}.

6.3.2 Block code

The block of 184 information bits is encoded, using a shortened binary cyclic code (Fire code), with the following generator polynomial:

$g(D) = (D^{23} + 1) \times (D^{17} + D^3 + 1)$

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

$u(0)D^{223} + u(1)D^{222} + \ldots + u(222)D + u(223)$,

where:

$u(k) = d(k)$ for $k = 0,\ldots, 183$ are information bits

and:

$u(k + 184) = p(k)$ for $k = 0,\ldots, 39$ are parity bits,

when divided by g(D) yields a remainder equal to $1 + D + D^2 + \ldots + D^{39}$. This description of the remainder implies that a one's complement notation shall be used in representing the Fire code parity bits.

The result is a block of 224 bits which are split into four 56-bit subblocks u1(k), u2(k), u3(k), and u4(k) with six tail bits added to each, as follows:

$u1(k) = u(k)$ for $k = 0, \ldots, 55$

$u1(k) = 0$ for $k = 56, \ldots, 61$

$u2(k) = u(k + 56)$ for $k = 0, \ldots, 55$

$u2(k) = 0$ for $k = 56, \ldots, 61$

$u3(k) = u(k + 112)$ for $k = 0, \ldots, 55$

$u3(k) = 0$ for $k = 56, \ldots, 61$

and:

$u4(k) = u(k + 168)$ for $k = 0, \ldots, 55$

$u4(k) = 0$ for $k = 56,\ldots, 61$

Note that each subblock contains 56 bits of the original 224 bit sequence and 6 tail bits.

6.3.3 Convolutional encoder

Each of the resulting subblocks of 62 bits is encoded with a rate 1/4, 64-state convolutional code defined by the following polynomials:

$G0 = 1 + D^2 + D^3 + D^4 + D^6$

$G1 = 1 + D^2 + D^3 + D^5 + D^6$

$G2 = 1 + D + D^4 + D^5 + D^6$

$G3 = 1 + D + D^2 + D^3 + D^6$

The result is four subblocks of 248 coded bits {c1'(0), . . . , c1'(247)}, {c2'(0), . . . , c2'(247)}, {c3'(0), . . . , c3'(247)}, and {c4'(0), . . . , c4'(247)}, where:

$c1a(4k) = u1(k) + u1(k - 2) + u1(k - 3) + u1(k - 4) + u1(k - 6)$

$c1a(4k + 1) = u1(k) + u1(k - 2) + u1(k - 3) + u1(k - 5) + u1(k - 6)$

$c1a(4k + 2) = u1(k) + u1(k - 1) + u1(k - 4) + u1(k - 5) + u1(k - 6)$

c1a(4k + 3) = u1(k) + u1(k - 1) + u1(k - 2) + u1(k - 3) + u1(k - 6) for k = 0, ..., 61

c2a(4k) = u2(k) + u2(k - 2) + u2(k - 3) + u2(k - 4) + u2(k - 6)

c2a(4k + 1) = u2(k) + u2(k - 2) + u2(k - 3) + u2(k - 5) + u2(k - 6)

c2a(4k + 2) = u2(k) + u2(k - 1) + u2(k - 4) + u2(k - 5) + u2(k - 6)

c2a(4k + 3) = u2(k) + u2(k - 1) + u2(k - 2) + u2(k - 3) + u2(k - 6) for k = 0, ..., 61,

c3a(4k) = u3(k) + u3(k - 2) + u3(k - 3) + u3(k - 4) + u3(k - 6)

c3a(4k + 1) = u3(k) + u3(k - 2) + u3(k - 3) + u3(k - 5) + u3(k - 6)

c3a(4k + 2) = u3(k) + u3(k - 1) + u3(k - 4) + u3(k - 5) + u3(k - 6)

c3a(4k + 3) = u3(k) + u3(k - 1) + u3(k - 2) + u3(k - 3) + u3(k - 6) for k = 0, ..., 61

and:

c4a(4k) = u4(k) + u4(k-2) + u4(k - 3) + u4(k - 4) + u4(k - 6)

c4a(4k + 1) = u4(k) + u4(k - 2) + u4(k - 3) + u4(k - 5) + u4(k - 6)

c4a(4k + 2) = u4(k) + u4(k - 1) + u4(k - 4) + u4(k - 5) + u4(k - 6)

c4a(4k + 3) = u4(k) + u4(k - 1) + u4(k - 2) + u4(k - 3) + u4(k - 6) for k = 0, ..., 61

In the definition of c1a(k), c2a(k), c3a(k), and c4a(k), note that u1(k), u2(k), u3(k), and u4(k) are assumed to equal 0 for $k < 0$, so that the initial states of the encoders are zero for each sequence.

For each of these four subblocks, the code is punctured in such a way that the following 8 bits are not transmitted:

cia(0), cia(1), cia(60), cia(61), cia(120), cia(121), cia(180), and cia(181) for I = 1, 2, 3, 4

The remaining four subblocks which contain 960 coded bits are denoted by

{c1(0), . . . , c1(239)}, {c2(0), . . . , c2(239)}, {c3(0), . . . , c3(239)}, and {c4(0), . . . , c4(239)}

6.3.4 Interleaving

The first stage of interleaving for half-rate robust S-FACCH shall be as follows:

The 960 coded bits are interleaved in accordance with the following procedures.

c1(k) is further split into two 120-bit blocks, c1e(k) and c1o(k), composed of the even and odd bits of c1(k):

c1e(k) = c1(2k)

c1o(k) = c1(2k + 1) for k = 0, . . ., 119

Similarly, c2(k), c3(k), and c4(k) are further split into two 120-bit blocks:

c2e(k) = c2(2k)

c2o(k) = c2(2k + 1) for k = 0, . . ., 119

c3e(k) = c3(2k)

c3o(k) = c3(2k + 1) for k = 0, . . ., 119

and:

c4e(k) = c4(2k)

c4o(k) = c4(2k + 1) for k = 0, . . ., 119

Each of the eight 120-bit subblocks represented by c1e(k), c1o(k), c2e(k), c2o(k), c3e(k), c3o(k), c4e(k), and c4o(k) is interleaved with a block interleaver with 12 rows and 10 columns. The procedure is to read the 120 bits of each subblock into the interleaver row by row and to read out the 120 bits for each interleaved subblock column by column. This results in eight 120-bit individually interleaved subblocks i1e(k), i1o(k), i2e(k), i2o(k), i3e(k), i3o(k), i4e(k), and i4o(k). For example, c1e(k) is written into a 12-row by 10-column matrix row-by-row, as follows:

c1e(0) c1e(1) c1e(2) c1e(3) c1(4) c1e(5) c1e(6) c1e(7) c1e(8) c1e(9)

c1e(10) c1e(11) c1e(12) c1e(13) c1e(14) c1e(15) c1e(16) c1e(17) c1e(18) c1e(19)

.

.

.

c1e(100) c1e(101) c1e(102) c1e(103) c1e(104) c1e(105) c1e(106) c1e(107) c1e(108) c1e(109)

c1e(110) c1e(111) c1e(112) c1e(113) c1e(114) c1e(115) c1e(116) c1e(117) c1e(118) c1e(119)

The resulting 120 block-interleaved bits from c1e(k) are read out column-by-column, as follows:

c1e(0), c1e(10), c1e(20), c(30), . . ., c1e(100), c1e(110), c1e(1), c1e(11), c1e(21), . . ., c1e(109), c1e(119)

The resulting 960 interleaved bits recombined to produce i(j,k), as follows:

i(0,2k) = i1e(k) for k = 0, . . ., 119

i(0,2k + 1) = i1o(k) for k = 0, . . ., 119

i(1,2k) = i2e(k) for k = 0, . . ., 119

i(1,2k + 1) = i2o(k) for k = 0, . . ., 119

i(2,2k) = i3e(k) for k = 0, . . ., 119

i(2,2k + 1) = i3o(k) for k = 0, . . ., 119

i(3,2k) = i4e(k) for k = 0, . . ., 119

i(3,2k + 1) = i4o(k) for k = 0, . . ., 119

At this point, the 960 bits of interleaved S-FACCH data, i(j,k) are inserted into the half-rate robust voice interleaver (at the beginning of a speech block) and further interleaved in accordance with clause 5.9.3.

For S-FACCH/HR2.4, the 960 bits of interleaved FACCH data, i(j,k), are inserted into the half-rate robust 2.4 kbps data/fax interleaver (at the beginning of a data block) and further interleaved in accordance with clause 5.7.4

6.3.5 Mapping on a burst

An S-FACCH/HR frame is mapped on consecutive bursts as specified for the S-TCH half-rate robust voice channel as described in clause 5.9.4. Note that 8 slots of speech data are replaced with S-FACCH/Q data per clause 6.3.4; however, 12 slots are needed to transmit a coded S-FACCH/HR message due to additional "voice" interleaving.

An S-FACCH/HR2.4 frame is mapped on consecutive bursts as specified for the S-TCH half-rate robust data/fax channel as described in clause 5.7.5. Note that 8 slots of TCH/HR2.4 data are replaced with FACCH/HR2.4 data per clause 6.3.4; however, 22 slots are needed to transmit an interleaved FACCH/HR2.4 message due to the additional "data/fax" interleaving.

6.4 Broadcast, Paging, and Access Grant Broadcast Channels

Broadcast Control Channels

Satellite Broadcast Control Channel (S-BCCH)

Common Control Channels

Satellite Paging Channel (S-PCH)

Satellite Access Grant Channel (S-AGCH)

6.4.1 Block constitution

The message delivered to the encoder has a fixed size of 184 information bits {d(0),...,d(183)}. It is delivered on a burst mode.

6.4.2 Block code

The block of 184 information bits is protected by 40 extra bits used for error correction and detection. These bits are added to the 184 bits according to a shortened binary cyclic code (FIRE code), using the generator polynomial:

$g(D) = (D^{23} + 1)\ (D^{17} + D^3 + 1)$

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

$u(0)D^{223} + u(1)D^{222} + ... + u(183)D^{40} + p(1)D^{38} + . . . + p(38)D + p(39)$

where {p(0), p(1), p(2), . . ., p(39)} are the parity bits, when divided by g(D) yields a remainder equal to:

$1 + D + D^2 + ... + D^{39}$

This description of the remainder implies that a one's complement notation shall be used in representing the Fire code parity bits.

Four tail bits equal to 0 are added to the information and parity bits, the result being a block of 228 bits.

$u(k) = d(k)$ for k = 0, 1, ..., 183

$u(k) = p(k - 184)$ for k = 184, 185, ..., 223

$u(k) = 0$ for k = 224, 225, 226, 227 (tail bits)

6.4.3 Convolutional Encoder

This block of 228 bits is encoded with the 1/2 rate, 16-state convolutional code defined by the polynomials:

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

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

This results in a block of 456 coded bits: {c(0), ..., c(455)} defined by:

$c(2k) = u(k) + u(k - 3) + u(k - 4)$

$c(2k + 1) = u(k) + u(k - 1) + u(k - 3) + u(k - 4)$ for k = 0, ..., 227

where $u(k) = 0$ for $k < 0$, so that the initial state of the encoder is zero for each sequence.

6.4.4 Interleaving

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

$i(B,j) = c(n,k)$ for $k = 0, 1, ..., 455$

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

$B = B0 + 4n + k \bmod (4)$

$j = 2[(49k) \bmod 57] + [(k \bmod 8)/4]$

Where [x] represents the integer of x.

The result of the reordering of bits is the same as that given for GSM BCCH, PCH, and AGCH channels per GSM 05.03 [6], clause 4.4. This can be seen from the evaluation of the bit number-index j, distributing the 456 bits over 4 blocks on even numbered bits and 4 blocks on odd numbered bits. The resulting 4 blocks are built by putting blocks with even numbered bits and blocks with odd numbered bits together into one block.

The block of coded data is interleaved "block rectangular" where a new data block starts every 4th block and is distributed over 4 blocks.

6.4.5 Mapping on a Burst

The resulting 456 interleaved bits for each message are mapped onto four consecutive bursts as described in GMR-2 05.002 [5], table 9.0.5a. The mapping is given by the rule:

$e(B,j) = i(B,j)$ for $j = 0,. . ., 56$

$e(B,63 + j) = i(B,57 + j)$ for $j = 0,. . ., 56$

$e(B,j) = 0$ for $j = 57, 58, 59, 60, 61, 62$

6.5 Standalone Dedicated Control Channel

Satellite Eighth Rate Standalone Dedicated Control Channel (S-SDCCH/E)

Satellite Quarter Rate Standalone Dedicated Control Channel (S-SDCCH/Q)

The S-SDCCH/HR channel is described in clause 6.14 of the present document.

The coding scheme used for the standalone dedicated control channel messages is the same as that for the Slow Associated Control Channels, as described in clause 6.1.

For the S-SDCCH/E, the resulting 480 interleaved bits for each S-SDCCH/E message are consecutively mapped onto four bursts of an eighth-rate channel as described in GMR-2 05.002 [5], table 9.0.5b.

For the S-SDCCH/Q, the mapping is reserved.

6.6 Random Access Channel

Satellite Random Access Channel (S-RACH)

The burst carrying the random access uplink message has a different structure detailed in clause 7.2 of GMR-2 05.002 [5]. It contains 28 information bits d(0), . . ., d(27), six parity bits, and six tailing bits for a total of 40 bits.

Six parity bits p(0) to p(5) are defined in such a way that in GF(2) the binary polynomial $d(0)D^{33} + . . . + d(27)D^6 + p(0) D^5 + . . . + p(5)$, when divided by $D^6 + D^5 + D^3 + D^2 + D + 1$ yields a remainder equal to $D^5 + D^4 + D^3 + D^2 + D + 1$. This description of the remainder implies that a one's complement notation shall be used in representing the parity bits.

The six bits of the SBIC, b(0) to b(5) are added bit-wise modulo 2 to the six parity bits, p(0) to p(5). This results in six color bits, c(0) to c(5) defined as:

$c(k) = b(k) + p(k)$ for $k = 0, 1, 2, 3, 4, 5$

where b(0) is the MSB of the SBIC and b(5) is the LSB of the SBIC.

The six tail bits, which are equal to zero, are then appended yielding a total of 40 bits.

This defines {u(0), . . ., u(39)} by:

$u(k) = d(k)$ for $k = 0, 1, . . ., 27$

$u(28 + k) = c(k)$ for $k = 0, 1, 2, 3, 4, 5$

$u(k) = 0$ for $k = 34, . . ., 39.$

The bits {e(0), . . ., e(119)} are obtained by a 64-state rate 1/3 convolutional code defined by the polynomials:

$G0 = 1 + D + D^2 + D^3 + D^6$

$G1 = 1 + D^2 + D^3 + D^5 + D^6$

$G2 = 1 + D + D^2 + D^4 + D^6$

and with:

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

$e(3k + 1) = u(k) + u(k - 2) + u(k - 3) + u(k - 5) + u(k - 6)$

$e(3k + 2) = u(k) + u(k - 1) + u(k - 2) + u(k - 4) + u(k - 6)$ for $k = 0,. . ., 39,$

where $u(k) = 0$ for $k < 0$, so that the initial state of the encoder is zero for each sequence.

The 120 bits obtained from the output of the coder are transmitted in the data part of the normal burst portion of the access burst for the S-RACH as defined in GMR-2 05.002 [5], table 9.0.5c. No interleaving is employed.

6.7 Synchronization Channel

Satellite Synchronization Channel (S-SCH)

The burst carrying the synchronization message has a different structure detailed in GMR-2 05.002 [5], clause 7.2. The burst contains 25 information bits d(0) to d(24), 10 parity bits p(0) to p(9), and 4 tail bits for a total of 39 bits. The precise ordering of the 25 information bits is given in GMR-2 04.008 [2], clause 10.1.

The ten parity bits are defined in such a way that in GF(2) the binary polynomial $d(0)D^{34} + . . . + d(24)D^{10} + p(0)\ D^9 + . . . + p(9)$, when divided by $D^{10} + D^8 + D^6 + D^5 + D^4 + D^2 + 1$ yields a remainder equal to $D^9 + D^8 + D^7 + D^6 + D^5 + D^4 + D^3 + D^2 + D + 1$. This description of the remainder implies that a one's complement notation shall be used in representing the parity bits.

Thus the encoded bits {u(0), . . ., u(38)} are:

$u(k) = d(k)$ for $k = 0, 1, . . ., 24$

$u(25 + k) = p(k)$ for $k = 0, 1, . . ., 9$

$u(35 + k) = 0$ for $k = 0, 1, 2, 3$

The bits {e(0), . . ., e(77)} are obtained by a rate 1/2 convolutional code defined by the following polynomials:

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

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

and with:

$e(2k) = u(k) + u(k - 3) + u(k - 4)$

$e(2k+1) = u(k) + u(k - 1) + u(k - 3) + u(k - 4)$ for k = 0, . . ., 77

where u(k) = 0 for k < 0, so that the initial state of the encoder is zero for each sequence.

The 78 bits obtained from the output of the coder are transmitted in the data portion of one S-SCH burst as defined in GMR-2 05.002 [5], table 9.0.5a. No interleaving is employed.

6.8 Handover Access Burst

Void

6.9 High Penetration Alerting Channel

Satellite High Penetration Alerting Channel (S-HPACH)

The burst carrying the High Penetration Alerting messages has a different structure detailed in GMR-2 05.002 [5], clause 5.3.

6.9.1 IMSI version

The coding/interleaving for the S-HPACH IMSI version, as indicated in the S-BCCH message is specified below.

9.9.1.1 Block constitution

The message delivered to the encoder is composed of 53 information bits d(0), ..., d(52).

6.9.1.2 Parity Bits and Tail Bits

Five parity bits are defined in such a way that in GF(2) the binary polynomial $d(0)D^{57} + d(1)D^{56} + . . . + d(6)D^5 + p(0) D^4 + . . . + p(4)$, when divided by $D^5 + D^2 + 1$ yields a remainder equal to $D^4 + D^3 + D^2 + D + 1$. This description of the remainder implies that a one's complement notation shall be used in representing the parity bits.

The 58 information and parity bits are defined as follows:

$u(k) = d(k)$ for k = 0, . . . ,52

$u(k + 53) = p(k)$ for k = 0, 1, 2, 3, 4

At this point, the 58 bits are divided into two subblocks which are separately encoded. The first subblock, u1(k) is composed of the first 10 bits of u(k) and four tail bits, as follows

$u1(k) = u(k)$ for k = 0, . . . , 9

$u1(k) = 0$ for k = 10, 11, 12, 13

The second subblock, u2(k) is composed of the last 48 bits of u(k) and four tail bits, as follows:

$u2(k) = u(k + 10)$ for k = 0, . . . , 47

$u2(k) = 0$ for k = 48, 49, 50, 51

6.9.1.3 Convolutional encoder

The 14 bits of u1(k) are encoded with a 1/2 rate, 16-state convolutional code defined by the polynomials:

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

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

This results in a block of 28 bits as follows:

$c1(2k) = u1(k) + u1(k - 3) + u1(k - 4)$

$c1(2k + 1) = u1(k) + u1(k - 1) + u1(k - 3) + u1(k - 4)$ for k = 0, ...,13

where u1(k) = 0 for k < 0, so that the initial state of the encoder is zero for each sequence.

The 52 bits of u2(k) are encoded with a 1/2 rate, 16-state convolutional code defined by the polynomials:

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

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

This results in a block of 105 bits as follows:

$c2(2k) = u2(k) + u2(k - 3) + u2(k - 4)$

$c2(2k + 1) = u2(k) + u2(k - 1) + u2(k - 3) + u2(k - 4)$ for k = 0, ..., 151

$c2(104) = 0$

where u2(k) = 0 for k < 0, so that the initial state of the encoder is zero for each sequence.

Note that the last bit of c2(k) is a spare bit.

6.9.1.4 Interleaving

The 28 bits of c1(k) are interleaved with a block interleaver with 7 rows and 4 columns. The procedure is to read the 28 bits of c1(k) into the interleaver row by row and to read out the 28 bits of i1(k) from the interleaver column by column.

The 105 bits of c2(k) are interleaved with a block interleaver with 7 rows and 15 columns. The procedure is to read the 105 bits of c2(k) into the interleaver row by row and to read out the 105 bits of i2(k) from the interleaver column by column, where columns 0 through 14 are read out in the following shuffled order 0, 5, 10, 3, 8, 13, 1, 6, 11, 4, 9, 14, 2, 7, and 12, in order to improve the interleaving distance.

6.9.1.5 Walsh Code

The 28 bits of i1(k) are divided into 4 seven-bit words, as follows:

$ii1(B,j) = i1(7B + j)$ for B = 0, ..., 3 and j = 0, ..., 6

where ii1(B,0) represents the MSB and ii1(B,6) represents the LSB. Each seven-bit word is encoded to 128 bits with the Walsh code given in table 6.9.1 to produce

iii1(B,j) for B = 0, ..., 3 and j = 0, ..., 127

where iii1(B,0) represents w0 and iii1(B,127) represents w127. Each resulting 128 bit sequence shall be Exclusive-Or'ed with the 128-bit m-sequence, bj, shown in table 6.9.1, to produce

$iiii1(B,j) = iii1(B,j) \oplus bj$ for B = 0, ..., 3 and j = 0, ..., 127

Table 6.9.1: M-Sequence for Exclusive-OR'in Operation

M-Sequence $\{b_j\}$	(b_0) 1 1 1 0 0 0 1 0 1 0 0 0 0 1 1 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 1 0 1 0 1 0 0 1 1 0 0 1 1 1 0 1 1 1 0 1 0 0 1 0 1 1 0 0 0 1 1 0 1 1 1 1 0 1 1 0 1 0 1 1 0 1 1 0 0 1 0 0 1 0 0 0 1 1 1 0 0 0 0 1 0 1 1 1 1 1 0 0 1 0 1 0 1 1 1 0 0 1 1 0 1 0 0 0 1 0 0 1 0 (b_{127})

The 105 bits of i2(k) are divided into 15 seven-bit words, as follows:

ii2(B,j) = i2(7B + j) for B = 0, ..., 14 and j = 0, ..., 6

where ii2(B,0) represents the MSB and ii2(B,6) represents the LSB. Each seven-bit word is encoded to 128 bits with the Walsh code given in table 4.10-1/05.03 to produce

iii2(B,j) for B = 0, ..., 14 and j = 0, ..., 127

where iii2(B,0) represents w0 and iii2(B,127) represents w127. Each resulting 128 bit sequence shall be Exclusive-Or'ed with the 128-bit m-sequence, bj, shown in table 4.9-1/05.03, to produce

iiii2(B,j) = iii2(B,j) $\oplus$ bj for B = 0, ..., 14 and j = 0, ..., 127

The result is iiii(B,j) which represents 19 groups of 128 bits defined by

iiii(B,j) = iiii1(B,j), B = 0, 1, 2, 3, j = 0, ..., 127

iiii(B,j) = iiii2(B - 4,j), B = 4, ..., 18, j = 0, ..., 127

6.9.1.6 Mapping on a burst

The resulting 19 groups of 128-bit sequences of iiii(B,k) are consecutively mapped onto the data fields of the 19 Satellite High Margin Bursts (HB) of the S-HPACH (IMSI) sequence per GMR-2 05.002 [5], clause 7.2.9 and table 9.0.5a. In other words, from GMR-2 05.002 [5], clause 7.2.9, iiii(B,0) thru iiii(B,63) are mapped onto en0 thru en63 and iiii(B,64) thru iiii(B,127) are mapped onto en64 to en127; and, from GMR-2 05.002 [5], table 9.0.5a iiii(0,j) thru iiii(18,j) are respectively mapped onto the 19 bursts denoted by Bi1 and Bi2, where i is number from 0 thru 4.

6.9.2 TMSI version (Optional)

The coding/interleaving for the S-HPACH TMSI version, as indicated in the S-BCCH message is specified below.

6.9.2.1 Block constitution

The message delivered to the encoder is composed of 30 information bits d(0), ..., d(29).

6.9.2.2 Parity bits and tail bits

Five parity bits are defined in such a way that in GF(2) the binary polynomial $d(0)D^{34} + d(1)D^{33} + \ldots + d(6)D^5 + p(0)D^4 + \ldots + p(4)$, when divided by $D^5 + D^2 + 1$ yields a remainder equal to $D^4 + D^3 + D^2 + D + 1$. This description of the remainder implies that a one's complement notation shall be used in representing the parity bits.

The 34 information and parity bits are defined as follows:

u(k) = d(k) for k = 0, . . . , 29

u(k + 30) = p(k) for k = 0, 1, 2, 3

Note that the highest order parity bit has been discarded.

At this point, the 34 bits are divided into two subblocks which are separately encoded. The first subblock, u1(k) is composed of the first 10 bits of u(k) and four tail bits, as follows:

u1(k) = u(k) for k = 0, . . . , 9

u1(k) = 0 for k = 10, 11, 12, 13

The second subblock, u2(k) is composed of the last 24 bits of u(k) and four tail bits, as follows:

$u2(k) = u(k + 10)$ for $k = 0, \ldots, 23$

$u2(k) = 0$ for $k = 24, 25, 26, 27$

6.9.2.3 Convolutional encoder

The 14 bits of u1(k) are encoded with a 1/2 rate, 16-state convolutional code defined by the polynomials:

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

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

This results in a block of 28 bits as follows:

$c1(2k) = u1(k) + u1(k - 3) + u1(k - 4)$

$c1(2k + 1) = u1(k) + u1(k - 1) + u1(k - 3) + u1(k - 4)$ for $k = 0, ..., 13$

where $u1(k) = 0$ for $k < 0$, so that the initial state of the encoder is zero for each sequence.

The 28 bits of u2(k) are encoded with a 1/2 rate, 16-state convolutional code defined by the polynomials:

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

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

This results in a block of 56 bits as follows:

$c2(2k) = u2(k) + u2(k - 3) + u2(k - 4)$

$c2(2k + 1) = u2(k) + u2(k - 1) + u2(k - 3) + u2(k - 4)$ for $k = 0, ..., 55$

where $u2(k) = 0$ for $k < 0$, so that the initial state of the encoder is zero for each sequence.

6.9.2.4 Interleaving

The 28 bits of c1(k) are interleaved with a block interleaver with 7 rows and 4 columns. The procedure is to read the 28 bits of c1(k) into the interleaver row by row and to read out the 28 bits of i1(k) from the interleaver column by column.

The 56 bits of c2(k) are interleaved with a block interleaver with 7 rows and 8 columns. The procedure is to read the 56 bits of c2(k) into the interleaver row by row and to read out the 56 bits of i2(k) from the interleaver column by column, where columns 0 through 7 are read out in the following shuffled order 0, 3, 6, 1, 4, 7, 2, and 5, in order to improve the interleaving distance.

6.9.2.5 Walsh Code

The 28 bits of i1(k) are divided into 4 seven-bit words, as follows:

$ii1(B,j) = i1(7B + j)$ for $B = 0, ..., 3$ and $j = 0, ..., 6$

where ii1(B,0) represents the MSB and ii1(B,6) represents the LSB. Each seven-bit word is encoded to 128 bits with the Walsh code given in table 6.9.1 to produce

$iii1(B,j)$ for $B = 0, ..., 3$ and $j = 0, ..., 127$

where iii1(B,0) represents w0 and iii1(B,127) represents w127. Each resulting 128 bit sequence shall be Exclusive-Or'ed with the 128-bit m-sequence, bj, shown in table 6.9.1, to produce

$iiii1(B,j) = iii1(B,j) \oplus bj$ for $B = 0, ..., 3$ and $j = 0, ..., 127$

The 56 bits of i2(k) are divided into 8 seven-bit words, as follows:

ii2(B,j) = i2(7B + j) for B = 0, ..., 7 and j = 0, ..., 6

where ii2(B,0) represents the MSB and ii2(B,6) represents the LSB. Each seven-bit word is encoded to 128 bits with the Walsh code given in table 4.9.1 to produce

iii2(B,j) for B = 0, ..., 7 and j = 0, ..., 127

where iii2(B,0) represents w0 and iii2(B,127) represents w127. Each resulting 128 bit sequence shall be Exclusive-Or'ed with the 128-bit m-sequence, bj, shown in table 6.9.1, to produce

iiii2(B,j) = iii2(B,j) ⊕ bj for B = 0, ..., 7 and j = 0, ..., 127

The result is iiii(B,j) which represents 12 groups of 128 bits defined by

iiii(B,j) = iiii1(B,j), B = 0, 1, 2, 3; j = 0, ..., 127

iiii(B,j) = iiii2(B - 4,j), B = 4, ..., 18; j = 0, ..., 7

6.9.2.6 Mapping on a burst

The resulting 12 groups of 128-bit sequences of iiii(B,k) are consecutively mapped onto the data fields of the 12 Satellite High Margin Bursts (HB) of the S-HPACH (TMSI) sequence per GMR-2 05.002 [5], clause 7.2.9 and table 9.0.5a. In other words, from GMR-2 05.002 [5], clause 7.2.9, iiii(B,0) thru iiii(B,63) are mapped onto en0 thru en63, and iiii(B,64) thru iiii(B,127) are mapped onto en64 to en127; and, from GMR-2 05.002 [5], table 9.0.5a, iiii(0,j) thru iiii(11,j) are respectively mapped onto the 12 bursts denoted by Bi1 and Bi2, where i is number from 0 thru 7.

6.10 High Margin Broadcast Control Channel

Satellite High Margin Broadcast Control Channel (S-HBCCH)

The burst carrying the High Margin Broadcast Control Channel has a different structure detailed in GMR-2 05.002 [5], clause 7.2. Figure 6.10.1 illustrates the basic flow for S-HBCCH coding and burst mapping.

6.10.1 Block constitution

The message delivered to the encoder has a fixed size of 194 information bits {d(0),...,d(194)}, where the first 10 bits represent the BCCH version identifier, as specified in GMR-2 04.008 [2], clause 10.1, and the last 184 bits represent BCCH information. Define the following two sub-blocks:

d1(k) = d(k) for k = 0, ..., 9

d2(k) = d(k + 10) for k = 0, . . ., 183

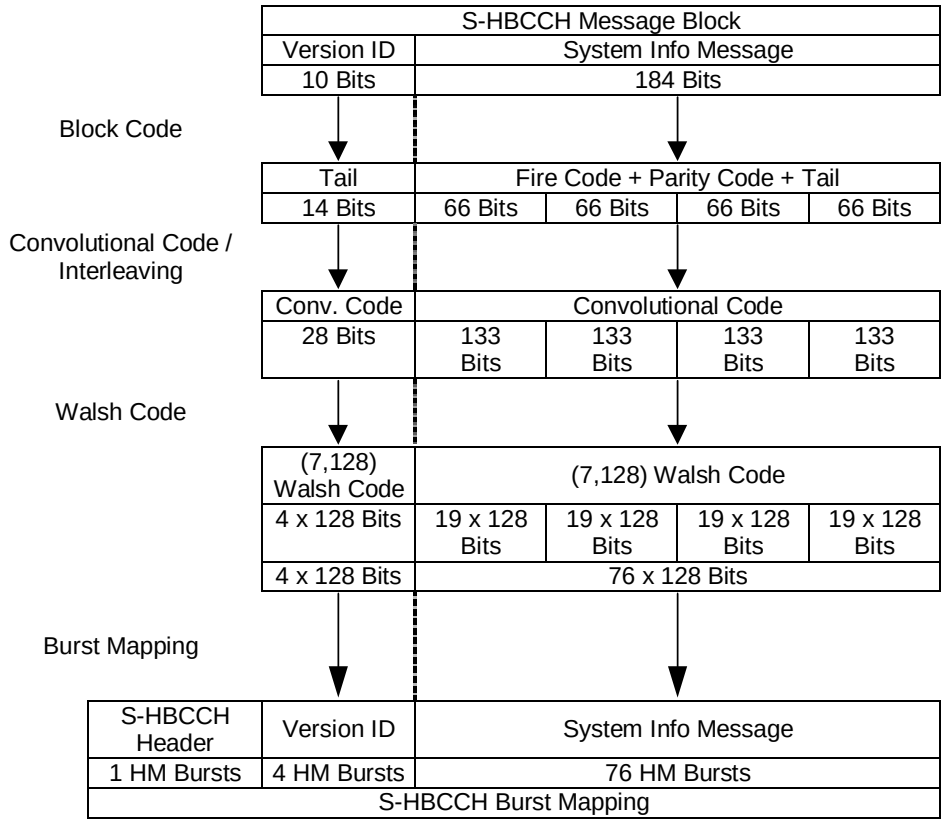


Figure 6.10.1: S-HBCCH Flow Diagram for Coding and Burst Mapping

6.10.2 Block code

Block coding is not applied to the version identifier subblock, d1(k). Four tail bits, however, are appended to the subblock, as follows:

$u1(k) = d1(k)$ for k = 0, ..., 9

$u1(k) = 0$ for k = 10, 11, 12, 13

A FIRE code, however, is applied to the 184 information bits, d2(k). The bits of d2(k) are encoded, using a shortened binary cyclic code (FIRE code), with the following generator polynomial:

$$g(D) = (D^{23} + 1) \times (D^{17} + D^3 + 1)$$

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

$$u2(0)D^{223} + u2(1)D^{222} + ... + u2(222)D + u2(223)$$

where {u2(0), u2(1),..., u2(183)} are the information bits (u2(k)=d2(k), k=0, ..., 183), and {u2(184), u2(185),..., u2(223)} are the parity bits (u2(k + 184) = p(k), k = 0, ..., 39), when divided by g(D) yields a remainder equal to $1 + D + D^2 + ... + D^{39}$. This description of the remainder implies that a one's complement notation shall be used in representing the Fire code parity bits. The result is a block of 224 bits.

At this point, the 224 bits of u2(k) are further divided into four 66-bit subblocks which are separately encoded. The first subblock, u2a(k) is composed of the first 56 bits of u(k), six parity bits, and four tail bits, as follows

u2a(k) = u2(k) for k = 0, . . . , 55

u2a(k + 56) = p2a(k) for k = 0, 1, 2, 3, 4, 5

u2a(k) = 0 for k = 62, 63, 64, 65

where the six parity bits are defined in such a way that in GF(2) the binary polynomial $u2a(0)D^{61} + u2a(1)D^{60} + . . . + u2a(7)D^6 + p2a(0) D^5 + . . . + p2a(5)$, when divided by $D^6 + D^5 + D^3 + D^2 + D + 1$ yields a remainder equal to $D^5 + D^4 + D^3 + D^2 + D + 1$. This description of the remainder implies that a one's complement notation shall be used in representing parity bits.

Similarly, the second through fourth subblocks are given as follows:

u2b(k) = u2(k + 56) for k = 0, . . . , 55

u2b(k + 56) = p2b(k) for k = 0, 1, 2, 3, 4, 5

u2b(k) = 0 for k = 62, 63, 64, 65

u2c(k) = u2(k + 112) for k = 0, . . . , 55

u2c(k + 56) = p2c(k) for k = 0, 1, 2, 3, 4, 5

u2c(k) = 0 for k = 62, 63, 64, 65

u2d(k) = u2(k + 168) for k = 0, . . . , 55

u2d(k + 56) = p2d(k) for k = 0, 1, 2, 3, 4, 5

u2d(k) = 0 for k = 62, 63, 64, 65

6.10.3 Convolutional encoder

The 14 bits of u1(k) are encoded with a 1/2 rate, 16-state convolutional code defined by the polynomials:

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

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

This results in a block of 28 bits as follows:

c1(2k) = u1(k) + u1(k - 3) + u1(k - 4)

c1(2k + 1) = u1(k) + u1(k - 1) + u1(k - 3) + u1(k - 4) for k = 0, ..., 13

where u1(k) = 0 for k < 0, so that the initial state of the encoder is zero for each sequence.

The 66 bits of u2a(k), u2b(k), u2c(k), and u2d(k) are each encoded with a 1/2 rate convolutional code defined by the polynomials:

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

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

This results in four subblocks of 133 bits as follows:

c2i(2k) = u2i(k) + u2i(k - 3) + u2i(k - 4)

c2i(2k + 1) = u2i(k) + u2i(k - 1) + u2i(k - 3) + u2i(k - 4) for k = 0, ..., 65

c2i(133) = 0

where u2i(k) = 0 for k < 0 so that the initial state of the encoder is zero for each sequence, and where i represents each of the four subblocks a, b, c, and d. Note that the last bit of c2i(k) is a spare bit.

6.10.4 Interleaving

The 28 bits of c1(k) are interleaved with a block interleaver with 7 rows and 4 columns. The procedure is to read the 28 bits of c1(k) into the interleaver row by row and to read out the 28 bits of i1(k) from the interleaver column by column.

The 133 bits of c2a(k), c2b(k), c2c(k), and c2d(k) are each interleaved with a block interleaver with 7 rows and 19 columns. The procedure is to read the 133 bits of each c2i(k) into the interleaver row by row and to read out the 133 bits of i2i(k) from the interleaver column by column, where columns 0 through 18 are read out in the following shuffled order 0, 5, 10, 15, 3, 8, 13, 18, 1, 6, 11, 16, 4, 9, 14, 2, 7, 12, and 17, in order to improve the interleaving distance. In other words, four interleaved subblocks result i2a(k), i2b(k), i2c(k), and i2d(k). These four subblock are reassembled into a block of 532 bits, as follows:

$i2(k) = i2a(k)$ for $k = 0, \ldots, 132$

$i2(k + 133) = i2b(k)$ for $k = 0, \ldots, 132$

$i2(k + 266) = i2c(k)$ for $k = 0, \ldots, 132$

$i2(k + 399) = i2d(k)$ for $k = 0, \ldots, 132$

6.10.5 Walsh code

The 28 bits of i1(k) are divided into 4 seven-bit words as follows:

$ii1(B,j) = i1(7B + j)$ for $B = 0, ..., 3$ and $j = 0, ..., 6$

where ii1(B,0) represents the MSB and ii1(B,6) represents the LSB. Each seven-bit word is encoded to 128 bits with the Walsh code given in table 6.10.2 to produce

$iii1(B,j)$ for $B = 0, ..., 3$ and $j = 0 , ..., 127$

where iii1(B,0) represents w0 and iii1(B,127) represents w127. Each resulting 128 bit sequence shall be Exclusive-Or'ed with the 128-bit m-sequence, bj, shown in table 6.9.1, to produce

$iiii1(B,j) = iii1(B,j) \oplus bj$ for $B = 0, ..., 3$ and $j = 0, ..., 127$

The 532 bits of i2(k) are divided into 76 seven-bit words, as follows:

$ii2(B,j) = i2(7B+j)$ for $B = 0, ..., 75$ and $j = 0, ..., 6$

where ii2(B,0) represents the MSB and ii2(B,6) represents the LSB. Each seven-bit word is encoded to 128 bits with the Walsh code given in table 6.10.1, to produce

$iii2(B,j)$ for $B = 0, ..., 75$ and $j = 0 , ..., 127$

where iii2(B,0) represents w0 and iii2(B,127) represents w127. Each resulting 128 bit sequence shall be Exclusive-Or'ed with the 128-bit m-sequence, bj, shown in table 6.9.1, to produce

$iiii2(B,j) = iii2(B,j) \oplus bj$ for $B = 0, ..., 75$ and $j = 0, ..., 127$

The result is iiii(B,j) which represents 80 groups of 128 bits defined by

$iiii(B,j) = iiii1(B,j),$ $B = 0, 1, 2, 3,$ $j = 0, ..., 127$

$iiii(B,j) = iiii2(B - 4,j),$ $B = 4, ..., 79,$ $j = 0, ..., 7$

Table 6.10.2: Walsh Code

Info bits (MSB . . . LSB)	Coded bits w0 .....w63 w64 .....w127
0000000	00000000000000000000000000000000000000000000000000000000000000000000000000000000 00000000000000000000000000000000000000000000000000000000000000000000000000000000
0000001	01010101010101010101010101010101010101010101010101010101010101010101010101010101 01010101010101010101010101010101010101010101010101010101010101010101010101010101
0000010	00110011001100110011001100110011001100110011001100110011001100110011001100110011 00110011001100110011001100110011001100110011001100110011001100110011001100110011
0000011	01100110011001100110011001100110011001100110011001100110011001100110011001100110 01100110011001100110011001100110011001100110011001100110011001100110011001100110
0000100	00001111000011110000111100001111000011110000111100001111000011110000111100001111 00001111000011110000111100001111000011110000111100001111000011110000111100001111
0000101	01011010010110100101101001011010010110100101101001011010010110100101101001011010 01011010010110100101101001011010010110100101101001011010010110100101101001011010
0000110	00111100001111000011110000111100001111000011110000111100001111000011110000111100 00111100001111000011110000111100001111000011110000111100001111000011110000111100
0000111	01101001011010010110100101101001011010010110100101101001011010010110100101101001 01101001011010010110100101101001011010010110100101101001011010010110100101101001
0001000	00000000111111110000000011111111000000001111111100000000111111110000000011111111 00000000111111110000000011111111000000001111111100000000111111110000000011111111
0001001	01010101101010100101010110101010010101011010101001010101101010100101010110101010 01010101101010100101010110101010010101011010101001010101101010100101010110101010
0001010	00110011110011000011001111001100001100111100110000110011110011000011001111001100 00110011110011000011001111001100001100111100110000110011110011000011001111001100
0001011	01100110100110010110011010011001011001101001100101100110100110010110011010011001 01100110100110010110011010011001011001101001100101100110100110010110011010011001
0001100	00001111111110000000011111111000000001111111100000000111111110000000011111111000 00001111111110000000011111111000000001111111100000000111111110000000011111111000
0001101	01011010101001010101101010100101010110101010010101011010101001010101101010100101 01011010101001010101101010100101010110101010010101011010101001010101101010100101
0001110	00111100110000110011110011000011001111001100001100111100110000110011110011000011 00111100110000110011110011000011001111001100001100111100110000110011110011000011
0001111	01101001100101100110100110010110011010011001011001101001100101100110100110010110 01101001100101100110100110010110011010011001011001101001100101100110100110010110
0010000	00000000000000001111111111111111000000000000000011111111111111110000000000000000 00000000000000001111111111111111000000000000000011111111111111110000000000000000
0010001	01010101010101011010101010101010010101010101010110101010101010100101010101010101 01010101010101011010101010101010010101010101010110101010101010100101010101010101
0010010	00110011001100111100110011001100001100110011001111001100110011000011001100110011 00110011001100111100110011001100001100110011001111001100110011000011001100110011
0010011	01100110011001101001100110011001011001100110011010011001100110010110011001100110 01100110011001101001100110011001011001100110011010011001100110010110011001100110
0010100	00001111000011111111000011110000000011110000111111110000111100000000111100001111 00001111000011111111000011110000000011110000111111110000111100000000111100001111
0010101	01011010010110101010010110100101010110100101101010100101101001010101100101101001 01011010010110101010010110100101010110100101101010100101101001010101100101101001
0010110	00111100001111001100001111000011001111000011110011000011110000110011110000111100 00111100001111001100001111000011001111000011110011000011110000110011110000111100
0010111	01101001011010011001011010010110011010010110100110010110100101100101101001011010 01101001011010011001011010010110011010010110100110010110100101100101101001011010
0011000	00000000111111111111111100000000000000000011111111111111110000000000000000111111 00000000111111111111111100000000000000000011111111111111110000000000000000111111
0011001	01010101101010101010101001010101010101010110101010101010100101010101010101101010 01010101101010101010101001010101010101010110101010101010100101010101010101101010
0011010	00110011110011001100110000110011001100111100110011001100000110011110011000011001 00110011110011001100110000110011001100111100110011001100000110011110011000011001
0011011	01100110100110011001100101100110011001101001100110011001011001100110011001011001 01100110100110011001100101100110011001101001100110011001011001100110011001011001
0011100	00001111111110000111100000000111100001111111100001111000000001111000011111111000 00001111111110000111100000000111100001111111100001111000000001111000011111111000
0011101	01011010101001011010010101011010010110101010010110100101010110100101101010010101 01011010101001011010010101011010010110101010010110100101010110100101101010010101

Info bits (MSB . . . LSB)	Coded bits w0 .....w63 w64 .....w127
0011110	00111100110000111100001100111100001111001100001111000011001111000011110000110011110000111100001100111100
0011111	01101001100101101001011001101001011010011001011010010110011010010110100101100110100101101001011001101001
0100000	00000000000000000000000000000000111111111111111111111111111111110000000000000000000000000000000011111111
0100001	01010101010101010101010101010101101010101010101010101010101010100101010101010101010101010101010110101010
0100010	00110011001100110011001100110011100110011001100110011001100110011001100110011001100110011001100110011001100
0100011	01100110011001100110011001100110100110011001100110011001100110011001100110011001100110011001100110011001100
0100100	00001111000011110000111100001111111100001111000011110000111100000000111100001111000011110000111100000000
0100101	01011010010110100101101001011010101001011010010110100101101001010110100101011010010110100101101001010110100101
0100110	00111100001111000011110000111100110000111100001111000011110000111100000011000011110000111100001111000000
0100111	01101001011010010110100101101001100101101001011010010110100101101001011010010110100101101001011010010110
0101000	00000000111111110000000011111111111111110000000011111111000000000000000011111111000000000000000000000000
0101001	01010101101010100101010110101010101010100101010110101010010101010101010110101010010101010101010110101010
0101010	00110011110011000011001111001100110011000011001111001100001100111100110000110011110011000011001111001100
0101011	01100110100110010110011010011001100110010110011010011001011001101001101001100101100110100110010110011010
0101100	00001111111110000000011111111000011110000000001111111100000000111100001111111100000000111100001111111100
0101101	01011010101001010101101010100101101001010101101010010101011010100101010110101001010101101010010101011010
0101110	00111100110000110011110011000011110000110011110011000011001111001100001100111100110000110011110011000011
0101111	01101001100101100110100110010110100101100110100110010110011010011001011001101001100101100110100110010110
0110000	00000000000000000111111111111111111111111111111111111111000000000000000000000000000000000000000000000000
0110001	01010101010101011010101010101010101010101010101010101010010101010101010101010101010101010101010101010101
0110010	00110011001100111100110011001100110011001100110011000011001100110011001100110011001100110011001100110011
0110011	01100110011001101001100110011001100110011001100110010110011001100101100110011001100101100110011001100110
0110100	00001111000011111111000011110000111100001111000000000111100001111000011110000111100001111000011110000111
0110101	01011010010110101010010110100101101001011010010101011010010110101011010010110100101101001011010010110101
0110110	00111100001111001100001111000011110000111100001100111100001111000011110000111100001111000011110000111100
0110111	01101001011010011001011010010110100101101001011001101001011010010110100101101001011010010110100101101001
0111000	00000000111111111111111100000000111111110000000000000000111111110000000011111111000000001111111100000000
0111001	01010101101010101010101001010101101010100101010101010101101010100101010110101010010101011010101001010101
0111010	00110011110011001100110000110011110011000011001100110011110011001100001100111100110000110011110011000011
0111011	01100110100110011001100101100110100110010110011001100101100110100110010110011001100101100110011001100110
0111100	00001111111110000111100000000111111110000000011110000111111110000000011111111000000001111111100000000111

Info bits (MSB . . . LSB)	Coded bits w0 .....w63 w64 .....w127
0111101	0101101010100101101001010101101010100101010110100101101010100101 0101101010100101101001010101101010100101010110100101101010100101
0111110	0011110011000011110000110011110011000011001111000011110011000011 0011110011000011110000110011110011000011001111000011110011000011
0111111	0110100110010110100101100110100110010110011010010110100110010110 0110100110010110100101100110100110010110011010010110100110010110
1000000	0000000000000000000000000000000000000000000000000000000000000000 1111111111111111111111111111111111111111111111111111111111111111
1000001	0101010101010101010101010101010101010101010101010101010101010101 1010101010101010101010101010101010101010101010101010101010101010
1000010	0011001100110011001100110011001100110011001100110011001100110011 1100110011001100110011001100110011001100110011001100110011001100
1000011	0110011001100110011001100110011001100110011001100110011001100110 1001100110011001100110011001100110011001100110011001100110011001
1000100	0000111100001111000011110000111100001111000011110000111100001111 1111000011110000111100001111000011110000111100001111000011110000
1000101	0101101001011010010110100101101001011010010110100101101001011010 1010010110100101101001011010010110100101101001011010010110100101
1000110	0011110000111100001111000011110000111100001111000011110000111100 1100001111000011110000111100001111000011110000111100001111000011
1000111	0110100101101001011010010110100101101001011010010110100101101001 1001011010010110100101101001011010010110100101101001011010010110
1001000	0000000011111111000000001111111100000000111111110000000011111111 1111111100000000111111110000000011111111000000001111111100000000
1001001	0101010110101010010101011010101001010101101010100101010110101010 1010101001010101101010100101010110101010010101011010101001010101
1001010	0011001111001100001100111100110000110011110011000011001111001100 1100110000110011110011000011001111001100001100111100110000110011
1001011	0110011010011001011001101001100101100110100110010110011010011001 1001100101100110100110010110011010011001011001101001100101100110
1001100	0000111111110000000011111111000000001111111100000000111111110000 1111000000001111111100000000111111110000000011111111000000001111
1001101	0101101010100101010110101010010101011010101001010101101010100101 1010010101011010101001010101101010100101010110101010010101011010
1001110	0011110011000011001111001100001100111100110000110011110011000011 1100001100111100110000110011110011000011001111001100001100111100
1001111	0110100110010110011010011001011001101001100101100110100110010110 1001011001101001100101100110100110010110011010011001011001101001
1010000	0000000000000000111111111111111100000000000000001111111111111111 1111111111111111000000000000000011111111111111110000000000000000
1010001	0101010101010101101010101010101001010101010101011010101010101010 1010101010101010010101010101010110101010101010100101010101010101
1010010	0011001100110011110011001100110000110011001100111100110011001100 1100110011001100001100110011001111001100110011000011001100110011
1010011	0110011001100110100110011001100101100110011001100101101001100110011001 1001100110011001011001100110011010011001100110011001011001100110011001
1010100	0000111100001111111100001111000000001111000011111111000011110000 1111000011110000000011110000111111110000111100000000111100001111
1010101	0101101001011010101001011010010101011010010110101010010110100101 1010010110100101010110100101101010100101101001010101101001011010
1010110	0011110000111100110000111100001100111100001111001100001111000011 1100001111000011001111000011110011000011110000110011110000111100
1010111	0110100101101001100101101001011001101001011010011001011010010110 1001011010010110011010010110100110010110100101100110100101101001
1011000	0000000011111111111111110000000000000000111111111111111100000000 1111111100000000000000001111111111111111000000000000000011111111
1011001	0101010110101010101010100101010101010101101010101010101001010101 1010101001010101010101011010101010101010010101010101010110101010
1011010	0011001111001100110011000011001100110011110011001100110000110011 1100110000110011001100111100110011001100001100110011001111001100
1011011	0110011010011001100110010110011001100110100110011001100101100110 1001100101100110011001101001100110011001011001100110011010011001

Info bits (MSB . . . LSB)	Coded bits w0 .....w63 w64 .....w127
1011100	00001111111100001111000000001111000011111111000011110000000011111111000011110000111111110000
1011101	01011010101001011010010101011010010110101010010110100101010110101010010110100101011010010100101011010010101011010010110101010010110100101101010100101
1011110	00111100110000111100001100111100001111001100001111000011001111000011110011000011110000110011110000111100001100111100001111001100001111000011001111
1011111	01101001100101101001011001101001011010011001011010010110011010010110011001011010010110011001011010010110011001011010010110011001011010010110011001
1100000	00000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
1100001	01010101010101010101010101010101101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010
1100010	00110011001100110011001100110011110011001100110011001100110011001100110011001100110011001100110011001100110011001100110011001100110011001100110011
1100011	01100110011001100110011001100110100110011001100110011001100110011001100110011001100110011001100110011001100110011001100110011001100110011001100110
1100100	00001111000011110000111100001111111100001111000011110000111100001111000011110000111100000111100001111000011110000111100000111100001111000011110000
1100101	01011010010110100101101001011010101001011010010110100101101001011010010110100101101001011010010110100101101001011010010110100101101001011010010110
1100110	00111100001111000011110000111100110000111100001111000011110000111100001111000011
1100111	01101001011010010110100101101001100101101001011010010110100101101001011010010110
1101000	00000000111111110000000011111111111111110000000011111111000000001111111100000000
1101001	01010101101010100101010110101010101010100101010110101010010101011010101001010101
1101010	00110011110011000011001111001100110011000011001111001100001100111100111000011001
1101011	01100110100110010110011010011001100110010110011010011001011001101100110100101100
1101100	00001111111110000000011111111000011110000000001111111100000000011111111100000000
1101101	01011010101001010101101010100101101001010101101010100101010110101010010101011010
1101110	00111100110000110011110011000011110000110011110011000011001111001100001100111100
1101111	01101001100101100110100110010110100101100110100110010110011010010110011001011010
1110000	00000000000000000111111111111111111111111111111111000000000000000011111111111111
1110001	01010101010101011010101010101010101010101010101010010101010101010101010101010101
1110010	00110011001100111100110011001100110011001100110000110011001100110011001100111100
1110011	01100110011001101001100110011001100110011001100101100110011001100110100110011001
1110100	00001111000011111111000011110000111100001111000000001111000011110000111100000000
1110101	01011010010110101010010110100101101001011010010101011010010110100101101001011010
1110110	00111100001111001100001111000011110000111100001100111100001111000011110000111100
1110111	01101001011010011001011010010110100101101001011001101001011001011010010110100101
1111000	00000000111111111111111100000000111111110000000000000000111111110000000011111111
1111001	01010101101010101010101001010101101010100101010101010101100101010101010101100101010
1111010	00110011110011001100110000110011110011000011001100110011110011001100110000110011

Info bits (MSB . . . LSB)	Coded bits w0 .....w63 w64 .....w127
1111011	01100110100110011001100101100110100110010110011001100110011010011001 10011001011001100110011010011001011001101001100110011001100101100110
1111100	00001111111100001111000000001111111100000000111100001111111100000 11110000000011110000111111110000000011111111000011110000000001111
1111101	01011010101001011010010101011010101001010101101001011010100101 10100101010110100101101010100101010110101010010110100101010110
1111110	00111100110000111100001100111100110000110011110000111100110000011 11000011001111000011110011000011001111001100001111000011001111100
1111111	01101001100101101001011001101001100101100110100101101001100101101 10010110011010010110100110010110011010011001011010010110011010011

6.10.6 Mapping on a burst

The S-HBCCH channel is composed of a series of 81 bursts.

The first burst of the S-HBCCH sequence is a high margin M-sequence burst as defined in GMR-2 05.002 [5], clause 7.2.

The remaining 80 bursts of the S-HBCCH sequence are comprised of the 80 groups of 128-bit sequences denoted by iiii(B,j). These 80 groups are consecutively mapped onto the data fields of the 80 Satellite High Margin Bursts (HB) of the S-HBCCH sequence per GMR-2 05.002 [5], clause 7.2.9, table 9.0.5a. In other words, from GMR-2 05.002 [5], clause 7.2.9, iiii(B,0) thru iiii(B,63) are mapped onto en0 thru en63, and iiii(B,64) thru iiii(B,127) are mapped onto en64 to en127; and, from GMR-2 05.002 [5], table 9.0.5a, iiii(0,j) thru iiii(79,j) are respectively mapped onto the 80 bursts denoted by B1 and B2, where the burst denoted by B0 corresponds to a High Margin M-sequence Burst (HMB).

6.11 Beam Broadcast Channel (S-BBCH) (Optional)

6.11.1 Block constitution

The message delivered to the encoder has a fixed size of 184 information bits {d(0),...,d(183)}.

6.11.2 Block code

The block of 184 information bits is encoded, using a shortened binary cyclic code (FIRE code), with the following generator polynomial:

$$g(D) = (D^{23} + 1) \times (D^{17} + D^3 + 1)$$

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

$$u(0)D^{223} + u(1)D^{222} + ... + u(222)D + u(223)$$

where {u(0), u(1),..., u(183)} are the information bits (u(k) = d(k), k = 0, ..., 183), and {u(184), u(185), ..., u(223)} are the parity bits (u(k + 184) =p(k), k = 0, ..., 39), when divided by g(D) yields a remainder equal to $1 + D + D^2 + ... + D^{39}$. This description of the remainder implies that a one's complement notation shall be used in representing the Fire code parity bits.

The result is a block of 224 bits, completed by 4 tail bits equal to 0 at the end of the block: {u(0), u(1),...,u(223)}, with u(224) to u(227) equal to 0.

6.11.3 Convolutional encoder

The 228 bits of u(k) are encoded with a 1/2 rate, 16-state convolutional code defined by the polynomials:

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

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

This results in a block of 456 bits as follows:

$c(2k) = u(k) + u(k-3) + u(k-4)$

$c(2k + 1) = u(k) + u(k - 1) + u(k - 3) + u(k - 4)$ for $k = 0, ..., 227$

where $u(k) = 0$ for $k < 0$, so that the initial state of the encoder is zero for each sequence.

6.11.4 Interleaving

The 456 bits of c(k) are interleaved with a block interleaver with 24 rows and 19 columns. The procedure is to read the 456 bits of u(k) into the interleaver row by row and to read out the 456 bits of i(k) from the interleaver column by column.

6.11.5 Non-linear block code

The 456 bits of i(k) are divided into 152 three-bit words, as follows:

$ii(B,j) = i(3B + j)$ for $B = 0, ..., 151$

$j = 0, 1, 2,$

where ii(B,0) represents the MSB and ii(B,2) represents the LSB.

Each three-bit word is encoded to 14 bits with the (14,3) code given in table 6.11.1 to produce

$iii(B,j)$ for $B = 0, . . ., 151$ and $j = 0, . . . , 13$

where iii(B,0) represents w0 and iii(B,13) represents w13.

Table 6.11.1: (14,3) Code

Information Bits (MSB ... LSB)	Coded Bits (w_0 ..... w_{13})
0 0 0	1 0 1 0 1 0 1 0 1 0 1 0
0 0 1	0 1 1 0 0 1 1 0 0 1 1 0 0 1
0 1 0	1 0 0 1 0 1 1 0 1 0 0 1 0 1
0 1 1	0 1 0 1 1 0 1 0 0 1 0 1 1 0
1 0 0	1 0 1 0 1 0 0 1 0 1 0 1 0 1
1 0 1	0 1 1 0 0 1 0 1 1 0 0 1 1 0
1 1 0	1 0 0 1 0 1 0 1 0 1 1 0 1 0
1 1 1	0 1 0 1 1 0 0 1 1 0 1 0 0 1

6.11.6 Mapping on a burst

The resulting 152 fourteen-bit sequences, iii(B,j), are consecutively mapped onto the fourteen-bit midamble fields of the Satellite High Margin Burst (HB) described in GMR-2 05.002 [5], clause 7.2.9, as follows: they are mapped onto the midamble fields of the last 76 HB bursts of two consecutive 81-burst S-HBCCH sequences shown in GMR-2 05.002 [5], table 9.0.5a.

For instances of:

a) the transmission of the S-HPACH;

b) the first four bursts of the S-HBCCH message; or

c) message not being transmitted on the midambles of the S-HBCCH.

The 14-bits corresponding to the 0 0 0 information shall be transmitted on the midamble field of the HB bursts. In other words, in these instances, the pattern:

[1 0 1 0 1 0 1 0 1 0 1 0 1 0]

shall be transmitted in the 14-bit HB midamble

6.12 Robust Slow Associated Control Channel

Satellite Slow, S-TCH/HR Associated, Control Channel (S-SACCH/THR)

Satellite Slow S-SDCCH/HR Associated, Control Channel (S-SACCH/CHR)

6.12.1 Block constitution

The message delivered to the encoder has a fixed size of 184 information bits {d(0), ..., d(183)}.

6.12.2 Block code

The block of 184 information bits is encoded, using a shortened binary cyclic code (Fire code), with the following generator polynomial:

$g(D) = (D^{23} + 1) \times (D^{17} + D^3 + 1)$

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

$u(0)D^{223} + u(1)D^{222} + \ldots + u(222)D + u(223)$

where:

$u(k) = d(k)$ for k = 0,. . ., 183 are information bits

and:

$u(k + 184) = p(k)$ for k = 0,. . ., 39 are parity bits,

when divided by g(D) yields a remainder equal to $1 + D + D^2 + \ldots + D^{39}$.

The result is a block of 224 bits which are split into four 56-bit subblocks u1(k), u2(k), u3(k), and u4(k) with six tail bits added to each, as follows:

$u1(k) = u(k)$ for k = 0, . . ., 55

$u1(k) = 0$ for k = 56, . . ., 61

$u2(k) = u(k + 56)$ for k = 0, . . ., 55

$u2(k) = 0$ for k = 56, . . ., 61

$u3(k) = u(k + 112)$ for k = 0, . . ., 55

$u3(k) = 0$ for k = 56, . . ., 61

and:

$u4(k) = u(k + 168)$ for k = 0, . . ., 55

$u4(k) = 0$ for k = 56,. . . , 61

Note that each subblock contains 56 bits of the original 224 bit sequence and 6 tail bits.

6.12.3 Convolutional encoder

Each of the resulting subblocks of 62 bits is encoded with a rate 1/4, 64-state convolutional code defined by the following polynomials:

$G0 = 1 + D^2 + D^3 + D^4 + D^6$

$G1 = 1 + D^2 + D^3 + D^5 + D^6$

$G2 = 1 + D + D^4 + D^5 + D^6$

$G3 = 1 + D + D^2 + D^3 + D^6$

The result is four subblocks of 248 coded bits {c1'(0), . . . , c1'(247)}, {c2'(0), . . . , c2'(247)}, {c3'(0), . . . , c3'(247)}, and {c4'(0), . . . , c4'(247)}, where:

c1a(4k) = u1(k) + u1(k - 2) + u1(k - 3) + u1(k - 4) + u1(k - 6)

c1a(4k + 1) = u1(k) + u1(k - 2) + u1(k - 3) + u1(k - 5) + u1(k - 6)

c1a(4k + 2) = u1(k) + u1(k - 1) + u1(k - 4) + u1(k - 5) + u1(k - 6)

c1a(4k + 3) = u1(k) + u1(k - 1) + u1(k - 2) + u1(k - 3) + u1(k - 6) for k = 0,...,61

c2a(4k) = u2(k) + u2(k - 2) + u2(k - 3) + u2(k - 4) + u2(k - 6)

c2a(4k + 1) = u2(k) + u2(k - 2) + u2(k - 3) + u2(k - 5) + u2(k - 6)

c2a(4k + 2) = u2(k) + u2(k - 1) + u2(k - 4) + u2(k - 5) + u2(k - 6)

c2a(4k + 3) = u2(k) + u2(k - 1) + u2(k - 2) + u2(k - 3) + u2(k - 6) for k = 0, ..., 61

c3a(4k) = u3(k) + u3(k - 2) + u3(k - 3) + u3(k - 4) + u3(k - 6)

c3a(4k + 1) = u3(k) + u3(k - 2) + u3(k - 3) + u3(k - 5) + u3(k - 6)

c3a(4k + 2) = u3(k) + u3(k - 1) + u3(k - 4) + u3(k - 5) + u3(k - 6)

c3a(4k + 3) = u3(k) + u3(k - 1) + u3(k - 2) + u3(k - 3) + u3(k - 6) for k = 0, ..., 61

and:

c4a(4k) = u4(k) + u4(k - 2) + u4(k - 3) + u4(k - 4) + u4(k - 6)

c4a(4k + 1) = u4(k) + u4(k - 2) + u4(k - 3) + u4(k - 5) + u4(k - 6)

c4a(4k + 2) = u4(k) + u4(k - 1) + u4(k - 4) + u4(k - 5) + u4(k - 6)

c4a(4k + 3) = u4(k) + u4(k - 1) + u4(k - 2) + u4(k - 3) + u4(k - 6) for k = 0, ..., 61

In the definition of c1a(k), c2a(k), c3a(k), and c4a(k), note that u1(k), u2(k), u3(k), and u4(k) are assumed to equal 0 for $k < 0$, so that the initial states of the encoders are zero for each sequence.

For each of these four subblocks, the code is punctured in such a way that the following 8 bits are not transmitted:

cia(0), cia(1), cia(60), cia(61), cia(120), cia(121), cia(180), and cia(181) for I = 1, 2, 3, 4

The remaining four subblocks which contain 960 coded bits are denoted by

{c1(0), . . . , c1(239)}, {c2(0), . . . , c2(239)}, {c3(0), . . . , c3(239)}, and {c4(0), . . . , c4(239)}

6.12.4 Interleaving

The 960 coded bits are interleaved in accordance with the following procedures.

c1(k) is further split into two 120-bit blocks, c1e(k) and c1o(k), composed of the even and odd bits of c1(k):

c1e(k) = c1(2k)

c1o(k) = c1(2k + 1) for k = 0, . . ., 119

Similarly, c2(k), c3(k), and c4(k) are further split into two 120-bit blocks:

c2e(k) = c2(2k)

c2o(k) = c2(2k + 1) for k = 0, . . ., 119

c3e(k) = c3(2k)

c3o(k) = c3(2k + 1) for k = 0, . . ., 119

and:

c4e(k) = c4(2k)

c4o(k) = c4(2k + 1) for k = 0, . . ., 119

From the four subgroups of 120 bits represented by cie(k), i = 1, 2, 3, 4, new bursts are formed with 4-subgroup diagonal interleaving as follows:

SubGroup 1': c1e(0) c2e(1) c3e(2) c4e(3) c1e(4) c2e(5) . . . c1e(116) c2e(117) c3e(118) c4e(119)

SubGroup 2': c2e(0) c3e(1) c4e(2) c1e(3) c2e(4) c3e(5) . . . c2e(116) c3e(117) c4e(118) c1e(119)

SubGroup 3': c3e(0) c4e(1) c1e(2) c2e(3) c3e(4) c4e(5) . . . c3e(116) c4e(117) c1e(118) c2e(119)

SubGroup 4': c4e(0) c1e(1) c2e(2) c3e(3) c4e(4) c1e(5) . . . c4e(116) c1e(117) c2e(118) c3e(119)

Each of these diagonally interleaved subgroups is further interleaved with a block interleaver with 12 rows and 10 columns. The 120 bits of each 120-bit diagonally-interleaved subgroup are read into the interleaver row-by-row and are read out column-by-column to generate bursts ije(k) for k=0, ..., 119, where j=1,2,3,4 refers to the subgroup number. For example, the SubGroup 1' is written into a 12-row by 10-column matrix row-by-row, as follows:

c1e(0) c2e(1) c3e(2) c4e(3) c1e(4) c2e(5) c3e(6) c4e(7) c1e(8) c2e(9)

c3e(10) c4e(11) c1e(12) c2e(13) c3e(14) c4e(15) c1e(16) c2e(17) c3e(18) c4e(19)

c1e(20) c2e(21) c3e(22) c4e(23) c1e(24) c2e(25) c3e(26) c4e(27) c1e(28) c2e(29)

.

.

.

c3e(110) c4e(111) c1e(112) c2e(113) c3e(114) c4e(115) c1e(116) c2e(117) c3e(118) c4e(119)

The resulting 120 block-interleaved bits from SubGroup 1' are read out column-by-column, as follows:

c1e(0), c3e(10), c1e(20), c3e(30), ..., c1e(100), c3e(110), c2e(1), c4e(11), c2e(21), c4e(31), ..., c2e(109), c4e(119)

Similarly, from the four subgroups of 120 bits represented by cio(k), i = 1,2,3,4 new bursts are formed with 4-subgroup diagonal interleaving as follows:

SubGroup 1': c1o(0) c2o(1) c3o(2) c4o(3) c1o(4) c2o(5) . . . c1o(116) c2o(117) c3o(118) c4o(119)

SubGroup 2': c2o(0) c3o(1) c4o(2) c1o(3) c2o(4) c3o(5) . . . c2o(116) c3o(117) c4o(118) c1o(119)

SubGroup 3': c3o(0) c4o(1) c1o(2) c2o(3) c3o(4) c4o(5) . . . c3o(116) c4o(117) c1o(118) c2o(119)

SubGroup 4': c4o(0) c1o(1) c2o(2) c3o(3) c4o(4) c1o(5) . . . c4o(116) c1o(117) c2o(118) c3o(119)

Each of these diagonally interleaved subgroups is further interleaved with a block interleaver with 12 rows and 10 columns. The 120 bits of each 120-bit diagonally-interleaved subgroup are read into the interleaver row-by-row and are read out column-by-column to generate bursts ijo(k) for k=0, ..., 119, where j = 1, 2, 3, 4 refers to the subgroup number. For example, the SubGroup 1' is written into a 12-row by 10-column matrix row-by-row, as follows:

c1o(0) c2o(1) c3o(2) c4o(3) c1o(4) c2o(5) c3o(6) c4o(7) c1o(8) c2o(9)

c3o(10) c4o(11) c1o(12) c2o(13) c3o(14) c4o(15) c1o(16) c2o(17) c3o(18) c4o(19)

c1o(20) c2o(21) c3o(22) c4o(23) c1o(24) c2o(25) c3o(26) c4o(27) c1o(28) c2o(29)

.

.

.

c3o(110) c4o(111) c1o(112) c2o(113) c3o(114) c4o(115) c1o(116) c2o(117) c3o(118) c4o(119)

The resulting 120 block-interleaved bits from SubGroup 1' are read out column-by-column, as follows:

c1o(0), c3o(10), c1o(20), c3o(30), ..., c1o(100), c3o(110), c2o(1), c4o(11), c2o(21), c4o(31), ..., c2o(109), c4o(119)

The resulting 960 interleaved bits recombined to produce i(j,k), as follows:

i(0,k) = i1e(k) for k = 0, . . ., 119

i(1,k) = i1o(k) for k = 0, . . ., 119

i(2,k) = i2e(k) for k = 0, . . ., 119

i(3,k) = i2o(k) for k = 0, . . ., 119

i(4,k) = i3e(k) for k = 0, . . ., 119

i(5,k) = i3o(k) for k = 0, . . ., 119

i i(6,k) = i4e(k) for k = 0, . . ., 119

i(7,k) = i4o(k) for k = 0, . . ., 119

6.12.5 Mapping on a burst

The resulting 960 interleaved bits for each message are consecutively mapped onto eight bursts of a half-rate channel, as described in GMR-2 05.002 [5], table 9.0.4. The mapping is given by the rule:

e(B, j) = i(B, j) for j = 0, ..., 119 with B = 0, 1, 2, 3, 4, 5, 6, 7

6.13 Robust Paging and Access Grant Broadcast (S-PCH/R and S-AGCH/R)

Reserved for future use. Not currently supported in the GMR-2 system.

6.14 Half-Rate Robust Standalone Dedicated Control Channel (S-SDCCH/HR)

Reserved for future use. Not currently supported in the GMR-2 system.

Annex A (informative): Summary of satellite channel types

Speech Traffic Channels

- Satellite half rate traffic channel for enhanced speech (S-TCH/HES)
- Satellite half rate traffic channel for robust speech (S-TCH/HRS)
- Satellite quarter rate traffic channel for basic speech (S-TCH/QBS)
- Satellite eighth rate traffic channel for low rate speech (S-TCH/ELS)

Data Traffic Channels

- Satellite full rate traffic channel for 9.6 kbps user data (S-TCH/F9.6)
- Satellite half rate traffic channel for 4.8 kbps user data (S-TCH/H4.8)
- Satellite half rate traffic channel for 2.4 kbps user data (S-TCH/HR2.4)
- Satellite quarter rate traffic channel for 2.4 kbps user data (S-TCH/Q2.4)

Broadcast Control Channels

- Satellite Broadcast Control Channel (S-BCCH)
- Satellite High Margin Broadcast Control Channel (S-HBCCH)
- Satellite Synchronization Channel (S-SCH)

Common Control Channels

- Satellite Random Access Channel (S-RACH)
- Satellite Paging Channel (S-PCH)
- Satellite Robust Paging Channel (S-PCH/R)
- Satellite High Penetration Alerting Channel (S-HPACH)
- Satellite Access Grant Channel (S-AGCH)
- Satellite Robust Access Grant Channel (S-AGCH/R)

Dedicated Control Channels

- Satellite Eighth Rate Standalone Dedicated Control Channel (S-SDCCH/E)
- Satellite Eighth Rate Standalone Dedicated Control Channel (S-SDCCH/Q)
- Satellite Half Rate Robust Standalone Dedicated Control Channel (S-SDCCH/HR)

Slow Associated Control Channels

- Satellite Slow, S-TCH/F Associated, Control Channel (S-SACCH/TF)
- Satellite Slow, S-TCH/H Associated, Control Channel (S-SACCH/TH)
- Satellite Robust S-SACCH/TH (S-SACCH/THR)
- Satellite Slow, S-TCH/Q Associated, Control Channel (S-SACCH/TQ)
- Satellite Slow, S-TCH/E Associated, Control Channel (S-SACCH/TE)
- Satellite Slow, S-SDCCH/E Associated, Control Channel (S-SACCH/CE)

Satellite Slow, S-SDCCH/Q Associated, Control Channel (S-SACCH/CQ)

Satellite Slow, S-SDCCH/HR Associated, Control Channel (S-SACCH/CHR)

Fast Associated Control Channels

Satellite Fast S-TCH/HES Associated Control Channel (S-FACCH/HES)

Satellite Fast S-TCH/QBS Associated Control Channel (S-FACCH/QBS)

Satellite Fast S-TCH/ELS Associated Control Channel (S-FACCH/ELS)

Satellite Fast S-TCH/H4.8 Associated Control Channel (S-FACCH/H4.8)

Satellite Fast S-TCH/Q2.4 Associated Control Channel (S-FACCH/Q2.4)

Satellite Fast S-TCH/F9.6 Associated Control Channel (S-FACCH/F9.6)

Satellite Fast S-TCH/HRS Associated Control Channel (S-FACCH/HRS)

Satellite Fast S-TCH/H2.4 Associated Control Channel (S-FACCH/HR2.4)

Satellite Beam Broadcast Control Channel (S-BBCH)

History

Document history		
V1.1.1	March 2001	Publication