ETSI TS 125 223 V3.6.0 (2001-06)

Technical Specification

Universal Mobile Telecommunications System (UMTS);
Spreading and modulation (TDD)
(3GPP TS 25.223 version 3.6.0 Release 1999)

Reference

RTS/TSGR-0125223UR5

Keywords

UMTS

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

Intellectual Property Rights

Foreword

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

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

The cross reference between GSM, UMTS, 3GPP and ETSI identities can be found under www.etsi.org/key .

Contents

Foreword

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

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

Version x.y.z

where:

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

1 Scope

The present document describes spreading and modulation for UTRA Physical Layer TDD mode.

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

[1] 3GPP TS 25.201: "Physical layer - general description".

[2] 3GPP TS 25.211: "Physical channels and mapping of transport channels onto physical channels (FDD)".

[3] 3GPP TS 25.212: "Multiplexing and channel coding (FDD)".

[4] 3GPP TS 25.213: "Spreading and modulation (FDD)".

[5] 3GPP TS 25.214: "Physical layer procedures (FDD)".

[6] 3GPP TS 25.215: "Physical layer – Measurements (FDD)".

[7] 3GPP TS 25.221: "Physical channels and mapping of transport channels onto physical channels (TDD)".

[8] 3GPP TS 25.222: "Multiplexing and channel coding (TDD)".

[9] 3GPP TS 25.102: "UTRA (UE) TDD; Radio Transmission and Reception".

[10] 3GPP TS 25.105: "UTRA (BS) TDD; Radio Transmission and Reception".

3 Symbols and abbreviations

3.1 Symbols

For the purposes of the present document, the following symbols apply:

C_p: PSC
C_i: i:th secondary SCH code

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

CCTrCH Coded Composite Transport Channel
DPCH Dedicated Physical Channel
CDMA Code Division Multiple Access
FDD Frequency Division Duplex

OVSF	Orthogonal Variable Spreading Factor
P-CCPCH	Primary Common Control Physical Channel
PN	Pseudo Noise
PRACH	Physical Random Access Channel
PSC	Primary Synchronisation Code
QPSK	Quadrature Phase Shift Keying
RACH	Random Access Channel
SCH	Synchronisation Channel
SF	Spreading Factor
SFN	System Frame Number
TDD	Time Division Duplex
TFC	Transport Format Combination
UE	User Equipment
UL	Uplink

4 General

In the following, a separation between the data modulation and the spreading modulation has been made. The data modulation is defined in clause 5 and the spreading modulation in clause 6.

Table 1: Basic modulation parameters

Chip rate	same as FDD basic chiprate: 3.84 Mchip/s	Low chiprate: 1.28 Mchip/s
Data modulation	QPSK	QPSK
Spreading characteristics	Orthogonal Q chips/symbol, where $Q = 2^p$, 0 <= p <= 4	Orthogonal Q chips/symbol, where $Q = 2^p$, 0 <= p <= 4

5 Data modulation

5.1 Symbol rate

The symbol duration T_S depends on the spreading factor Q and the chip duration T_C: $T_s = Q \times T_c$, where $T_c = \frac{1}{chiprate}$.

5.2 Mapping of bits onto signal point constellation

5.2.1 Mapping for burst type 1 and 2

The data modulation is performed to the bits from the output of the physical channel mapping procedure in [8] and combines always 2 consecutive binary bits to a complex valued data symbol. Each user burst has two data carrying parts, termed data blocks:

$$\underline{\mathbf{d}}^{(k,i)} = (\underline{d}_1^{(k,i)}, \underline{d}_2^{(k,i)}, ..., \underline{d}_{N_k}^{(k,i)})^{\mathrm{T}} \quad \mathrm{i} = 1, 2; \mathrm{k} = 1, ..., \mathrm{K}. \tag{1}$$

K is the number of users, max K =16. N_k is the number of symbols per data field for the user k. This number is linked to the spreading factor Q_k as described in table 1 of [7].

Data block $\underline{\mathbf{d}}^{(k,1)}$ is transmitted before the midamble and data block $\underline{\mathbf{d}}^{(k,2)}$ after the midamble. Each of the N_k data symbols $\underline{d}_n^{(k,i)}$; i=1, 2; k=1,...,K; n=1,...,N_k; of equation 1 has the symbol duration $T_s^{(k)} = Q_k.T_c$ as already given.

The data modulation is QPSK, thus the data symbols $\underline{d}_n^{(k,i)}$ are generated from two consecutive data bits from the output of the physical channel mapping procedure in [8]:

$$b_{l,n}^{(k,i)} \in \{0,1\} \qquad l = 1,2;\ k = 1,\ldots K;\ n = 1,\ldots,N_k;\ i = 1,2 \tag{2}$$

using the following mapping to complex symbols:

consecutive binary bit pattern	complex symbol
$b_{1,n}^{(k,i)}\ b_{2,n}^{(k,i)}$	$\underline{d}_n^{(k,i)}$
00	+j
01	+1
10	-1
11	-j

The mapping corresponds to a QPSK modulation of the interleaved and encoded data bits $b_{l,n}^{(k,i)}$ of equation 2.

5.2.2 Mapping for PRACH burst type

In case of PRACH burst type, the definitions in subclause 5.2.1 apply with a modified number of symbols in the second data block. For the PRACH burst type, the number of symbols in the second data block $\underline{\mathbf{d}}^{(k,2)}$ is decreased by $\frac{96}{Q_K}$ symbols.

6 Spreading modulation

6.1 Basic spreading parameters

Spreading of data consists of two operations: Channelisation and Scrambling. Firstly, each complex valued data symbol $\underline{d}_n^{(k,i)}$ of equation 1 is spread with a real valued channelisation code $\mathbf{c}^{(k)}$ of length $Q_k \in \{1,2,4,8,16\}$. The resulting sequence is then scrambled by a complex sequence $\underline{\mathbf{v}}$ of length 16.

6.2 Channelisation codes

The elements $c_q^{(k)}$; k=1,...,K; q=1,...,Q_k; of the real valued channelisation codes

$$\mathbf{c}^{(k)}=(c_1^{(k)},c_2^{(k)},...,c_{Q_k}^{(k)})\ ;\ k=1,...,K;$$

shall be taken from the set

$$V_c = \{1, -1\} \tag{3}$$

The $\mathbf{c}_{Q_k}^{(k)}$ are Orthogonal Variable Spreading Factor (OVSF) codes, allowing to mix in the same timeslot channels with different spreading factors while preserving the orthogonality. The OVSF codes can be defined using the code tree of figure 1.

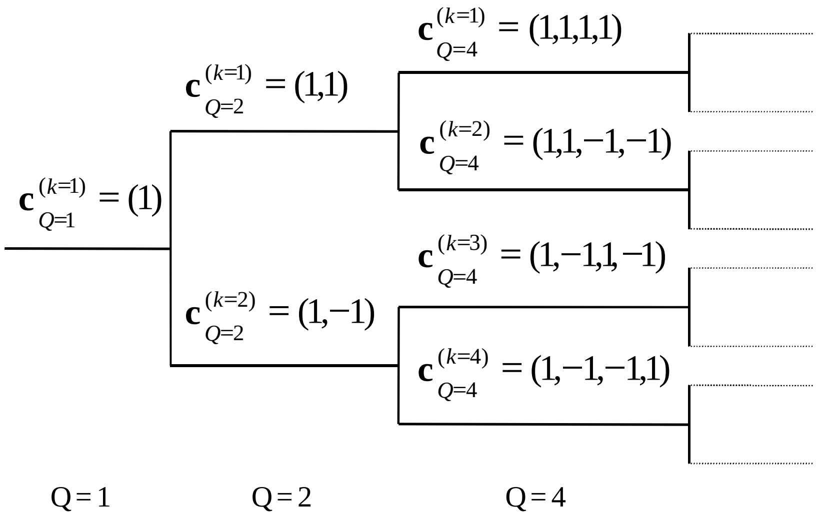

Figure 1: Code-tree for generation of Orthogonal Variable Spreading Factor (OVSF) codes for Channelisation Operation

Each level in the code tree defines a spreading factor indicated by the value of Q in the figure. All codes within the code tree cannot be used simultaneously in a given timeslot. A code can be used in a timeslot if and only if no other code on the path from the specific code to the root of the tree or in the sub-tree below the specific code is used in this timeslot. This means that the number of available codes in a slot is not fixed but depends on the rate and spreading factor of each physical channel.

The spreading factor goes up to Q_{MAX}=16.

6.3 Channelisation Code Specific Multiplier

Associated with each channelisation code is a multiplier $w_{Q_k}^{(k)}$ taking values from the set $\left\{e^{j\pi/2 \cdot p_k}\right\}$, where p_k is a permutation of the integer set {0, ..., Q_k -1} and Q_k denotes the spreading factor. The multiplier is applied to the data sequence modulating each channelisation code. The values of the multiplier for each channelisation code are given in the table below:

k	$w_{Q=1}^{(k)}$	$w_{Q=2}^{(k)}$	$w_{Q=4}^{(k)}$	$w_{Q=8}^{(k)}$	$w_{Q=16}^{(k)}$
1	1	1	-j	1	-1
2		+j	1	+j	-j
3			+j	+j	1
4			-1	-1	1
5				-j	+j
6				-1	-1
7				-j	-1
8				1	1
9					-j
10					+j
11					1
12					+j
13					-j
14					-j
15					+j
16					-1

6.4 Scrambling codes

The spreading of data by a real valued channelisation code $\mathbf{c}^{(k)}$ of length Q_k is followed by a cell specific complex scrambling sequence $\underline{\mathbf{v}} = (\underline{v}_1, \underline{v}_2, \ldots, \underline{v}_{16})$. The elements $\underline{v}_i; i = 1,\ldots,16$ of the complex valued scrambling codes shall be taken from the complex set

$$\underline{V}_{\underline{v}} = \{1, \mathrm{j}, -1, -\mathrm{j}\} \qquad (4)$$

In equation 4 the letter j denotes the imaginary unit. A complex scrambling code $\underline{\mathbf{v}}$ is generated from the binary scrambling codes $\mathbf{v} = (v_1, v_2, \ldots, v_{16})$ of length 16 shown in Annex A. The relation between the elements $\underline{\mathbf{v}}$ and $\mathbf{v}$ is given by:

$$\underline{v}_i = (\mathrm{j})^i \cdot v_i \qquad v_i \in \{1, -1\}, \ \mathrm{i} = 1, \ldots, 16 \qquad (5)$$

Hence, the elements $\underline{v}_i$ of the complex scrambling code $\underline{\mathbf{v}}$ are alternating real and imaginary.

The length matching is obtained by concatenating Q_{MAX}/Q_k spread words before the scrambling. The scheme is illustrated in figure 2 and is described in more detail in subclause 6.4.

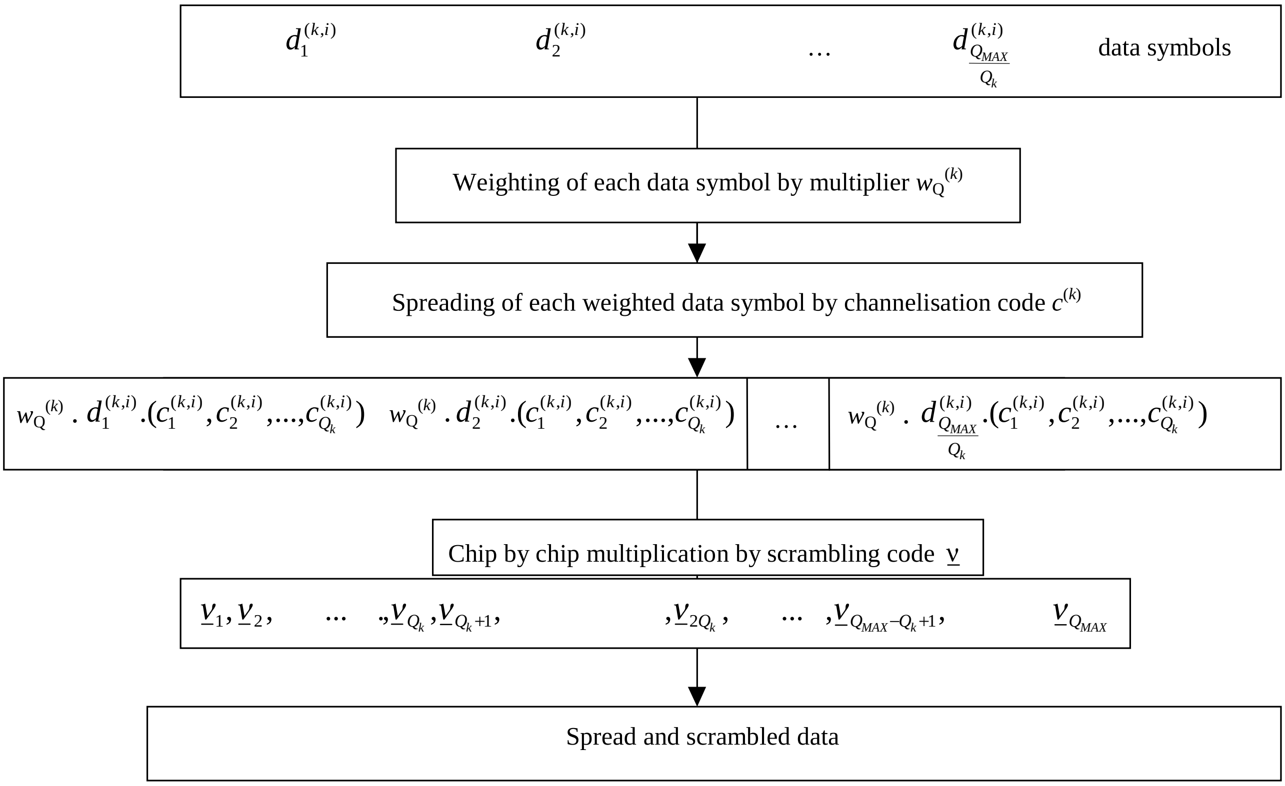

Figure 2: Spreading of data symbols

6.5 Spread signal of data symbols and data blocks

The combination of the user specific channelisation and cell specific scrambling codes can be seen as a user and cell specific spreading code $\mathbf{s}^{(k)} = \left(s_p^{(k)}\right)$ with

$$ {}_p^{(k)} = c^{(k)}_{1+[(p-1) \bmod Q_k]} \cdot \underline{v}_{1+[(p-1) \bmod Q_{MAX}]} $$, k=1,…,K, p=1,…,$N_k Q_k$.

With the root raised cosine chip impulse filter $Cr_0(t)$ the transmitted signal belonging to the data block $\underline{\mathbf{d}}^{(k,1)}$ of equation 1 transmitted before the midamble is

$$ d^{(k,1)}(t) = \sum_{n=1}^{N_k} d_n^{(k,1)} w_{Q_k}^{(k)} \sum_{q=1}^{Q_k} s_{(n-1)Q_k+q}^{(k)} . Cr_0(t-(q-1)T_c-(n-1)Q_k T_c) \tag{6} $$

and for the data block $\underline{\mathbf{d}}^{(k,2)}$ of equation 1 transmitted after the midamble

$$ d^{(k,2)}(t) = \sum_{n=1}^{N_k} d_n^{(k,2)} w_{Q_k}^{(k)} \sum_{q=1}^{Q_k} s_{(n-1)Q_k+q}^{(k)} . Cr_0(t-(q-1)T_c-(n-1)Q_k T_c - N_k Q_k T_c - L_m T_c) \tag{7} $$

where L_m is the number of midamble chips.

6.6 Modulation

The complex-valued chip sequence is QPSK modulated as shown in figure 3.

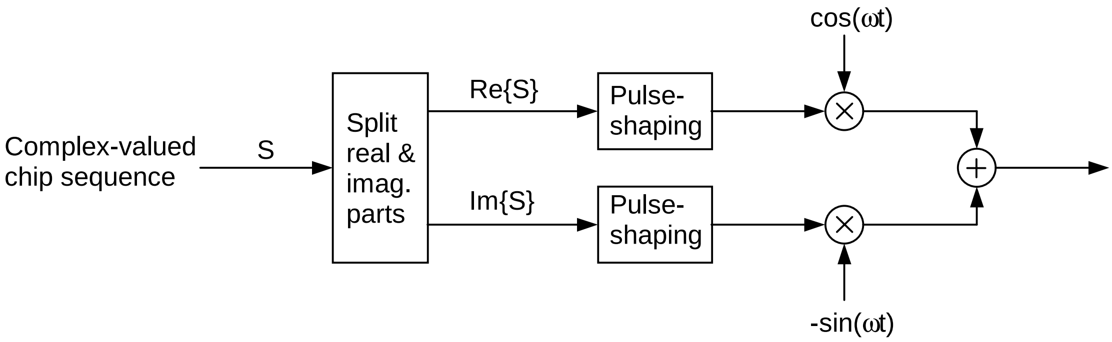

Figure 3: Modulation of complex valued chip sequences

The pulse-shaping characteristics are described in [9] and [10].

6.6.1 Combination of physical channels in uplink

Figure 4 illustrates the principle of combination of two different physical uplink channels within one timeslot. The DPCHs to be combined belong to same CCTrCH, did undergo spreading as described in sections before and are thus represented by complex-valued sequences. First, the amplitude of all DPCHs is adjusted according to UL open loop power control as described in [10]. Each DPCH is then separately weighted by a weight factor γ_i and combined using complex addition. After combination of Physical Channels the gain factor β_j is applied, depending on the actual TFC as described in [10].

In case of different CCTrCH, principle shown in Figure 4 applies to each CCTrCH separately.

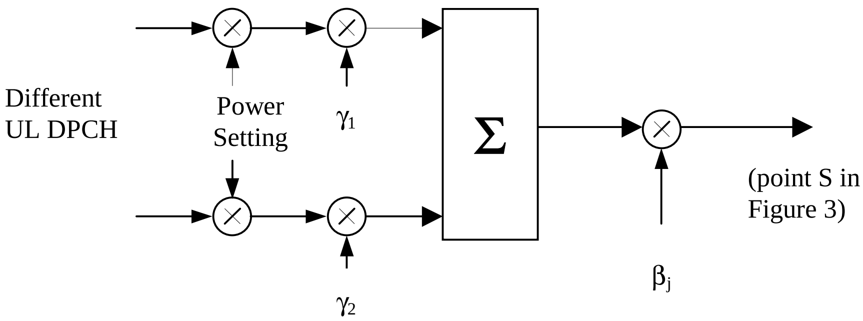

Figure 4: Combination of different physical channels in uplink

The values of weight factors γ_i are depending on the spreading factor SF of the corresponding DPCH:

SF of DPCH$_i$	γ_i
16	1
8	$\sqrt{2}$
4	2
2	$2\sqrt{2}$
1	4

The possible values for gain factors β_j (corresponding to *j*–th TFC) are listed in table below:

Signalling value for β_j	Quantized value β_j
15	16/8
14	15/8
13	14/8
12	13/8
11	12/8
10	11/8
9	10/8
8	9/8
7	8/8
6	7/8
5	6/8
4	5/8
3	4/8
2	3/8
1	2/8
0	1/8

6.6.2 Combination of physical channels in downlink

Figure 5 illustrates how different physical downlink channels are combined within one timeslot. Each complex-valued spread channel is separately weighted by a weight factor G_i. If a timeslot contains the SCH, the complex-valued SCH, as described in [7] is separately weighted by a weight factor G_{SCH}. All downlink physical channels are then combined using complex addition.

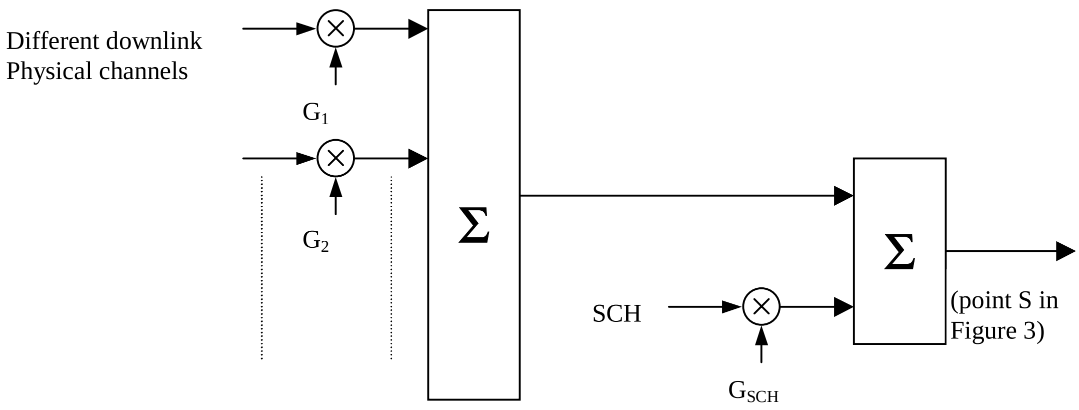

Figure 5: Combination of different physical channels in downlink in case of SCH timeslot

7 Synchronisation codes

7.1 Code Generation

The primary synchronisation code (PSC), C_p , is constructed as a so-called generalised hierarchical Golay sequence. The PSC is furthermore chosen to have good aperiodic auto correlation properties.

Define a = < x_1, x_2, x_3, …, x_{16} > = < 1, 1, 1, 1, 1, 1, -1, -1, 1, -1, 1, -1, 1, -1, -1, 1 >

The PSC is generated by repeating the sequence 'a' modulated by a Golay complementary sequence and creating a complex-valued sequence with identical real and imaginary components.

The PSC, C_p , is defined as C_p = < y(0),y(1),y(2),...,y(255) >

where $y = (1+\mathrm{j}) \times < a, a, a, -a, -a, a, -a, -a, a, a, a, -a, a, -a, a, a >$

and the left most index corresponds to the chip transmitted first in time.

The 12 secondary synchronization codes, {C_0, C_1, C_3, C_4, C_5, C_6, C_8, C_{10}, C_{12}, C_{13}, C_{14}, C_{15} } are complex valued with identical real and imaginary components, and are constructed from the position wise multiplication of a Hadamard sequence and a sequence z, defined as

z = $< b, b, b, -b, b, b, -b, -b, b, -b, b, -b, -b, -b, -b, -b >$, where

b = $< x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, -x_9, -x_{10}, -x_{11}, -x_{12}, -x_{13}, -x_{14}, -x_{15}, -x_{16} >$

and x_1, x_2, x_3, …, x_{16} are the same as in the definition of the sequence 'a' above.

The Hadamard sequences are obtained as the rows in a matrix H_8 constructed recursively by:

$$H_0 = (1)$$
$$H_k = \begin{pmatrix} H_{k-1} & H_{k-1} \\ H_{k-1} & -H_{k-1} \end{pmatrix}, \quad k \geq 1$$

The rows are numbered from the top starting with row *0* (the all ones sequence).

Denote the *n*:th Hadamard sequence h_n as a row of H_8 numbered from the top, n = 0, 1, 2, …, 255, in the sequel.

Furthermore, let $h_m(l)$ and $z(l)$ denote the *l*th symbol of the sequence h_m and *z*, respectively where l = 0, 1, 2, …, 255 and l = 0 corresponds to the leftmost symbol.

The i:th secondary SCH code word, C_i, i = 0, 1, 3, 4, 5, 6, 8, 10, 12, 13, 14, 15 is then defined as

$C_i = (1 + j) \times \langle h_m(0) \times z(0), h_m(1) \times z(1), h_m(2) \times z(2), \ldots, h_m(255) \times z(255) \rangle$,

where $m = (16 \times i)$ and the leftmost chip in the sequence corresponds to the chip transmitted first in time.

7.2 Code Allocation

Three secondary SCH codes are QPSK modulated and transmitted in parallel with the primary synchronization code. The QPSK modulation carries the following information:

- the code group that the base station belongs to (32 code groups:5 bits; Cases 1, 2);
- the position of the frame within an interleaving period of 20 msec (2 frames:1 bit, Cases 1, 2);
- the position of the SCH slot(s) within the frame (2 SCH slots:1 bit, Case 2).

The modulated secondary SCH codes are also constructed such that their cyclic-shifts are unique, i.e. a non-zero cyclic shift less than 2 (Case 1) and 4 (Case 2) of any of the sequences is not equivalent to some cyclic shift of any other of the sequences. Also, a non-zero cyclic shift less than 2 (Case 1) and 4 (Case 2) of any of the sequences is not equivalent to itself with any other cyclic shift less than 8. The secondary synchronization codes are partitioned into two code sets for Case 1 and four code sets for Case 2. The set is used to provide the following information:

Case 1:

Table 2: Code Set Allocation for Case 1

Code Set	Code Group
1	0-15
2	16-31

The code group and frame position information is provided by modulating the secondary codes in the code set.

Case 2:

Table 3: Code Set Allocation for Case 2

Code Set	Code Group
1	0-7
2	8-15
3	16-23
4	24-31

The slot timing and frame position information is provided by the comma free property of the code word and the Code group is provided by modulating some of the secondary codes in the code set.

The following SCH codes are allocated for each code set:

Case 1

Code set 1: C_1, C_3, $C_{5.}$

Code set 2: C_{10}, C_{13}, $C_{14.}$

Case 2

Code set 1: C_1, C_3, $C_{5.}$

Code set 2: C_{10}, C_{13}, $C_{14.}$

Code set 3: C_0, C_6, $C_{12.}$

Code set 4: C_4, C_8, $C_{15.}$

The following subclauses 7.2.1 to 7.2.2 refer to the two cases of SCH/P-CCPCH usage as described in [7].

Note that in the tables 4 and 5 corresponding to Cases 1 and 2, respectively, Frame 1 implies the frame with an odd SFN and Frame 2 implies the frame with an even SFN.

7.2.1 Code allocation for Case 1

Table 4: Code Allocation for Case 1

Code Group	Code Set	Frame 1			Frame 2			Associated t_{offset}
0	1	C_1	C_3	C_5	C_1	C_3	$-C_5$	t_0
1	1	C_1	$-C_3$	C_5	C_1	$-C_3$	$-C_5$	t_1
2	1	$-C_1$	C_3	C_5	$-C_1$	C_3	$-C_5$	t_2
3	1	$-C_1$	$-C_3$	C_5	$-C_1$	$-C_3$	$-C_5$	t_3
4	1	jC_1	jC_3	C_5	jC_1	jC_3	$-C_5$	t_4
5	1	jC_1	$-jC_3$	C_5	jC_1	$-jC_3$	$-C_5$	t_5
6	1	$-jC_1$	jC_3	C_5	$-jC_1$	jC_3	$-C_5$	t_6
7	1	$-jC_1$	$-jC_3$	C_5	$-jC_1$	$-jC_3$	$-C_5$	t_7
8	1	jC_1	jC_5	C_3	jC_1	jC_5	$-C_3$	t_8
9	1	jC_1	$-jC_5$	C_3	jC_1	$-jC_5$	$-C_3$	t_9
10	1	$-jC_1$	jC_5	C_3	$-jC_1$	jC_5	$-C_3$	t_{10}
11	1	$-jC_1$	$-jC_5$	C_3	$-jC_1$	$-jC_5$	$-C_3$	t_{11}
12	1	jC_3	jC_5	C_1	jC_3	jC_5	$-C_1$	t_{12}
13	1	jC_3	$-jC_5$	C_1	jC_3	$-jC_5$	$-C_1$	t_{13}
14	1	$-jC_3$	jC_5	C_1	$-jC_3$	jC_5	$-C_1$	t_{14}
15	1	$-jC_3$	$-jC_5$	C_1	$-jC_3$	$-jC_5$	$-C_1$	t_{15}
16	2	C_{10}	C_{13}	C_{14}	C_{10}	C_{13}	$-C_{14}$	t_{16}
17	2	C_{10}	$-C_{13}$	C_{14}	C_{10}	$-C_{13}$	$-C_{14}$	t_{17}
…	…	…	…	…	…	…	…	…
20	2	jC_{10}	jC_{13}	C_{14}	jC_{10}	jC_{13}	$-C_{14}$	t_{20}
…	…	…	…	…	…	…	…	…
24	2	jC_{10}	jC_{14}	C_{13}	jC_{10}	jC_{14}	$-C_{13}$	t_{24}
…	…	…	…	…	…	…	…	…
31	2	$-jC_{13}$	$-jC_{14}$	C_{10}	$-jC_{13}$	$-jC_{14}$	$-C_{10}$	t_{31}

NOTE: The code construction for code groups 0 to 15 using only the SCH codes from code set 1 is shown. The construction for code groups 16 to 31 using the SCH codes from code set 2 is done in the same way.

7.2.2 Code allocation for Case 2

Table 5: Code Allocation for Case 2

Code Group	Code Set	Frame 1						Frame 2						Associated t_{offset}
		Slot k			Slot k+8			Slot k			Slot k+8			
0	1	C_1	C_3	C_5	C_1	C_3	$-C_5$	$-C_1$	$-C_3$	C_5	$-C_1$	$-C_3$	$-C_5$	t_0
1	1	C_1	$-C_3$	C_5	C_1	$-C_3$	$-C_5$	$-C_1$	C_3	C_5	$-C_1$	C_3	$-C_5$	t_1
2	1	jC_1	jC_3	C_5	jC_1	jC_3	$-C_5$	$-jC_1$	$-jC_3$	C_5	$-jC_1$	$-jC_3$	$-C_5$	t_2
3	1	jC_1	$-jC_3$	C_5	jC_1	$-jC_3$	$-C_5$	$-jC_1$	jC_3	C_5	$-jC_1$	jC_3	$-C_5$	t_3
4	1	jC_1	jC_5	C_3	jC_1	jC_5	$-C_3$	$-jC_1$	$-jC_5$	C_3	$-jC_1$	$-jC_5$	$-C_3$	t_4
5	1	jC_1	$-jC_5$	C_3	jC_1	$-jC_5$	$-C_3$	$-jC_1$	jC_5	C_3	$-jC_1$	jC_5	$-C_3$	t_5
6	1	jC_3	jC_5	C_1	jC_3	jC_5	$-C_1$	$-jC_3$	$-jC_5$	C_1	$-jC_3$	$-jC_5$	$-C_1$	t_6
7	1	jC_3	$-jC_5$	C_1	jC_3	$-jC_5$	$-C_1$	$-jC_3$	jC_5	C_1	$-jC_3$	jC_5	$-C_1$	t_7
8	2	C_{10}	C_{13}	C_{14}	C_{10}	C_{13}	$-C_{14}$	$-C_{10}$	$-C_{13}$	C_{14}	$-C_{10}$	$-C_{13}$	$-C_{14}$	t_8
9	2	C_{10}	$-C_{13}$	C_{14}	C_{10}	$-C_{13}$	$-C_{14}$	$-C_{10}$	C_{13}	C_{14}	$-C_{10}$	C_{13}	$-C_{14}$	t_9
10	2	jC_{10}	jC_{13}	C_{14}	jC_{10}	jC_{13}	$-C_{14}$	$-jC_{10}$	$-jC_{13}$	C_{14}	$-jC_{10}$	$-jC_{13}$	$-C_{14}$	t_{10}
11	2	jC_{10}	$-jC_{13}$	C_{14}	jC_{10}	$-jC_{13}$	$-C_{14}$	$-jC_{10}$	jC_{13}	C_{14}	$-jC_{10}$	jC_{13}	$-C_{14}$	t_{11}
12	2	jC_{10}	jC_{14}	C_{13}	jC_{10}	jC_{14}	$-C_{13}$	$-jC_{10}$	$-jC_{14}$	C_{13}	$-jC_{10}$	$-jC_{14}$	$-C_{13}$	t_{12}
13	2	jC_{10}	$-jC_{14}$	C_{13}	jC_{10}	$-jC_{14}$	$-C_{13}$	$-jC_{10}$	jC_{14}	C_{13}	$-jC_{10}$	jC_{14}	$-C_{13}$	t_{13}
14	2	jC_{13}	jC_{14}	C_{10}	jC_{13}	jC_{14}	$-C_{10}$	$-jC_{13}$	$-jC_{14}$	C_{10}	$-jC_{13}$	$-jC_{14}$	$-C_{10}$	t_{14}
15	2	jC_{13}	$-jC_{14}$	C_{10}	jC_{13}	$-jC_{14}$	$-C_{10}$	$-jC_{13}$	jC_{14}	C_{10}	$-jC_{13}$	jC_{14}	$-C_{10}$	t_{15}
16	3	C_0	C_6	C_{12}	C_0	C_6	$-C_{12}$	$-C_0$	$-C_6$	C_{12}	$-C_0$	$-C_6$	$-C_{12}$	t_{16}
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
23	3	jC_6	$-jC_{12}$	C_0	jC_6	$-jC_{12}$	$-C_0$	$-jC_6$	jC_{12}	C_0	$-jC_6$	jC_{12}	$-C_0$	t_{20}
24	4	C_4	C_8	C_{15}	C_4	C_8	$-C_{15}$	$-C_4$	$-C_8$	C_{15}	$-C_4$	$-C_8$	$-C_{15}$	t_{24}
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
31	4	jC_8	$-jC_{15}$	C_4	jC_8	$-jC_{15}$	$-C_4$	$-jC_8$	jC_{15}	C_4	$-jC_8$	jC_{15}	$-C_4$	t_{31}

NOTE: The code construction for code groups 0 to 15 using the SCH codes from code sets 1 and 2 is shown. The construction for code groups 16 to 31 using the SCH codes from code sets 3 and 4 is done in the same way.

7.3 Evaluation of synchronisation codes

The evaluation of information transmitted in SCH on code group and frame timing is shown in table 6, where the 32 code groups are listed. Each code group is containing 4 specific scrambling codes (cf. subclause 6.3), each scrambling code associated with a specific short and long basic midamble code.

Each code group is additionally linked to a specific t_{Offset}, thus to a specific frame timing. By using this scheme, the UE can derive the position of the frame border due to the position of the SCH sequence and the knowledge of t_{Offset}. The complete mapping of Code Group to Scrambling Code, Midamble Codes and t_{Offset} is depicted in table 6.

Table 6: Mapping scheme for Cell Parameters, Code Groups, Scrambling Codes, Midambles and t_{Offset}

CELL PARA-METER	Code Group	Associated Codes			Associated t_{Offset}
		Scrambling Code	Long Basic Midamble Code	Short Basic Midamble Code	
0	Group 0	Code 0	m_{PL0}	m_{SL0}	t_0
1		Code 1	m_{PL1}	m_{SL1}	
2		Code 2	m_{PL2}	m_{SL2}	
3		Code 3	m_{PL3}	m_{SL3}	
4	Group 1	Code 4	m_{PL4}	m_{SL4}	t_1
5		Code 5	m_{PL5}	m_{SL5}	
6		Code 6	m_{PL6}	m_{SL6}	
7		Code 7	m_{PL7}	m_{SL7}	
. . . .					
124	Group 31	Code 124	m_{PL124}	m_{SL124}	t_{31}
125		Code 125	m_{PL125}	m_{SL125}	
126		Code 126	m_{PL126}	m_{SL126}	
127		Code 127	m_{PL127}	m_{SL127}	

For basic midamble codes m_P cf. [7], annex A 'Basic Midamble Codes'.

Each cell shall cycle through two sets of cell parameters in a code group with the cell parameters changing each frame. Table 7 shows how the cell parameters are cycled according to the SFN.

Table 7: Alignment of cell parameter cycling and SFN

Initial Cell Parameter Assignment	Code Group	Cell Parameter used when SFN mod 2 = 0	Cell Parameter used when SFN mod 2 = 1
0	Group 0	0	1
1		1	0
2		2	3
3		3	2
4	Group 1	4	5
5		5	4
6		6	7
7		7	6
. . . .			
124	Group 31	124	125
125		125	124
126		126	127
127		127	126

Annex A (normative): Scrambling Codes

The applicable scrambling codes are listed below. Code numbers are referring to table 6 'Mapping scheme for Cell Parameters, Code Groups, Scrambling Codes, Midambles and t_{offset}' in subclause 7.3 'Evaluation of synchronisation codes'.

Scrambling Code	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	v_{15}	v_{16}
Code 0	-1	1	-1	-1	-1	1	-1	-1	1	-1	1	1	-1	1	-1	-1
Code 1	1	1	1	1	1	-1	1	-1	1	-1	-1	1	1	1	-1	-1
Code 2	1	-1	1	1	1	-1	1	1	-1	1	1	1	1	-1	-1	-1
Code 3	1	1	1	-1	-1	-1	-1	1	-1	-1	1	-1	-1	-1	1	-1
Code 4	1	1	1	-1	-1	-1	-1	1	1	1	1	-1	1	1	1	-1
Code 5	-1	1	1	-1	-1	-1	1	1	1	1	1	1	1	-1	1	-1
Code 6	-1	1	-1	-1	-1	1	-1	-1	-1	1	1	1	1	-1	-1	-1
Code 7	1	-1	1	-1	-1	-1	-1	-1	1	1	-1	-1	-1	1	1	-1
Code 8	1	1	1	-1	-1	-1	1	-1	1	1	-1	1	1	1	1	-1
Code 9	1	1	-1	1	1	1	1	-1	1	1	1	-1	-1	-1	1	-1
Code 10	1	-1	1	-1	1	1	1	1	-1	-1	1	1	-1	1	1	-1
Code 11	-1	1	1	1	1	-1	-1	-1	-1	1	-1	-1	-1	1	-1	-1
Code 12	-1	-1	1	-1	-1	-1	1	-1	-1	-1	-1	1	1	1	1	-1
Code 13	1	-1	1	1	1	-1	-1	-1	1	-1	-1	-1	-1	1	-1	-1
Code 14	1	-1	-1	-1	-1	1	-1	-1	1	-1	1	1	1	-1	-1	-1
Code 15	1	1	-1	-1	-1	1	1	-1	1	-1	1	-1	-1	-1	-1	-1
Code 16	1	-1	-1	1	-1	1	-1	1	-1	-1	-1	-1	1	1	-1	-1
Code 17	1	1	1	-1	1	1	1	-1	1	1	-1	1	-1	-1	1	-1
Code 18	-1	1	1	1	-1	1	-1	-1	-1	1	-1	-1	1	-1	-1	-1
Code 19	-1	1	-1	-1	1	-1	-1	-1	-1	1	1	1	-1	1	-1	-1
Code 20	-1	-1	-1	-1	1	-1	1	-1	-1	1	1	-1	1	1	-1	-1
Code 21	1	1	1	1	-1	-1	1	1	-1	1	1	-1	1	-1	1	-1
Code 22	1	-1	-1	-1	-1	1	1	1	-1	1	-1	-1	-1	1	-1	-1
Code 23	-1	1	1	1	-1	1	1	1	1	-1	1	1	-1	1	-1	-1
Code 24	-1	-1	1	-1	1	1	1	-1	-1	-1	-1	1	-1	-1	1	-1
Code 25	1	-1	1	1	1	-1	1	1	1	-1	1	1	-1	1	-1	-1
Code 26	1	-1	-1	-1	1	-1	-1	-1	-1	1	1	1	1	-1	-1	-1
Code 27	-1	1	-1	-1	-1	1	1	1	1	-1	-1	-1	-1	1	-1	-1
Code 28	-1	-1	-1	1	-1	-1	-1	1	-1	-1	-1	1	1	1	1	-1
Code 29	1	-1	1	1	-1	1	-1	-1	-1	1	-1	-1	-1	1	-1	-1
Code 30	-1	-1	-1	-1	-1	-1	1	1	1	-1	-1	1	1	-1	1	-1
Code 31	1	1	-1	-1	1	1	1	1	-1	1	-1	1	-1	1	1	-1
Code 32	1	-1	-1	-1	1	-1	1	1	-1	1	-1	-1	1	-1	-1	-1
Code 33	-1	-1	-1	1	1	1	1	-1	1	1	1	-1	1	1	1	-1
Code 34	1	-1	-1	-1	1	-1	-1	-1	1	-1	1	1	-1	1	-1	-1
Code 35	1	-1	1	1	-1	1	-1	-1	1	-1	-1	-1	1	-1	-1	-1
Code 36	1	1	-1	1	1	1	-1	1	-1	-1	-1	1	1	1	1	-1
Code 37	-1	-1	-1	1	-1	-1	1	-1	-1	-1	1	-1	1	1	1	-1
Code 38	-1	1	-1	-1	1	-1	1	1	1	-1	-1	-1	1	-1	-1	-1
Code 39	-1	1	1	1	1	-1	-1	-1	1	-1	-1	-1	1	-1	-1	-1
Code 40	-1	1	-1	1	-1	-1	-1	-1	-1	-1	1	1	-1	1	1	-1
Code 41	1	1	-1	1	-1	-1	1	-1	-1	-1	1	-1	-1	-1	1	-1
Code 42	1	-1	-1	-1	-1	1	1	1	1	-1	-1	-1	1	-1	-1	-1
Code 43	-1	-1	1	1	-1	-1	-1	-1	-1	1	-1	1	-1	1	1	-1
Code 44	-1	-1	1	-1	-1	-1	-1	1	1	1	1	-1	-1	-1	1	-1

Scrambling Code	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	v_{15}	v_{16}
Code 45	-1	-1	1	-1	1	1	-1	1	1	1	1	-1	1	1	1	-1
Code 46	-1	1	1	-1	1	1	-1	-1	-1	-1	-1	-1	1	-1	1	-1
Code 47	1	-1	-1	1	1	1	-1	-1	1	1	1	1	1	-1	1	-1
Code 48	1	1	-1	1	1	1	-1	1	1	1	-1	1	-1	-1	1	-1
Code 49	-1	-1	1	1	-1	1	1	-1	-1	1	-1	1	-1	-1	-1	-1
Code 50	1	1	-1	1	-1	-1	1	-1	1	1	1	-1	1	1	1	-1
Code 51	1	-1	-1	1	1	1	-1	-1	1	-1	1	-1	-1	-1	-1	-1
Code 52	1	1	1	-1	1	1	1	-1	-1	-1	-1	1	1	1	1	-1
Code 53	-1	1	1	1	-1	-1	-1	1	-1	1	1	1	1	1	1	-1
Code 54	-1	-1	1	-1	-1	-1	1	-1	1	1	-1	1	-1	-1	1	-1
Code 55	-1	1	1	-1	-1	-1	-1	-1	1	-1	1	-1	1	1	-1	-1
Code 56	-1	1	1	1	-1	1	1	1	-1	1	1	1	1	-1	-1	-1
Code 57	-1	1	1	-1	-1	-1	1	1	-1	1	-1	1	-1	-1	-1	-1
Code 58	-1	1	-1	1	-1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1
Code 59	1	1	-1	-1	-1	-1	-1	-1	1	-1	1	-1	-1	1	1	-1
Code 60	-1	1	1	-1	1	1	1	1	-1	1	-1	1	1	1	-1	-1
Code 61	-1	-1	1	1	1	-1	-1	1	1	-1	1	-1	-1	-1	-1	-1
Code 62	-1	1	-1	-1	1	1	1	-1	1	-1	-1	-1	-1	-1	1	-1
Code 63	-1	1	-1	1	-1	-1	1	1	1	-1	-1	1	-1	-1	-1	-1
Code 64	1	-1	-1	1	-1	-1	1	1	-1	-1	-1	-1	1	-1	1	-1
Code 65	-1	-1	-1	1	1	1	1	-1	-1	-1	1	-1	-1	-1	1	-1
Code 66	-1	-1	-1	-1	1	-1	-1	1	1	1	-1	-1	1	-1	1	-1
Code 67	-1	-1	-1	1	1	1	-1	1	1	1	-1	1	1	1	1	-1
Code 68	1	-1	1	1	-1	-1	-1	1	1	-1	-1	-1	-1	-1	1	-1
Code 69	-1	-1	1	-1	1	-1	-1	-1	1	1	1	-1	-1	1	-1	-1
Code 70	1	1	-1	1	-1	-1	-1	1	-1	-1	-1	1	-1	-1	1	-1
Code 71	1	-1	-1	1	-1	-1	-1	-1	-1	1	-1	1	1	1	-1	-1
Code 72	1	1	1	1	-1	1	1	-1	1	1	-1	-1	1	-1	1	-1
Code 73	-1	1	1	1	-1	-1	-1	1	-1	1	-1	-1	-1	-1	1	-1
Code 74	1	1	-1	1	-1	1	-1	-1	-1	-1	-1	1	1	-1	-1	-1
Code 75	1	1	-1	-1	1	-1	-1	1	-1	1	-1	1	-1	-1	-1	-1
Code 76	-1	1	-1	-1	-1	-1	-1	1	-1	1	1	1	-1	-1	1	-1
Code 77	-1	1	-1	1	1	1	1	1	-1	1	1	-1	1	1	-1	-1
Code 78	-1	1	-1	1	-1	1	1	-1	-1	-1	1	1	-1	-1	-1	-1
Code 79	-1	1	-1	1	1	1	-1	-1	-1	1	1	-1	-1	-1	-1	-1
Code 80	1	1	-1	1	-1	1	-1	-1	-1	-1	1	-1	-1	1	-1	-1
Code 81	1	1	1	1	1	-1	1	-1	-1	-1	1	1	-1	1	1	-1
Code 82	-1	1	-1	1	1	1	1	1	1	1	-1	-1	-1	1	1	-1
Code 83	1	1	-1	-1	1	-1	1	-1	-1	-1	-1	-1	-1	1	1	-1
Code 84	-1	-1	1	-1	1	1	-1	1	-1	-1	1	-1	-1	-1	1	-1
Code 85	-1	1	1	-1	-1	1	-1	1	1	1	1	1	1	1	-1	-1
Code 86	-1	-1	-1	1	-1	-1	-1	1	1	1	-1	1	-1	-1	1	-1
Code 87	1	1	-1	-1	-1	1	-1	1	1	1	1	1	-1	1	1	-1
Code 88	-1	1	1	-1	1	1	-1	-1	1	-1	1	-1	-1	-1	-1	-1
Code 89	-1	1	-1	-1	1	-1	1	1	-1	1	-1	-1	-1	1	-1	-1
Code 90	1	-1	-1	-1	-1	-1	-1	1	1	-1	1	1	-1	-1	1	-1
Code 91	-1	1	-1	-1	-1	-1	1	-1	1	-1	1	1	-1	-1	1	-1
Code 92	-1	1	1	-1	1	-1	1	-1	-1	-1	-1	-1	1	1	-1	-1
Code 93	-1	-1	-1	-1	-1	1	1	-1	-1	-1	1	1	1	-1	1	-1
Code 94	1	-1	1	-1	-1	1	1	-1	1	1	-1	-1	-1	-1	-1	-1
Code 95	1	1	1	1	1	-1	-1	1	-1	-1	1	1	1	-1	1	-1
Code 96	1	1	-1	-1	-1	1	1	-1	-1	-1	-1	-1	1	-1	1	-1
Code 97	1	1	-1	-1	1	-1	-1	1	1	1	1	1	1	-1	1	-1

Scrambling Code	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	v_{15}	v_{16}
Code 98	1	1	-1	1	1	-1	1	1	1	1	1	-1	1	-1	-1	-1
Code 99	1	-1	1	-1	1	-1	-1	1	-1	-1	1	1	-1	-1	-1	-1
Code 100	1	-1	1	1	-1	-1	1	-1	-1	1	-1	-1	-1	-1	1	-1
Code 101	1	1	1	1	-1	1	-1	1	1	1	-1	-1	-1	1	1	-1
Code 102	1	-1	1	-1	1	1	1	1	-1	1	1	-1	1	1	-1	-1
Code 103	-1	-1	1	-1	-1	1	-1	-1	1	1	1	-1	1	-1	-1	-1
Code 104	1	-1	1	1	-1	1	1	1	-1	1	1	1	-1	1	-1	-1
Code 105	1	1	1	1	1	1	-1	-1	1	-1	-1	1	1	-1	1	-1
Code 106	1	1	-1	-1	-1	1	-1	1	-1	-1	-1	-1	-1	1	1	-1
Code 107	-1	-1	-1	-1	1	1	-1	-1	-1	1	1	-1	1	-1	1	-1
Code 108	-1	-1	-1	1	-1	1	-1	-1	1	1	-1	1	1	-1	-1	-1
Code 109	-1	1	-1	1	1	-1	-1	1	1	1	-1	-1	-1	-1	-1	-1
Code 110	-1	-1	1	1	-1	1	-1	1	1	1	1	1	-1	1	1	-1
Code 111	1	1	1	-1	-1	1	1	1	1	1	-1	1	-1	1	-1	-1
Code 112	-1	-1	1	1	1	-1	1	-1	1	1	1	1	-1	1	1	-1
Code 113	1	1	-1	-1	1	-1	1	-1	1	1	1	1	-1	1	1	-1
Code 114	-1	-1	-1	1	1	-1	-1	-1	1	1	-1	1	-1	1	-1	-1
Code 115	1	-1	-1	1	1	1	1	1	1	-1	1	-1	1	1	-1	-1
Code 116	-1	1	1	1	1	-1	1	1	1	-1	1	1	1	-1	-1	-1
Code 117	1	1	1	-1	1	1	-1	1	-1	-1	1	-1	1	1	1	-1
Code 118	-1	-1	-1	-1	-1	1	-1	1	1	-1	-1	1	1	1	-1	-1
Code 119	-1	-1	-1	1	-1	1	1	1	-1	-1	1	-1	-1	1	-1	-1
Code 120	-1	-1	1	-1	1	-1	1	1	-1	-1	1	-1	-1	1	-1	-1
Code 121	-1	1	1	1	1	1	1	-1	1	-1	1	1	-1	-1	1	-1
Code 122	-1	-1	-1	1	1	-1	1	1	-1	-1	1	-1	1	-1	-1	-1
Code 123	1	-1	1	-1	1	1	-1	-1	1	-1	-1	1	-1	-1	-1	-1
Code 124	-1	-1	1	1	1	1	1	1	1	-1	1	-1	-1	1	1	-1
Code 125	1	-1	-1	1	1	-1	1	-1	1	1	1	1	1	1	-1	-1
Code 126	1	1	1	1	-1	1	-1	1	-1	1	1	-1	1	1	-1	-1
Code 127	1	-1	1	-1	-1	-1	-1	-1	1	-1	-1	1	1	1	-1	-1

Annex B (informative): Generalised Hierarchical Golay Sequences

B.1 Alternative generation

The generalised hierarchical Golay sequences for the PSC described in 7.1 may be also viewed as generated (in real valued representation) by the following methods:

Method 1.

The sequence y is constructed from two constituent sequences x_1 and x_2 of length n_1 and n_2 respectively using the following formula:

- $y(i) = x_2(i \bmod n_2) * x_1(i \text{ div } n_2)$, $i = 0 \ldots (n_1 * n_2) - 1$.

The constituent sequences x_1 and x_2 are chosen to be the following length 16 (i.e. $n_1 = n_2 = 16$) sequences:

- x_1 is defined to be the length 16 ($N^{(1)}=4$) Golay complementary sequence obtained by the delay matrix $D^{(1)} = [8, 4, 1,2]$ and weight matrix $W^{(1)} = [1, -1, 1,1]$.
- x_2 is a generalised hierarchical sequence using the following formula, selecting s=2 and using the two Golay complementary sequences x_3 and x_4 as constituent sequences. The length of the sequence x_3 and x_4 is called n_3 respectively n_4.
- $x_2(i) = x_4(i \bmod s + s*(i \text{ div } sn_3)) * x_3((i \text{ div } s) \bmod n_3)$, $i = 0 \ldots (n_3 * n_4) - 1$.
- x_3 and x_4 are defined to be identical and the length 4 ($N^{(3)}= N^{(4)}=2$) Golay complementary sequence obtained by the delay matrix $D^{(3)} = D^{(4)} = [1, 2]$ and weight matrix $W^{(3)} = W^{(4)} = [1, 1]$.

The Golay complementary sequences x_1,x_3 and x_4 are defined using the following recursive relation:

$a_0(k) = \delta(k)$ and $b_0(k) = \delta(k)$;

$a_n(k) = a_{n-1}(k) + W^{(j)}_n \cdot b_{n-1}(k-D^{(j)}_n)$;

$b_n(k) = a_{n-1}(k) - W^{(j)}_n \cdot b_{n-1}(k-D^{(j)}_n)$;

$k = 0, 1, 2, \ldots, 2^{**}N^{(j)} -1$;

$n = 1, 2, \ldots, N^{(j)}$.

The wanted Golay complementary sequence x_j is defined by a_n assuming $n=N^{(j)}$. The Kronecker delta function is described by δ, k,j and n are integers.

Method 2

The sequence y can be viewed as a pruned Golay complementary sequence and generated using the following parameters which apply to the generator equations for a and b above:

(a) Let j = 0, $N^{(0)} = 8$.

(b) $[D_1^0,D_2^0,D_3^0,D_4^0,D_5^0,D_6^0,D_7^0,D_8^0] = [128, 64, 16, 32, 8, 1, 4, 2]$.

(c) $[W_1^0,W_2^0,W_3^0,W_4^0,W_5^0,W_6^0,W_7^0,W_8^0] = [1, -1, 1, 1, 1, 1, 1, 1]$.

(d) For n = 4, 6, set $b_4(k) = a_4(k)$, $b_6(k) = a_6(k)$.

Annex C (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
14/01/00	RAN_05	RP-99593	-		Approved at TSG RAN #5 and placed under Change Control	-	3.0.0
14/01/00	RAN_06	RP-99696	001	01	Primary and Secondary CCPCH in TDD	3.0.0	3.1.0
14/01/00	RAN_06	RP-99695	003	1	Alignment of Terminology Regarding Spreading for TDD Mode	3.0.0	3.1.0
14/01/00	RAN_06	RP-99696	004	-	Code allocation for Case 3	3.0.0	3.1.0
14/01/00	-	-	-		Change history was added by the editor	3.1.0	3.1.1
31/03/00	RAN_07	RP-000069	002	3	Cycling of cell parameters	3.1.1	3.2.0
31/03/00	RAN_07	RP-000069	005	-	Removal of Synchronisation Case 3 in TDD	3.1.1	3.2.0
31/03/00	RAN_07	RP-000069	006	1	Signal Point Constellation	3.1.1	3.2.0
03/05/00	-	-	-	-	Revision marks accepted to create clean version	3.2.0	3.2.1
26/06/00	RAN_08	RP-000273	008	-	Editorial Modifications for 25.223	3.2.1	3.3.0
26/06/00	RAN_08	RP-000273	009	-	Editorial modification of 25.223	3.2.1	3.3.0
26/06/00	RAN_08	RP-000273	010	-	Editorial modification of 25.223	3.2.1	3.3.0
26/06/00	RAN_08	RP-000273	011	2	Editorial modification of 25.223	3.2.1	3.3.0
26/06/00	RAN_08	RP-000273	012	2	Modified code sets on SCH for cell search in UTRA TDD	3.2.1	3.3.0
26/06/00	RAN_08	RP-000273	013	1	Editorial update of TS25.223	3.2.1	3.3.0
23/09/00	RAN_09	RP-000346	007	1	Gain Factors for TDD Mode	3.3.0	3.4.0
23/09/00	RAN_09	RP-000346	014	-	Synchronisation codes	3.3.0	3.4.0
16/03/01	RAN_11	RP-010064	015	1	Code specific phase offsets for TDD	3.4.0	3.5.0
15/06/01	RAN_12	RP-010337	018	-	Addition to the abbreviation list and definition of a constant	3.5.0	3.6.0

History

Document history		
V3.1.1	January 2000	Publication
V3.2.0	March 2000	Publication
V3.3.0	June 2000	Publication
V3.4.0	September 2000	Publication
V3.5.0	March 2001	Publication
V3.6.0	June 2001	Publication