3GPP (S1.13) V1.1.01999-3

3GPP FDD, spreading and modulation specification

Reference

xxxx

Keywords

Digital cellular telecommunications system,
Universal Mobile Telecommunication System
(UMTS), UTRAN, 3GPP

ETSI Secretariat

Postal address

F-06921 Sophia Antipolis Cedex - FRANCE

Office address

650 Route des Lucioles - Sophia Antipolis
Valbonne - 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

X.400

c= fr; a=atlas; p=etsi; s=secretariat

Internet

secretariat@etsi.fr
http://www.etsi.fr

Contents

1 Intellectual Property Rights

IPRs essential or potentially essential to the present deliverable may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members, free of charge**. This can be found in the latest version of the ETSI Technical Report: ETR 314: "Intellectual Property Rights (IPRs); Essential or potentially Essential, IPRs notified to ETSI in respect of ETSI standards". The most recent update of ETR 314, is available on the ETSI web server or on request from the Secretariat.

Pursuant to the ETSI Interim 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 the ETR 314, which are, or may be, or may become, essential to the present document.

2 Foreword

This Technical Report (TR) has been produced by the Special Mobile Group (SMG) of the European Telecommunications Standards Institute (ETSI).

This TR describes spreading and modulation for UTRA Physical Layer FDD mode.

The contents of this TR are subject to continuing work within SMG2 and SMG2 UMTS layer 1 expert group and may change following approval by either of these two groups.

<Editor’s note: this boilerplate text should be replaced by 3GPP equivalent.>

3 Scope

This European Telecommunication Report (ETR) describes spreading and modulation for UTRA Physical Layer FDD mode.

Text without revision marks has been approved in the previous SMG2 Layer 1 expert group meetings, while text with revision marks is subject to approval.

3.1 Status

The status of all the sections in this specification is working assumption.

4 References

References may be made to:

a) specific versions of publications (identified by date of publication, edition number, version number, etc.), in which case, subsequent revisions to the referenced document do not apply;

b) publications without mention of a specific version, in which case the latest version applies.

A non-specific reference to an ETS shall also be taken to refer to later versions published as an EN with the same number.

5 Definitions and abbreviations

BCH	Broadcast Control Channel
BER	Bit Error Rate
BS	Base Station

CCPCH	Common Control Physical Channel
DCH	Dedicated Channel
DL	Downlink (Forward link)
DPCH	Dedicated Physical Channel
DPCCH	Dedicated Physical Control Channel
DPDCH	Dedicated Physical Data Channel
DS-CDMA	Direct-Sequence Code Division Multiple Access
FACH	Forward Access Channel
FDD	Frequency Division Duplex
Mcps	Mega Chip Per Second
MS	Mobile Station
OVSF	Orthogonal Variable Spreading Factor (codes)
PCH	Paging Channel
PG	Processing Gain
PRACH	Physical Random Access Channel
RACH	Random Access Channel
RX	Receive
SCH	Synchronisation Channel
SF	Spreading Factor
SIR	Signal-to-Interference Ratio
TDD	Time Division Duplex
TFCI	Transport-Format Combination Indicator
TPC	Transmit Power Control
TX	Transmit
UE	User Equipment
UL	Uplink (Reverse link)

6 Uplink spreading and modulation

6.1 Overview

Spreading is applied after modulation and before pulse shaping. It consists of two operations. The first is the spreading operation, which transforms every data symbol into a number of chips, thus increasing the bandwidth of the signal. The number of chips per data symbol is called the Spreading Factor (SF). The second operation is the scrambling operation, where a scrambling code is applied to the spread signal.

With the spreading, data symbol on so-called I- and Q-branches are independently multiplied with spreading code. With the scrambling operation, the resultant signals on the I- and Q-branches are further multiplied by complex-valued scrambling code, where I and Q denote real and imaginary parts, respectively. Note that before complex multiplication binary values 0 and 1 are mapped to +1 and -1, respectively.

6.2 Spreading

6.2.1 Uplink Dedicated Physical Channels (uplink DPDCH/DPCCH)

Figure 1 illustrates the spreading and modulation for the case of multiple uplink DPDCHs when total data rate is less than or equal to 1024kbps in the 5MHz band. Note that this figure only shows the principle, and does not necessarily describe an actual implementation. Figure 2 illustrates the case for data rate at 2048kbps in the 5 MHz band. Modulation is dual-channel QPSK (i.e.; separate BPSK on I- and Q-channel), where the uplink DPDCH and DPCCH are mapped to the I and Q branch respectively. The I and Q branches are then spread to the chip rate with two different channelization codes and subsequently complex scrambled by a UEspecific complex scrambling code C_{scramb}.

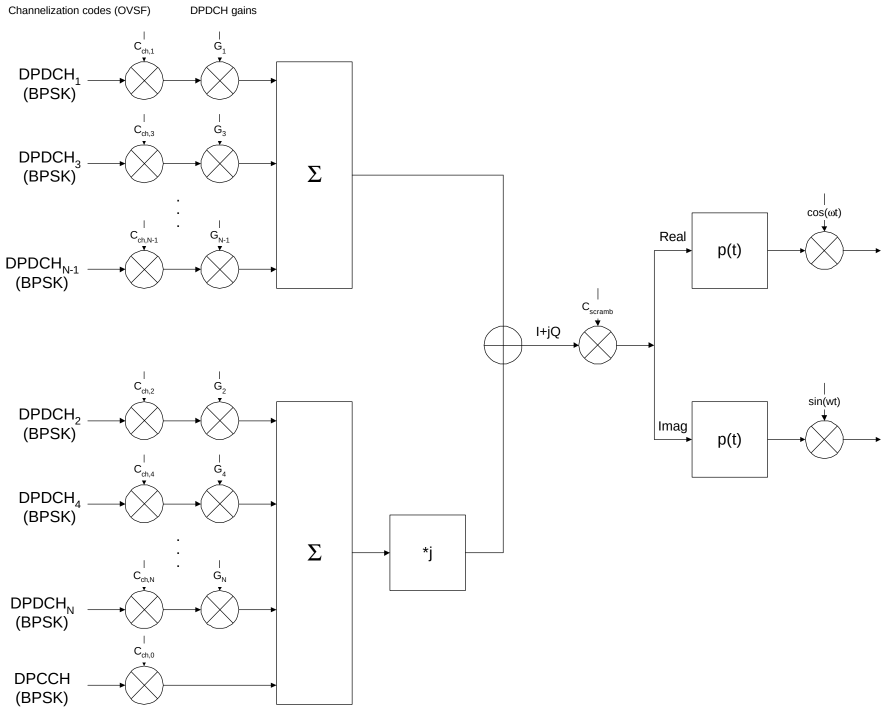

Figure 1 Spreading/modulation for uplink DPDCH/DPCCH for user services less than or equal to 1024kbps in the 5MHz band

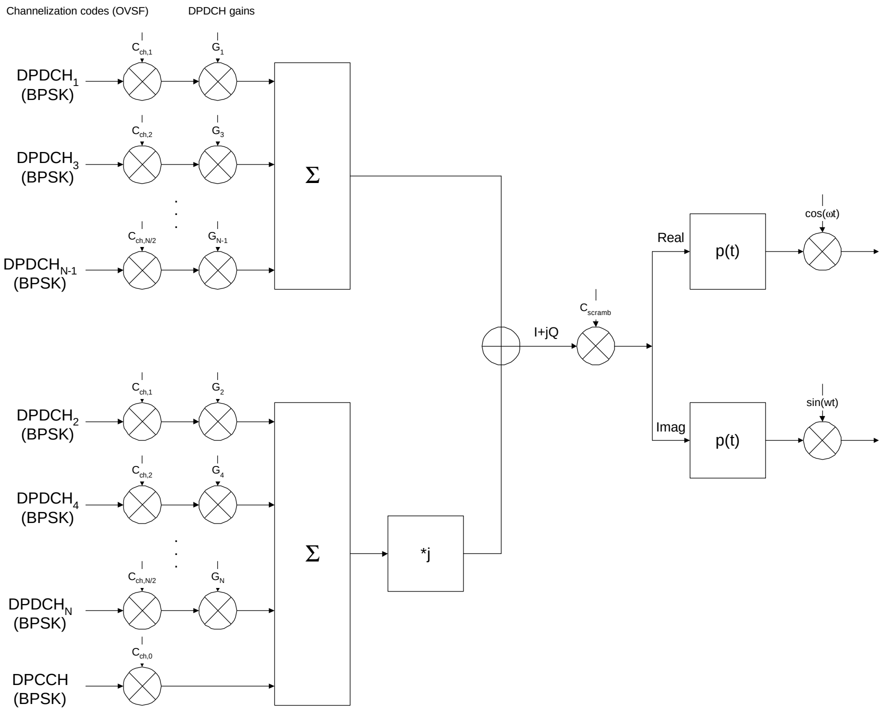

Figure 2. Spreading/modulation for uplink DPDCH/DPCCH for user services at 2048kbps in the 5MHz band

<Editor's note: discussion on Ad Hoc 10 raised the question of the 'sin' term. Mathematically it would be more elegant to have '-sin(wt)'.>

For a single uplink DPDCH transmission, only $DPDCH_1$ and DPCCH are transmitted.]

For services less than or equal to 1024kbps in the 5MHz band, the DPCCH is spread by the channelization code $C_{ch,0}$ and each $DPDCH_i$ is spread by a predefined individual channelization codes, $C_{ch,i}$ (i=1,2,…). For 2048kbps ratein the 5MHz band, the DPCCH is spread by the channelization code $C_{ch,0}$ and each pair of $DPDCH_{2i-1}$ and $DPDCH_{2i}$ is spread by a predefined individual channelization codes, $C_{ch,i}$. The data symbols of both the DPDCHs and the DPCCH are BPSK-modulated and the channelization codes are real-valued. The real-valued signals of the I- and Q-branches are then summed and treated as a complex signal. This complex signal is then scrambled by the complex-valued scrambling code, C_{scramb}.. The powers of the DPDCHs may be adjusted by gain factors, $G_{i.}$

6.2.2 PRACH

The spreading and modulation of the message part of the Random-Access burst is basically the same as for the uplink dedicated physical channels, see Figure 1, where the uplink DPDCH and uplink DPCCH are replaced by the data part and the control part respectively. The scrambling code for the message part is chosen based on the base-station-specific preamble code.

6.3 Code generation and allocation

6.3.1 Channelization codes

The channelization codes of Figure 1 are Orthogonal Variable Spreading Factor (OVSF) codes that preserve the orthogonality between a user's different physical channels. The OVSF codes can be defined using the code tree of Figure 3.

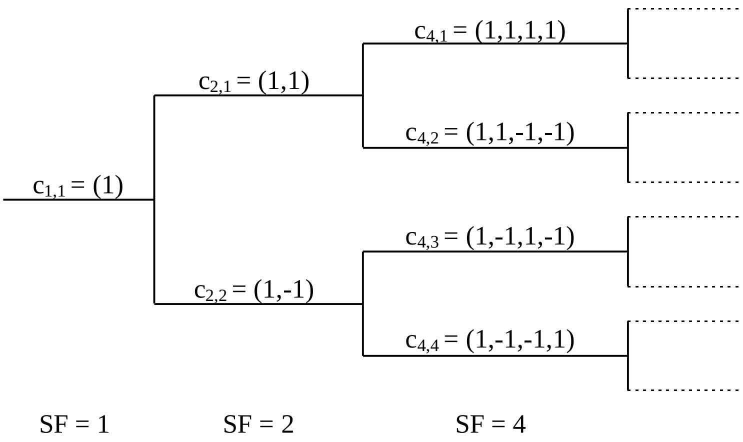

Figure 3. Code-tree for generation of Orthogonal Variable Spreading Factor (OVSF) codes.

In Figure 3, the OVSF code is described as $C_{SF,\text{code number}}$, where $SF_{d,n}$ represents the spreading factor of n^{th} DPDCH. Then the DPCCH is spread by code number 1 with a spreading factor of SF_c.

Each level in the code tree defines channelization codes of length SF, corresponding to a spreading factor of SF in Figure 3. All codes within the code tree cannot be used simultaneously by one mobile station. A code can be used by a mobile station 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 by the same mobile station. This means that the number of available channelization codes is not fixed but depends on the rate and spreading factor of each physical channel.

The generation method for the channelization code can also be explained in Figure 4.

$$C_{1,1} = 1$$

$$\begin{bmatrix} C_{2,1} \\ C_{2,2} \end{bmatrix} = \begin{bmatrix} C_{1,1} & C_{1,1} \\ C_{1,1} & \overline{C_{1,1}} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$\begin{bmatrix} C_{4,1} \\ C_{4,2} \\ C_{4,3} \\ C_{4,4} \end{bmatrix} = \begin{bmatrix} C_{2,1} & C_{2,1} \\ C_{2,1} & \overline{C_{2,1}} \\ C_{2,2} & C_{2,2} \\ C_{2,2} & \overline{C_{2,2}} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

$$: \; :$$

$$\begin{bmatrix} C_{2^{n+1},1} \\ C_{2^{n+1},2} \\ C_{2^{n+1},3} \\ C_{2^{n+1},4} \\ \vdots \\ C_{2^{n+1},2^{n+1}-1} \\ C_{2^{n+1},2^{n+1}} \end{bmatrix} = \begin{bmatrix} C_{2^n,1} & C_{2^n,1} \\ C_{2^n,1} & \overline{C_{2^n,1}} \\ C_{2^n,2} & C_{2^n,2} \\ C_{2^n,2} & \overline{C_{2^n,2}} \\ \vdots & \vdots \\ C_{2^n,2^n} & C_{2^n,2^n} \\ C_{2^n,2^n} & \overline{C_{2^n,2^n}} \end{bmatrix}$$

Figure 4. Spreading Code Generation Method

Binary code words are equivalent to the real valued sequences by the transformation ‘0’ -> ‘+1’, ‘1’ -> ‘-1’.

The spreading code cycle is the symbol cycle. Thus, for a given chip rate, the spreading code cycle depends on the symbol rate. Furthermore, the number of codes that can be used also differs according to the symbol rate. The relations between symbol rate, spreading code types, spreading code cycle and number of spreading codes is listed in .

The spreading code phase synchronises with the modulation/demodulation symbols. In other words, the head chip of the symbol is spreading code phase=0.

Symbol rate (ksps)				spreading code cycle(chip) SF	No. of Spreading codes
Chip rate= [1.024 Mcps]	4.096 Mcps	[8.192 Mcps]	[16.384 Mcps]		
[256]	1024	[2048]	4096]	4	4
[128]	512	[1024]	2048]	8	8
[64]	256	[512]	[1024]	16	16
[32]	128	[256]	[512]	32	32
[16]	64	[128]	[256]	64	64
[8]	32	[64]	[128]	128	128
-	16	[32]	[64]	256	256
-	[8]	[16]	[32]	512	512
-	-	[8]	[16]	1024	1024
			[8]	2048	2048

Table 1. Correspondence between Symbol Rate and Spreading Code Types

<Editor’s note: 1.024, 8.192 and 16.384Mcps are ffs.>

The DPCCH is spread by code number 1 in any code tree as described in Section 6.3.1. The first DPDCH is spread by code number ($SF_{d,1}$ / 4 + 1). Subsequently added DPDCHs for multi-code transmission are spread by codes in ascending order starting from code number 2 excepting the one used for the first DPDCH. However to guarantee the orthogonality between channels, any subtree below the specified node is not used for the channelization code of a DPDCH.

(Note: The case of OVSF code allocation with multiple DPDCHs with different spreading factors is for further study)

6.3.2 Scrambling codes

6.3.2.1 General

Either short or long scrambling codes should be used on the uplink. The short scrambling code is typically used in cells where the base station is equipped with an advanced receiver, such as a multi-user detector or interference canceller. With the short scrambling code the cross-correlation properties between different physical channels and users does not vary in time in the same way as when a long code is used. In cells where there is no gain in implementation complexity using the short scrambling code, the long code is used instead due to its better interference averaging properties. Both short and long scrambling codes are represented with complex-value.

[Alternatively, if the system chooses, RSTS for uplink transmission, the scrambling code is the same as the downlink scrambling code described in 7.2.2. In this case, the same scrambling code is allocated to all dedicated physical channels in the cell.]

Both short and long scrambling codes are formed as follows:

$$C_{scramb} = c_1(w_0 + jc_2'w_1)$$

where w_0 and w_1 are chip rate sequences defined as repetitions of:

$w_0 = \{1 \quad 1\}$

$w_1 = \{1 \quad -1\}$

Also, c_1 is a real chip rate code, and c_2' is a decimated version of the real chip rate code c_2. The preferred decimation factor is 2, however other decimation factors should be possible in future evolutions of 3GPP if proved desirable.

With a decimation factor N=2, c_2' is given as:

$$c_2'(2k) = c_2'(2k+1) = c_2(2k), \quad k=0,1,2\ldots.$$

The constituent codes c_1 and c_2 are formed differently for the short and long scrambling codes as described in Sections 6.3.2.2 and 6.3.2.3.

6.3.2.2 Long scrambling code

The long scrambling codes are formed as described in section 6.3.2, where c_1 and c_2 are constructed as the position wise modulo 2 sum of 40960 chip segments of two binary *m*-sequences generated by means of two generator polynomials of degree 41. Let *x*, and *y* be the two *m*-sequences respectively. The *x* sequence is constructed using the primitive (over GF(2)) polynomial $1+X^3+X^{41}$. The *y* sequence is constructed using the polynomial $1+ X^{20}+X^{41}$. The resulting sequences thus constitute segments of a set of Gold sequences.

The code, c_2, used in generating the quadrature component of the complex spreading code is a 875 μs shifted version of the code, c_1, used in generating the in phase component.

The uplink scrambling code word has a period of one radio frame of 10 ms.

Let $n_{40} \ldots n_0$ be the binary representation of the scrambling code number *n* (decimal) with n_0 being the least significant bit. The *x* sequence depends on the chosen scrambling code number *n* and is denoted x_n, in the sequel. Furthermore, let $x_n(i)$ and $y(i)$ denote the *i:*th symbol of the sequence x_n and *y*, respectively

The *m*-sequences x_n and *y* are constructed as:

Initial conditions:

$x_n(0)=n_0$, $x_n(1)= n_1$, … $=x_n(39)= n_{39}$, $x_n(40)= n_{40}$

$y(0)=y(1)= \ldots =y(39)= y(40)=1$

Recursive definition of subsequent symbols:

$x_n(i+41) =x_n(i+3) + x_n(i)$ *modulo 2, i=0,…,* 2^{41}-43,

$y(i+41) = y(i+20)+y(i)$ modulo 2, $i=0,\dots, 2^{41}-43$.

The definition of the *n*:th scrambling code word for the in phase and quadrature components follows as (the left most index correspond to the chip scrambled first in each radio frame):

$c_{1,n} = < x_n(0)+y(0), x_n(1)+y(1), \dots, x_n(N-1)+y(N-1) >$,

$c_{2,n} = < x_n(M)+y(M), x_n(M+1)+y(M+1), \dots, x_n(M+N-1) + y(M+N-1) >$,

again all sums being modulo 2 additions. (Both N and M are defined in Table 2.)

These binary code words are converted to real valued sequences by the transformation ‘0’ -> ‘+1’, ‘1’ -> ‘-1’.

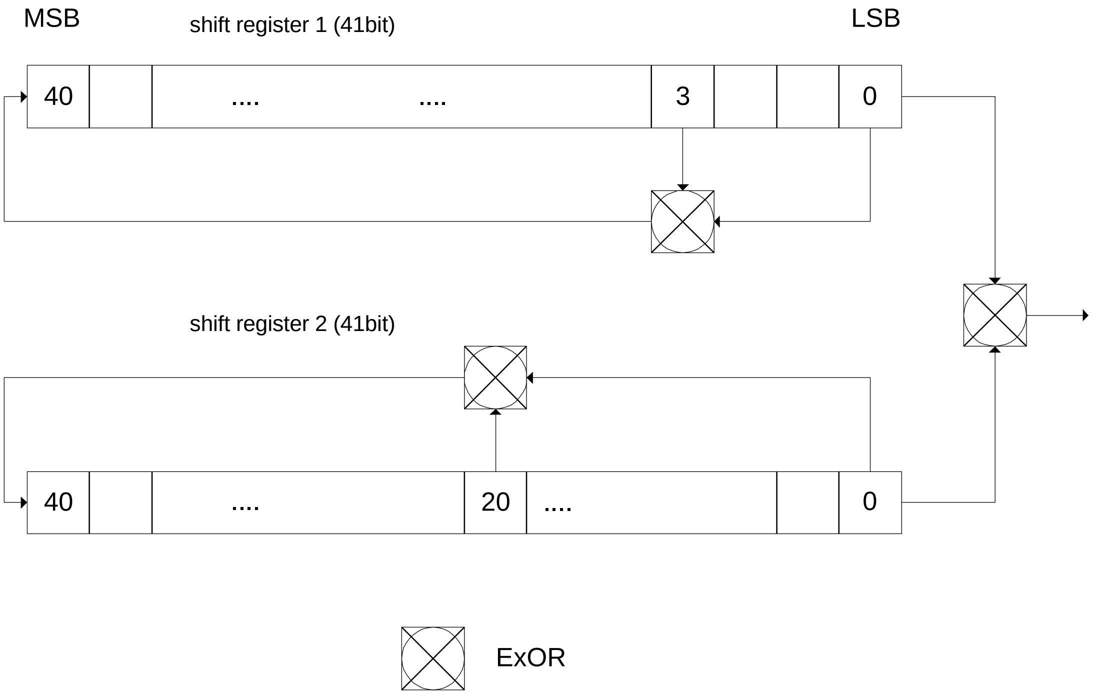

Figure 5. Configuration of uplink scrambling code generator

Chip rate (Mcps)	Period N (chips)	I/Q Offset M (chips)	Range of phase (chip)	
			(c_1)	($c_2{}'$)
[1.024	10240	896	0 – N-1	M –N+(M-1)
4.096	40960	3584		
[8.192	81920	7168		
[16.384	163840	14336		

Table 2. Correspondence between chip rate and uplink scrambling code phase range

6.3.2.3 Short scrambling code

The short scrambling codes are formed as described in Section 6.3.2.1, where, c_1 and c_2 are two different codes from the [periodic extended S(2) code family.

The reverse link short codes $S_v(n)$, n=0,1,…255, of length 256 chips are obtained by one chip periodic extension of S(2) sequences of length 255. It means that the first chip ($S_v(0)$) and the last chip ($S_v(255)$) of any reverse short scrambling code are the same.

The quaternary S(2) sequence $z_v(n)$, $0 \le v \le 16{,}777{,}216$, of length 255 is obtained by modulo 4 addition of three sequences, a quaternary sequence $a_r(n)$ and two binary sequences $b_s(n)$ and $c_t(n)$, according to the following relation:

$$z_v(n) = a_r(n) + 2 \cdot b_s(n) + 2 \cdot c_t(n) \quad (mod\ 4)\ , \quad n = 0, 1, \dots, 254.$$

The user index v determines the indexes r, s, and t of the constituent sequences in the following way:

$v = t \cdot 2^{16} + s \cdot 2^{8} + r$,

$r = 0, 1, 2, \dots, 255,$

$s = 0, 1, 2, \dots, 255,$

$t = 0, 1, 2, \dots, 255.$

The quaternary sequence $a_r(n)$ is generated by the recursive generator G_0 defined by the polynomial

$g_0(x)= x^8+x^5+3x^3+x^2+2x+1$ as

$$a_r(n)= 3.a_r(n-3) +1.a_r(n-5) + 3.a_r(n-6) +2.a_r(n-7) + 3.a_r(n-8) \ \ (mod\ 4).$$

$n = 0, 1, 2, \dots, 255.$

The binary sequence $b_s(n)$ is generated by the recursive generator G_1 defined by the polynomial

$g_1(x)= x^8+x^7+x^5+x+1$ as

$$b_s(n)= b_s(n-1)+ b_s(n-3)+ b_s(n-7)+ b_s(n-8) \ \ (mod\ 2).$$

The binary sequence $c_t(n)$ is generated by the recursive generator G_2 defined by the polynomial

$g_2(x)= x^8+x^7+x^5+x^4+1$ as

$$c_t(n)= c_t(n-1)+ c_t(n-3)+ c_t(n-4)+ c_t(n-8) \ \ (mod\ 2).$$

An implementation of the short scrambling code generator is shown in Figure 6. The initial states for the binary generators G_1 and G_2 are the two 8-bit words representing the indexes s and t in the 24-bit binary representation of the user index v, as it is shown in Figure 7.

The initial state for the quaternary generator G_0 is according to Figure 7. obtained after the transformation of 8-bit word representing the index r. This transformation is given by

$$a_r(0) = 2v(0)+1 \ \ (mod\ 4), \quad a_r(n) = 2v(n) \ \ (mod\ 4), \quad n = 1,\dots,7.$$

The complex quadriphase sequence $S_v(n)$ is obtained from quaternary sequence $z_v(n)$ by the mapping function given in Table 3.

The Re{Sv(n)} and Im{Sv(n)} of the S(2) code are the pair of two binary sequences corresponding to input binary sequences c_1 and c_2 respectively described in 6.3.2.

.

zv(n)	Sv(n)
0	+1 + j1
1	-1 + j1
2	-1 - j1
3	+1 - j1

Table 3. Mapping between $S_v(n)$ and $z_v(n)$

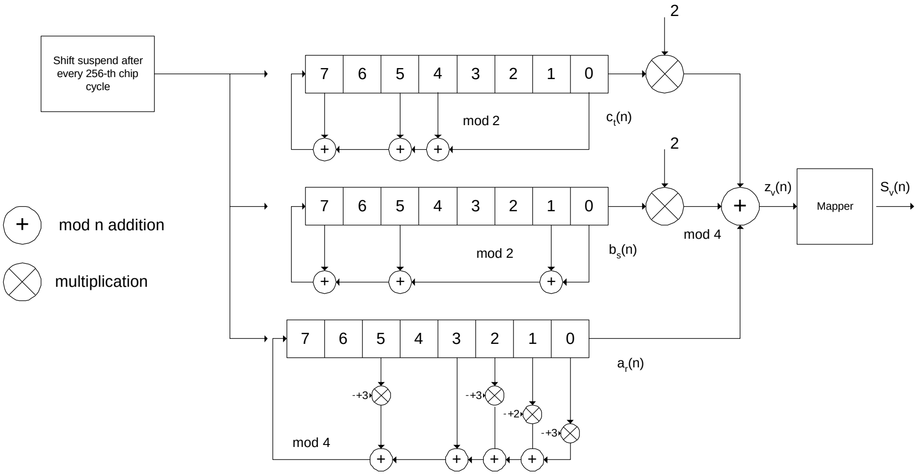

Figure 6. Uplink short scrambling code generator

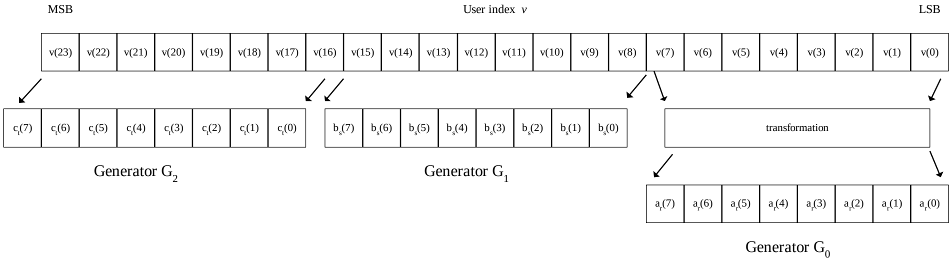
MSB
User index v
LSB
v(23) v(22) v(21) v(20) v(19) v(18) v(17) v(16) v(15) v(14) v(13) v(12) v(11) v(10) v(9) v(8) v(7) v(6) v(5) v(4) v(3) v(2) v(1) v(0)
$c_t(7)$ $c_t(6)$ $c_t(5)$ $c_t(4)$ $c_t(3)$ $c_t(2)$ $c_t(1)$ $c_t(0)$
$b_s(7)$ $b_s(6)$ $b_s(5)$ $b_s(4)$ $b_s(3)$ $b_s(2)$ $b_s(1)$ $b_s(0)$
transformation
Generator G_2
Generator G_1
$a_r(7)$ $a_r(6)$ $a_r(5)$ $a_r(4)$ $a_r(3)$ $a_r(2)$ $a_r(1)$ $a_r(0)$
Generator G_0

Figure 7. Uplink short scrambling code generator state initialisation

The short scrambling code may, in rare cases, be changed during a connection.

6.3.3 Random access codes

6.3.3.1 Preamble spreading code

The spreading code for the preamble part is cell specific and is broadcast by the base station. More than one preamble code can be used in a base station if the traffic load is high. The preamble codes must be code planned, since two neighbouring cells should not use the same preamble code.

The code used is a real-valued 256 chip Orthogonal Gold code. All 256 codes are used in the system.

The code sequences are constructed with the help of two binary *m*-sequences of length 255, *x*, and *y*, respectively. The *x* sequence is constructed using the polynomial $1+X^2+X^3+X^4+X^8$. The *y* sequence is constructed using the polynomial $1+X^3+X^5+X^6+X^8$.

Let $n_7 \dots n_0$ be the binary representation of the code number *n* (decimal) with n_0 being the least significant bit. The *x* sequence depends on the chosen code number *n* and is denoted x_n in the sequel. Furthermore, let $x_n(i)$ and $y(i)$ denote the *i:*th symbol of the sequence x_n and *y*, respectively

The *m*-sequences x_n and *y* are constructed as:

Initial conditions:

$x_n(0)=n_0$, $x_n(1)= n_1$, … $=x_n(6)= n_6$, $x_n(7)= n_7$

$y(0)=y(1)= \dots =y(6)= y(7)=1$

Recursive definition of subsequent symbols:

$x_n(i+8) =x_n(i+4) + x_n(i+3) + x_n(i+2) + x_n(i)$ modulo 2, $i=0,\dots, 246$,

$y(i+8) = y(i+6)+ y(i+5)+ y(i+3)+y(i)$ modulo 2, $i=0,\dots, 246$.

The definition of the *n*:th code word follows (the left most index correspond to the chip transmitted first in each slot):

$C_{RACH,n} = < 0, x_n(0)+y(0), x_n(1)+y(1), \dots,x_n(254)+y(254) >$,

All sums of symbols are taken modulo 2.

The preamble spreadingcode is described in Figure 8.

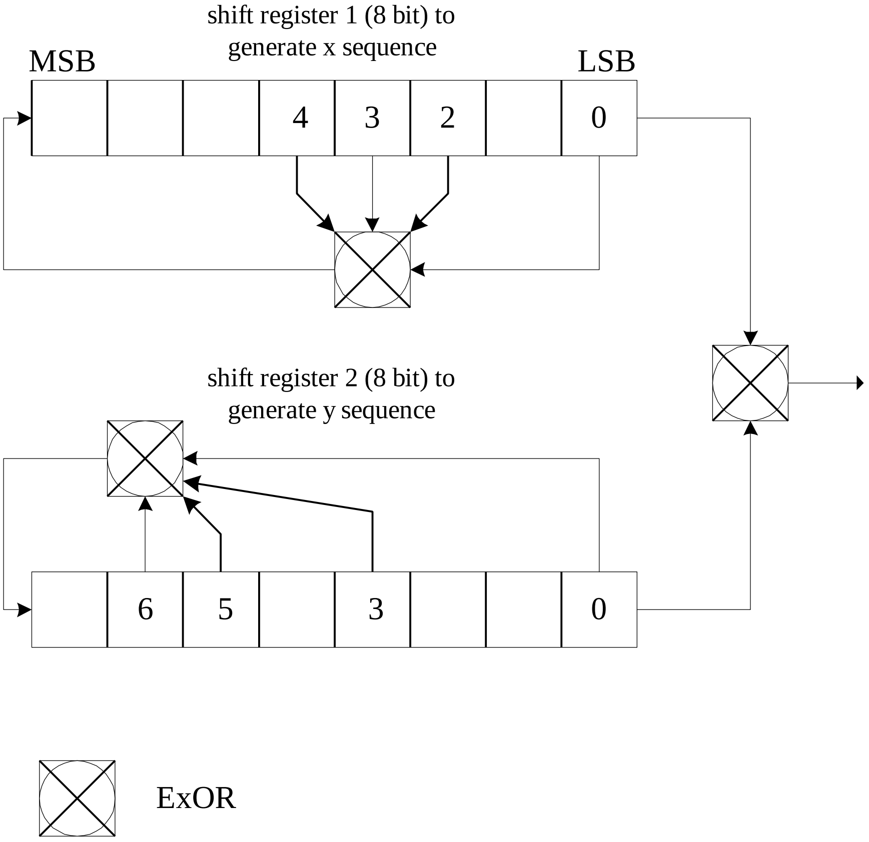
shift register 1 (8 bit) to generate x sequence
MSB
LSB
4
3
2
0
shift register 2 (8 bit) to generate y sequence
6
5
3
0
ExOR

Figure 8. Preamble spreadingcode generator

Note that the code words always start with a constant ‘0’. symbol.

Before modulation and transmission these binary code words are converted to real valued sequences by the transformation ‘0’ -> ‘+1’, ‘1’ -> ‘-1’.

6.3.3.2 Preamble signature

The preamble part carries one of 16 different orthogonal complex signatures of length 16, <P_0, P_1, ..., P_{15}>. The signatures are based on a set of Orthogonal Gold codes of length 16 and are specified in Table 4. The base station broadcasts which signatures are allowed to be used in a cell.

	Preamble symbols															
Signature	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}
1	A	A	A	-A	-A	-A	A	-A	-A	A	A	-A	A	-A	A	A
2	-A	A	-A	-A	A	A	A	-A	A	A	A	-A	-A	A	-A	A
3	A	-A	A	A	A	-A	A	A	-A	A	A	A	-A	A	-A	A
4	-A	A	-A	A	-A	-A	-A	-A	-A	A	-A	A	-A	A	A	A
5	A	-A	-A	-A	-A	A	A	-A	-A	-A	-A	A	-A	-A	-A	A
6	-A	-A	A	-A	A	-A	A	-A	A	-A	-A	A	A	A	A	A
7	-A	A	A	A	-A	-A	A	A	A	-A	-A	-A	-A	-A	-A	A
8	A	A	-A	-A	-A	-A	-A	A	A	-A	A	A	A	A	-A	A
9	A	-A	A	-A	-A	A	-A	A	A	A	-A	-A	-A	A	A	A
10	-A	A	A	-A	A	A	-A	A	-A	-A	A	A	-A	-A	A	A
11	A	A	A	A	A	A	-A	-A	A	A	-A	A	A	-A	-A	A
12	A	A	-A	A	A	A	A	A	-A	-A	-A	-A	A	A	A	A
13	A	-A	-A	A	A	-A	-A	-A	A	-A	A	-A	-A	-A	A	A
14	-A	-A	-A	A	-A	A	A	A	A	A	A	A	A	-A	A	A
15	-A	-A	-A	-A	A	-A	-A	A	-A	A	-A	-A	A	-A	-A	A
16	-A	-A	A	A	-A	A	-A	-A	-A	-A	A	-A	A	A	-A	A

Table 4. Preamble signatures. A = 1+j.

6.3.3.3 Channelization codes for the message part

The signature in the preamble specifies one of the 16 nodes in the code-tree that corresponds to channelization codes of length 16, as shown in Figure 9. The sub-tree below the specified node is used for spreading of the message part. The control (Q-branch) is spread with the channelization code of spreading factor 256 in the lowest branch of the sub-tree. The data part (I-branch) can use any of the channelization codes from spreading factor 32 to 256 in the upper-most branch of the sub-tree. However, the system may restrict the set of codes (spreading factors) actually allowed in the cell, through the use of a BCH message.

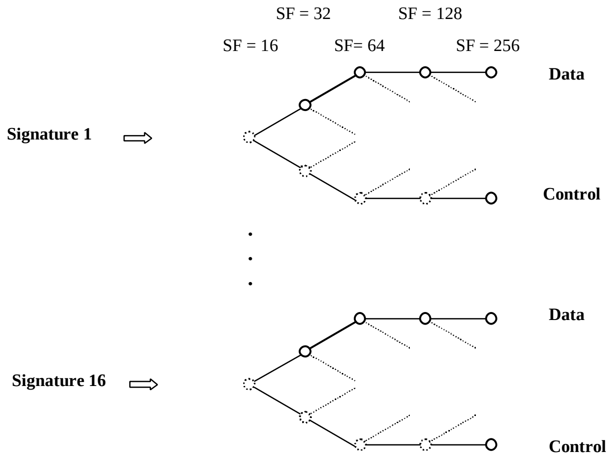

Figure 9. Channelization codes for the random access message part.

Since the control part is always spread with a known channelization code of length 256, it can be detected by the base station. The rate information field of the control part informs the base station about the spreading factor used on the data part. With knowledge of the sub-tree (obtained from the preamble signature) and the spreading factor (obtained from the rate information), the base station knows which channelization code is used for the data part.

6.3.3.4 Scrambling code for the message part

In addition to spreading, the message part is also subject to scrambling with a 10 ms complex code. The scrambling code is cell-specific and has a one-to-one correspondence to the spreading code used for the preamble part.

The scrambling codes used are from the same set of codes as is used for the other dedicated uplink channels when the long scrambling codes are used for these channels. The first 256 of the long scrambling codes are used for the random access channel. The generation of these codes is explained in Section 6.3.2.2. The mapping of these codes to provide a complex scrambling code is also the same as for the other dedicated uplink channels and is described in Section 6.3.2.

6.4 Modulation

6.4.1 Modulating chip rate

The modulating chip rate is 4.096 Mcps. This basic chip rate can be extended to [1.024,]8.192 or 16.384 Mcps.

6.4.2 Pulse shaping

The pulse-shaping filters are root-raised cosine (RRC) with roll-off α=0.22 in the frequency domain.

6.4.3 Modulation

In the uplink, the modulation of both DPCCH and DPDCH is BPSK. The modulated DPCCH is mapped to the Q-branch, while the first DPDCH is mapped to the I-branch. Subsequently added DPDCHs are mapped alternatively to the I or Q-branches.

7 Downlink spreading and modulation

7.1 Spreading

Figure 10 illustrates the spreading and modulation for the downlink DPCH and CCPCHs. Data modulation is QPSK where each pair of two bits are serial-to-parallel converted and mapped to the I and Q branch respectively. The I and Q branch are then spread to the chip rate with the same channelization code c_{ch} (real spreading) and subsequently scrambled by the same cell specific scrambling code C_{scramb} (complex scrambling).

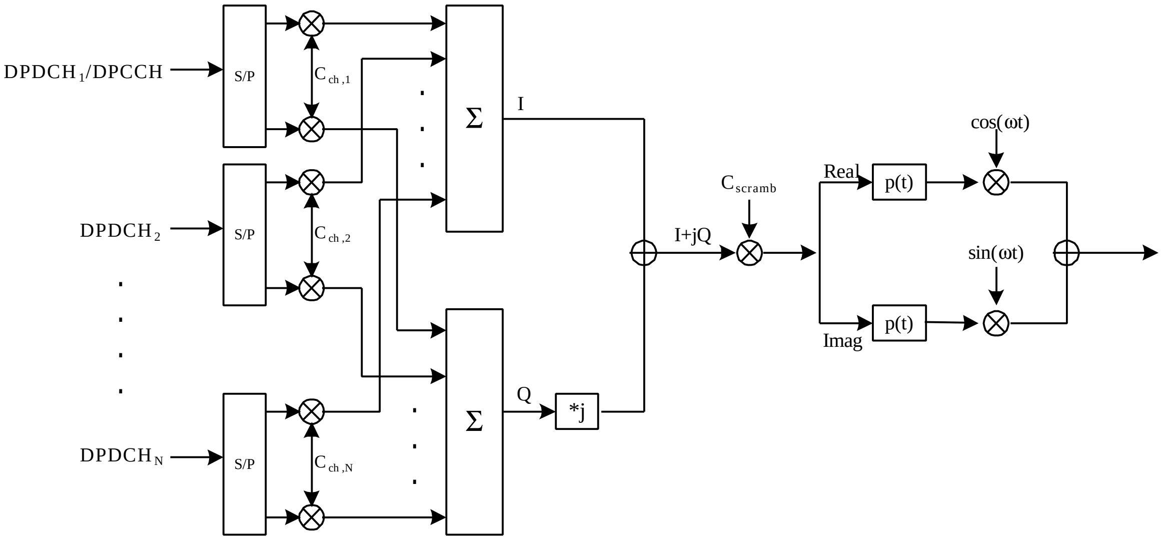

Figure 10. Spreading/modulation for downlink DPCH and CCPCHs.

The different physical channels use different channelization codes, while the scrambling code is the same for all physical channels in one cell.

The multiplexing of the [SCH]{1st and 2nd search codes} with [Primary CCPCH] {Perch channel} is illustrated in Figure 11. The figure illustrates that the [SCH]{1st and 2nd search code} are transmitted alternately. [Primary SCH and Secondary SCH] {The first search code and the second search code} are code multiplexed and transmitted simultaneously during the 1st symbol (256 chips) of each slot. In a frame. The transmission power of [SCH] {search code symbol} can be adjusted by a gain factor $G_{P\text{-}SCH}$ and $G_{S\text{-}SCH}$, respectively, independent of transmission power of P-CCPCH. The [SCH]{search code} is *non-orthogonal* to the other downlink physical channels.

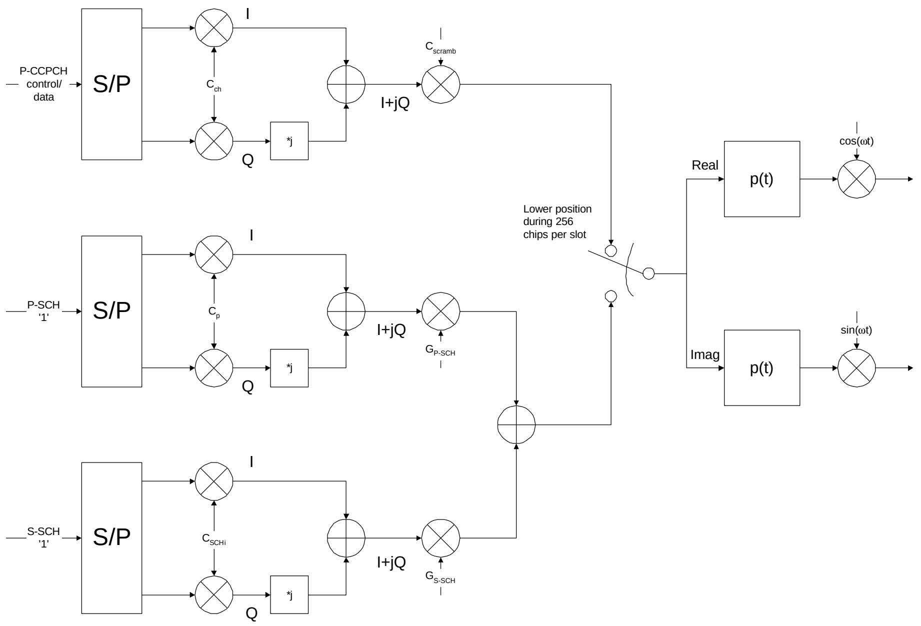

Figure 11. [Multiplexing of SCH]{Transmission of PCCPCH}

7.2 Code generation and allocation

7.2.1 Channelization codes

The channelization codes of Figure 10 are the same codes used in the uplink, namely Orthogonal Variable Spreading Factor (OVSF) codes that preserve the orthogonality between downlink channels of different rates and spreading factors. The OVSF codes are defined in Figure 3 in Section 6.3.1. The same restriction on code allocation applies as for the uplink, but for a cell and not a UEas in the uplink. Hence, in the downlink a code can be used in a cell 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 the same cell.

The channelization code for the BCH is a predefined code which is the same for all cells within the system.

The channelization code(s) used for the [Secondary Common Control Physical Channel][Common Physical Channel] is broadcast on the BCH.

< *Editor's note: There is a choice in terminology.*>

7.2.2 Scrambling code

The total number of available scrambling codes is 512, divided into 32 code groups with 16 codes in each group.

[In order to avoid code limitation in some cases, e.g. when increasing the capacity using adaptive antennas, the possibility to associate several scrambling codes with one cell (BCH area) has been identified as one solution. The exact implementation of such a scheme is still to be determined.]

<Editor's note: Use of multiple downlink scrambling codes to aid adaptive antennas are ffs.>

The scrambling code sequences are constructed by combining two real sequences into a complex sequence. Each of the two real sequences are constructed as the position wise modulo 2 sum of [40960 chip segments of] two binary *m*-sequences generated by means of two generator polynomials of degree 18. The resulting sequences thus constitute segments of a set of Gold sequences. The scrambling codes are repeated for every 10 ms radio frame. Let *x* and *y* be the two sequences respectively. The *x* sequence is constructed using the primitive (over GF(2)) polynomial $1+X^7+X^{18}$. The y sequence is constructed using the polynomial $1+X^5+X^7+ X^{10}+X^{18}$.

<Editor’s note: [ ] is due to the fact that only 4.096Mcps is a working assumptions. 1.024, 8.196, and 16.384Mcps are ffs.>

Let $n_{17} \ldots n_0$ be the binary representation of the scrambling code number *n* (decimal) with n_0 being the least significant bit. The *x* sequence depends on the chosen scrambling code number *n* and is denoted x_n, in the sequel. Furthermore, let $x_n(i)$ and $y(i)$ denote the *i*:th symbol of the sequence x_n and *y*, respectively

The *m*-sequences x_n and *y* are constructed as:

Initial conditions:

$x_n(0)=n_0$, $x_n(1)= n_1$, … $=x_n(16)= n_{16}$, $x_n(17)= n_{17}$

$y(0)=y(1)= \ldots =y(16)= y(17)=1$

Recursive definition of subsequent symbols:

$x_n(i+18) =x_n(i+7) + x_n(i)$ modulo 2, $i=0,\ldots,2^{18}-20$,

$y(i+18) = y(i+10)+y(i+7)+y(i+5)+y(i)$ modulo 2, $i=0,\ldots, 2^{18}-20$.

The n:th Gold code sequence z_n is then defined as

$z_n(i) = x_n(i) + y(i)$ modulo 2, $i=0,\ldots, 2^{18}-2$.

These binary code words are converted to real valued sequences by the transformation ‘0’ -> ‘+1’, ‘1’ -> ‘-1’.

Finally, the n:th complex scrambling code sequence C_{scramb} is defined as (the lowest index corresponding to the chip scrambled first in each radio frame): (see Table 5 for definition of N and M)

$C_{scramb}(i) = z'_n(i) + j\, z'_n(i+M)$, $i=0,1,\ldots,N-1$.
<Editor’s note: the values 3584 and 40960 are based on an assumption of a chip rate of 4.096 Mcps.>

Note that the pattern from phase 0 up to the phase of 10 msec is repeated.

The index *n* runs from 0 to 511 giving 512 distinct 40960 chip sequences.

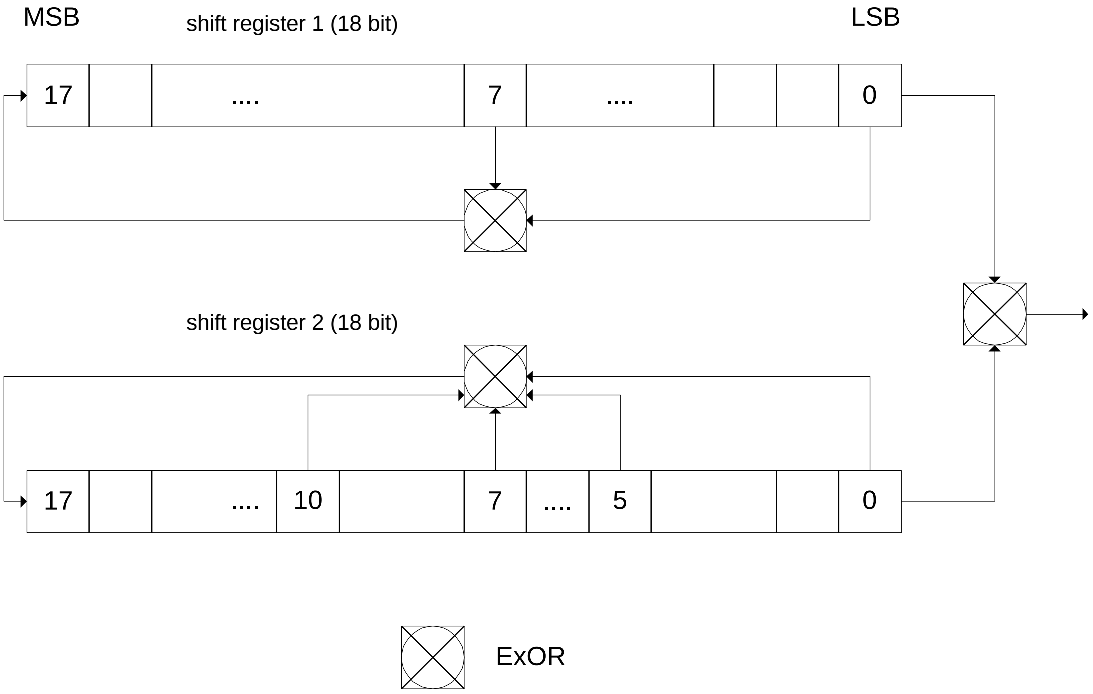

Figure 12. Configuration of downlink scrambling code generator

chip rate (Mcps)	Period	IQ Offset	Range of phase (chip)	
	N	M	for in-phase component	for quadrature component
[1.024]	[10240]	[896]	0 – N-1	M – N+M-1
4.096	40960	3584		
[8.192]	[81920]	[7168]		
[16.384]	[163840]	[14336]		

Table 5. Correspondence between chip rate and downlink scrambling code phase range

7.2.3 Synchronisation codes

7.2.3.1 Code Generation

The Primary and Secondary code words, C_p and $\{C_1,\ldots,C_{17}\}$ are constructed as the position wise addition modulo 2 of a Hadamard sequence and a fixed so called hierarchical sequence. The [Primary SCH] [1st search code] is furthermore chosen to have good aperiodic auto correlation properties.

< *Editor's note: There is a choice on the terminology.*>

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

$y(i) = x_2(i \bmod n_2) + x_1(i \operatorname{div} n_2) \text{ modulo } 2,\ i = 0 \ldots (n_1 * n_2) - 1$

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

$x_1 = x_2 = < 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0 >$

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

$$H_0 = (0)$$

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

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

The Hadamard sequence *h* depends on the chosen code number *n* and is denoted h_n in the sequel.

This code word is chosen from every 8th row of the matrix H_8. Therefore, there are 32 possible code words out of which 18 are used.

Furthermore, let $h_n(i)$ and $y(i)$ denote the *i*:th symbol of the sequence h_n and *y*, respectively.

Then h_n is equal to the row of H_8 numbered by the bit reverse of the 8 bit binary representation of *n*.

The definition of the *n*:th [SCH][search] code word follows (the left most index correspond to the chip transmitted first in each slot):

$C_{SCH,n} = < h_n(0)+y(0), h_n(1)+y(1), h_n(2)+y(2), \ldots, h_n(255)+y(255) >$,

All sums of symbols are taken modulo 2.

These binary code words are converted to real valued sequences by the transformation '0' -> '+1', '1' -> '-1'.

The [Primary SCH][1st search] and [Secondary SCH][2nd search] code words are defined in terms of $C_{SCH,n}$ and the definition of C_p and $\{C_1,\ldots,C_{17}\}$ now follows as:

$C_p = C_{SCH,0}$

and

$C_i = C_{SCH,i}$, i=1,…,17

The definitions of C_p and $\{C_1,…,C_{17}\}$ are such that a 32 point fast Hadamard transform can be utilised for detection.

<Editor's note: choice has to be made between terminology primary SCH code and 1st search code.>

7.2.3.2 Code Allocation

The 32 sequences are constructed such that their cyclic-shifts are unique, i.e., a non-zero cyclic shift less than 16 of any of the 32 sequences is not equivalent to some cyclic shift of any other of the 32 sequences. Also, a non-zero cyclic shift less than 16 of any of the sequences is not equivalent to itself with any other cyclic shift less than 16. The following sequences are used to encode the 32 different code groups each containing 16 scrambling codes (note that c_i indicates the i'th Secondary Short code [2nd search code] of the 17 codes). Note that a Secondary Short code [2nd search code] can be different from one time slot to another and that the sequence pattern can be different from one cell to another, depending on Scrambling Code Group of Scrambling Code the cell uses

Scrambling Code Groups	Slot Number															
	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12	#13	#14	#15	#16
Group1	C_1	C_1	C_2	C_{11}	C_6	C_3	C_{15}	C_7	C_8	C_8	C_7	C_{15}	C_3	C_6	C_{11}	C_2
Group2	C_1	C_2	C_9	C_3	C_{10}	C_{11}	C_{13}	C_{13}	C_{11}	C_{10}	C_3	C_9	C_2	C_1	C_{16}	C_{16}
Group 3	C_1	C_3	C_{16}	C_{12}	C_{14}	C_2	C_{11}	C_2	C_{14}	C_{12}	C_{16}	C_3	C_1	C_{13}	C_4	C_{13}
Group 4	C_1	C_4	C_6	C_4	C_1	C_{10}	C_9	C_8	C_{17}	C_{14}	C_{12}	C_{14}	C_{17}	C_8	C_9	C_{10}
Group 5	C_1	C_5	C_{13}	C_{13}	C_5	C_1	C_7	C_{14}	C_3	C_{16}	C_8	C_8	C_{16}	C_3	C_{14}	C_7
Group 6	C_1	C_6	C_3	C_5	C_9	C_9	C_5	C_3	C_6	C_1	C_4	C_2	C_{15}	C_{15}	C_2	C_4
Group 7	C_1	C_7	C_{10}	C_{14}	C_{13}	C_{17}	C_3	C_9	C_9	C_3	C_{17}	C_{13}	C_{14}	C_{10}	C_7	C_1
Group 8	C_1	C_8	C_{17}	C_6	C_{17}	C_8	C_1	C_{15}	C_{12}	C_5	C_1	C_7	C_{13}	C_5	C_{12}	C_{15}
Group 9	C_1	C_9	C_7	C_{15}	C_4	C_{16}	C_{16}	C_4	C_{15}	C_7	C_9	C_1	C_{12}	C_{17}	C_{17}	C_{12}
Group 10	C_1	C_{10}	C_{14}	C_7	C_8	C_7	C_{14}	C_{10}	C_1	C_9	C_5	C_{12}	C_{11}	C_{12}	C_5	C_9
Group 11	C_1	C_{11}	C_4	C_{16}	C_{12}	C_{15}	C_{12}	C_{16}	C_4	C_{11}	C_1	C_6	C_{10}	C_7	C_{10}	C_6
Group 12	C_1	C_{12}	C_{11}	C_8	C_{16}	C_6	C_{10}	C_5	C_7	C_{13}	C_{14}	C_{17}	C_9	C_2	C_{15}	C_3
Group 13	C_1	C_{13}	C_1	C_{17}	C_3	C_{14}	C_8	C_{11}	C_{10}	C_{15}	C_{10}	C_{11}	C_8	C_{14}	C_3	C_{17}
Group 14	C_1	C_{14}	C_8	C_9	C_7	C_5	C_6	C_{17}	C_{13}	C_{17}	C_6	C_5	C_7	C_9	C_8	C_{14}
Group 15	C_1	C_{15}	C_{15}	C_1	C_{11}	C_{13}	C_4	C_6	C_{16}	C_2	C_2	C_{16}	C_6	C_4	C_{13}	C_{11}
Group 16	C_1	C_{16}	C_5	C_{10}	C_{15}	C_4	C_2	C_{12}	C_2	C_4	C_{15}	C_{10}	C_5	C_{16}	C_1	C_8
Group 17	C_1	C_{17}	C_{12}	C_2	C_2	C_{12}	C_{17}	C_1	C_5	C_6	C_{11}	C_4	C_4	C_{11}	C_6	C_5
Group 18	C_2	C_8	C_{11}	C_{15}	C_{14}	C_1	C_4	C_{10}	C_{10}	C_4	C_1	C_{14}	C_{15}	C_{11}	C_8	C_2
Group 19	C_2	C_9	C_1	C_7	C_1	C_9	C_2	C_{16}	C_{13}	C_6	C_{14}	C8	C_{14}	C_6	C_{13}	C_{16}
Group 20	C_2	C_{10}	C_8	C_{16}	C_5	C_{17}	C_{17}	C_5	C_{16}	C_8	C_{10}	C_2	C_{13}	C_1	C_1	C_{13}
Group 21	C_2	C_{11}	C_{15}	C_8	C_9	C_8	C_{15}	C_{11}	C_2	C_{10}	C_6	C_{13}	C_{12}	C_{13}	C_6	C_{10}
Group 22	C_2	C_{12}	C_5	C_{17}	C_{13}	C_{16}	C_{13}	C_{17}	C_5	C_{12}	C_2	C_7	C_{11}	C_8	C_{11}	C_7
Group 23	C_2	C_{13}	C_{12}	C_9	C_{17}	C_7	C_{11}	C_6	C_8	C_{14}	C_{15}	C_1	C_{10}	C_3	C_{16}	C_4
Group 24	C_2	C_{14}	C_2	C_1	C_4	C_{15}	C_9	C_{12}	C_{11}	C_{16}	C_{11}	C_{12}	C_9	C_{15}	C_4	C_1
Group 25	C_2	C_{15}	C_9	C_{10}	C_8	C_6	C_7	C_1	C_{14}	C_1	C_7	C_6	C_8	C_{10}	C_9	C_{15}
Group 26	C_2	C_{16}	C_{16}	C_2	C_{12}	C_{14}	C_5	C_7	C_{17}	C_3	C_3	C_{17}	C_7	C_5	C_{14}	C_{12}
Group 27	C_2	C_{17}	C_6	C_{11}	C_{16}	C_5	C_3	C_{13}	C_3	C_5	C_{16}	C_{11}	C_6	C_{17}	C_2	C_9
Group 28	C_2	C_1	C_{13}	C_3	C_3	C_{13}	C_1	C_2	C_6	C_7	C_{12}	C_5	C_5	C_{12}	C_7	C_6

Group 29	C_2	C_2	C_3	C_{12}	C_7	C_4	C_{16}	C_8	C_9	C_9	C8	C_{16}	C_4	C_7	C_{12}	C_3
Group 30	C_2	C_3	C_{10}	C_4	C_{11}	C_{12}	C_{14}	C_{14}	C_{12}	C_{11}	C_4	C_{10}	C_3	C_2	C_{17}	C_{17}
Group 31	C_2	C_4	C_{17}	C_{13}	C_{15}	C_3	C_{12}	C_3	C_{15}	C_{13}	C_{17}	C_4	C_2	C_{14}	C_5	C_{14}
Group 32	C_2	C_5	C_7	C_5	C_2	C_{11}	C_{10}	C_9	C_1	C_{15}	C_{13}	C_{15}	C_1	C_9	C_{10}	C_{11}
[SyncBTS]	C_2	C_6	C_{14}	C_{14}	C_6	C_2	C_8	C_{15}	C_4	C_{17}	C_9	C_9	C_{17}	C_4	C_{15}	C_8

Table 9 Spreading Code allocation for Secondary SCH [Second Search] Code

7.3 Modulation

7.3.1 Modulating chip rate

The modulating chip rate is 4.096 Mcps. This basic chip rate can be extended to [1.024,]8.192 or 16.384 Mcps.

7.3.2 Pulse shaping

The pulse-shaping filters are root raised cosine (RRC) with roll-off α=0.22 in the frequency domain.

7.3.3 Modulation

QPSK modulation is used.

8 History

Document history		
draft	1999-02-12	New document merged from ETSI XX.05 and ARIB 3.2.4 sources.
0.0.1	1999-02-12	Corrected typo in table2.
0.0.2	1999-02-16	Added sec. SCH code table, option for HPSK on S(2) codes, scale on SCH.
0.0.3	1999-02-18	Reflected decision made on SCH multiplexing (see document titled 'Report from Ad Hoc #2 SCH multiplexing'.) and additional description on the use of S(2) for uplink short scrambling code.
0.1.0	1999-02-28	Raised to 0.1.0 after TSG RAN WG1#2 meeting (Yokohama).
1.0.0	1999-03-12	Raised to 1.0.0 when presented to TSG RAN.
1.0.1	1999-03-17	Raised to 1.0.1 incorporated Ad Hoc changes and errata from e-mail.
1.0.2	1999-03-23	Raised to 1.0.2 incorporated reports from Ad Hocs plus editorial matters.
1.0.3	1999-03-24	Raised to 1.0.3 incorporated actions from WG1#3 plenary..
1.1.0	1999-03-26	Raised to 1.1.0 changed as result of text proposal, Tdoc 298.

Editor for S1.13, spreading and modulation description, is:

Peter Chambers
Siemens Roke Manor Research
Email: peter.chambers@roke.co.uk

This document is written in Microsoft Word 97.