3GPP TSG RAN Meeting #19 **RP-030140**
Birmingham, United Kingdom, 11 - 14 March 2003

Title: **CRs (Rel-5) to TS 25.223**

Source: **TSG-RAN WG1**

Agenda item: **8.1.5**

TS 25.223 **(RP-030140)**

Doc-1st-	Doc-2nd-	Spec	CR	Rev	Subject	Phase	Ca	Versio	Versio	Workitem
RP-030140	(R1-030368	25.223	034	3	Miscellaneous Corrections	Rel-5	F	5.2.0	5.3.0	TEI-5

3GPP TSG-RAN1 Meeting #31
Tokyo, Japan, 18-21 February 2003

Tdoc ⌘R1-030368

CR-Form-v7

CHANGE REQUEST

⌘	**25.223**	**CR**	**034**	⌘**rev**	**3**	⌘	Current version:	**5.2.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: UICC apps⌘ [ ] ME [X] Radio Access Network [X] Core Network [ ]

Title:	⌘	Miscellaneous Corrections
Source:	⌘	TSG RAN WG1
Work item code:	⌘	TEI-5 — ***Date:*** ⌘ 06/03/2003
Category:	⌘	**F** — ***Release:*** ⌘ Rel-5

Use one of the following categories:
- ***F*** *(correction)*
- ***A*** *(corresponds to a correction in an earlier release)*
- ***B*** *(addition of feature),*
- ***C*** *(functional modification of feature)*
- ***D*** *(editorial modification)*

Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
- *2 (GSM Phase 2)*
- *R96 (Release 1996)*
- *R97 (Release 1997)*
- *R98 (Release 1998)*
- *R99 (Release 1999)*
- *Rel-4 (Release 4)*
- *Rel-5 (Release 5)*
- *Rel-6 (Release 6)*

Reason for change:	⌘	To correct the editorial mistakes
Summary of change:	⌘	Correction to some wrong references and typing mistakes
Consequences if not approved:	⌘	Confusion, misunderstanding, and unreadability of the spec

Clauses affected:	⌘	6.4, 7.2, 7.3, 8.2, AnnexB.1			
		Y	**N**		
Other specs affected:	⌘		**N**	Other core specifications	⌘
			N	Test specifications	
			N	O&M Specifications	
Other comments:	⌘				

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

$$\underline{V}_{\underline{v}} = \{1, \mathrm{j}, -1, -\mathrm{j}\} \tag{4}$$

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:

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

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 6.5~~5A.4~~.

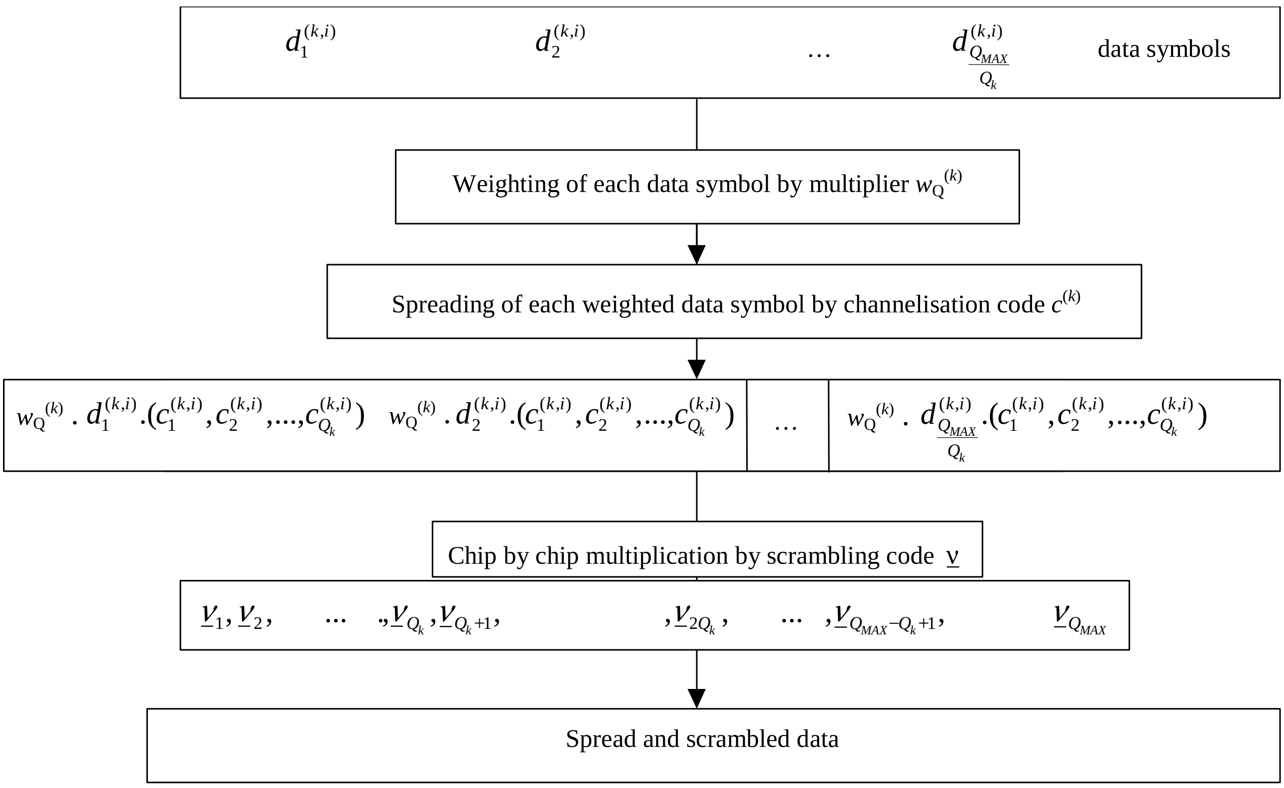

Figure 2: Spreading of data symbols

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 ~~6.2.1 to 6.2.2~~ 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.

7.3 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 6.4~~5A.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

8.2 The uplink pilot timeslot (UpPTS)

The contents in UpPTS is composed of 128 chips of a SYNC-UL sequence, cf. [AA.2 Basic SYNC-UL sequence] and 32chips of guard period (GP) .The SYNC-UL code is not scrambled.

There should be 256 different basic SYNC-UL codes (see Table AA.2) for the whole system.

For the generation of the complex valued SYNC-UL codes of length 128, the basic binary SYNC-UL codes $=\left(s_1, s_2, \ldots, s_{128}\right)$ of length 128 shown in Table AA.2 are used. The relation between the elements $\underline{\mathbf{s}}$ and $\mathbf{s}$ is given by:

$$\underline{s}_i = (j)^i \cdot s_i \qquad s_i \in \{1, -1\}, \; i = 1, \ldots, 128 \qquad (2)$$

Hence, the elements $\underline{\mathbf{s}}_i$ of the complex SYNC-UL code $\underline{\mathbf{s}}$ are alternating real and imaginary.

Annex B (informative): Generalised Hierarchical Golay Sequences

B.1 Alternative generation

The generalised hierarchical Golay sequences for the PSC described in 7.1 ~~6.1~~ 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:

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

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.
- $x_2(i) = x_4(i \bmod s + s*(i \text{ div } sn_3)) * x_3((i \text{ div } s) \bmod n_3)$, $i = 0 \dots (n_3 * n_4) - 1$.
- 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 $b_0(k) = \delta(k)$;

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

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

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

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

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 $b_4(k) = a_4(k)$, $b_6(k) = a_6(k)$.