Technical Specification Group, Radio Access Network
Meeting #3, Yokohama, 21-23 April 1999

TSGR#3(99)239

Source: **RAN WG1**

Title: **TS 1.22 V. 2.0.0Agenda Item:**

TS 1.22 V2.0.0 1999-04

Technical Report

3rd Generation Partnership Project (3GPP); Technical Specification Group (TSG) Radio Access Network (RAN); Working Group 1 (WG1); Multiplexing and channel coding (TDD);

Reference

<Workitem>

Keywords

<keyword[, keyword]>

3GPP

Postal address

Office address

Internet

http://www.3gpp.org

Contents

1 Intellectual Property Rights

<Editor's note: this section will be completed when an official format for the document is agreed>

2 Foreword

This Technical Specification has been produced by the 3rd Generation Partnership Project, Technical Specification Group Radio Access Network, Working Group 1 (3GPP TSG RAN WG1).

The contents of this TS may be subject to continuing work within the 3GPP and may change following formal TSG approval. Should the TSG modify the contents of this TS, it will be re-released with an identifying change of release date and an increase in version number as follows:

Version m.x.y
where:
- m indicates [major version number]
- x the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- y the third digit is incremented when editorial only changes have been incorporated into the specification.

3 Scope

This 3GPP Report describes multiplexing, channel coding and interleaving for UTRA Physical Layer TDD mode. Text without revision marks has been approved in the previous TSG-RAN WG1 meetings, while text with revision marks is subject to approval.

4 References

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

[1] TS S1.02 (V1.0.0): “UE capabilities”
[2] TS S1.11 (V1.0.0): “Transport channels and physical channels (FDD)”
[3] TS S1.12 (V1.0.0): “Multiplexing and channel coding (FDD)”
[4] TS S1.13 (V1.0.0): “Spreading and modulation (FDD)”
[5] TS S1.14 (V1.0.0): “Physical layer procedures (FDD)”
[6] TS S1.21 (V1.0.0): “Transport channels and physical channels (TDD)”
[7] TS S1.22 (V1.0.0): “Multiplexing and channel coding (TDD)”
[8] TS S1.23 (V1.0.0): “Spreading and modulation (TDD)”
[9] TS S1.24 (V1.0.0): “Physical layer procedures (TDD)”
[10] TS S1.31 (V1.0.0): “Measurements”
[11] TS S2.01 (V1.0.0): “Radio Interface Protocol Architecture”

5 Definitions, symbols and abbreviations

5.1 Definitions

For the purposes of the present document, the following definitions apply:
<defined term>: <definition>.

5.2 Symbols

For the purposes of the present document, the following symbols apply:
<symbol> <Explanation>

5.3 Abbreviations

ARQ Automatic Repeat on Request

BCCH Broadcast Control Channel
BER Bit Error Rate
BPSK Binary Phase Shift Keying
BS Base Station
BSS Base Station Subsystem

CA Capacity Allocation
CAA Capacity Allocation Acknowledgement
CBR Constant Bit Rate

CCCH	Common Control Channel
CD	Capacity Deallocation
CDA	Capacity Deallocation Acknowledgement
CDMA	Code Division Multiple Access
CTDMA	Code Time Division Multiple Access
CRC	Cyclic Redundancy Check
DCA	Dynamic Channel Allocation
DCCH	Dedicated Control Channel
DL	Downlink
DRX	Discontinuous Reception
DTX	Discontinuous Transmission
FACH	Forward Access Channel
FDD	Frequency Division Duplex
FDMA	Frequency Division Multiple Access
FEC	Forward Error Control
FER	Frame Error Rate
GMSK	Gaussian Minimum Shift Keying
HCS	Hierarchical Cell Structure
JD	Joint Detection
L1	Layer 1
L2	Layer 2
LLC	Logical Link Control
MA	Multiple Access
MAC	Medium Access Control
MAHO	Mobile Assisted Handover
MO	Mobile Originated
MOHO	Mobile Originated Handover
MS	Mobile Station
MT	Mobile Terminated
NRT	Non-Real Time
PC	Power Control
PCH	Paging Channel
ODMA	Opportunity Driven Multiple Access
QoS	Quality of Service
QPSK	Quaternary Phase Shift Keying
RACH	Random Access Channel
RF	Radio Frequency
RLC	Radio Link Control
RRC	Radio Resource Control
RRM	Radio Resource Management
RT	Real Time
RU	Resource Unit
SCH	Synchronization Channel
SDCCH	Stand-alone Dedicated Control Channel
SP	Switching Point
TCH	Traffic channel
TDD	Time Division Duplex

TDMA	Time Division Multiple Access
UL	Uplink
UMTS	Universal Mobile Telecommunications System
VBR	Variable Bit Rate

6 Multiplexing, channel coding and interleaving

6.1 General

Data stream from/to MAC and higher layers (Transport block / Transport block set) is encoded/decoded to offer transport services over the radio transmission link. Channel coding scheme is a combination of error detection, error correcting (including rate matching), and interleaving and transport channels mapping onto/splitting from physical channels.

In the UTRA-TDD mode, the total number of basic physical channels (a certain time slot one spreading code on a certain carrier frequency) per frame is given by the maximum number of time slots which is 16 and the maximum number of CDMA codes per time slot. This maximum number of codes is 8 in case the different codes within one time slot are allocated to different users in the uplink and is higher than 8 (e.g. 9 or 10) in the downlink or if several codes are allocated to one single user in the uplink.

6.2 Transport-channel coding/multiplexing

Figure 6-1 illustrates the overall concept of transport-channel coding and multiplexing. Data arrives to the coding/multiplexing unit in form of transport block sets, once every transmission time interval. The transmission time interval is transport-channel specific from the set {10 ms, 20 ms, 40 ms, 80 ms}.
Two or more services having different Quality of Service (QoS) requirements are multiplexed into one or more physical channels using a physical channel segmentation unit, as shown in Fig 6-1. Rate matching is used to adjust the channel symbol rates (i.e., symbol rate after physical channel segmentation) to an optimum level, where minimum QoS requirement of each service is fulfilled with the same channel symbol energy. The rate matching uses the algorithm described in section 6.2.4.
The following coding/multiplexing steps can be identified:

- Add CRC to each transport block
  ? *<Editor's note: It should still be possible to code transport channels with same QoS separately. Combining several transport blocks from one or different transport channels before coding is a study item in Ad Hoc 5. Hence, a new name could perhaps be introduced for the block saying multiplexing of transport channels with the same QoS.>*
- Possibility to multiplex transport channels with same QoS before coding.
- Channel coding.
- Rate matching
- Interleaving (two steps)
- Multiplexing of transport channels with different QoS.
- Physical channel segmentation
- Mapping to physical channels
  ?

?
?

?
?

?

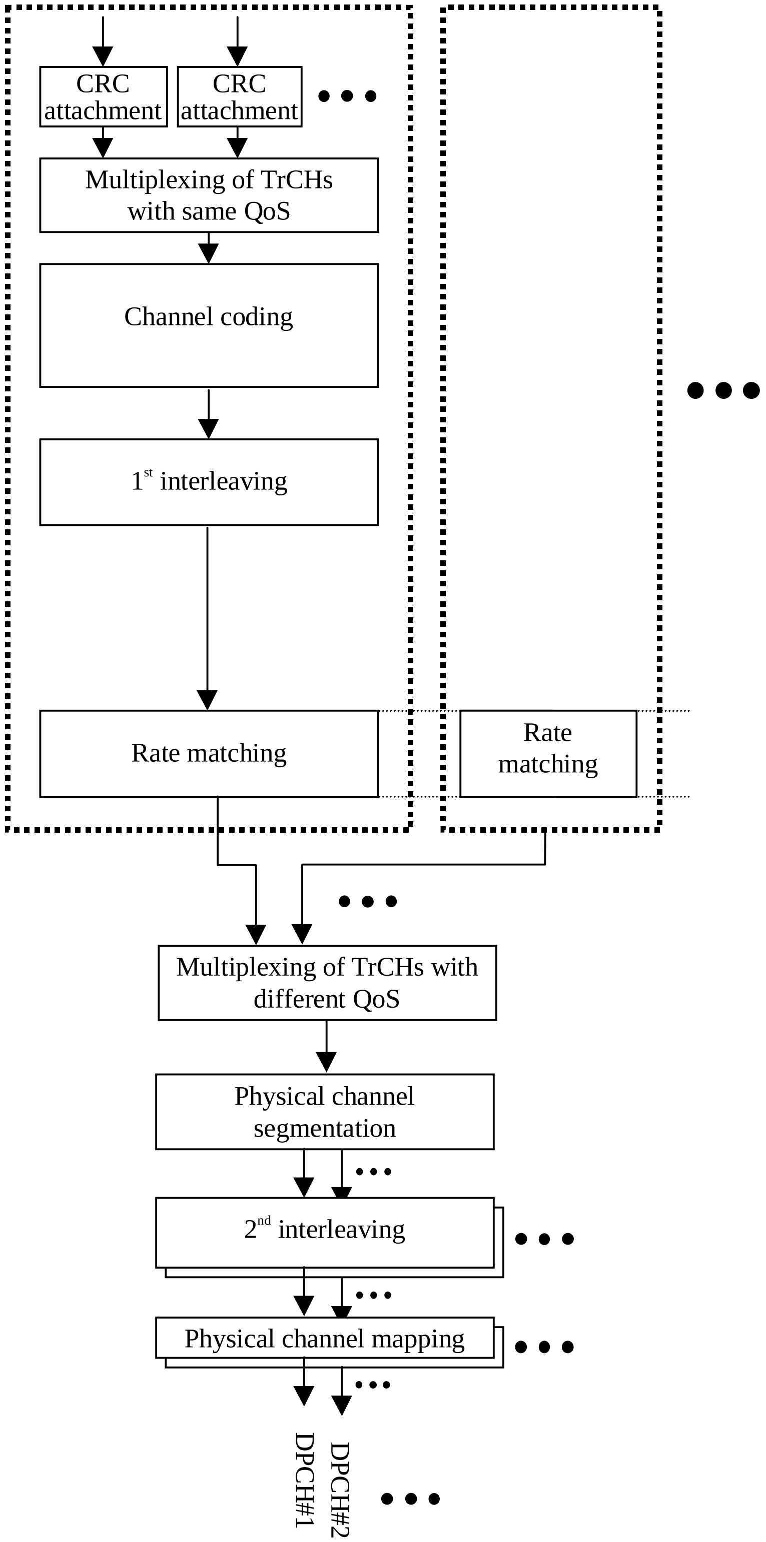


Figure 6–1. Transport channel multiplexing structure for uplink and downlink

Primarily, transport channels are multiplexed as described above, i.e. into one data stream mapped on one or several physical channels. However, an alternative way of multiplexing services is to use multiple CCTrCHs (Coded Composite

Transport Channels), which corresponds to having several parallel multiplexing chains as in Figure 6-1, resulting in several data streams, each mapped to one or several physical channels.

6.2.1 CRC calculation

Cyclic Redundancy Check is applied as an error detection scheme of transport blocks.
A 16-bit CRC code is applied to each transport block of all logical channels. Overall transport block is used to calculate the CRC for each transport block. Generator Polynomial is as follows:
$G_{CRC16}(X)= X^{16}+X^{12}+X^5+1$
The initial value of the CRC arithmetic calculator shall be "all 0". Output to the convolutional coder or Turbo coder shall be done from the most significant bit.

6.2.2 Channel coding

The following options are available for the transport-channel specific coding, see also Figure 6-2:

- Convolutional coding
- Turbo coding
- Service-specific coding, e.g. unequal error protection for some types of speech codecs.
  ?

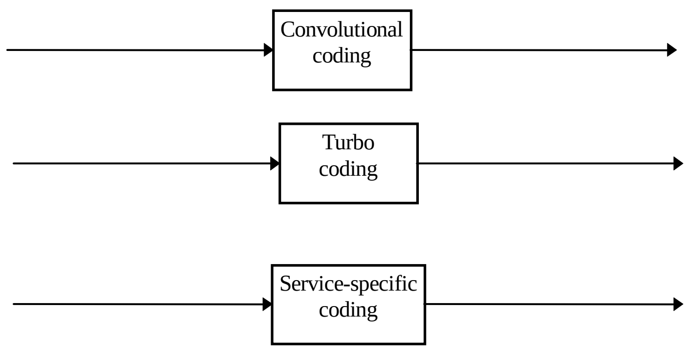


Figure 6-2. Channel coding in UTRA/TDD.

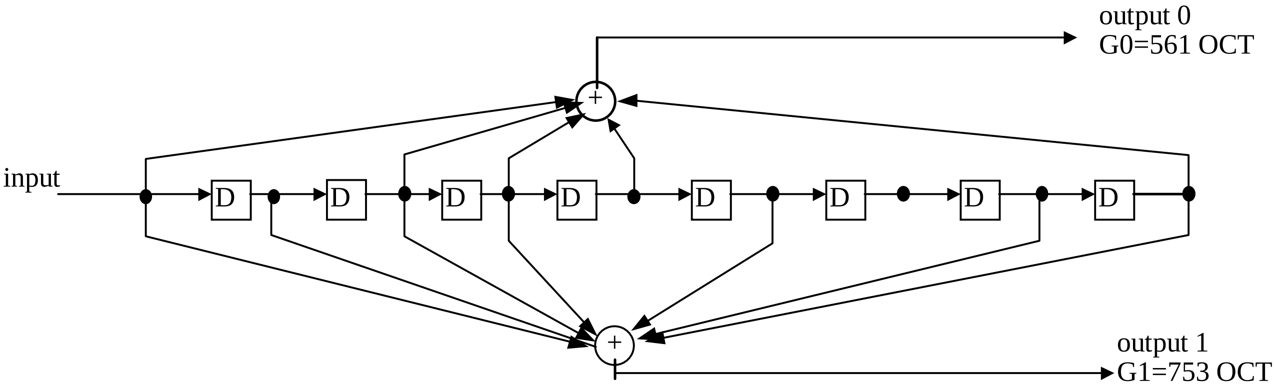
In Real Time (RT) services a FEC coding is used, instead Non Real Time (NRT) services could be well managed with a proper combination of FEC and ARQ.

Table Error! Style not defined.**-**Error! Bookmark not defined. **Error Correction Coding Parameters**

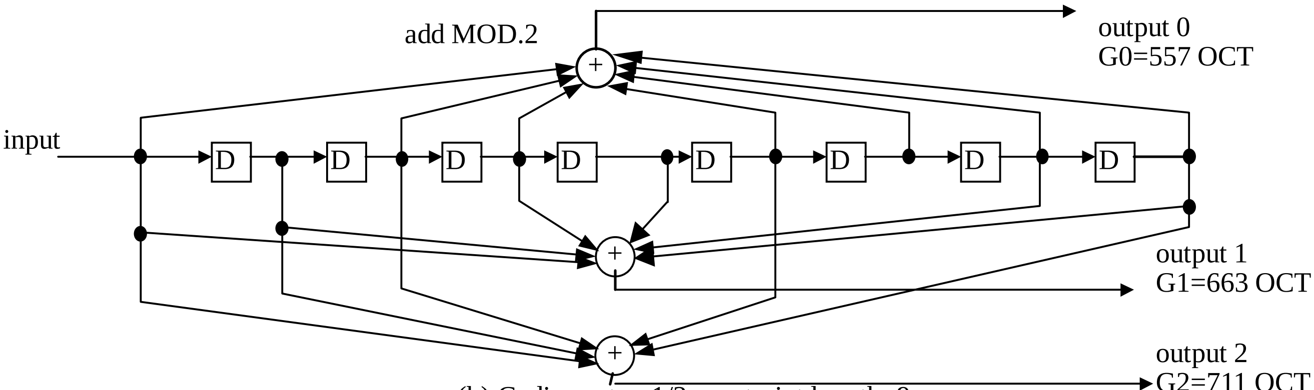
Transport channel type (Maximum coding unit size)	Coding scheme (constraint length)	Coding rate
BCH	Convolutional code (K=9)	1/2
PCH		
FACH		
RACH		1/2, [2/3, 7/8] *<Editor's note: the values in square brackets have not yet been approved.>*
DCH (less than or equal to 32kbps)		1/2 or 1/3
DCH (more than 32 kbps)	Turbo code	

6.2.2.1 Convolutional Coding

- The configuration of the convolutional coder is presented in Fig. 6-3.
- The output from the convolutional coder shall be done in the order starting from output0, output1 and output2. (When coding rate is 1/2, output is done up to output 1).
- The initial value of the shift register of the coder shall be "all 0".
- K-1 tail bits (value 0) shall be added to the end of the coding block.



(a) Coding rate =1/2 constraint length=9



(b) Coding rate =1/3 constraint length=9

Fig. 6-4 Convolutional Coder

6.2.2.2 Turbo coding

6.2.2.2.1 Turbo Coder

<Editor's note: The following is a working assumption of AdHoc 5.>

<Editor's note: It needs to be clarified from TSG SA what are the service specifications with respect to different qualities of service.>

For data services requiring quality of service between 10^{-3} and 10^{-6} BER inclusive, parallel concatenated convolutional code (PCCC) with 8-state constituent encoders is used. If data services requiring quality of service below 10^{-6} BER are to be specified, the possibility of adopting serial concatenated convolutional code (SCCC) with 4-state constituent encoders for those services should be considered for further study.
The transfer function of the 8-state constituent code for PCCC is

$$G(D)=\left[1, \frac{n(D)}{d(D)}\right]$$

where,

$$d(D)=1+D^2+D^3$$
$$n(D)=1+D+D^3.$$

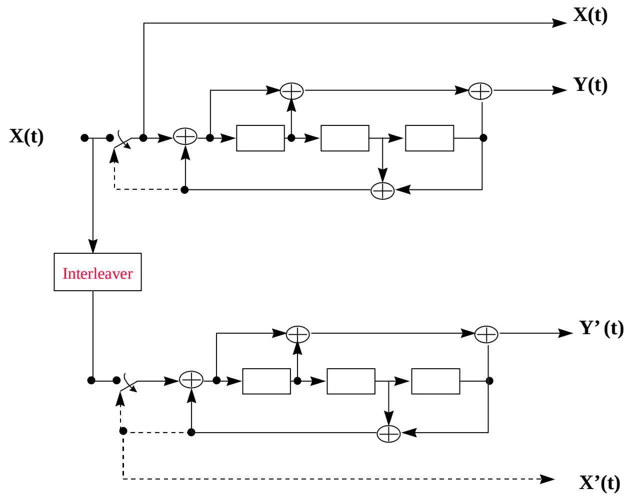


Figure 6-5. Structure of the 8 state PCCC encoder (dotted lines effective for trellis termination only)

The initial value of the shift registers of the PCCC encoder shall be all zeros.
The output of the PCCC encoder is punctured to produce coded bits corresponding to the desired code rate 1/3 or 1/2. For rate 1/3, none of the systematic or parity bits are punctured, and the output sequence is X(0), Y(0), Y'(0), X(1), Y(1), Y'(1), etc. For rate 1/2, the parity bits produced by the constituent encoders are alternately punctured to produce the output sequence X(0), Y(0), X(1), Y'(1), X(2), Y(2), X(3), Y'(3), etc.
The SCCC is a rate 1/3 SCCC, The outer code of the SCCC is a rate 2/3 obtained by puncturing a rate ½ code with generating matrix

$$G^{(o)}(Z) = (1,(1+Z^2)/(1+Z+Z^2))$$

The rate 2/3 is obtained by puncturing every other parity-check bit.
The inner code is a rate ½ systematic recursive convolutional code with the same previous generating matrix

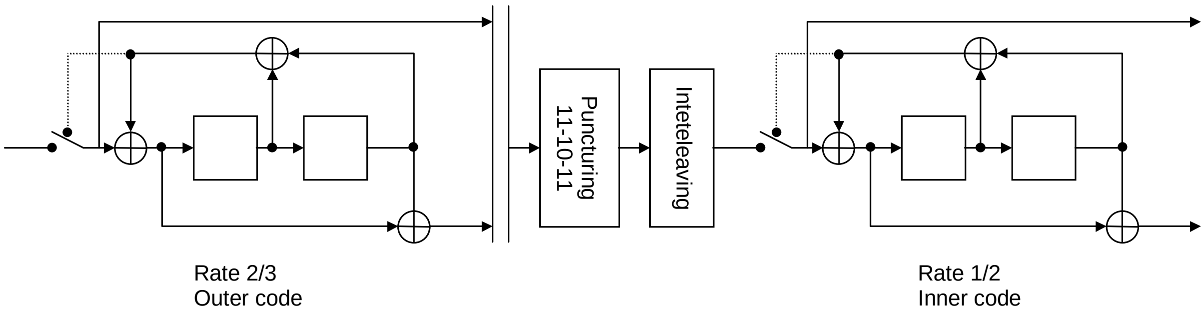


Figure 6-6. Structure of the 4 state SCCC encoder (dotted line effective for trellis termination only).

6.2.2.2.2 Trellis termination in Turbo code

The conventional method of trellis termination is used in which the tail bits are taken from the shift register feedback after all information bits are encoded. Tail bits are added after the encoding of information bits.

Trellis termination for PCCC

The first three tail bits shall be used to terminate the first constituent encoder (upper switch of Figure 6-5 in lower position) while the second constituent encoder is disabled. The last three tail bits shall be used to terminate the second constituent encoder (lower switch of Figure 6-5 in lower position) while the first constituent encoder is disabled.
The transmitted bits for trellis termination shall then be
X(t) Y(t) X(t+1) Y(t+1) X(t+2) Y(t+2) X'(t) Y'(t) X'(t+1) Y'(t+1) X'(t+2) Y'(t+2).

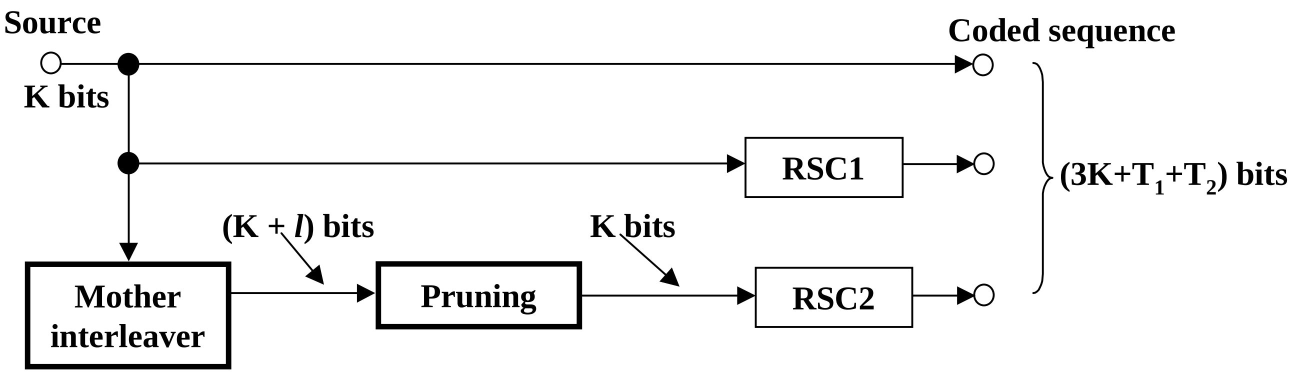
Trellis termination for SCCC

The conventional method of trellis termination is used also for SCCC in which the tail bits are taken from the shift register feedback after all bits are encoded (Figure 6-6). The tailing bits of the outer encoder are included in the interleaver. The outer code is terminated first with two additional input bits taken from the shift register feedback (dotted line), the outer code thus, after puncturing, outputs three additional bit that are feeded into the interleaver. After

that all bits have been encoded from the inner encoder (included the interleaved tail bit of the outer encoder), two additional input bits are taken from the shift register feedback of the inner encoder producing four tail bits.
Thus the total overhead due to the tailing bits is 3*2+4 =10 bits.

6.2.2.2.3 Turbo code internal interleaver

<Editor's note: The following is a working assumption of Ad Hoc 5>
Figure 6-7 depicts the overall 8 State PCCC Turbo coding scheme including Turbo code internal interleaver. The Turbo code internal interleaver consists of mother interleaver generation and pruning. For arbitrary given block length K, one mother interleaver is selected from the 224 mother interleavers set. After the mother interleaver generation, *l*-bits are pruned in order to adjust the mother interleaver to the block length K. The definition of *l* is shown in Figure 6-7.



(1) K=481 to 530 bits
$K + l = 0 \pmod{10}$;
$p-1 < (K + l) / 10 < p+2$
(2) K=320 to 8192 bits excluding 481 to 530bits
$K + l = 0 \pmod{20}$;
$p-2 < (K + l) / 20 < p+2$
where p is a prime number.

Figure 6-7. Overall 8 State PCCC Turbo Coding

6.2.2.2.3.1 Mother interleaver generation

The interleaving consists of three stages. In first stage, the input sequence is written into the rectangular matrix row by row. The second stage is intra-row permutation. The third stage is inter-row permutation. The three-stage permutations are described as follows, the input block length is assumed to be K (320 to 8192 bits). *<Editor's note: Segment length of 8192 is FFS.>*

First Stage:

(1) Determine a row number R such that
R=10 (K = 481 to 530 bits; Case-1)
R=20 (K = any other block length except 481 to 530 bits; Case-2)

(2) Determine a column number C such that
Case-1; C = 53
Csae-2;
(i) find minimum prime p such that,
$0 < (p+1)-K/R$
(ii) if $0 < p-K/R$ then go to (iii)
else $C = p+1$.
(iii) if $0 < p-1-K/R$ then $C=p-1$.
else $C = p$.

(3) The input sequence of the interleaver is written into the RxC rectangular matrix row by row.

Second Stage:

A. If C = p

(1) Select a primitive root g_0 from Table 6.2.2-2.

(2) Perform the first (0-th) intra-row permutation as:

$c(i) = [g_0 \times c(i-1)] \bmod p$, i =1,2,…C-2., c(0) = 1, and c(C-1) = 0,

where c(i) is the input bit position of i-th output after the permutation.

(3) Select the minimum prime integer set $\{p_j\}$ (j=1,2,…R-1) such that

g.c.d$\{p_j$, p-1} =1

$p_j > 6$

$p_j > p_{(j-1)}$

where g.c.d. is greatest common divider

(4) Perform the j-th (j = 1, 2, …, C-1) intra-row permutation as:

$c_j(i) = c([i \times p_j] \bmod (p-1))$, i =1,2,…, (C-2)., $c_j(0) = 1$, and $c_j(C-1) = 0$,

where $c_j(i)$ is the input bit position of i-th output after the permutation of j-th row.

B. If C = p+1

(1) Same as case A.

(2) Perform the first (0-th) intra-row permutation as:

$c(i) = [g_0 \times c(i-1)] \bmod p$, i =1,2,…, (C-2)., c(0) = 1, c(C-1) = 0, and c(C) = p,

where c(i) is the input bit position of i-th output after the permutation.

(3) Same as case A.

(4) Perform the j-th (j = 1, 2, …, C-1) intra-row permutation as:

$c_j(i) = c([i \times p_j] \bmod (p-1))$, i =1,2,…, (C-2)., $c_j(0) = 1$, $c_j(C-1) = 0$, and $c_j(C) = p$,

where $c_j(i)$ is the input bit position of i-th output after the permutation of j-th row.

C. If C = p-1

(1) Same as case A.

(2) Perform the first (0-th) intra-row permutation as:

$c(i) = [g_0 \times c(i-1)] \bmod p$, i =1,2,…,(C-2)., and c(0) = 1,

where c(i) is the input bit position of i-th output after the permutation.

(3) Same as case A.

(4) Perform the j-th (j = 1, 2, …, C-1) intra-row permutation as:

$c_j(i) = c([i \times p_j] \bmod (p-1))$, i =1,2,…, (C-2)., and $c_j(0) = 1$,

where $c_j(i)$ is the input bit position of i-th output after the permutation of j-th row.

Third Stage:

Perform the inter-row permutation based on the following patterns.

A: {19, 9, 14, 4, 0, 2, 5, 7, 12, 18, 10, 8, 13, 17, 3, 1, 16, 6, 15, 11} for R=20

B: {19, 9, 14, 4, 0, 2, 5, 7, 12, 18, 16, 13, 17, 15, 3, 1, 6, 11, 8, 10} for R=20

C: {9, 8, 7, 6, 5, 4, 3, 2, 1, 0} for R=10

The usage of these patterns is as follows:

Block length K: Pattern
320 to 480-bit: A
481 to 530-bit: C
531 to 2280-bit: A
2281 to 2480-bit: B
2481 to 3160-bit: A
3161 to 3210-bit: B
3211 to 8192-bit: A

Table 6.2.2-2. Table of prime p and associated primitive root

p	g_o	*p*	g_o	*P*	g_o	*p*	g_o	*p*	g_o	*p*	g_o	*p*	g_o	*p*	g_o
17	3	59	2	103	5	157	5	211	2	269	2	331	3	389	2
19	2	61	2	107	2	163	2	223	3	271	6	337	10	397	5
23	5	67	2	109	6	167	5	227	2	277	5	347	2	401	3
29	2	71	7	113	3	173	2	229	6	281	3	349	2	409	21
31	3	73	5	127	3	179	2	233	3	283	3	353	3		

p	g_o	*p*	g_o	*P*	g_o	*p*	g_o	*p*	g_o	*p*	g_o	*p*	g_o	*p*	g_o
37	2	79	3	131	2	181	2	239	7	293	2	359	7		
41	6	83	2	137	3	191	19	241	7	307	5	367	6		
43	3	89	3	139	2	193	5	251	6	311	17	373	2		
47	5	97	5	149	2	197	2	257	3	313	10	379	2		
53	2	101	2	151	6	199	3	263	5	317	2	383	5		

6.2.2.2.4 Encoding blocks for turbo code

<Editor’s note: Segment length of 8192 is FFS.>

Input data blocks for a turbo encoder consist of the user data and possible extra data being appended to the user data before turbo encoding. The encoding segments for a turbo encoder are defined in terms of systematic bits. The segment includes the user data, a possible error detection field (CRC), possible filler bits, and the termination. The maximum encoding segment length is 8192. The Algorithm for combining and segmentation is as follows:

Inputs:

R_{DATA} the user data rate (bits per second)

T_{DELAY} transmission time interval (seconds)

N_{EXTRA} extra data to be appended to the user data before encoding (CRC bits etc)

N_{TAIL} number of tail bits to be appended to the encoding segments (termination)

Outputs:

N_S number of segments

N_{TB} number of bits in the turbo encoder input segments

N_{FILL} number of filler (zero) bits in the last turbo encoder input segment

Do:

1. Let N_S = round_up((R_{DATA} * T_{DELAY} + N_{EXTRA}) / (8192 – N_{TAIL}))
2. Let N_{TB} = round_up ((R_{DATA} * T_{DELAY} + N_{EXTRA}) / N_S) + N_{TAIL};
3. Let N_{REM} = remainder of (R_{DATA} * T_{DELAY} + N_{EXTRA}) / N_S;
4. If N_{REM} not equal to 0 then insert N_{FILL} = (N_S – N_{REM}) zero bits to the end of the input data else N_{FILL} = 0.
5. End.

Here *round_up(x)* stands for an smallest interger number being larger or equal to *x*.

All turbo encoder input segments are of equal size and therefore the same turbo interleaver can be used for all turbo segments. A number of systematic bits over an entire channel interleaving block at output of the encoder is

N_S * (round_up((R_{DATA} * T_{DELAY} + N_{EXTRA}) / N_S) + N_{TAIL}).

The N_{FILL} filler bits are padded to the end of the last encoding segment in order to make the last segment equal size to the precedent ones. The filler bits are encoded.

6.2.2.3 Service specific coding

<Editor’s note: The application of service specific coding is still under discussion in AdHoc #4.>

In addition to standard channel coding options a service specific encoder can be used. In the simplest case it can mean that there is no channel coding at all. A more typical example is an unequal-error-protection channel code for a specific speech codec.

6.2.3 1st interleaving

<Editor's note: the following paragraph was taken from ETSI xx.10 document.>

1st interleaving is carried out on a per-transport-channel basis. The exact interleaver structure is TBD.

? *< Editors Note : In ARIB, Inter-frame MIL is carried out as described in the following paragraph.Interleaving patterns were defined as TBD in ARIB TDD text so none are depicted here. >*

he channel interleaver has the interleaving pattern obtained by using Multi-stage Interleaving Method (MIL). The iannel MIL consists of two-step interleaving process; inter-frame MIL and intra-frame MIL. In the transmitter side, the ter-frame MIL is processed in advance of the intra frame MIL as shown in Figure 6-1.

6.2.4 Rate matching

For each combination of rates of the different transport channels, a puncturing/repetition factor is assigned to each transport channel. The set of puncturing/repetition factors is determined based on following criteria:

- desired transmission quality requirements of each transport channel is fulfilled and not significantly exceeded. This means that required transmission power to meet quality requirements for all transport channels is as low as possible.
- on uplink and downlink, the total number of allocated resource units should be minimised
- the puncturing factors should not exceed a certain maximum puncturing factor, specific for each transport channel.
  ?
  ? *<Editor's note:, Maximum puncturing factor will require further discussion.>*

6.2.4.1 Rate matching algorithm

<Editor's note: the rate matching algorithms in ETSI and ARIB are similar. However, in ARIB the maximum amount of puncturing allowed is already defined.>

Let's denote:

$S_N = \{N_1, N_2, \ldots, N_L\}$ = ordered set (in ascending order from left to right) of allowed number of bits per block

N_C = number of bits per matching block

$S_0 = \{d_1, d_2, \ldots, d_{N_C}\}$ = set of N_C data bits

P = maximum amount of puncturing allowed (tentatively 0.2, for further study)

$P = 0.2$: for downlink
0.2: for uplink

<Editor's note: values for P taken from ARIB Volume 3. These may need to be reconsidered in WG1 discussion.>

The rate matching rule is as follows:

find N_i and N_{i+1} so that $N_i \le N_C < N_{i+1}$

$$if\left(\frac{N_i}{N_C} > 1 - P\right)$$

```
y = NC-Ni

e =  NC              -- initial error between current and desired puncturing ratio
                     -- this offset is flexible, e.g. e = 2Nc
m = 1          -- index of current bit
do while m <= NC
e = e – 2 * y           -- update error
if e <= 0 then          -- check if bit number m should be punctured
puncture bit m from set S0
e = e + 2*NC         -- update error
end if
m = m + 1               --  next bit
end do
else
y = Ni+1-NC

e = NC          -- initial error between current and desired puncturing ratio
                     -- this offset is flexible, e.g. e = 2Nc
m = 1                -- index of current bit
```

```
do while m <= N_C
e = e – 2 * y          -- update error
do while e <= 0        -- check if bit number m should be repeated
repeat bit m from set S_0
e = e +  2*N_C   -- update error
enddo
m = m + 1              --  next bit
end do
end if
```

6.2.5 Multiplexing of transport channels with different QoS

The coded transport channels are serially multiplexed within one radio frame. The output after the multiplexer (before the 2nd interleaver) will thus be according to Fig. 6-8. Several CCTrCHs are supported.

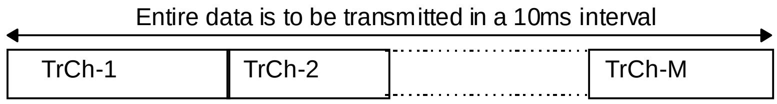


Figure 6-8. Transport channel multiplexing.

6.2.6 Physical channel segmentation

<Editor's note: Physical channel segmentation will depend upon the QoS parameters of the different transport channels in each CCTrCH. It will be necessary to specify what are the parameters expected from L2 for it.>

Data after multiplexing of transport channels with different QoS can get segmented into multiple physical channels which are transmitted in parallel during a 10ms interval.

6.2.7 2nd interleaving

The 2nd interleaving is carried out over one radio frame (10 ms) and is applied to the multiplexed set of transport channels.

6.2.8 Physical channel mapping

- If transport data is less than the number of DPCH bits in a radio frame, dynamic rate matching is applied to ensure that all resource units which belong to one user are either completely filled with data or empty.
- The transmission of the DPCH symbols shall be ON, only if there is data to transmit. If there is no data, the transmission shall be OFF.
- For transport channels not relying on TFCI for rate detection (blind rate detection), the positions of the transport channels within the frame should be fixed.
- For transport channels relying on TFCI for rate detection, the positions of the transport channels should be non-fixed.

6.2.9 Multicode Transmission

- For multi-code transmission several codes within one or several timeslots can be allocated for each user independently for uplink and downlink. Those resource units are negotiated by means of higher layers at the beginning of a transmission dependent on the individual services.

- For one user within a certain timeslot there is only one midamble code.

- Dependent on the actual amount of data to be transmitted not all resource units which are allocated to a certain link are used continuously. If no data is transmitted within a time slot during a frame, then also the midamble will be omitted.

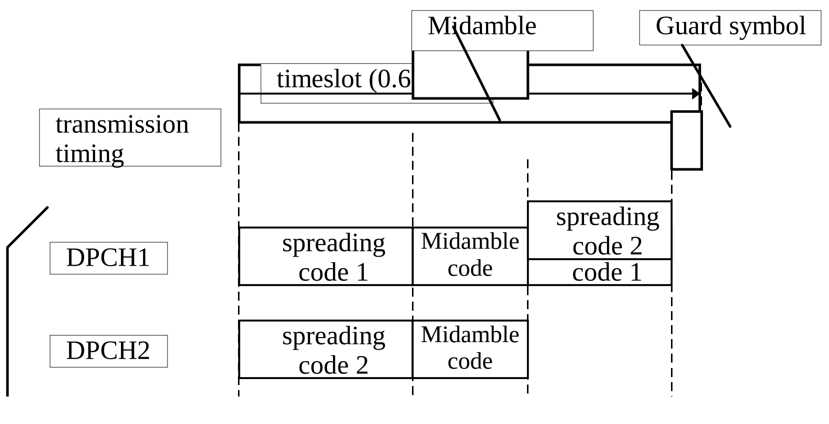


Fig. 6-9 Spreading code and midamble in multi-code transmission

6.2.10 Rate detection

Two kinds of rate detection can be employed; explicit rate detection and blind rate detection. With explicit rate detection, transmitter side transmits Transport Format Combination Indicator (TFCI), and receiver side detects transport format combination using TFCI. In case of blind rate detection, transmitter side does not transmit TFCI, and receiver side uses the possible transport format combinations as a priori information.

6.2.10.1 Blind Rate Detection

<Editor's note: The status of the support for blind rate detection with more than one possible transport format in downlink is FFS. BRD may require high implementation complexity in UE.>

Blind rate detection may be performed in the receiver by trying all possible combinations of the transport format.

6.2.10.2 Explicit Rate Detection

6.2.10.2.1 Transport Format Combination Indicator

Transport Format Combination Indicator (TFCI) informs the receiver of the number of bits in each frame of each of the services currently in use. As soon as a certain bit-rate is known, the number of code channels, the spreading factor and the puncturing/repetition rate is immediately known from the rules described in section 6.2.4.
This document therefore only explains the mapping from TFCI bits to TFCI service rate combinations.
A connection may in general include the variable-rate services S_1, S_2, …, S_K. Each service S_i has a set of possible transport format combination indicators $TF_{i,1}$,$TF_{i,2}$,…$TF_{i,Li}$:
S_1: $TF_{1,1}$, … , $TF_{1,L1}$
S_2: $TF_{2,1}$, … , $TF_{2,L2}$
…
S_K: $TF_{K,1}$, … , $TF_{K,LK}$
This gives L=L1xL2x…xLK service rate combinations, and thus it is required that L is less than or equal to 64 with the default TFCI word or 1024 with the extended TFCI word.
These service rate combinations shall be mapped to a certain service rate combination number, *m*, in the following way:

```
For j=K:-1:1,
  SRC[j]= m MOD L[j];
  m = m DIV L[j];
End;
```

From this pseudo-code, given a service rate combination number, i.e. a certain combination of TFCI bits, *m*, SRC contains the rates of each of the K services. The integer values used for *m* shall be consecutive, starting from 0.
Note that this code gives the mapping rule from *m* to SRC, i.e. the rule used in the receiving side. The mapping rule from SRC to *m*, i.e. the transmitting side rule, is [TBD].

6.3 Coding for layer 1 control

6.3.1 Coding of Transport-format-combination indicator (TFCI)

<Editor’s note: TFCI text is taken from ARIB Volume 3. In ETSI TDD, the slot structure contains a variable number of TFCI bits (See last version of xx.09). The following text should be taken as an example for 32 transmitted TFCI bits. The interleaving is taken from FDD mode, different interleaving might be necessary for TDD.>

Encoding of the TFCI bits depends on the number of them. If there are at most 6 bits of TFCI the channel encoding is done as described in section 0. Correspondingly, if the TFCI word is extended to 7-10 bits the channel encoding is done as explained in the section 0. When decoding, default TFCI words are assumed.

6.3.1.1 Default TFCI word

If the number of TFCI bits is 6 or less a biorthogonal (32, 6) block code is used. The code words of the biorthogonal (32, 6) code are from two mutually biorthogonal sets, $S_{C_6} = \{C_6(0), C_6(1), \ldots, C_6(31)\}$ and its binary complement, $\bar{S}_{C_6} = \{\bar{C}_6(0), \bar{C}_6(1), \ldots, \bar{C}_6(31)\}$. Words of set S_{C_6} are from the level 5 of the code three, which is generated, using the short code generation method defined in chapter 6.2 of S1.23. The mapping of information bits to code words is shown in the Table 6.3.1- 1.

Table 6.3.1- 1 Mapping of information bits to code words for biorthogonal (32, 6) code.

Information bits	Code word
000000	$C_6(0)$
000001	$\overline{C_6(0)}$
000010	$C_6(1)$
...	...
111101	$\overline{C_6(30)}$
111110	$C_6(31)$
111111	$\overline{C_6(31)}$

6.3.1.2 Extended TFCI word

If the number of TFCI bits is 7-10 the TFCI information field is split into two words of length 5 bits as shown in the following formula:

$n := \left\lfloor \sqrt{TFCI} \right\rfloor$; n is the largest integer being smaller than or equal to the square root of the transmitted TFCI value.

if $TFCI < n^2 + n$

then $Word1 := n$; $Word2 := TFCI - n^2$

else $Word2 := n$; $Word1 := n^2 + 2n - TFCI$

Both of the words are encoded using biorthogonal (16, 5) block code. The code words of the biorthogonal (16, 5) code are from two mutually biorthogonal sets, $S_{C_5} = \{C_5(0), C_5(1), \ldots, C_5(15)\}$ and its binary complement, $\overline{S}_{C_5} = \{\overline{C}_5(0), \overline{C}_5(1), \ldots, \overline{C}_5(15)\}$. Words of set S_{C_5} are from the level 4 of the code three, which is generated, using the short code generation method defined in chapter 3.2.4.2.2.1.1.1. The mapping of information bits to code words is shown in theTable 6.3.1-2.

Table 6.3.1- 2 Mapping of information bits to code words for biorthogonal (16, 5) code.

Information bits	Code word
00000	$C_5(0)$
00001	$\overline{C_5(0)}$
00010	$C_5(0)$
...	...
11101	$\overline{C_5(14)}$
11110	$C_5(15)$
11111	$\overline{C_5(15)}$

6.3.1.3 Interleaving of TFCI

6.3.1.3.1 Default TFCI word

As only one code word for TFCI of maximum length of 6 bits is needed no channel interleaving for the encoded bits are done. Instead, the bits of the code word are directly mapped to the slots of the radio frame as depicted in the Fig.6- 10.

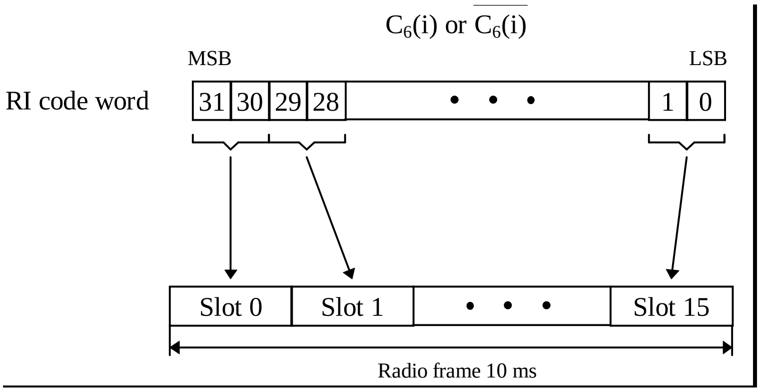


Fig.6- 10. Time multiplexing of code words of (32, 6) code to the slots of the radio frame.

6.3.1.3.2 Extended TFCI word

After channel encoding of the two 5 bit TFCI words there are two code words of length 16 bits. They are interleaved and mapped to DPCCH as shown in the Fig. 6- 11. Note that $b_{1,i}$ and $b_{2,i}$ denote the bit *i* of code word 1 and code word 2, respectively.

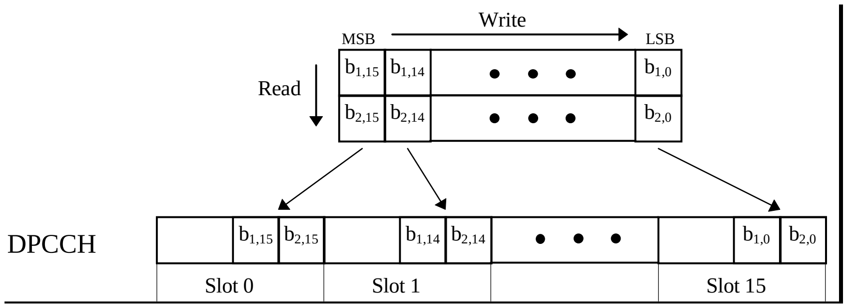


Fig. 6- 11 Interleaving of TFCI code word

<Editor's note: Chapter 6.4 on inter-frequency handover was removed, since its content is handled by the MAC layer.>

7 History

Document history		
v0.0.1	1999-02-12	Document created based on the documents Tdoc UMTS (xx.10) V0.5.0 and ARIB Volume 3 Version 1.0-1.0.
V0.0.2	1999-02-18	Document updated based on comments in the TSG-RAN WG1 e-mail reflector.

V0.1.0	1999-02-26	This version was approved at WG1#2 as the initial merge of ARIB and ETSI documents.
V0.1.1	1999-03-03	Document updated based on Adhoc decisions and technical comments made at WG1#2.
V1.0.1	1999-03-23	Document updated based on AdHoc decisions made at WG1#3.
V1.0.2	1999-03-24	Corrections made based on feedback in WG1#3 meeting.
V1.1.0	1999-03-26	This version was approved at WG1#3 meeting.
V1.1.1	1999-04-19	Document updated based on AdHoc decisions at WG1#4.
V1.1.2	1999-04-20	Document updated based on additional text proposals at WG1#4.
V2.0.0	1999-04-20	This version was approved at WG1#4 as an input to TSG RAN meeting.

The editor for 3GPP RAN WG1 S1.22 Multiplexing, channel coding and interleaving description (TDD), is:

Jussi Kahtava
Nokia-Japan
Email: jussi.kahtava @ nmp.nokia.com

This document is written in Microsoft Word 97.