vonnis

RECHTBANK 's-GRAVENHAGE

Sector civiel recht

Vonnis van 20 juni 2012

in de zaak met zaaknummer / rolnummer:

400367 / HA ZA 11-2212

van

de rechtspersoon naar buitenlands recht
SAMSUNG ELECTRONICS CO. LTD.,
gevestigd te Gyeonggi-do, Korea,
eiseres in conventie,
verweerster in reconventie,
advocaat mr. W.P. den Hertog te 's-Gravenhage,

tegen

1. de vennootschap naar buitenlands recht
APPLE INC.,
gevestigd te Cupertino, Californië, Verenigde Staten van Amerika,
2. de vennootschap naar buitenlands recht
APPLE SALES INTERNATIONAL,
gevestigd te Cork, Ierland,
3. de besloten vennootschap met beperkte aansprakelijkheid
APPLE HOLDING B.V.,
gevestigd te Bunnik,
4. de besloten vennootschap met beperkte aansprakelijkheid
APPLE BENELUX B.V.,
gevestigd te Bunnik,
5. de besloten vennootschap met beperkte aansprakelijkheid
APPLE NETHERLANDS B.V.,
gevestigd te Bunnik,
6. de besloten vennootschap met beperkte aansprakelijkheid
APPLE RETAIL NETHERLANDS B.V.,
gevestigd te Amsterdam,
gedaagden in conventie,
eiseressen in (voorwaardelijke) reconventie,
advocaat mr. D. Knottenbelt te Rotterdam.

Partijen zullen hierna Samsung en Apple genoemd worden.

1. DE PROCEDURE

1.1. Het verloop van de procedure blijkt uit:
- het tussenvonnis van 14 maart 2012 en de daarin genoemde stukken;
- de akte akte houdende overlegging reactieve productie van Apple van 16 maart 2012, met productie 83;
- de akte houdende overlegging reactieve producties van Samsung van 16 maart 2012, met producties 63-66;
- de akte na tussenvonnis van Apple van 28 maart 2012, met productie 84 (het aanvullende deskundigenrapport van professor Charles Fried);
- de akte houdende reactie op het aanvullende deskundigenrapport van Charles Fried zijdens Samsung van 10 april 2012;
- het e-mailbericht van de advocaat van Samsung van 10 april 2012, met als bijlage een tweetal brieven van het EOB uit de verleningsgeschiedenis van EP 516;
- het e-mailbericht van de advocaat van Apple van 12 april 2012, met als bijlage een drietal brieven van het EOB uit de verleningsgeschiedenis van EP 516;
- de brief van de advocaat van Apple van 12 april 2012, met een cd-rom met het verleningsdossier van EP 516.

1.2. Ter zitting van 13 april 2012 is de zaak inhoudelijk bepleit door mr. B.J. Berghuis van Woortman en mr. ir. M.W. de Koning voor Samsung en door mr. R.M. Kleemans en mr. ir. T.M. Blomme voor Apple. Zij zijn allen advocaat te Amsterdam. De pleitnotities behoren tot de stukken. De rechtbank heeft aan het begin van de zitting aangegeven dat de e-mails van 10 en 12 april alsmede de brief van 12 april, alsmede alle bijlagen daarbij, vanwege hun laattijdigheid niet als stukken worden geaccepteerd.

1.3. Vonnis is nader bepaald op heden.

2. DE FEITEN

2.1. Samsung is houdster van de volgende, onder meer voor Nederland geldende, Europese octrooien:

2.1.1. EP 1 114 528 (verder: EP 528) voor een "*Apparatus and method for controlling a demultiplexer and a multiplexer used for rate matching in a mobile communication system*", verleend op 7 juni 2006 op een aanvrage van 8 juli 2000 onder inroeping van prioriteit van drie Zuid-Koreaanse octrooiaanvragen (KR 9927407, KR 9930095 en KR 9937496) van respectievelijk 8 juli, 23 juli en 30 augustus 1999;

2.1.2. EP 1 097 516 (verder: EP 516) voor een "*Turbo interleaving apparatus and method*", verleend op 20 december 2006 op een aanvrage van 19 mei 2000 onder inroeping van prioriteit van twee Zuid-Koreaanse octrooiaanvragen (KR 9918928 en KR 9918560) van respectievelijk 19 en 21 mei 1999;

2.2. Conclusie 1 van EP 528 luidt (in de originele Engelse tekst) als volgt:

1. A uplink transmitting device for a mobile communication system, comprising:

an encoder (110) for receiving an information bit stream and for outputting three streams, a systematic symbol stream (X_k), a first parity symbol stream (Y_k), and a second parity symbol stream (Z_k) by encoding the information bit stream;
an interleaver (120) for interleaving the three streams of encoded symbols with each other by a predetermined interleaving rule and for outputting an interleaved stream;
a radio frame segmenter (130) for receiving the interleaved stream and mapping the received stream onto at least one radio frame;
a demultiplexer (141); and
a rate matcher (143, 144) adapted to puncture a part of the first and second parity symbols according to a given rate matching rule.

2.3. Conclusie 1 van EP 528 luidt (in de Nederlandse vertaling) als volgt:

1. Een uplink zendinrichting voor een mobiel communicatiesysteem, omvattende:

een codeerinrichting (110) voor het ontvangen van een informatie bitstroom en voor het afgeven van drie stromen door de informatiebitstroom te coderen,[1] een stroom van systematische symbolen (X_k), een eerste pariteitsymboolstroom (Y_k) en een tweede pariteitsymboolstroom (Z_k);
een verwevingsinrichting (120) voor het met elkaar verweven van de drie stromen van gecodeerde symbolen door een vooraf bepaalde verwevingsregel en het afgeven van een verwevingsstroom;
een radiorastersegmenteerinrichting (130) voor het ontvangen van de verwevingsstroom en het afbeelden van de ontvangen stroom op ten minste één radioraster;
een demultiplexer (141); en
een snelheidsaanpassingsinrichting (143, 144) ingericht voor het doorboren van een deel van de eerste en tweede pariteitsymbolen volgens een gegeven snelheidsaanpassingsregel.

2.4. In de beschrijving van EP 528 wordt onder meer het volgende geopenbaard:

[0001] The present invention relates generally to the rate matching of a channel encoded signal, and in particular, to an apparatus and method for controlling a demultiplexer (DEMUX) and a multiplexer (MUX) used for rate matching.
[0002] In general, radio communication systems, such as satellite, ISDN (Integrated Services Digital Network), WCDMA (Wide band-Code Division Multiple Access), UMTS (Universal Mobile Telecommunication System), and IMT (International Mobile Telecommunication)-2000 systems, channel-encode source user data with an error correction code prior to transmission, in order to increase system reliability. Typical codes used for channel encoding are convolutional codes and linear blocks code for which a single decoder is used. Lately, turbo codes, which are useful for data transmission and reception, have been suggested.
[0003] A multiple-access and multiple-channel communication system matches the number of channel encoded symbols to a given number of transmission data symbols to increase data transmission efficiency and system performance. This operation is called rate matching. Puncturing and repetition are widely performed to match the data rate of channel

[1] De woorden “door de informatiestroom te coderen” zijn kennelijk abusievelijk weggelaten in de door Samsung overgelegde Nederlandse vertaling van conclusie 1.

encoded symbols. Rate matching has recently emerged as a significant factor in UMTS for increasing data transmission efficiency in the air interface and for improving system performance.

[0004] FIG. 1 is a block diagram of an uplink transmitting device in a general mobile communication system (a UMTS system, herein).

[0005] Referring to FIG. 1, a channel encoder 110 receives frame data at predetermined TTIs (Transmission Time Intervals) which may be 10, 20, 40, or 80ms, and encodes the received frame data. And the channel encoder 110 outputs encoded symbols according to a predetermined coding rate R. The frame data size (number of information bits) is determined by a (data rate of the frame data) * (TTI). If we don't consider tail bits, the number of encoded symbols are determined by the (frame data size) * (coding rate R). A 1st interleaver 120 interleaves the output of the channel encoder 110. A radio frame segmenter 130 segments interleaved symbols received from the 1st interleaver 120 into 10-ms radio frame blocks of which size is determined by (the number of encoded symbols)/(10), wherein 10 is the radio frame length unit. A rate matcher 140 matches the data rate of a radio frame received from the radio frame segmenter 130 to a preset data rate by puncturing or repeating symbols of the radio frame. The above-described components can be provided for each service.

[0006] A MUX 150 multiplexes rate-matched radio frames from each service. A physical channel segmenter 160 segments the multiplexed radio frames received from the MUX 150 into physical channel blocks. A 2nd interleaver 170 interleaves the physical channel blocks received from the physical channel segmenter 160. A physical channel mapper 180 maps the 2nd-interleaved blocks on physical channels for transmission.

[0007] As shown in FIG. 1, the UMTS uplink transmitting device is provided with rate matchers 140. The rate matcher 140 varies in configuration depending on whether the channel encoder 110 is a convolutional encoder or a turbo encoder.

[0008] When a linear block code is used (a convolutional encoder and a single decoder are used in this case) for the channel encoder, the following requirements of rate matching should be satisfied to increase data transmission efficiency and system performance in a multiple-access/multiple-channel scheme.

1. An input symbol sequence is punctured/repeated in a predetermined periodic pattern.
2. The number of punctured symbols is minimized whereas the number of repeated symbols is maximized.
3. A uniform puncturing/repeating pattern is used to puncture/repeat encoded symbols uniformly.

[0009] The above requirements are set on the assumption that the error sensitivity of a code symbol at any position in one frame output from a convolutional encoder is similar. Although some favorable results can be produced in the above requirement, a rate matching scheme different from the convolutional encoder should be employed when using a turbo encoder because of the different error sensitivities of symbols at different positions in one frame.

[0010] When a turbo encoder is used, it is preferred that the systematic information part of the encoded symbols is not punctured since the turbo encoder is a systematic encoder. Due to the two component encoder structure of the turbo encoder, the minimum free distance of the output code is maximized when the free distance of each of the two component codes is maximized. To do so, the output symbols of the two component encoders should be punctured equally to thereby achieve optimal performance.

[0011] "Proposal for rate matching for Turbo Codes", TSGR1 #4 (99) 467, May 12, 1999, pages 1 to 5, proposes a rate matching for Turbo Codes. In this proposal, puncturing for Turbo Codes is discussed for UTRA uplink and downlink, wherein a solution is described which allows to perform optimal puncturing on Turbo Codes in UTRA uplink: In order not

to puncture systematic code symbols, it is proposed to use separate rate matching for systematic bits and parity bits.

[0012] As described above, a distinction should be made between the information symbols and the parity symbols in the encoded symbols when a turbo encoder is used, to achieve optimal rate matching. Processing, such as channel interleaving, can be interposed between the turbo encoder and a rate matcher. Nevertheless, the distinction between information symbols and parity symbols should be preserved. However, this is impossible because all of the channel encoded symbols are randomly mixed after channel interleaving.

(…)

[0018] To achieve the above object and other aspects, there is provided a transmitting device in a mobile communication system. In the preferred embodiments of the transmitting device, an encoder receives an information bit stream in a frame as long as an integer multiple of a predetermined size and generates an information symbol and first and second parity symbols by encoding each information bit. An interleaver sequentially arranges information symbols and the first and second parity symbols corresponding to each of the information symbols row by row in an array having number of rows and number of columns. The number of rows and the number of columns in the array are both integers, reorders the columns according to a predetermined rule, reading the symbols down by column from left to right, and outputs a plurality of radio frames in a stream, each radio frame having a size determined by L/(TTI/10ms), where L is number of coded symbols. A demultiplexer demultiplexes each of the radio frames received from the interleaver to the information symbols, the first parity symbols, and the second parity symbols of the radio frame. Rate matchers bypass the information symbols and puncture or repeat the first and second parity symbols for rate matching.

(…)

[0021] For rate matching, the UMTS uplink transmitting device of FIG. 1 has rate matcher 140 that varies in structure depending on whether channel encoder 110 is a convolutional encoder or a turbo encoder, as stated before. When a turbo encoder is used as the channel encoder 110 according to the preferred embodiments of the present invention, the rate matcher 140 is so constituted as to include a DEMUX 141, component rate matchers 142, 143, and 144, and a MUX 145, as shown in FIG. 2. The DEMUX 141 separates the output symbols of the radio frame segmenter 130 into information symbols and parity symbols and switches them to the corresponding component rate matchers 142, 143, and 144. The MUX 145 multiplexes symbols received from the component rate matchers 142, 143, and 144 and feeds the multiplexed symbols to the MUX 150 of FIG. 1.

(…)

[0024] In the embodiment of the present invention shown in Fig. 2, the DEMUX 141 and MUX 145 are synchronized with each other such that the DEMUX 141 and MUX 145 switch to the same rate matcher block (i.e., if DEMUX 141 switches to rate matcher 142 to input a symbol into the DEMUX 141, then MUX also switches to the rate matcher 142 after the input symbol has been rate matched to receive the rate matched symbol.).

[0025] The turbo code used in turbo encoder 110 of FIG. 2 is a systematic code and, thusly, can be separated into a systematic information symbol X_k and parity symbol Y_k and Z_k. For turbo encoder 110, code rate R = 1/3. Hereinafter, the systematic information symbol will be labeled with x and the first parity symbols with y and second parity symbols with z. When R = 1/3, the relationship between the input and output of the turbo encoder 110 is shown in FIG. 3.

(…)

[0047] In the rate matching structure shown in FIG. 2, rate matching is implemented separately for each component rate machers. The first, second, and third component rate matchers 142, 143, and 144 subject an information symbol x, a first parity symbol y, and a second parity symbol z, respectively, to rate matching. According to a given input and output sizes, each rate matcher performs puncturing/_repetition on a predetermined number of symbols. This rate matching structure is built on the assumption that the DEMUX 141

outputs x, y, z, separately. Hence, the DEMUX 141 should be able to separate a radio frame received from the radio frame segmenter 130 into symbol x, y, z in a certain order.
(…)
[0083] Referring to FIG. 2 again, the MUX 145 multiplexes three streams received from the component rate matchers 142, 143, and 144 to one stream, to thereby generate a rate-matched radio frame with the same symbol pattern as before rate matching. Because this MUX 145 is the counterpart of the DEMUX 141, it switches according to the same switching patterns.

2.5. Bij EP 528 behoren onder meer de volgende figuren:

FIG. 1

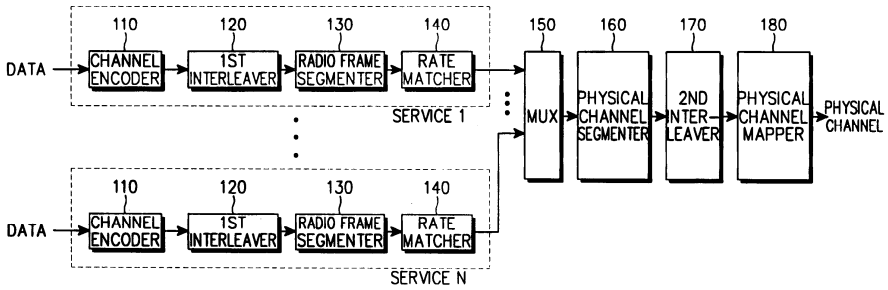
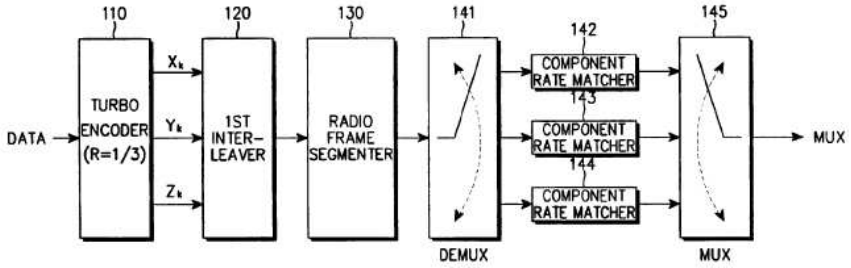


FIG. 2



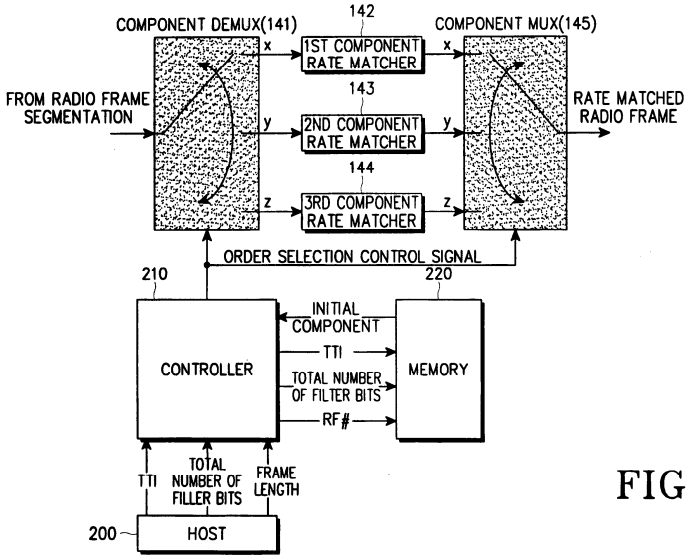

COMPONENT DEMUX(141)
142
COMPONENT MUX(145)
1ST COMPONENT RATE MATCHER
FROM RADIO FRAME SEGMENTATION
143
2ND COMPONENT RATE MATCHER
RATE MATCHED RADIO FRAME
144
3RD COMPONENT RATE MATCHER
x
y
z
ORDER SELECTION CONTROL SIGNAL
210
220
CONTROLLER
INITIAL COMPONENT
TTI
TOTAL NUMBER OF FILTER BITS
RF#
MEMORY
TTI
TOTAL NUMBER OF FILLER BITS
FRAME LENGTH
200
HOST

FIG. 13

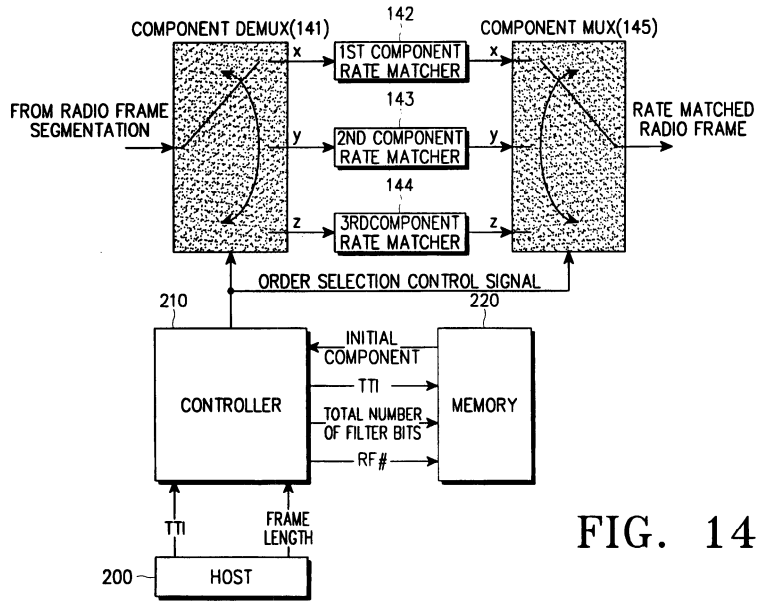

COMPONENT DEMUX(141)
142
COMPONENT MUX(145)
1ST COMPONENT RATE MATCHER
FROM RADIO FRAME SEGMENTATION
143
2ND COMPONENT RATE MATCHER
RATE MATCHED RADIO FRAME
144
3RDCOMPONENT RATE MATCHER
x
y
z
ORDER SELECTION CONTROL SIGNAL
210
220
CONTROLLER
INITIAL COMPONENT
TTI
TOTAL NUMBER OF FILTER BITS
RF#
MEMORY
TTI
FRAME LENGTH
200
HOST

FIG. 14

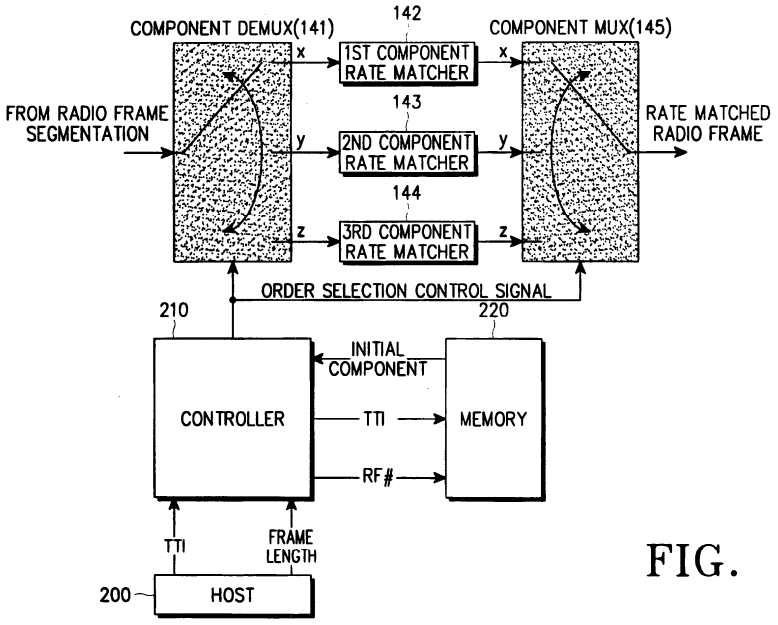


FIG. 15

2.6. De conclusies van EP 516 luiden (in de originele Engelse tekst) als volgt:

1. A turbo encoder comprising:

a first encoder (111) arranged for encoding a frame of K input information bits to generate first coded symbols;

an interleaver (112) arranged for sequentially writing the K input information bits into a R × C rectangular matrix row by row starting in the first column of the first row, intra row permuting the positions of the information bits in the R × C rectangular matrix in each row according to a given interleaving rule, wherein said intra row permuting leaves the positions of the bits in the last column of said matrix unchanged subsequent to said permuting, exchanging the position of the information bit in the last column of the last row with a position within the last row which precedes the last column, performing inter-row permutations of the R × C rectangular matrix, and reading out the information bits from the permuted R × C rectangular matrix column by column starting in the first row of the first column; and

a second encoder (113) arranged for encoding the read out information bits to generate second coded symbols,

wherein the R × C rectangular matrix has R rows and C columns, K = R × C and specifies the number of input information bits in the frame, and K > R > 1.

2. The turbo encoder as claimed in claim 1, wherein the turbo encoder is further arranged to store the input information bits in a memory, to perform the interleaving of the information bits in the R x C rectangular matrix based on generated read addresses corresponding to the permuted R x C rectangular matrix, and to output the information bits from the memory by means of the generated read addresses.

3. The turbo encoder as claimed in claim 1, wherein the turbo encoder is further arranged to exchange a position of an information bit in the last column of the last row with a position of an information bit in the first column of the last row.

2.7. De conclusies van EP 516 luiden (in de Nederlandse vertaling) als volgt:

1. Turbocodeerorgaan, omvattende:

> een eerste codeerorgaan (111) dat is ingericht voor het coderen van een frame van K inkomende informatiebits voor het genereren van eerste gecodeerde symbolen;
> een interleaver (112) die is ingericht voor het in opeenvolging schrijven van de K inkomende informatiebits in een R × C rechthoekige matrix, rij voor rij, beginnend in de eerste kolom van de eerste rij, het intra-rij permuteren van de posities van de informatiebits in de R x C rechthoekige matrix in elke rij volgens een gegeven interleavingregel,waarbij genoemd intra-rij permuteren de posities van de bits in de laatste kolom van genoemde matrix ongewijzigd laat, na genoemd permuteren het onderling verwisselen van de positie van de informatiebit in de laatste kolom van de laatste rij en een positie binnen de laatste rij die voorafgaat aan de laatste kolom, het uitvoeren van inter-rij permutaties van de R x C rechthoekige matrix, en het kolom voor kolom uitlezen van de informatiebits uit de gepermuteerde R x C rechthoekige matrix, beginnend in de eerste rij van de eerste kolom; en
> een tweede codeerorgaan (113) dat is ingericht voor het coderen van de uitgelezen informatiebits voor het genereren van tweede gecodeerde symbolen,

waarbij de R × C rechthoekige matrix R rijen en C kolommen heeft, K = R × C en het aantal inkomende informatiebits in het frame specificeert, en K > R > 1.

2. Turbocodeerorgaan volgens conclusie 1, waarbij het turbocodeerorgaan verder is ingericht voor het opslaan van de inkomende informatiebits in een geheugen, voor het uitvoeren van het interleaven van de informatiebits in de R × C rechthoekige matrix op basis van gegenereerde leesadressen die overeenkomen met de gepermuteerde R × C rechthoekige matrix, en voor het verschaffen van de informatiebits uit het geheugen door middel van de gegenereerde leesadressen.

3. Turbocodeerorgaan volgens conclusie 1, waarbij het turbocodeerorgaan verder is ingericht voor het onderling verwisselen van een positie van een informatiebit in de laatste kolom van de laatste rij en een positie van een informatiebit in de eerste kolom van de laatste rij.

2.8. In de beschrijving van EP 516 wordt onder meer het volgende geopenbaard:

[0001] The present invention relates generally to a turbo encoder used for radio communication systems. (including satellite, ISDN, digital cellular, W-CDMA, and IMT-2000 systems), and in particular, to an internal interleaver of a turbo encoder.
[0002] In general, an interleaver used for a turbo encoder randomizes an address of input information word and improves a distance property of a codeword. In particular, it has been decided that a turbo code will be used in a supplemental channel (or data transmission channel) of IMT-2000 (or CDMA-2000) and IS-95C air interfaces and in a data channel of UMTS (Universal Mobile Telecommunication System) proposed by ETSI (European Telecommunication Standards Institute). Thus, a method for embodying an interleaver for this purpose is required. In addition, the invention relates to an error correction code which greatly affects performance improvement of the existing and future digital communication systems.

[0003] For an existing internal interleaver for a turbo encoder (hereinafter, referred to as a turbo interleaver), there have been proposed various interleavers such as PN (Pseudo Noise) random interleaver, random interleaver, block interleaver, non-linear interleaver, and S-random interleaver. However, so far, such interleavers are mere algorithms designed to improve their performances in terms of scientific researches rather than implementation. Therefore, when implementing an actual system, the hardware implementation complexity must be taken into consideration. A description will now be made of properties and problems associated with the conventional interleaver for the turbo encoder.
[0004] Performance of the turbo encoder is dependent upon its internal interleaver. In general, an increase in the input frame size (i.e., the number of information bits included in one frame) enhances the effectiveness of the turbo encoder. However, an increase in interleaver size causes a geometric increase in calculations. Therefore, in general, it is not possible to implement the interleaver for the large frame size.
[0005] Therefore, in general, the interleavers are implemented by determining conditions satisfying several given criteria. The criteria are as follows:
[0006] Distance Property : The distance between adjacent codeword symbols should be maintained to a certain extent. This has the same function as a codeword distance property of the convolutional code, and as a criterion indicating this, a minimum free distance is used which is a value of a codeword path or a codeword sequence with the minimum Hamming weight out of the code symbol sequences (or codeword paths) output on the trellis. In general, it is preferable that the interleaver should be designed to have the longer free distance, if possible.
[0007] Random Property : A correlation factor between output word symbols after interleaving should be much lower than a correlation factor between original input word symbols before interleaving. That is, randomization between the output word symbols should be completely performed. This makes a direct effect on the quality of extrinsic information generated in continuous decoding.
[0008] Although the above criteria are applicable to a general turbo interleaver, it is difficult to clearly analyze the properties when the interleaver increases in size.
[0009] In addition, another problem occurring when designing the turbo interleaver is that the minimum free distance of the turbo code varies according to the type of the input codeword. That is, when the input information word has a specific sequence pattern defined as a critical information sequence pattern (CISP), the free distance of the output code symbols generated from the turbo encoder has a very small value. If the input information word has a Hamming weight 2, the CISP occurs when the input information word has two information bits of ’1’ and can also occur when the input information word has 3 or more information bits of ’1’. However, in most cases, when the input information word has 2 information bits of ’1’, the minimum free distance is formed and most error events occur in this condition. Therefore, when designing the turbo interleaver, an analysis is generally made on the case where the input information word has the Hamming weight 2. A reason that the CISP exists is because the turbo encoder generally uses RSC (Recursive Systematic Convolutional Codes) encoders for the component encoders shown in FIG. 1 (described further below). To improve performance of the turbo encoder, a primitive polynomial should be used for a feedback polynomial (gf(x) of FIG. 1) out of the generator polynomials for the component encoder. Therefore, when the number of the memories of the RSC encoder is m, a feedback sequence generated by the feedback polynomial continuously repeats the same pattern at a period of 2^m-1. Therefore, if an input information word ’1’ is received at the instance corresponding to this period, the same information bits are exclusive-ORed, so that the state of the RSC encoder becomes an all-zero state henceforth, thus generating the output symbols of all 0’s. This means that the Hamming weight of the codeword generated by the RSC encoder has a constant value after this event. That is, the free distance of the turbo code is maintained after this time, and the CISP becomes a main cause of a reduction in the free distance of the turbo encoder, whereas, as noted above, a larger free distance is desirable.

[0010] In this case (In the prior art of turbo interleaver) to increase the free distance, the turbo interleaver randomly disperses the CISP input information word so as to prevent a decrease in the free distance at the output symbol of the other component RSC encoder.
[0011] The above-stated properties are fundamental features of the known turbo interleaver. However, for the CISP, it is conventional that the information word has the minimum Hamming weight, when the input information word has the Hamming weight 2. In other words, the fact that the CISP can be generated even when the input information word has the Hamming weight 1 (i.e., when the input information word has one information bit of '1') was overlooked, when the information word input to the turbo encoder had the type of a block comprised of frames.
[0012] For example, a prime interleaver (PIL) designated as the working model of the turbo code interleaver specified by the present UMTS standard exhibits such problems, thus having a degraded free distance property. That is, the implementation algorithm of the model PIL turbo interleaver include 3 stages, of which the second stage, which plays the most important role, performs random permutation on the information bits of the respective groups. The second stage is divided into three cases of Case A, Case B and Case C, and the Case B always involves the case where the free distance is decreased due to the event where the input information word has the Hamming weight 1. In addition, even the Case C involves a possibility that such an event will occur. The detailed problems will be described later with reference to the PIL.
(…)
[0033] From Table 1, it is noted that if X(t)=1 at time t=7, then m(t), m(t-1) and m(t-2) become all zero states henceforth. Therefore, the Hamming weight of the following output symbols becomes always zero. In this case, if the turbo interleaver provides the RSC2 with the input information sequence '10000001000..._' as it is, the Hamming weight of the output symbols at the following time of t=7 will not change thereafter even in the RSC2 using the same feedback polynomial, for the same reason. This causes a decrease in the free distance of the whole output symbols of the turbo encoder. To prevent this, the turbo interleaver changes the original input information sequence '10000001000...' to an input information sequence of a different pattern (for example, changes a position of the information bit '1' such as 110000000...) and provides the resulting sequence to the RSC2. Therefore, even though an increase in the Hamming weight is stopped in the RSC1, the Hamming weight continuously increases in the RSC2, so that the total free distance of the turbo encoderincreases. This is because the feedback polynomial, having the infinite impulse response (IIR) filter type, continuously generates the infinite output symbol '1' even for one input information bit '1'. Equation 1 below shows the relationship between the RSC1 and the RSC2 in terms of the Hamming weight or free distance of the turbo encoder.

[Equation 1]

$$HW(\text{Output code sequence}) = HW(\text{RSC1 code sequence}) + HW(\text{RSC2 code sequence})$$

where HW is the Hamming weight.
(…)
[0036] For example, when only the information bit located at the last position of the input information word, i.e., the last position of the frame, is '1' and all the other information bits are 0's, the Hamming weight of the input information word becomes 1. In this case, the number of the symbols '1' output from the RSC 1 becomes very small, because there is no more input information word. Of course, when zero-tail bits are used, there exist two symbols but those are independently used rather than undergoing turbo interleaving. Therefore, it is assumed herein that the weight is slightly increased. Since the constant weight is added, this will be excluded from an analysis of the interleaver. In this case, it is

noted from Equation 1 that the RSC2 should generate a great number of the output symbols '1' to increase the total free distance.
(…)
[0039] If, as shown in FIG. 4, the turbo interleaver shifts (or permutes) the position of the input information word, where the original symbol of the RSC1 is '1', to the last position of the frame after interleaving, the number of the output symbols '1' generated from the RSC2 will be very small. In this case, since the RSC1 and the RSC2 generate a very small number of the output symbols '1' in accordance with Equation 1, the total free distance decreases drastically. However, if, as shown in FIG. 5, the turbo interleaver shifts the position of the input information word, where the original symbol of the RSC1 is '1', to the first position or a position near the leading position of the frame after interleaving, the number of the output symbols '1' generated from the RSC2 will be increased. This is because a plurality of symbols '1' are output through (N(Interleaver Size)-h(a number of '1')) state transitions of the RSC2 encoder. In this case, the RSC2 generates a great number of the output symbols '1', thereby increasing the total free distance.
(…)
[0045] Condition 2 : The information bits corresponding to the last position of the frame should be shifted to a position preceding the last position (if possible, to the leading position of the frame) by interleaving, to increase the free distance of the turbo code.
(…)
[0051] A second stage, Case-B, if C=p+1 out of an interleaving algorithm for the PIL interleaver which was provisionally determined as the UMTS turbo interleaver will be first described. In Equation 2 below, R indicates the number of groups (or rows), and has a value of R=10 or R=20. Further, C indicates the size of each group and is determined by the prime number p satisfying $0 \leq (p+1) - K/R$ as determined in Stage 1 according to a value K/R where K is the size of the actual input information bits of a frame. In Case-B, it is always that C=p+1. Therefore, the actual size of the PIL interleaver becomes a value determined by R×C, which is larger than K. Further, Cj(i) indicates a position of the information bits obtained by randomly permuting the position of the input information bits in the group on the basis of an ith group, where i=0,1,2,3,..., p. In addition, Pj indicates an initial seed value given for an jth row vector, and is initially given by the algorithm.
[Equation 2]
[0052]
B-1) A primitive root g0 is selected from a given random initialization constant table(3GPP TS 25.212 table 2; table of prime p and associated primitive root) such that g0 is a primitive root of a field based on prime p.
B-2) Construct base sequence C(i) to be used for row vector randomization is generated using the following formula.

$$C(i)=[g0 \times C(i-1)] \bmod p, \quad i=1,2,3,..,p-2, C(0)=1$$

B-3) Select the minimum prime integer set $\{q_j, j=0,1,2,\ldots, R-1\}$ such that $g.c.d\{q_j, p-1\} = 1$, $q_j > 6$ and $q_j > q_{(j-1)}$, where g.c.d is a greatest common divider and $q_0=1$
B-4) $\{p_j, j=0,1,2,\ldots, R-1\}$ which is a new prime number set is calculated from $\{q_j, j=0,1,2,\ldots, R-1\}$ such that $p_{p(j)} = q_j$ where, j=0, 1, ... R-1 and p(j) is the inter-row permutation pattern defined in the third stage.
B-5) Elements of the jth intra-row permutation as following method.

$$C_j(i)=C([i \times p_j] \bmod (p-1)), \quad i=0,1,2,3,\ldots,p-2,$$

$$C_j(p-1)=0,$$

and

$$C_j(p)=p$$

[0053] A third stage,
Perform the row-permutation based on the following p(j) (j=0,1,2...., R-1) patterns, where $p_{(j)}$ is the original row position of the j-th permuted row. The usage of these patterns is as follows; when the number of input information bit K is 320 to 480 bit perform group

selection pattern p_A, when the number of input information bit K is 481 to 530 bit perform group selection pattern p_C, when the number of input information bit K is 531 to 2280 bit perform group selection pattern p_A, when the number of input information bit K is 2281 to 2480 bit perform group selection pattern p_B, when the number of input information bit K is 2481 to 3160 bit perform group selection pattern p_A, when the number of input information bit K is 3161 to 3210 bit perform group selection pattern p_B, and when the number of input information bit K is 3211 to 5114 bit perform group selection pattern p_A. The group selection pattern is as follow;
p_A: {19, 9, 14, 4 0, 2, 5, 7, 12, 18, 10, 8, 13, 17, 3, 1, 16, 6, 15, 11} for R=20
p_B: {19, 9, 14, 4 0, 2, 5, 7, 12, 18, 16, 13, 17, 15, 3, 1, 6, 11, 8, 10} for R=20
p_C: {9,8,7,6,5,4,3,2,1,0} for R=10.
[0054] It should be noted herein that the last operation of B-5) is defined as $C_j(p)=p$. That is, this means that when the position of the input information bit before interleaving is p, the position of the input information bit is maintained at the position p even after PIL interleaving. Therefore, for the last group (j=19), the information bits $C_{R-1}(P)=C_{19}(p)$ existing at the last position maintain the same position i=P which is the last position of the 19th group. Therefore, Condition 2 for designing the turbo interleaver is not satisfied.
[0055] That is, to solve the problem that the PIL interleaver has, algorithm step B-5) may be modified as follows. The invention presents six methods of B-5-1) to B-5-6), by way of example Among these, an optimal performance can be determined through simulations in the light of the properties of the turbo interleaver.
(…)
[0058] Referring to FIG. 11, a row vector permutation block (or row vector permutation index generator) 912 generates an index for selecting a row vector according to counting of a row counter 911, and provides the generated index to a high address buffer of the address buffer 918. The row vector permutation block 912 is a group selector for sequentially or randomly selecting, when the input information word is divided into a plurality of groups, the divided groups. A column vector permutation block (or column vector's elements permutation index generator) 914 generates, depending on a modified PIL algorithm 915, an index for permuting the positions of the elements in the corresponding row vector (or group) according to counting of a column counter 913, and provides the generated index to a low address buffer of the address buffer 918. The column vector permutation block 914 is a randomizer for permuting the position of the information bits in the group, which were sequentially stored in the order of input, according to a given rule. A RAM (Random Access Memory) 917 stores temporary data generated in the process of the program. A look-up table 916 stores parameters for interleaving and the primitive root. The addresses obtained by row permutation and column permutation (i.e., the addresses stored in the address buffer 918) are used as addresses for interleaving.

2.9. Bij EP 516 behoren onder meer de volgende figuren:

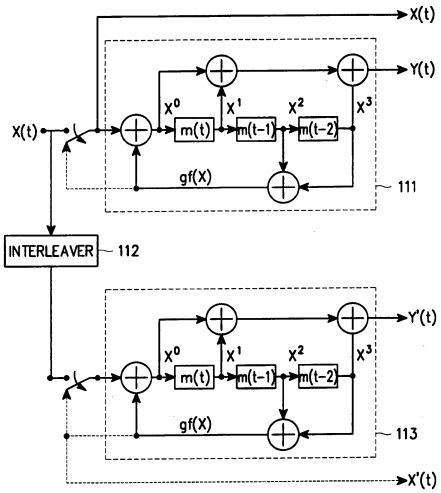


FIG. 1

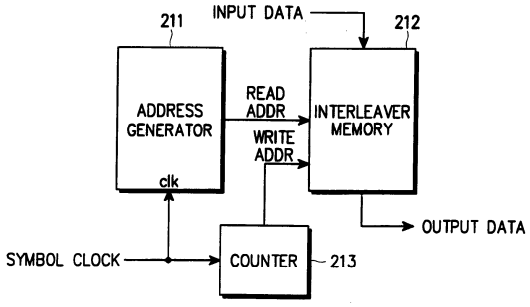


FIG. 2

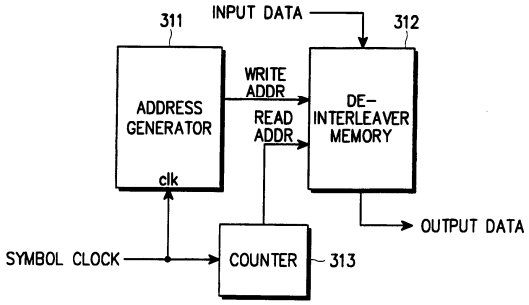


FIG. 3

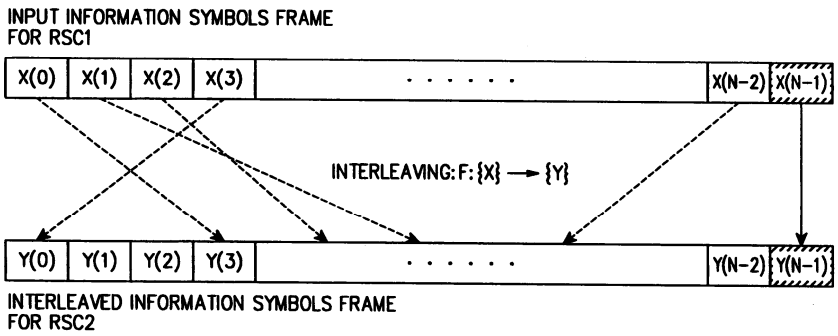


FIG. 4

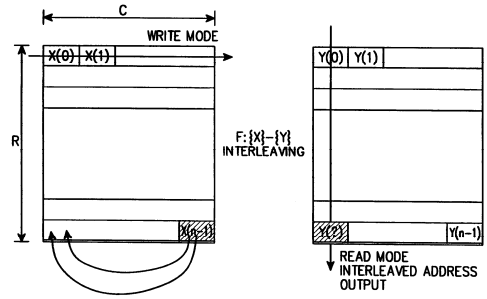


FIG. 9

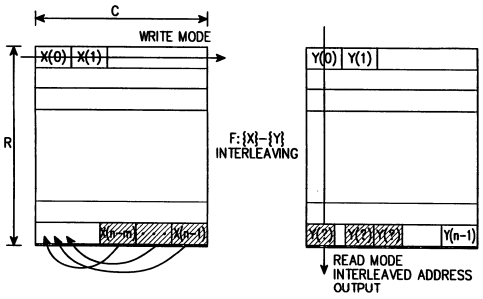


FIG. 10

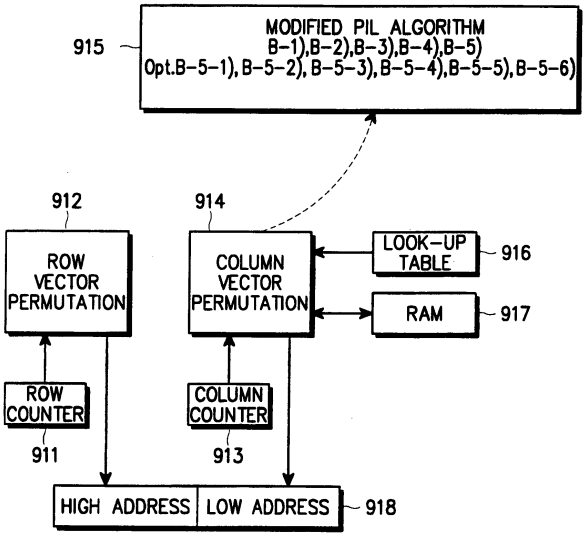


FIG. 11

2.10. Op 27 januari 2012 heeft het Landgericht Mannheim de vorderingen van Samsung tegen Apple op basis van EP 528 afgewezen (Geschäftsnummer 7 O 325/11).

3. DE VERDERE BEOORDELING

in conventie

3.1. Inleidende opmerkingen

3.1.1. De rechtbank volhardt bij hetgeen in het tussenvonnis van 14 maart 2012 is vastgesteld en overwogen. Uit dit tussenvonnis volgt dat enkel nog de door Samsung gestelde inbreuk door de Infineon/Intel geproduceerde *baseband chips* beoordeeld behoeft te worden. Met partijen zal de rechtbank ervan uitgaan dat de Intel-chips toegepast in de iPhone 3G, 3GS en 4 alsmede in de iPad 1 en 2 technisch gelijk zijn aan de door Infineon tot 31 januari 2011 geproduceerde chips.

3.1.2. Er wordt heden tevens afzonderlijk vonnis gewezen in de twee andere zaken tussen dezelfde partijen met zaak/rolnummers 400376 / HA ZA 11-2213 (EP 269) en 400385 / HA ZA 11-2215 (EP 136) waar het tussenvonnis op zag.

3.1.3. De opbouw van de beoordeling in dit vonnis is als volgt. De rechtbank zal eerst bij ieder octrooi (hoofdstukken 3.2 respectievelijk 3.4) een inleiding van de daarin aan de orde zijnde techniek geven, welke inleiding is ontleend aan de toelichting door Apple bij conclusie van antwoord, tevens conclusie van eis in voorwaardelijke reconventie. Vervolgens zal de rechtbank per octrooi haar beoordeling van de door partijen ingenomen stellingen en daartegen opgeworpen verweren geven, een en ander voor zover nodig (hoofdstukken 3.3 en 3.5). Tot slot zal een conclusie worden getrokken en zal over de proceskosten worden beslist (hoofdstuk 3.6).

3.1.4. Apple heeft ter zitting desgevraagd nader uitgelegd hoe zij precies het voorwaardelijke karakter van haar eis in reconventie bedoelt. De rechtbank begrijpt dit thans aldus dat uitsluitend aan de reconventionele vordering wordt toegekomen als de rechtbank in conventie een nietigheidsargument gegrond oordeelt. Als de rechtbank de vorderingen in conventie op andere gronden afwijst, acht Apple het niet nodig dat ook de nietigheid zal worden beoordeeld en uitgesproken van het betreffende octrooi. Aan de rechtbank wordt door Apple dus de vrije keuze gelaten of de zaak wordt afgedaan op nietigheid of op andere verweren.

dagvaarding niet nietig

3.1.5. Als verweer van de verste strekking heeft Apple een beroep gedaan op de nietigheid van de dagvaarding in de zin van artikel 111 lid 1 jo. 111 lid 2 aanhef en sub d jo. artikel 120 lid 1 Rv. Zij heeft aangevoerd dat Samsung de gestelde inbreuk in haar dagvaarding slechts heeft onderbouwd met een verwijzing naar de 3G-standaard, terwijl zulks niet ‘zonder meer’ volstaat om de inbreuk te bewijzen nu standaarden volgens haar vaak een ruime mate van vrijheid toelaten met betrekking tot de implementatie ervan. Dit betekent volgens Apple dat de dagvaarding nietig dient te worden verklaard.

3.1.6. Het beroep wordt verworpen. Samsung heeft in de dagvaarding gesteld dat de producten van Apple voldoen aan de 3G-standaard en de gestelde inbreuk aan de hand van die standaard onderbouwd. Indien die gronden, tezamen met de bij pleidooi gegeven

toelichting, onvoldoende zijn voor toewijzing van de eis, betekent zulks niet dat daarmee ook de dagvaarding nietig zou zijn.

3.2. EP 528 - Inleiding op de techniek

Hoe werkt digitale draadloze telefonie?

3.2.1. Figuur A hieronder is een sterk vereenvoudigde voorstelling van een mobiel of draadloos digitaal telefoonnetwerk. Wanneer mobiele telefoons A en B ten behoeve van een telefoongesprek met elkaar zijn verbonden en de gebruiker van mobiele telefoon A in de telefoon spreekt, vindt in wezen het volgende proces plaats: (1) de elektronica in mobiele telefoon A zet het geluid van de stem van de spreker om in een digitaal elektrisch signaal en zendt dit signaal via een digitale draadloze radioverbinding naar basisstation 1; (2) basisstation 1 geeft dit signaal via een netwerkverbinding en zonodig tussengeschakelde telefooncentrales vervolgens door aan basisstation 2; (3) basisstation 2 zendt het signaal via een andere digitale draadloze radioverbinding vervolgens naar mobiele telefoon B; en (4) mobiele telefoon B zet het ontvangen digitale signaal vervolgens om in een geluidssignaal dat het stemgeluid van de gebruiker van de mobiele telefoon A reconstrueert en voor de gebruiker van mobiele telefoon B hoorbaar maakt. Er vindt een soortgelijk proces plaats om de gebruiker van mobiele telefoon A te laten horen wat er door de spreker van mobiele telefoon B wordt gezegd. Soortgelijke processen laten mobiele telefoons met draadgebonden of vaste telefoons communiceren.

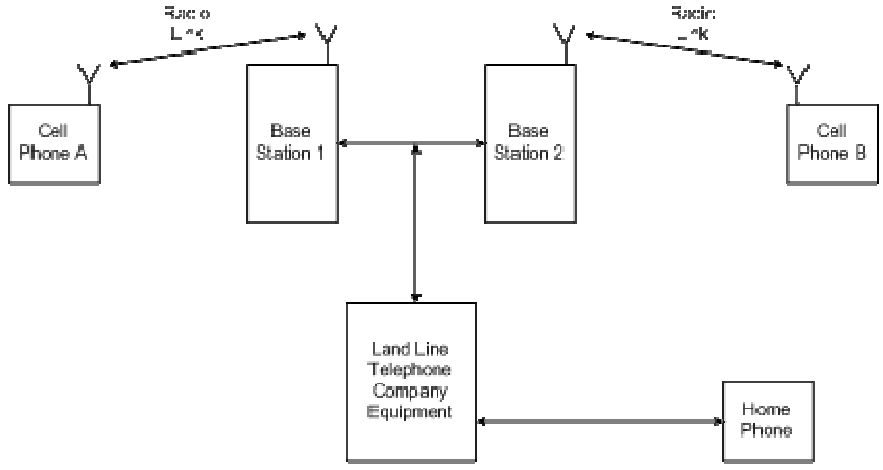


Figure A

3.2.2. UMTS staat voor Universal Mobile Telecommunications System. Naast het digitaal verzenden en ontvangen van spraaksignalen zijn UMTS telefoons in staat om met relatief hoge snelheid bijvoorbeeld tekst, beeld of video en andere audiosignalen te verzenden en te ontvangen, zoals bijvoorbeeld foto's, draadloos internet, e-mail, presentaties en databestanden.

3.2.3. Omdat UMTS naast mobiele telefonie verschillende andere vormen van telecommunicatie ondersteunt, spreekt men meer algemeen ook van een mobiel communicatienetwerk of mobiel communicatiesysteem en mobiele gebruikersapparatuur,

mobiele zend/ontvanginrichting en dergelijke. Met de aanduiding mobiele telefoon en/of mobiele telefonie worden in het navolgende ook dergelijke andere vormen van telecommunicatie omvat.

3.2.4. Zoals hieronder in hoofdstukken 4 en 8 verder beschreven wordt, hebben EP 528 en EP 516 betrekking op bepaalde processen die in een mobiele telefoon plaatsvinden.

3.2.5. Figuur B hieronder is een vereenvoudigde afbeelding van een mobiele zendinrichting. De ingeroepen octrooien hebben voornamelijk betrekking op de verwerking die plaatsvindt in het blok dat is aangeduid als “Processing B”, maar om dit blok in zijn context te plaatsen worden de andere verwerkingsblokken hieronder ook kort beschreven.

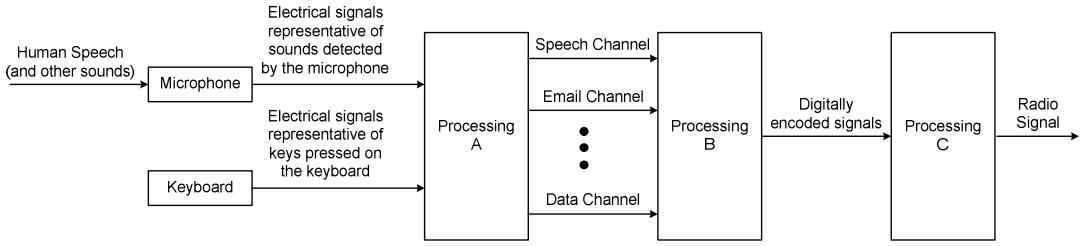


Figure B

3.2.6. Ter vereenvoudiging zijn het c.q. de basisstation(s) en delen van de mobiele telefoon die signalen van het basisstation ontvangen in Figuur B weggelaten.

3.2.7. Zoals in Figuur B te zien is, bevatten mobiele telefoons over het algemeen verschillende mogelijkheden voor het door een gebruiker invoeren van data: zoals bijvoorbeeld een microfoon en een toetsenbord of aanraakscherm. De microfoon zet de geluiden van menselijke spraak om in elektrische signalen die door de elektronica in de telefoon verder worden verwerkt tot digitale datasignalen voor verzending daarvan (dat wil zeggen bemonsterd, gedigitaliseerd, gecomprimeerd, etc.). Het toetsenbord (of andere data-invoerinrichtingen, zoals een aanraakscherm) laat toe dat telefoonnumers gekozen worden en laat de gebruiker (bij moderne mobiele telefoons) ook data invoeren voor het versturen van e-mails, browsen op het internet, evenals andere computernetwerkfuncties.

3.2.8. De data die door de microfoon en toetsenbord (of aanraakscherm) gegenereerd worden, worden gewoonlijk opgedeeld in virtuele “kanalen” waarvan ieder kanaal separaat verwerkt wordt. Zoals bijvoorbeeld illustratief te zien is in Figuur B, plaatst het blok aangeduid als “Processing A” (1) alle spraakdata met betrekking tot een telefoongesprek op een “Spraakkanaal,” (2) alle data met betrekking tot e-mails op een “e-mailkanaal,” en (3) alle data met betrekking tot andere functies (bijvoorbeeld browsing op het internet of versturen van tekstboodschappen) op andere kanalen.

3.2.9. Het blok aangeduid als “Processing B” zal hieronder nader beschreven worden. Kort gezegd, codeert het blok “Processing B” de te verzenden data om te verzekeren dat deze betrouwbaar naar een basisstation kunnen worden verzonden. Het blok aangeduid als “Processing C” zet de data tenslotte om in een radiosignaal dat draadloos naar een basisstation kan worden verzonden.

Binaire signalen – Wat zijn bits en bytes?

3.2.10. Moderne mobiele telefoons, zoals de in de deze zaak bestreden iPhones, verwerken informatie in de vorm van digitale of "binaire" signalen. Een binair signaal is een volgorde van "bits," waarvan ieder een waarde heeft van of "1" of "0". Verzamelingen van zulke bits kunnen bijvoorbeeld gebruikt worden om het geluid van een mobiel telefoongesprek of een via een mobiele telefoon verzonden e-mail te representeren. Wanneer een persoon in een mobiele telefoon spreekt, zet de telefoon de geluiden van de spraak letterlijk om in een digitaal of binair signaal, dat wil zeggen een reeks van de getallen "1"en "0" die het spraakgeluid representeren. Mobiele telefoons kunnen dergelijke binaire volgordes ook weer omzetten in hoorbare signalen die mensen kunnen herkennen als spraak.

3.2.11. Groepen bits kunnen gebruikt worden om in het "binaire of tweetallige getalstel" getallen te representeren, op dezelfde manier als dit gebeurt met de ons meer bekende cijfers 0 t/m 9 in het "decimale getalstelsel". In het binaire getalstelsel representeert elke bitpositie een waarde gelijk aan een macht van het getal 2, van rechts naar links te beginnen bij 2^0=1, 2^1=2, 2^2=4, 2^3=8. De uiteindelijke waarde die een reeks van bits voorstelt is dan de som van al deze waardes, waarbij een "0" aangeeft dat de betreffende waarde niet en een "1" aangeeft dat de betreffende waarde wel moet worden opgeteld. Het binaire blok bestaande uit 4 bits bijvoorbeeld "1101" representeert dan het decimale getal "13". Ieder decimaal getal kan op dezelfde manier voorgesteld worden door een reeks bits in het binaire getalstelsel. Hoewel mensen over het algemeen bekender zijn met het decimale stelsel, is digitale elektronische apparatuur, zoals mobiele telefoons, meer geschikt om te werken met gebruikmaking van het binaire getalstelsel.

3.2.12. Zoals hierboven opgemerkt is "bit" een enkel element van een binair signaal en kan slechts twee mogelijke waarden hebben: "1" of "0". Een "byte" is verzameling van acht bits, een blok van 8 bits. Binaire signalen worden vaak in een verzameling van "bytes" georganiseerd. Drie willekeurige voorbeelden van "bytes" worden hieronder gegeven:

"11010011"
"01010101"
"01110010"

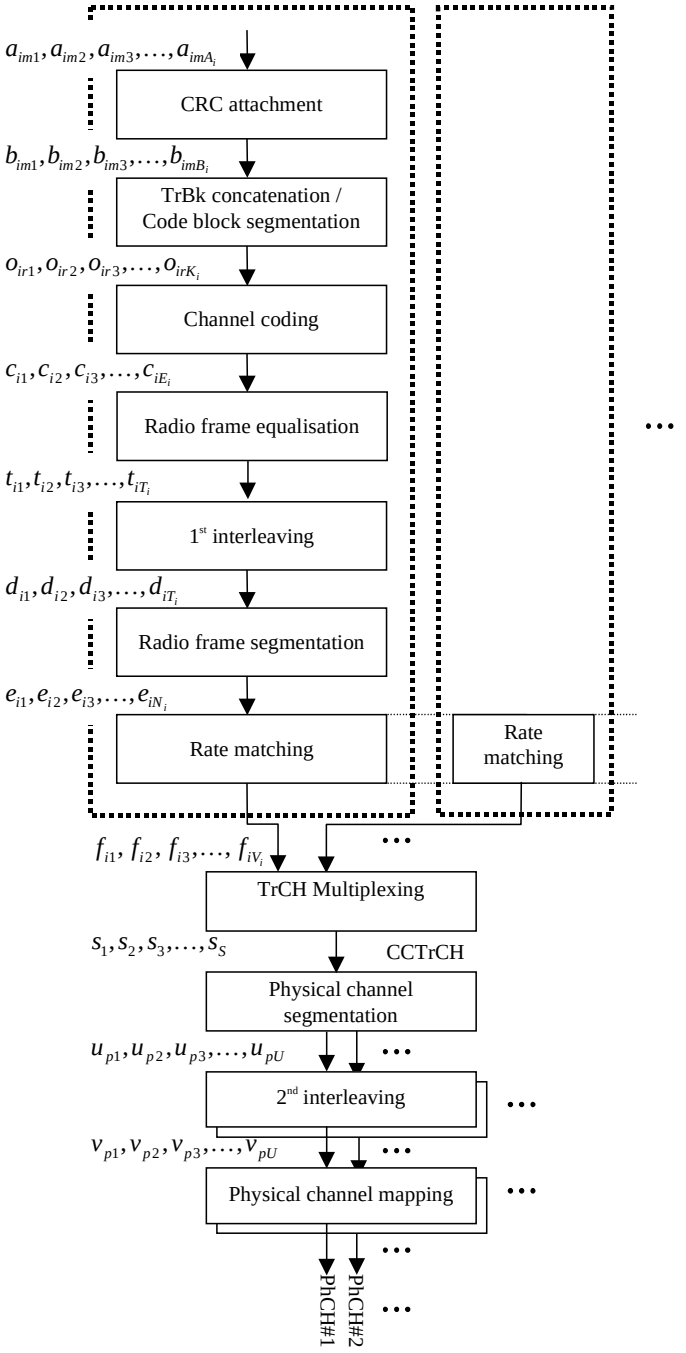
ETSI TS 125.212

3.2.13. Samsung stelt dat de twee ingeroepen octrooien essentieel zijn voor de ETSI TS 125.212 communicatiestandaard (de "25.212 standaard" of kortweg de "standaard"). Indien in het navolgende niet specifiek verwezen wordt naar een (eerdere) versie van de 25.212 standaard, dient begrepen te worden versie 3.11.0 ervan (Samsung productie 2). Figuur 1 van de 25.212 standaard is hierna afgebeeld. Figuur 1 is getiteld "Transport channel multiplexing structure for uplink". De term "uplink" verwijst naar data die van een mobiele telefoon naar een basisstation verzonden worden.[2] Figuur 1 illustreert het verwerkingsproces zoals uitgevoerd door mobiele telefoons (die voldoen aan de standaard) voor data die van de mobiele telefoon naar het basisstation verzonden worden. Met andere woorden, Figuur 1

[2] Op dezelfde manier verwijst de term "downlink" naar data die van een basisstation naar een mobiele telefoon verzonden worden.

laat zien hoe alle binaire bits die door de telefoon gegenereerd worden (bijvoorbeeld bits die door de microfoon van de mobiele telefoon gegenereerd worden), zo verwerkt worden dat ze betrouwbaar naar een basisstation verzonden kunnen worden.

3.2.14. Alle verwerkingsprocessen die door Figuur 1 van de 25.212 standaard zijn weergegeven, zijn processen die zich afspelen in het blok aangeduid als “Processing B” in bovenstaande Figuur B.



Figuur 1: Transport channel multiplexing structure for uplink

3.2.15. In Figuur 1 stelt de grote gestippelde rechthoek (welke blokken omvat met de volgende aanduidingen: “CRC attachment,” “TrBk concatenation / Code block segmentation,” “Channel coding,” “Radio frame equalisation,” “1st interleaving,” “Radio frame segmentation,” en “Rate matching”) het verwerkingsproces voor dat in een betreffende kanaal wordt uitgevoerd, bijvoorbeeld het spraakkanaal. Een andere grote gestippelde rechthoek in Figuur 1 geeft aan dat data in alle andere kanalen (bijvoorbeeld het e-mail kanaal or het internet browser kanaal) op dezelfde manier verwerkt worden. De informatiestroom loopt van boven naar beneden in de figuur.

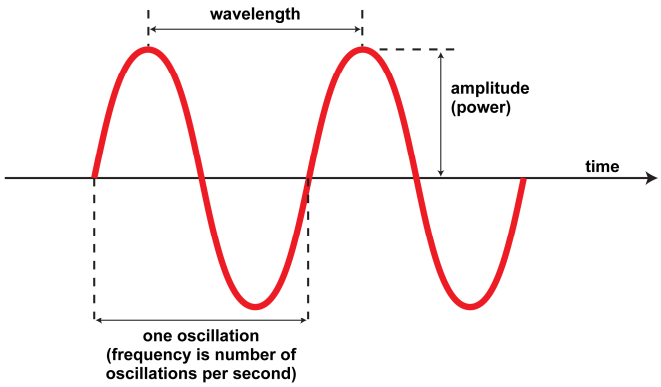
3.2.16. De in Figuur 1 van de 25.212 standaard geïllustreerde verwerkingsprocessen die voor de onderhavige zaak van belang zijn worden hieronder kort beschreven.

Kanaalcodering (“channel coding”) van Fig. 1 van de 25.212 standaard

3.2.17. Het doel van het kanaalcoderingsblok “Channel coding” van Figuur 1 is om schadelijke gevolgen van storingen op het radiopad zoveel mogelijk te reduceren.

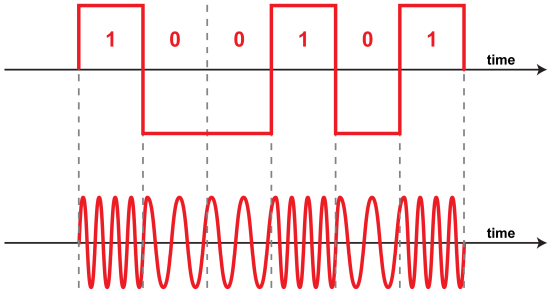
Het storingsprobleem

3.2.18. Zoals hierboven opgemerkt, werken mobiele telefoons door radiosignalen draadloos naar een basisstation te verzenden. Deze radiosignalen bestaan uit elektromagnetische golven die gemoduleerd zijn om informatie te kunnen dragen. Een algemene illustratie van een elektromagnetische golf ziet er als volgt uit:

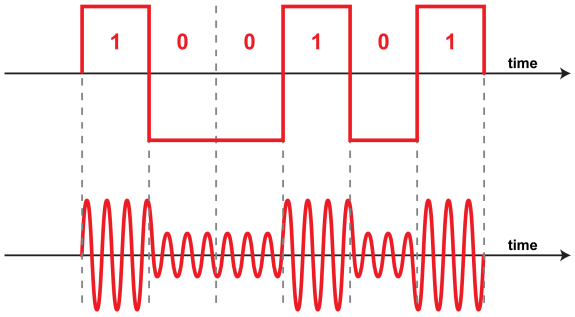


3.2.19. Basiseigenschappen van een elektromagnetische golf zijn frequentie en amplitude. De frequentie van een elektromagnetische golf verwijst naar het aantal trillingen daarvan per seconde. De amplitude van een golf verwijst naar de signaalsterkte hiervan. Modulatie verwijst naar het proces om deze en/of andere eigenschappen van een golf in de tijd te wijzigen om zo informatie te kunnen representeren.

3.2.20. Het diagram hierna bijvoorbeeld illustreert het gebruik van frequentie-modulatie om een reeks van bits elk met de waarde “0” of “1” voor te stellen. Een bit met de waarde “1” wordt bijvoorbeeld gerepresenteerd door een hogere frequentie, dat wil zeggen meer golven per tijdseenheid, dan een bit met de waarde “0” die door een lagere frequentie, dat wil zeggen minder golven per tijdseenheid wordt gerepresenteerd.



3.2.21. Het volgende diagram illustreert amplitudemodulatie van een golf om dezelfde volgorde van de bits “0” en “1” voor te stellen. Dat wil zeggen, een hogere signaalsterkte of grotere uitslag van de golf representeert een bit met de waarde “1” en een kleinere signaalsterkte representeert een bit met de waarde “0”.



3.2.22. De eenvoudige frequentie- en amplitude-modulatiediagrammen waarnaar in de vorige paragrafen verwezen wordt, geven een algemene illustratie van een modulatie van een signaal om informatie te dragen. Het UMTS systeem dat hier ter discussie staat, gebruikt complexere en samengestelde modulatiediagrammen dan de hierboven getoonde, maar het basisidee is hetzelfde. Ook in UMTS worden elektromagnetische golven gemoduleerd om digitale informatie voor te stellen.

3.2.23. Figuur C hierna geeft een algemeen beeld van een draadloze transmissie.

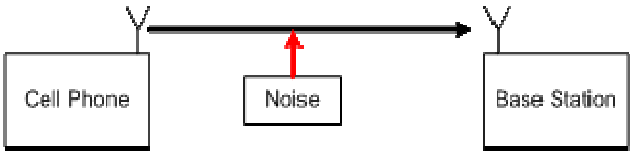


Figure C

Zoals te zien is in Figuur C, wordt het signaal dat door de mobiele telefoon naar het basisstation wordt verzonden door storing op het radiopad, hier weergegeven door ruis (“Noise”), verstoord. Er zijn veel bronnen van elektromagnetische straling die interfereren met een bepaald radiosignaal, zoals TV- en radiouitzendingen, mobilofoons, kosmische straling of omgevingsstraling van bijvoorbeeld magnetronovens maar ook signaaltransmissies van andere mobiele telefoons. Een radiosignaal zwakt ook in sterkte af als het zich door de lucht voortplant en door materie zoals de muren van gebouwen heen

gaat. Het radiosignaal dat bij een ontvanger zoals een basisstation ontvangen wordt, heeft in het algemeen een erg lage sterkte. Om het signaal te verwerken moet het bij de ontvanger worden versterkt. Ook deze verwerking introduceert een zekere mate van verstoring. Praktisch gezien betekent dit, dat het basisstation een bepaald percentage van de door de mobiele telefoon verzonden signaalbits niet correct zal ontvangen. Dat wil zeggen dat in sommige gevallen de mobiele telefoon een golfvorm zal verzenden die een “1” voorstelt, maar vanwege storing zal het basisstation een golfvorm ontvangen die er als een “0” uitziet. Op dezelfde manier zal de mobiele telefoon in andere gevallen een “0” verzenden terwijl het basisstation een golfvorm ontvangt die op een “1” lijkt.

3.2.24. Storing uitschakelen is praktisch niet mogelijk, maar het is wel mogelijk om de gevolgen hiervan voor het betrouwbaar overdragen van digitale informatie te verkleinen. Er zijn manieren om de betrouwbaarheid van door de lucht verzonden radiosignalen te verhogen. Een strategie is bijvoorbeeld het vergroten van de zendsterkte om bijvoorbeeld boven het ruisniveau te blijven. Dit is gelijk aan stemverheffing in een rumoerige kamer om zo beter gehoord te kunnen worden. Maar het verhogen van de sterkte is over het algemeen geen erg effectieve oplossing voor het probleem van het verhogen van de betrouwbaarheid van radioverzending. Regelgevende- en veiligheidsinstanties leggen bijvoorbeeld wettelijke beperkingen op aan het sterkteniveau dat door mobiele telefoons en andere draadloze apparaten verzonden mag worden. Bovendien vereist het versterkt versturen ook dat de mobiele telefoon meer energie *verbruikt*, hetgeen tot een verminderde gebruikduur voor de accu van de mobiele telefoon leidt. Een andere reden waarom het verhogen van de sterkte geen goede oplossing is in mobiele telefoonsystemen, is dat als alle mobiele telefoons hun signaalsterkte zouden vergroten, zij sterker met elkaar zouden interfereren bij deze toegenomen signaalsterkte. Dit is net alsof iedereen in een volle kamer begint te schreeuwen. Het is derhalve over het algemeen gewenst om de betrouwbaarheid van signalen van een gegeven signaalsterkte te verhogen.

3.2.25. Een andere manier op het schadelijke effect van storingen te reduceren is door foutencorrectie-codeertechnieken te gebruiken. Zulke codering houdt het coderen van informatie in op zo’n manier, dat de data in aanwezigheid van storing, zoals ruis, betrouwbaar kunnen worden overgedragen. Een manier om te coderen is het gebruik van een code welke als “systematische code” wordt aangeduid. Bij dergelijke codes worden extra bits toegevoegd, die uit de oorspronkelijke data van een signaal of bericht worden afgeleid. Deze additionele bits worden in dit verband ook wel pariteitbits (“parity bits”) genoemd. Het gecodeerde bericht bestaat dan uit de bits van de oorspronkelijke boodschap (ook wel de systematische bits genoemd) die naar het basisstation overgebracht moeten worden, tesamen met overtollige informatie ook wel redundantie genoemd, dat wil zeggen de pariteitbits die in relatie staan tot de oorspronkelijke bits en die assisteren bij het proces van een betrouwbare transmissie.

<u>Een eenvoudig codevoorbeeld</u>

3.2.26. Een heel eenvoudig soort code behelst het repliceren van elke informatiebit. Een voorbeeld van een dergelijke code is hieronder te zien. In het onderstaande voorbeeld wil de mobiele telefoon een enkele informatiebit, dat wil zeggen “0,” naar het basisstation verzenden. Maar in plaats van alleen die ene bit te verzenden, verzendt de mobiele telefoon drie bits, die in het geval van een “0” alle drie een “0” zijn:

Oorspronkelijk bericht: 0
Gecodeerd bericht: 000
en in het geval van een “1” alle drie een “1” zijn:
Oorspronkelijk bericht: 1
Gecodeerd bericht: 111

3.2.27. Als er zich een enkele bitfout voordoet tijdens verzending, zou het basisstation bijvoorbeeld het onderstaande bericht kunnen ontvangen:

Bericht ontvangen door basisstation: 001

3.2.28. Bij het ontvangen van het bericht “001,” kan het basisstation afleiden dat er een fout is opgetreden tijdens de verzending. Dat wil zeggen, het basisstation kent de door de mobiele telefoon gebruikte code en weet daarom dat de mobiele telefoon altijd drie identieke bits op een rij verzendt. Ontvangst van drie opeenvolgende bits die niet identiek zijn, geeft aan dat er een fout is opgetreden. Aangezien twee van de ontvangen bits het getal “0” zijn en slechts één van de bits een “1” is, kan het basisstation concluderen dat er waarschijnlijk een “0” is verzonden en dat tijdens de verzending één van de nullen in een “1” is verwisseld. Met andere woorden, het basisstation kan een eenvoudige “meerderheidsregel” gebruiken. De tabel hieronder geeft aan hoe het basisstation alle drie mogelijke bitvolgordes zou interpreteren met gebruikmaking van deze twee-uit-drie regel.

Drie bit-patroon ontvangen door het basisstation	Interpretatie van het basisstation van wat de mobiele telefoon had willen verzenden	Interpretatie van het basisstation of er een fout is opgetreden tijdens de verzending en, indien dit het geval is, wat voor soort fout
000	0	Waarschijnlijk geen fout opgetreden, alle drie bits komen overeen
001	0	Een fout opgetreden, hoewel de laatste bit is ontvangen als een “1” heeft de mobiele telefoon waarschijnlijk een “0” gestuurd
010	0	Een fout opgetreden, hoewel de middelste bit is ontvangen als een “1” heeft de mobiele telefoon waarschijnlijk een “0” gestuurd
011	1	Een fout opgetreden, hoewel de eerste bit als een “0” ontvangen is, heeft de mobiele telefoon waarschijnlijk een “1” gestuurd
100	0	Een fout opgetreden, hoewel het eerste bit als een “1” ontvangen is, heeft de mobiele telefoon waarschijnlijk een “0” gestuurd
101	1	Een fout opgetreden, hoewel de middelste bit als een “0” ontvangen is, heeft de mobiele telefoon waarschijnlijk een “1” gestuurd
110	1	Een fout opgetreden, hoewel de laatste bit als een “0” ontvangen is, heeft de mobiele telefoon waarschijnlijk een “1” gestuurd
111	1	Waarschijnlijk geen fout opgetreden, alle drie bits komen overeen

3.2.29. De codeertechniek van het drie keer herhalen van iedere bit kan uitgebreid worden tot het versturen van langere reeksen zoals hieronder aangegeven:

Oorspronkelijke boodschap:	01101111
Gecodeerde boodschap:	000 111 111 000 111 111 111 111

3.2.30. Hoewel de hierboven beschreven herhaalcode eenvoudig is, is zij niet bijzonder efficiënt in de zin dat er drie keer zoveel bits verstuurd moeten worden voor een beperkte foutbescherming. Een voorbeeld van de problemen met deze code is wanneer er twee bitfouten in het drie bits bevattende verzonden signaal optreden. Het voorbeeld hieronder laat een geval zien waarin de mobiele telefoon een "0" verzendt maar dat door twee bitfouten gedurende de transmissie het basisstation het bericht foutief als een "1" decodeert:

Oorspronkelijk bericht:	0
Gecodeerd bericht:	000
Bits ontvangen door basisstation:	110
[Let op: er zijn twee fouten opgetreden]	
Bericht gedecodeerd door basisstation:	1

3.2.31. De sleutel tot een goede foutencorrectie is dan ook het kiezen van een geschikte foutencorrectiecode waarmee zoveel mogelijk verstoorde bits van een informatiestroom kunnen worden gecorrigeerd. Het is bekend dat dit bereikt kan worden door een zogeheten bloklengtecode te gebruiken. Bijvoorbeeld, een met snelheid 1/3 aangeduide bloklengtecode, waarmee een boodschap die bestaat uit een blok van k bits wordt gecodeerd in een blok van n=3*k bits. Als de code systematisch zou zijn, zou de codeerinrichting het codewoordblok vormen door berekening van 2*k pariteitbits die samen met de oorspronkelijke k informatiebits van het boodschapblok verstuurd worden. Voor iedere oorspronkelijke bit worden dan twee additionele pariteitbits toegevoegd; dat wil zeggen er gaat 1 bit de codeerinrichting in en er komen er 3 uit (1 op 3, geschreven als 1/3).

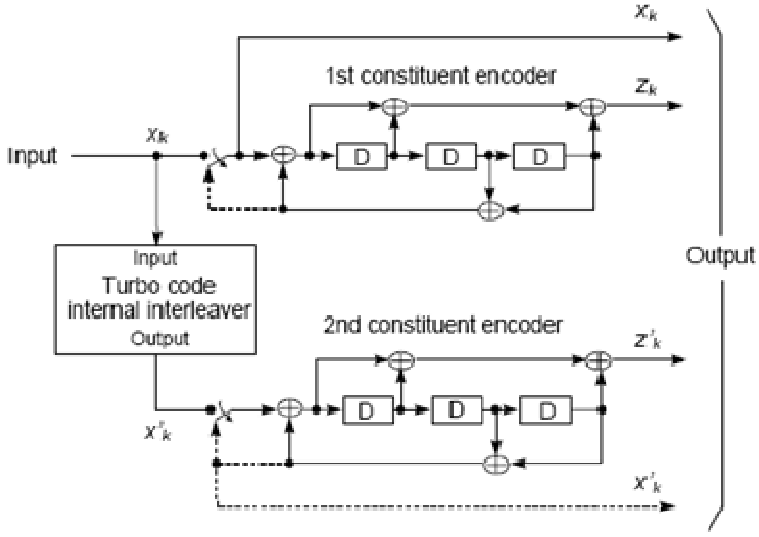
3.2.32. Een andere methode om informatie over verschillende transmissies te verdelen is het gebruik van een "convolutionele code". Een convolutionele codeerinrichting is een inrichting met een geheugen waarin gecodeerde informatie wordt gebruikt om de staat van de codeerinrichting te ontwikkelen. Een convolutionele code en een andere geheugeninrichting die (interne) interleaver[3] genoemd wordt, zijn de belangrijkste componenten in een klasse van krachtige foutencorrectiecodes die bekend staan als "turbocodes".

Turbocodes

3.2.33. Turbocodes zijn efficiënter in het reduceren van fouten in de gecodeerde data dan de simpele codeerinrichting zoals boven bij wijze van voorbeeld is beschreven. De tekening hieronder laat een voorbeeld zien van een turbocoder.[4] De geïllustreerde turbocoder

[3] Een interne interleaver (in een codeerinrichting) is iets anders dan de (eerste) interleaver die hierna zal worden besproken.
[4] De tekening van de turbocoder is overgenomen van de 25.212 standaard. Zie 25.212 bij Figuur 4.

ontvangt een invoerstroom van bits aangeduid als X_k en genereert uit die invoer drie stromen uitvoerbits aangeduid als X_k, Z_k, Z'_k.[5]



3.2.34. De uitvoerstroom X_k is identiek aan de invoerstroom. Deze uitvoerstroom wordt ook als de "systematische bits" aangeduid. De uitvoerstroom Z_k wordt de "eerste pariteitbits" en de uitvoerstroom Z'_k wordt als de "tweede pariteitbits" aangeduid.

3.2.35. Hoewel deze turbocode complexer is dan de hierboven besproken eenvoudige (iedere bit drie keer herhalen) code, delen beide codes belangrijke kenmerken. Ten eerste, in beide codes worden extra bits, dat wil zeggen pariteitbits, gegenereerd. Om voordeel te halen uit foutencorrectie, moet de mobiele telefoon tenminste sommige van deze extra bits naar het basisstation verzenden. Ten tweede, in beide codes worden er voor iedere bit die de mobiele telefoon wil verzenden, drie bits gegenereerd. In technische termen hebben beide codes een codeersnelheid van 1/3 (dat wil zeggen, voor iedere invoerbit worden er drie uitvoerbits gegenereerd).

<u>De kanaalcodeerinrichting ("channel coder") van de 25.212 standaard</u>

3.2.36. De kanaalcodeerinrichting van de 25.212 standaard gebruikt de hierboven afgebeelde turbocoder of een andere codeerinrichting, die bekend staat als een convolutionele codeerinrichting ("convolutional coder").

3.2.37. Onder verwijzing naar Figuur 1 van de 25.212 standaard (zie hiervoor) hebben de eerste drie blokken, dat wil zeggen de blokken aangeduid als "CRC attachment," "TrBk concatenaton / Code block segmentation," en "Channel coding", alle van doen met het reduceren van storingseffecten. De "CRC attachment" en "TrBk concatenation / Code block segmentation" blokken bewerken data voor verzending. Het "Channel coding"-blok codeert

[5] De turbocoder genereert ook een vierde uitvoerstroom aangeduid als X'_k. Die vierde stroom die gebruikt wordt voor wat bekend staat als "trellis termination," wordt in het vervolg van deze conclusie verder besproken.

vervolgens de data waarbij of een turbocoder of een convolutionele codeerinrichting wordt gebruikt.

3.2.38. EP 516 heeft betrekking op de hiervoor beschreven processtap van de kanaalcodering (channel coding) in een turbocoder.

Radioframevereffening ("Radio Frame Equalization") van de 25.212 standaard

3.2.39. Mobiele telefoons die voldoen aan de 25.212 standaard verzenden data in tijdsperioden die "transmission time intervals" (TTI's) genoemd worden. Een enkel TTI kan 10 milliseconden ("ms"), 20 ms, 40 ms of 80 ms lang zijn. Binnen een enkel TTI groepeert de mobiele telefoon de gegevens in één of meer radioframes. Een TTI dat 10 ms lang is, zal slechts één radioframe hebben. Een TTI dat 20 ms lang is, zal twee radioframes hebben. Een TTI dat 40 ms lang is zal vier radioframes hebben en een TTI dat 80 ms lang is, zal acht radioframes hebben.

3.2.40. Onder verwijzing naar Figuur 1 van de 25.212 standaard, is het doel van het "Radio frame equalization"-blok om te bewerkstelligen dat ieder "radio frame" hetzelfde aantal bits bevat. Het radio frame equalization-proces voegt, indien noodzakelijk,vulbits ("filler bits") toe om er zeker van te zijn dat ieder radio frame telkens hetzelfde aantal bits bevat.

3.2.41. Om een voorbeeld te geven, stel dat een TTI 20 ms lang is en dat de mobiele telefoon daarom twee radioframes moet genereren. Stel eveneens dat de kanaalcodeerinrichting 8.000 bits in dat TTI produceert. In dat geval zal het radio frame equalization-proces de 8.000 bits eenvoudig in twee gelijke radio frames van 4.000 bits elk verdelen. Als de kanaalcodeerinrichting echter 7.999 bits in het TTI produceert, zal het radio frame equalization-proces een filler bit toevoegen, zodat het totale aantal bits deelbaar wordt door twee. Dat wil zeggen, nadat het filler bit is toegevoegd, zal er een totaal van 8.000 bits zijn, en van deze bits vormt het radio frame equalization-proces twee radioframes waarvan ieder 4.000 bits omvat.

3.2.42. Over het algemeen is het zo dat als een mobiele telefoon "n" radioframes gebruikt (het getal "n" zal of 1, 2, 4 of 8 zijn), de radio frame equalization-stap evenveel filler bits aan de uitvoer van de kanaalcodeerinrichting toevoegt als noodzakelijk om er voor te zorgen dat het totale aantal bits exact gedeeld kan worden door "n". Nadat de noodzakelijke filler bits zijn toegevoegd, wordt tijdens de radio frame equalization-stap het totale aantal bits (dat wil zeggen, bit uitvoer vanuit de kanaalcodeerinrichting plus toegevoegde filler bits) in "n" eenheden van gelijke afmeting verdeeld, aangeduid als "radio frames."

Verweving en salvofouten ("interleaving" en "burst errors")

3.2.43. Onder verwijzing naar Figuur 1 van de 25.212 standaard wordt de stap na "radio frame equalization" aangeduid als "1st interleaving". Interleaving (vervlechting) is een proces dat gebruikt wordt om de robuustheid van de foutbescherming die door de kanaalcodeerinrichting verschaft wordt nog verder te verhogen. In de praktijk zijn storingen op het radiopad meestal niet beperkt tot één bit, maar wordt door zogeheten foutensalvo's of "burst errors" een groep van bits tegelijkertijd verstoord. Interleaving helpt bijvoorbeeld het effect van "burst errors" te reduceren. Interleaving staat bekend als een manier om de

invloed van een storing te verdelen over verschillende opeenvolgende bits van bijvoorbeeld een in de tijd variërend signaal.

3.2.44. Een burst error is dus een serie fouten die tegelijk gebeuren in een enkel salvo of "burst". Om een voorbeeld te geven: een burst error kan voorkomen als een mobiele telefoon wordt gestoord door een voorbijrijdende trein of een andere bijvoorbeeld periodiek werkende stroringsbron, in het bijzonder op een locatie met slechte ontvangst van het basisstation. Onder zulke omstandigheden kan de communicatie tussen de mobiele telefoon en het basisstation onderbroken worden met verscheidene bitfouten als gevolg van de onderbreking.

3.2.45. Figuur D hieronder laat het gevolg zien van een burst error.

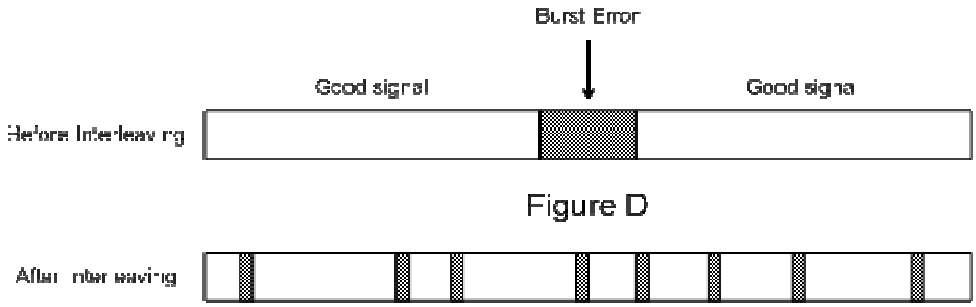


Figure D

Figure E

Zoals aangegeven in Figuur D, kan een periode waarin er een goed signaal is (dat wil zeggen een te corrigeren hoeveelheid transmissiefouten) onderbroken worden door een burst error waarin een groter aantal bitfouten optreedt. De door de kanaalcodeerinrichting gebruikte codes zijn dan niet in staat tot het corrigeren van een dergelijk groot aantal bitfouten. Deze codes zijn ontworpen om willekeurig verspreide fouten te corrigeren tussen voor het overige juiste data. Een krachtige en praktische manier om een sterke foutencorrectie te bereiken is (1) een geschikte foutencorrectiecode toe te passen (bijvoorbeeld een turbocode) en (2) een interleaver te gebruiken, om de fouten op een kanaal met onbekende of tijdsgevoelige foutendynamiek willekeurig te spreiden.

3.2.46. Hoewel de turbo codes en convolutionele codes die door de kanaalcodeerinrichting gebruikt worden superieur zijn aan de eenvoudige "herhaal iedere bit 3 keer"-code die hierboven besproken is, kan het probleem met burst errors aan de hand van die eenvoudige code geïllustreerd worden. In voorbeelden 1 en 2 hieronder wil een mobiele telefoon in beide gevallen de informatiebits "111" naar het basisstation verzenden en de mobiele telefoon verstuurt daartoe een gecodeerde boodschap bestaande uit negen keer "1" op een rij, dat wil zeggen drie groepen van elk drie bits. Veronderstel dat in beide voorbeelden 1 en 2 tijdens de transmissie door een storingsburst drie bitfouten optreden. In voorbeeld 1, waarin de bij een groep behorende bits direct achter elkaar worden verzonden, wordt door de burst de complete rechtergroep (onderstreept) van drie bits gestoord. Stel dat alle "1" van deze groep door de storing in een "0" worden veranderd. Het resultaat na decodering (meerderheidsbeslissing) is dat de rechterbit van de ontvangen informatiebits als een "0" wordt gedecodeerd. De burst error zorgt ervoor dat het basisstation een bit van het bericht fout decodeert. In het geval van voorbeeld 2 zijn de bits van de verschillende groepen voor

verzending door elkaar verspreid (“interleaved”). Bijvoorbeeld de eerste bit van de eerste groep (schuin gedrukt) van de te verzenden reeks is telkens de eerste bit van een gecodeerde groep; de tweede bit van de eerste groep (vet gedrukt) van de te verzenden reeks is telkens de tweede bit van een gecodeerde groep, etc. Wanneer nu door de storingsburst weer alle bits van de rechtergroep van de verzonden bits worden veranderd in een “0” is het basisstation in staat om, hoewel hetzelfde aantal fouten tijdens de verzending plaatsvindt, het hele bericht wel correct te decoderen:

Voorbeeld 1:	burst error
Oorspronkelijk bericht:	*1***1**1
Gecodeerd bericht:	*111* **111** 111
Ontvangen bericht:	111 111 000
Gedecodeerd bericht:	110
Voorbeeld 2:	verdeelde fouten (door interleaving)
Oorspronkelijk bericht:	*1***1**1
Gecodeerd bericht:	*1***1**1 *1***1**1 *1***1**1
Ontvangen bericht:	*1***1**1 *1***1**1 000
Onvlecht bericht:	0**1**1 *10*1 *1***1**0
Gedecodeerd bericht:	111

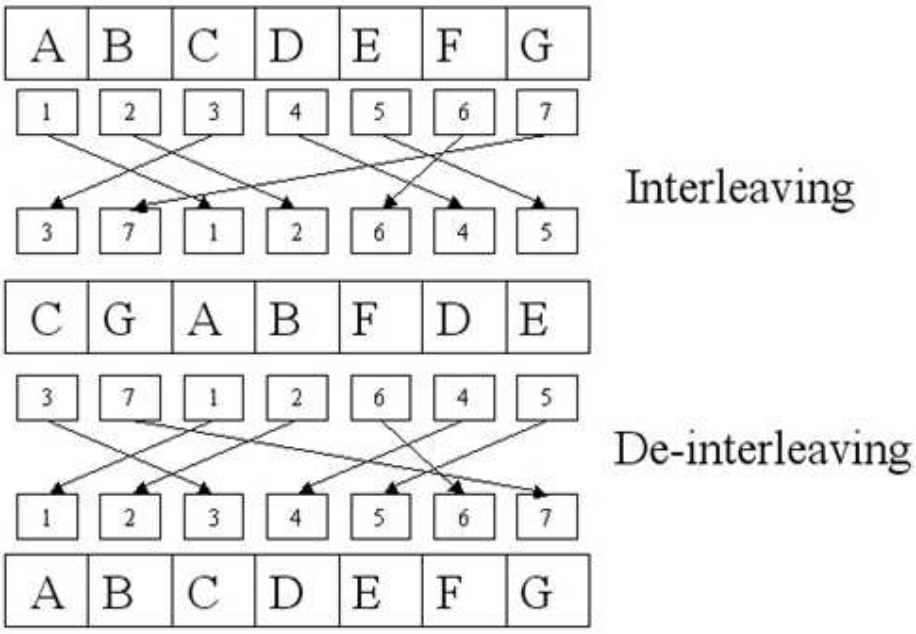
3.2.47. Zoals aangegeven in Figuur E, smeert interleaving het effect van de burst error in het verweven verzonden bericht uit, met het gevolg dat een lange burst error getransformeerd wordt in een serie korte fouten in het oorspronkelijke, niet verweven, bericht. De kanaalcodering is goed in het corrigeren van het soort fouten in de data van Figuur E, dat wil zeggen een klein aantal fouten die willekeurig verspreid zijn tussen juiste data.

3.2.48. Met andere woorden, het foutencorrectiedecodeerproces van de data in Figuur E, zal eerder de fouten herstellen dan het foutencorrectiedecodeerproces van de data in Figuur D. Interleaving op zichzelf corrigeert geen fouten. Maar als interleaving in samenhang met een codeerschema gebruikt wordt, zoals turbocoding, kan dit het gevolg van fouten door bijvoorbeeld burst errors verkleinen.

3.2.49. Zoals voorgesteld door Figuren D en E, is een interleaver een inrichting die de volgorde van de data (bits) verweeft of vervlecht. De mobiele telefoon verweeft (vervlecht) gegevens voorafgaande aan verzending naar een basisstation. Het basisstation ontweeft (ontvlecht) vervolgens de gegevens die het ontvangt van de mobiele telefoon en decodeert deze dan met een kanaaldecodeerinrichting die past bij de kanaalcodeerinrichting van de mobiele telefoon, om zo de oorspronkelijke datavolgorde te reconstrueren. Als een burst error tijdens een verzending plaatsvindt, zal het ontvlechten leiden tot het uitsmeren van de burst error zoals hierboven in Figuren D en E aangegeven, en aldoende de prestatie van de transmissie als geheel verbeteren.

3.2.50. De tekening hieronder laat het basisidee van “block interleaving” zien. In dit voorbeeld genereert de mobiele telefoon het oorspronkelijke bericht bestaande uit blokken bits “ABCDEFG.” Voor de verzending vervlecht de interleaver de data zodanig dat het bericht dat in feite naar het basisstation verstuurd wordt, de herschikte volgorde

"CGABFDE" vormt. Vervolgens ontvlecht het basisstation de ontvangen data om het oorspronkelijke bericht "ABCDEFG" te herstellen.



3.2.51. De eigenlijke interleavers die in draadloze telefoonsystemen gebruikt worden, werken op veel langere dataketens dan de volgorde die hierboven te zien is, maar zij maken gebruik van hetzelfde idee. Dat wil zeggen zij herschikken de volgorde van de data volgens een bekend en afgesproken algoritme zodat het basisstation het oorspronkelijke bericht kan herstellen door volgens het omgekeerde algoritme te ontvlechten.

Radioframesegmentatie ("radio frame segmentation") van de 25.212 standaard

3.2.52. Na de 1e interleaving komt het volgende blok in Figuur 1 van de 25.212 standaard radioframesegmentatie ("radio frame segmentation"). De radio frame segmentation-stap heeft betrekking op de voorgaande radio frame equalization-stap. Zoals hierboven beschreven, vormt de radio frame equalization-stap gedurende ieder TTI radioframes, die allemaal eenzelfde aantal bits bevatten. De 1e interleaving stap herschikt vervolgens de volgorde van al deze bits.

3.2.53. De radio frame segmentation-stap rangschikt de uitvoer van de interleaver weer in aparte radio frames, die allemaal dezelfde omvang hebben als gebruikt tijdens de radio frame equalization-stap. Het is onwaarschijnlijk dat een radio frame dat door de radio frame segmentation-stap gegenereerd wordt, identiek is aan een radioframe dat door de radio frame equalization-stap gegenereerd wordt (omdat de 1e interleaving de bits herschikt heeft). Deze twee radioframes zullen echter hetzelfde aantal bits bevatten.

3.2.54. Meer in het algemeen geldt voor ieder TTI dat als de radio frame equalization-stap "n" radioframes (van gelijke grootte) gevormd heeft, tijdens de radio frame segmentation-stap ook "n" radioframes (van gelijke grootte) gevormd zullen worden uit de data die door de interleaver gegenereerd worden.

Snelheidsaanpassing (“rate matching”) van de 25.212 standaard

3.2.55. Na de radio frame segmentation is de volgende stap in Figuur 1 van de 25.212 standaard “rate matching” (snelheidsaanpassing).

3.2.56. Voorafgaande aan de rate matching-stap genereert de mobiele telefoon gegevens, bijvoorbeeld radio frames, gebaseerd op inkomende informatie ontvangen door de telefoon. Een mobiele telefoon kan bijvoorbeeld meer data genereren wanneer de mobiele telefoongebruiker tijdens een gesprek ook een e-mail verzendt. De mobiele telefoon genereert deze data zonder de eigenschappen van de radioverbinding tussen de mobiele telefoon en het basisstation in acht te nemen. Het basisstation controleert deze kanaaleigenschappen echter wel en instrueert de mobiele telefoon hoeveel data verzonden moeten worden in ieder TTI afhankelijk van de radioverbinding. In de rate matching-stap past de mobiele telefoon de door dit apparaat gegenereerde data aan overeenkomstig de instructies die van het basisstation ontvangen zijn.

3.2.57. In een bepaald TTI kan een mobiele telefoon bijvoorbeeld acht radio frames van ieder 4.000 bits genereren, of een totaal van 32.000 bits aan data. Maar het basisstation kan de mobiele telefoon op dat moment toestaan slechts 30.000 bits aan data te verzenden gedurende ieder TTI. In dat geval zou de mobiele telefoon 2.000 van de in totaal 32.000 bits kwijt moeten zien te raken om alleen de overblijvende 30.000 bits te versturen. In een ander voorbeeld kan het basisstation op een gegeven moment verwachten gedurende ieder TTI 34.500 bits van de mobiele telefoon te kunnen ontvangen. In dat geval zou de mobiele telefoon 2.500 filler bits moeten toevoegen aan de oorspronkelijke data die hij gedurende het betreffende TTI wil verzenden.

3.2.58. De rate matching-stap is analoog aan het plannen van inkopen voordat er over een budget voor die inkopen wordt nagedacht. Voorafgaande aan het nadenken over het budget, kan iemand plannen twintig dingen te kopen. Maar na het budget nader bekeken te hebben, kan de betreffende persoon (moeten) besluiten slechts vijftien dingen te kopen. Onder verwijzing naar Figuur 1 van de 25.212 standaard zijn alle stappen voorafgaande aan de rate matching analoog aan het plannen van het doen van inkopen zonder na te denken over het budget – de mobiele telefoon genereert eenvoudigweg de data in respons op de ontvangen invoer. De rate matching-stap brengt het budget in het vizier. Als de mobiele telefoon meer bits gegenereerd heeft dan het basisstation op dat moment mag verwerken, schudt de mobiele telefoon bits af. Met andere woorden, de mobiele telefoon brengt de te verzenden data in overeenstemming met het “budget” dat door het basisstation vastgesteld is.

3.2.59. Het budget dat door het basisstation wordt vastgesteld kan in de loop van de tijd variëren. Factoren die het budget beïnvloeden, omvatten: (a) de afstand tussen de mobiele telefoon en het basisstation (mate van bitfouten neemt toe naarmate die afstand toeneemt omdat de ontvangen signaalsterkte afneemt), (b) obstructies tussen de mobiele telefoon en het basisstation (bijvoorbeeld een groot gebouw in de zichtlijn tussen de mobiele telefoon en basisstation of reflecties van objecten die signaalverstoringen veroozaken kunnen de mate van bitfouten vergroten), (c) aanwezigheid van blokkerende signalen, bijvoorbeeld van andere mobiele telefoons op hetzelfde basisstation, en (d) vele andere factoren.

3.2.60. Het kan een verrassing lijken dat de mobiele telefoon zich kan ontdoen van bits zonder de communicatie tussen de mobiele telefoon en het basisstation te verstoren. De

foutencorrectie die door de kanaalcodeerinrichting wordt geboden, kan echter een dergelijke verstoring compenseren. De in de kanaalcoderingsstap gebruikte turbocoder genereert systematische bits en twee keer zo veel pariteitbits. De systematische bits zijn identiek aan de oorspronkelijke informatiebits van het bericht dat de mobiele telefoon eigenlijk aan het basisstation wil overdragen. De pariteitbits zijn overtollige of redundante bits; hun doel is het compenseren voor fouten die gedurende de verzending optreden. Maar in het uiterste geval, als er geen verzendingsfouten zouden zijn, zouden alle pariteitbits in de rate matching-stap kunnen worden weggelaten en zou het basisstation nog steeds het betreffende bericht kunnen ontvangen, dat wil zeggen het zou nog steeds alle systematische bits ontvangen. Omdat er voor iedere systematische bit, twee pariteitbits gegenereerd worden, kan een groot aantal bits gedurende de rate matching step worden weggelaten zonder de communicatie tussen de mobiele telefoon en basisstation te belemmeren. Het weglaten van bits in de rate matching-stap reduceert weliswaar de effectiviteit van de kanaalcodeerinrichting, maar het systeem is zo ontworpen dat een aanzienlijk aantal bits tijdens de rate matching kan worden weggelaten zonder daarmee de betrouwbaarheid van de communicatie tussen de mobiele telefoon en het basisstation teveel schade te berokkenen, uiteraard afhankelijk van de storende omgeving.

3.2.61. Samenvattend, laat de mobiele telefoon tijdens de rate matching-stap bits weg, of voegt bits toe aan de gedurende de TTI gegenereerde radio frames, om er zeker van te zijn dat precies het aantal bits wordt verzonden dat het basisstation verwacht te ontvangen gedurende die TTI. “Puncteren” is een vakterm voor het weglaten van gecodeerde bits gedurende rate matching. Een tijdens de rate matching weggelaten bit wordt aangeduid als een “gepuncteerde” bit. Op dezelfde manier is “herhaling” een vakterm voor het toevoegen van bits gedurende de rate matching door bepaalde bits te repliceren. Een tijdens de rate matching toegevoegde bit wordt aangeduid als een “herhaalde” bit (dit is anders dan een hierboven beschreven filler bit; een herhaalde bit is namelijk een kopie van een gecodeerde bit).

3.2.62. Zoals hierboven beschreven, herschikt de mobiele telefoon de bits gedurende de 1e interleaving-stap en het basisstation herstelt de oorspronkelijke bits via een ontvlechtingsstap (de-interleaving). Rate matching gebruikt ook een overeenkomende stap aan het eind van het verzendproces bij de ontvanger. Gedurende ieder TTI gebruikt de mobiele telefoon een algoritme om te bepalen welke bits of gepuncteerd of herhaald moeten worden. Vervolgens gebruikt het basisstation een overeenkomstig algoritme om de locaties van de gepuncteerde of herhaalde bits te bepalen.

3.2.63. EP 528 heeft voornamelijk betrekking op de processtappen van 1e interleaving en rate matching.

Volgende stappen in de 25.212 standaard

3.2.64. Onder verwijzing naar Figuur 1 van de 25.212 standaard verzamelt de “TrCH Multiplexing” stap alle data van de rate matching-stappen in alle kanalen en schikt deze in een enkele seriële bitstroom.

3.2.65. De “Physical channel segmentation” stap creëert dan een bitstroom voor ieder fysiek kanaal. Een fysiek kanaal komt in het algemeen gesproken overeen met een bepaalde zend/ontvangfrequentie of een bepaalde zend/ontvangtijdsperiode; een mobiele telefoon met

verschillende frequenties of verschillende zend/ontvangtijdsperiodes beschikt over verschillende fysieke kanalen.

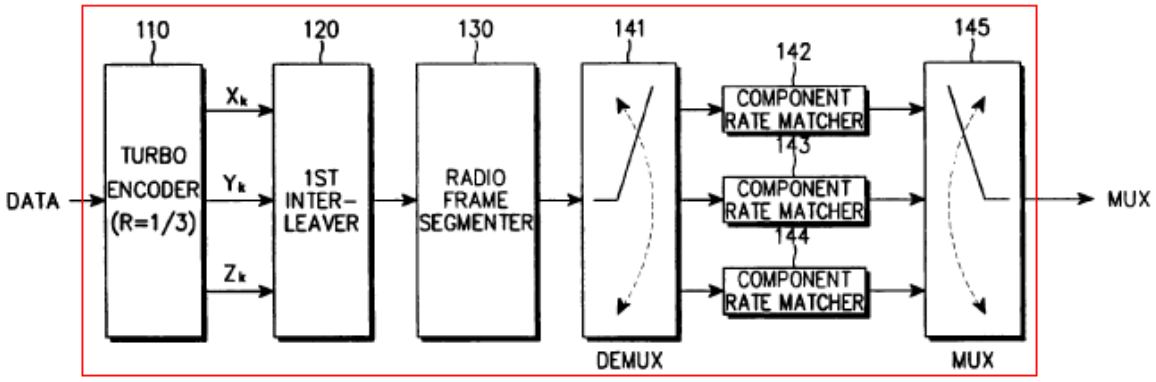
3.2.66. De “2e interleaving”-stap herschikt databits over verschillende kanalen. Zoals bij de “1e interleaving” gebruikt het basisstation een overeenkomstig ontvlechtingsalgoritme om de de oorspronkelijke data te herstellen zoals die door de de 2e interleaving stap geordend c.q. gespreid zijn.

3.2.67. Tenslotte maakt de “Physical channel mapping” stap de data in ieder fysiek kanaal klaar om door de lucht te worden verzonden.

EP 528

3.2.68. EP 528 betreft een uplink zendinrichting in een mobiel communicatiesysteem. De inrichting omvat een turbocoder (110), een (eerste) interleaver (120), een radio frame segmenter (130), een demultiplexer (141), een rate matcher (143, 144) en een multiplexer (145)[6]. Deze technische onderdelen kunnen zich bijvoorbeeld in een enkele UMTS-chip van een mobiele telefoon bevinden en hun onderlinge verhouding wordt het best geïllustreerd in figuur 2 van EP 528, dat een uplink zendinrichting volgens de uitvinding weergeeft:

FIG. 2

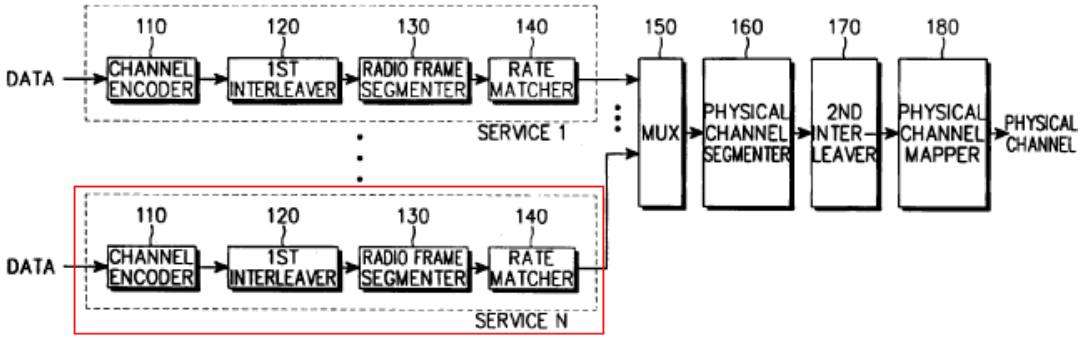


3.2.69. De in conclusie 1 gebruikte technische begrippen encoder (codeerinrichting) zoals een turbocoder, interleaver, radio frame segmenter en rate matcher zijn reeds in het hoofdstuk over de algemene technische achtergrond van UMTS technologie aan de orde gekomen. De begrippen demultiplexer en multiplexer worden in dit hoofdstuk uitgelegd.

3.2.70. Figuur 1 van EP 528 geeft volgens de beschrijving de stand van de techniek weer:

[6] De multiplexer is als zodanig niet in conclusie 1 opgenomen.

FIG. 1



3.2.71. De roodomlijnde delen van figuren 1 en 2 komen wat betreft het deel van de totale UMTS-tranmissiestructuur overeen (hetgeen in figuur 1 ook al inzichtelijk wordt gemaakt door de stippellijn). Uit die vergelijking volgt dat de uplink zendinrichting van conclusie 1 van EP 528 afwijkt van de stand van de techniek voor wat betreft het gebruik van een demultiplexer en drie separate rate matchingscomponenten (in plaats van één).

3.2.72. De geclaimde demultiplexer scheidt de bitstroom die uit de eerste interleaver (en radio frame segmenter) komt in drie stromen met respectievelijk systematische bits, eerste pariteitbits en tweede pariteitbits. Daarna wordt er op de drie stromen (separaat) rate matching uitgevoerd (bits worden al dan niet gepuncteerd of herhaald) en tenslotte worden de drie stromen door middel van een multiplexer weer bij elkaar gebracht tot één bitstroom.

3.2.73. EP 528 geeft aan dat het voordelig is om tijdens het rate matchingsproces uitsluitend pariteitbits te puncteren, en dus geen systematische bits (par. [0011] en [0012]). Dit is dan ook de reden om de bitstroom door middel van een demultiplexer ten behoeve van de rate matching in drie stromen te scheiden.

3.2.74. Partijen hebben conclusie 1 in de volgende afzonderlijke deelkenmerken opgedeeld (Apple heeft nog een verdere opsplitsing van deelkenmerk 6 aangehouden; Engelse termen zijn tussen aanhalingstekens toegevoegd):

1. Een uplink zendinrichting voor een mobiel communicatiesysteem, omvattende:

2. een codeerinrichting (110) (“turbo coder”)

   2.1. voor het ontvangen van een informatie bitstroom en

2.2. voor het afgeven van drie stromen door de informatiebitstroom te coderen[7], een stroom van systematische symbolen (X_k), een eerste pariteitsymboolstroom (Y_k) en een tweede pariteitsymboolstroom (Z_k);

3. een verwevingsinrichting (120) ("interleaver")

3.1. voor het met elkaar verweven van de drie stromen van gecodeerde symbolen door een vooraf bepaalde verwevingsregel en

3.2. het afgeven van een verwevingsstroom;

4. een radiorastersegmenteerinrichting (130) ("radio frame segmenter")

4.1. voor het ontvangen van de verwevingsstroom en

4.2. het afbeelden van de ontvangen stroom op ten minste één radioraster;

5. een demultiplexer (141); en

6. een snelheidsaanpassingsinrichting (143, 144) ("rate matcher")

6.1. ingericht voor het doorboren ("puncteren") van een deel van de eerste en tweede pariteitsymbolen

6.2. volgens een gegeven snelheidsaanpassingsregel.

3.2.75. Conclusie 1 heeft het over symboolstromen (systematische symbolen X_k, eerste pariteitsymbolen Y_k en tweede pariteitsymbolen Z_k, waarbij de index k het volgnummer van een betreffend symbool in een stroom aangeeft).

3.2.76. Deelkenmerken 5, 6, 6.1 en 6.2 van conclusie 1 betreffen de demultiplexer om de door de radio frame segmenter afgegeven bitstroom van systematische bits, eerste pariteitbits en tweede pariteitbits in stromen te scheiden (deelkenmerk 5) en het uitsluitend puncteren van (een deel van) de eerste en tweede pariteitbits tijdens de rate matchingsstap (deelkenmerk 6.1).

3.2.77. Om het door het octrooi geclaimde puncteren van uitsluitend pariteitbits te bewerkstelligen, dient tijdens rate matching een onderscheid te kunnen worden gemaakt tussen systematische bits enerzijds en pariteitbits anderzijds (eerste regel par. [0012]). EP 528 stelt dat dit in de stand van de techniek niet goed mogelijk was omdat de systematische bits en de pariteitbits tijdens het interleaven willekeurig met elkaar vervlochten werden (slot par. [0012]).

[7] Samsung heeft de woorden "door de informatiestroom te coderen" kennelijk abusievelijk weggelaten in de Nederlandse vertaling van conclusie 1 (par. 4.10 van de dagvaarding).

3.2.78. Om die reden stelt EP 528 een oplossing voor waarbij de interleaver de systematische bits en de pariteitbits volgens een “vooraf bepaalde verwevingsregel” met elkaar vervlecht (deelkenmerk 3.1) zodat de uitgaande (enkele) verwevingsstroom een bekend (en doorgaans vast) herhaalpatroon heeft, bijvoorbeeld eerst een systematische bit X_1, dan een eerste pariteitbit Y_5, daarna een tweede pariteitbit Z_3, dan weer een systematische bit X_5, een eerste pariteitbit Y_2, een tweede pariteitbit Z_1, etc. Het interleavingsalgoritme (de verwevingsregel) komt er dus op neer dat de informatiebits weliswaar vervlochten worden (van plaats verwisselen), maar dat het herhaalpatroon van de bits (bijvoorbeeld X,Y,Z,X,Y,Z etc) hetzelfde blijft, althans dat dit patroon achteraf vrij eenvoudig (door de demultiplexer) kan worden gereconstrueerd. Op die wijze kan met een simpele, op zichzelf voor een vakman bekende, demultiplexer de enkele stroom informatiebits die uit de interleaver komt in drie separate stromen van enkel systematische bits en enkel pariteitbits gescheiden worden, om daarna tijdens rate matching alleen in de twee stromen met pariteitbits te puncteren.

3.3. EP 528 - Inbreuk

3.3.1. De rechtbank kan Samsung’s stelling dat sprake is van een demultiplexer in de Intel/Infineon baseband chips niet volgen, waartoe als volgt wordt overwogen.

3.3.2. Niet in geschil is dat de geclaimde demultiplexer ertoe dient om data-input te scheiden in verschillende delen data-output, en meer specifiek dat de demultiplexer ertoe dient om een radioframe dat wordt geleverd door de radioframesegmenter, te scheiden in enerzijds systematische bits en anderzijds pariteitsbits. Apple, Samsung en de door hen ingeschakelde deskundigen hebben dat uitdrukkelijk zo betoogd.

3.3.3. Apple heeft naar het oordeel van de rechtbank terecht aangevoerd dat de Infineon/Intel chips geen demultiplexer in de hierboven bedoelde zin omvatten. Tussen partijen staat namelijk vast dat in die chips de systematische bits en pariteitsbits niet in fysiek gescheiden gegevensstromen worden geleverd aan de ratematcher. De ratematcher ontvangt één gegevensstroom die zowel de systematische bits als de pariteitsbits bevat.

3.3.4. Samsung heeft gesteld dat de gemiddelde vakman op de prioriteitsdatum zal onderkennen dat de scheiding die de demultiplexer uitvoert, “conceptueel” is bedoeld en dus net zo goed softwarematig kan worden uitgevoerd, zonder daadwerkelijke fysieke scheiding van de gegevensstromen. Die stelling moet worden verworpen. Na lezing van de octrooibeschrijving zal de gemiddelde vakman de conclusie trekken dat die demultiplexer de datastroom in drie afzonderlijke, fysiek te onderscheiden, datastromen opdeelt. Deze verschillende datastromen worden daarna volgens de uitvinding afzonderlijk aan de ratematchers aangeboden. Dat blijkt naar het oordeel van de rechtbank uit de volgende passages uit de beschrijving van het octrooi (vetdruk door de rechtbank, zie r.o. 2.4):

> ***A demultiplexer demultiplexes each of the radio frames received from the interleaver to the information symbols, the first parity symbols, and the second parity symbols of the radio frame.*** *Rate matchers bypass the information symbols and puncture or repeat the first and second parity symbols for rate matching.* (par. [0018], r. 25-27)

The DEMUX 141 separates the output symbols *of the radio frame segmenter 130 into information symbols and parity symbols* ***and switches them to the corresponding component rate matchers 142, 143, and 144****. The MUX 145 multiplexes symbols received from the component rate matchers 142, 143, and 144 and feeds the multiplexed symbols to the MUX 150 of FIG. 1.* (par. [0021], p. 4, r. 4-7)

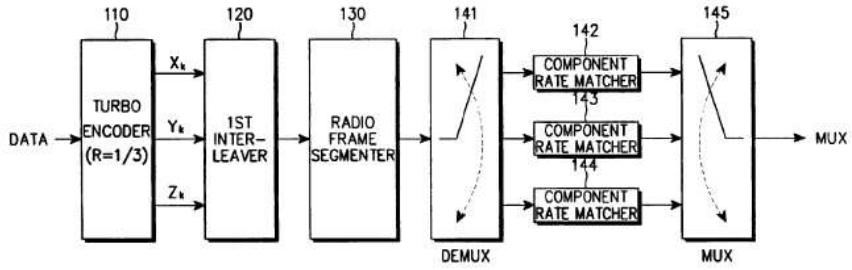
In the embodiment of the present invention shown in Fig. 2, ***the DEMUX 141 and MUX 145 are synchronized with each other such that the DEMUX 141 and MUX 145 switch to the same rate matcher block*** *(i.e., if DEMUX 141 switches to rate matcher 142 to input a symbol into the DEMUX 141, then MUX also switches to the rate matcher 142 after the input symbol has been rate matched to receive the rate matched symbol.).* (par. [0024], r. 17-20)

[0047] In the rate matching structure shown in FIG. 2, ***rate matching is implemented separately for each component rate machers****. The first, second, and third component rate matchers 142, 143, and 144 subject an information symbol x, a first parity symbol y, and a second parity symbol z, respectively, to rate matching. According to a given input and output sizes, each rate matcher performs puncturing/repetition on a predetermined number of symbols.* ***This rate matching structure is built on the assumption that the DEMUX 141 outputs x, y, z, separately****. Hence, the DEMUX 141 should be able to separate a radio frame received from the radio frame segmenter 130 into symbol x, y, z in a certain order.* (par. [0047], r. 5-10)

Referring to FIG. 2 again, ***the MUX 145 multiplexes three streams received from the component rate matchers 142, 143, and 144 to one stream****, to thereby generate a rate-matched radio frame with the same symbol pattern as before rate matching.* ***Because this MUX 145 is the counterpart of the DEMUX 141, it switches according to the same switching patterns****.* (par. [0083], r. 41-44)

3.3.5. Ook de figuren bij EP 528 dragen bij aan de conclusie van de gemiddelde vakman dat er een demultiplexer is bedoeld die een daadwerkelijke fysieke scheiding aanbrengt tussen de bitstromen. De fysieke scheiding wordt in figuur 2 treffend geïllustreerd met de op en neer slingerende haak die een verbinding maakt voor nu eens de bitstroom die wordt aangeboden aan ratematcher genummerd 142, dan weer de bitstroom die aan ratematcher genummerd 143 of tot slot 144 wordt aangeboden.

FIG. 2



Figuren 13, 14 en 15 van EP 528 laten hetzelfde zien (zie r.o. 2.5)

3.3.6. Met partijen uitgaande van de prioriteitsdatum als relevante peildatum, is daarnaast niet komen vast te staan dat de gemiddelde vakman destijds wist dat de scheiding van de gegevensstromen ook softwarematig kon worden uitgevoerd en dat de term demultiplexer tevens betrekking kon hebben op een dergelijke softwarematige uitvoering, waarbij dan voorts nog in feite helemaal geen fysieke scheiding van de datastromen meer plaats zou vinden maar.dit enkel nog conceptueel in de betreffende software terugkomt. Apple heeft die stelling van Samsung uitdrukkelijk weersproken onder verwijzing naar verklaringen van een door haar ingeschakelde deskundige (productie 51). Daar heeft Samsung onvoldoende tegen ingebracht. De door haar ingeschakelde deskundige Melham is juist op dit punt nogal terughoudend en overtuigt niet (productie 17 Samsung, nr. 6):

> The term "demultiplexer" appearing in the patent admits of being reasonably understood, in 1999, as potentially encompassing a software implementation, as well as a hardware one. More specifically, the DEMUX component is described in the patent almost exclusively in terms of its logical functioning - what it "does"- rather than how it is implemented - what it "is". Moreover, this logical functioning, as described in the patent, does not in principle exclude a software implementation. This is not to say that, at the time, it would have been taken for granted that a software implementation, with contemporary technology, would be practically viable, given the application's performance, and possibly power consumption, requirements. However, by 1999 it was well known that processor speed was increasing, and likely to continue increasing - so a future software implementation would not have been ruled out.

3.3.7. Samsung heeft erop gewezen dat het octrooi in paragraaf 12 slechts een “distinction” tussen information symbols en parity symbols vereist en dat niet belangrijk is hoe de demultiplexer volgens paragraaf 15 werkt “to separate symbol data into information symbols and parity symbols”. De gemiddelde vakman zal uit die paragrafen echter geenszins afleiden dat er geen fysieke scheiding behoeft plaats te vinden, zeker niet na lezing van het gehele octrooischrift inclusief de tekeningen.

3.3.8. De door Samsung gewilde lezing van het octrooi, waarbij geen fysieke scheiding tussen de systematische en pariteitsbits plaatsvindt maar deze slechts (conceptueel) van elkaar behoeven te worden onderscheiden, komt voorts in strijd met de redelijke rechtszekerheid. In het octrooi ontbreekt immers iedere hint dat dit slechts “conceptueel” zou moeten gebeuren en dat van daadwerkelijk “demultiplexen” geen sprake hoeft te zijn. Deze omissie klemt te meer als met Samsung, in weerwil van het hiervoor overwogene, zou worden aangenomen dat de gemiddelde vakman op de prioriteitsdatum al bekend was met softwarematige toepassing.

3.3.9. Het is na lezing van het octrooi zonneklaar dat ook de uitvinders een softwarematige toepassing niet voor ogen heeft gestaan, zodat een – tegen de redelijke rechtszekerheid af te zetten – redelijke bescherming voor de octrooihouder niet vereist dat deze thans toch daaronder zou moeten worden begrepen.

3.3.10. Het voorgaande betekent dat de toepassing van Intel/Infineon baseband chips geen inbreuk maakt op conclusie 1 van EP 528. Dit oordeel strookt met hetgeen het Landgericht Mannheim reeds heeft bevonden op 27 januari 2012 (zie r.o. 2.10). Zodoende komt de rechtbank niet toe aan de overige verweren van Apple, waaronder dat (conclusie 1 van) EP 528 ongeldig is.

3.4. EP 516 - Inleiding op de techniek

Shannonlimiet / Kanaalcapaciteit

3.4.1. Informatiesignalen kunnen worden verzonden via allerlei media. Verzending kan bijvoorbeeld plaatsvinden via media waarbij gebruik wordt gemaakt van kabels, zoals koperdraad of glasvezelkabels. Bij draadloze communicatiesystemen, waar het hier om gaat, is de lucht het medium. In de wereld van de communicatie wordt de term 'kanaal' gewoonlijk gebruikt als model hoe een signaal tijdens het verzenden ervan via een medium verzwakt wordt.

3.4.2. De lucht heeft de neiging een 'ruizig' kanaal voor te stellen, vooral in vergelijking met verzending via kabels. Ook in de diepe ruimte (bijvoorbeeld bij communicatie via satellieten of ruimtesondes) is er heel veel ruis.

3.4.3. Zo kan het gebeuren dat een door een radiozender via de lucht verzonden signaal dat een '0' moet vertegenwoordigen, tegen de tijd dat het door de ontvanger wordt ontvangen is veranderd in een '1' als gevolg van ruis in de lucht.

3.4.4. Zoals hiervoor reeds bij de inleiding op de techniek van EP 528 is opgemerkt, is één van de manieren waarop de betrouwbaarheid van een verzonden signaal kan worden verhoogd ondanks de aanwezigheid van ruis, de signaalsterkte te vergroten; hieraan kleven echter talloze nadelen, onder andere dat deze benadering niet bijzonder efficiënt is.

3.4.5. Het probleem van het op betrouwbare en efficiënte wijze verzenden van informatie via een kanaal vormt al decennialang onderwerp van onderzoek. De Amerikaanse wiskundige Claude Shannon was een pionier op dit gebied; hij wordt wel de vader van de informatietheorie genoemd.

3.4.6. In een baanbrekend artikel uit 1948 introduceerde Shannon het idee van **kanaalcapaciteit**, die tegenwoordig ook wel bekend staat als de **Shannonlimiet**; dit is de theoretische maximumsnelheid waarmee informatie op betrouwbare wijze kan worden verzonden via een bepaald kanaal. Shannon toonde aan dat de kanaalcapaciteit van een kanaal kan worden vastgesteld op basis van informatie ten aanzien van zijn bandbreedte en de ruiskarakteristieken. Shannon toonde verder aan dat zolang de feitelijke snelheid waarmee de informatie over het kanaal wordt verzonden lager is dan de kanaalcapaciteit, het mogelijk is informatie met een willekeurige lage foutensnelheid te versturen, ondanks de aanwezigheid van ruis – mits de informatie op correcte wijze met redundante informatie wordt gecodeerd.

3.4.7. Wat met name van belang is, is dat Shannon aantoonde dat er voor elk communicatiekanaal een foutcorrectiecode moet zijn die het mogelijk maakt op betrouwbare wijze informatie te verzenden die de kanaalcapaciteit zo dicht mogelijk benadert. Shannon gaf echter niet aan hoe praktische voorbeelden van dergelijke codes geconstrueerd moesten worden. Op het moment van publicatie van het artikel van Shannon bestonden er nog geen praktische codes die prestaties die de kanaalcapaciteit benaderden mogelijk maakten.

3.4.8. De inspanningen van Shannon hebben geleid tot een fundamentele verandering in het denken van mensen die werkzaam zijn op het gebied van communicatie – men had zich

neergelegd bij de overtuiging dat sommige kanalen ofwel te veel ruis bevatten om informatie veilig over te brengen ofwel steeds meer energie vereisten om de betrouwbaarheid van de verzendingen te vergroten. Na het baanbrekende werk van Shannon begon de zoektocht naar codes die de Shannonlimiet benaderden. In de decennia die volgden ontwikkelden onderzoekers een reeks codes die de Shannonlimiet steeds dichter naderden, maar tegen de jaren tachtig leek er nauwelijks nog vooruitgang te worden geboekt en bleef er een grote kloof bestaan tot de theoretische capaciteit. Het duurde nog tot het begin van de jaren negentig, voordat een groep Franse onderzoekers met turbocodes op de proppen kwamen, die de Shannonlimiet dicht benaderden.

Foutcorrectiecodes

3.4.9. Foutcorrectiecodes worden toegepast om de nauwkeurigheid van informatieverzendingen via een 'ruizig' kanaal te vergroten. Dergelijke codes coderen informatie zodanig dat deze door de ontvanger ondanks het bestaan van ruis in het kanaal op betrouwbare kan worden ontvangen.

3.4.10. Foutcorrectiecodes werken door het toevoegen van redundante informatie, zodat de ontvanger in staat wordt gesteld fouten te ontdekken en te corrigeren. De redundante informatie wordt toegevoegd in een **codeerder** voor de code.

3.4.11. De codeerder accepteert een informatiesequentie als invoer ('input') en produceert een **codewoord** als uitvoer ('output'). Omdat foutcorrectiecodes redundante informatie toevoegen, is het codewoord altijd langer dan de oorspronkelijke informatie. Alle codewoorden tezamen vormen de **code**, of het codeboek.

3.4.12. Gewoonlijk wordt een codeerder omschreven als het accepteren van ingevoerde informatie met een lengte K en het produceren van codewoorden met een lengte N, waarbij N groter is dan K. De **codesnelheid** van een code wordt gegeven door K/N. Zo heeft een code die 10-bit informatiesequenties codeert als 30-bit codewoorden een codesnelheid van 1/3.

3.4.13. De waarden K en N verschaffen daarnaast informatie over het aantal codewoorden in de code en over hoeveel N-bit-sequenties geen codewoorden zijn.

3.4.14. Als K de lengte van de informatiesequentie is en als wordt aangenomen dat de invoer binair is, dan zijn er 2^K verschillende invoeren mogelijk. Als K = 10, dan zijn er bijvoorbeeld 2^{10}, oftewel 1024, mogelijke invoerpatronen. Dat houdt in dat een codeerder die 10-bit invoeren accepteert, 1024 verschillende codewoorden zal genereren, omdat de invoerinformatie en de codewoorden van de code één-op-één overeen moeten komen (een bepaalde informatiesequentie wordt gecodeerd tot slechts één codewoord, en een bepaald codewoord kan uitsluitend worden gedecodeerd tot één informatiesequentie). Dit is ongeacht de lengte van de codewoorden. Als de lengte van de codewoorden N is, dan is het aantal N-bit sequenties dat *geen* codewoord is het resultaat van 2^N-2^K. Als N = 30, dan zijn er 2^{30}, oftewel circa 1,07 miljard, mogelijke sequenties van N, maar slechts 1024 van deze sequenties zouden geldige codewoorden zijn – de overige zouden ongeldige sequenties zijn.

3.4.15. Het hiervoor besproken voorbeeld toont een herhalingscode die een enkele bit drie keer repliceert voor verzending, waarbij K is = 1 en N is = 3. De codesnelheid is derhalve

1/3 en er zijn twee codewoorden (‘000’ en ‘111’) op acht mogelijke 3-bit-sequenties. Als de ontvanger een van de zes andere 3-bit-sequenties, die geen codewoorden zijn, ontvangt, dan weet deze dat het signaal is verstoord.

3.4.16. De herhalingscode is uiteraard een triviaal voorbeeld. In werkelijkheid zouden de K- en N-waarden groter zijn. Zo is de waarde K in de UMTS-turbocode variabel en kan deze bestaan uit 40 tot 5114 bits, waarbij N dienovereenkomstig bestaat uit 120 tot 15342 bits.

De ‘Afstand’- en ‘Gewicht’-eigenschappen van foutcorrectiecodes

3.4.17. Van verder belang voor foutcorrectiecodes zijn de concepten **Hammingafstand** en **Hamminggewicht.** Deze concepten zijn vernoemd naar de Amerikaanse wiskundige Richard Hamming, die in de jaren vijftig baanbrekend werk heeft verricht op het gebied van foutcorrectiecodes; er wordt ook vaak simpelweg naar verwezen als afstand en gewicht.

3.4.18. Afstand heeft betrekking op het aantal posities waarop twee bit-sequenties van elkaar verschillen. De sequenties ‘11111111’ en ‘10101010’ verschillen bijvoorbeeld op 4 plaatsen; dat houdt in dat hun afstand 4 is. Dit kan worden beschouwd als een maat om het verschil of de scheiding tussen twee bit-sequenties aan te geven.

3.4.19. Gewicht heeft betrekking op het aantal bits zonder nul in een bit-sequentie. Het gewicht van ‘11111111’ is bijvoorbeeld 8 en van ‘10101010’ is dat 4.

3.4.20. De **minimumafstand** van een code heeft betrekking op de kleinste afstand die bestaat tussen twee willekeurige codewoorden ervan. Voor een code met een verzameling van 1024 codewoorden verwijst de minimumafstand van de code bijvoorbeeld naar de kleinste afstand die twee van de 1024 codewoorden van elkaar scheidt.

3.4.21. In het algemeen geldt dat hoe groter de minimumafstand is, hoe beter de code in staat is tot foutdetectie en -correctie (hoewel er ook andere factoren zijn die de prestaties beïnvloeden). Dit is gevoelsmatig ook logisch, aangezien een grotere minimumafstand inhoudt dat codewoorden gemakkelijker van elkaar onderscheiden worden en er dus minder kans bestaat dat de ontvanger codewoorden met elkaar verwart. Als de minimumafstand van een code bijvoorbeeld 4 is tussen de codewoorden ‘10101010’ en ‘11111111,’ dan zou deze code een grotere mate van foutdetectie- en -correctie-eigenschappen bezitten dan een code, met bijvoorbeeld de codewoorden ‘10101010’ en ‘1010101**1**’, waarvan de minimumafstand 1 is.

3.4.22. Zoals eerder opgemerkt, is de minimumafstand van een code de minimumafstand tussen twee van zijn codewoorden. Een manier waarop de minimumafstand van een code zou kunnen worden vastgesteld, is het genereren van alle codewoorden van de code om vervolgens die twee te vinden die het dichtst bij elkaar liggen. Afhankelijk van de lengte van de codeerderinvoer en -uitvoer zou dit echter een tijdrovende bezigheid worden die veel rekenkracht vergt. Dit komt omdat zelfs codeerders die invoersequenties van gemiddelde lengte verwerken een gigantische verzameling van codewoorden kunnen produceren. Zo kan de lengte van de invoersequenties die worden verwerkt door de UMTS turbocoder, zoals reeds eerder opgemerkt, variëren van 40 tot 5114 bits. Zelfs in het geval van een invoer met een minimumlengte van 40 bits heeft de code al 2^{40}, oftewel ca. 1,10 biljoen,

codewoorden. Het is ondoenlijk om van een dergelijke code alle codewoorden te genereren en te vergelijken.

3.4.23. Gelukkig bestaan er snellere manieren voor het analyseren van het minimumgewicht van codes die **lineaire codes** zijn. Een lineaire code is een code waarin elk van zijn codewoorden hetzelfde aantal 'buren' heeft (dat wil zeggen, andere codewoorden), op dezelfde afstand. Als '11111100' bijvoorbeeld een codewoord is en zijn dichtstbijzijnde buur is '11110000', op een afstand van 2, dan betekent dit dat elk ander codewoord van die code een buur heeft die op dezelfde afstand daarvan staat.

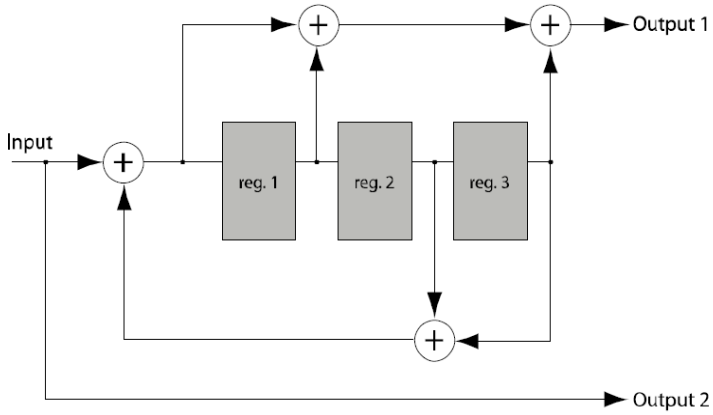
3.4.24. Een andere eigenschap van lineaire codes is dat de uitvoersequentie die uit allemaal nullen bestaat (bijvoorbeeld '00000000') altijd een codewoord is.

3.4.25. Hieruit volgt dat er voor een lineaire code een codewoord bestaat dat op de minimumafstand van het allemaal-nullencodewoord is gescheiden. Een codewoord dat op de minimumafstand ligt van het allemaal-nullencodewoord is uiteraard het codewoord dat het **minimumgewicht** heeft van alle geldige codewoorden van de code (met uitzondering van het allemaal-nullencodewoord). Het is derhalve mogelijk de minimumafstand van een code vast te stellen door het codewoord te vinden dat het minimumgewicht heeft. Dit is veel praktischer dan het berekenen en vergelijken van de gehele verzameling codewoorden. Mensen die zich bezighouden met coderen verwijzen bij het analyseren of ontwikkelen van codes derhalve gewoonlijk naar het minimumgewicht van een code, en in algemene zin is het wenselijk om het minimumgewicht van een code te verhogen.

Recursieve Convolutionele Codes

3.4.26. Een recursieve convolutionele code is een soort code waarvan de codeerder **geheugen** en **terugkoppeling** gebruikt voor het genereren van codewoorden. Dergelijke codes hebben interessante eigenschappen die in het onderhavige geval van belang zijn.

3.4.27. Hieronder wordt een voorbeeld gegeven van een simpele recursieve convolutionele codeerder:



3.4.28. De hiervoor afgebeelde recursieve convolutionele codeerder accepteert bij de Input een bitstroom, en produceert voor elke ingevoerde bit twee uitvoerbits. De twee uitvoerstromen worden aaneengekoppeld om het gehele codewoord te maken. Zichtbaar is

dat de Input rechtstreeks naar Output 2 wordt gevoerd; dit wordt een ‘systematische’ uitvoer genoemd, zoals hiervoor reeds is besproken. Output 1 wordt gegenereerd door de werking van de andere componenten van de codeerder.

3.4.29. De recursieve convolutionele codeerder bevat een verschuivingsregister met drie geheugenelementen (afgebeeld als grijze vakken). Elk van deze geheugenelementen kan een ‘0’ of een ‘1’ opslaan. De opgeslagen waarden worden bij elk verstrijken van een tijdseenheid ‘verschoven’ van het ene register naar het volgende. Als er bijvoorbeeld een ‘1’ is opgeslagen in register 1, dan wordt deze verschoven en opgeslagen in register 2 en vervolgens in register 3. De verschuivingsregisters zorgen er voor dat de codeerder de waarden van de drie voorgaande inkomende signalen kan opslaan. De codeerder begint in de ‘allemaal-nullenstand,’ waarbij zijn geheugenelementen allemaal nullen opslaat.

3.4.30. De in het diagram weergegeven ‘+’ symbolen staan voor een speciaal soort binaire opteller die gelijk is aan een ‘exclusieve-OF’ (XOR)-functie. Zoals te zien is in het diagram, bevat de recursieve convolutionele codeerder meerdere optellers, voor het optellen van verschillende waarden, inclusief de ingevoerde bits en waarden die in de registers zijn opgeslagen. In de volgende tabel wordt het gedrag van de binaire opteller/XOR-functie beschreven. Invoer A en Invoer B hebben betrekking op de twee waarden die opgeteld moeten worden, en de uitvoer representeert het resultaat van die optelling. In het hierboven getoonde diagram van de recursieve convolutionele codeerder, waarin twee pijlen samenkomen bij een + teken, vertegenwoordigt iedere pijl een invoerwaarde (A of B):

Invoer A	Invoer B	Uitvoer
0	0	0
0	1	1
1	0	1
1	1	0

3.4.31. De hierboven afgebeelde recursieve convolutionele codeerder bevat terugkoppeling. Deze is met name zodanig ingericht om de in register 2 en 3 opgeslagen waarden op te tellen en de som daarvan terug te koppelen naar de invoer die moet worden opgeteld bij het inkomende invoerbit. (De verplaatsingsregisters slaan dus in feite vorige waarden op van de som van de invoerbit en de bits in registers 2 en 3.)

3.4.32. Uit het voorgaande kan worden afgeleid dat de bit in Output 1 van de afgebeelde recursieve convolutionele codeerder wordt geproduceerd door de invoerbit op te tellen bij inhoud van geheugenelementen. Dit houdt in dat een recursieve convolutionele codeerder uitvoeren genereert die niet alleen afhankelijk zijn van de waarde van de huidige invoer, maar tevens van de waarden van de voorgaande invoeren.

3.4.33. Recursieve convolutionele codeerders kunnen op andere wijze worden geconstrueerd dan de codeerder die in bovenstaand diagram als voorbeeld is genomen. Zo kan een recursieve convolutionele codeerder een afwijkend aantal geheugenelementen bevatten en kunnen de optellers en terugkoppelroutes ervan op een andere manier zijn ingericht.

3.4.34. Het gebruik van geheugen en terugkoppeling in een recursieve convolutionele codeerder leidt tot een aantal interessante eigenschappen.

3.4.35. Een van die eigenschappen is dat als de codeerder invoer ontvangt die uit allemaal nullen bestaat, deze een allemaal-nullencodewoord zal genereren. Dit komt omdat de geheugenelementen in het begin allemaal in de allemaal-nullenstand staan en als er nooit ‘1’-en binnenkomen, zal de uitvoer ook nooit een ‘1’ bevatten.

3.4.36. Een andere belangwekkende eigenschap is dat als de codeerder als invoer een enkele ‘1’ ontvangt, gevolgd door een onbepaalde stroom ‘0’-en, de codeerder op Output 1 een uitvoerstroom genereert die uit meerdere ‘1’-en bestaat. Een algemene verklaring waarom dit gebeurt, is dat een enkele ‘1’ die verschijnt bij de invoer tot gevolg heeft dat ‘1’-en worden opgeslagen in de geheugenelementen van de codeerder en de geheugenelementen herhaaldelijk worden teruggekoppeld naar de invoer. Een explicieter voorbeeld van dit verschijnsel wordt getoond in de onderstaande tabel voor de hierboven afgebeelde codeerder (in de tabel, geeft de ‘toestand’ de inhoud weer van de drie geheugenelementen van de codeerder):

Tijdstip	Invoer	Huidige toestand	Uitvoer 1	Volgende toestand
0	1	000	1	100
1	0	**100**	1	010
2	0	010	1	101
3	0	101	1	110
4	0	110	0	111
5	0	111	0	011
6	0	011	1	001
7	0	001	0	100
8	0	**100**	1	010
9	0	010	1	101

3.4.37. Zoals bovenstaande tabel laat zien, zal de codeerder, als deze een ‘1’ (getoond op tijdstip 0) ontvangt, ‘1’ genereren aan de uitvoerkant op de volgende tijdstippen, zelfs wanneer alle van de opeenvolgende invoerbits‘0’-en zijn. Daarnaast laten de gemarkeerde rijen (die tijdstip 1 en 8 weergeven) zien dat de interne toestand van de codeerder wordt herhaald, terwijl deze verder allemaal ‘0’-en ontvangt, terwijl deze nooit terugkeert naar de allemaal-nullenstand. Hieruit volgt dat als de codeerder een onveranderlijke stroom ‘0’-en ontvangt na de enkele ‘1,’ deze door zal gaan met het genereren van ‘1’-en bij de uitvoer. (Dit kenmerk van de codeerder wordt ‘infinite impulse response’ genoemd). Om deze reden is een recursieve convolutionele codeerder vaak in staat om zelfs uit laag-gewicht invoer codewoorden met een relatief hoog gewicht te genereren.

3.4.38. Niet alle invoersequenties zorgen er echter voor dat een recursieve convolutionele codeerder hoog-gewicht codewoorden uitvoert. Met name bepaalde gewicht-2-invoerpatronen zorgen ervoor dat een dergelijke codeerder een laag-gewichtuitvoer genereert. In de hierboven afgebeelde codeerder gebeurt dit voor een invoersequentie die bestaat uit twee ‘1’-en die zijn gescheiden door zeven bitposities (bijvoorbeeld ‘10000001’). Dit wordt zichtbaar gemaakt in de volgende transitietabel:

Tijdstip	Invoer	Huidige Toestand	Uitvoer 1	Volgende Toestand
0	1	000	1	100
1	0	100	1	010

2	0	010	1	101
3	0	101	1	110
4	0	110	0	111
5	0	111	0	011
6	0	011	1	001
7	1	001	1	000
8	0	000	0	000

3.4.39. De tabel hierboven laat zien dat als de codeerder zeven bitposities na de eerste ‘1’ een tweede ‘1’ ontvangt, de codeerder terugkeert naar de allemaal-nullenstand.[8] Dit houdt in dat als de codeerder na dit punt een onveranderlijke stroom ‘0-en’ ontvangt, deze voor de opeenvolgende uitvoeren uitsluitend ‘0’-en zal genereren – in contrast met de hierboven besproken situatie waarbij de codeerder een enkele ‘1’ ontvangt, gevolgd door allemaal ‘0’-en. Uitgaand van onbepaald lange invoerstromen zou een patroon als dit derhalve resulteren in een codewoord met een veel lager gewicht dan de hierboven beschreven gewicht-1-sequentie.

3.4.40. Het is belangrijk op te merken dat dit fenomeen zich niet voordoet bij alle gewicht-2-sequenties. De situatie doet zich bijvoorbeeld niet voor bij de hiervoor beschreven codeerder als de twee ‘1’-en worden gescheiden door acht bitposities. Zoals onderstaande tabel van toestandsovergangen laat zien, veroorzaakt de tweede ‘1’ in dit geval niet dat de codeerder terugkeert naar de allemaal-nullenstand. NB de toestand op tijdstip 9 (nadat de tweede ‘1’ op tijdstip 8 is ontvangen) is gelijk aan de toestand op tijdstip 4, wat inhoudt dat als de codeerder allemaal ‘0’-en achtereenvolgens ontvangt, deze terugkeert naar het repeterende patroon van toestanden zonder terug te keren naar de allemaal-nullenstand:

Tijdstip	Invoer	Huidige Toestand	Uitvoer 2	Volgende Toestand
0	1	000	1	100
1	0	**100**	1	010
2	0	010	1	101

[8] Zoals te zien is in de tabel, is de interne status van de recursieve convolutionele codeerinrichting 001 bij tijd 7, hetgeen betekent dat de drie geheugenelementen daarvan respectievelijk “0,” “0,” en “1” opslaan. Als er op dat moment een “1” ontvangen wordt bij de invoer gebeurt het volgende in de codeerinrichting:
(1) De meest linkse toevoeginrichting (+ teken) telt de waarden van de huidige invoerbit en de som van de waarden die in geheugenelement 2 en geheugenelement 3 zijn opgeslagen bij elkaar op. In het gegeven voorbeeld zijn deze waarden “1” (huidige invoerbit), “0” (waarde opgeslagen in geheugenelement 2) en “1” (waarde opgeslagen in geheugenelement 3). De som van de waarden van de twee geheugenelementen is “1” (0+1=1). Als deze som wordt opgeteld bij de invoer levert dit “0” (1+1=0) op.
(2) De waarden van de drie geheugenelementen worden naar rechts geschoven. Zodoende wordt de “0” opgeslagen in geheugenelement 1 verschoven naar geheugenelement 2, en de “0” opgeslagen in geheugenelement 2 wordt verschoven naar geheugenelement 3. Van de inhoud van geheugenelement 3 (“1”) wordt afstand gedaan. De som die door de meest linkse toevoeginrichting (“0”) geproduceerd wordt, wordt opgeslagen in geheugenelement 1. De nieuwe inhoud van de drie geheugenelementen wordt dan “0”, “0” en “0” — de algehele nulstand.

3	0	101	1	110
4	0	110	0	111
5	0	111	0	011
6	0	011	1	001
7	0	001	0	100
8	1	100	0	110
9	0	**110**	0	111
10	0	111	0	011

3.4.41. Met andere woorden, het probleem waarbij gewicht-2-sequenties leiden tot laag-gewichtcodewoorden doet zich alleen voor als de twee '1'-en worden gescheiden door een specifiek aantal posities. Het specifieke patroon dat problematisch is, verschilt afhankelijk van het ontwerp van de codeerder.

3.4.42. Het concept van minimumafstand met betrekking tot recursieve convolutionele codes en turbocodes wordt vaak ook wel '**vrije afstand**' genoemd.

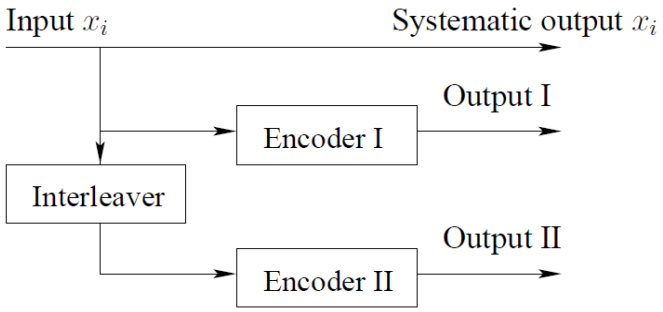
Turbocodes

3.4.43. Turbocodes zijn een soort uitstekend presterende foutcorrectiecodes.

3.4.44. Turbocodes zijn voor het eerst besproken in een artikel uit 1993 getiteld "*Near Shannon limit Error-Correcting Coding and Decoding: Turbo-Codes*" van Claude Berrou en andere onderzoekers in Frankrijk (productie 38 Apple).

3.4.45. De invoering van turbocodes veroorzaakte een sensatie in de coderingsgemeenschap omdat de prestaties van dergelijke codes – onder bepaalde omstandigheden – dicht in de buurt van de Shannonlimiet komen, veel dichter dan andere destijds bekende codes. Dit houdt in dat een zender door het gebruiken van turbocodes in staat is een gegeven bitfoutensnelheid te behalen bij een ongeëvenaard lage bitenergie-ruisverhouding. Dat wil zeggen dat er dankzij turbocodes bij een gegeven niveau van kanaalruis aanzienlijk minder transmissievermogen nodig zou zijn – in de buurt van het theoretisch minimum dat Shannon wiskundig aantoonde – terwijl de zender nog steeds nagenoeg foutloos werkt.

3.4.46. In zijn algemeenheid bevat een turbocoder, zoals beschreven door Berrou in 1993, meervoudige, parallel met elkaar verbonden **constituente codeerders** en een **interleaver**. Onderstaand diagram is een afbeelding van een generieke turbocoder:



3.4.47. Zoals het diagram laat zien, accepteert een turbocoder een sequentie van bits als invoer en genereert deze daaruit drie signalen: één uitvoer van elk van de constituente codeerders en een systematische uitvoer, die niet meer is dan een kopie van de invoersequentie. Deze drie uitvoeren worden aaneengeschakeld om één enkel codewoord te genereren voor elke invoersequentie. Het valt dus te begrijpen dat het gewicht van een door de turbocoder geproduceerd codewoord de som vormt van de gewichten van de invoer en de uitvoer van de twee constituente codeerders.

3.4.48. Deze basisstructuur van een turbocoder is beschreven in het oorspronkelijke artikel uit 1993. De volgende figuur is afkomstig uit datzelfde artikel:

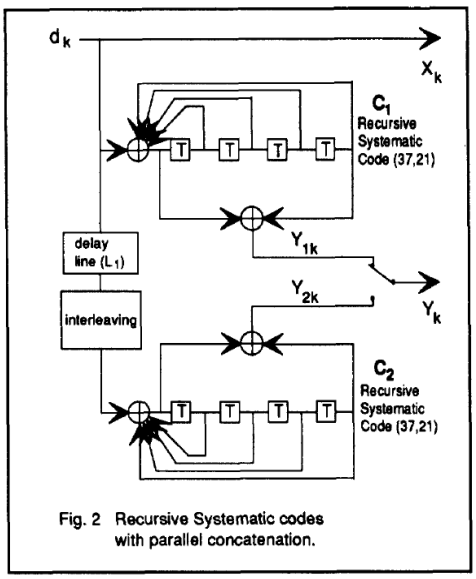


Fig. 2 Recursive Systematic codes with parallel concatenation.

3.4.49. In zijn artikel uit 1993 beschrijft Berrou het gebruik van recursieve convolutionele codes (in bovenstaand diagram aangeduid als C_1 en C_2) als constituente codeerders binnen de turbocoder. (Hoewel de constituente codeerders die staan afgebeeld in het artikel van Berrou een structuur hebben die enigszins afwijkt van de recursieve convolutionele codeerder die hierboven is afgebeeld in par. 3.5.27, zijn het evengoed recursieve convolutionele codeerders aangezien deze ook gebruikmaken van geheugen en terugkoppeling om uitvoerbits te genereren.)

3.4.50. Zoals de diagrammen van de algemene turbocoder en de turbocoder uit het artikel van Berrou ook laten zien, worden de invoerbits rechtstreeks toegevoegd aan de eerste constituente codeerder, maar passeren zij voordat zij worden toegevoegd aan de tweede codeerder eerst een (interne) **interleaver**.[9]

3.4.51. Zoals hiervoor beschreven, permuteert, of herschikt, een interleaver de volgorde van de bits binnen een sequentie van bits. Op dezelfde wijze als waarop hij wordt gebruikt in een turbocoder, permuteert de interleaver derhalve de invoerbits voordat zij worden verwerkt door de tweede constituente codeerder, op zodanige wijze dat de eerste en tweede constituent codeerders verschillende invoerpatronen verwerken.

3.4.52. Een van de basisgedachten achter turbocodes is dat als gevolg van het 'interleaving'-proces de **minimumafstand** tussen codewoorden van de turbocode toeneemt. Dit komt omdat zelfs als een specifiek ingevoerd bitpatroon een problematisch patroon is dat resulteert in een laag-gewichtuitvoer geproduceerd door de eerste constituente codeerder (zoals hiervoor beschreven), de interleaver het patroon idealiter zou herschikken tot een patroon dat niet zou resulteren in de tweede constituente codeerder die ook een laag-gewichtuitvoer produceert. Als gevolg daarvan genereert de algehele turbocoder codewoorden met een hoger gewicht vergeleken met een situatie waarin geen interleaver wordt gebruikt.

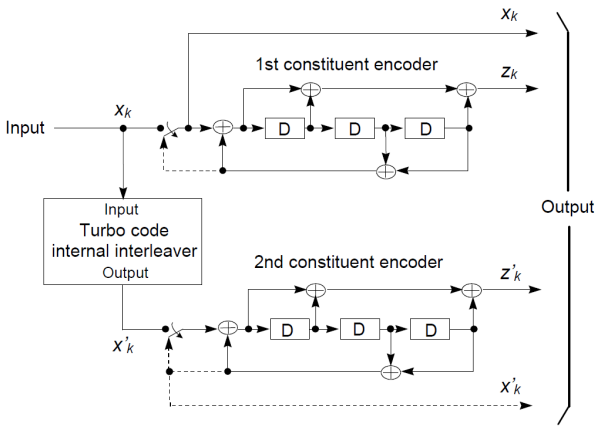
3.4.53. Met andere woorden, de gedachte is dat de interleaver ervoor zorgt dat ten minste een van de constituente codeerders een hoog-gewichtbijdrage aan het codewoord produceert. Als beide constituent codeerders laag-gewichtuitvoeren produceren, zou het codewoord van de algehele turbocoder ook laag-gewicht zijn, hetgeen ongewenst is.

3.4.54. Overigens moet de invoersequentie door de interleaver op reproduceerbare wijze worden herschikt, omdat de ontvanger, bij zijn poging de boodschap te decoderen, het proces zal moeten omkeren. Interleavers kunnen de bits dan ook niet op volstrekt willekeurige wijze herschikken, maar moeten goed-gedefinieerde processen uitvoeren voor het herschikken van een bitpatroon. Er zijn verschillende soorten interleavers bekend in de stand der techniek.

<u>De UMTS Turbocoder en Interne Interleaver</u>

3.4.55. Samsung baseert haar inbreukvordering op versie 3.11.0 van de 25.212-standaard (productie 2 Samsung). Die standaardversie beschrijft de volgende turbocoder voor gebruik in UMTS:

[9] Voor alle duidelijkheid: de <u>interne</u> interleaver van de turbo coder (hetgeen onderdeel is van de kanaalcodering) wordt onderscheiden van andere interleavingstappen die naast kanaalcodering gekozen kunnen worden. EP '528 betreft bijvoorbeeld onder meer het interleaven van een bitvolgorde door de <u>eerste</u> interleaver *nadat* deze al door codeerproces (turbo coder) is gegaan.



3.4.56. De UMTS-turbocoder heeft dezelfde algemene structuur als de turbocoder die is geopenbaard door Berrou in zijn oorspronkelijke artikel uit 1993. Dat wil zeggen, hij bevat twee recursieve convolutionele codeerders als constituente codeerders en een interne interleaver, waarbij de eerste codeerder de oorspronkelijke invoersequentie verwerkt en de tweede codeerder een gepermuteerde (dat wil zeggen door de interleaver opnieuw gerangschikte) versie van de invoersequentie.

3.4.57. De constituente codeerders van de UMTS-turbocoder hebben dezelfde structuur als de recursieve convolutionele codeerder afgebeeld in par. 3.4.27 en zoals besproken in de paragrafen daarna.

3.4.58. Standaard TS 25.212 beschrijft, algemeen gezegd, een tweedimensionale interleaver die een uitgebreide reeks stappen uitvoert teneinde het gewenste permutatie-effect te bereiken. Allereerst worden de invoerinformatiebits rijgewijs opeenvolgend geschreven naar een tweedimensionale matrix. Het aantal ingevoerde bits is variabel, zodat ook de grootte van de matrix variabel is. Vervolgens voert de interleaver intrarij permutaties uit, dat wil zeggen dat de bits van elke rij binnen de rij worden herschikt. Na de intrarij-permutaties voert de interleaver interrij-permutaties uit, dat wil zeggen dat complete rijen in de matrix worden herschikt. De herschikte informatiebits worden vervolgens kolomsgewijs uitgelezen van de matrix en doorgegeven aan de tweede constituent codeerder.

3.4.59. Zoals reeds opgemerkt, verwerkt de eerste constituente codeerder van de turbocoder de originele ongewijzigde invoersequentie en wordt de tweede constituent codeerder sequentie verwerkt door de interleaver. De uitvoeren die door beide constituente codeerders wordt geproduceerd (in het diagram hierboven aangeduid met z_k and z'_k) worden tijdens het coderen aaneengeschakeld met de ‘systematische uitvoer’ (aangeduid met x_k). De UMTS-turbocoder wordt op deze wijze zo gedefinieerd dat hij drie uitvoerbits per invoerbit genereert.

3.4.60. Zodra alle invoerbits door de turbocoder zijn verwerkt, voert de codeerder een procedure uit dat bekendstaat als ‘trellis beëindiging’. Deze procedure houdt in dat er een paar extra bits worden ingevoerd in de twee constituent codeerders, zodat deze terugkeren in de allemaal-nullenstand. Zoals beschreven staat in de 25.212-standaard, worden er als onderdeel van deze procedure 12 extra bits toegevoegd aan de uitvoer van de turbocoder

3.4.61. Hieronder volgt een meer gedetailleerde uitleg van de turbocode interleaver, zoals gedefinieerd in de 25.212 standaard:

(a) Matrix genereren en schrijven van informatiebits

1 De interleaver construeert een tweedimensionale matrix waarin de ingevoerde bits worden opgenomen. Standaard TS 25.212 bepaalt dat het aantal ingevoerde bits, uitgedrukt door een variabele K, kan variëren van 40 tot 5114. Op basis van de waarde van K construeert de interleaver een matrix die groot genoeg is om alle ingevoerde bits te bevatten.

2 De afmetingen van de matrix worden uitgedrukt door een variabele R (aantal rijen) en een variabele C (aantal kolommen).

3 De waarden van R en C worden vastgesteld op basis van de waarde van K.

4 De waarde van R wordt ingesteld op 5, 10 of 20, afhankelijk van de waarde van K. Dat wil zeggen, de matrix kan 5, 10, of 20 rijen bevatten, afhankelijk van de grootte van de invoersequentie. Dit zijn de enige aantallen rijen die de matrix kan bevatten.

5 De waarde van C kan een grotere variatie waarden bevatten. Ten eerste wordt op basis van de waarde van K, een getal p geselecteerd uit een in de standaard opgenomen tabel van priemgetallen. Vervolgens wordt C bepaald op p-1, p of p+1, afhankelijk van welke waarde de kleinst mogelijke matrix oplevert die nog steeds groot genoeg is om alle ingevoerde bits te bevatten.

6 De uit standaard TS 25.212 overgenomen tabel 2 hieronder geeft de mogelijke waarden aan van p op basis waarvan C wordt bepaald. Zoals daaruit kan worden afgeleid, kan p minimaal 7 en maximaal 257 zijn. Er is dus een grotere variatie in het aantal kolommen in de matrix dan in het aantal rijen (die alleen maar 5, 10 of 20 kunnen zijn).

Table 2: List of prime number *p* and associated primitive root *v*

p	*v*	*p*	*v*	*p*	*v*	*p*	*v*	*p*	*v*
7	3	47	5	101	2	157	5	223	3
11	2	53	2	103	5	163	2	227	2
13	2	59	2	107	2	167	5	229	6
17	3	61	2	109	6	173	2	233	3
19	2	67	2	113	3	179	2	239	7
23	5	71	7	127	3	181	2	241	7
29	2	73	5	131	2	191	19	251	6
31	3	79	3	137	3	193	5	257	3
37	2	83	2	139	2	197	2		
41	6	89	3	149	2	199	3		
43	3	97	5	151	6	211	2		

7 Nadat de interleaver een matrix van de juiste grootte heeft geconstrueerd, schrijft de interleaver de ingevoerde bits op opeenvolgende wijze regel voor regel naar de matrix. Als de matrix meer cellen bevat dan de grootte van het invoerblok (dat wil zeggen, als K < R x C), worden er opvulbits toegevoegd om de matrix op te vullen. Als de grootte van de matrix gelijk is aan de grootte van het invoerblok (dat wil zeggen, K = R x C), worden er geen opvulbits toegevoegd.

8 Als bijvoorbeeld K=40, worden de overige waarden als volgt vastgesteld: R=5, p=7, en C=p+1=8. Dit levert de volgende matrix op (de getallen geven de bitposities aan van de invoer sequentie; bijvoorbeeld ‘1’ geeft de eerste bit van de invoer aan).

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40

(Matrix met 5 rijen en 8 kolommen, die 40 ingevoerde bits bevat.)

(b) Intrarij-permutaties

1 Nadat de matrix is geconstrueerd en de data ernaar toe zijn geschreven, voert de interleaver intrarij-permutaties uit. Dat wil zeggen, voor elke rij van de matrix worden de bits van de rij binnen de rij herschikt.

2 De standaard definieert een aantal mathematische formules die worden toegepast voor het uitvoeren van intrarij-permutaties.

3 Van belang voor het doel van EP 516 is dat de standaard licht afwijkende definities verstrekt voor de intrarij-permutatie voor de gevallen waarbij C=p, C=p-1, of C=p+1.

4 In de gevallen C=p en C=p-1, resulteren de intrarij-permutaties in het herschikken van de bits van elke rij, inclusief de allerlaatste bit (dat wil zeggen, de bit in de laatste kolom).

5 Anders dan in de gevallen C=p en C=p-1, leidt de intrarij-permutatie, zoals gedefinieerd in de standaard, in het geval C=p+1 ertoe dat de laatste bit van elke rij niet wordt verplaatst; dat wil zeggen dat de laatste bits van elke rij na de intrarij-permutatie op dezelfde positie blijven staan. Dit kan worden afgeleid uit onderstaande tabel van de K=40 matrix na intrarij-permutaties, waarbij de gemarkeerde kolom laat zien dat geen van de laatste bits is verplaatst:

2	6	5	7	3	4	1	8
10	12	11	15	13	14	9	16
18	22	21	23	19	20	17	24
26	28	27	31	29	30	25	32
34	36	35	39	37	38	33	40

(c) De 'bijzonder geval'-bewerking

1 Zoals zojuist opgemerkt, resulteert de in de standaard gegeven intrarij-permutatieformule in het geval van C=p+1 niet tot verplaatsing van de laatste bit van elke rij. (Zoals eerder vermeld sub (b) onder (iv) wordt de laatste bit verplaatst wanneer C=p en C=p-1.)

2 De standaard schrijft echter een enkelvoudige extra bewerking voor die ervoor moet zorgen dat de laatste bit van elke rij in bepaalde omstandigheden toch wordt verplaatst, zelfs wanneer C=p+1.

3 Met andere woorden, de standaard bepaalt dat als C=p+1, de interleaver moet controleren of K=R×C, en dat als dit het geval is, de posities van de eerste en laatste bit van de laatste rij verwisseld moeten worden.

4 Voor de K=40 matrix wordt dit als volgt geïllustreerd, waarbij de twee verwisselde bits zijn gemarkeerd:

2	6	5	7	3	4	1	8

10	12	11	15	13	14	9	16
18	22	21	23	19	20	17	24
26	28	27	31	29	30	25	32
40	36	35	39	37	38	33	34

5 Deze bewerking (verwisselen van de posities van de eerste en laatste bits van de laatste rij) wordt in de standaard uitsluitend voorgeschreven voor toepassing in een beperkt aantal omstandigheden, waarbij C=p+1 and K=R×C. Zij mag niet worden toegepast wanneer C=p of C=p-1. Ook wordt zij niet toegepast als K < R×C.

6 Of het bijzonder geval waarbij C=p+1 en K=R×C ontstaat, hangt af van de waarde van K, omdat de waarden van de andere parameters worden bepaald door de waarde van K.

7 Zoals eerder opgemerkt, kan de waarde van K volgens de standaard variëren van 40 tot 5114, zodat K dus volgens de standaard 5075 verschillende waarden kan hebben. Het bijzondere geval van de situatie waarin C=p+1 en K=R×C komt voor bij 59, oftewel 1,2%, van deze waarden.

(d) Interrij-permutaties

1 Na op elke rij intrarij-permutaties te hebben uitgevoerd en, indien nodig, de bewerking voor het 'bijzondere geval' waarin C=p+1 en K=R×C, worden volledige rijen van de matrix binnen de matrix herschikt.

2 De standaard definieert vier verschillende interrij-permutatie patronen. Elk daarvan vereist echter dat in deze stap de laatste rij wordt verplaatst naar de eerste rij.

3 Voortbordurend op hetzelfde voorbeeld waarbij K=40, levert de interrij-permutatie de volgende matrix op:

40	36	35	39	37	38	33	34
26	28	27	31	29	30	25	32
18	22	21	23	19	20	17	24
10	12	11	15	13	14	9	16
2	6	5	7	3	4	1	8

4 Uit de figuur valt af te leiden dat na uitvoering van de 'bijzonder geval' bewerking en interrij-permutaties, de laatste bit van de invoer zich nu bevindt in de eerste kolom van de eerste rij.

(e) Uitlezen van informatiebits

1 De gepermuteerde bits worden vervolgens kolom voor kolom uitgelezen uit de matrix, beginnend met de meest linkse kolom.

2 Voortbordurend op hetzelfde voorbeeld waarbij K=40, is de laatste uitvoer sequentie die door de interleaver wordt geproduceerd als volgt:

40, 26, 18, 10, 2, 36, 28, 22, 12, 6, 35, 27, 21, 11, 5, 39, 31, 23,
15, 7, 37, 29, 19, 13, 3, 38, 30, 20, 14, 4, 33, 25, 17, 9, 1, **34**,
32, 24, 16, 8.

3 De laatste bit van de invoer (bit 40 in het voorbeeld) wordt uiteindelijk de eerste bit van de interleaver-uitvoer. Zonder aanpassing vanwege een 'bijzonder geval' (waarbij bit 40 in het voorbeeld zou worden verwisseld met bit 34), zou bit 40 dichter aan het eind van de uitvoersequentie (vijfde van achteren) verschijnen.

EP 516

3.4.62. EP 516 betreft een turbocoder die wordt gebruikt in de processtap van kanaalcoderen (channel coding). Dat begrip en de daarin gebruikte turbocoder zijn hiervoor uitgebreid aan de orde gekomen.

3.4.63. De in conclusie 1 van EP 516 geclaimde turbocoder bestaat – zoals gebruikelijk was in de stand van de techniek – uit een eerste codeerinrichting (111), een interne interleaver (112)[10] en een tweede codeerinrichting (113). De interne interleaver zorgt ervoor dat het invoerbitpatroon van de tweede codeerinrichting anders is dan dat van de eerste codeerinrichting om zodoende te voorkomen dat uit beide codeerinrichtingen een codewoord met een laag Hamming-gewicht zou kunnen komen.

3.4.64. Partijen hebben conclusie 1 in de volgende afzonderlijke deelkenmerken opgedeeld:

1. Turbocodeerorgaan, omvattende:

2. een eerste codeerorgaan (111) dat is ingericht voor

2.1. het coderen van een frame van K inkomende informatiebits voor het genereren van eerste gecodeerde symbolen;

3. een interleaver (112) die is ingericht voor

3.1. het in opeenvolging schrijven van de K inkomende informatiebits in een RxC rechthoekige matrix rij voor rij, beginnend in de eerste kolom van de eerste rij,

3.2. het intra-rij permuteren van de posities van de informatiebits in de RxC rechthoekige matrix in elke rij volgens een gegeven interleavingregel,

3.2.1. waarbij genoemd intra-rij permuteren de posities van de bits in de laatste kolom van genoemde matrix ongewijzigd laat,

3.3. na genoemd permuteren, het onderling verwisselen van de positie van de informatiebit in de laatste kolom van de laatste rij en een positie binnen de laatste rij die voorafgaat aan de laatste kolom,

3.4. het uitvoeren van inter-rij permutaties van de RxC rechthoekige matrix, en

[10] Zoals gezegd heeft deze interne interleaver een andere functie dan de (eerste) interleaver die wordt gebruikt in de processtap ná het channel coding (en welke interleaver een rol speelt in EP 528).

3.5. het kolom voor kolom uitlezen van de informatiebits uit de gepermuteerde RxC rechthoekige matrix, beginnend in de eerste rij van de eerste kolom; en

4. een tweede codeerdorgaan (113) dat is ingericht voor

4.1. het coderen van de uitgelezen informatiebits voor het genereren van tweede gecodeerde symbolen,

5. waarbij de RxC rechthoekige matrix R rijen en C kolommen heeft, K = RxC en het aantal inkomende informatiebits in het frame specificeert, en K > R > 1.

3.4.65. Conclusie 1 heeft feitelijk enkel betrekking op het (interne) interleavingsalgoritme van de turbocoder.

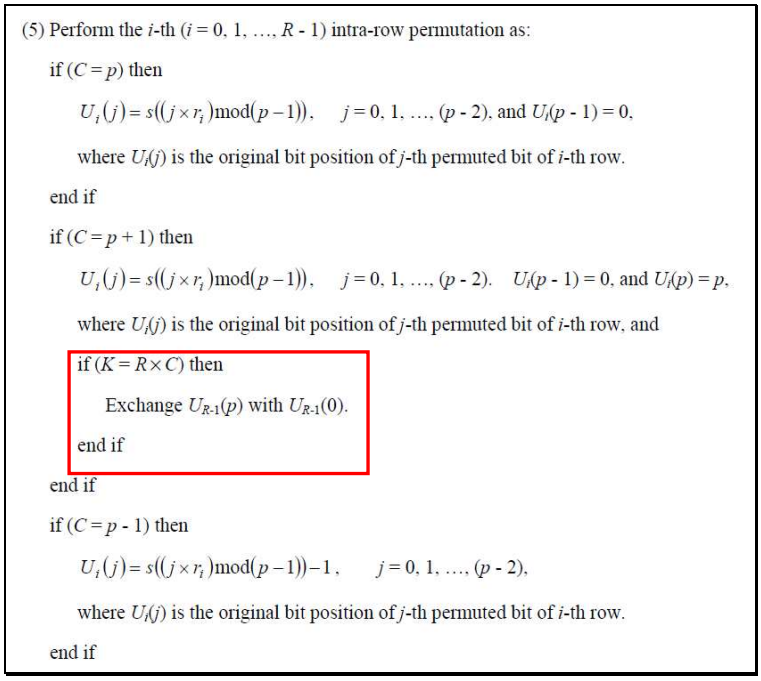
3.4.66. Volgens EP 516 vertoont een “prime interleaver” (PIL) – die is aangemerkt als het werkmodel van de turbo code interleaver die wordt gespecificeerd door de prior art versie van de standaard (1.0.0) – bepaalde problemen waardoor hij een aangetaste vrije-afstandseigenschap heeft (‘degraded free distance property’; par. [0012]).

3.4.67. Vervolgens wordt in EP 516 erkend dat de PIL interleaver uit de stand der techniek reeds alle op-één-na in conclusie 1 geclaimde stappen uitvoerde, inclusief het bouwen van een matrix, het schrijven van informatiebits naar de matrix door middel van rijen, intrarij-permutaties, interrij-permutaties en het kolom voor kolom uitlezen van bits uit de matrix (par. [0016]).

3.4.68. In het octrooi wordt verder opgemerkt dat in bepaalde gevallen (C=p+1 en K=R×C; zie hiervoor) de intrarij-permutaties in de interleaver uit de stand der techniek de laatste bit in elke laatste positie van elke rij behield (de laatste kolom veranderde niet) en dat dit een probleem vormde dat opgelost diende te worden (par. [0054]).

3.4.69. De stap in het interleavingproces die volgens EP 516 wordt toegevoegd aan de interleaver zoals beschreven in versie 1.0.0 van de standaard, is de hiervoor beschreven ‘bijzonder geval’ bewerking, waarbij de positie van de laatste bit in de laatste rij met een ander bit in dezelfde rij wordt verwisseld (deelkenmerk 3.3).

3.4.70. Hieronder wordt een gedeelte uit versie 3.11.0 van de standaard weergegeven met betrekking tot de intrarij-permutatiestap, die slechts één van de uitgebreide reeks stappen vormt die met betrekking tot het turbocoder interleaven in de standaard worden gedefinieerd. De in EP 516 als uitvinding geclaimde “bijzonder geval’-bewerking is daarin rood gemarkeerd (p. 19):



(5) Perform the i-th ($i = 0, 1, \ldots, R - 1$) intra-row permutation as:

if ($C = p$) then

$U_i(j) = s((j \times r_i) \bmod (p-1)), \quad j = 0, 1, \ldots, (p - 2)$, and $U_i(p - 1) = 0$,

where $U_i(j)$ is the original bit position of j-th permuted bit of i-th row.

end if

if ($C = p + 1$) then

$U_i(j) = s((j \times r_i) \bmod (p-1)), \quad j = 0, 1, \ldots, (p - 2)$. $U_i(p - 1) = 0$, and $U_i(p) = p$,

where $U_i(j)$ is the original bit position of j-th permuted bit of i-th row, and

if ($K = R \times C$) then

Exchange $U_{R-1}(p)$ with $U_{R-1}(0)$.

end if

end if

if ($C = p - 1$) then

$U_i(j) = s((j \times r_i) \bmod (p-1)) - 1, \quad j = 0, 1, \ldots, (p - 2)$,

where $U_i(j)$ is the original bit position of j-th permuted bit of i-th row.

end if

3.4.71. EP 516 claimt dat de daarin opgenomen ‘bijzonder geval’-aanpassing van de interleaver van de standaard uit de stand der techniek een octrooieerbare uitvinding was omdat deze rekening houdt met de problematische inkomende bitsequenties met een **Hamming-gewicht 1** (slechts één “1” en de rest nullen) – en in het bijzonder een bitsequentie met de ene “1” aan het eind van de bitsequentie (000….0001) –, als gevolg waarvan een turbocoder codewoorden met een laag Hamming-gewicht produceert (par. [0011], [0012] en [0036], zie r.o. 2.8).

3.4.72. Het octrooi stelt dat dergelijke bitsequenties eerder nog niet als problematisch waren aangemerkt in de stand der techniek (par. [0011]).

3.4.73. EP 516 stelt dat de prestatie van een turbocoder wordt beïnvloed door de interne interleaver ervan en dat het wenselijk is dat een interleaver de minimumafstand tussen de codewoorden van de turbo code vergroot (par. [0004]-[0006]). In het octrooi wordt erkend dat er vele soorten interleavers bekend zijn in de techniek (par. [0003]).

3.4.74. Tevens wordt in EP 516 erkend dat het reeds bekend was in de techniek dat bepaalde invoersequenties met een Hamming-gewicht 2 (slechts twee “1”-en en de rest nullen) ertoe leidt dat een turbocoder codewoorden met een laag Hamming-gewicht genereert (par. [0009]-[0011]).

3.4.75. Het octrooi wijst specifiek het gebruik van recursieve convolutionele (turbo) coders aan als reden waarom turbo codes minimum-gewichtcodewoorden produceren voor gewicht-2 inkomende patronen. Volgens EP 516 gebeurt dit omdat als de twee “1”-en van de bitsequentie worden gescheiden door een specifiek aantal posities (het aantal hangt af van het specifieke ontwerp van de codeerder, inclusief het aantal geheugenelementen dat het bevat), de geheugenelementen van de recursieve convolutionele codeerder terugkeren naar

de positie waarin zij allemaal een nul hebben en het gewicht van de uitvoer niet verder toeneemt. Dit betreft hetzelfde probleem dat hiervoor is besproken.

3.4.76. Het octrooi noemt een invoersequentie die ervoor zorgt dat de turbocoder een codewoord met een laag Hamming-gewicht uitvoert een kritisch informatie sequentiepatroon (‘critical information sequence pattern of CISP’; par. [0009]). Het octrooi stelt dat, aangezien deze problematische patronen in de standaard Hamming-gewicht 2 bitsequenties zijn, men zich bij het ontwerp en de analyse van de interne interleaver van een turbo code concentreerde op die problematische bitsequenties (en dus niet op Hamming-gewicht 1 bitsequenties die de aanleiding van het octrooi vormden).

3.4.77. De uitvinders van EP 516 stellen dat over het hoofd werd gezien dat invoersequenties met een Hamming-gewicht 1 in bepaalde omstandigheden kunnen leiden tot het genereren van minimum-gewicht codewoorden (par. [0011] en [0036]). Zie par. [0011]):

> *However, for the CISP, it is conventional that the information word has the minimum Hamming weight, when the input information word has the Hamming weight 2. In other words,* ***the fact that the CISP can be generated even when the input information word has the Hamming weight 1 (i.e., when the input information word has one information bit of ‘1’) was overlooked****, when the information word input to the turbo-coder had the type of a block comprised of frames.* (nadruk toegevoegd).

3.4.78. Voor wat betreft Hamming-gewicht 1 bitsequenties wordt in EP 516 opgemerkt dat een recursieve convolutionele codeerder de eigenschap vertoont dat hij continu een oneindig aantal uitgaande pariteitbits “1” genereert, zelfs bij slechts één inkomend informatiebit “1” (par. [0033]). Het octrooi verwijst hier naar hetzelfde bekende fenomeen dat is besproken in r.o. 3.4.36, waarin een recursieve convolutionele codeerder die een enkele “1” heeft ontvangen als invoer, continu een groot aantal “1”-en zal genereren bij de uitvoer, zelfs als hij daarna een constante stroom “0”-en als invoer ontvangt.

3.4.79. Niettegenstaande deze welbekende eigenschap van recursieve convolutionele codeerders stelt EP 516 dat een Hamming-gewicht 1 bitseqentie er nog steeds voor kan zorgen dat een recursieve convolutionele codeerder een minimum-gewicht codewoord genereert als de enkele “1” van de invoer zich aan het einde van de bitsequentie bevindt, dus 0000…00001 (par. [0036]):

> *When only the information bit located at the last position of the input information word, i.e., the last position of the frame, is ‘1’ and all the other information bits are 0’s, the Hamming weight of the input information word becomes 1. In this case, the number of the symbols ‘1’ output from the [recursive convolutional encoder] becomes very small, because there is no more input information word.*

3.4.80. Bovenstaande passage uit het octrooi houdt in dat wanneer de “1” zich aan het einde van de invoer bevindt, de codeerder geen mogelijkheid heeft een groot aantal “1”-en te generen bij de uitvoer – anders dan het geval is bij andere Hamming-gewicht 1 inkomende bitsequenties (zoals bijvoorbeeld 0100….0000) – omdat het coderingsproces

onmiddellijk nadat de “1” is ontvangen eindigt. Dit kan worden geïllustreerd aan de hand van de volgende tabel met toestandsovergangen voor de in par. 3.4.26 e.v. besproken recursieve convolutionele codeerder, voor de situatie waarin de “1” als allerlaatste bit wordt ontvangen:

Tijd	Invoer	Huidige toestand	Uitvoer 1	Volgende toestand
0	0	000	0	0
1	0	000	0	0
…	0	000	0	0
K-3	0	000	0	0
K-2	0	000	0	0
K-1	1	000	1	100

3.4.81. Bovenstaande tabel laat zien dat als de codeerder een stroom “0”-en ontvangt terwijl de geheugens allemaal op nul staan, deze uitsluitend nieuwe “0”-en zal genereren bij de uitvoer. Als de codeerder vervolgens een “1” ontvangt als laatste bit (waarbij K-1 de laatste tijdseenheid vertegenwoordigt aangezien er K inkomende bits zijn en de eerste inkomende bit op tijdstip 0 arriveert), zal deze bij de uitvoer weliswaar een “1” genereren, maar zal deze niet doorgaan met nog meer “1”-en te genereren omdat de codering is afgerond. Dus terwijl van een Hamming-gewicht bitsequentie waarbij de “1” *niet* aan het eind staat, verwacht kan worden dat deze een hoog-gewicht codewoord genereert, is het tegenovergestelde het geval bij een Hamming-gewicht 1 bitsequentie waarbij de “1” *wel* aan het eind staat.

3.4.82. Met betrekking tot de hierboven beschreven kwestie wordt in EP 516 opgemerkt dat als een turbocoder een (problematische) bitsequentie met een enkele “1” aan het eind ontvangt, de *eerste* constituente codeerder (die de oorspronkelijke niet-vervlochten inkomende bitsequentie codeert) een laag-gewicht codewoord zal genereren (par. [0039]). Indien nu de interne interleaver de “1” van diezelfde bitsequentie ná interleaving in de laatste positie zou laten staan (zodat de bitsequentie dus niet zou veranderen en nog steeds 0000…0001 zou zijn), dan zou de *tweede* constituente codeerder (die de door de interne interleaver vervlochten bitsequentie codeert) ook een laag-gewicht codewoord genereren (par. [0039]). Als gevolg daarvan zou het door de overkoepelende turbo code gegenereerde codewoord een laag Hamming-gewicht hebben, omdat het Hamming-gewicht daarvan de som is van de Haming-gewichten van de door beide codeerders uitgevoerde codewoorden (par. [0039]).

3.4.83. De onderstaande figuur 4 van EP 516 toont een problematische interleaver die er niet in slaagt de laatste bit te verplaatsen. De gemarkeerde posities geven aan dat de laatste bit van de inkomende bitsequentie ná interleaving in de laatste positie blijft staan:

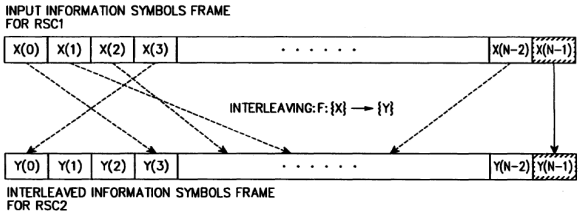


FIG. 4

3.4.84. Volgens EP 516 kan het probleem waarbij beide codeerders laag-gewicht uitvoer produceren, worden vermeden door de interleaver zo te ontwerpen dat deze de laatste bit verplaatst, op zodanige wijze dat de “1” niet meer de laatste bit is die door de tweede codeerder wordt verwerkt (slot par. [0039]):

> *However, if, as shown in FIG. 5, the turbo interleaver shifts the position of the input information word, where the original symbol of the RSC1 is ‘1’, to the first position or a position near the leading position of the frame after interleaving, the number of the output symbols ‘1’ generated from the RSC2 will be increased. This is because a plurality of symbols ‘1’ are output through …* [a number of] *state transitions of the RSC2 encoder. In this case, the RSC2 generates a great number of the output symbols ‘1’, thereby increasing the total free distance.*

3.4.85. De onderstaande figuur 5 van EP 516 toont wat volgens het octrooi een verbeterde interleaver is die de laatste bit verplaatst. De gemarkeerde posities in de bitsequentie die interleaving heeft ondergaan, geven aan dat de laatste bit kan worden verplaatst naar de eerste positie van de vervlochten bitsequentie (in de figuur geïndexeerd als Y(0)) of naar een andere positie nabij de eerste positie (Y(2)):

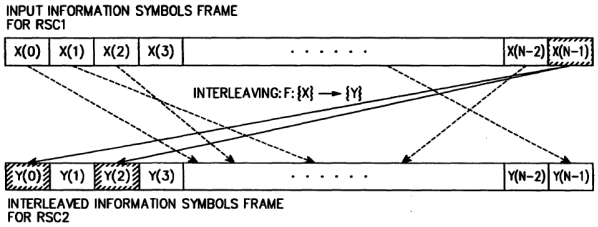


FIG. 5

3.4.86. EP 516 vat de bespreking van het vermeende probleem en de oplossing ervan samen door te verklaren dat bij het ontwerp van de interleaver aan bepaalde voorwaarden voldaan moet worden om de prestaties van de turbocoder en de vrije afstand van de turbocoder te garanderen, alsmede dat de informatiebits die corresponderen met de laatste positie van het frame door middel van interleaving moeten worden verschoven naar een positie voorafgaand aan de laatste positie (indien mogelijk naar de voorste positie van het frame), teneinde de vrije afstand van de turbo code te vergroten (par. [0045]).

3.4.87. De uitvindingsgedachte van EP 516 komt derhalve neer op het zodanig ontwerpen van de interne interleaver van een turbocoder dat deze de laatste bit van de inkomende bitsequentie verplaatst naar een positie dichter bij het begin van de uitvoer van de interleaver, om het probleem op te lossen waarbij, wanneer de laatste bit in de laatste positie zou blijven, een turbocoder een minimum-gewicht codewoord zou genereren.

3.4.88. EP 516 stelt verder dat het daarin beschreven beweerdelijke probleem uitsluitend wordt aangetroffen in de interne interleaver van de turbocoder van versie 1.0.0 van de standaard, voor de situatie waarin C=p+1 (par. [0051]-[0054]). Het octrooi wijst erop dat indien C=p+1, de intrarij-permutatiestap zoals gedefinieerd in de standaard, de laatste bit van elke rij in de matrix op dezelfde positie laat staan. Dit houdt in dat de laatste bit van de inkomende bitsequentie niet wordt verplaatst, aangezien deze bit de laatste bit van de laatste rij in de matrix is.

3.4.89. EP 516 geeft aan dat dit probleem kan worden opgelost door na de intrarij-permutaties een extra stap toe te voegen. De octrooibeschrijving geeft voor deze extra stap zes opties, die enigszins van elkaar verschillen maar die stuk voor stuk neerkomen op het verplaatsen van de positie van de laatste bit in de laatste rij van de matrix naar een andere positie in de laatste rij (par. [0055]). In conclusie 1 is uiteindelijk gekozen voor de technische maatregel dat de informatiebit in de laatste kolom van de laatste rij verwisseld wordt met de informatiebit in de laatste rij die voorafgaat aan de laatste kolom (deelkenmerk 3.3).

3.5. EP 516 - Geldigheid

Prioriteit/nieuwheid

3.5.1. Apple heeft aangevoerd dat EP 516 geen prioriteit mag ontlenen aan de Koreaanse aanvraag met nummer 18928/1999 ingediend op 19 mei 1999 (hierna P1), waarvan Apple een (onbestreden) vertaling als productie 40 heeft overgelegd. Apple stelt onder meer dat in P1 geen inter-row permutation (deelkenmerk 3.4) is terug te vinden. Die stelling treft doel.

3.5.2. Samsung heeft gewezen op een passage uit P1 op pagina 18 en figuur 11:

> Referring to FIG. 11, a row vector permutation block (or row vector permutation index generator) 912 generates an index for selecting a row vector according to counting of a row counter 911, and provides the generated index to a high address buffer of the address buffer 918. A column vector permutation block (or column vector's elements permutation index generator) 914 generates, depending on a modified PIL algorithm 915, an index for permuting the positions of the elements in the corresponding row vector (or group) according to counting of a column counter 913, and provides the generated index to a low address buffer of the address buffer 918. A RAM (Random Access Memory) 917 stores temporary data generated in the process of the program. A look-up table 916 stores parameters for interleaving and the primitive root. The addresses obtained by row permutation and column permutation (i.e., the addresses stored in the address buffer 918) are used as addresses for interleaving.

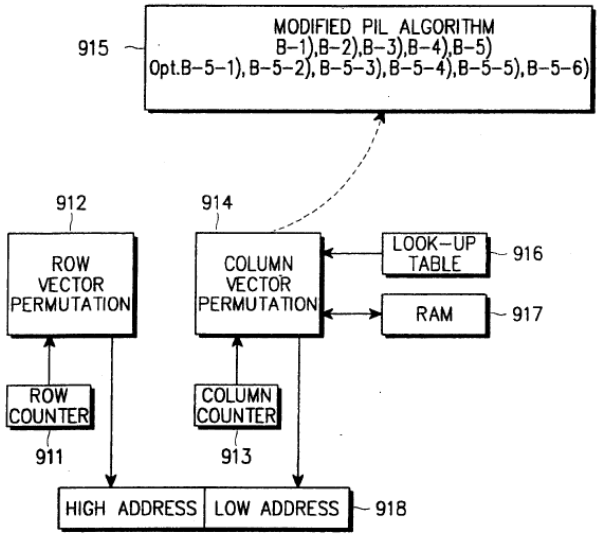


FIG. 11

Apple heeft onder verwijzing naar bevindingen van haar deskundige Dr. Hamkins (Apple productie 71) uitvoerig bestreden dat uit deze passage en figuur de inter-rij permutatie stap blijkt. Zelfs echter als met Samsung aan te nemen is dat een gemiddelde vakman uit figuur 11 en de beschrijving, en dan met name door gebruik van de woorden “row vector permutation”, (met moeite) af zou kunnen leiden dat in die uitvoering een inter-rij permutatie plaatsvindt, dan nog valt daaruit niet voldoende duidelijk en ondubbelzinnig af te leiden dat dit een kenmerk zou zijn van de in P1 geopenbaarde uitvinding. Dit geldt te minder nu figuren 9 en 10 evenzeer ter illustratie van de uitvinding worden gepresenteerd (zie r.o. 2.9). Die figuren, hierna opgenomen, komen ongewijzigd terug in het uiteindelijk verleende octrooi.

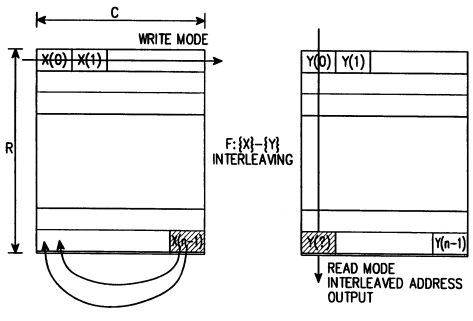


FIG. 9

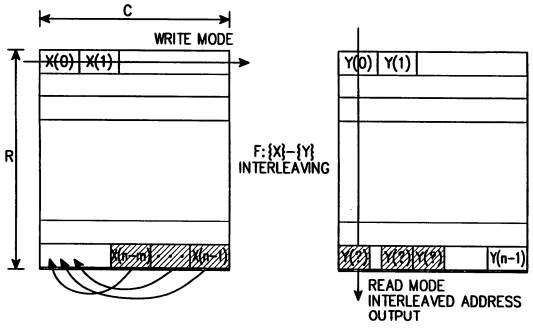


FIG. 10

Net als bij de andere figuren van P1 zijn de te verplaatsen bits gearceerd in figuren 9 en 10 weergegeven. Als er ook nog inter-rij verplaatsing zou plaatsvinden, zouden die verplaatste

gearceerde bits niet langer in de onderste rij staan, maar naar boven zijn verplaatst, alvorens te worden uitgelezen (in de rechter figuur van figuren 9 en 10). Uit deze figuren kan derhalve zonder meer worden afgeleid dat in deze uitvoeringsvoorbeelden geen inter-rij permutatie plaatsheeft. Deze interpretatie is door Samsung niet weersproken.

3.5.3. De conclusie dat inter-rij permutatie niet voldoende duidelijk en ondubbelzinnig wordt geopenbaard in P1 wordt ondersteund door het feit dat juist de inter-rij permutatie beschrijvende passages uit EP 516 ten opzichte van P1 zijn toegevoegd. Zo is er in P1 geen verwijzing naar een “third stage” voor inter-rij permutatie zoals wel in paragraaf [0052] (r. 15) van het octrooi en ontbreekt paragraaf [0053] van het octrooi, waarin deze permutatiestap uitgebreid wordt toegelicht, in zijn geheel in P1. Ook is in paragraaf [0058] van het octrooi (zie r.o. 2.8) aan de hiervoor weergegeven passage uit P1 op een cruciale plek een zin toegevoegd die inter-rij permutatie moet verduidelijken (onderstreping rechtbank):

> Referring to FIG. 11, a row vector permutation block (or row vector permutation index generator) 912 generates an index for selecting a row vector according to counting of a row counter 911, and provides the generated index to a high address buffer of the address buffer 918. The row vector permutation block 912 is a group selector for sequentially or randomly selecting, when the input information word is divided into a plurality of groups, the divided groups. A column vector permutation block (or column vector’s elements permutation index generator) 914 generates, depending on a modified PIL algorithm 915, an index for permuting the positions of the elements in the corresponding row vector (or group) according to counting of a column counter 913, and provides the generated index to a low address buffer of the address buffer 918. The column vector permutation block 914 is a randomizer for permuting the position of the information bits in the group, which were sequentially stored in the order of input, according to a given rule. A RAM (Random Access Memory) 917 stores temporary data generated in the process of the program. A look-up table 916 stores parameters for interleaving and the primitive root. The addresses obtained by row permutation and column permutation (i.e., the addresses stored in the address buffer 918) are used as addresses for interleaving.

3.5.4. Dat betekent dat EP 516 zich ten onrechte beroept op de indiendatum van P1, waardoor EP 516 onbestreden niet langer nieuw is vanwege de reeds bekende standaard 3.1.1 (productie 72 Apple) ten tijde van de aanvraagdatum.

Inventiviteit

3.5.5. Zelfs als er niettegenstaande het voorgaande een geldig beroep op prioriteit zou kunnen worden gedaan, voert Apple terecht aan dat EP 516 niet inventief is.

3.5.6. Met partijen zal de rechtbank uitgaan van versie 1.0.0 van standaard TS 25.212 van april 1999 (productie 39 Apple) als meest nabije stand van de techniek. Het eerst bij pleidooi door Samsung opgeworpen argument dat deze standaard een fout bevat en aldus niet als in aanmerking te nemen stand van de techniek heeft te gelden, wordt als onvoldoende onderbouwd gepasseerd. Apple heeft daar immers onweersproken tegenin gebracht dat deze fout door een gemiddelde vakman onmiddellijk zou worden onderkend en op juiste wijze zou worden gelezen, onder meer onder verwijzing naar EP 516 waar de uitvinders kennelijk het algoritme van de standaard ook zonder meer zo hadden opgevat. De

vakman zal na lezing van standaard 1.0.0 derhalve hebben gezien dat de bits in de laatste kolom van de matrix bij C = p + 1 niet worden verplaatst.

3.5.7. EP 516 verschilt van deze stand van de techniek door de omwisseling na de intra-rij permutatie van het laatste bit bij een volledig met te coderen informatiebits gevulde matrix (zodat K = R × C, als er minder informatiebits zijn, zou het laatste bit ergens anders staan, zie Hamkins paragraaf 40, productie 73 Apple) en C = p + 1 (deelkenmerk 3.3). Nu voormelde fout in de 1.0.0 standaard onderkend zal worden, is de gemiddelde vakman tevens bekend dat bij C = p + 1 volgens het verhusselingsalgoritme van de 1.0.0 standaard de laatste kolom blijft staan tijdens de intra-rij verhusseling (kenmerk 3.2.1). Evenmin kan Samsung's – voor het eerst bij pleidooi naar voren gebrachte – stelling worden gevolgd dat bij de 1.0.0 standaard niet ook reeds kolom voor kolom wordt uitgelezen (deelkenmerk 3.5). Ten eerste had Apple reeds bij conclusie van antwoord in conventie, tevens voorwaardelijke conclusie van eis in reconventie, gesteld dat ook dit kenmerk uit de 1.0.0 standaard volgde (paragraaf 8.6-8.12), hetgeen Samsung bij conclusie van antwoord in reconventie onbestreden heeft gelaten. Bovendien lazen de uitvinders kennelijk al wel in de 1.0.0 standaard dat kolom voor kolom wordt uitgelezen, getuige paragraaf [0016] van EP 516.

3.5.8. Het technische effect van de maatregel uit deelkenmerk 3.3 is dat wanneer een informatiebitsequentie allemaal nullen en slechts als laatste bit een "1" heeft, volgens standaard 1.0.0 door de ene encoder weliswaar een codewoord met een heel laag Hamming gewicht wordt gegenereerd, maar door de interleaving en codering in de andere encoder een codewoord met hoog Hamming gewicht wordt gegenereerd. Daardoor wordt de codeerprestatie van de 1.0.0 standaard verbeterd.

3.5.9. Het objectieve op te lossen probleem kan aan de hand daarvan worden geformuleerd als het verbeteren van de codeerprestatie in de zin van verhoging van het Hamming gewicht van de uitgevoerde codewoorden door de 1.0.0 standaard, met name in het geval waarbij een bitsequentie met allemaal nullen en slechts als laatste bit een "1" wordt ingevoerd (en het aantal kolommen C = p + 1). Samsung heeft nog betoogd dat het probleem breder moet worden geformuleerd, namelijk meer algemeen als het verbeteren van de codeerprestatie van de 1.0.0 standaard, doch dat standpunt kan niet worden gevolgd. EP 516 biedt immers die verbetering ook slechts in dat hele specifieke geval en laat het al geformuleerde algoritme van standaard 1.0.0 voor het overige in stand. Samsung heeft voorts nog gesuggereerd dat de uitvinding (tevens) zou liggen in de onderkenning van het probleem bij bitsequenties met allemaal nullen en een laatste "1". Terecht heeft Apple daartegenin gebracht dat algemeen bekend was onder coderingsdeskundigen dat ook deze sequentie problematisch voor turbocodering zou zijn. Deze sequentie is op dezelfde manier problematisch als de bitreeksen met Hamming gewicht 2 waarvan Samsung aangeeft dat dit algemeen bekend was (zie onder meer 7.22 conclusie van antwoord in reconventie). Ook in het artikel van Hé wordt dit bekend verondersteld (p. 453, linker kolom, tweede alinea van de Introduction, productie 43 Apple). De reden hiervoor is dat een eigenschap van dit type turbocodeerder is dat als hij steeds nullen ontvangt, hij ook nullen uitvoert. Als de codeerder een "1" ontvangt als laatste bit, zal deze bij de uitvoer weliswaar een "1" genereren, maar zal de codeerder niet doorgaan met nog meer "1"-en te genereren omdat de codering al is afgerond. Dus terwijl van een bitsequentie met Hamming-gewicht 1 waarbij de "1" niet aan het eind maar ergens aan het begin staat, verwacht kan worden dat deze een hoog-gewicht codewoord genereert, is het tegenovergestelde het geval bij een Hamming-gewicht 1 bitsequentie waarbij de "1" wel aan het eind staat.

3.5.10. Een gemiddelde vakman zou zonder inventieve denkarbeid komen tot de oplossing om het laatste “1” bit om te wisselen met een eerder gelegen bit teneinde het Hamming gewicht te verhogen. Zo hij dit al niet zonder meer op basis van zijn algemene vakkennis zou inzien, zou hij die informatie zonder meer vinden in de publicatie van Hé (p. 453, rechterkolom onderaan):

> The weights of output sequence should be as big as possible to obtain a good performance and this is achieved by the interleaver. For the sequences which generate only small weights from the first encoder, they should be interleaved to other patterns so that they generate large weights from the second encoder. Several circumstances are considered under which small weights are generated. By avoiding the situations, the interleaver design method may be derived.
>
> (1) For the transmission frame of size L, if the ‘1’s only occur near the end of the sequence, such as 0…01 and 0…0110. The sequence will generate small output weights in the first encoder. The reason is that the output is cut off by the termination required in the decoding process. **Therefore the interleaver should map the bits at the end to the front or middle of the sequence**. **(nadruk toegevoegd)**

Zowel het probleem van sequenties met een laag (Hamming) gewicht, waaronder specifiek ook die met een laatste “1” in een systeem met twee encoders wordt besproken en hoe die laatste bits dan naar voren moet worden verplaatst (“gemapt”). Een zelfde leer valt uit het artikel van Robertson te halen (productie 44 Apple). Weliswaar zijn er ook andere plaatsen denkbaar om het laatste bit naar voren te verplaatsen, en ook om dat pas aan het einde van de intra- en inter-rij verplaatsingen te doen, maar dat levert technisch in hoge mate equivalente oplossingen op die geen van alle inventief zijn.

3.5.11. Samsung’s stelling dat de gemiddelde vakman ook het codeeralgoritme van de 1.0.0 standaard had kunnen wijzigen, in die zin dat de laatste kolom niet zou blijven staan in het bijzondere geval, gaat evenmin op. Apple heeft voldoende inzichtelijk gemaakt dat een gemiddelde vakman op de prioriteitsdatum daar niet voor zou hebben gekozen, omdat hij weet dat dit algoritme al in hoge mate vastlag vanwege de standaardisering. De gemiddelde vakman op zoek naar een oplossing voor het laatste “1”-geval wist dat het moest gaan om een zo eenvoudig mogelijke aanpassing om dat probleem op te lossen, zonder dat het al bereikte akkoord op het algoritme van de standaard op losse schroeven zou komen te staan. Andersom gezegd, de rechtbank acht het niet van uitvindingshoogte getuigen dat een gemiddelde vakman in hoge mate door de standaardisering gebonden, daarin een voor de hand liggende aanpassing bedenkt.

3.5.12. Bij deze stand van zoeken komt de rechtbank niet toe aan de overige verweren van Apple, waaronder dat er geen sprake zou zijn van inbreuk op EP 516.

3.6. Conclusie in conventie

3.6.1. Op grond van het voorgaande moet worden geconcludeerd dat er geen sprake is van inbreuk op EP 528 en dat EP 516 ongeldig is. De vorderingen in conventie moeten om die reden worden afgewezen.

3.6.2. Samsung zal als de in het ongelijk gestelde partij worden veroordeeld in de proceskosten. Partijen zijn overeengekomen dat kosten van de totale procedure moeten worden begroot op € 500.000,00 (€ 250.000,00 per octrooi). Partijen hebben geen

onderverdeling gemaakt tussen de kosten van de procedure in conventie en reconventie, maar zij hebben wel aangegeven dat een bedrag van € 100.000,00 (per octrooi) betrekking heeft op de geldigheid. Aangezien in de reconventie uitsluitend de geldigheid aan de orde was, zal de rechtbank de overige € 150.000,00 (per octrooi) volledig toerekenen aan de conventie, en de kosten betreffende de geldigheid gelijkelijk verdelen over de conventie en reconventie. Samsung zal dus in conventie worden veroordeeld tot betaling van een bedrag van 2 x € 200.000,00 = € 400.000,00.

3.7. Conclusie in reconventie

3.7.1. Gegeven de uitkomst van de procedure in conventie moet in reconventie worden vastgesteld dat de voorwaarde waaronder Apple haar vorderingen heeft ingesteld niet is ingetreden voor EP 528. De beoordeling van deze vordering kan dus achterwege blijven. Conclusie 1 van EP 516 zal worden vernietigd.

3.7.2. Het instellen van de voorwaardelijke reconventie kan worden aangemerkt als een redelijke vorm van verdediging tegen de vorderingen in conventie. Daarom komen de in reconventie voor EP 528 gemaakte kosten ook in aanmerking voor vergoeding. De totale kosten aan de zijde van Apple worden, onder verwijzing naar rechtsoverweging 3.6.2, begroot op 2 x € 50.000,00 = € 100.000,00.

4. DE BESLISSING

De rechtbank

in conventie

4.1. wijst het gevorderde af;

4.2. veroordeelt Samsung in de proceskosten, tot op heden aan de zijde van Apple begroot op € 400.000,00;

in voorwaardelijke reconventie

4.3. vernietigt conclusie 1 van EP 516;

4.4. verstaat dat de voorwaarde waaronder de vordering ter zake EP 528 is ingesteld, niet is vervuld;

4.5. veroordeelt Samsung in de proceskosten, tot op heden aan de zijde van Apple begroot op € 100.000,00;

4.6. wijst het meer of anders gevorderde af;

In conventie en in voorwaardelijke reconventie

4.7. verklaart dit vonnis, voor zover het de kostenveroordelingen betreft, uitvoerbaar bij voorraad.

Dit vonnis is gewezen door mr. E.F. Brinkman, mr. P.H. Blok en mr. J.Th. van Walderveen en in het openbaar uitgesproken op 20 juni 2012.