(12) UK Patent Application (19) GB (11) 2 443 866 (13) A

(43) Date of A Publication 21.05.2008

(21) Application No: 0622764.9

(22) Date of Filing: 15.11.2006

(71) Applicant(s):
Motorola Inc
(Incorporated in USA - Delaware)
1303 East Algonquin Road, Schaumburg,
Illinois 60196, United States of America

(72) Inventor(s):
Jean Sidon
Mihael S Bercovici
Yaron Shemesh

(74) Agent and/or Address for Service:
Optimus
Grove House, Lutyens Close,
Chineham Court, BASINGSTOKE,
Hampshire, RG24 8AG, United Kingdom

(51) INT CL:
H03M 13/27 (2006.01) *H03M 13/29* (2006.01)

(56) Documents Cited:
US 20050204262 A1 **US 20030048206 A1**
IEEE Transactions on Information Theory, Volume 46 Number 6, September 2000, Takeshita O Y et al, "New deterministic interleaver designs for turbo codes", pages 1988-2006.

(58) Field of Search:
INT CL **H03M, H04L**
Other: **EPODOC, WPI, INSPEC**

(54) Abstract Title: **Quadratic congruence condition turbo coding interleaver calculates and swaps multiple element pairs in parallel in each interleaving step**

(57) An interleaver (100) is for re-arranging in each of L cycles a first set of K data elements into a second set of K data elements using a permutation vector which defines, based on the calculation in each cycle of a c_m, an even number of pairs of positions of in the first set to be swapped, wherein c_m satisfies a quadratic congruence condition.

A processor (103) computes for each of the L cycles a value of the sequence c_m and, based on the value, a set of indexes indicating at least two pairs of elements to be swapped, 15 and a processor (101) swaps in each cycle pairs of indicated elements, the swapping of pairs in each cycle being done together in a parallel swapping operation.

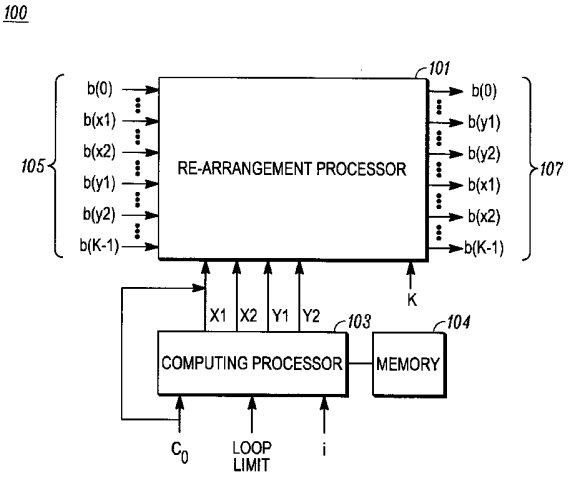

FIG. 1

GB 2 443 866 A

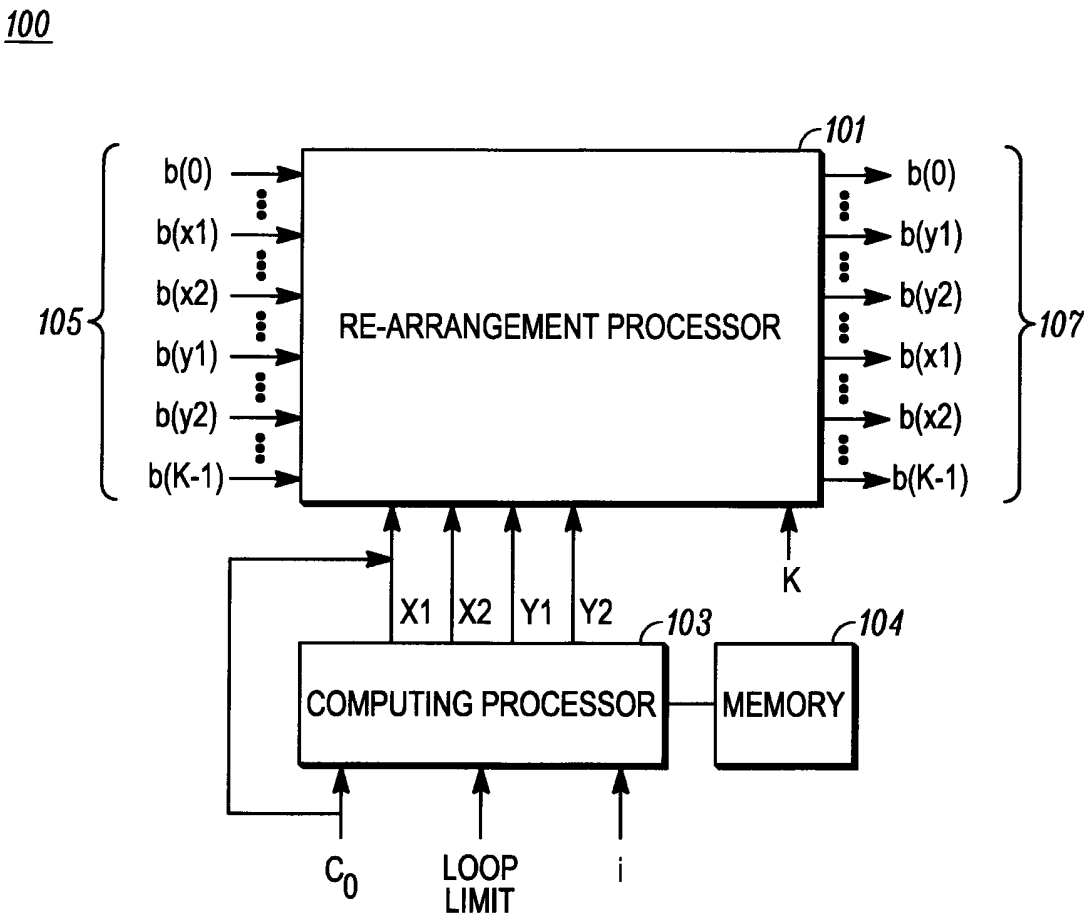
100
101
b(0)
b(x1)
b(x2)
b(y1)
b(y2)
b(K-1)
105
RE-ARRANGEMENT PROCESSOR
b(0)
b(y1)
b(y2)
b(x1)
b(x2)
b(K-1)
107
K
X1
X2
Y1
Y2
103
104
COMPUTING PROCESSOR
MEMORY
C0
LOOP LIMIT
i

FIG. 1

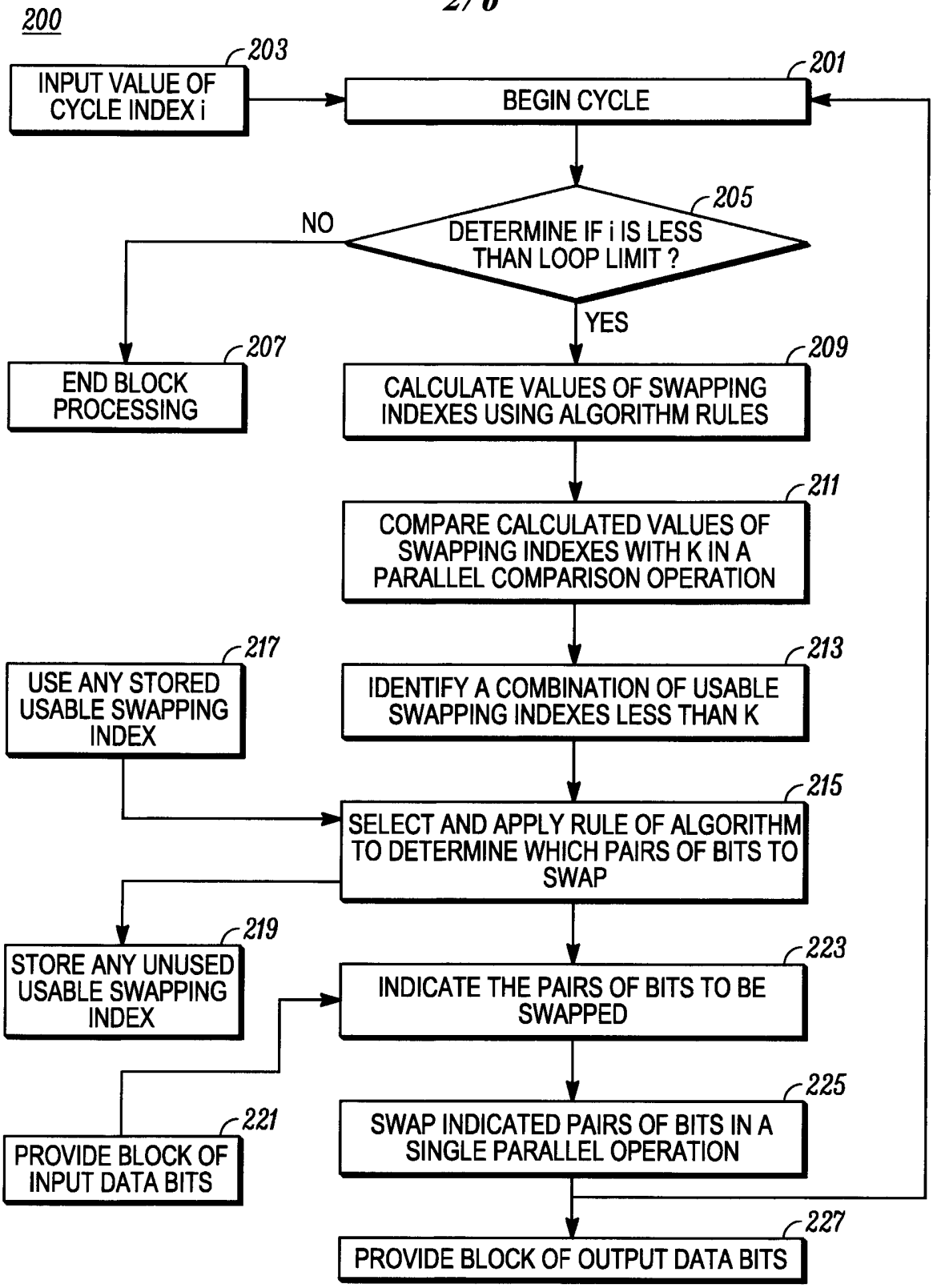
200
203
INPUT VALUE OF CYCLE INDEX i
201
BEGIN CYCLE
205
DETERMINE IF i IS LESS THAN LOOP LIMIT ?
NO
YES
207
END BLOCK PROCESSING
209
CALCULATE VALUES OF SWAPPING INDEXES USING ALGORITHM RULES
211
COMPARE CALCULATED VALUES OF SWAPPING INDEXES WITH K IN A PARALLEL COMPARISON OPERATION
217
USE ANY STORED USABLE SWAPPING INDEX
213
IDENTIFY A COMBINATION OF USABLE SWAPPING INDEXES LESS THAN K
215
SELECT AND APPLY RULE OF ALGORITHM TO DETERMINE WHICH PAIRS OF BITS TO SWAP
219
STORE ANY UNUSED USABLE SWAPPING INDEX
223
INDICATE THE PAIRS OF BITS TO BE SWAPPED
225
SWAP INDICATED PAIRS OF BITS IN A SINGLE PARALLEL OPERATION
221
PROVIDE BLOCK OF INPUT DATA BITS
227
PROVIDE BLOCK OF OUTPUT DATA BITS

FIG. 2

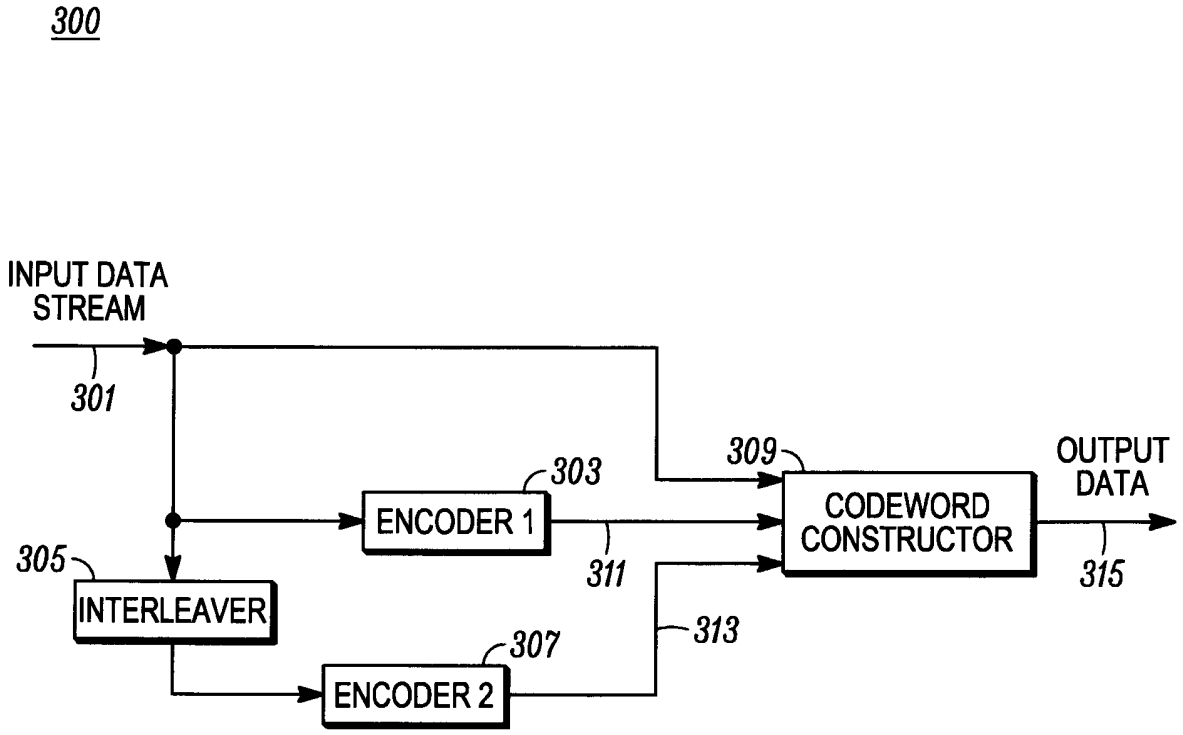
300
INPUT DATA STREAM
301
303
ENCODER 1
309
CODEWORD CONSTRUCTOR
OUTPUT DATA
305
INTERLEAVER
311
315
307
ENCODER 2
313

FIG. 3

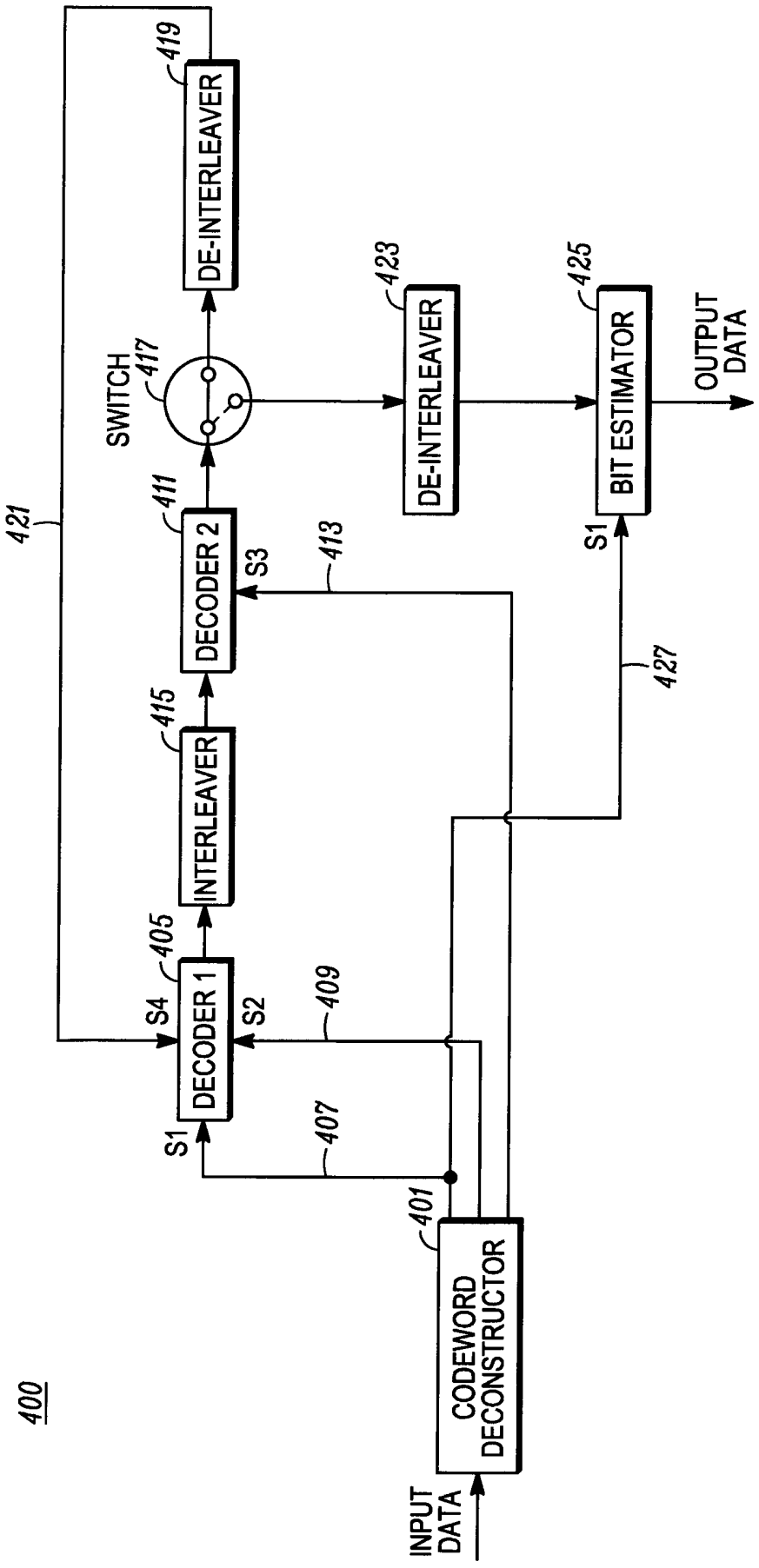
400
INPUT DATA
CODEWORD DECONSTRUCTOR
401
DECODER 1
405
INTERLEAVER
415
DECODER 2
411
DE-INTERLEAVER
419
DE-INTERLEAVER
423
BIT ESTIMATOR
425
OUTPUT DATA
SWITCH
417
S1
S2
S3
S4
407
409
413
421
427

FIG. 4

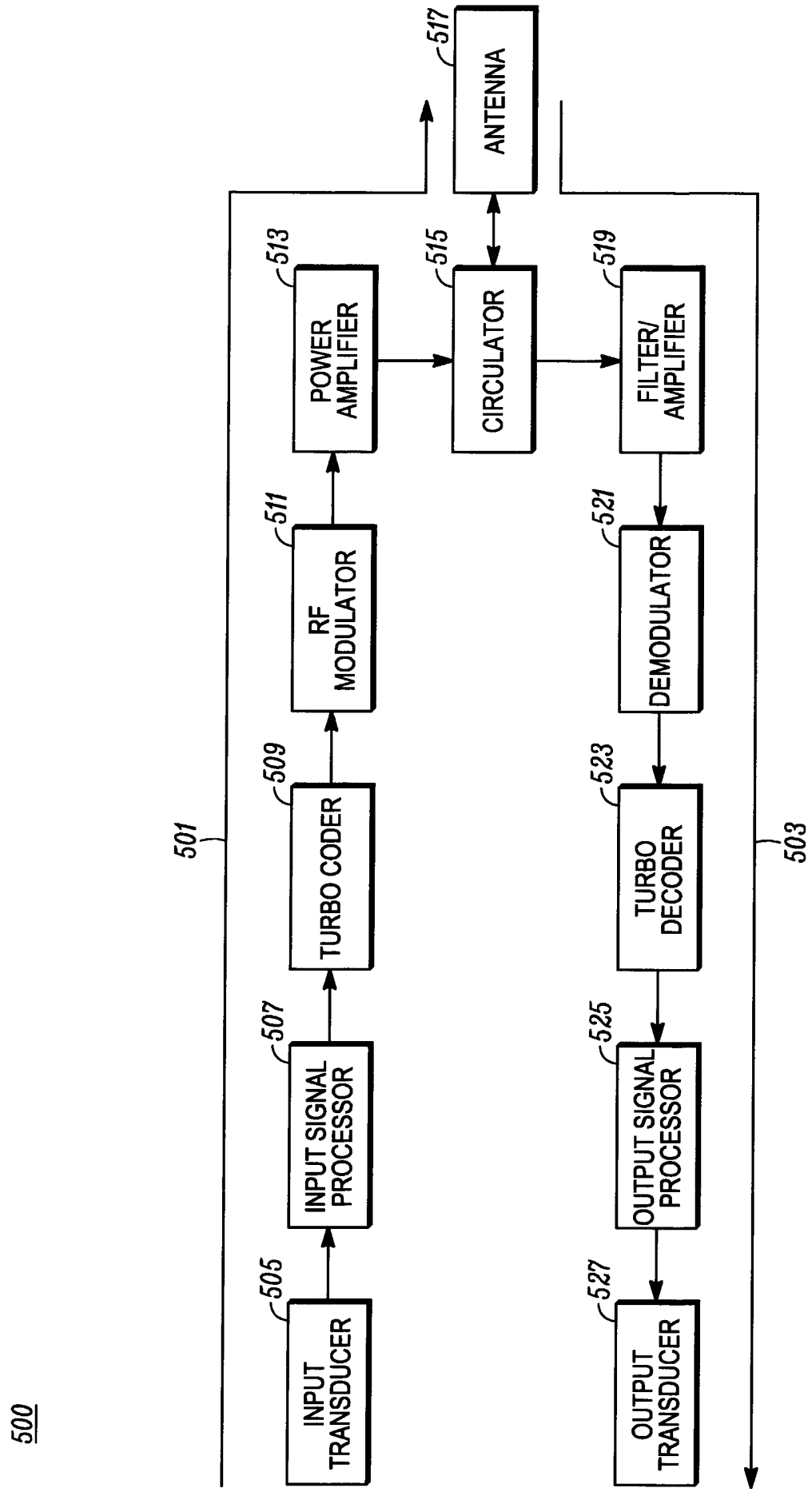
500
501
503
505
INPUT TRANSDUCER
507
INPUT SIGNAL PROCESSOR
509
TURBO CODER
511
RF MODULATOR
513
POWER AMPLIFIER
515
CIRCULATOR
517
ANTENNA
519
FILTER/ AMPLIFIER
521
DEMODULATOR
523
TURBO DECODER
525
OUTPUT SIGNAL PROCESSOR
527
OUTPUT TRANSDUCER

FIG. 5

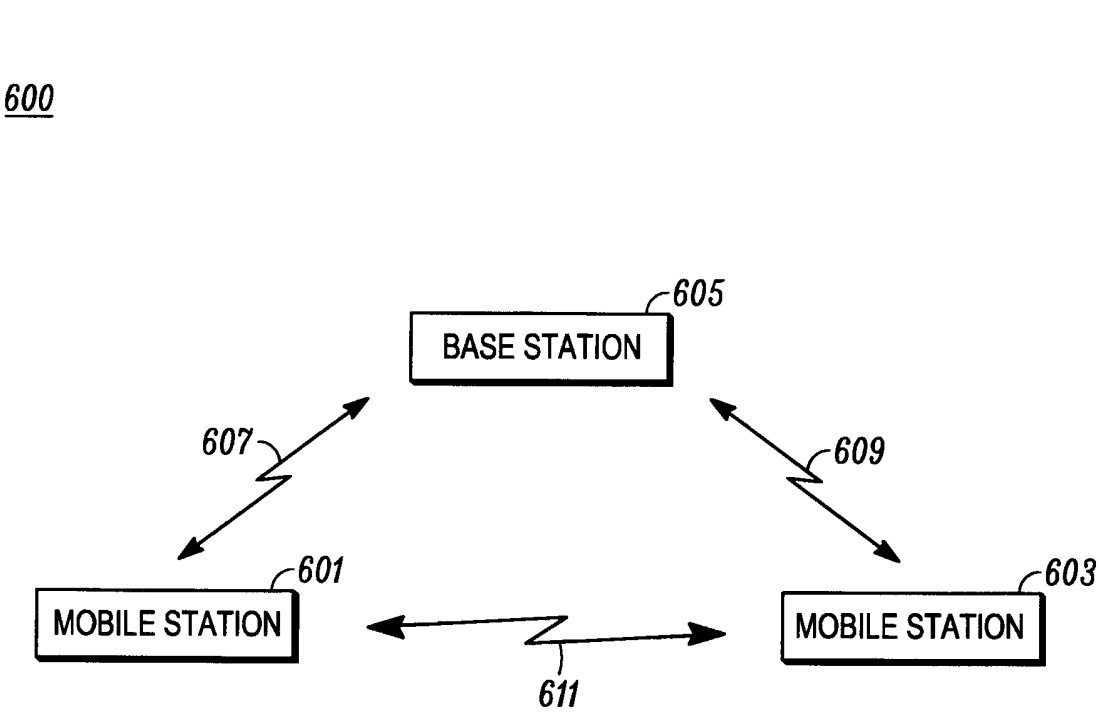
600
605
BASE STATION
607
609
601
MOBILE STATION
603
MOBILE STATION
611

FIG. 6

TITLE: INTERLEAVER FOR USE IN TURBO CODING

FIELD OF THE INVENTION

The present invention relates to an interleaver for use in turbo coding. In particular, the present invention relates to a quadratic congruence interleaver suitable for use in turbo coding in forward error correction in a digital communication system.

BACKGROUND OF THE INVENTION

Digital communication systems, particularly for specialized uses such as applications of the emergency and public safety services, need to perform accurately and reliably in the presence of noise and interference. Forward error correction ('FEC') coding is a known way of helping to achieve this goal. Forward error correction coding (also called 'channel coding') is a type of digital signal processing that improves data reliability by introducing a known structure into a data sequence prior to transmission. This known structure is introduced by a coder (encoder) associated with a transmitter that is to transmit a signal including the data sequence. The coder inserts redundant (or 'parity') bits into the data stream, thereby providing an output formed of a longer sequence of data bits, called a 'codeword'. A decoder associated with a receiver which receives the signal is able to extract the original data sequence and to detect and correct errors caused by corruption during communication without requesting retransmission of the original data.

A powerful known form of forward error correction coding is turbo coding. In turbo coding, a turbo coder includes a combination of at least two component encoders. The turbo coder also includes an interleaver (also sometimes referred to as a 'permuter' or a 'shuffler'). The interleaver is a processor that receives data and re-arranges it in a different order using a programmed re-arrangement operation. A first of the component encoders and the interleaver receive each block of input data. Usually, the first component encoder and the interleaver receive each input data block in parallel in a mode known as a 'parallel concatenation' mode. The interleaver processes the input data block and delivers the processed block to a second of the component encoders. The first and second component encoders thereby generate different sets of code bits based upon the input data block. This provides diversity to the coded data sequence being transmitted. Although in principle any interleaving processing pattern can be adopted for use by the interleaver, different patterns can result in significant differences in the communication bit-error rate. Therefore, the interleaver design contributes significantly to the overall error correction performance of the turbo code system.

The turbo decoding procedure must employ as many component decoders associated with the receiver as component encoders associated with the transmitter. These decoders may be concatenated in a serial fashion and may be joined by a series of interleavers and de-

interleavers in a feedback loop arrangement. Typically, the decoding procedure is iterative by sending decoded data round the feedback loop in a number of iterations to improve the quality of the data, i.e. to improve the probability of the data being correct. By use of the iterative decoding process, turbo codes can achieve a bit-error rate that approaches the theoretical Shannon limit.

Various interleavers are known in the prior art. For use in turbo code processing, interleavers are typically selected which provide a combination of desirable computational properties. Such interleavers include pseudo-random or deterministic interleavers. An example of such a known deterministic interleaver is the Takeshita-Costello interleaver which is described in the paper, herein referred to as Reference 1, entitled 'New Deterministic Interleaver Designs for Turbo Codes, by Oscar Y. Takeshita and Daniel J. Costello Jr., published in IEEE Transactions on Information Theory, Vol. 46., No. 6, September 2000. This interleaver is known to be suitable for use in a convolutional turbo coder in its parallel concatenation mode for forward error correction. This interleaver provides an operation which has a relatively low mathematical complexity, but its known implementation in a real time environment is relatively complex.

SUMMARY OF THE INVENTION

According to the present invention in a first aspect there is provided an interleaver as defined in claim 1 of the accompanying claims.

According to the present invention in a second aspect there is provided a turbo coder as defined in claim 11 of the accompanying claims.

According to the present invention in a third aspect there is provided a turbo decoder as defined in claim 12 of the accompanying claims.

According to the present invention in a fourth aspect there is provided a digital communication system as defined in claim 13 of the accompanying claims.

According to the present invention in a fifth aspect there is provided a digital communication terminal as defined in claim 14 of the accompanying claims.

According to the present invention in a sixth aspect there is provided method of operation in an interleaver, the method being as defined in claim 15 of the accompanying claims.

Further features of the invention are as defined in the accompanying dependent claims and as disclosed in the embodiments of the invention to be described.

Embodiments of the present invention will now be described by way of example with reference to the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, in which like reference numerals refer to identical or functionally similar elements throughout the separate views and which together with the detailed description below are incorporated in and form part of the specification, serve to further illustrate various embodiments and to explain various principles and advantages of the present invention.

In the accompanying drawings:

FIG. 1 is a block schematic diagram of an interleaver embodying the invention.

FIG. 2 is a flow chart of a method embodying the invention employed in the interleaver of FIG. 1.

FIG. 3 is a block schematic diagram of an illustrative turbo coder embodying the invention.

FIG. 4 is a block schematic diagram of an illustrative turbo decoder embodying the invention.

FIG. 5 is a block schematic diagram of a transceiver of a digital communication terminal embodying the invention.

FIG. 6 is a block schematic diagram of a digital wireless communication system adapted in accordance with an embodiment of the invention.

DESCRIPTION OF EMBODIMENTS OF THE INVENTION

Before describing in detail embodiments that are in accordance with the present invention, it should be observed that the embodiments reside primarily in

combinations of method steps and apparatus components related to interleavers and their use in turbo coders for communication between a first terminal and a second terminal. Accordingly, the apparatus components and method steps have been represented where appropriate by conventional symbols in the drawings, showing only those specific details that are pertinent to understanding the embodiments of the present invention so as not to obscure the disclosure with details that will be readily apparent to those of ordinary skill in the art having the benefit of the description herein.

In this document, relational terms such as 'first' and 'second', 'top' and 'bottom', and the like may be used solely to distinguish one entity or action from another entity or action without necessarily requiring or implying any actual such relationship or order between such entities or actions. The terms 'comprises', 'comprising', or any other variation thereof, are intended to cover a non-exclusive inclusion, such that a process, method, article, or apparatus that comprises a list of elements does not include only those elements but may include other elements not expressly listed or inherent to such process, method, article, or apparatus. An element preceded by 'comprises ...a' does not, without more constraints, preclude the existence of additional identical elements in the process, method, article, or apparatus that comprises the element.

It will be appreciated that embodiments of the invention described herein may be comprised of one or more conventional processors and unique stored program instructions that control the one or more processors to

implement, in conjunction with certain non-processor circuits, some, most, or all of the functions of interleavers and their use in turbo coding for communication between a first terminal and a second terminal. As such, these functions may be interpreted as steps of a method of operation to perform interleaving in turbo coding in communication between a first terminal and a second terminal. Alternatively, some or all functions could be implemented by a state machine that has no stored program instructions, or in one or more application specific integrated circuits (ASICs), in which each function or some combinations of certain of the functions are implemented as custom logic. Of course, a combination of the two approaches could be used. Thus, methods and means for these functions have been described herein. Further, it is expected that one of ordinary skill, notwithstanding possibly significant effort and many design choices motivated by, for example, available time, current technology, and economic considerations, when guided by the concepts and principles disclosed herein will be readily capable of generating such software instructions and programs and ICs with minimal experimentation.

It has been recognised in the art that quadratic congruence interleavers, as described in Reference 1 specified earlier, are powerful interleavers because they can give a performance similar to random interleavers. For example, the TETRA Series 2 set of standards being developed by ETSI (the European Telecommunications Standards Institute) has specified that for forward error correction this type of

interleaver is to be used in a turbo encoder in the parallel concatenation mode. In consequence, there is considerable current interest in such interleavers.

In operation of a quadratic congruence interleaver, also known as a 'Takeshita - Costello' interleaver, data elements, e.g. bits or symbols each of which can have different possible values, of an input set or block of K data elements is delivered to the interleaver. The interleaver re-arranges the data elements using an algebraically derived permutation table defining permutations of pairs of data elements whose positions are to be swapped. The interleaver may construct the permutation table by use of an index mapping function giving for each of L re-arrangement cycles sequences c_m defining the position of pairs of the data elements to be swapped. Each sequence c_m is given by:

$$c_m = [c_{m-1} + hm] \bmod S, m = 1,2,...,S-1 \quad \text{Equation 1}$$

where S is the smallest power of two which is greater than or equal to K and h is a constant which is an odd integer. Reference 1 shows that Equation 1 is equivalent to a quadratic congruence condition for c_m. The permutation table which results is equivalent to constructing a permutation vector in which the elements are shifted left by an amount equal to h.

In the TETRA Series 2 set of standards referred to earlier, operation of the quadratic congruence interleaver is defined in terms of a two-step algorithm that should be applied to give the required data element re-arrangement. This two step algorithm, which is referred to herein as 'Algorithm 1', is specified in the

TETRA Series 2 set of standards (written using well known algorithm shortened notations) as follows:

'A quadratic-congruence block interleaver shall re-order K bits $b_2(1)$, $b_2(2)$,…, $b_2(K)$ into K permuted bits $b'_2(1)$, $b'_2(2)$,…, $b'_2(K)$, by means of the following two-step algorithm:

a) First, the sequence of indices c_m, $m = 0, 1,\ldots,S-1$, is calculated, where S is the smallest power of 2 larger or equal than K as follows:

$c_0 = 0, and$

$c_m = [c_{m-1} + m] \bmod S, m = 1,2,...,S-1$

b) Second, the K bits $b_2(1)$, $b_2(2)$,…, $b_2(K)$ undergo the following procedure:

```
flag ← false
i ← 0
while i ≤ (S − 2)/2
    x ← c_{i+1}
    y ← [c_i + S/2] mod S
    if (x < K_2 and y < K_2)
        swap bits b_2(x+1) and b_2(y+1)
    else if (x < K_2 and y ≥ K_2)
        if (flag = true)
            swap bits b_2(x+1) and b_2(t+1)
            flag ← false
        else
            t ← x
            flag ← true
    else if (x ≥ K_2 and y < K_2)
        if (flag = true)
            swap positions b_2(y+1) and b_2(t+1)
            flag ← false
        else
            t ← y
            flag ← true
    i ← i+1'
```

Thus, in the second step, step b of Algorithm 1, a series of re-arrangement cycles are to be applied to a block of data of length K. Each of these cycles is identified by an index *i*. A series of algorithmic operations is presented by Algorithm 1 defining pairs of elements to be swapped for each cycle. Algorithm 1 implies that the swapping of pairs of elements in each cycle is carried out as a series of individual swapping steps. Carried out in this way step b requires many individual swapping operations carried out serially. The number of swapping operations is related to the size of the block of data elements to be re-arranged. For example, in accordance with the TETRA Release 2 standards, K can be up to 5,536. In a hardware (programmable logic) implementation, this requires typically 5536/2 clock cycles.

In relation to an embodiment of the invention it has been recognised that a method equivalent to Algorithm 1 can be implemented in less clock cycles than as contemplated by the prior art referred to above. An illustrative quadratic congruence interleaver 100 embodying the invention which is capable of implementing a method equivalent to Algorithm 1 in a reduced number of clock cycles will now be described with reference to FIGS. 1 and 2.

Referring first to FIG. 1, the interleaver 100 embodying the invention comprises a re-arrangement processor 101 which is operably connected to a computing processor 103 operably connected in turn to a memory 104. A block 105 of K input data bits b is applied to the re-arrangement processor 103. The block 105 may be a

sub-block of a larger block of data bits that has to be re-arranged in steps or cycles of operation by the interleaver 100. Each of the K input data bits b of the block 105 is applied in parallel to the re-arrangement processor 103. The block 105 is shown as having K input data bits b having position indexes from 0 to K-1. For illustrative simplicity, four (i.e. two pairs) of the input bits b are candidates to be swapped in position in pairs. However, in practice the block 105 may have many more pairs of input data bits which are candidates for swapping in each re-arrangement cycle. An input position index of each of the input data bits b in turn is indicated in FIG. 1 in brackets following each designation of b, i.e. as 0,… x1,… x2,… y1,… y2 … K-1 respectively. The input data bits b having the bracketed position indexes x1, x2, y1 and y2 are the bits which are candidates to be swapped. These bits are not necessarily adjacent in position in the block 105 of K bits as indicated by dots between the various data bits b. The input data bits b having the bracketed position indexes x1, x2, y1 and y2 are re-arranged in position by the re-arrangement processor 101 in each of L re-arrangement cycles, where L is an integer herein referred to as the 'loop limit' and is equal to (S-2)/2, where S is as defined earlier. The data bits b are re-arranged to form a block 107 of output data bits. In a first re-arrangement cycle, the data bits of the block 107 of output data bits are as shown in FIG. 1, in which the indexes in brackets of the output data bits refer to the positions of the corresponding input bits b in the block 105. Thus it can be seen for the first re-

arrangement cycle that the input bits b(y1) and b(x1) have been swapped in position and the input bits b(y2) and b(x2) have been swapped in position. The swapping operations in each of the L re-arrangement cycles are carried out together as a parallel operation.

The swapping operations between pairs of bits b required in each of the re-arrangement cycles may be defined by swapping indexes. In FIG. 1, the swapping indexes associated with each of bits b(x1), b(x2), b(y1) and b(y2) are indicated respectively as X1, X2, Y1 and Y2. Thus, the swapping index X1 defines the position index of the input data bit b (if any) with which the data bit b(x1) is to be swapped in a given cycle. The swapping index X2 defines the position index of the input data bit b (if any) with which the data bit b(x2) is to be swapped, and so on.

The values of the swapping indexes X1, X2, Y1 and Y2 are computed by the computing processor 101 to give output blocks 107 for each re-arrangement cycle which are equivalent to output data as obtained by the operation of Algorithm 1. The computing processor 101 is provided with the following inputs:

(i) an input c_0 which is a parameter for the index calculation, having an initial value of zero;

(ii) an input i which is a value of the current cycle index; this may be provided by a cycle counter (not shown);

(iii) an input indicating the 'loop limit', which is a limit L of the number of re-arrangement cycles; this input is an integer equal to the number (S-2)/2.

The computing processor 103 operates in a manner described with reference to FIG. 2 to compute for each cycle a sequence c_m as defined earlier and, from the sequence, pairs of bits b to be re-arranged in each cycle. The computing processor 103 indicates to the re-arrangement processor 101 the computed pairs, e.g. using an indication provided in the form of values of the swapping indexes X1, X2, Y1 and Y2. The re-arrangement processor 101 applies the swapping operations defined by the indication from the computing processor 103. Multiple swapping operations are carried out in parallel by the re-arrangement processor 101.

FIG. 2 is a flow chart of an illustrative method 200 of operation of the interleaver 100 in accordance with an embodiment of the invention. The method 200 comprises multiple re-arrangement cycles or iterations each indicated by a cycle index i. Each cycle begins at a step 201. The cycle index is indicated in a step 203 by input to the computing processor 103 of a value of i for the current cycle. In a step 205, the computing processor 103 determines if the value of i for the current cycle is less than a loop limit which is equal to (S-2)/2. If step 205 produces a 'No' result, i.e. a determination that the loop limit has been reached, processing for the current block of input data ends at a step 207. If step 205 produces a 'Yes' result, i.e. a determination that the loop limit has not been reached, the method 200 proceeds to a step 209. In step 209, the computing processor 103 calculates values of swapping indexes for the current cycle i. The computing processor 103 uses a set of relationships, obtained by inspection

of Algorithm 1, to compute values of the swapping indexes. Where the swapping indexes are specifically X1, X2, Y1 and Y2 as defined earlier, the relationships used are as follows:

X1 is given by $(c_0 + i) \bmod S$;

X2 is given by $(c_0 + 2i + 1) \bmod S$;

Y1 is given by $(c_0 + S/2) \bmod S$;

Y2 is given by $(c_0 + 2i + 1 + S/2) \bmod S$.

The parameters c_0, i and S are as defined earlier.

The calculations of step 209 are carried out in parallel (in contrast to the serial procedure implied by Algorithm 1), for a plurality of bit pairs, in a hardware implementation. In a step 211, the computing processor 103 compares each of the swapping index values calculated in step 211 with the value K, the number of data bits in the block 107 of input data bits, to determine whether each swapping index is less than or not less than K. The comparisons for each of the swapping indexes with the value K are carried out in parallel in a single step operation in step 211. Each calculated swapping index is not used further if it is found in step 211 to be not less than K. Each calculated swapping index is used further if it is found in step 211 to be less than K.

In a step 213, the computing processor 103 identifies a combination of the swapping indexes which are less than K. For example, where the swapping indexes X1, X2, Y1 and Y2 as described earlier are used, there are four (or, in general p) such indexes giving sixteen (or, in general, 2^p) such combinations possible. This arises as follows. For each and every one of the above

four variable swapping indexes X1, X2, Y1 and Y2 there are two possibilities that can be found in step 211: (i) the index is greater than or equal to K; and (ii) the index is less than K. This gives a total number of two to the power of four, that is sixteen, possible combinations of the comparison results of the four variables X1, X2, Y1, Y2 with respect to the value of K.

In a step 215, the computing processor 103 selects and applies a logical rule of an algorithm to determine which pairs of bits to swap. The algorithm essentially has a different logical rule for each of the possible combinations which can be identified in step 213. The rules to be applied are obtained by inspection of Algorithm 1. Thus, when a particular combination is identified in step 213, the algorithm applied by the computing processor 103 instantly selects and applies the rule corresponding to that combination. This may be done by the algorithm looking in parallel at every one of the possible combinations from step 213 to find which one is currently 'true' and then operating a corresponding rule defined in the algorithm in response to the particular 'true' finding. The computing processor 103 may for example include a lookup table or a multiplexer to implement the algorithm in step 215.

In a first example of step 215, if all of the indexes X1, X2, Y1 and Y2 are found in step 211 to be less than K, then step 213 will identify the combination of the four indexes X1, X2, Y1 and Y2 as usable, and step 215 determines by application of the rule corresponding to this combination that b(x1) should be swapped with b(y1), and b(x2) should be swapped with

b(y2). In a second example, if both of X2 and Y2 are less than K but both of X1 and Y1 are not less than K, step 215 determines by application of the rule corresponding to this combination that only b(x2) should be swapped with b(y2).

The rules applied in step 215 also deal with the following situation. If, as noted earlier, any calculated swapping index is found in step 211 to be less than K it can and will be used in swapping. However, if one of the calculated indexes (say X1) is unusable (i.e. not less than K), while the index with which it should pair (say Y1) as identified in the rule selected in step 215 is usable (i.e. is less than K) a swap cannot be carried out between them. If there is stored in the memory 104 any usable index from previous calculations of the indexes, it is used by pairing with Y1 for a swap. This possibility, the use of any unused usable swapping index, is indicated in FIG. 2 as a step 217. If no such index is already stored, no swap is carried out in the current cycle using the usable swapping index (Y1) and the usable swapping index (X1) is temporarily stored in the memory 104. This possibility, the storage of any unused usable swapping index, is indicated in FIG. 2 as a step 219. Such a stored index is to be used for pairing at the next opportunity, e.g. in the next cycle of the method 200 following the current cycle.

In a step 221, a block of input data bits, e.g. as illustrated by the block 105 in FIG. 1, is provided to the re-arrangement processor 101. In a step 223, the computing processor 103 provides to the re-arrangement

processor 101 an indication of the pairs of bits in the input block that the re-arrangement processor 101 has to swap, as determined by the computing processor 103 in steps 209 to 215. Only the swapping indexes of bits that have to be swapped may be indicated in step 223. Each input indicates the indexes of the pair(s) of bits, e.g. b(x1) with b(y1) and b(x2) with b(y2), to be swapped (if any). Thus, the minimum information that needs to be indicated by the computing processor 103 to the re-arrangement processor 101 consists of the swapping indexes of the pairs of bits that are to be swapped. In a step 225, the re-arrangement processor 101 swaps the pairs of input bits indicated in step 223. Step 225 is carried out as a single parallel operation in which all swaps are carried out together. Finally, in a step 227, a block of output data bits is provided by the re-arrangement processor 101, e.g. as illustrated by the block 107 in FIG. 1. This block includes output data bits which have been swapped in position in step 225. Following step 225, the method returns to step 201 to begin the next cycle having a cycle index i+1, where i is the index of the cycle which has just ended. In each cycle of the method 400 for each given block of input data, a different output block 107 of data is produced.

Since each of steps 209, 211, 223 and 225 in the method 200 is carried out as a parallel operation, the number of overall steps required to achieve an output equivalent to that produced by Algorithm 1 may be considerably reduced compared with use of serial operations, particularly serial swapping operations, contemplated in the prior art. Beneficially, this allows

the interleaver 100 to operate more rapidly using fewer processing resources. The interleaver 100 may therefore be fabricated in a form which is more compact and less expensive than a comparable serial interleaver.

FIG. 3 is a block schematic diagram of an illustrative turbo coder 300 embodying the invention. An input data stream is delivered from a digital signal processor (not shown) via a connection 301. The input data stream represents information to be communicated, e.g. speech information or text, picture or video information. The input data stream comprises a series of discrete, consecutive blocks of data upon each of which the turbo coder 300 operates. The input data stream is applied in parallel to a first component encoder, Encoder 1 303 and an interleaver 305. Each data block of the input data stream is encoded in a known manner by the Encoder 1 303 and is also rearranged by the interleaver 305 in a manner embodying the invention described earlier with reference to FIGS. 1 and 2 to form a re-arranged data block. Each of the re-arranged data blocks is delivered to a second component encoder, Encoder 2 307 and is encoded in a known manner by the Encoder 2 307.

A codeword constructor 309 receives three input signals derived from the input data stream. A first input signal delivered via the connection 301 is provided by the unprocessed input data stream. A second input signal is delivered via a connection 311 from the Encoder 1 303 and is a series of data blocks which have been encoded by the Encoder 1 303. A third input signal is delivered via a connection 313 from the Encoder 2 307

and is a series of re-arranged data blocks which have been encoded by the Encoder 2 307. The codeword constructor 309 uses its three input signals in a known manner to produce output data comprising a series of code words each containing components from each of the three input signals to the codeword constructor 309. The output data is delivered by an output connection 315 to a modulator (not shown) to provide modulation of a communication signal, e.g. an RF carrier signal, using the output data, in a known manner.

FIG. 4 is a block schematic diagram of an illustrative turbo decoder 400 embodying the invention. The turbo decoder 400 receives input data comprising a demodulated signal comprising communicated code words. The input data is delivered to a codeword deconstructor 401 which produces three output signals corresponding respectively to the three input signals employed in the code word constructor 309 shown in FIG. 3. A first output signal S1 which comprises unencoded data is applied to a first decoder, Decoder 1 405 via a connection 407. A second output signal S2 which comprises data encoded by the Encoder 1 303 (FIG. 3) is applied to the Decoder 1 405 via a connection 409. A third output signal S3 which comprises data encoded by the Encoder 2 307 is applied to a second decoder, Decoder 2 411 via a connection 413. The Decoder 1 405 is serially coupled via an interleaver 415 to the Decoder 2 411.

A cycle of a loop processing operation in the turbo decoder 400 proceeds as follows. The Decoder 1 405 decodes and uses the encoded data of the signal S2

correlated with the decoded data of the signal S1 to produce statistical information relating to a bit error rate of received data. The statistical information is delivered to the Decoder 2 411. The Decoder 2 411 decodes and uses the encoded data of the signal S3 correlated with the unencoded data of the signal S1 re-arranged by the interleaver 415 to produce statistical information relating to a bit error rate of received data. The interleaver 415 operates in the manner embodying the invention described earlier with reference to FIGS. 1 and 2. The Decoder 2 411 produces an output signal including statistical information produced by the Decoder 1 405 and improved by the Decoder 2 411 which is delivered via a two-way switch 417 and a de-interleaver 419 to a feedback loop 421 which applies the signal back to the Decoder 1 405. The output signal provided by the Decoder 2 411 also includes decoded data which corresponds to the unencoded but re-arranged data delivered from the interleaver 305 to the Encoder 2 307 (FIG. 3). The de-interleaver 419 produces a reverse re-arrangement of the data to provide an input signal S4 for the Decoder 1 405 which corresponds to a processed version of the unencoded data of the signal S1. The Decoder 1 405 uses the statistical information it receives and the processed version of the encoded data of the signal S4 delivered via the feedback loop 421 to improve the reliability of the input unencoded data of the signal S1. Another cycle of the loop processing operation which has been described then begins. Several cycles may be applied successively for each block of

data received, the gathered statistical information and reliability of the data improving with each cycle.

When sufficient cycles of the loop processing operation have been applied, the switch 417 is operated to divert the output signal produced by the Decoder 2 411 via a de-interleaver 423 to a bit estimator 425. Like the de-interleaver 419, the de-interleaver 423 produces a reverse re-arrangement of the decoded data produced by the Decoder 2. The de-interleavers 419 and 423 may in practice be provided by a single interleaver. Each of the de-interleavers 419 and 423 may be provided by an interleaver operating in the manner embodying the invention described earlier with reference to FIGS. 1 and 2. The signal S1 comprising unencoded data from the codeword deconstructor 401 is also delivered to the bit estimator 425 via a connection 427. The bit estimator 425 uses the respective inputs it receives to estimate which bits in the unencoded data of the signal S1 require correction. The bit estimator 425 produces output data including bits corrected by the bit estimator 425 based upon error estimations made by the bit estimator 425.

FIG. 5 is a block schematic diagram of an illustrative transceiver 500 embodying the invention. The transceiver 500 is suitable for use in a mobile digital wireless communication terminal and includes a transmitter chain 501 and a receiver chain 503 either one of which can be selected to be in operation for a given operational period in a transmitting mode or a receiving mode respectively. The transmitter chain 501 includes an input transducer 505 which in a transmitting

mode receives input information to be communicated, e.g. speech from a user. The input information is applied in the form of an input electronic signal to an input signal processor 507 which provides preliminary signal processing, e.g. to enhance signal quality of the electronic signal. An output signal from the input signal processor 507 comprises a digital data signal which is delivered in the form of consecutive data blocks to a turbo coder 509 which includes an interleaver embodying the invention, e.g. as described earlier with reference to FIGS. 1 and 2. The turbo coder 509 may for example be the turbo coder 300 shown in FIG. 3. An output from the turbo coder 509 in the form of a series of code words is delivered to an RF modulator 511 which uses the code words to modulate an RF carrier signal. An output from the RF modulator 511 comprising a modulated RF signal is amplified by a power amplifier 513 and is delivered via a circulator 515 (or switch or like device) to an antenna 517. The amplified modulated RF signal is transmitted over-the-air by the antenna 517 to a receiver of another terminal (not shown).

When the transceiver 500 is in a receiving mode, the antenna 517 receives a modulated RF signal sent over-the-air by a transmitter of another terminal (not shown). The antenna 517 delivers the received signal to the receiver chain 503 via the circulator 515. A filter/amplifier 519 provides front end RF channel filtering and amplification of the received signal. The received signal after filtering and amplification is delivered to a demodulator 521 which downconverts the RF signal to baseband frequency to extract information

added to the RF signal as a modulation signal when it was transmitted. The demodulator 521 produces a digital output which is a series of code words corresponding to the series formed in a coder similar to the coder 509 of the transmitter that transmitted the signal. The output signal from the demodulator 521 is applied to a turbo decoder 523 including one or more interleavers embodying the invention, e.g. as described earlier with reference to FIGS. 1 and 2. For example, the turbo decoder 523 may be the turbo decoder 400 shown in FIG. 4. Output data from the turbo decoder 523 is further processed in an output signal processor 525, e.g. to enhance the quality of the signal represented by the data, and is delivered to an output transducer 527, e.g. a speaker where the transducer 527 is to deliver output speech information to a user.

FIG. 6 is a block schematic diagram of a wireless communication system 600 embodying the invention. The system 600 includes a mobile station 601 and a mobile station 603. Each of the mobile stations 601 and 603 may be a mobile radio, a portable telephone, a wireless enabled pda (personal digital assistant) or the like. The mobile station 601 may communicate with the mobile station 603 via a base station 605 in which case the system 100 is a cellular or trunked mobile communication system. This communication is established via a radio link 607 between the mobile station 601 and the base station 605 and a radio link 609 between the base station 605 and the mobile station 603. Alternatively, the mobile station 601 may communicate directly with the mobile station 603 via a direct radio link 611. Each of

the mobile stations 601 and 603 includes a transceiver which may be the transceiver 500 as shown in FIG. 5. Thus, if the mobile station 601 is a transmitting terminal and the mobile station 603 is a receiving terminal in the same communication, the mobile station 601 includes a turbo coder embodying the invention, e.g. the turbo coder 300, which encodes the data to be communicated, and the mobile station 603 includes a turbo decoder embodying the invention, e.g. the turbo decoder 400, which decodes the communicated data when received. The turbo coder of the mobile station 601 and the turbo decoder of the mobile station 603 form a forward error correction code system in communication between the mobile station 601 and the mobile station 603.

As will be readily apparent to those of ordinary skill in the art, two or more of the operational functions of components of the interleaver 100 described earlier with reference to FIG. 1 may be combined in a signal digital signal processor. Furthermore, other operational functions of the transceiver 500, e.g. functions of the turbo coder 509 and/or of the turbo decoder 523 and/or functions of the processors 507 and 509 may be incorporated into the same digital signal processor. Thus, a digital signal processor including the interleaver 100, optionally together with other functional components or operations, may be in the form of a suitable programmed microprocessor such as one using ASIC (Application Specific Integrated Circuit) technology or FGPA (Field Programmable Gate Array) technology.

In the foregoing specification, specific embodiments of the present invention have been described. However, one of ordinary skill in the art will appreciate that various modifications and changes can be made without departing from the scope of the present invention as set forth in the accompanying below. Accordingly, this specification and the accompanying drawings are to be regarded in an illustrative rather than a restrictive sense, and all such modifications are intended to be included within the scope of present invention as defined by the accompanying claims.

CLAIMS

1. An interleaver for re-arranging in each of a plurality of L re-arrangement cycles a first set of K data elements into a second set of K data elements using a permutation vector defining an even number of swapping indexes of positions of data elements in the first set to be swapped, wherein the computation of the swapping indexes is based on a sequence c_m that satisfies a quadratic congruence condition, the interleaver including a computing processor for computing in each cycle a set of swapping indexes based on the sequence c_m indicating pairs of data elements to be swapped in the cycle, and a re-arrangement processor for swapping in each cycle pairs of data elements in the first set indicated by the swapping indexes computed by the computing processor, the re-arrangement processor being operable in each cycle to swap the pairs of elements together in a parallel swapping operation.

2. An interleaver according to claim 1 wherein the computing processor is operable to calculate in each of the L cycles a value of the sequence c_m which is given by $c_m = [c_{m-1} + hm]$ modulo S, where m is an integer in the integer series 0,1,..,S-1, where h is a given integer, and S is the smallest power of two which is not less than K.

3. An interleaver according to claim 2 wherein the computing processor is operable to calculate in each cycle, based on the values of the sequence c_m calculated, the swapping indexes for a number of bit pairs from the K data elements in the first set and to detect whether

each of the values of the swapping indexes computed has a value less than the value K, and to indicate that each of the computed indexes is usable in re-arrangement only if the computed value of the indexes is less than the value K.

4. An interleaver according to claim 2 or claim 3 wherein the computing processor is operable to compute at least one of: (i) values of the sequence c_m; and (ii) the swapping indexes of the pairs of bits to be swapped; by a parallel computing operation.

5. An interleaver according to claim 3 or claim 4 wherein the computing processor is operable to compare computed values of the swapping indexes with the value K in a parallel comparison operation.

6. An interleaver according to any one of the preceding claims wherein the computing processor is operable to identify in each cycle a particular combination of swapping indexes, greater than or equal to four, having a value less than K and to select and apply a particular rule corresponding to the particular identified combination which identifies the pairs of data elements to be swapped.

7. An interleaver according to claim 6 wherein the computing processor is operable to run an algorithm having defined rules for each of the possible combinations of swapping indexes having a value less than K, and the corresponding rule is executed by the algorithm when the particular combination for a given cycle is found.

8. An interleaver according to claim 6 or claim 7 wherein the computing processor is operable to apply in

a given cycle a rule which requires use of a stored usable swapping index computed in a previous cycle to identify a data element to be subject to swapping in the given cycle.

9. An interleaver according to any one of claims 6 to 8 wherein the computing processor is operable to apply in a given cycle a rule which requires, for a usable swapping index computed in the given cycle which cannot be used in the given cycle, storage of the index for use in a subsequent cycle.

10. An interleaver according to any one of claims 1 to 9 including a single processor operable to carry out the combined functions of the computing processor and the re-arrangement processor.

11. An turbo coder including a first component encoder for encoding a first set of data elements, a second component encoder for encoding a second set of data elements and, operably coupled to the first and second component encoders to re-arrange the first set of data elements into the second set of data elements, an interleaver according to any one of the preceding claims.

12. An turbo decoder including a first component decoder for decoding a first set of data elements, a second component decoder for decoding a second set of data elements and, operably coupled to at least one of the first and second component decoders, at least one interleaver according to any one of the preceding claims 1 to 10.

13. A digital communication system including a transmitter including at least one turbo coder according

to claim 11 and a receiver including at least one turbo decoder according to claim 12.

14. A digital communication terminal including at least one of: (i) a transmitter including at least one turbo coder according to claim 11; and (ii) a receiver including at least one turbo decoder according to claim 12.

15. A method of operation in an interleaver for re-arranging in each of a plurality of L re-arrangement cycles a first set of K data elements into a second set of K data elements using a permutation vector defining an even number of swapping indexes of positions of data elements in the first set to be swapped, including computing for each of the cycles a sequence c_m that satisfies a quadratic congruence condition, computing in each cycle from the sequence c_m at least four swapping indexes defining the positions of data elements in the first set to be swapped, and swapping in each cycle by a re-arrangement processor the pairs of data elements in the first set indicated by the computed swapping indexes, the swapping of pairs of data elements in each cycle by the re-arrangement processor being carried out together in a parallel swapping operation.

Application No: GB0622764.9 **Examiner:** Owen Wheeler

Claims searched: 1-15 **Date of search:** 22 February 2007

Patents Act 1977: Search Report under Section 17

Documents considered to be relevant:

Category	Relevant to claims	Identity of document and passage or figure of particular relevance
A	-	US 2003/048206 A1 [GATHERER] Sec abstract.
A	-	US 2005/204262 A1 [BERENS] See abstract.
A	-	IEEE Transactions on Information Theory, Volume 46 Number 6, September 2000, Takeshita O Y et al, "New deterministic interleaver designs for turbo codes", pages 1988-2006.

Categories:

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.

Field of Search:

Search of GB, EP, WO & US patent documents classified in the following areas of the UKC^{X} :

Worldwide search of patent documents classified in the following areas of the IPC

H03M; H04L

The following online and other databases have been used in the preparation of this search report

EPODOC, WPI, INSPEC