

Apparatus and Method for deinterleaving.

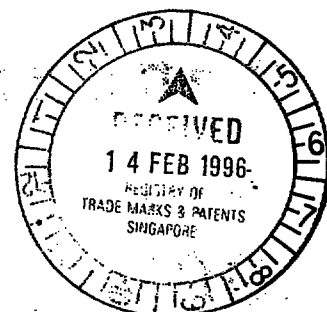
The invention relates to an apparatus and method for deinterleaving, in particular for implementing deinterleaving in a paging system.

It is common practice in communication systems for a transmitter to arrange the data bits to be transmitted into a different order prior to their transmission. This is known as interleaving. The advantage of such interleaving is that burst errors on the transmission channel can be spread throughout the data. For example, if the data consisted of a plurality of code words the interleaving would temporally disperse the data bits representing individual code words. Thus, only parts of a code word would be affected by burst errors and thus error correction of such burst errors is enhanced. Such interleaving has to be reversed or undone at the receiver, and this is known as de-interleaving.

A typical known example of a system using interleaving involves the transmission of 30 bit code words. Groups of 9 code words are interleaved and this results in 270 bit deep interleaving. In this known example the most significant bit (MSB) of code word 0 is transmitted first, then the MSB of code word 1 etc. and continuing sequentially through to the least significant bit (LSB) of code word 8.

The following array illustrates the form which the transmitted data takes.

8_0	7_0	6_0	5_0	4_0	3_0	2_0	1_0	0_0
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
8_{28}	7_{28}	6_{28}	5_{28}	4_{28}	3_{28}	2_{28}	1_{28}	0_{28}
8_{29}	7_{29}	6_{29}	5_{29}	4_{29}	3_{29}	2_{29}	1_{29}	0_{29}



The bit array is formed by writing one code word at a time into a memory column by column, and reading the memory row by row, thus carrying out the interleaving. In this array, for example, 4_{28} stands for the 28th bit of the 4th code word.

In the prior art, deinterleaving is generally performed using a random access memory (RAM) equipped with a serial bus, since the data is written into the RAM in one direction and read from it in another, relative to the columns and rows of the RAM.

If, for example, 8-bit words are used for loading the RAM, the reading out is difficult to implement. If it is necessary to use serial-form RAM, the number of memory addresses required is equal to the number of memory elements, and in the present embodiment this is $30 \times 9 = 270$. The state-of-the-art method is very complicated and takes up a lot of the address space of the RAM.

In a first aspect the present invention provides apparatus for deinterleaving comprising a memory means including memory elements arranged in an array of columns and rows, wherein the memory means has addressing means adapted to write data into the memory means row by row and to read data from the memory means column by column, and in a second aspect the invention provides a method for deinterleaving data utilising RAM, comprising sequentially selecting a row into which data is to be written,

enabling clock inputs of memory elements comprising said row,

clocking data into the memory elements of said row,

selecting a next row into which next data is to be written and deselecting said row and continuing the above steps until the data has been loaded into the RAM, and sequentially

selecting a column from which data is to be read,

enabling clock inputs of memory elements comprising said column,

clocking data from the memory elements of said column thereby forming a serial output, and

selecting a next column from which data is to be read and deselecting said column and continuing the above steps until the data has been read from the RAM.

An advantage of the present invention is that it reduces the amount of address space required by the RAM and thus reduces the overall space required by the RAM. This means that a higher degree of integration becomes possible which reduces cost, weight and volume constraints for products using the invention.

An embodiment of the invention is described below in detail by way of example only, and with reference to the accompanying figures, in which:

Figure 1 depicts the structure of a memory cell used in the circuitry in accordance with the invention,

Figure 2 depicts the structure of a memory element, used in the circuitry in accordance with the invention, made up of memory cells,

Figure 3 depicts the implementation of the writing into the memory, in accordance with the method according to the invention,

Figure 4 depicts the implementation of the reading from the memory, in accordance with the method according to the invention.

Figure 1 shows the structure of a memory element used in an embodiment in accordance with the invention. The memory element is a D flip-flop 1 having a multiplexed input. The input has AND gates 2 and 3 and OR gates 4 and 5.

Such multiplexing makes it possible to carry out the writing and the reading to the RAM in different directions, writing being done horizontally and reading vertically.

D-type flip-flop 1 has a data input D, a clock input C, a clock enable input CE and an output Q. One, other or both inputs EN0 and EN1 of OR gate 5 being HI results in a HI at input CE. Data on lines D0 and D1 is input to D when line 5 is LO or HI respectively.

Figure 2 shows the structure of the memory, comprising memory elements utilising circuitry in accordance with the invention. Memory elements 6 - 14 are arranged so that there are 9 in each row. 30 such rows are placed vertically in succession, to form a RAM comprising 30 rows x 9 columns. One row at a time, i.e. 9 bits at a time, is written into this memory. Since the data is transmitted to the receiver in serial form, it must be converted into parallel form prior to being input to the memory. Such a conversion is easy to implement using a simple well known transfer register. The writing into the memory is started from the bottom row, the address of which is 01 (HEX).

Each memory element in a row has its D0 data line connected to respective data lines 0 to 8 of data lines IN8(8:0), and its D1 data line connected to respective data lines 0 to 8 of data bus IN (0...8). Column enable lines EN0 to EN8 are input to respective EN1 inputs of memory elements (1) in each row and respective row enable lines ROW-ENAB are input to EN0 inputs of memory elements in respective rows. Clock and select signals are connected to the C and S inputs respectively of each memory element.

Figure 3 shows an embodiment of the invention in which the RAM comprises 10 RAM cells 15...24 sequentially connected together via their output buses OUT(0...8) and input data buses IN(0...8). Each RAM cell 15...24 has input to its clock enable inputs EN(0...8) enable signals ENAOUT (8:0).

Figure 4 shows three RAMS, 25, 26, 27 coupled together via their output and input buses OUT (8...0), IN (8...0) respectively. The output bus OUT (8...0) of RAM 27 is coupled via selecting logic 28...39 under control of respective enable lines ENAOUT (8:0) to D type flip flop 40 such that data stored in the RAMS 25, 26, 27 is output serially.

When writing data into the RAM, the non-active LO state of the multiplexing selection line SEL is asserted. Thus, data lines DO are active for each memory element; i.e. data bus IN8 (8:0). Respective rows are then selected by known addressing methods. The addressing method activates the ROW-ENAB line of a first row, enabling the clock inputs of the memory elements of that row. Then the first 9 bits from the series to parallel converter are clocked in parallel into that row.

Memory elements 15 - 24 are disposed such that the i^{th} memory element 15 - 24 of each row, i.e. the write input of memory element 15 - 24, is wired to the i^{th} bit of the series to parallel converter. On the address being changed, the ROW-ENAB line of the row corresponding to this new address is activated and the ROW-ENAB line of the previous row is deactivated. Again, the arrival of 9 bits is waited for, whereafter this nine-bit word is clocked into the memory elements. This procedure is continued until all of the 30 rows have been loaded.

On reading data from RAM the active HI state of the multiplexing selection line SEL is asserted. Each column of the RAM is addressed using enable signals ENAOUT (8:0) which are coupled to clock enabling lines ENO...EN8 of memory cells in respective columns 0 to 8. To read the data the addressing method activates the signal EN0 first, thus enabling the clock input of the memory elements of column 0. Thereafter the data in the column (30 bits) is sequentially read from the column.

The memory elements are coupled vertically in series so that, in reading out, an entire 30 bit-word is obtained with 30 clock

cycles from the RAM, with the MSB first. When the addresses have been selected, the entire 30-bit column is clocked out by using the CLK signal. When the whole word has been read out, a new address is selected, which activates a new ENO .. EN7 line, and the word corresponding to this address can be read out.

The selection of output lines OUT (8:0) is carried out by logic gates 28 to 39 under control of enable signals ENAOUT (8:0).

Either one or more RAMs described above can be used for deinterleaving. In the event that two RAMs are used, when one is being loaded, i.e. it is being written into, the other one is being unloaded, i.e. it is being read from simultaneously. If only one RAM is used, the memory is read into in the normal manner until even the last row has been loaded. Thereafter, the memory is unloaded while the next 9 bits are being received. This, however, sets high speed requirements on the blocks subsequent to the deinterleaving.

For a RAM thus wired, the generation of addresses is quite easy, since the rows and columns are loaded/unloaded in order. Before the loading/unloading begins, the address counters are set to zero. The desired next address is produced by clocking the address counters one step forward. Thus the memory control is very easy, and it can be effected by using two address counters.

In the present specification, a deinterleaving depth (30 x 9 bits) compatible with the ERMES system is discussed, but the same method can be used for implementing any depth of deinterleaving. The method according to the invention is very suitable for use in an ERMES paging system.

By means of the invention, deinterleaving can be implemented with very little address coding. The basic memory elements have multiplexed inputs. If these inputs are wired in the manner according to the invention, deinterleaving can be implemented in a simple manner.

In the apparatus and method in accordance with the invention, only 30 addresses are required for the writing in and 9 addresses for the reading out, i.e. in total $30 + 9 = 39$ addresses. This makes it possible to use only one RAM for effecting the deinterleaving, since only one address is required for the reading of 30 bits, instead of the 30 addresses required in a conventional method.

The invention is significant mainly when so high a speed is required in interleaving or deinterleaving that the processor in the device will not have time to carry out the operation through software. The method according to the invention speeds up the operation considerably.

In view of the foregoing description it will be evident to a person skilled in the art that various modifications may be made within the scope of the invention.

The scope of the present disclosure includes any novel feature or combination of features disclosed therein either explicitly or implicitly or any generalisation thereof irrespective of whether or not it relates to the claimed invention or mitigates any or all of the problems addressed by the present invention. The applicant hereby gives notice that new claims may be formulated to such features during prosecution of this application or of any such further application derived therefrom.

CLAIMS

1. Apparatus for deinterleaving comprising a memory means (25,26,27) including memory elements (6...14,15...24) arranged in an array of columns and rows, wherein the memory means (25,26,27) has addressing means adapted to write data into the memory means (25,26,27) row by row and to read data from the memory means (25,26,27) column by column.
2. Apparatus according to claim 1, wherein data is written into the memory means (25,26,27) in parallel.
3. Apparatus according to claim 1 or claim 2, wherein data is read from the memory means (25,26,27) in series.
4. Apparatus according to any of claims 1 to 3, wherein each row comprises a number of memory elements corresponding to the number of bits in a code word input to the RAM (25,26,27).
5. Apparatus according to any of claims 1 to 4, wherein the number of columns corresponds to the number of code words input to the RAM (25,26,27).
6. A memory element (6...14), comprising addressing means (2,3,4,5) adapted to selectively enable a clock input C of a flip flop (1) and to selectively enable one or other of two data lines input to a data input D of the flip flop (1).
7. A method for deinterleaving data utilising RAM, comprising sequentially selecting a row into which data is to be written,

enabling clock inputs of memory elements comprising said row,

clocking data into the memory elements of said row,

selecting a next row into which next data is to be written



and deselecting said row and continuing the above steps until the data has been loaded into the RAM, and sequentially

selecting a column from which data is to be read,

enabling clock inputs of memory elements comprising said column,

clocking data from the memory elements of said column thereby forming a serial output, and

selecting a next column from which data is to be read and deselecting said column and continuing the above steps until the data has been read from the RAM.

8. A method according to claim 7, wherein two RAMS are used in such a manner that when one is being written into, reading out from the other takes place simultaneously.

9. A method according to claim 7, wherein one RAM is used in such a manner that the memory is used in a normal way until the last row has been loaded, whereafter the memory is unloaded while the number of bits corresponding to the next row is being received.

10. A method according to Claim 8 or 9, wherein the selection of rows and columns is implemented by setting address counters to zero before beginning loading/unloading RAM thereafter the desired next row or column is obtained by clocking the address counters one step forward.

