



Europäisches Patentamt
European Patent Office
Office européen des brevets



(11) **EP 1 038 215 B9**

(12) **CORRECTED EUROPEAN PATENT SPECIFICATION**

Note: Bibliography reflects the latest situation

- (15) Correction information:
Corrected version no 1 (W1 B1)
Corrections, see page(s) 3
- (48) Corrigendum issued on:
03.12.2003 Bulletin 2003/49
- (45) Date of publication and mention
of the grant of the patent:
04.06.2003 Bulletin 2003/23
- (21) Application number: **98962562.9**
- (22) Date of filing: **18.12.1998**
- (51) Int Cl.7: **G06F 7/02, G06F 15/80**
- (86) International application number:
PCT/GB98/03835
- (87) International publication number:
WO 99/032961 (01.07.1999 Gazette 1999/26)

(54) **HAMMING VALUE COMPARISON FOR UNWEIGHTED BIT ARRAYS**

HAMMINGWERTVERGLEICHUNG FÜR UNGEWOGENE BITFELDER

COMPARAISON DE VALEUR HAMMING POUR RESEAUX BINAIRES NON PONDERES

- (84) Designated Contracting States:
DE ES FR GB IT NL SE
- (30) Priority: **19.12.1997 GB 9726752**
27.10.1998 GB 9823398
- (43) Date of publication of application:
27.09.2000 Bulletin 2000/39
- (73) Proprietor: **BAE SYSTEMS plc**
Farnborough, Hampshire GU14 6YU (GB)
- (72) Inventor: **KING, Douglas, Beverley, Stevenson**
Warton, Nr. Preston, Lancashire PR4 1AX (GB)
- (74) Representative: **Newell, William Joseph**
Wynne-Jones, Lainé & James
22 Rodney Road
Cheltenham Gloucestershire GL50 1JJ (GB)
- (56) References cited:
EP-A- 0 319 421 EP-A- 0 591 846
- VAN DE PANNE M ET AL: "MACHAM: A BEST MATCH CONTENT ADDRESSABLE MEMORY" PROCEEDINGS OF THE PACIFIC RIM CONFERENCE ON COMMUNICATIONS, COMPUTERS AND SIGNAL PROCESSING, VICTORIA, JUNE 1 - 2, 1989, 1 June 1989, pages 612-615, XP000077554 INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS**

EP 1 038 215 B9

Note: Within nine months from the publication of the mention of the grant of the European patent, any person may give notice to the European Patent Office of opposition to the European patent granted. Notice of opposition shall be filed in a written reasoned statement. It shall not be deemed to have been filed until the opposition fee has been paid. (Art. 99(1) European Patent Convention).

Description

5 **[0001]** This invention relates to methods and apparatus for determining the Hamming values of binary vectors or arrays in general and in particular, but not exclusively, to methods and apparatus for use in determining the Hamming value relationship of a set of weightless bits otherwise referred to herein as a weightless tuple or weightless vector. The invention also relates to apparatus and methods for determining the Hamming value relationship of two thermometer codes or weightless vectors. The invention further extends to a Hamming value comparator for comparing the Hamming value relationship of two 2-dimensional arrays of data. The invention still further extends to sum and threshold devices using Hamming value comparators of this invention.

10 **[0002]** The apparatus and methods described herein may usefully incorporate, utilise, be used with or be incorporated into any of the apparatus or methods described in our co-pending WO99/33184, WO99/33019, WO99/33175, WO99/32962, the entire contents of which are incorporated herein by reference.

15 Terminology

[0003] The term "Hamming value" is used to define the number of logic 1's set in 1-dimensional arrays such as a binary number, tuple, vector, or 2 or higher dimensional arrays. The Hamming value relationship of two binary numbers or arrays indicates which has the greater Hamming value or whether the Hamming values are the same.

20 **[0004]** The term "weighted binary" is used in the conventional sense to indicate that successive bit positions are weighted, particularly 16, 8, 4, 2, 1, although other weighted representations are possible. "Weightless binary" is a set of binary digits 1 and 0, each representing just "1" and "0" respectively. There is no least significant bit (LSB) or most significant bit (MSB). The set of bits may be ordered or without order. If all the 1's are grouped together e.g. [111000] then the code is referred to as a thermometer code, thermocode or bar graph code, all collectively referred to herein as "thermometer codes".

25 **[0005]** The term thermometer code is also used herein to describe arrays where the set bits are aggregated around a preset bit within the tuple, rather than aggregating towards one of the ends of the tuple. Likewise the term is also used to describe 2, 3 or higher dimensional arrays in which the set bits are clustered or aggregated around a preset focal bit position which may be anywhere within the array, e.g. at one corner or in the centre in a 2 or 3 dimensional array.

30 **[0006]** A set of weightless bits is referred to herein as a "weightless tuple" or "weightless vector" and these terms are not intended to be restricted to ordered sets.

35 **[0007]** In traditional neural networks, a real-valued synaptic value is multiplied by a synaptic connection strength or weight value, and summed with other similarly treated synapses before they are all summed and thresholded to form a neural output. The weight value is a real-valued synaptic connection strength and hence the common usage of the term "weighted neural network". However, it is also possible to have binary RAM-based neural networks that do not employ real-valued connection weights but instead rely on the values of the binary bits being either 0 or 1. Accordingly, there are two contexts of weightlessness: without synaptic connection strength, and without binary code weighting. The arrangements described herein employ weightless binary manipulation mechanisms and may be used to engineer weightless artificial neural networks, otherwise referred to as weightless-weightless artificial neural networks.

40 **[0008]** In one context, this invention is concerned with the comparison of two weightless vectors in terms of their Hamming values. This process is broadly equivalent to the function of a binary neuron. If the neuron receives a vector, A, of weightless synaptic values (e.g. [10110010]), and a vector, T, of weightless neural threshold values (e.g. [00101000]), the neuron may be required to fire, when the Hamming value of A is greater than the Hamming value of T. In this example, the threshold, T, can be thought of as a set of inhibitory synaptic values which must be exceeded if the neuron is to be fired. This is one particular example of an instance where it is required to determine the Hamming value relationship between two binary vectors, but there are very many other types of systems requiring this or similar processing. For example, this comparison and the other techniques disclosed herein may be used in flight control systems, voting systems with redundancy, safety critical systems, telecommunications systems, decision making systems, and artificial intelligence systems, such as neural networks.

45 **[0009]** In a prior art arrangement, a state machine based system is used to count the number of 1's set and digital arithmetic units are used for the binary comparison. For each weightless vector, each bit is scanned sequentially and a counter or arithmetic register incremented accordingly to evaluate the Hamming value. Thereafter, the contents of the respective counters or arithmetic registers are compared to determine the Hamming value relationship. This prior art technique is suited to implementation in software, using a microprocessor or similar state machine.

50 **[0010]** However, this technique is slow and prone to both conductive and emissive radio frequency interference (RFI) as it relies principally on clocks, counters and microprocessors. Both the speed of operation and susceptibility to disruption or corruption by other noise makes such a system ill-suited for safety critical systems such as flight control systems. By way of illustration, a flight control system may need to make rapid decisions based on the instantaneous "HIGH" or "LOW" state of several transducers reporting various aspects of the aircraft operation; corruption of one of

the inputs to the decision making process (caused e.g. by noise or RFI disruption) could result in misinterpretation of one of the inputs with the result that the decision making processor or neural network make the wrong decision with potentially catastrophic consequences.

[0011] British Patent No. GB-A-1,588,535 discloses an arrangement in which file words are searched one at a time and the Hamming distance between a file word and a search word measured. It is noted that the Hamming distance (i.e. the number of bits in which the two words differ), is quite different from the Hamming value comparison, which typically determines which of two binary vectors or tuples has the greatest number of logic 1's set.

[0012] Accordingly, disclosed herein are various mechanisms for determining the Hamming value relationships of two binary vectors which use asynchronous Boolean logical bit cell manipulation schemes which are very fast for small weightless systems. In addition these tend to be more RFI resilient and more fault-tolerant than the weighted binary alternatives.

[0013] In the first described arrangement, two weightless binary vectors are first converted separately into thermometer code and thereafter the thermometer codes are compared by a thermocode comparator to determine the Hamming value relationship.

[0014] In the two subsequent related schemes, two weightless vectors are compared in parallel, without requiring separate conversion into thermometer code and subsequent comparison. However, the thermometer code converter and the thermocode comparator may each be used in different applications and the invention extends to these per se.

[deletion(s)]

[0015] According to one aspect of this invention there is provided a Hamming value comparator for providing an output indicative of the Hamming value relationship of a first one- or more-dimensional weightless array of weightless bits and a second one- or more-dimensional weightless array of weightless bits,

wherein the Hamming value of a given weightless array is the number of logic 1's therein, and the Hamming value relationship of two weightless arrays indicates which has the greater Hamming value or whether the Hamming values thereof are the same,

said Hamming value comparator comprising:-

means for applying the bits from said arrays to processing means comprising a plurality of layers of (n+n)-bit manipulation cells, each manipulation cell having a first n-inputs for receiving respective bits corresponding to respective bits in said first array, and a second n-inputs for receiving bits corresponding to respective bits in said second array, a first n-outputs and a second n-outputs corresponding to said first and second n-inputs respectively,

wherein each manipulation cell comprises logic means for performing at least one of a bit shift operation and a bit elimination operation,

the (n+n)-bit manipulation cells being interconnected by passing the outputs of each layer apart from the final layer to respective inputs of the next or the next-but-one layer,

wherein in successive layers, the bits set at the output of said bit manipulation cells migrate in successive layers towards a common bit position relative to said layers, with the final layer providing an output indicative of said Hamming relationship.

[0016] The Hamming value comparator preferably includes processing means comprising a plurality of layers of bit manipulation cells each having a plurality of inputs and a plurality of outputs, input means for passing corresponding bits from each of the two binary arrays to respective inputs of bit manipulation cells in the first layer of said processing means, the bit manipulation cells being interconnected by passing the outputs of each layer apart from the last to respective inputs of the next or next-but-one layer, thereby to provide an output indicative of said Hamming value relationship.

[0017] The comparator is particularly intended for use with weightless arrays or words, but its use for weighted arrays or words is not excluded.

[0018] Each bit manipulation cell may include four inputs and four outputs, the bit manipulation cells in said first layer being arranged such that, for each bit manipulation cell, two inputs each receive a respective bit from one of said binary arrays, and the other two inputs each receive a respective bit from the other binary array.

[0019] Each of said two inputs in the bit manipulation cells in said first layer preferably receives adjacent bits of the corresponding binary array, taken on an even boundary. Each manipulation cell preferably performs an operation on the bits from said first and second binary arrays received at the respective inputs, whereby a set bit is shifted in a common direction and correspondingly positioned set bits are fully or partially eliminated. Thus, each of said bit manipulation cells may effect a shift and full elimination process, as set out in the truth table below (or the inverse thereof):-

A in L	A in R	B in L	B in R	A out L	A out R	B out L	B out R
0	0	0	0	0	0	0	0

EP 1 038 215 B9 (W1B1)

(continued)

5
10
15
20
25

A in L	A in R	B in L	B in R	A out L	A out R	B out L	B out R
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	0	0	0	0	0
0	1	1	1	0	0	1	0
1	0	0	0	1	0	0	0
1	0	0	1	0	0	0	0
1	0	1	0	0	0	0	0
1	0	1	1	0	0	1	0
1	1	0	0	1	1	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	0	0	0	0

wherein AinL, AinR, BinL, BinR, are the four inputs to the cell and AoutL, AoutR, BoutL, BoutR are the four outputs.
[0020] Alternatively, each of said bit manipulation cells may effect a shift and partial elimination process as set out in the truth table below or the inverse thereof:-

30
35
40
45
50
55

A in L	A in R	B in L	B in R	A out L	A out R	B out L	B out R
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	0	1	0	1	0
0	1	1	1	0	0	1	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	1	0
1	0	1	0	0	0	0	0
1	0	1	1	0	0	1	0
1	1	0	0	1	1	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	0	0	0	0

[0021] Preferably, each bit manipulation cell has a first plurality of inputs for receiving respective bits from one of

said binary words, a second plurality of inputs for receiving respective bits from the other of said binary words, a corresponding plurality of outputs, and logic means for outputting on a respective pre-selected one of said outputs a set bit if the Hamming value of the bits on said first plurality of inputs is greater than the Hamming value of the bits set on said second plurality of inputs.

5 [0022] Preferably the bit manipulation cell has a first plurality of inputs for receiving respective bits from one of said binary arrays and a second plurality of inputs for receiving respective bits from the other of said binary arrays, said bit transformation means having a corresponding plurality of outputs, and logic means for outputting on a pre-selected one of said outputs a set bit if the Hamming value of the bits set on said first plurality of inputs is greater than the Hamming value of the bits set on said second plurality of inputs.

10 [0023] In a preferred embodiment said bit transformation means has a first plurality of inputs (AINL AINR) for receiving respective bits from one of said binary arrays and a second plurality of inputs (BINL BINR) for receiving respective bits from the other of said binary arrays, said bit transformation means having a corresponding plurality of outputs (AOUTL AOUTR BOUTL BOUTR), and logic means applying a Boolean logic operation for outputting on a pre-selected one of said outputs a set bit if the number of the logic 1's set on said first plurality of inputs is greater than the Hamming value of the bits set on said second plurality of inputs.

15 [0024] In a preferred embodiment, this invention provides a Hamming value comparator, for comparing the Hamming values of two 2-dimensional arrays of binary data, said comparator comprising a plurality of layers of bit manipulation cells, each cell including at least two inputs and two outputs and being adapted to shift set bits towards a preset bit position, and being interconnected between layers to effect at least one of bit shifting and bit elimination, thereby to obtain an output indicative of the Hamming value relationship of said arrays.

20 [0025] The invention also provides a method of determining which, if either, of a first one- or more-dimensional weightless array of weightless bits and a second one- or more-dimensional weightless array of weightless bits, has the greater number of logic 1's therein, said method comprising:-

25 (i) applying the bits from said arrays to processing means comprising a plurality of layers of (n+n)-bit manipulation cells, each manipulation cell having a first n-inputs for receiving respective bits corresponding to respective bits in said first array, and a second n-inputs for receiving bits corresponding to respective bits in said second array, a first n-outputs and a second n-outputs corresponding to said first and second n-inputs respectively,

30 wherein each manipulation cell comprises logic means for performing at least one of a bit shift operation and a bit elimination operation,

the (n+n)-bit manipulation cells, being interconnected by passing the outputs of each layer apart from the final layer to respective inputs of the next or the next-but-one layer,

35 wherein in successive layers, the bits set at the output of said bit manipulation cells migrate in successive layers towards a common bit position relative to said layers, with the final layer providing an output indicative of said Hamming relationship.

[0026] Whilst the invention has been described above, it extends to any inventive combination of the features set out above or in the following description.

40 [0027] The invention may be performed in various ways, and, by way of example only, various embodiments thereof will now be described in detail, reference being made to the accompanying drawings which utilise the conventional symbols for logic gates and, in which: -

Figure 1 is a circuit diagram of a first embodiment of weightless binary comparator in accordance with this invention for determining the Hamming value relationship of two weightless binary vectors;

45 Figures 2 (a) and 2 (b) are circuit diagrams of two of the bit manipulation cells for use in the weightless binary comparator of Figure 1, one for performing full shift and full Hamming elimination and the other for performing partial shift and Hamming elimination respectively;

Figure 3 is a worked example of the weightless binary comparator of Figure 1 showing the bits set on the inputs and outputs of the bit manipulation cells in response to the input of two typical weightless binary vectors;

50 Figure 4 is a circuit diagram of a second embodiment of weightless binary comparator in accordance with this invention for determining the Hamming value relationship of two binary vectors;

Figure 5 is a circuit diagram of one of the bit manipulation cells of the weightless binary comparator of Figure 4 for performing a full shift and a partial Hamming elimination;

55 Figure 6 is a worked example-of the weightless binary comparator of Figure 4, showing the bits set on the inputs and outputs of the bit manipulation cells in response to the input of two typical weightless binary vectors;

Figure 7 is a circuit diagram of a 2 x 2-bit manipulator cell for performing shift and Hamming elimination;

Figure 8 is a circuit diagram of a 2 x 4-bit manipulator cell performing shift and Hamming Elimination;

Figure 9 is a circuit diagram of an array of the manipulation cells of Figure 8 providing a Hamming value comparator

for comparing two 12-bit weightless inputs;
 Figure 10 is a worked example of the array of Figure 11;
 Figure 11 is a further embodiment of a Hamming value comparator for comparing two 8-bit weightless inputs;
 Figure 12 is an enlarged view of part of Figure 11 for explaining the structure thereof;
 Figure 13 is an enlarged view of the back end decoder used in the embodiment of Figure 11;
 Figure 14 is a diagrammatic representation of the odd and even layers, and the cell number identification used in
 an embodiment of a planar Hamming value comparator structure;
 Figure 15 is a diagram of the 2 x 2-bit manipulation cell used in the embodiment of the planar Hamming value
 comparator structure of Figure 14;
 Figure 16 is a diagram of a 2 x 4-bit manipulation cell for use in the embodiment of Planar Hamming value com-
 parator of Figure 14;
 Figures 17 and 18 are worked examples of the planar Hamming value comparator of Figure 14, and
 Figure 19 is a diagram of a sum and threshold (SAT) device which may be constructed by a Hamming value com-
 parator as described herein.

DESCRIPTION OF PREFERRED EMBODIMENTS

Hamming Value Comparator for Thermometer Codes

Hamming Value comparator for Weightless Tuples

[0028] Referring to the embodiments of Figures 1 to 6, here the Hamming value relationship of two N-tuple weightless
 vectors is determined using a single comparator. In both embodiments, a four input, four output modular bit manipulation
 cell is defined to operate on two elements of each of the N-tuples A and B, and the manipulation cells are built into an
 architecture that performs the required comparison in parallel.

[0029] Referring firstly to the embodiment of Figures 1 and 2, each of the bit manipulation cells 22 has 4 inputs and
 4 outputs. In the input layer, and throughout the network, two inputs of each cell 22 are associated with digits of the A
 vector and two inputs are associated with digits of the B vector. In Figure 1, as viewed, the lower two inputs of each
 cell 22 are associated with the A vector, and the upper two inputs are associated with the B vector.

[0030] As is apparent from Figure 1, the number of bit manipulation cells 22 in each layer alternates between even
 and odd, although the seventh and succeeding stages are truncated because the relevant outputs are taken from a
 single manipulation cell 22 in the final stage, and so the bit manipulation cells removed by truncation are redundant
 as these cells would not contribute to this single manipulation cell. The connections between bit manipulation cells 22
 in adjacent layers are staggered such that, apart from in the input layer, each bit manipulation cell 20 receives respective
 inputs from two bit manipulation cells in either one or two previous layers depending on whether the bit manipulation
 cell is at either end of the layer and whether the layer has an odd or even number of cells 22.

[0031] Referring to Figures 2(a) and 2(b), these show two alternative logic circuits providing the required manipula-
 tion. For ease of comparison, it should be noted that the cell inputs labelled (a, b, c, d) in Figure 1 correspond to inputs
 labelled (AinL, AinR, BinL, BinR) in Figures 2 (a) and 2 (b), and outputs labelled (Ya, Yb, Yc, Yd) in Figure 1 correspond
 to outputs labelled (AoutL, Aout R, BoutL, BoutR) in Figure 2 (a). In Figure 2(a) the arrangement comprises inverters
 24, **AND** gates 26 and **OR** gates 28.

[0032] The reduced Boolean equations for AoutL, AoutR, BoutL, and BoutR are stated below, where q, r, s and t
 represent AinL, AinR, BinL and BinR respectively. The Boolean operators "**NOT**", "**AND**" or "**OR**" are represented by !,
 & and #. respectively.

$$A \text{ out L} = q \& r \& !s$$

$$\# r \& !s \& !t$$

$$\# q \& !s \& !t$$

$$\# q \& r \& !t$$

$$A \text{ out R} = q \& r \& !s \& !t$$

$$B \text{ out L} = !q \& !r \& s$$

EP 1 038 215 B9 (W1B1)

!r & s & t

!q & s & t

!q & !r & t

B out R = !q & !r & s & t

(Equation Set (1))

5

10 **[0033]** Each of the bit manipulation cells 22 performs a combinatorial full shift and full Hamming elimination operation, as shown in the following truth table in which the four inputs to the cell are designated AinL, AinR, BinL, and BinR with the corresponding outputs being identified AoutL, AoutR, BoutL and BoutR reading both inputs and outputs in the same direction, left to right.

15

A in L	A in R	B in L	B in R	A out L	A out R	B out L	B out R
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	0	0	0	0	0
0	1	1	1	0	0	1	0
0	0	0	1	0	0	0	
1	0	0	1	0	0	0	0
1	0	1	0	0	0	0	0
1	0	1	1	0	0	1	0
1	1	0	0	1	1	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	0	0	0	0

20

25

30

35

40 Inspection of the truth table will show that the bit manipulation cells shift a set bit from a right hand position (AinR or BinR) to the left if the bit of that position has not already been set, but it does not transfer bits across the boundary between AinR and BinL; for example (0001) is mapped to (0010), whereas (0010) is unaltered. Likewise (0100) is mapped to (1000). In addition, a cancellation operation is carried out whereby a set bit on one side of the A:B boundary cancels a set bit (in either position) on the other side of the boundary, thus unsetting both bits. This is illustrated for example by the following mappings; (0101) to (0000); (0110) to (0000); (1101) to (1000), and (1111) to (0000). It will also be noted from the truth table that the 16 possible input states have been reduced to 5 possible output states.

45

[0034] An alternative form of bit manipulation cell which performs a generally similar operation, but with some differences, is shown in Figure 2(b). This is made up of AND gates 30, OR gates 32 and inverters 34 as shown and performs a combinatorial partial shift and full Hamming . elimination. Thus an input (1110) to the Figure 2(b) cell, which in the cell of Figure 2(a) would give an output of (1000), gives instead the output (0100). However the cell of Figure 2 (b) may be built into an array and used to achieve Hamming value comparison.

50

[0035] The AoutL and BoutL bits of the final bit cell manipulator are used to determine the comparator outputs as defined by the truth table below. AoutL and BoutL cannot be high simultaneously.

55

EP 1 038 215 B9 (W1B1)

(Ya) Final AoutL	(Yb) Final BoutL	A>B	B>A	A-B
0	0	FALSE	FALSE	TRUE
1	0	TRUE	FALSE	FALSE
0	1	FALSE	TRUE	FALSE

5

10

15

20

[0036] Although the comparator is described by reference to use with weightless binary tuples, it will of course be appreciated that it will operate with thermometer code.

[0037] Figure 3 is a worked example showing comparison of the tuple A (10100110) with a Hamming value of 4, with B (00100000) having a Hamming value of 1. At the final output stage 45, which shows values (1100) it will be seen that AoutL is high and BoutL is low indicating that the Hamming value of A is greater than that of B, as defined in the truth table above.

[0038] Referring now to the embodiment of Figures 5 and 6, the array of bit manipulation cells 36 and their interconnections are the same as that of the embodiment of Figure 4, but the logic implemented in each cell is different and the logic operations carried out on the output of the bit manipulation cell and final layout are different. The logic circuit for the bit manipulation cell 36 is shown in Figure 5 and comprises six AND gates 38 and two OR gates 40 and four inverters 42 providing the simpler logic operations for AoutL, AoutR, BoutL, and BoutR as stated below, where q, r, s, t, represent AinL, AinR, BinL and BinR respectively. The Boolean operators "NOT", "AND" and "OR" are represented by !, & and # respectively.

25

$$A \text{ out L} = q \& !s$$

$$\# r \& !t$$

$$A \text{ out R} = q \& r \& !s \& !t$$

30

$$B \text{ out L} = !q \& s$$

$$\# !r \& t$$

35

$$B \text{ out R} = !q \& !r \& s \& t \quad (\text{Equation set (2)})$$

[0039] The truth table is set out below in which it will be seen that, for two bits of tuples A and B, taken on an even boundary of two, the truth table performs a combinatorial full shift and partial Hamming elimination operation.

40

A in L	A in R	B in L	B in R	A out L	A out R	B out L	B out R
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	0	1	0	1	0
0	1	1	1	0	0	1	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	1	0
1	0	1	0	0	0	0	0

55

EP 1 038 215 B9 (W1B1)

(continued)

A in L	A in R	B in L	B in R	A out L	A out R	B out L	B out R
1	0	1	1	0	0	1	0
1	1	0	0	1	1	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	0	0	0	0

[0040] The truth table is the same as that for the previous embodiment except for the mappings of (0110) to (1010) and the mapping of (1001) to (1010). Accordingly, we refer to this arrangement as a "partial" Hamming elimination operation, that is to say set bits to either side of the A:B boundary are only cancelled if they are in the corresponding positions prior to the shift operation. Thus for example (0101) and (1010) each map to (0000), but (0110) and (1001) (which in the previous embodiment mapped to (0000)) in this embodiment each map to (1010). Figure 6 is a worked example for comparison of a weightless tuple A(11001011) having a Hamming value of 5 and a tuple B (00111100) having a Hamming value of 4. At the final stage 65 of the manipulation cells 57,, the bit set on AoutL indicates that the Hamming value of A is greater than that of B. The logic table for deriving the Hamming value relationships from AoutL and BoutL are shown in the following table:-

Final AoutL	Final BoutL	A>B	B>A	A=B
0	0	FALSE	FALSE	TRUE
0	1	FALSE	TRUE	FALSE
1	0	TRUE	FALSE	FALSE
1	1	FALSE	FALSE	TRUE

[0041] It should be noted that this comparison also works with thermometer codes.

[0042] The number of cells and number of layers required depends on the width of the words to be compared and can be determined empirically, to give a stable output for all input states.

[0043] It should be appreciated that, if required, the arrangement could be made larger or smaller to deal with tuples of different length, or if required, a tuple may be padded out with 0's.

[0044] The invention also extends to arrays achieving the same function using different bit manipulation cells, or employing De Morgan equivalents of the items shown. It should also be appreciated that the arrangements may use inverse logic and De Morgan equivalents. Furthermore, the invention may be extended to allow comparison of more than two binary tuples by suitable comparison and combination stages. For example three tuples A, B, and C, may be compared by making the comparison A:B, A:C and B:C and then ordering the tuples A,B,C, knowing the results of the comparisons.

[0045] The Hamming value comparators of Figures 1 to 6 inclusive use an array of common 2 x 2-bit manipulation cells operating on two bits from each tuple to determine the Hamming value relationship of two binary tuples without requiring prior conversion to thermometer code. These principles may also be modified and applied to arrangements which use bit manipulation cells which operate on more than 2 bits, hereafter referred to as n-bit cells. The generic technique for systems using n-bit cells is described below and an example is given for a 2 x 4-bit cell.

[0046] The general principle is that the two weightless strings (A,B) for Hamming value comparison are segmented into n-tuples, where n=2,3,4 are suitable. Larger values of n are theoretically possible, but are cumbersome to implement. If the -length of the binary strings is indivisible by n, then the strings are padded with additional 0's. In the input layer, and odd layers, the tuples are split on even boundaries, whereas in the even layers the tuples are split on odd boundaries. This staggered effect is to allow set bits progressively to migrate across even and odd boundaries in successive layers. The bit position within each tuple, indeed the order of the tuples within each binary string, is irrelevant because the bits are weightless. The n-tuples comprising A and B are fed into a cellular lattice structure having a first stage which effects thermometer code conversion followed by a second stage which effects Hamming elimination. Each tuple is converted into thermometer code using a Boolean mapping as described in WO99/33184.

[0047] Thus, for a two bit cell, the arrangements of Figures 1 to 3 employ a two bit cell in a lattice structure, but it is also possible to provide a similar structure which employs 3,4 etc bits. As noted above, for a 2 bit cell, the manipulator

EP 1 038 215 B9 (W1B1)

equations are:

$$Y_a = a \# b$$

$$Y_b = a \& b, \quad (\text{Equation Set (3)})$$

where # is the OR function and & is the AND function, a and b are the inputs, and Y_a and Y_b are the corresponding outputs.

[0048] For a three bit cell; with inputs a, b, c, and outputs Y_a , Y_b , Y_c , the manipulator equations are:-

$$Y_a = a \# b \# c$$

$$Y_b = (b \& c) \# (a \& c) \# (a \& b)$$

$$Y_c = a \& b \& c \quad (\text{Equation Set (4)})$$

[0049] Similarly for a four bit cell, with inputs a, b, c, d, and outputs Y_a , Y_b , Y_c , Y_d , the manipulator equations are:-

$$Y_a = a \# b \# c \# d$$

$$Y_b = (d \& c) \# (d \& b) \# (d \& a) \# (c \& b) \# (c \& a) \# (b \& a)$$

$$Y_c = (d \& c \& b) \# (d \& c \& a) \# (d \& b \& a) \# (c \& b \& a)$$

$$Y_d = a \& b \& c \& d \quad (\text{Equation Set (5)})$$

[0050] In the second stage, bits set in the respective bit positions of the tuples A and B are eliminated by a process referred to herein as Hamming elimination. Thus the thermometer coded tuples A and B (1100) (1000), become (0100) and (0000) respectively. The Boolean equation that performs this Hamming elimination at each bit position is $y_a = a \& !b$ for the A tuples, and $y_b = b \& !a$ for the B tuples. Examples of the complete cell, for $n = 2$ and $n = 4$ are given with Figure 7 showing a dual 2 input structure (identical to the structure of Figure 2 (b)), and Figure 8 showing a dual 4 input structure.

[0051] Inspection of Figures 7 and 8 shows that each structure 44, 46 respectively includes a first stage 48, 50, of **AND** and **OR** gates which effects the conversion into thermometer code and a second stage 52, 54 which effects the Hamming Elimination. The second stages of each are similar. The similarity of the first stage of the two input structure and the four input structure to the thermometer code converter structure of WO99/33184 will be noted.

[0052] A lattice structure of bit cell manipulators is then formed to execute the Boolean processes in parallel. Figure 9 shows an example for a Hamming value comparator made up of an array of 2×4 -bit manipulation cells 56 of the type illustrated in Figure 8, to compare the Hamming values of two 12-bit words A (comprising bits $a_0 \dots a_{11}$) and B (comprising bits $b_0 \dots b_{11}$). In this array, the inputs and outputs are rearranged so that they are in line instead of at 90° as in Figure 8, but the logical operations are unchanged. As previously the structure has alternate odd and even layers of bit manipulation cells 56 and the cells in adjacent layers are staggered to provide an offset or overlap, to ensure that the set bits migrate through the array as required. The array is truncated as before, and respective two's of the outputs from the final cell are ORed together at a final stage decoder 58. If a bit is set on the output marked "B" but not on that marked "A", then the Hamming value of B is greater than that for A. If a bit is set on the output marked "A" but not on that marked "B", then the Hamming value of B is less than that for A. If the bits on the outputs marked "A" and "B" are the same, then the Hamming values of A and B are the same.

[0053] Accordingly, at the decoder 58 the OR, EX-NOR, and AND gates 60, 62, 64 and the inverters 66 set the appropriate bit on one of the decoder outputs A less than B ($A < B$), A greater than B ($A > B$) or A equal to B ($A = B$). Figure 10 is a worked example for the circuit of Figure 9, comparing two twelve bit binary tuples (111011000000) and (000000111111) respectively and determining the correct result.

[0054] It is also possible to design comparators which shift and compare towards the centre of the strings. This design is faster and uses less logic. Referring now to Figures 11 to 13, there is illustrated an embodiment of Hamming value comparator which uses an array of 2×2 -bit manipulation cells taking two bits from each binary tuple - here each of eight bits - and performing shift and elimination operations. -In this arrangement the bits are shifted towards the

middle of the tuple rather than to one end. The similarities with the original 2-bit manipulation cell will be apparent.

[0055] It will be noted from the representation in Figure 12 that the array is made up of a number of cells designated respectively as odd shift cells 68, even shift cells 70 and Hamming Elimination blocks 72. The odd and even shift cells 68 are symmetrically placed about the mid-line of the array, with shift cells below the mid line shifting upwards and shift cells above the mid line shifting downwards. The Hamming Elimination blocks 72 are located on the array mid line and are special manipulation cells operating on four bits organised as a block of 2 x 2 bits and made up of a AND gates with selected inputs inverted as shown. The elimination comprises a vertical stage 74 and a cross stage 76 and the stages can be in either order. With reference to the schematic representation in Figures 11 or 13, the following operations apply, with the inputs and outputs of the array being designated (A1,B1,A2,B2) reading from the bottom to the top of the cell as viewed: -

- If A1 and B1 are 1 they will both become 0 (vertical)
- If A2 and B2 are 1 they will both become 0 (vertical)
- If A1 and B2 are 1 they will both become 0 (cross)
- If A2 and B1 are 1 they will both become 0 (cross)

with any other values passing straight through.

[0056] A backend decoder 78 is required to provide the A>B, A<B and A=B outputs from the values received from the final Hamming Elimination block and the logic circuits for this are shown in Figures 11 and 13.

Planar Hamming Value Comparator

[0057] Referring now to the embodiment of Figures 14 to 18, there is illustrated a Boolean lattice structure for planar Hamming value comparison. The arrangements described above with reference to Figures 1 to 13 effect the Hamming value comparison of 1-dimensional strings of weightless binary bits of various word lengths using an array of manipulation cells which perform a full or partial shift and full or partial bit elimination operation on selected bits from the two 1-dimensional strings. This technique has been developed to provide an arrangement for comparing the Hamming value relationships of two equally sized square weightless arrays.

[0058] Referring to Figure 14, the planar Hamming value device is made up of alternate odd and even layers 80, 82. Each layer in this example processes a first array of 6 x 6 bits (identified by nine lots of capital letters (A, B, C, D) and coordinates R₁ to R₆ and C₁ to C₆) and a second array of bits (identified by nine lots of lowercase letters (a, b, c, d) and coordinates r₁ to r₆, c₁ to c₆). Each layer effects a shift of set bits towards the upper top left corner as viewed. Each layer also effects a Hamming Elimination between, the arrays, similar to that performed in the 1-dimensional arrays, and as to be illustrated below, so that in successive layers the bits are shifted to the top left hand corner of the array and Hamming elimination is performed. Eventually, in the final layer a single bit at the output indicates the Hamming value relationship of the two arrays.

[0059] It should be noted that each layer is symmetrical about the diagonal running from top left to bottom right, with the cells on and above the diagonal running:-

$$\begin{pmatrix} A & C \\ B & D \end{pmatrix}$$

and those below running:-

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix}$$

[0060] In the first layer 80, each 6 x 6 array of inputs is segregated into nine 2 x 2 tuples. For one input array there are nine tuples each identified {a, b, c, d} (though a, b, c, d will not usually be the same between tuples), and for the other array there are nine tuples {A,B,C,D}.

[0061] In the first layer, two lots of 2x2 tuples from the same part of the input arrays are combined, as indicated by the concentric arrangement. Thus at each tuple position the tuples (a, b, c, d) in the centre and the tuples {A,B,C,D} around the centre are processed together so that, in effect, the first layer 80 is made up of an array of nine 8-bit

manipulator cells 84. The 8-bit manipulation cell 84 is shown in Figure 16 and is the same as that shown in Figure 8 above except, for aiding understanding, the labelling has been changed to reflect the identification of the tuples by capital and lowercase letters. The 8-bit manipulation cell receives the two tuples {a,b,c,d} and {A,B,C,D}, performs a shift and elimination operation and outputs the result as tuples {Ya, Yb, Yc, Yd} and {YA, YB, YC, YD}. The same operations are performed in all of the other odd layers.

[0062] In the second and even layers 82, there is an offset and overlap, and feedthrough 86 at the four corners. To achieve the required offset and overlap, in this example the even layers are made up of four 8-bit manipulator cells 84 of the type used in the odd layers, grouped centrally and symmetric about the usual top left to bottom right diagonal, and eight peripheral 4-bit manipulator cells 88 spaced about the periphery, and eight feedthroughs 86 at the corners. The 4-bit manipulation cells in the even layers operate on two tuples {A, B} and {a,b}, performing a shift and Hamming elimination as described in Equation Set (1). Figure 15 shows the logic circuit of earlier Figures 2 and 7 labelled with {A,B} and {a,b} input tuples and {YA.YB} {Ya Yb} output tuples respectively.

[0063] Thus a bit with coordinates (C_m, R_n) in an odd layer will map to (C_m, R_n) in the even layer, and (c_m, r_n) in an odd layer will map to (c_m, r_n) in an even layer. So, for example, the outputs in the top right of the first layer with coordinates C_6, R_1 and (c_6, r_1) will be YC and Yc respectively as viewed in Figure 16. The elements at these co-ordinates in the second layer are feedthroughs 86, and so YC and Yc will pass unchanged through the second layer to form the "C" and "c". inputs to the top right hand manipulation cell 84 in the third layer. The mapping of the other first layer outputs to the second or subsequent layer inputs can be readily determined in a similar manner.

[0064] Figures 17 and 18 show worked examples for the 6x6 planar Hamming value structure just described, for two sets of 6x6 inputs respectively. It should be noted that in Figure 17, the output is stable after five layers, whereas Figure 18 a stable output is achieved after six layers.

[0065] In Figure 17, the inner array of inputs (i.e. the lower case array of Figure 14) has a Hamming value of 18, whereas the outer array of inputs (i.e. the Capital letter array of Figure 14) has a Hamming value of 14. The outputs of each of the layers are shown together with the respective Hamming values, resulting from elimination, for each array. The output of the array is read by monitoring the bits at positions (C_1, R_1) and (c_1, r_1) . If the Hamming value of the capital letter array is greater, then at the output of the final layer the bit at (C_1, R_1) will be set (as in Figure 18(b)). If the Hamming value of the lowercase array is greater, then the bit at position (c_1, r_1) will be set (as in Figure 17). If the bits at both positions are the same, this indicates that the Hamming values of both arrays are the same.

[0066] Figures 14 to 16 above show an example of a cell layering structure for operating on two 6 x 6 arrays of bits. It is however possible to provide structures for smaller or larger arrays using the same general principles. If desired, larger operating blocks may be used instead of the 8 bit blocks used in this example.

[0067] It should be noted that the techniques described above with respect to aggregate code formation and planar Hamming value structure can be extended beyond two dimensional arrays to provide three and higher dimensional arrays (cubic and hyperplane systems). The Hamming value comparators described above may be used to perform a sum-and-threshold (SAT) function in existing and novel weightless neural networks. As shown in Figure 19, a tuple of neural data, and a tuple defining a neural threshold may be supplied to a Hamming value comparator 90 of one of the types discussed above, whether of one, two or of greater dimension. The relevant output of the comparator is then checked to see whether a bit is set on the output which indicates that the neural data has exceeded the neural threshold. The relevant output is then viewed as the single output of the neuron which is taken to have "fired" if the bit has set. Naturally one of the other outputs could be monitored if the neuron was intended to respond to another condition.

Claims

1. A Hamming value comparator for providing an output indicative of the Hamming value relationship of a first one- or more-dimensional weightless array of weightless bits $(a_1...a_8)$ and a second one- or more-dimensional weightless array of weightless bits $(b_1...b_8)$,

wherein the Hamming value of a given weightless array is the number of logic 1's therein, and the Hamming value relationship of two weightless arrays indicates which has the greater Hamming value or whether the Hamming values thereof are the same,

said Hamming value comparator comprising:-

means for applying the bits from said arrays to processing means comprising a plurality of layers of $(n+n)$ -bit manipulation cells (22, 36, 46, 56 etc), each manipulation cell (22, 36, 46, 56 etc) having a first n-inputs for receiving respective bits corresponding to respective bits $(a_1, a_2$ etc.) in said first array $(a_1...a_8)$, and a second n-inputs for receiving bits corresponding to respective bits $(b_1, b_2$ etc.) in said second array $(b_1...b_8)$, a first n-outputs and a second n-outputs corresponding to said first and second n-inputs respectively,

EP 1 038 215 B9 (W1B1)

wherein each manipulation cell (26, 36, 46, 56 etc), comprises logic means (24, 26, 28 etc), for performing at least one of a bit shift operation and a bit elimination operation,

the (n+n)-bit manipulation cells (22, 36, 46, 56 etc), being interconnected by passing the outputs of each layer apart from the final layer to respective inputs of the next or the next-but-one layer,

wherein in successive layers, the bits set at the output of said bit manipulation cells migrate in successive layers towards a common bit position relative to said layers, with the final layer providing an output indicative of said Hamming relationship.

2. A Hamming value comparator according to Claim 1, wherein said processing means comprises a plurality of substantially common (n+n)-bit manipulation cells (22, 36, 46, 56 etc), each manipulation cell comprising logic means for performing:

- (i) a bit shift operation wherein set bits in each of said first and second n-inputs are shifted in a pre-set direction, where the presence of unset bits allows, to appear in a bit shifted position in the corresponding n-outputs, and
- (ii) a Hamming elimination operation wherein in the event of there being a logic 1 at a given bit position in said first inputs which corresponds to a logic 1 at a related bit position in said second n-inputs, both of said set bits are set to logic 0.

3. A Hamming value comparator according to Claim 2, wherein said Hamming elimination operation is a full elimination operation whereby a logic 1 in one of said first n-inputs is capable of eliminating a logic 1 at any bit position in said second n-inputs.

4. A Hamming value comparator according to Claim 2, wherein said Hamming elimination operation is a partial elimination operation wherein a logic 1 at a given bit position in said first n-inputs may only eliminate a logic 1 at the same bit position in said second n-inputs.

5. A Hamming value comparator according to any of the preceding claims wherein said bit manipulation cells perform a full bit shift wherein, for either of said first and second n-inputs, a logic 1 is always shifted in said preset direction if there is one or more logic 0's adjacent said logic 1 in the direction of said bit shift.

6. A Hamming value comparator according to any of the preceding claims, wherein each bit manipulation cell (22, 36, 46, 56 etc.) includes four inputs (AINL, AINR, BINL, BINR) and four outputs (AOUTL, AOUTR, BOUTL, BOUTR), and the bit manipulation cells in said first layer are arranged such that, for each bit manipulation cell (22,36), two inputs (AINL, AINR) each receive a respective bit derived from one of said binary arrays (a₁...a₈) and the other two inputs (BINL, BINR) each receive a respective bit derived from the other binary array (b₁...b₈).

7. A Hamming value comparator according to Claim 6, wherein each of said respective sets of two inputs (AINL, AINR; BINL, BINR) in the bit manipulation cells (22,36) in said first layer receive adjacent bits derived from the corresponding binary array, (a₁...a₈;b₁...b₈ etc), taken on an even boundary.

8. A Hamming value comparator according to Claim 7, wherein each manipulation cell (32,36) performs an operation on the bits (AINL, AINR; BINL, BINR) derived from said first and second binary arrays ((a₁.....a₈;a₁.....a₈) received at the respective inputs, whereby a set bit is shifted towards a preset bit direction and correspondingly positioned set bits are fully or partially eliminated.

9. A Hamming value comparator according to Claim 5 when dependent on Claim 3, wherein each of said bit manipulation cells has a truth table substantially as set out below, or the inverse thereof:-

A in L	A in R	B in L	B in R	A out L	A out R	B out L	B out R
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	1	0	0	0

EP 1 038 215 B9 (W1B1)

(continued)

5
10
15
20

A in L	A in R	B in L	B in R	A out L	A out R	B out L	B out R
0	1	0	1	0	0	0	0
0	1	1	0	0	0	0	0
0	1	1	1	0	0	1	0
1	0	0	0	1	0	0	0
1	0	0	1	0	0	0	0
1	0	1	0	0	0	0	0
1	0	1	1	0	0	1	0
1	1	0	0	1	1	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	0	0	0	0

wherein AINL, AINR, BINL, BINR, are the four inputs to the cell and AOUTL, AOUTR, BOUTL, BOUTR are the four outputs.

25 **10.** A Hamming value comparator according to Claim 5 when dependent on Claim 4, wherein each of said bit manipulation cells has a truth table substantially as set out below, or the inverse thereof:-

30
35
40
45
50

A in L	A in R	B in L	B in R	A out L	A out R	B out L	B out R
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	0	1	0	1	0
0	1	1	1	0	0	1	0
1	0	0	0	1	0	0	0
1	0	0	1	0	0	1	0
1	0	1	0	0	0	0	0
1	0	1	1	0	0	1	0
1	1	0	0	1	1	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	0	0	0	0

55 **11.** A Hamming value comparator according to any of the preceding Claims, wherein each bit manipulation cell (22,36) has a first plurality of inputs (AINL, AINR) for receiving respective bits (a_1, a_2 etc) derived from one of said arrays ($a_1...a_8$) and a second plurality of inputs (BINL, BINR) for receiving respective bits ($b_1...b_2$) derived from the other of said binary arrays ($b_1...b_8$), and said bit manipulation cell (22,36) has a corresponding plurality of outputs, and

logic means for outputting on a pre-selected one of said outputs a set bit if the Hamming value of the bits ($a_1...a_8$) set on said first plurality of inputs is greater than the Hamming value of the bits ($b_1...b_8$) set on said second plurality of inputs.

- 5 **12.** A Hamming value comparator according to Claim 1 for comparing two one or more dimensional binary arrays ($a_1...a_8; b_1...b_8$) and comprising a plurality of alternate layers of bit manipulation cells (68, 70, 72), ones of said layers comprising a plurality of first bit manipulation cells (68) for shifting set bits in the input arrays towards a pre-set bit position, others of said layers comprising a plurality of said first bit manipulation cells (68), in combination with a Hamming Elimination cell (72) for eliminating corresponding set bits in each of said arrays.
- 10 **13.** A Hamming value comparator according to Claim 1, for comparing two 2-dimensional arrays of binary data (ABCD; abcd) and determining which, if either, has the greater number of logic 1's set, said comparator comprising a plurality of layers (80, 82) of bit manipulation cells (84), each cell (84) including at least two inputs and two outputs and being adapted to shift set bits towards a preset bit position, and being interconnected between layers to effect at least one of bit shifting and bit elimination, thereby to obtain an output indicative of which, if either, array has the greater number of logic 1's set.
- 15 **14.** A Hamming value comparator according to Claim 13, wherein said comparator comprises alternate layers (80, 82), ones of said layers (80) comprising an array of 4+4 bit manipulation cells (84), and others (82) of said layers comprising an array of said 4+4 bit- manipulation cells (84), a plurality of 2+2 bit manipulation cells (88) disposed peripherally, and a plurality of passthroughs (I).
- 20 **15.** A Hamming value comparator according to Claim 14, wherein the bit manipulation cells in each layer are configured to shift the bits towards a common axis or focal point relative to said layer.
- 25 **16.** A Hamming value comparator according to Claim 14 or 15, wherein said 2+2 bit manipulation cells (88) perform the logic operations substantially as set out below, with inputs a,b,c,d, and outputs Ya, Yb, Yc, Yd:-

30
$$Y_a = a \# b \# c \# d$$

$$Y_b = (d \& c) \# (d \& b) \# (d \& a) \# (c \& b) \# (c \& a) \# (b \& a)$$

35
$$Y_c = (d \& c \& b) \# (d \& c \& a) \# (d \& b \& a) \# (c \& b \& a)$$

$$Y_d = a \& b \& c \& d \quad \text{(Equation Set (4))}$$

- 40 wherein "&" represents the AND operator and # represents the OR operator.
- 17.** A Hamming value comparator according to any of Claims 14 to 16, wherein said 4+4 bit manipulation cell (84) performs on the inputs ({ABCD} and {abcd}) to provide the outputs YA, YB, YC, YD, and Ya, Yb, Yc, Yd in accordance with the logic operations set out in Figure 17.
- 45 **18.** A binary sum and threshold device comprising a Hamming value comparator according to any of the preceding claims.
- 50 **19.** A method of determining which, if either, of a first one- or more-dimensional weightless array of weightless bits ($a_1...a_8$) and a second one- or more-dimensional weightless array of weightless bits ($b_1...b_8$), has the greater number of logic 1's therein, said method comprising:-

(i) applying the bits from said arrays to processing means comprising a plurality of layers of (n+n)-bit manipulation cells (22, 36, 46, 56 etc), each manipulation cell (22, 36, 46, 56 etc) having a first n-inputs for receiving respective bits corresponding to respective bits (a_1, a_2 etc.) in said first array ($a_1...a_8$), and a second n-inputs for receiving bits corresponding to respective bits (b_1, b_2 etc.) in said second array ($b_1...b_8$), a first n-outputs and a second n-outputs corresponding to said first and second n-inputs respectively,

55

wherein each manipulation cell (26, 36, 46, 56 etc), comprises logic means (24, 26, 28 etc), for performing at least one of a bit shift operation and a bit elimination operation,
 the (n+n)-bit manipulation cells (22, 36, 46, 56 etc), being interconnected by passing the outputs of each layer apart from the final layer to respective inputs of the next or the next-but-one layer,
 wherein in successive layers, the bits set at the output of said bit manipulation cells migrate in successive layers towards a common bit position relative to said layers, with the final layer providing an output indicative of said Hamming relationship.

Patentansprüche

1. Hammingwert-Komparator zur Lieferung eines Ausgangs, der die Hammingwert-Beziehung eines ersten ein- oder mehrdimensionalen ungewogenen Feldes ungewogener Bits ($a_1 \dots a_8$) und eines zweiten ein- oder mehrdimensionalen ungewogenen Feldes ungewogener Bits ($b_1 \dots b_8$) anzeigt,

wobei der Hammingwert eines gegebenen ungewogenen Feldes die Zahl der darin befindlichen logischen 1's ist und die Hammingwert-Beziehung von zwei ungewogenen Feldern anzeigt, welches den größeren Hammingwert hat oder ob die Hammingwerte hiervon die gleichen sind;

wobei der Hammingwert-Komparator die folgenden Merkmale aufweist:

Mittel, um die Bits von den Feldern einem Prozessor zuzuführen, der mehrere Lagen von (n+n)-Bit-Manipulationszellen (22, 36, 46, 56 usw.) aufweist, wobei jede Manipulationszelle (22, 36, 46, 56 usw.) erste n-Eingänge zum Empfang der jeweiligen Bits aufweist, die den jeweiligen Bits ($a_1 a_2$ usw.) in dem ersten Feld ($a_1 \dots a_8$) entsprechen und zweite n-Eingänge vorgesehen sind, um Bits zu empfangen, die den jeweiligen Bits (b_1, b_2 usw.) in dem zweiten Feld ($b_1 \dots b_8$) entsprechen und erste n-Ausgänge und zweite n-Ausgänge jeweils den ersten bzw. zweiten n-Eingängen entsprechen,

wobei jede Manipulationszelle (22, 36, 46, 56 usw.) Logikmittel (24, 26, 28 usw.) aufweist, um wenigstens eine Bit-Verschiebeoperation oder eine Bit-Eliminierungsoperation durchzuführen,

wobei die (n+n)-Bit-Manipulationszellen (22, 36, 46, 56 usw.) miteinander verbunden sind, indem die Ausgänge jeder Schicht außer der letzten Schicht nach den jeweiligen Eingängen der nächsten oder übernächsten Lage verlaufen,

und wobei in aufeinanderfolgenden Lagen die Bit-Gruppen am Ausgang der Manipulationszellen in aufeinanderfolgende Lagen nach einer gemeinsamen Bit-Position relativ zu den Lagen wandern und die Endlage einen Ausgang liefert, der die Hamming-Beziehung anzeigt.

2. Hammingwert-Komparator nach Anspruch 1, bei welchem der Prozessor mehrere im Wesentlichen gemeinsame (n+n)-Bit-Manipulationszellen (22, 36, 46, 56 usw.) aufweist und jede Manipulationszelle Logikmittel umfasst, um die folgenden Schritte durchzuführen:

(i) eine Bit-Verschiebeoperation, wobei Bit-Gruppen in jedem der ersten und zweiten n-Eingänge in einer vorbestimmten Richtung verschoben werden und das Vorhandensein genullter Bits ein Erscheinen in einer Bit-verschobenen Position in dem entsprechenden n-Ausgang ermöglicht, und

(ii) eine Hamming-Eliminierungsoperation, wobei dann, wenn eine logisch 1 an einer gegebenen Bit-Position in den ersten Eingängen steht, die einer logisch 1 an einer entsprechenden Bit-Position in den zweiten Eingängen entspricht, beide Bit-Gruppen auf logisch 0 gesetzt werden.

3. Hammingwert-Komparator nach Anspruch 2, bei welchem die Hamming-Eliminierungsoperation eine volle Eliminierungsoperation ist, wobei eine logisch 1 in einem der ersten Eingänge in der Lage ist, eine logisch 1 an irgendeiner Bit-Position in den zweiten n-Eingängen zu eliminieren.

4. Hammingwert-Komparator nach Anspruch 2, bei welchem die Hamming-Eliminierungsoperation eine partielle Eliminierungsoperation ist, wobei eine logisch 1 an einer gegebenen Bit-Position in den ersten n-Eingängen nur eine logisch 1 an der gleichen Bit-Position in den zweiten n-Eingängen eliminieren kann.

5. Hammingwert-Komparator nach einem der vorhergehenden Ansprüche, bei welchem die Bit-Manipulationszellen eine vollständige Bit-Verschiebung durchführen, wobei sowohl für die ersten als auch die zweiten n-Eingänge eine logisch 1 immer in der vorbestimmten Richtung verschoben wird, wenn eine oder mehrere logische 0 benachbart

EP 1 038 215 B9 (W1B1)

zur logisch 1 in Richtung der Bit-Verschiebung vorhanden sind.

- 5 **6.** Hammingwert-Komparator nach einem der vorhergehenden Ansprüche, bei welchem jede Bit-Manipulationszelle (22, 36, 46, 56 usw.) vier Eingänge (AINL, AINR, BINL, BINR) und vier Ausgänge (AOUTL, AOUTR, BOUTL, BOUTR) aufweist und die Bit-Manipulationszellen in der ersten Lage so angeordnet sind, dass für jede Bit-Manipulationszelle (22, 36) zwei Eingänge (AINL, AINR) jeweils ein entsprechendes Bit empfangen, das von einer der Binärfelder ($a_1 \dots a_8$) abgeleitet wird und die anderen beiden Eingänge (BINL, BINR) jeweils ein entsprechendes Bit empfangen, das von dem anderen Binärfeld ($b_1 \dots b_8$) abgeleitet wurde.
- 10 **7.** Hammingwert-Komparator nach Anspruch 6, bei welchem jede der entsprechenden Gruppen von zwei Eingängen (AINL, AINR; BINL, BINR) nach den Bit-Manipulationszellen (22, 36) in der ersten Lage benachbarte Bits empfangen, die von den entsprechenden Binärfeldern ($a_1 \dots a_8$; $b_1 \dots b_8$ usw.) abgeleitet sind, abgenommen von einer geraden Begrenzung.
- 15 **8.** Hammingwert-Komparator nach Anspruch 7, bei welchem jede Manipulationszelle (22, 36) eine Operation auf die Bits (AINL, AINR; BINL, BINR) durchführt, die von den ersten und zweiten Binärfeldern ($a_1 \dots a_8$; $b_1 \dots b_8$) abgeleitet wurden und an den entsprechenden Eingängen empfangen wurden, wodurch eine Bit-Gruppe nach einer vorbestimmten Bit-Richtung verschoben und entsprechende positionierte Bit-Gruppen vollständig oder teilweise eliminiert werden.
- 20 **9.** Hammingwert-Komparator nach Anspruch 5 bei Abhängigkeit von Anspruch 3, bei welchem jede der Bit-Manipulationszellen eine Wahrheitstabelle aufweist, die im Wesentlichen wie folgt oder invers hierzu aufgebaut ist.

25

A in L	A in R	B in L	B in R	A out L	A out R	B out L	B out R
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	0	0	0	0	0
0	1	1	1	0	0	1	0
1	0	0	0	1	0	0	0
1	0	0	1	0	0	0	0
1	0	1	0	0	0	0	0
1	0	1	1	0	0	1	0
1	1	0	0	1	1	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	0	0	0	0

50

wobei AINL, AINR, BINL, BINR die vier Eingänge nach der Zelle sind und AOUTL, AOUTR, BOUTL, BOUTR die vier Ausgänge sind.

55

- 10.** Hammingwert-Komparator nach Anspruch 5 bei Abhängigkeit von Anspruch 4, bei welchem jede Bit-Manipulationszelle eine Wahrheitstabelle enthält, die im Wesentlichen wie im Folgenden beschrieben oder invers hierzu aufgebaut ist.

EP 1 038 215 B9 (W1B1)

A in L	A in R	B in L	B in R	A out L	A out R	B out L	B out R
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	0	1	0	1	0
0	1	1	1	0	0	1	0
1	0	0	0	1	0	0	0
1	0	0	1	0	0	1	0
1	0	1	0	0	0	0	0
1	0	1	1	0	0	1	0
1	1	0	0	1	1	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	0	0	0	0

11. Hammingwert-Komparator nach einem der vorhergehenden Ansprüche, bei welchem jede Bit-Manipulationszelle (22, 36) eine erste Mehrzahl von Eingängen (AINL, AINR) zum Empfang der jeweiligen Bits (a_1, a_2 usw.) aufweist, die von einem der Felder ($a_1 \dots a_8$) abgeleitet wurden und eine zweite Mehrzahl von Eingängen (BINL, BINR) zum Empfang der jeweiligen Bits (b_1, b_2), die von den anderen Binärfeldern ($b_1 \dots b_8$) abgeleitet wurden und die Bit-Manipulationszelle (22, 36) eine entsprechende Mehrzahl von Ausgängen aufweist und Logikmittel vorgesehen sind, um auf einem vorgewählten Ausgang eine Bit-Gruppe auszugeben, wenn der Hammingwert der Bits ($a_1 \dots a_8$) auf der ersten Mehrzahl von Eingängen größer ist als der Hammingwert der Bits ($b_1 \dots b_8$) auf der zweiten Mehrzahl von Eingängen.
12. Hammingwert-Komparator nach Anspruch 1 zum Vergleich von zwei einoder mehrdimensionalen Binärfeldern ($a_1 \dots a_8; b_1 \dots b_8$), der mehrere abwechselnde Lagen von Bit-Manipulationszellen (68, 70, 72) aufweist und eine der Lagen eine Mehrzahl von ersten Bit-Manipulationszellen (68) zur Verschiebung der Bit-Gruppen in den Eingangsfeldern nach einer vorbestimmten Bit-Position aufweist und andere der Lagen mehrere Bit-Manipulationszellen (68) in Verbindung mit einer Hamming-Eliminierungszelle (72) aufweisen, um entsprechende Bit-Gruppen in jedem der Felder zu eliminieren.
13. Hammingwert-Komparator nach Anspruch 1 zum Vergleich von zwei zweidimensionalen Feldern binärer Daten (ABCD; abcd) und zur Bestimmung, welche, wenn überhaupt, die größere Zahl von logischen 1's enthält, wobei der Komparator eine Mehrzahl von Lagen (80, 82) von Bit-Manipulationszellen (84) aufweist und jede Zelle (84) wenigstens zwei Eingänge und zwei Ausgänge aufweist und so ausgebildet ist, dass die Bit-Gruppen nach einer vorgewählten Bit-Position verschoben werden und eine Verbindung zwischen den Lagen vorhanden ist, um wenigstens eine Bit-Verschiebung oder eine Bit-Eliminierung durchzuführen, wodurch ein Ausgang erhalten wird, der anzeigt, welches Feld die größere Zahl von logischen 1's enthält, wenn überhaupt welche vorhanden sind.
14. Hammingwert-Komparator nach Anspruch 13, bei welchem der Komparator abwechselnde Lagen (80, 82) aufweist und eine der Lagen (80) ein Feld von 4+4-Bit-Manipulationszellen (84) umfasst und die andere Lage (82) ein Feld von 4+4-Bit-Manipulationszellen (84) aufweist und mehrere 2+2-Bit-Manipulationszellen (88) in Umfangsrichtung angeordnet sind und mehrere Durchgänge (I) vorgesehen sind.
15. Hammingwert-Komparator nach Anspruch 14, bei welchem die Bit-Manipulationszellen in jeder Lage so ausge-

EP 1 038 215 B9 (W1B1)

bildet sind, dass die Bits nach einer gemeinsamen Achse oder einem Brennpunkt relativ zu den Lagen verschoben werden.

- 5 16. Hammingwert-Komparator nach einem der Ansprüche 14 oder 15, bei welchem die 2+2-Bit-Manipulationszellen (88) die Logikoperationen wie nachstehend beschrieben unter Benutzung der Eingänge a, b, c, d und der Ausgänge Ya, Yb, Yc, Yd durchführen:

$$Y_a = a \# b \# c \# d$$

$$Y_b = (d \& c) \# (d \& b) \# (d \& a) \# (c \& b) \# (c \& a) \# (b \& a)$$

$$Y_c = (d \& c \& b) \# (d \& c \& a) \# (d \& b \& a) \# (c \& b \& a)$$

$$Y_d = a \& b \& c \& d \quad \text{(Gleichungsgruppe 4)}$$

wobei "&" den UND-Operator und "#" den ODER-Operator repräsentiert.

- 20 17. Hammingwert-Komparator nach einem der Ansprüche 14 bis 16, bei welchem die 4+4-Bit-Manipulationszelle (84) über die Eingänge ({ABCD} und {abcd}) Ausgänge YA, YB, YC, YD und Ya, Yb, Yc, Yd gemäß den Logikoperationen liefert, wie sie in Fig. 17 dargestellt sind.

- 25 18. Binär-Summierungs- und Schwellwertvorrichtung mit einem Hammingwert-Komparator nach einem der vorhergehenden Ansprüche.

- 30 19. Verfahren zur Bestimmung, ob, wenn überhaupt, ein erstes ein- oder mehrdimensionales ungewogenes Feld ungewogener Bits ($a_1 \dots a_8$) oder ein zweites ein- oder mehrdimensionales ungewogenes Feld ungewogener Bits ($b_1 \dots b_8$) die größere Zahl logischer 1's aufweist, wobei das Verfahren die folgenden Schritte umfasst:

(i) es werden die Bits aus den Feldern einem Prozessor zugeführt, der eine Mehrzahl von Lagen von (n+n)-Bit-Manipulationszellen (22, 36, 46, 56 usw.) aufweist und jede Manipulationszelle (22, 36, 46, 56 usw.) erste n-Eingänge zum Empfang der jeweiligen Bits aufweist, die den Bits (a_1, a_2 usw.) in dem ersten Feld ($a_1 \dots a_8$) entsprechen und zweite n-Eingänge zum Empfang von Bits entsprechend den jeweiligen Bits (b_1, b_2 usw.) in dem zweiten Feld ($b_1 \dots b_8$) und erste n-Ausgänge und zweite n-Ausgänge entsprechend den ersten bzw. den zweiten Eingängen vorgesehen sind,

wobei jede Manipulationszelle (22, 36, 46, 56 usw.) Logikmittel (24, 26, 28 usw.) aufweist, um wenigstens eine Bit-Verschiebeoperation oder eine Bit-Eliminierungsoperation durchzuführen,

wobei die n+n-Bit-Manipulationszellen (22, 36, 46, 56 usw.) miteinander verbunden sind, indem die Ausgänge jeder Lage außer der Endlage nach den jeweiligen Eingängen der nächsten oder übernächsten Lage hindurchlaufen,

wobei in aufeinanderfolgenden Lagen die Bit-Gruppen am Ausgang der Bit-Manipulationszellen in aufeinanderfolgende Lagen nach einer gemeinsamen Bit-Position relativ zu den Lagen wandern und die Endlage einen Ausgang liefert, der die Hamming-Beziehung anzeigt.

Revendications

- 50 1. Comparateur de valeur de Hamming pour fournir une sortie indiquant la relation de valeur de Hamming entre un premier tableau sans pondération uni- ou multidimensionnel de bits sans pondération ($a_1 \dots a_8$) et un deuxième tableau sans pondération uni- ou multidimensionnel de bits sans pondération ($b_1 \dots b_8$),

dans lequel la valeur de Hamming d'un tableau sans pondération donné est le nombre de 1 logiques qui y sont contenus et la relation de Hamming entre deux tableaux sans pondération indique quelle valeur de Hamming est la plus grande ou bien l'égalité des valeurs de Hamming en question,

ledit comparateur de valeur de Hamming comprenant :

des moyens pour appliquer les bits desdits tableaux au moyen de traitement comprenant une pluralité de couches de cellules de manipulation de (n+n) bits pratiquement communes (22, 36, 46, 56, etc.), chaque

EP 1 038 215 B9 (W1B1)

cellule de manipulation (22, 36, 46, 56, etc.) ayant un premier ensemble de n entrées pour recevoir les bits respectifs correspondant à ceux ($a_1, a_2, \text{etc.}$) dudit premier tableau ($a_1 \dots a_8$), et un deuxième ensemble de n entrées pour recevoir les bits respectifs correspondant à ceux ($b_1, b_2, \text{etc.}$) dudit deuxième tableau ($b_1 \dots b_8$), un premier et un deuxième ensemble de n sorties correspondant respectivement aux dits premier et deuxième ensembles de n entrées,

dans lequel chaque cellule de manipulation (26, 36, 46, 56, etc.) comprend des moyens logiques (24, 26, 28, etc.) pour effectuer au moins une parmi une opération de décalage de bit et une opération d'élimination de bit, les cellules de manipulation de $(n+n)$ bits (22, 36, 46, 56, etc.) étant interconnectées en passant les sorties de chaque couche à l'exception de la couche finale vers les entrées respectives de la couche suivante ou de celle d'après,

dans lequel, dans les couches successives, les bits positionnés à la sortie desdites cellules de manipulation de bits migrent en couches successives vers une position commune des bits par rapport aux dites couches, la couche finale fournissant une sortie qui indique ladite relation de Hamming.

2. Comparateur de valeur de Hamming selon la revendication 1, dans lequel lesdits moyens de traitement comprennent une pluralité de cellules de manipulation de $(n+n)$ bits pratiquement communes (22, 36, 46, 56, etc.), chaque cellule de manipulation comprenant des moyens logiques pour effectuer :

(i) une opération de décalage de bit dans laquelle les bits positionnés dans chacun desdits premier et deuxième ensembles de n entrées sont décalés dans une direction prédéterminée, où la présence des bits non positionnés leur permet d'apparaître dans une position décalée des bits dans les ensembles de n sorties correspondants, et

(ii) une opération d'élimination de Hamming, dans laquelle, dans le cas où un 1 logique se trouverait à une position de bit donnée dans lesdites premières entrées correspondant à un 1 logique dans une position de bit équivalente dans ledit deuxième ensemble de n entrées, lesdits deux bits positionnés sont repositionnés à un 0 logique.

3. Comparateur de valeur de Hamming selon la revendication 2, dans lequel ladite opération d'élimination de Hamming est une opération d'élimination complète moyennant quoi un 1 logique dans un dudit premier ensemble de n entrées est capable d'éliminer un 1 logique à n'importe quelle position de bit dans ledit deuxième ensemble de n entrées.

4. Comparateur de valeur de Hamming selon la revendication 2, dans lequel ladite opération d'élimination de Hamming est une opération d'élimination partielle dans laquelle un 1 logique à une position de bit donnée dans ledit premier ensemble de n entrées ne peut éliminer qu'un 1 logique à la même position de bit dans ledit deuxième ensemble de n entrées.

5. Comparateur de valeur de Hamming selon l'une quelconque des revendications précédentes, dans lequel lesdites cellules de manipulation de bits effectuent un décalage de bits complet dans lequel, pour chacun desdits premier et deuxième ensembles de n entrées, un 1 logique est toujours décalé dans ladite direction prédéterminée s'il existe un ou plusieurs 0 logiques adjacents au dit 1 logique dans la direction dudit décalage de bits.

6. Comparateur de valeur de Hamming selon l'une quelconque des revendications précédentes, dans lequel chaque cellule de manipulation de bits (22, 36, 46, 56, etc.) comprend quatre entrées (AINL, AINR, BINL, BINR) et quatre sorties (AOUTL, AOUTR, BOUTL, BOUTR) et les cellules de manipulation de bits dans ladite première couche sont disposées afin que pour chaque cellule de manipulation de bits (22, 36) chacune des deux entrées (AINL, AINR) reçoive un bit respectif dérivé d'un desdits tableaux binaires ($a_1 \dots a_8$) et que chacune des deux autres entrées (BINL, BINR) reçoive un bit respectif dérivé de l'autre tableau binaire ($b_1 \dots b_8$).

7. Comparateur de valeur de Hamming selon la revendication 6, dans lequel chacun desdits ensembles respectifs de deux entrées (AINL, AINR ; BINL, BINR) dans les cellules de manipulation des bits (22, 36) de ladite première couche reçoit les bits adjacents dérivés du tableau binaire correspondant, ($a_1 \dots a_8$; $b_1 \dots b_8$, etc.), pris sur une frontière paire.

8. Comparateur de valeur de Hamming selon la revendication 7, dans lequel chaque cellule de manipulation (32, 36) effectue une opération sur les bits (AINL, AINR; BINL, BINR) dérivée desdits premier et deuxième tableaux binaires ($a_1 \dots a_8$; $a_1 \dots a_8$) reçus lors des entrées respectives, moyennant quoi un bit positionné est décalé vers une direction

EP 1 038 215 B9 (W1B1)

de bit prédéfinie et les bits positionnés correspondants sont éliminés totalement ou partiellement.

- 5 9. Comparateur de valeur de Hamming selon la revendication 5 lorsqu'elle dépend de la revendication 3, dans lequel chacune desdites cellules de manipulation de bits a une table de vérité disposée pratiquement comme indiqué ci-dessous ou bien inversée :

10

15

20

25

30

AinL	AinR	BinL	BinR	AoutL	AoutR	BoutL	BoutR
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	0	0	0	0	0
0	1	1	1	0	0	1	0
1	0	0	0	1	0	0	0
1	0	0	1	0	0	0	0
1	0	1	0	0	0	0	0
1	0	1	1	0	0	1	0
1	1	0	0	1	1	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	0	0	0	0

où AINL, AINR, BINL, BINR, sont les quatre entrées vers la cellule et AOUTL, AOUTR, BOUTL, BOUTR sont les quatre sorties.

- 35 10. Comparateur de valeur de Hamming selon la revendication 5 lorsqu'elle dépend de la revendication 4, dans lequel chacune desdites cellules de manipulation de bits a une table de vérité disposée pratiquement comme indiqué ci-dessous ou bien inversée :

40

45

50

55

AinL	AinR	BinL	BinR	AoutL	AoutR	BoutL	BoutR
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	1	0	0	0
0	1	0	1	0	0	0	0
0	1	1	0	1	0	1	0
0	1	1	1	0	0	1	0
1	0	0	0	1	0	0	0
1	0	0	1	0	0	1	0
1	0	1	0	0	0	0	0
1	0	1	1	0	0	1	0

EP 1 038 215 B9 (W1B1)

(suite)

AinL	AinR	BinL	BinR	AoutL	AoutR	BoutL	BoutR
1	1	0	0	1	1	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	0	0	0	0

5
10
15
11. Comparateur de valeur de Hamming selon l'une quelconque des revendications précédentes, dans lequel chaque cellule de manipulation de bits (22, 36) a une première pluralité d'entrées (AINL, AINR) pour recevoir les bits respectifs (a_1, a_2, \dots) dérivés à partir d'un desdits tableaux ($a_1 \dots a_8$) et une deuxième pluralité d'entrées (BINL, BINR) pour recevoir les bits respectifs ($b_1 \dots b_2$) dérivés à partir de l'autre desdits tableaux binaires ($b_1 \dots b_8$), et que ladite cellule de manipulation de bits (22, 36) a une pluralité correspondante de sorties, et des moyens logiques pour la sortie sur une desdites sorties ayant été présélectionnée d'un bit positionné si la valeur de Hamming des bits ($a_1 \dots a_8$) positionnés sur ladite première pluralité d'entrées est plus grande que la valeur de Hamming des bits ($b_1 \dots b_8$) positionnés sur ladite deuxième pluralité d'entrées.

20
25
12. Comparateur de valeur de Hamming selon la revendication 1, pour comparer deux tableaux binaires uni- ou multidimensionnels ($a_1 \dots a_8; b_1 \dots b_8$) et comprenant une pluralité de couches alternées de cellules de manipulation de bits (68, 70, 72), les unes desdites couches comprenant une pluralité de cellules de manipulation du premier bit (68) pour décaler les bits positionnés dans les tableaux d'entrée vers une position de bits présélectionnée, les autres desdites couches comprenant une pluralité de cellules de manipulation du premier bit (68), combinée à une cellule d'élimination de Hamming (72), pour éliminer les bits positionnés correspondants dans chacun desdits tableaux.

30
35
13. Comparateur de valeur de Hamming selon la revendication 1, pour comparer deux tableaux bidimensionnels de données binaires (ABCD ; abcd) et déterminer lequel, si il y a, a le plus grand nombre de 1 logiques positionnés, ledit comparateur comprenant une pluralité de couches (80, 82) de cellules de manipulation de bits (84), chaque cellule (84) comprenant au moins deux entrées et deux sorties et étant adaptée pour décaler les bits positionnés vers une position de bits prédéterminée et étant interconnectée entre les couches pour effectuer au moins une des opérations de décalage de bit et d'élimination de bit, afin d'obtenir de la sorte une sortie qui indique lequel, si c'est le cas, des tableaux a le plus grand nombre de 1 logiques positionnés.

40
14. Comparateur de valeur de Hamming selon la revendication 13, dans lequel ledit comparateur comprend des couches alternées (80, 82), les unes desdites couches (80) comprenant un tableau de cellules de manipulation de 4+4 bits (84) et les autres (82) desdites couches comprenant un tableau desdites cellules de manipulation de 4+4 bits (84), une pluralité de cellules de manipulation de 2+2 bits (88) disposées à la périphérie et une pluralité de voies de traversée (I).

45
15. Comparateur de valeur de Hamming selon la revendication 14, dans lequel les cellules de manipulation des bits de chaque couche sont configurées pour décaler les bits vers un axe commun ou un point focal relatif à ladite couche.

50
16. Comparateur de valeur de Hamming selon la revendication 14 ou 15, dans lequel lesdites cellules de manipulation de bits 2+2 (88) effectuent les opérations logiques pratiquement comme présenté ci-dessous, les entrées étant a, b, c, d et les sorties Ya, Yb, Yc, Yd :

$$Y_a = a \# b \# c \# d$$

$$Y_b = (d \& c) \# (d \& b) \# (d \& a) \# (c \& b) \# (c \& a)$$

$$\# (b \& a)$$

$$Y_c = (d \& c \& b) \# (d \& c \& a) \# (d \& b \& a) \# (c \&$$

EP 1 038 215 B9 (W1B1)

b & a)

$$Yd = a \& b \& c \& d$$

(ensemble d'équations (4))

5

où & représente l'opérateur AND et # représente l'opérateur OR.

10

17. Comparateur de valeur de Hamming selon l'une quelconque des revendications 14 à 16, dans lequel ladite cellule de manipulation de 4+4 bits (84) opère sur les entrées ({ABCD} et {abcd}) pour fournir les sorties YA, YB, YC, YD et Ya, Yb, Yc, Yd selon les opérations logiques présentées sur la figure 17.

18. Dispositif à somme binaire et seuil comprenant un comparateur de valeur de Hamming selon l'une quelconque des revendications précédentes.

15

19. Procédé de détermination lequel, si il y a, qui, d'un premier tableau sans pondération uni- ou multidimensionnel de bits sans pondération ($a_1...a_8$) et d'un deuxième tableau sans pondération uni- ou multidimensionnel de bits sans pondération ($b_1...b_8$), contient le plus grand nombre de 1 logiques, ledit procédé comprenant :

20

(i) l'application des bits desdits tableaux aux moyens de traitement comprenant une pluralité de cellules de manipulation de (n+n) bits (22, 36, 46, 56, etc.), chaque cellule de manipulation (22, 36, 46, 56, etc.) ayant un premier ensemble de n entrées pour recevoir les bits respectifs correspondant aux bits respectifs (a_1, a_2, \dots) dudit premier tableau ($a_1...a_8$), et un deuxième ensemble de n entrées pour recevoir les bits correspondant aux bits respectifs (b_1, b_2, \dots) dans ledit deuxième tableau ($b_1...b_8$), un premier ensemble de n sorties et un deuxième ensemble de n sorties correspondant respectivement au premier et au deuxième ensemble de n entrées,

25

dans lequel chaque cellule de manipulation (26, 36, 46, 56, etc.), comprend des moyens logiques (24, 26, 28, etc.) pour exécuter au moins une parmi une opération de décalage de bit et une opération d'élimination de bit, les cellules de manipulation de (n+n) bits (22, 36, 46, 56, etc.), étant interconnectées par le passage des sorties de chaque couche, à l'exception de la couche finale, aux entrées respectives de la couche suivante ou de celle d'après,

30

dans lequel, à travers les couches successives, les bits positionnés à la sortie desdites cellules de manipulation de bits migrent en couches successives vers une position de bits commune relative aux dites couches, la couche finale fournissant une sortie indiquant ladite relation de Hamming.

35

40

45

50

55

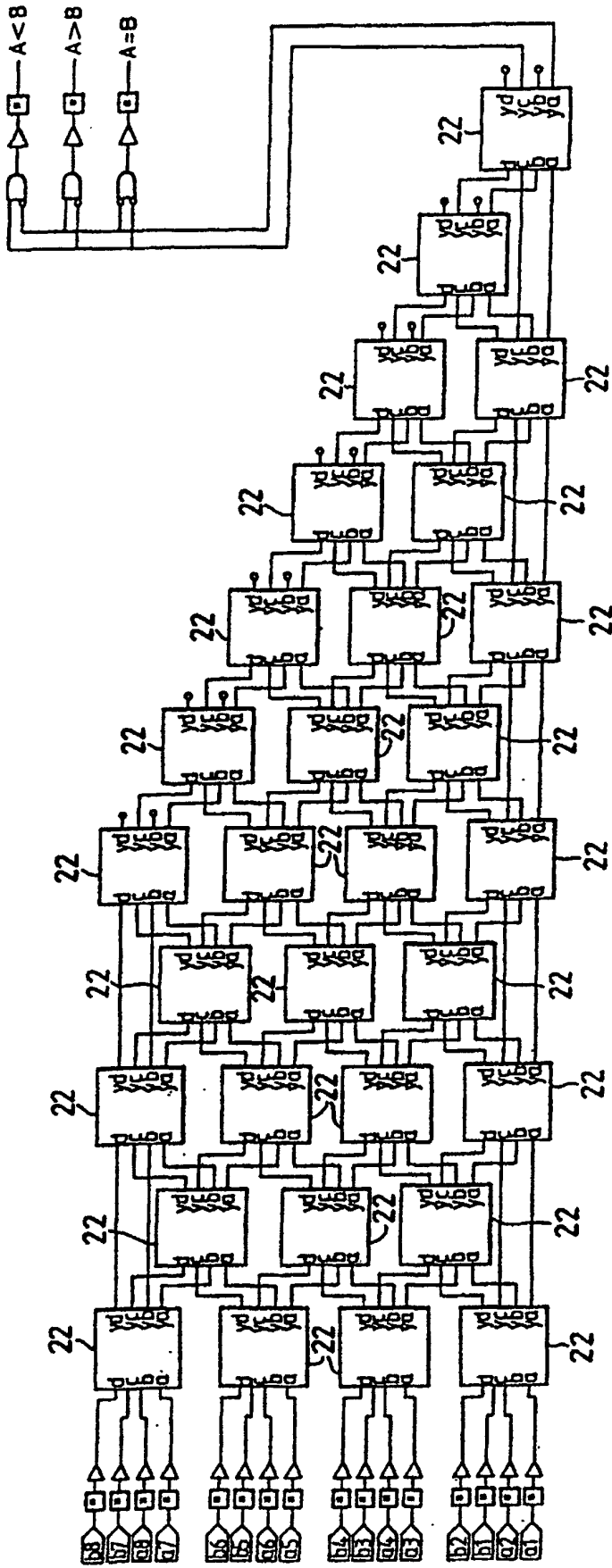


Fig. 1

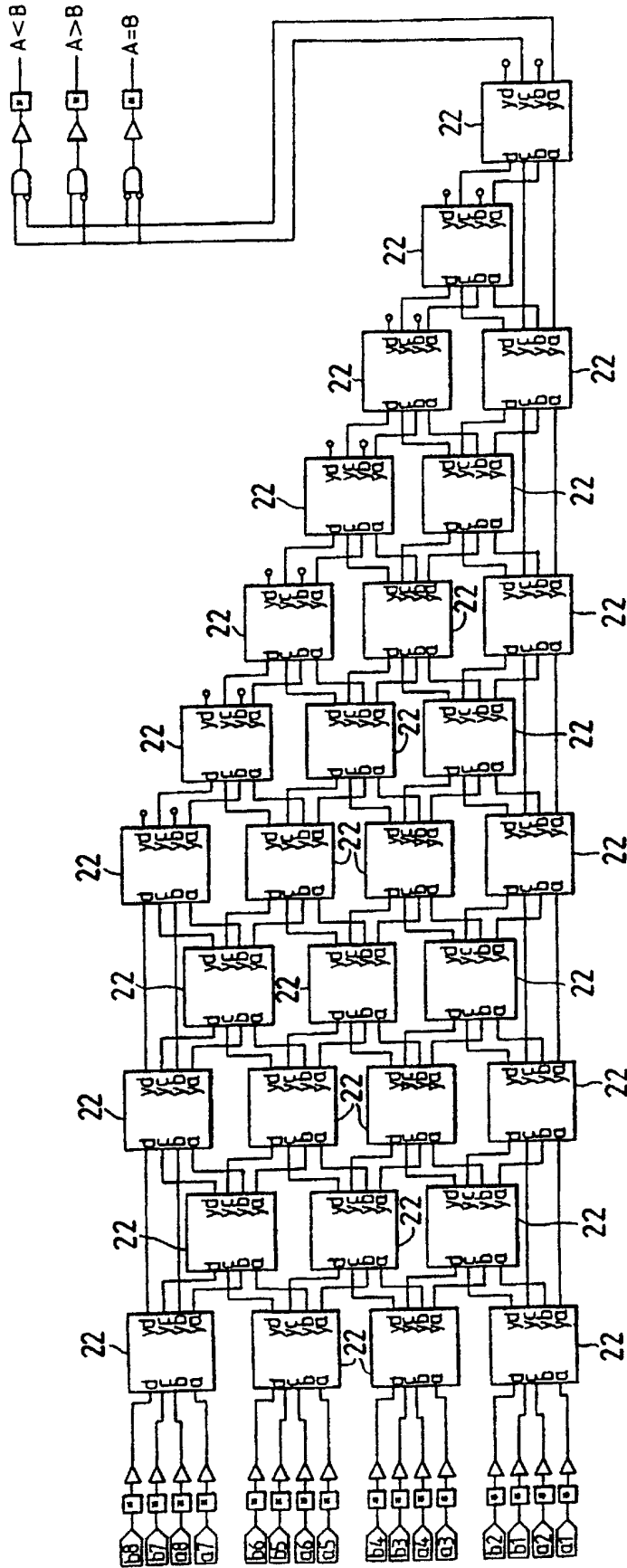


Fig. 2 (a)

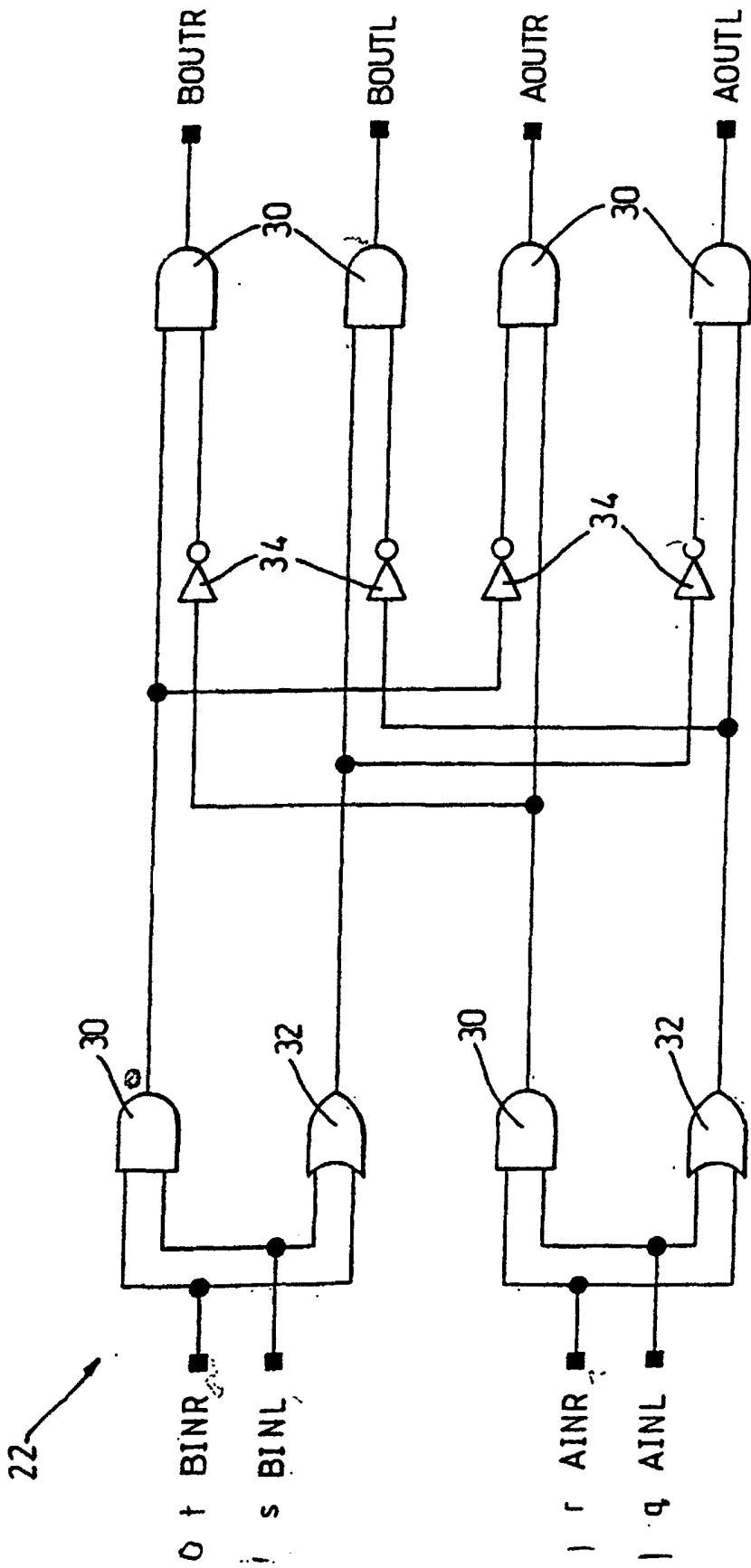


Fig. 2(b)

A = 10100110
 B = 00100000

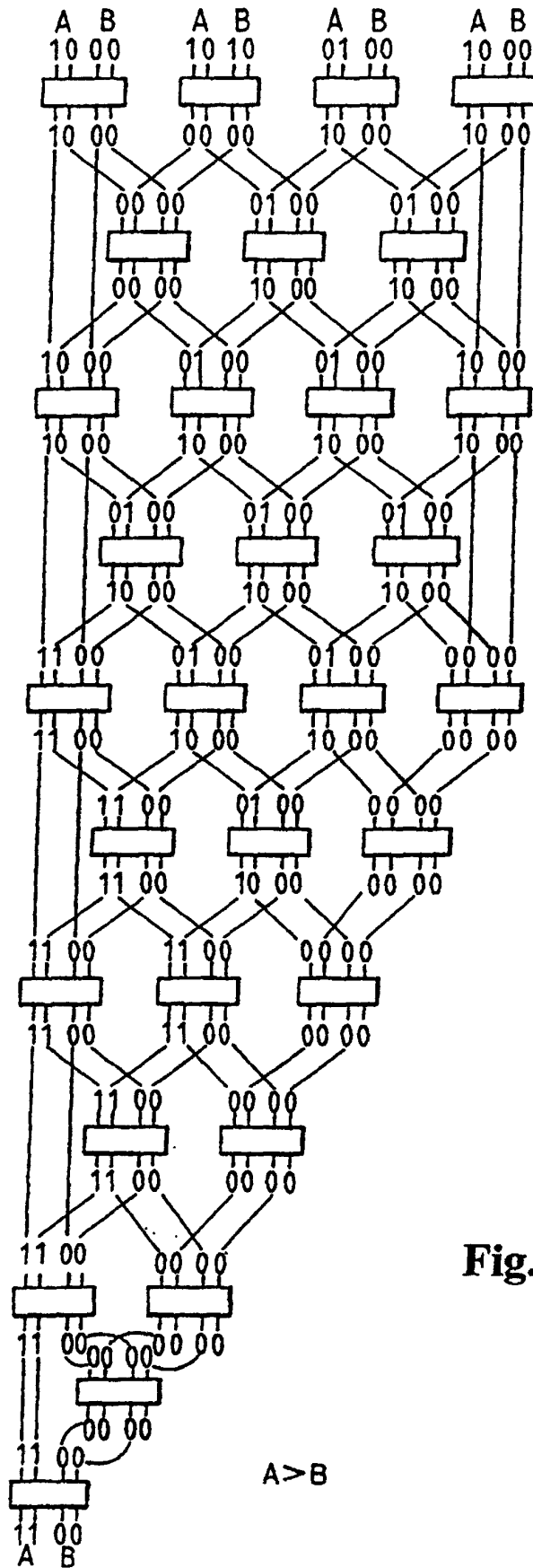


Fig. 3

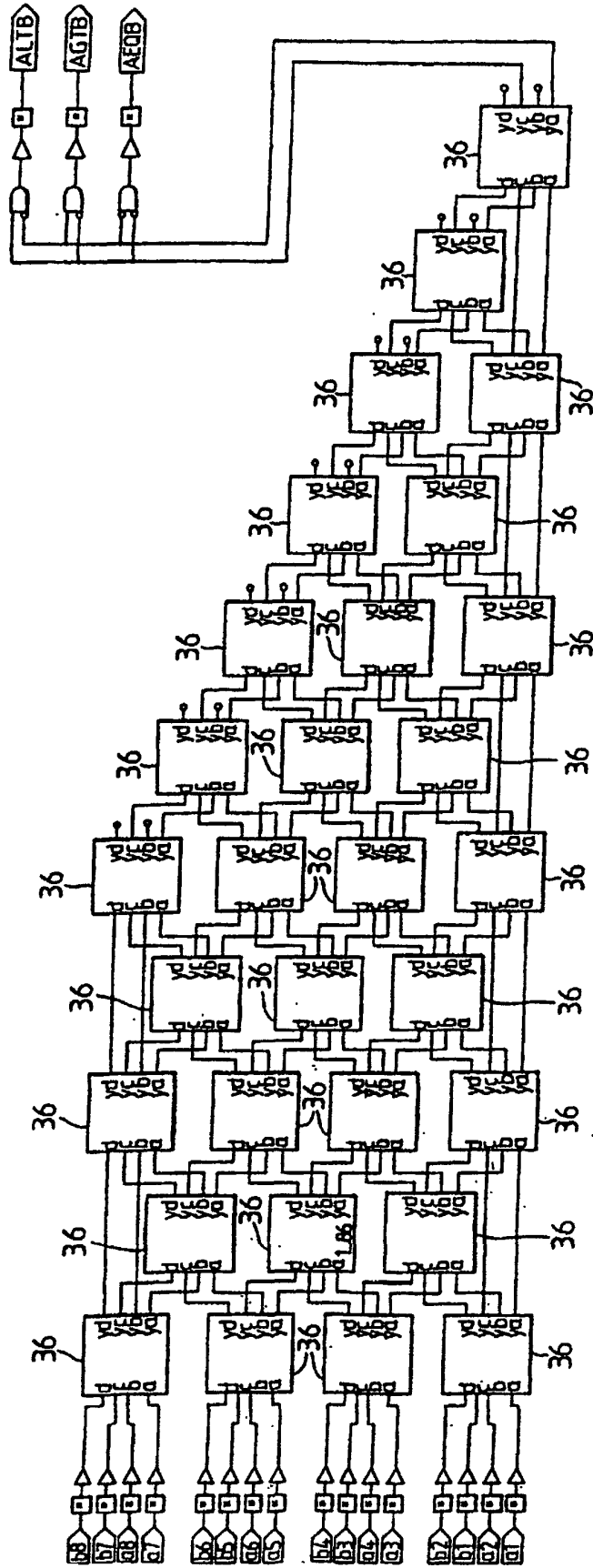


Fig. 4

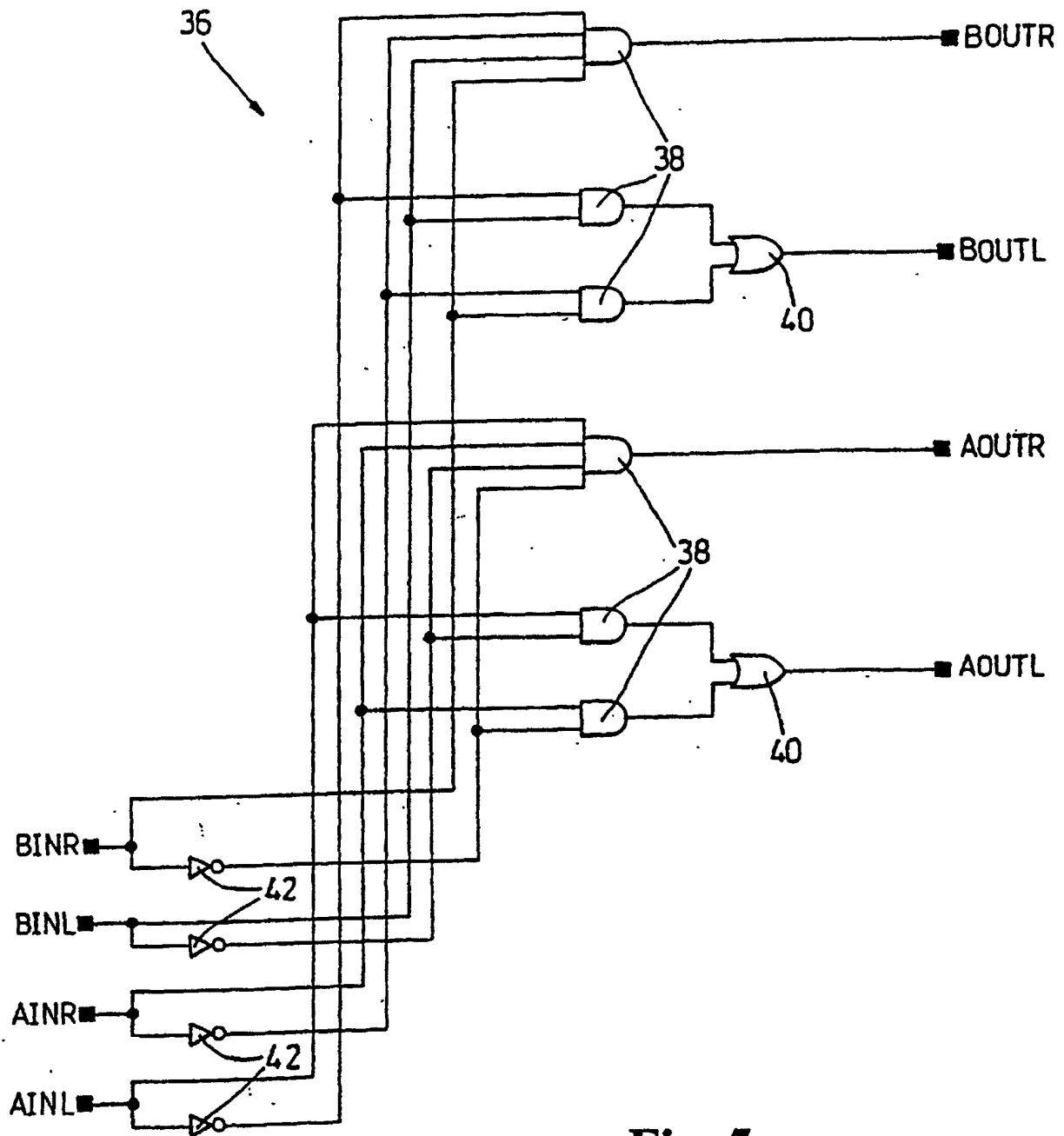
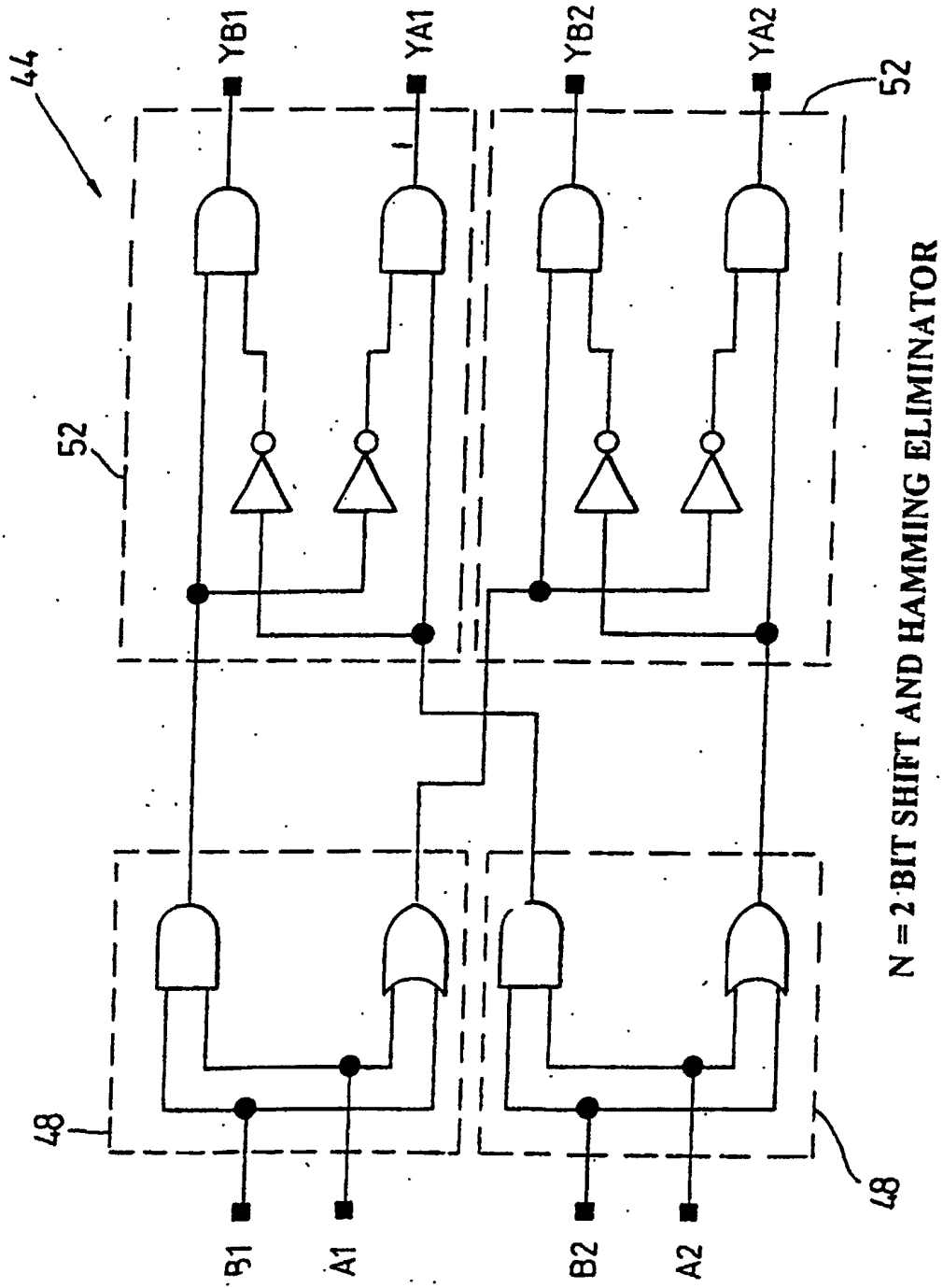


Fig. 5



SYMBOL HCOMP2.1

Fig. 7

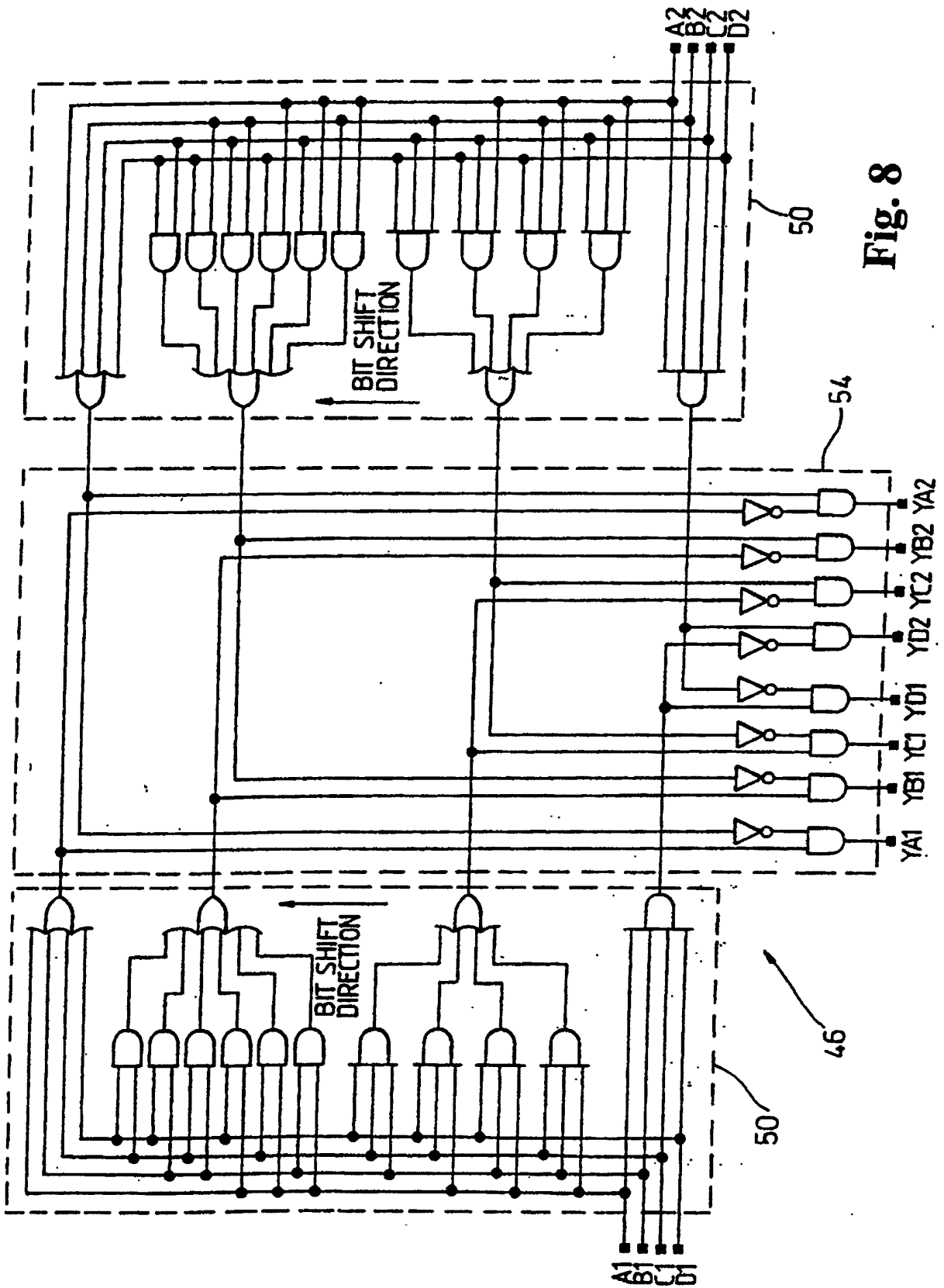


Fig. 8

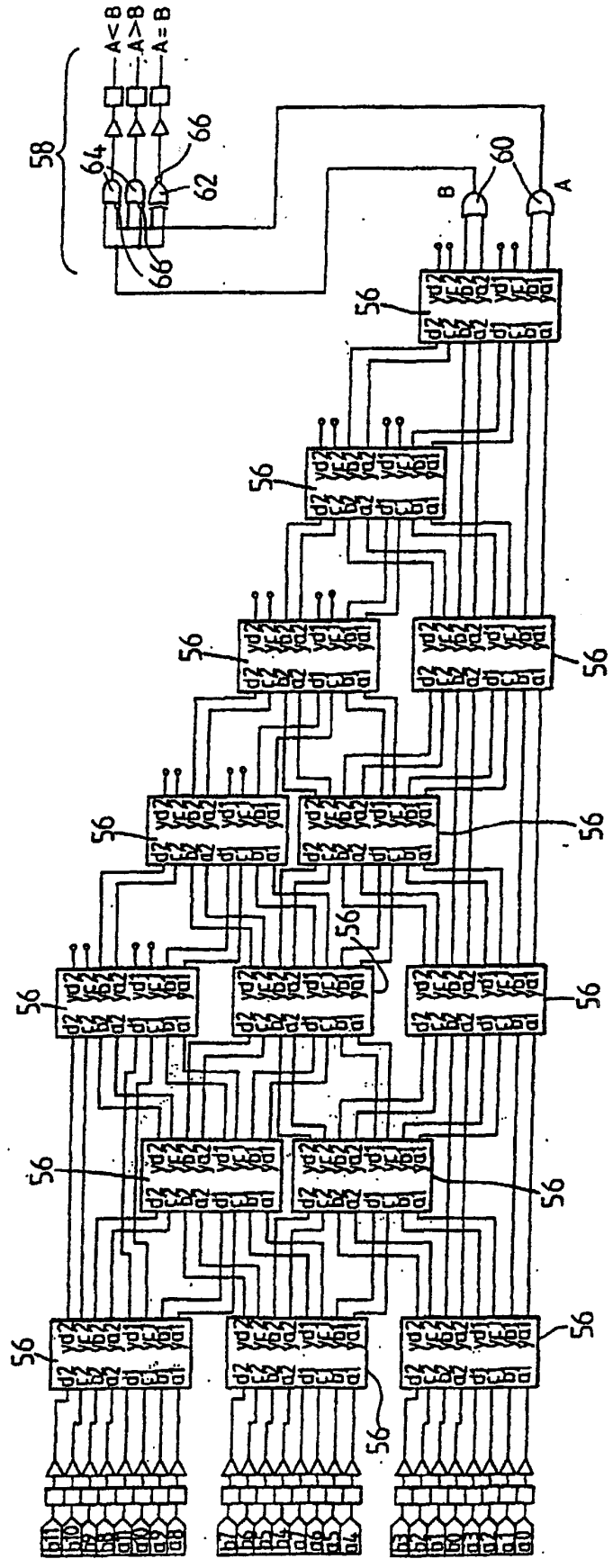
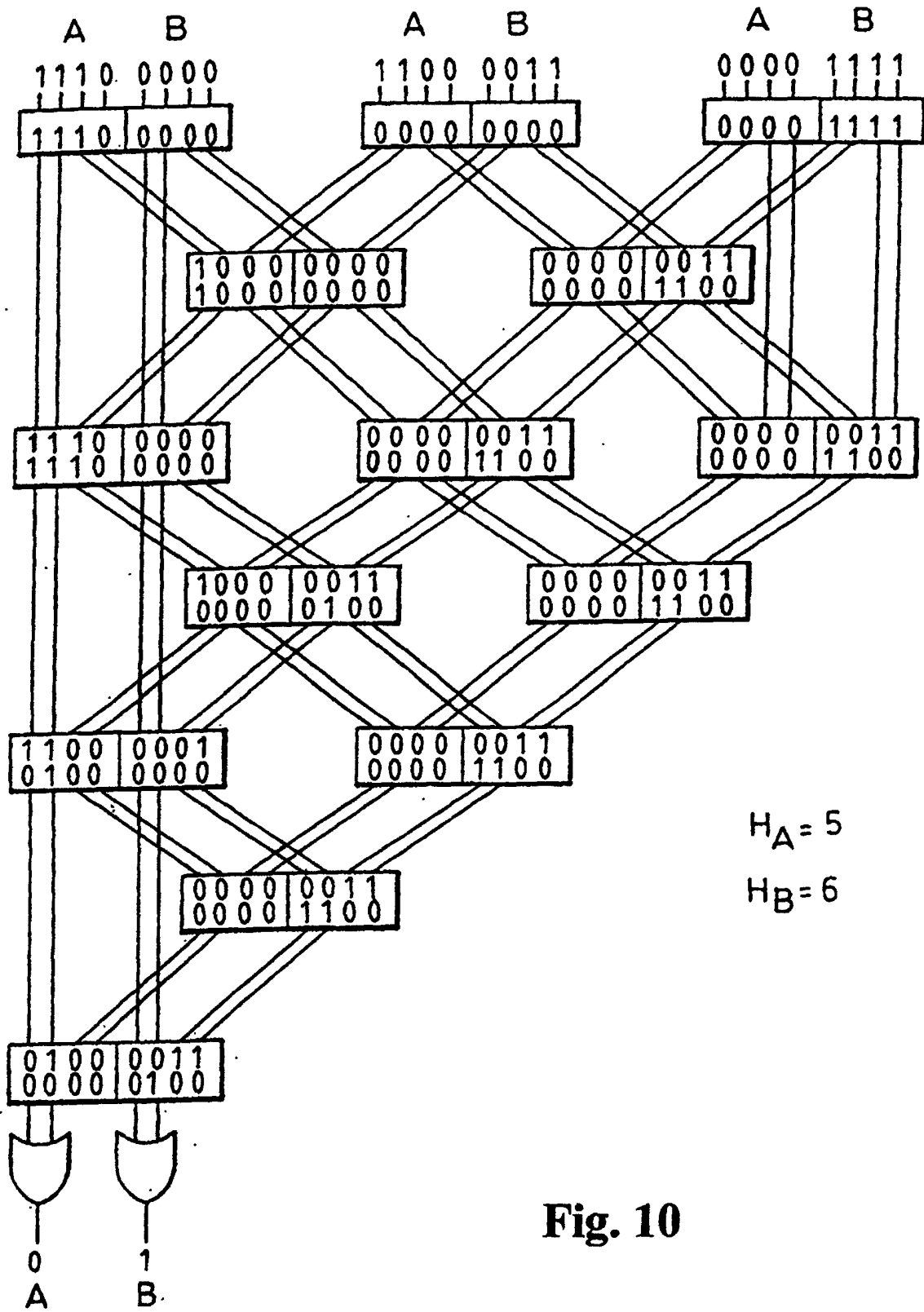


Fig. 9



$H_A = 5$
 $H_B = 6$

Fig. 10

$H_B > H_A$

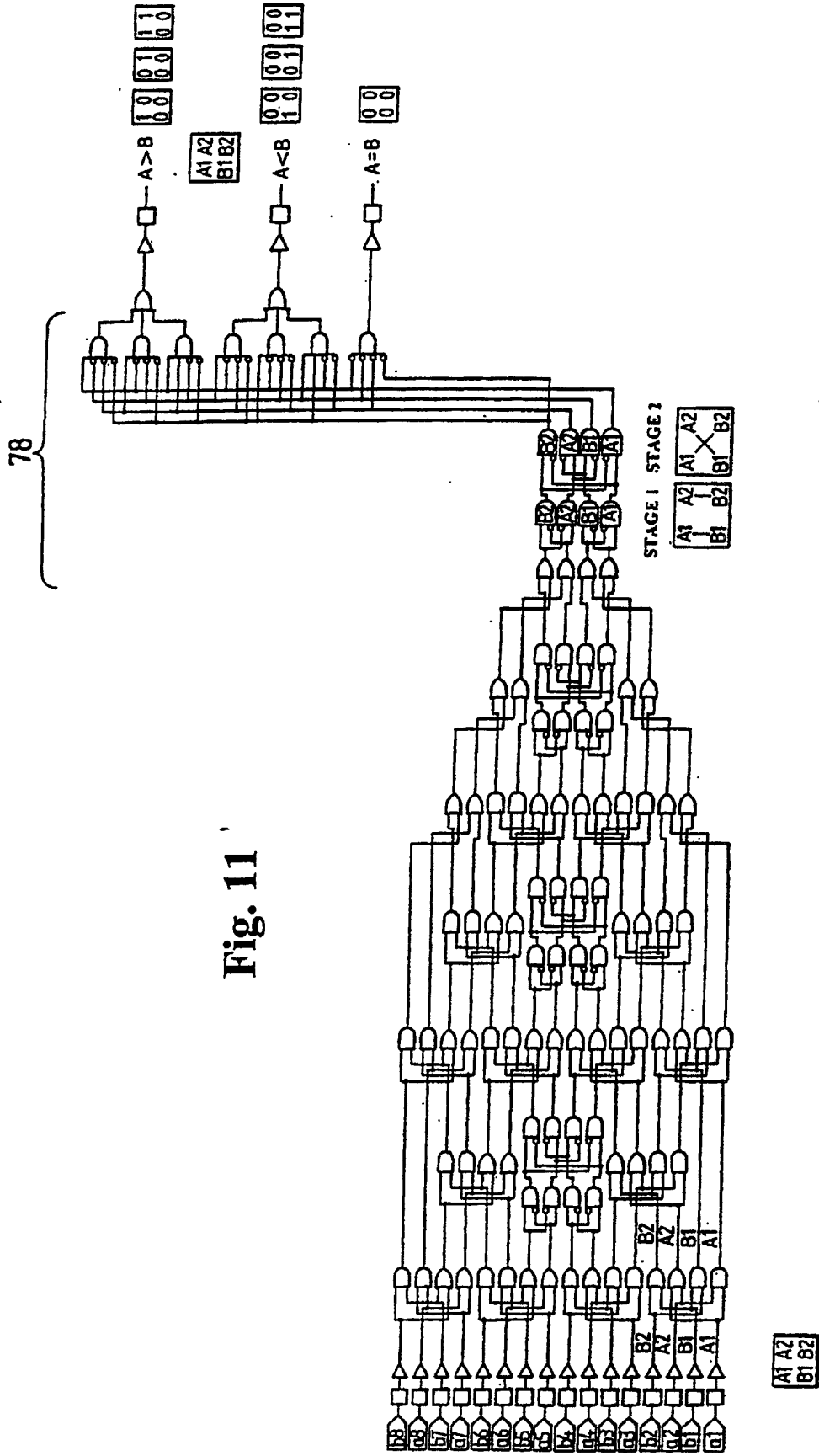


Fig. 11

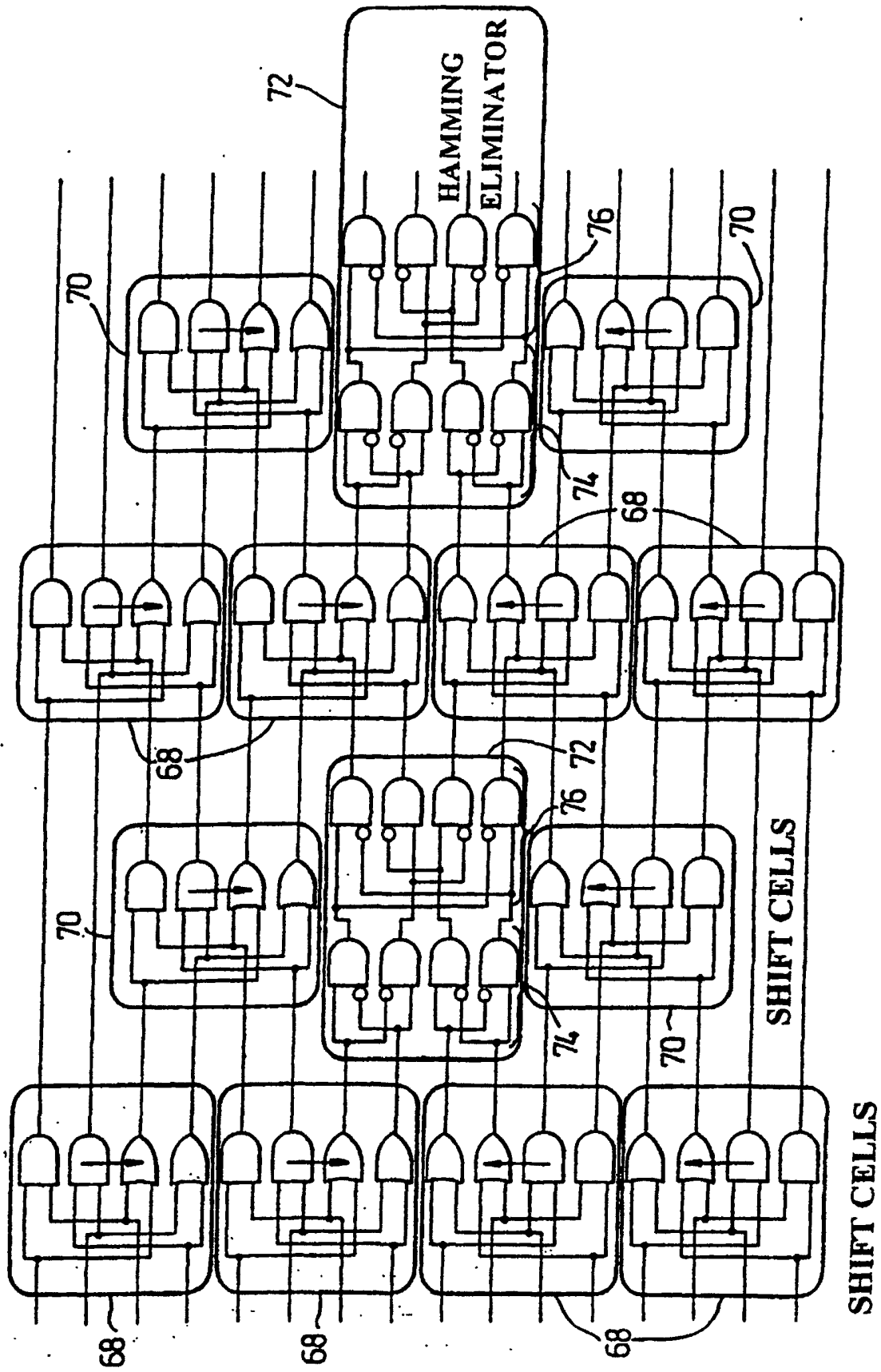


Fig. 12

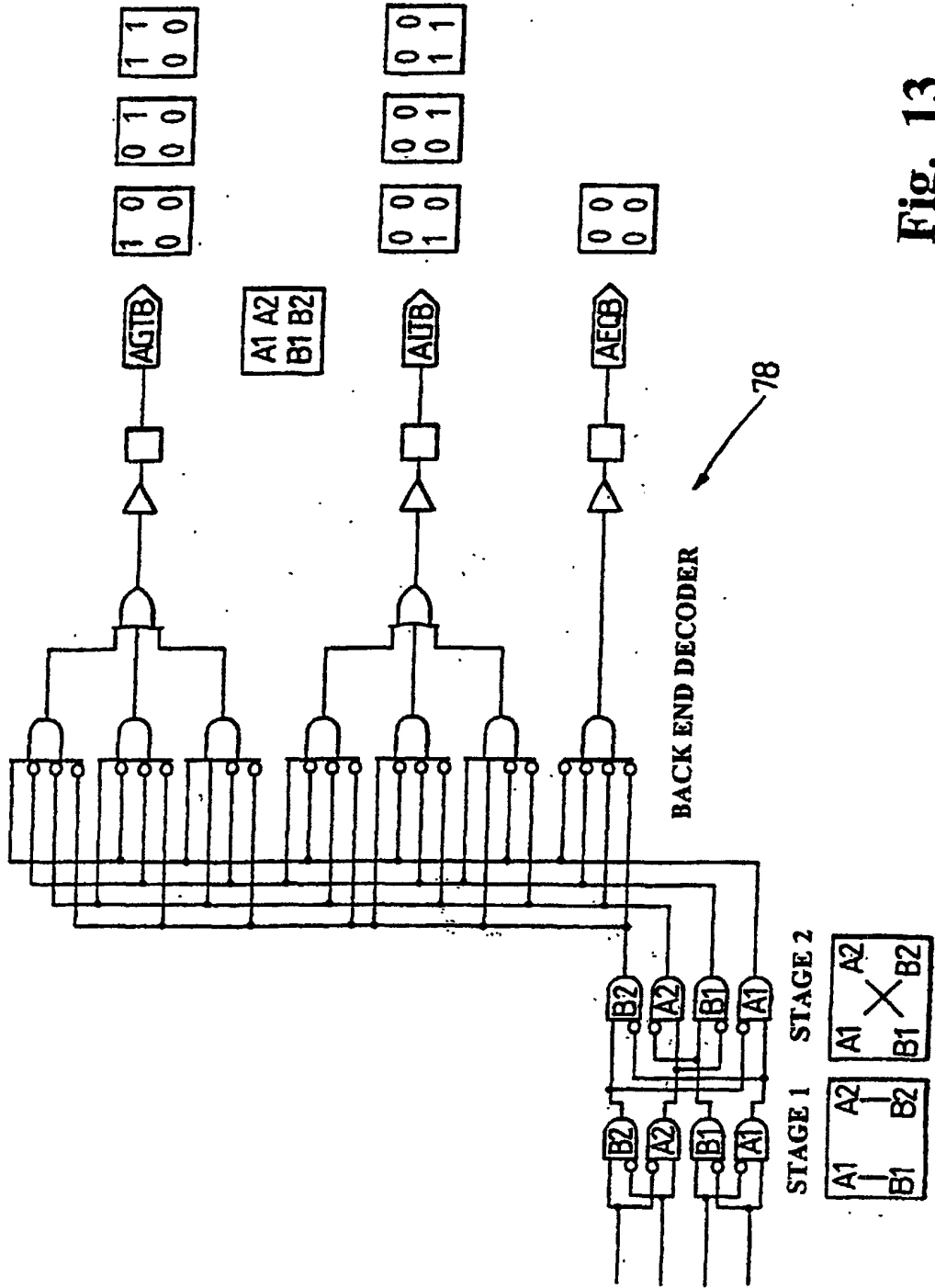


Fig. 13

VERTICAL AND CROSS HAMMING ELIMINATOR CELL
STAGE 1 IS A VERTICAL A1/B1 A2/B2 ELIMINATOR
STAGE 2 IS A CROSS A1/B2 A2/B1 ELIMINATOR
STAGES 1 AND 2 CAN BE PLACED IN EITHER ORDER

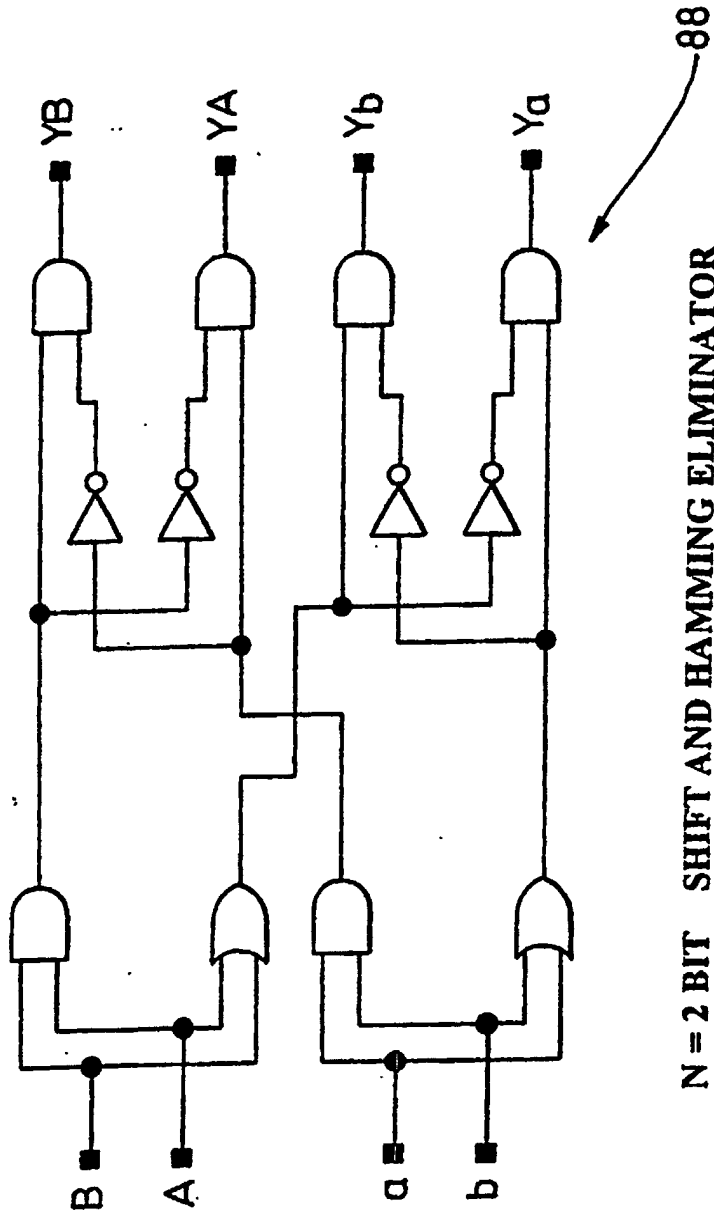
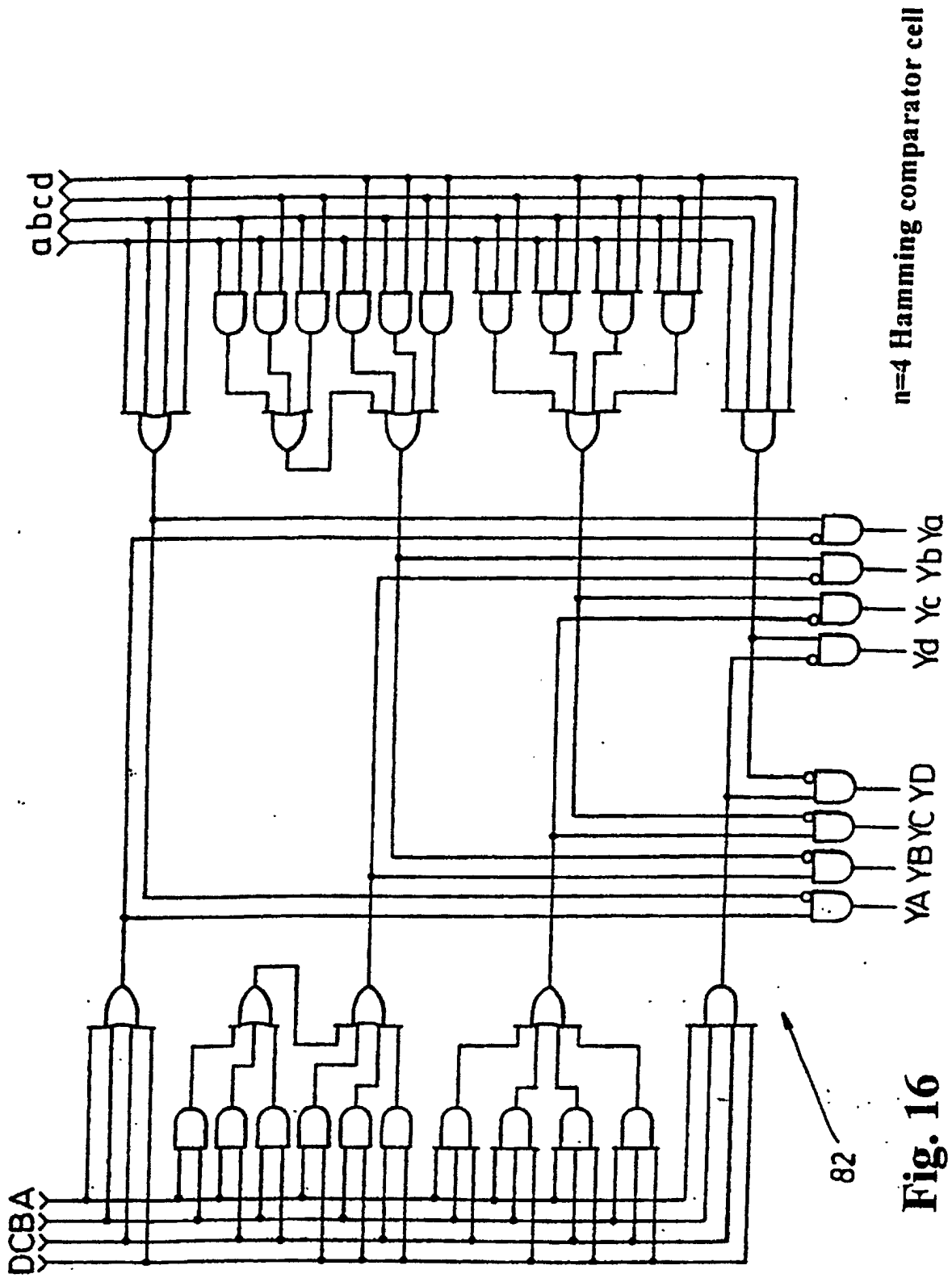


Fig. 15



$n=4$ Hamming comparator cell

Fig. 16

2 Input Arrays

0		0	1		0	0		0		
	1	1			0	0		1	0	
	0	0			0	1			0	1
0			1	1			1	0		0
1			1	0			1	0		1
	0	1			1	0			0	0
	1	1			1	0			0	1
1		0	0		0	1			0	
0		0	1			1	0			1
	1	0			0	1			0	1
	1	0			1	1			0	1
0		0	0		1	0			0	

{ Outer array = 14
Inner array = 18

1st layer output

0		0	0		1	0		0	
	0	0			0	0		1	0
	1	0			0	0		1	0
0		0	1		0	0		0	0
0		0	0		0	0		0	0
	0	0			0	0		0	0
	0	0			1	0		0	0
0		0	0		0	1		0	
0		0	0		0	0		0	0
	1	1			0	0		0	0
	0	0			0	0		1	0
0		0	0		0	0		0	0
	0	0			0	0		1	0
0		0	0		0	0		0	0

{ Outer array = 3 (11 bits eliminated)
Inner array = 7 (11 bits eliminated)

0		0	0		0	0		0	
	0	0			0	0		0	0
	1	0			0	1		0	0
0		1	0		0	0		0	0
0		0	0		0	0		0	0
	0	0			0	0		0	0
	1	1			1	0		0	0
0		0	0		1	0		0	0
0		0	0		0	0		0	0
	0	0			0	0		0	0
	0	0			0	1		0	0
0		0	0		0	0		0	0

{ Outer array = 2
Inner array = 6

0		0	0		0	0		0	
	0	0			1	0		0	0
	0	0			0	0		0	0
0		0	0		0	0		0	0
0		0	0		0	0		0	0
	1	1			0	0		0	0
	0	0			0	0		0	0
0		0	0		0	0		0	0
0		0	0		0	0		0	0
	0	0			1	0		0	0
	0	0			0	0		0	0
0		0	0		0	0		0	0

{ Outer array = 0
Inner array = 4

0		0	0		0	0		0	
	0	1			0	0		0	0
	1	1			0	0		0	0
0		0	0		0	0		0	0
0		0	0		0	0		0	0
	0	0			0	0		0	0
	0	1			0	0		0	0
0		0	0		0	0		0	0
0		0	0		0	0		0	0
	0	0			0	0		0	0
	0	0			0	0		0	0
0		0	0		0	0		0	0

Inner array = 4

0		0	0		0	0		0	
	1	1			0	0		0	0
	1	0			0	0		0	0
0		0	0		0	0		0	0
0		0	0		0	0		0	0
	1	0			0	0		0	0
	0	0			0	0		0	0
0		0	0		0	0		0	0
0		0	0		0	0		0	0
	0	0			0	0		0	0
	0	0			0	0		0	0
0		0	0		0	0		0	0

Final Stable State
Inner > Outer

Fig. 17

0		1	1		0	0		1	
	0	1			1	0		0	1
	1	0			0	0		1	0
0		1	1		1	1			0
	1		0	1		1	0		0
	0	1			0	0		1	0
	1	0			0	0		0	1
0		1	1		1	1			1
1		1	1		1	1			1
	0	1			0	0		0	0
	0	0			1	1		0	0
1		0	0		0	1			1

INPUT { Outer Array = 24
Inner Array = 12

0			0	0		1	0		0
	0	0			0	0		0	0
	0	0			0	0		0	0
0		0	1		0	0			0
0		0	1		1	0			0
	0	0			0	0		0	0
	0	0			0	0		0	0
0		0	1		1	0			0
0		1	0		0	1			1
	0	0			0	0		0	0
	0	0			0	0		0	0
1		0	0		0	1			1

{ Outer Array = 12 Layer 1
Inner Array = 0

0		0	0		1	0		0	
	0	0			0	0		0	0
	0	0			0	0		0	0
0		1	0		1	0		0	0
0		1	0		0	0			0
	0	0			0	0		0	0
	0	0			0	0		0	0
0		1	1		1	0			1
0		0	0		1	0			0
	0	0			0	0		0	0
	0	0			0	0		0	0
1		0	0		1	0			1

{ Outer Array = 12 Layer 2
Inner Array = 0

1		0	1		0	0		0	
	0	0			0	0		0	0
	0	0			0	0		0	0
0		0	1		0	0			0
1		1	1		0	1			0
	0	0			0	0		0	0
	0	0			0	0		0	0
0		0	1		0	0			0
1		0	1		1	1			0
	0	0			0	0		0	0
	0	0			0	0		0	0
0		0	0		0	0		0	0

{ Outer Array = 12 Layer 3

1		1	0		0	0		0	
	0	0			0	0		0	0
	0	0			0	0		0	0
1		1	1		1	0			0
0		1	0		0	0			0
	0	0			0	0		0	0
	0	0			0	0		0	0
1		1	1		1	0			0
0		0	0		1	0			0
	0	0			0	0		0	0
	0	0			0	0		0	0
0		0	0		0	0		0	0

Layer 4

1		1	1		0	0		0	
	0	0			0	0		0	0
	0	0			0	0		0	0
1		1	1		0	0			0
1		1	1		0	0			0
	0	0			0	0		0	0
	0	0			0	0		0	0
1		0	1		0	0			0
0		0	1		0	0			0
	0	0			0	0		0	0
	0	0			0	0		0	0
0		0	0		0	0		0	0

Layer 5

Fig. 18(a)

Final part of example 2

1		1	1		0	0		0	
	0	0			0	0		0	0
	0	0			0	0		0	0
1		1	1		0	0		0	
1		1	1		0	0		0	
	0	0			0	0		0	0
	0	0			0	0		0	0
1		1	1		0	0		0	
0		0	0		0	0		0	
	0	0			0	0		0	0
	0	0			0	0		0	0
0		0	0		0	0		0	

Layer 6

Outer > Inner
Set Set

Fig. 18(b)

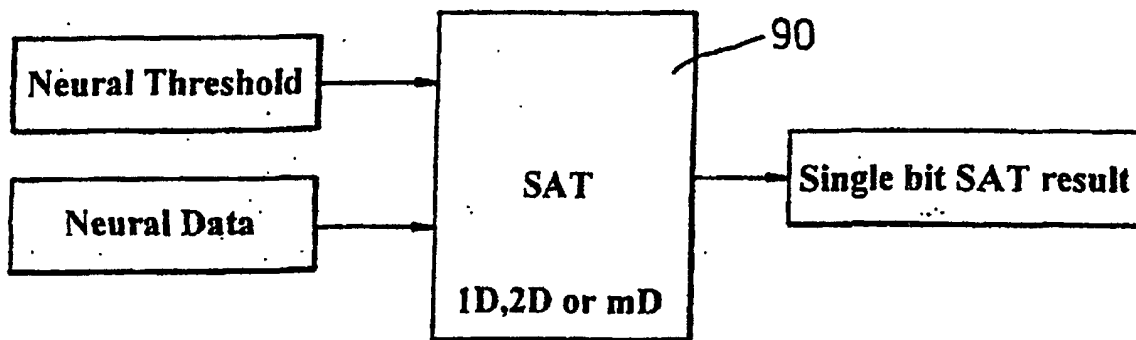


Fig. 19