Feb. 18, 1969 K. E. BATCHER 3,428,946

MEANS FOR MERGING DATA

Filed Sept. 19, 1967 Sheet 1 of 4

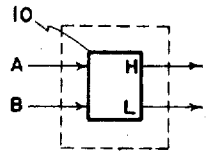

FIG.-1

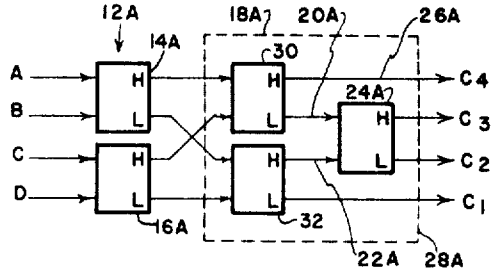

FIG.-2

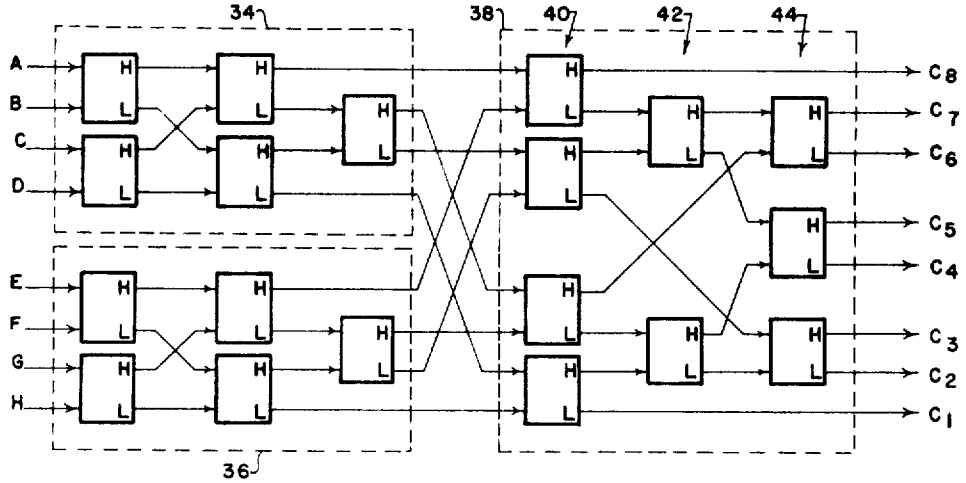

FIG.-3

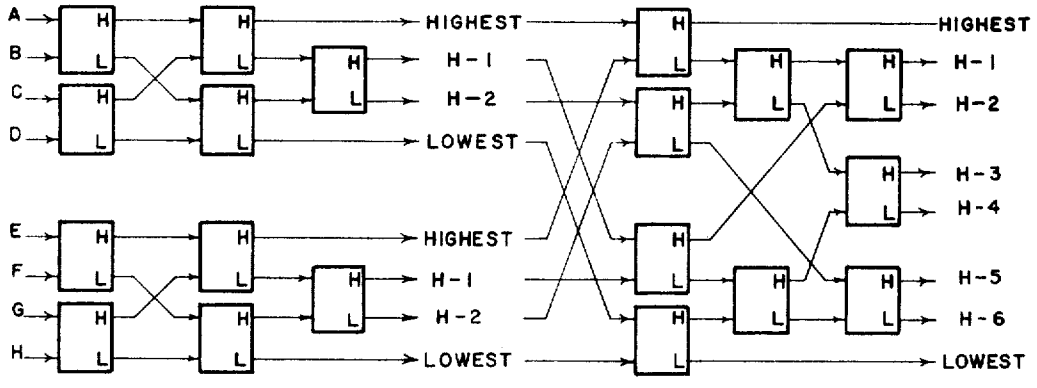

FIG.-4

INVENTOR
KENNETH E. BATCHER

BY:
Oldham & Oldham
ATTORNEYS

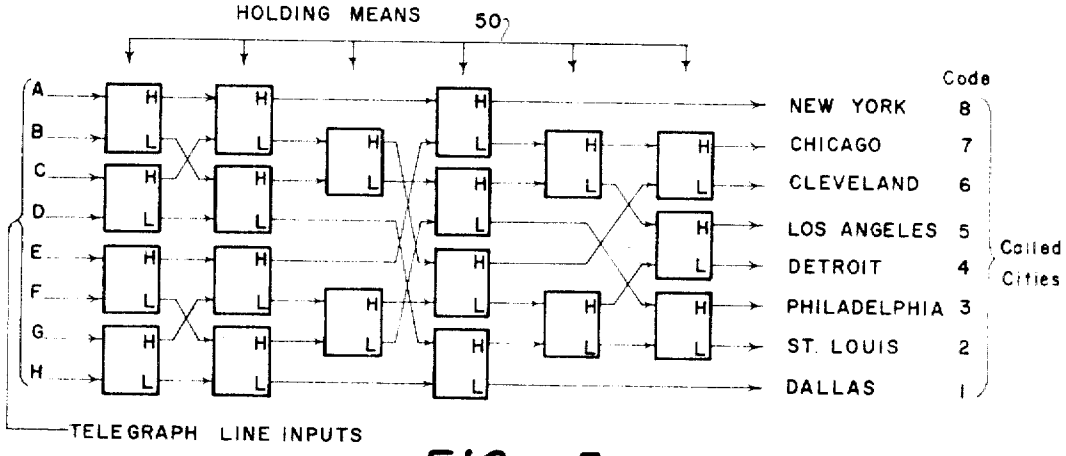

FIG. – 5

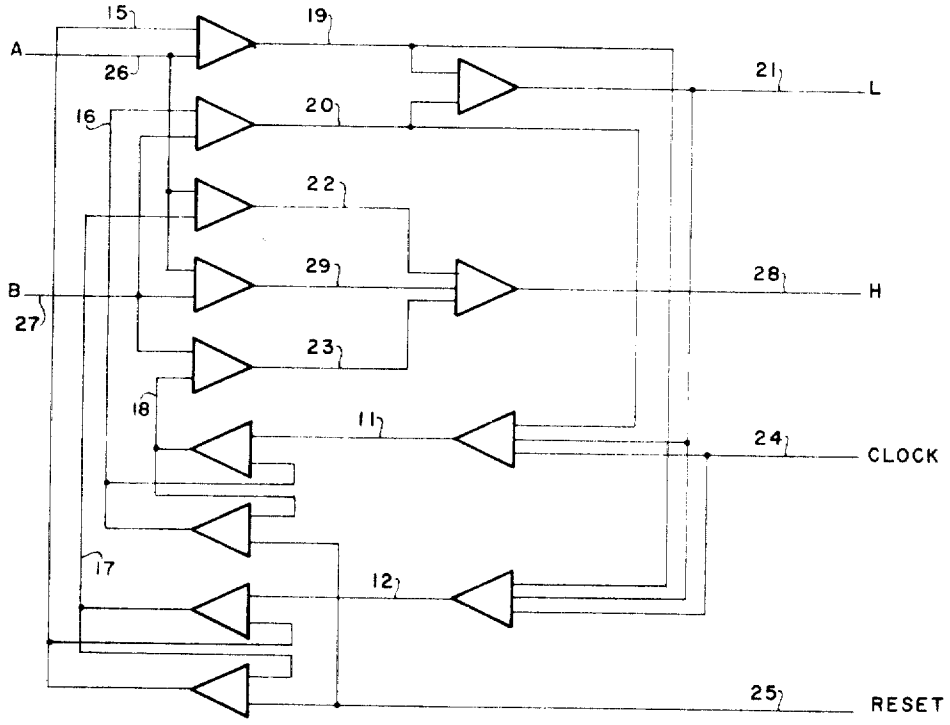

FIG. – 6

INVENTOR
KENNETH E. BATCHER
BY:
Oldham & Oldham
ATTORNEYS

MEANS FOR MERGING DATA

Filed Sept. 19, 1967

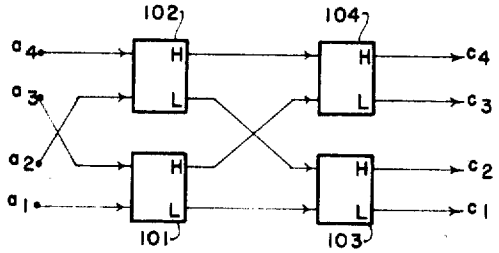

FIG.-7

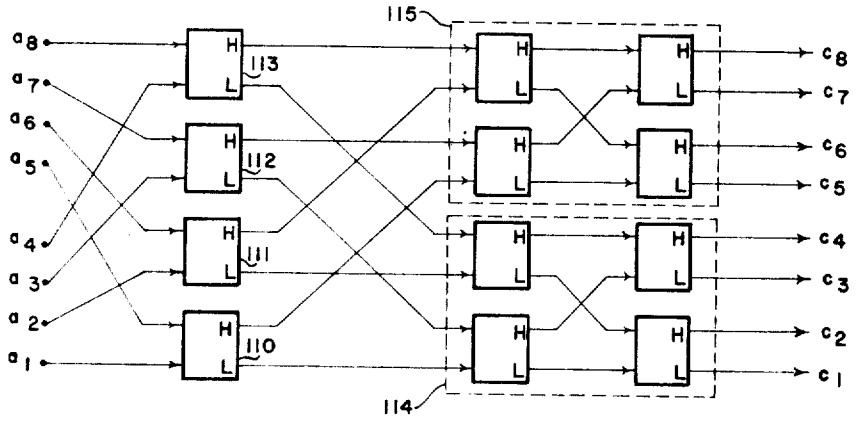

FIG.-8

INVENTOR
KENNETH E. BATCHER
BY:
Oldham & Oldham
ATTORNEYS

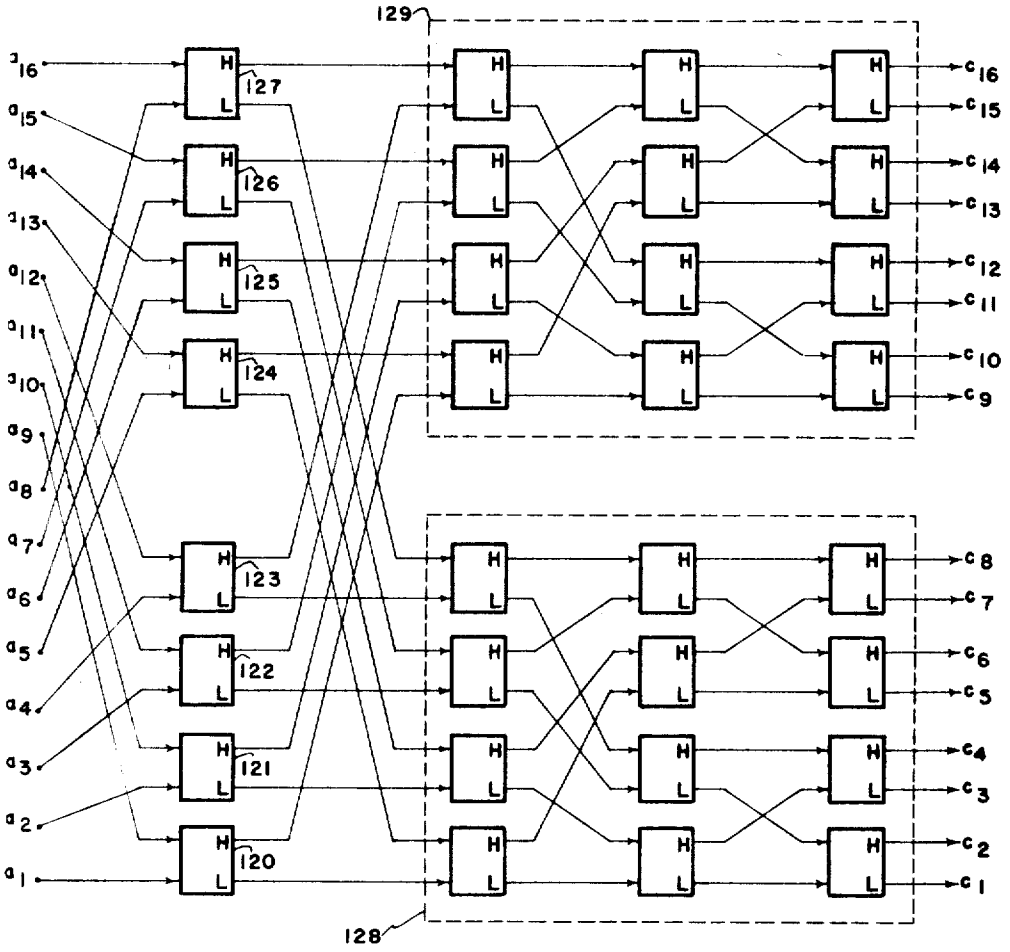

FIG. — 9

INVENTOR
KENNETH E. BATCHER
BY:
ATTORNEYS

United States Patent Office 3,428,946
Patented Feb. 18, 1969

3,428,946

MEANS FOR MERGING DATA

Kenneth E. Batcher, Akron, Ohio, assignor to Goodyear Aerospace Corporation, Akron, Ohio, a corporation of Delaware

Continuation-in-part of application Ser. No. 482,695, Aug. 26, 1965. This application Sept. 19, 1967, Ser. No. 668,809
U.S. Cl. 340—146.2 5 Claims
Int. Cl. G08b 5/00

ABSTRACT OF THE DISCLOSURE

This invention relates to a merging means, and more particularly to a unique arrangement of basic elements each adapted to receive two data, compare their magnitudes, and present the greater of the two at one output, the lesser at another output, and whereby data in random order, or ordered sequences of various numbers, can be arranged into one sequence in ascending order.

It is a continuation-in-part of patent application Ser. No. 482,695, filed Aug. 26, 1965.

Heretofore, it has been well known that there have been many and various types of merging means adapted to merge groups of numbers, but these prior art systems have been unduly cumbersome, extremely expensive, difficult to construct, and requiring many components. Various of these prior art method are mentioned in the parent application.

It is the general object of the present invention to overcome the defects in the prior art, and to provide an improvement in the apparatus defined in the parent application by providing a sorting and/or merging work which can effectively arrange randomly presented data into sequence.

A further object of the invention is to provide a network wherein positive completion of the comparison circuits can be selectively incorporated to facilitate the circuit for adaptation to selective circuit switching and control, rather than arrangement of data into merged sequences.

A further object of the invention is to provide an alternative merging network which is not limited to input sets of sequentially arranged data of fixed size but rather can be used to merge any two sets of sequentially arranged data of any size as long as the total number of data items does not exceed a predetermined limit.

For a better understanding of the invention reference should be had to the accompanying drawings wherein:

FIG. 1 is a schematic illustration of a basic comparison element to arrange two random input data into known ascending order;

FIG. 2 is a schematic block diagram of five of the elements of FIG. 1 arranged to merge four data at random into one group of four data in ascending order;

FIG. 3 is a schematic block diagram utilizing the basic components taught by FIG. 1 to arrange eight data at random into one group of data in ascending order;

FIG. 4 is a schematic block diagram illustrating the same merging ability as FIG. 3 except broken down to illustrate how the random data are first arranged into ascending groups and the ascending groups then merged into a single ascending group;

FIG. 5 illustrates the basic block diagram of FIGS. 3 and 4 with the incorporation of holding means and application to a practical usage in a telegraph network;

FIG. 6 illustrates a basic comparison element utilizing NOR logic elements; and

FIG. 7, 8 and 9 illustrate schematic block diagrams of novel merging networks.

There are many basic elements which will compare two data and present the highest of the two on one output, the lowest on another output. For example, if the data are represented by analog voltages, then a well known diode bridge circuit can be used to arrange them in the desired order. If the data are represented by digital numbers there are well known logic circuits existing for comparing such number representations and presenting the higher on one output, the lower on another. For example, the circuit of FIG. 6 would meet these requirements if the bits of the numbers are presented sequentially, most-significant bit first.

The operation of the logic circuit of FIG. 6 is as follows. It consists of thirteen NOR logic elements. (The output of a NOR logic element is a zero level if one or more of its inputs has a one level and the output is a one level if all the inputs have zero levels.)

The clock input **24** is for the purpose of disabling the levels on wires **11** and **12** during the transients of levels applied to the A(**26**) and B(**27**) input terminals. A one level is applied to the clock input **24** during transients and a zero level is applied when the A and B inputs, **26** and **27** wires **15**, **16**, **17**, **18**, **19**, **20**, and **21** all have steady levels.

Initially the circuit is put in the reset state by applying a one pulse to the reset input **25**. This puts one levels on wires **17** and **18** and zero levels on wires **15**, **16**, **22** and **23**.

Then the bits of one of the numbers to be compared are applied, one at a time, most-significant bit first, to input A(**26**) while the corresponding bits of the other number are applied to input B(**27**). A zero pulse is applied to the clock input **24** for each bit applied to input A after the circuit levels have steadied.

As long as each bit applied to input A equals the corresponding bit applied to input B wires **29**, **19** and **20** will have levels opposite that of the input bits, wires **28** and **21** will have the same levels as the inputs, and wires **11**, **12**, **15** and **16** remain at the zero level while wires **17** and **18** remain at the one level. Thus the bits seen on the H(**28**) and L(**21**) inputs will be the same as the input bits and in the same sequence.

If the two numbers being compared differ then the following action occurs the first time that the input A bit differs from the input B bit. Wire **29** has a zero level since one of the input bits is a one and the H output (wire **28**) has a one level. Wires **19** and **20** will be at opposite levels so the L output (wire **21**) has a zero level. When the clock input changes to zero either wire **11** or wire **12** will change to the one level. If the input A bit is one and the input B bit is zero then wire **12** changes to the one level with the clock pulse and wires **17** and **15** will change to the zero and one levels, respectively. The circuit is then in the "A greater than B" state. Conversely, if the input A bit is zero and the input B bit is one, then wire **11** changes to the one level with the clock pulse and wires **18** and **16** change to the zero and one levels, respectively. The circuit is then in the "A less than B" state.

If the circuit is in the "A greater than B" state it will remain in the "A greater than B" state for all succeeding bits until another reset pulse is applied. During this time the wire **22** level will be opposite that of the input A bits and the H output (wire **28**) level will be the same as that of the input A bits. The wire **20** level will be opposite that of input B and the L output (wire **21**) will be the same as that of input B. Wires **11**, **19**, **23**, **16**, and **17** stay in the zero state and wires **15** and **18** stay in the one state.

Similarly, if the circuit is in the "A less than B" state it will stay in that state until another reset pulse is

applied. The H output (wire **28**) level will be the same as the input B level and the L output (wire **21**) level will be the same as the input A level.

It is a well-known property of binary numbers that to compare two numbers one merely has to observe the most-significant place where the two numbers disagree. The number with the one bit in this place is higher than the number with the zero bit in this place. Thus the comparison circuit described above will put the bits of the higher number on its H output and the bits of the lower number on its L output. The bits are put out in the same sequence as the input numbers; one at a time, most-significant bit first.

A circuit such as that of FIG. 6 which compares two data and presents the higher on one output and the lower on another output is called hereinafter a comparison circuit. In the figures (see FIG. 1) it is represented by a block **10** with two inputs and two outputs. The higher output is labeled H while the lower output is labeled L. If the two data to be compared happen to be equal then it is immaterial which of the data is presented on which output.

In order to sort or arrange into order four random data A, B, C, and D, a circuit such as that illustrated in FIG. 2 is utilized. In the first level or first tier **12A** any two numbers are compared in elements **14A** and **16A** to provide outputs now arranged in ascending order as desired. In the second tier indicated by dotted block **18A** the highest output of the first merge in the first tier **12A** is compared with the highest output of the second merge. Similarly, the lower output from the first merge is compared with the lower output from the second merge. This automatically provides the highest and lowest values. Then, by simply comparing the lower output of the first merge in the second tier **18A** with the higher output of the second merge in the second tier **18A**, the determined order of the four data is achieved. In other words, the rule to be followed in the second tier is that the second output of the first merge, namely output **20A** is combined with the first output of the second merge, namely output **22A**, in an element **24A**. These outputs, in combination with the outputs **26A** and **28A** from elements **30** and **32**, respectively, arrange the random data into ascending order C1 through C4. This can be called a four-item sorting network.

FIG. 3 illustrates the arrangement of eight data at random designated A through H into sequential order. Essentially, two of the circuits of FIG. 2 are utilized as indicated by dotted blocks **34** and **36**. The resultant outputs from these four-item sorting networks are then arranged in sequential order. These outputs are fed to a four by four merging network indicated by dotted block **38**. In effect the highest output from block **34** is compared with the highest output from block **36** and so on in the first tier **40** of block **38**. This then automatically produces the highest and lowest outputs. Considering the top two elements in the first tier as the first merging group, and the bottom two elements as the second merging group the resultant outputs from the merges in both groups are then sent to a second tier of elements **42** where the general rule set forth above is followed. Specifically, the lower output of the first merge in the first group is compared to the higher output of the second merge in the first group while the same process takes place in the second group. The invention then contemplates a third tier **44** wherein the following merges or comparisons take place. Specifically, the first element merges the higher of the first group second merge to the higher of the second group first merge. The second element merges the lower of the first group second merge to the higher of the second group second merge, and the third element merges the lower of the first group second merge to the lower of the second group second merge. This then results in a complete arrangement of the random data A through H into arranged data C1 through C8. This is called an eight-item sorting network.

In general, a sorting network for any given number of items can be constructed by dividing the plurality of items to be sorted into two groups of approximately equal size, using two sorting means to arrange each group into monotonic order and using the merging means of the parent application to merge the two ordered pluralities into one ordered plurality. Each of the sorting means for each of two groups can in turn be constructed by dividing each group into two smaller groups and using the same principle of sorting each smaller group separately and using the merging means of the parent application to merge the two ordered smaller groups into one ordered group. By using this same principle over and over a sorting network for any number of items can eventually be reduced to the case of sorting several groups each with one or two datums. A two-item group can be sorted by a comparison element (FIG. 1) while a one-item group needs no sorting. Thus, several of the merging means of the parent application can be used to effect a sorting means for any given number of items.

FIG. 4 illustrates the same element arrangement as FIG. 3 except that it is illustrated how the eight random data are first broken into two groups of four in arranged sequence and then merged into a single sequence. These basic structural components to merge two with two or four with four can then be used to build any size data arrangement circuit and merging circuit with a small number of comparison elements.

The invention contemplates that if this same circuit is utilized with holding means incorporated in the comparison elements, a very practical, and flexible circuit arrangement can be achieved. Specifically, this set up is illustrated in FIG. 5 and indicates the holding means by numeral **50** associated with each tier of elements in the circuit network. For practical purposes, this circuit is illustrated as a telegraph line connection circuit for connecting eight telegraph lines to various cities in a random arrangement over the same circuit pattern.

Each city is given a code number and connected to one side of the network in the order of their code numbers. Specifically, if it is desired to connect telegraph line A to Los Angeles, line B to Chicago, line C to Cleveland, line D to Detroit, line E to Dallas, line F to Philadelphia, line G to St. Louis, and line H to New York, then each line should first send the code number for the specific city it is calling, that is, line A sends Code **5**, line B sends Code **7**, line C sends Code **6**, line D sends Code **4**, line E sends Code **1**, line F sends Code **3**, line G sends Code **2**, and line H sends Code **8**. The network of comparison elements with holding means **50** will order the codes so each code arrives at the right end of the network on the line connected to its corresponding city. The holding means preserve the paths each code took through the network allowing telegraph messages to be sent from each line to its called city. When a change is desired the code numbers on the telegraph lines are changed and all codes sent in again after the holding means **50** are released to permit the network to change state, for example if after the aforementioned connection it is desired to connect line A to Philadelphia and line F to Los Angeles keeping all other connections fixed then the code numbers on lines A and F are changed to **3** and **5**, respectively, and all code numbers sent into the network again. This will effect the desired change. The comparison element of FIG. 6 has holding means already incorporated in. After comparing the two numbers presented serially over inputs A(**26**) and B(**27**) and presenting the lower number on output L(**21**) and the higher number on output H(**28**) the circuit will remain in the "A greater than B" state or the "A less than B" state until a reset pulse is applied as described in the circuit description above; thus it can hold the paths the numbers took through the element and allow other data such as telegraph messages to be transmitted over the same paths.

It is readily seen that a mere instantaneous switching of the code numbers and the holding means in association with the elements will allow very rapid circuit changes to connect desired calling cities to connection cities, pro-

viding a considerable improvement in the present telegraph switchboard design.

To help describe an alternative construction of a merging means the following definition must be understood:

A sequence of data is said to be bi-tonic if it satisfies at least one of the following five conditions:

(1) It consists of an ascendingly-ordered part followed by a descendingly-ordered part.

(2) It consists of a descendingly-ordered part followed by an ascendingly-ordered part.

(3) It consists of an ascendingly-ordered part followed by a descendingly-ordered part followed by an ascendingly-ordered part in which the last datum is not greater than the first datum.

(4) It consists of a descendingly-ordered part followed by an ascendingly-ordered part followed by a descendingly-ordered part in which the last datum is not less than the first datum.

(5) It is monotonic.

It should be understood that one or more of the sequence parts in the definition may consist of only one datum and a one-datum part is regarded to be both ascendingly-ordered and descendingly-ordered.

As an example, the sequence 5, 9, 12, 15, 20, 19, 7, 2, 3, 4 is bi-tonic since it satisfies condition (3) while if the first datum in this sequence was a 1 instead of a 5 then the sequence would not be bi-tonic since 4 is greater than 1.

If we divide the example sequence into two parts:

5, 9, 12, 15, 20

and

19, 7, 2, 3, 4

and compare corresponding items in the two parts, i.e., 5 with 19, 9 with 7, 12 with 2, 15 with 3, and 20 with 4, and if we interchange the items of any corresponding pair where the item in the second half is less than the corresponding item in the first half, we obtain the two sequences 5, 7, 2, 3, 4 and 19, 9, 12, 15, 20. Notice that both of these sequences are bi-tonic and furthermore that the greatest item in the first half (7) is not more than the least item in the second half (9). This holds true for any bi-tonic sequence of an even number of data. When a bi-tonic sequence of an even number of data is split into a first half and a second half and corresponding items in the halves interchanged wherever the datum in the second half is less than the corresponding datum in the first half, the two halves are each bi-tonic and the greatest datum in the first half is never more than the least datum in the second half. Thus, to arrange such a bi-tonic sequence into monotonic order, it is sufficient to arrange each of these two bi-tonic halves into monotonic order since each item in the first half is never more than any item in the second half and so each item in the first half belongs in the first half of the final monotonic order. The fact that each of these halves is also bi-tonic can be used to help arrange each of these halves in monotonic order if the halves contain an even number of data since they in turn can be split in halves (quarters of the original sequence) and comparisons and interchanges made between their halves to obtain four quarters of the original sequence wherein each datum in a quarter belongs in the same quarter in the final monotonic order. The quarters are bi-tonic so the process could be repeated if each quarter contains an even number of data.

If the number of data in the original sequence is an exact power of two the above process could be iterated fully to arrange the data in monotonic order. If the number of data is not a power of two dummy data could be added to the sequence to round it out to a power of two, the sequence arranged in monotonic order by the above process and the dummy data removed. The dummy data has to be of such magnitude as to preserve the bi-tonic property of the original sequence.

FIG. 7 illustrates the construction of a means for arranging a bi-tonic sequence of four data into monotonic order using comparison elements as taught in FIG. 1. The bi-tonic sequence, a_1, a_2, a_3, a_4 is placed on the input lines and comparison elements **101** and **102** compare datum a_1 (the first datum of the first half of the sequence) with datum a_3 (the first datum of the second half) and datum a_2 with datum a_4 (the second data in each of the two halves). The data on the L outputs of elements **101** and **102** are compared in element **103** to place them in monotonic order and form the first half of the final monotonic order, c_1 and c_2. The data on the H outputs of elements **101** and **102** are compared in element **104** to form the second half of the final monotonic orde, c_3 and c_4.

FIG. 8 illustrates the construction of a means for arranging an 8-datum bi-tonic sequence into monotonic order using the elements taught in FIG. 1 and the construction taught in FIG. 7. The bi-tonic sequence, a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8 is placed on the input lines, comparison elements **110**, **111**, **112**, and **113** compare datum a_1 with datum a_5, datum a_2 with datum a_6, datum a_3 with datum a_7 and datum a_4 with datum a_8, respectively. A construction as taught in FIG. 7 enclosed in dotted box **114** in FIG. 8 is used to arrange the 4-datum bi-tonic sequence on the L outputs of comparison elements **110**, **111**, **112**, and **113** into monotonic order to form the first half, c_1, c_2, c_3, c_4 of the final monotonic order. Similarly dotted box **115**, also a construction of FIG. 7, arranges the data on the H outputs of comparison elements **110**, **111**, **112**, and **113** into monotonic order to form the second half, c_5, c_6, c_7, and c_8 of the final monotonic order.

FIG. 9 illustrates the construction of a means for arranging a 16-datum bi-tonic sequence, a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, a_{10}, a_{11}, a_{12}, a_{13}, a_{14}, a_{15}, a_{16} into monotonic order using the comparison elements taught in FIG. 1 and the construction of FIG. 8. Comparison elements **120** through **127**, inclusive compare corresponding data in the first and second half of the sequence, that is, element **120** compares a_1 with a_9, element **121** compares a_2 with a_{10}, etc. A construction taught in FIG. 8 shown in dotted box **128** of FIG. 9 arranges the 8-datum bi-tonic sequence on the L outputs of comparison elements **120** through **127**, inclusive, into monotonic order to form the first half, c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8 of the final monotonic order. Similarly the second half, c_9, c_{10}, c_{11}, c_{12}, c_{13}, c_{14}, c_{15}, c_{16} of the final monotonic order is formed by another construction of FIG. 8, dotted box **129** of FIG. 9, from the 8-datum bi-tonic sequence on the H outputs of comparison elements **120** through **127**, inclusive.

FIGS. 7, 8, and 9 are examples of the general rule to be followed to construct a means for arranging a bi-tonic sequence with a number of data equal to a power of two into monotonic order. The general rule is to use comparison elements, such as that of FIG. 1, to compare corresponding data in the first and second half of the sequence group the data appearing on the L outputs of the comparison elements to form a bi-tonic sequence of half as many data as the original sequence, arrange this sequence into monotonic order to form the first half of the final monotonic order and similarly group the data on the H outputs of the comparison elements and arrange this sequence into monotonic order to form the second half of the final monotonic order.

A means for arranging bi-tonic sequences into monotonic order can be used as a merging means to merge two ordered sequences into one ordered sequence by inverting one of the two ordered input sequences so that one is in ascending order and the other is in descending order and then placing the two sequences together to form a bi-tonic sequence which can then be arranged into monotonic order. If the total number of data is not an exact power of two dummy data can be added to the sequence to make it an exact power of two. To preserve the bi-tonic property of the sequence, each dummy datum should either equal the smallest possible datum, if the second sequence is in decreasing order, or equal the

largest possible datum if the second sequence is in increasing order. After the sequence is put in monotonic order all such dummy data will be either at the start or the end of the sequence since they equal either the smallest possible datum or the largest possible datum and the dummy data can be easily removed.

A merging means which first forms a bitonic sequence and then arranges the bitonic sequence into monotonic order is hereinafter called a bitonic merging means. They have the advantage over the merging means taught in the above-identified parent application of flexibility, that is, the same construction can be used to merge any two ordered sequences as long as the total number of data can be handled. For example, the construction of FIG. 9 can be used to merge an 8-datum sequence with an 8-datum sequence, or a 10-datum sequence with a 6-datum sequence, etc. The construction taught in the parent application would have to be changed every time the number of data in either sequence changes. This advantage is counterbalanced by the fact that more comparison elements are used in bitonic merging means. For example, the construction of the parent application for a merge of 8-datum and 8-datum requires only 25 comparison elements while the bitonic merging means of FIG. 9 uses 32 comparison elements.

Bitonic merging means can be used in place of the merging means of the parent application in the sorting means and communication networks set forth with reference to FIG. 5 above since they are alternative constructions of merging means using the same comparison elements.

While in accordance with the patent statutes only one best known embodiment of the invention has been illustrated and described in detail, it is to be particularly understood that the invention is not limited thereto or thereby.

What is claimed is:

1. A means for arranging into monotonic order a sequence of an even number of eight or more data initially arranged into an ascendingly-ordered part followed by a descendingly-ordered part, or into a descendingly-ordered part followed by an ascendingly-ordered part, or into an ascendingly-ordered part followed by descendingly-ordered part followed by an ascendingly-ordered part in which the last datum in the sequence is not greater than the first datum in the sequence, or into a descendingly-ordered part followed by an ascendingly-ordered part followed by a descendingly-ordered part in which the last datum is not less than the first datum in the sequence, or into monotonic order which consists of

a first plurality of electrical means for sorting or arranging into monotonic order the first, third, fifth, and so on items of the plurality with each other,

a second plurality of electrical means for sorting or arranging into monotonic order the second, fourth, sixth, and so on items of the plurality with each other, and a third plurality of electrical means for comparing,

the least item of those treated by the first means to the least item of those treated by the second means and placing the two items in order in the least and second least places of the final monotonic order, the second least item of those treated by the first means to the second least item of those treated by the second means and placing the two items in order in the third least and forth least places of the final monotonic order, the third least item of those treated by the first means to the third least item of those treated by the second means and placing the two items in order in the fifth least and sixth least places of the final monotonic order and so on, until each item treated by the first means is compared to a corresponding item treated by the second means and is placed together with its corresponding item in order in the final monotonic order.

2. A means according to claim **1** where the two parts of the even plurality are initially arranged in just the reverse order.

3. A means for sorting or arranging into monotonic order a randomly-ordered plurality of ten or more data according to claim **1** which includes

a primary means for arranging into monotonic order part of the randomly-ordered plurality,

a secondary means for arranging into monotonic order that part of the randomly-ordered plurality not treated by the primary means, and

a merging means according to claim **1** to arrange the ordered data of the primary means with the ordered data of the secondary means to arrange all data in monotonic order.

4. A means according to claim **3** where the elements include holding means to preserve the paths the data took through the network so that other data may be transmitted over the same paths in any direction.

5. A network according to claim **3** where the relay holds function on the driving of a clock pulse, and a reset pulse is provided to allow resetting of the elements when desired.

References Cited

UNITED STATES PATENTS

3,015,089 12/1961 Armstrong ..... 340—172.5

MALCOLM A. MORRISON, *Primary Examiner.*

DAVID H. MALZAHN, *Assistant Examiner.*

U.S. Cl. X.R.

340—172.5