

APPARATUS FOR BINARY ADDITION

BACKGROUND OF THE INVENTION

1. Field of the invention

5 The present invention relates to the field of binary addition. More particularly, it relates to apparatus for adding a plurality of numbers having a number of bits each.

2. Description of Prior Art

10 Binary addition plays an important role in digital systems and for this reason the speed at which a sum may be computed is of extreme importance. Basic binary addition may be performed using a ripple carry adder but this has the disadvantage that, as the size of the adder increases, carry propagation delays become
15 intolerable. Many methods of addition have been developed to speed up this process, for example those known by the names of Carry Look-Ahead, Carry Save Addition and Carry Select Addition.

The Carry Select Adder is a trade off between size and speed
20 compared to a ripple carry adder and is described in many textbooks. Two ripple-carry-adder structures are used (although any adder structure may be used), one with a hard-wired zero carry-in and the other with a hard-wired one carry-in. The sums of each adder are fed into a multiplexer which selects one of the
25 sums as an output in response to a carry-in bit, either one or zero from a previous stage. Carry-out bits are generated for each sum and forwarded to a successive stage for use by the multiplexer of that stage, (except for the first stage). With such a structure, carry propagation delay is reduced. Since two

parallel adders are required for each sum generation, the complexity of the Carry Select Adder is increased compared to a ripple carry adder.

5 It is a disadvantage of the Carry Select Adder, however, that the Adder does not work well as the number of numbers to be added increases.

It is an object of the invention to provide an improved apparatus
10 for binary addition.

SUMMARY OF THE INVENTION

According to the invention, there is provided an apparatus for binary addition comprising:

15 a rank ordered plurality of parallel connected adder trees, each of said adder trees providing a partial sum of a portion of a plurality of input numbers to be added;
a rank ordered plurality of parallel connected addition stages connected to the adder trees, each addition stage providing a
20 portion of the final result of the addition and a carry-in to a higher ordered rank addition stage; and
means for combining said portions of the final result of the addition together.

25 In the present invention, by dividing the input numbers into portions, all portions can be computed simultaneously, thus reducing the total timing delay to compute the addition directly without division.

Preferably, the size of each portion of the plurality of input numbers is in the range 2-5 bits and the size of the portions of the plurality of input numbers input to a said adder tree is less than or equal to the size of the portions input to the next 5 highest ranked adder tree.

Preferably, each adder tree comprises a rank ordered plurality of first stage adders, each of said first stage adders receiving said portion of two said input numbers and providing a sum 10 output; and a plurality of further stages of adders, each adder of each further stage receiving the sum output from two adders of a previous stage and providing a sum output to a subsequent stage, the sum output of the adder of a last further stage providing said partial sum.

15

Preferably, at least one addition stage receives a part of the partial sum from two said adder trees, a first part from a lower ordered adder tree and a second part from the next higher ordered adder tree, the second part being the same bit length as the 20 portions input to the adder tree from which the second part is output and the first part having a bit length less than or equal to that of the second part.

Preferably at least one addition stage comprises first and second 25 binary adders having a presumed "one" carry and a presumed "zero" carry respectively and each providing a sum output and selection means for receiving a carry in from the lower rank ordered addition stage and for selecting the sum output of one of the binary adders as said portion of the final result based on the

carry-in.

In the described embodiment, a predetermined number of parallel-connected adder trees are provided, each with N input binary numbers, where N is an integer, having a rank ordered plurality of adders, for adding the numbers. The m bits of each N input number are split into bit slice cells of predetermined length and are fed into input terminals of each adder tree. The first stage adders of each adder tree, which comprise two-input adders, are adapted to receive these bit slice cells. The same bit slices of the N input numbers are fed into a respective input terminal of an adder in the first stage of the adder tree. Each adder tree comprises a predetermined number of adders and an output terminal, which provides a portion of the sum of the N input numbers. All these portions of the sum of the N input numbers are combined together to form the final result of the addition by implementing Carry Select Addition. The high range of bits of each portion of the sum are added with the corresponding low range of bits of a higher ordered portion of the sum (except the highest ordered portion of the sum). Two parallel additions are implemented with presumed zero carry-in and presumed one carry-in. The first sum of the two parallel additions is a sum generated without carry-in and the second sum of the two parallel additions is a sum generated with a carry-in. When the carry-in from the lower ordered portion of the sum is calculated it is used to pick the proper sum of the two parallel additions.

BRIEF DESCRIPTION OF THE DRAWINGS

An embodiment of the invention will now be described, by way of example, with reference to the accompanying drawings, in which:

5 Fig.1. is the block diagram of an embodiment of the invention.

Fig.2. is a detailed diagram of one of the adder trees of the embodiment of Figure 1.

Fig.3. is a detailed diagram of one of the carry select
10 addition stages of the embodiment of Figure 1.

DESCRIPTION OF THE PREFERRED EMBODIMENT

Fig.1 is the block diagram of the whole adder tree circuit of the
15 preferred embodiment of the invention. Although the preferred form of the adder tree has fourteen, nineteen-bit input numbers with a sign bit in the most significant bit position, the invention may be practised with any number of integer input numbers and bits. Similarly the adders illustrated here are not
20 limited to any particular type.

In Fig.1, the thick vertical arrow lines 101-105 represent multi-input paths, which each have fourteen inputs in this embodiment. The big rectangle boxes 111-115 with label "adder tree" inside
25 represent an adder tree structure, which computes a portion of the sum of the fourteen input numbers. The boxes 121-125 represent the portions of the sum computed from adder trees 111-115 and the bit length of each portion. The boxes and circles 131-135 with "+" represent addition stages for computing the

portions of the sum from adder trees 111-115, which implement Carry Select Addition to reduce timing delays (except addition 131 for which there is no carry-in delay). The box 141 with label "Combine" represents a combination computation, which forms the
 5 final result of the addition with each portion of the sum computed from addition stages 131-135 and with the lowest ordered bits of the first portion of the sum 121, which are bits one to zero of the first portion of the sum in this embodiment.

10 The adder tree circuit consists of five parallel adder trees 111-115 for adding bit-sliced input numbers, which compute from lowest ordered to highest ordered. Adder tree 111 has fourteen input terminals for receiving bits zero to one of the fourteen input numbers. Similarly, adder trees 112-115 each have fourteen
 15 input terminals for receiving bits five to two, bits nine to six, bits thirteen to ten and bits eighteen to fourteen, of the fourteen input numbers.

The inputs to adder tree 111, which are the bits one to zero of
 20 the fourteen input numbers, are fed to through path 101. Similarly, the inputs to adder tree 112, which are the bits five to two of the fourteen input numbers, the inputs to adder tree 113, which are the bits nine to six of the fourteen input numbers, the inputs to adder tree 114, which are the bits
 25 thirteen to ten of the fourteen input numbers and the inputs to adder tree 115, which are the bits eighteen to fourteen of the fourteen input numbers, are fed to through paths 102, 103, 104 and 105 respectively.

Adder tree 111 has an output 121 for providing a six-bit length sum for bits one to zero of the fourteen input numbers. As there are fourteen inputs fed into adder tree 111, only bits one to zero of the output 121 will be used to form bits one to zero of the final result of the addition directly. Bits five to two are the accumulated carry-in of the sum for bits one to zero of the fourteen input numbers and will be sent to the higher ordered output 122 as a carry-in.

Adder tree 112 has an output 122 for providing an eight-bit length sum for bits five to two of fourteen input numbers. Bits three to zero of the output 122 will be used to form bits five to two of the final result of the addition together with the accumulated carry-in from bits five to two of lower ordered output 121. Bits seven to four of the output 122 are the accumulated carry-in of the sum for bits five to two of the fourteen input numbers and will be sent to the higher ordered output 123 as a carry-in.

Adder tree 113 has an output 123 for providing an eight-bit length sum for bits nine to six of fourteen input numbers. Bits three to zero of the output 123 will be used to form bits nine to six of the final result of the addition together with the accumulated carry-in from bits seven to four of lower ordered output 122, bits seven to four of the output 123 are the accumulated carry-in of the sum for bits nine to six of the fourteen input numbers and will be sent to the higher ordered output 124 as carry-in.

Adder tree 114 has an output 124 for providing an eight-bit length sum for bits thirteen to ten of fourteen input numbers. Bits three to zero of the output 124 will be used to form bits thirteen to ten of the final result of the addition together with the accumulated carry-in from bits seven to four of lower ordered output 123. Bits seven to four of the output 124 are the accumulated carry-in of the sum for bits thirteen to ten of the fourteen input numbers and will be sent to the higher ordered output 125 as a carry-in.

10

Adder tree 115 has an output 125 for providing a nine-bit length sum for bits eighteen to fourteen of fourteen input numbers. Bits three to zero of the output 125 will be used to form bits seventeen to fourteen of the final result of the addition together with the accumulated carry-in from bits seven to four of lower ordered output 124, only bits eight to four of the output 125 are the accumulated carry-in of the sum for bits eighteen to fourteen of the fourteen input numbers and will be used to form bits twenty-two to eighteen of the final result of the addition.

20 Addition stages 131-135 are adders and four of them, 132-135, are implemented with Carry Select Addition. Addition stage 131 has a first input terminal for receiving bits five to two of output 121. Similarly, addition stages 132-134 each have a first input terminal for receiving bits seven to four of output 122-124, respectively. Addition stage 131 has a second input terminal for receiving bits three to zero of output 122. Similarly, addition stages 132-134 each have a second input terminal for receiving bits three to zero of output 123-125, respectively. Addition stage 131 has a first output terminal 162 for providing bits five

to two of the final result of the addition. Similarly, addition stage 132-134 each have an output terminals 163-165 for providing bits nine to six, bits thirteen to ten, bits seventeen to fourteen, respectively, of the final result of the addition.

5 Addition stage 131 has a second output terminal 151 for providing the most significant bit of the output 162 as a carry-in. Similarly, addition stages 132-134 each have an output terminals 152-154 for providing the most significant bit of the output 163-165 to the next adder 133-135 as a carry-in. Addition stage 135

10 has only one other input terminal for receiving bits eight to four of output 125 and has only one output terminal 166 for providing bits twenty-two to eighteen of the final result of the addition.

15 Combination computation 141 has six inputs, which are bits one to zero from output 121, bits five to two from addition 131, bits nine to six from addition 132, bits thirteen to ten from addition 133, bits seventeen to fourteen from addition 134 and bits twenty-two to eighteen from output 135. This forms the final

20 result of the addition.

Adder tree structure 111-115 will now be described in greater detail. Although the preferred form of the adder tree structure has fourteen nineteen-bit input numbers with a sign bit in the

25 most significant bit position, the invention may be practised with any integer input numbers and bits. Similarly the adders illustrated here are not limited to certain type.

Fig.2 is a detailed sub-block of adder tree 111. There are seven two-input adders 221-227 in the first stage of adder tree

receiving bits one to zero of fourteen input numbers. Adder 221 has two input terminals for receiving bits one to zero of the first input number a_1 and the second input number a_2 . Similarly, adders 222-227 each have two input terminals for receiving bits one to zero of the third input number a_3 and the fourth input number a_4 , the fifth input number a_5 and the sixth input number a_6 , the seventh input number a_7 and the eighth input number a_8 , the ninth input number a_9 and the tenth input number a_{10} , the eleventh input number a_{11} and the twelfth input number a_{12} , the thirteenth input number a_{13} and the fourteenth input number a_{14} , respectively. There are three adders 241-243 in the second stage of adder tree receiving sums from the first stage. Adder 241 has two input terminals for receiving the sum 231 from adder 221 and the sum 232 from adder 222. Similarly, adder 242-243 each have two input terminals for receiving the sum 233 from adder 223 and the sum 234 from adder 224, the sum 235 from adder 225 and the sum 236 from adder 226, respectively. Since there are only seven sums generated from the first stage of the adder tree, the seventh sum 237 from adder 227 will be passed through directly to next stage of the adder tree. There are two adders 261-262 in the third stage of adder tree receiving sums from the second stage.

Adder 261 has two input terminals for receiving the sum 251 from adder 241 and the sum 252 from adder 242. Similarly, adder 262 has two input terminals for receiving the sum 253 from adder 243 and the sum 237 directly from adder 227 of the first stage. Adder 281 is the only adder in the fourth stage of adder tree and has two input terminals for receiving the sum 271 from adder 261 and the sum 272 from adder 262. Output 121, as shown in Fig. 1,

provides an output for adder 281.

Similarly, adder trees 112-115 each have the same structure as adder tree 111. The adders in the first stage of adder trees 112-115 each receive bits five to two, bits nine to six, bits thirteen to ten, bits eighteen to fourteen, respectively, of fourteen input numbers a_1 - a_{14} . The adders in the fourth stage of adder trees 112-115 each output a portion of sums of fourteen input numbers and pass to addition computation 131-135 as shown in Fig.1.

10

Addition stages 132-135 will now be described. Fig.3 is a detailed block diagram of addition stage 132, which implements Carry Select Addition, to reduce timing delays. Adders 311 and 312 receiving bits seven to four of output 122 and bits three to zero of output 123 are computed in parallel, with adder 311 having a hard-wired "one" carry and adder 312 having a hardwired "zero" carry. Both outputs 321 and 322 are fed to selection logic 331. The most significant bit from the lower ordered portion of sum $out_2[6]$ will also be fed into selection logic 331 on line 151 as a carry-in 323. The selected sum out_3 based on carry-in 323 will be passed to combination computation 141 to form the final result of the addition. The most significant bit of the selected sum out_3 will be passed to the next addition stage 133 on line 152 as a carry-in.

25

Addition stages 133 and 134 have the same structure as addition stage 132. Addition stage 135 has the similar structure as addition stage 132 except that there is only one input terminal for receiving bits eight to four of output 125 for both parallel

adders and a carry-in on line 154. As addition stage 135 is the highest ordered portion of sum, there is no carry-in bit generated.

5 In selecting the bit slice cell size, the following factors should be taken into consideration:

1. So that the computation of the adder tree can be simplified and thus higher speed achieved, each bit slice cell
10 should preferably be of a length 2-5 bits.

2. The lengths of the bit slice cells should be similar and generally increase from the lowest significant bit slice cell to the highest significant bit slice cell. Adjacent bit slice
15 cells can be of the same length and increases in length are preferably no more than a single step. This will result in similar propagation delays in each adder tree, which will optimise the speed of the binary addition as a whole.

20 3. The size of the two parts of the output sum from each adder tree which are input to the addition stages are preferably chosen so that the number of bits of the lower range part is the same as that of the respective bit slice cell input to the adder tree and so that the number of bits of the higher range part do
25 not exceed the number of bits of the lower range part of the output sum of the next higher ordered adder tree, to simplify the computation of the addition stages.

It should be noted while the invention has been described in the

context of a preferred embodiment, it will be apparent to those skilled in the art that the present invention may be modified in numerous ways and may assume many embodiments other than that specifically set out and described above. Accordingly, it is
5 intended by the appended claims to cover all modifications of the invention which fall within the scope of the claims.

CLAIMS

1. Apparatus for binary addition comprising:
 - 5 a rank ordered plurality of parallel connected adder trees, each of said adder trees providing a partial sum of a portion of a plurality of input numbers to be added;
 - a rank ordered plurality of parallel connected addition stages connected to the adder trees, each addition stage
 - 10 providing a portion of the final result of the addition and a carry-in to a higher ordered rank addition stage; and
 - means for combining said portions of the final result of the addition together.
- 15 2. Apparatus as claimed in Claim 1 wherein the size of each portion of the plurality of input numbers is in the range 2-5 bits.
3. Apparatus as claimed in Claim 1 or Claim 2 wherein the size
- 20 of the portions of the plurality of numbers input to a said adder tree is less than or equal to the size of those portions input to the next highest ranked adder tree.
4. Apparatus as claimed in any one of the preceding Claims
- 25 wherein each adder tree comprises a rank ordered plurality of first stage adders, each of said first stage adders receiving said portion of two said input numbers and providing a sum output; and a plurality of further stages of adders, each adder of each further stage receiving the sum output from two adders of



a previous stage and providing a sum output to a subsequent stage, the sum output of the adder of a last further stage providing said partial sum.

5 5. Apparatus as claimed in any one of the preceding Claims wherein at least one addition stage receives a part of the partial sum from two said adder trees.

6. Apparatus as claimed in Claim 5 wherein the parts comprise
10 a first part from a lower ordered adder tree and a second part from the next higher ordered adder tree.

7. Apparatus as claimed in Claim 6 wherein the second part is the same bit length as the portions of the numbers input to the
15 adder tree from which the second part is output.

8. Apparatus as claimed in Claim 6 or Claim 7 wherein the first part has a bit length less than or equal to that of the second part.

20

9. Apparatus as claimed in any one of the preceding Claims wherein at least one addition stage comprises:

first and second binary adders having a presumed "one" carry and a presumed "zero" carry respectively and each providing a sum
25 output; and

selection means for receiving a carry in from a lower rank ordered addition stage and for selecting the sum output of one of the binary adders as said portion of the final result based on the carry-in.



ABSTRACT

APPARATUS FOR BINARY ADDITION

5

An N-input m-bit high-speed adder tree circuit, where N and m are integers, employing Carry Select Addition and bit-sliced design configuration, is disclosed. The m bits of each input are split
10 into bit slice cells of predetermined length, which is determined according to total number of bits of input, so as to minimize the timing delay through the whole adder tree. Sum computation of each bit slice cell is performed independently and combined together to form the final result of the addition by implementing
15 Carry Select Addition.

Fig. 1

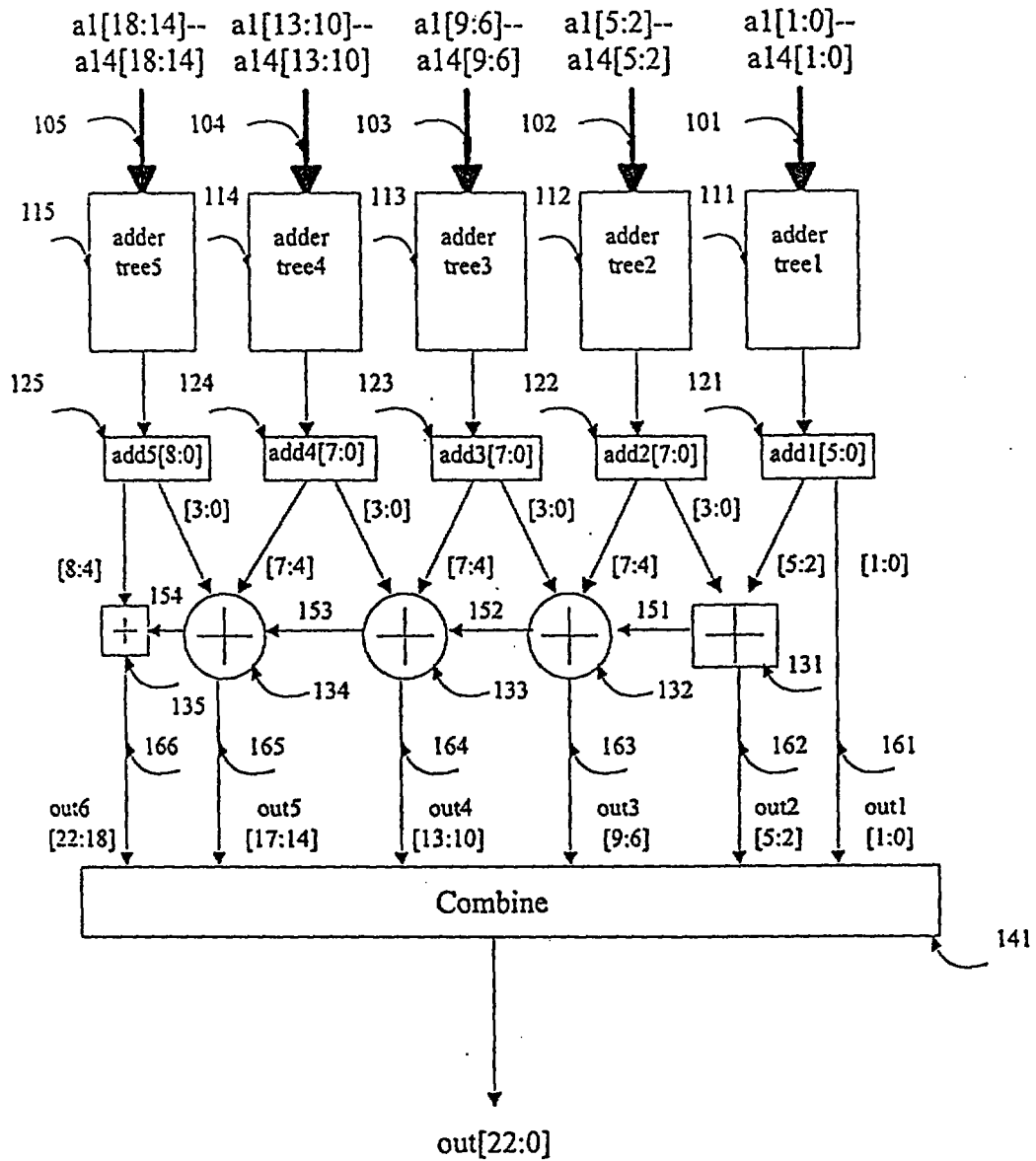


Fig.1

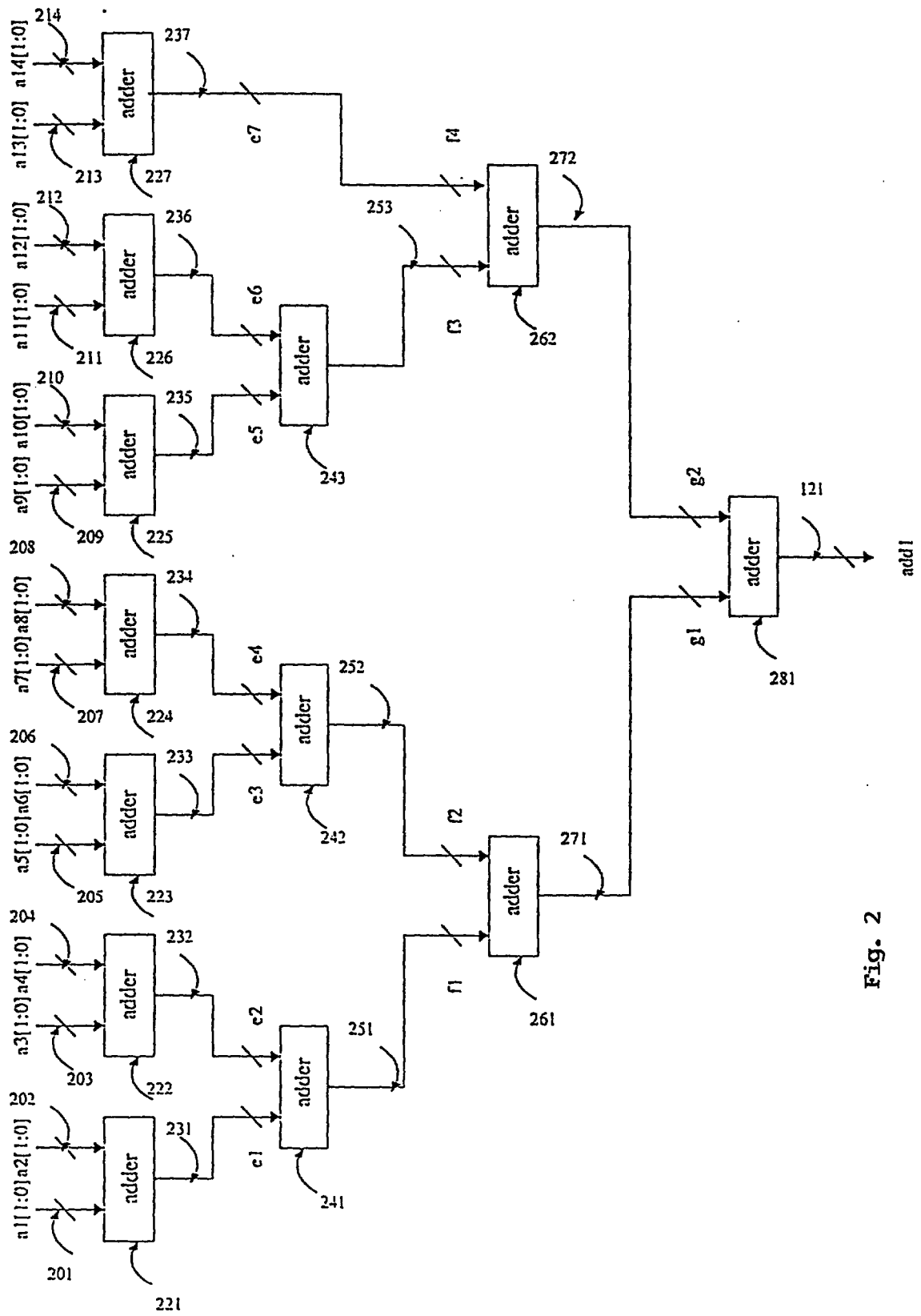


Fig. 2

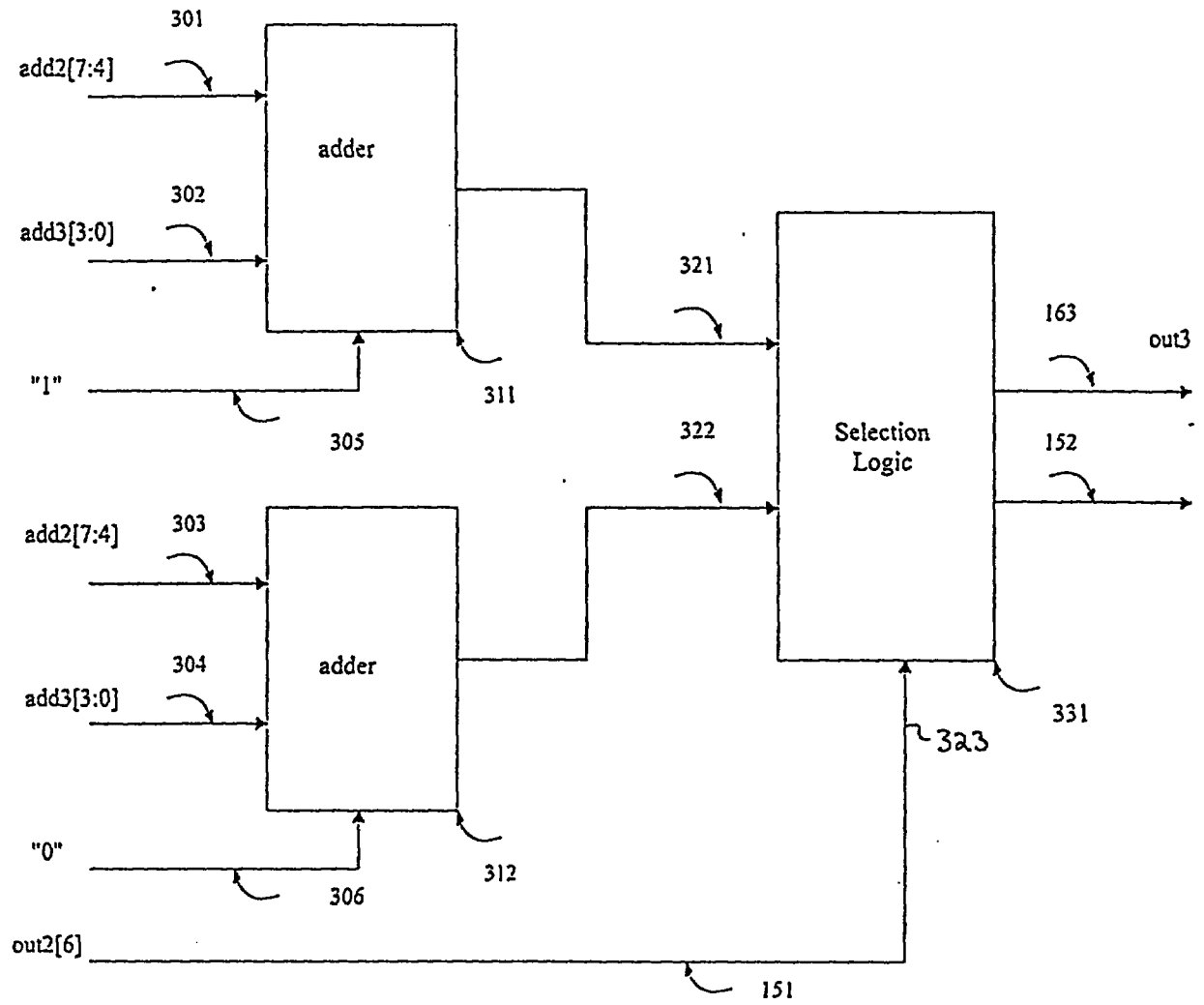


Fig. 3