REAL-TIME MULTIPLIERS

The present invention generally relates to a real-time multiplier (high speed multiplier) for use in a digital signal processing system.

In general, a digital signal processing system does not always deal with positive values. For instance, considering a case that an A/D (analog to digital) converter converts analog input data into a digital signal to obtain a new value by subtracting so-converted digital signals, when a subtraction of A-B is implemented, if the value A is bigger than the value B the output will be produced as a positive value, where the output data is, of course, valid. However, if the value A is smaller than the value B the output data will be produced as a negative value, undesirably being an invalid value.

Therefore, it is understood that the digital code data must be able to deal with not only the positive value (number) but also the negative value. In practice, to meet this end conventional digital signal processing systems adopt various code systems.

In one such conventional system a signed-magnitude code is used, which has an extra bit at the foremost of the code additionally for indicating whether the value of the code is positive or negative. This code system is divided into a part for indicating the absolute value of the code and another part for indicating the sign value thereof.

In another system the so-called 2's complement code is used. The basic principle of this code is to use the concept of complement numbers. Being consistently

calculable both for addition and subtraction, this code system is widely used in general digital signal processing systems.

In a general digital signal processing system, the same result can be obtained regardless of the code system. When, however, a very high frequency (maybe, tens of MHz) is used for the operating clock, the various characteristics, particularly the speed characteristic of the digital signal processing system is strongly dependent upon the code system. The substantial digital signal processing system can be regarded as a combination of an adder, a subtractor, a multiplier and a divider, all of which is for processing correspondingly an input data code (for example, the straight code, the signed-magnitude code or the 2's complement code).

Of those operations of the adder, the subtractor, the multiplier and the divider above, the operations of the multiplier and the divider take a relatively long time. Therefore, the system speed is essentially dependent upon the construction of the multiplier and the divider.

It is well known that the 2's complement code system is advantageous in that the negative numbers can be easily expressed and operated so that the consistency for both the addition and subtraction can be used fruitfully in operating the numbers.

Referring to Figures 1A and 1B, the addition (X+Y) can be implemented by the circuit of Figure 1A, while the subtraction (X-Y) by the circuit of Figure 1B. It is accordingly noted that the difference between the above two circuits lies in the state of the carry input terminal Cin and an inverter 108 included only in the circuit of

Figure 1B. That is, when the carry input signal is 0 the circuit of Figure 1A implements the addition and, when the carry signal is 1 the circuit of Figure 1B implements the subtraction.

According to the general theory of the binary operation, all the bits of a binary number (addend) are added to the corresponding bits of a binary number to be added so as to perform the addition. However, for the subtraction of two binary numbers, all the bits of an addend are added to the corresponding bits of the 2's complement number of the binary number to be subtracted. It should be noticed in advance that a 2's complement number can be obtained by adding 1 to the 1's complement of a number. This relation can be expressed as follows;

$$A - B = A + (-B) = A + \overline{B} + 1 \quad \ldots\ldots\ldots\ldots\ldots\ldots (1)$$

The formula (1) can be embodied as an adder/subtractor shown in Figure 2.

In the case where 2'S complement codes are to be multiplied, the 2'S complement codes are divided into a sign and an absolute value (magnitude of the code) and the absolute value is multiplied first and then the sign is corrected separately.

Considering only the magnitude of the code, it can be obtained by repeatedly implementing the 2's complement addition. However, if the multiplier and the multiplicand are the codes of big number, then the system must be complicated to implement the addition. To settle this problem, a known apparatus for simplifying the hardware of the system is used in general, in which the apparatus

calculates the partial sums by adding the two codes. Then the partial sums are added again to each other to obtain the multiplied result. When adapting such an apparatus, it is the most important drawback that the operation speed is considerably reduced.

There is disclosed another apparatus to speed up the operation, which introduces a pipeline-structured multiplier. In Figure 3, a conventional magnitude multiplier having the pipeline structure is described. All latches in the drawing are synchronized with the system clock. The latch circuit 301 latches the input value X of n-bit and the respective adders 310, 313, 317, 323 are all provided with the carry signal Cin of logic low state.

Thereafter, if the input value Y of n-bit is latched in the latch circuit 302, the gates 303 and 304 perform the ANDing operation of the value Y with the least significant bit (LSB) $X_0$and the second LSB bit X_1, respectively. At the same moment, latch 306 latches the input value Y. Then, output of the adder 310 is latched at the latch circuit 311 to be added to the output of the AND gate 308, at the adder 313. Thus, the output Sn, or partial sum, of the latch circuit 311 becomes

$$Sn = X \cdot Y_0 + X \cdot Y_1 \cdot 2$$

Further, the partial sum Ps at the output of latch 316 in this case is

$$Ps = X \cdot Y_0 + X \cdot Y_1 \cdot 2 + X \cdot Y_2 \cdot 2^2$$

In this way, partial sums are obtained at each adder and the result will be transferred to a next stage to be added to the next adder.

Such a magnitude multiplier employing the pipeline structure has the advantage of speeding up the operation. It has, however, the disadvantage that this magnitude multiplier can not deal with the 2's complement.

Accordingly, specific embodiments of the present invention aim to provide a magnitude multiplier capable of increasing operation speed by employing the principle of the 2's complement.

According to the present invention a circuit is provided arranged to multiply 2's complement codes, the codes comprising an n-bit multiplier and an m-bit multiplicand, the circuit comprising:

multiplication means for performing a multiplication by multiplying the m-bit multiplicand by each of the n bits of the multiplier;

a first stage of a partial sum formation means comprising: first adding means for adding together a first and second product, the first product being obtained by multiplying together the m-bit multiplicand and a first, least significant, bit of the multiplier, and the second product being obtained by multiplying together the m-bit multiplicand and a second bit of the multiplier; a first stage of a partial sum correcting means for receiving a most significant bit of the first product and a most significant bit of the second product and logically OR-ing them together; and a first corrected partial sum generating means for receiving an output from the first adding means and an output from the first stage of the partial sum correcting means, and generating a first corrected partial sum;

a second stage of the partial sum formation means comprising: second adding means for adding together the

first corrected partial sum and a third product, the third product being obtained by multiplying together the m-bit multiplicand and a third bit of the multiplier; a second stage of a partial sum correcting means for receiving a most significant bit of the third product and a most significant bit of the first corrected partial sum and logically OR-ing them together; and a second corrected partial sum generating means for receiving an output from the second adding means and output from the second stage of the partial sum correcting means and generating a second corrected partial sum;

where n is a number greater than 3, successive stages of the partial sum formation means are provided for the generation of third through to last corrected partial sums, each successive stage being substantially identical to the second stage; and

a 2's complement correcting means for subtracting the last corrected partial sum from a value produced by NANDing the most significant bit (MSB) of the multiplier and the m-bit multiplicand.

Preferably, said 2's complement correcting means comprises:

means operably coupled to said multiplicand and said multiplier for selectively transferring said multiplicand on the basis of the MSB of said multiplier; and

means operably coupled to said transferring means, for subtracting the last corrected partial sum from the output of said transferring means.

The invention includes a pipeline multiplier having a plurality of latch circuits and a plurality of adders,

and including a circuit substantially as herein described.

The invention also includes a digital signal processing system comprising a circuit substantially as herein described.

For a better understanding of the invention and to show how the same may be carried into effect, reference will now be made, by way of example, to the accompanying diagrammatic drawings, in which:

Figures 1A and 1B are respectively a known adder and a known subtractor for operating 2's complement codes;

Figure 2 is a known conventional adder/subtractor for general use;

Figure 3 is a known conventional pipeline multiplier;

Figure 4 is a circuit diagram of a pipeline multiplier according to an embodiment of the present invention; and

Figure 5 is an example illustration for showing the multiplication of 5x12 according to an embodiment of the present invention.

Referring to Figure 4, there is shown a magnitude multiplier in which the latch circuits, AND gates, and adders are like the ones shown in Figure 3. In the drawing, it is different from the conventional magnitude multiplier in that the most significant bit (MSB) values of the output of the AND gates 303, 304 are applied to an OR gate 401, the output of the OR gate 401 being applied to the latch 311, and that the MSB output from the AND gate 308 and the MSB output from the latch 311 are applied to an OR gate 402 to correct a partial sum therein. In

the same way, the MSB output from the latch 318 and the MSB output from the latch 319 are provided to a NAND gate 322, output of the NAND gate 322 being coupled to the input terminals of the adder 323 of which the other input terminal is coupled to the output terminal of the latch 320, so as to correct the 2's complement codes. In this way, a subtracting operation for the outputs of the NAND gate 322 and the latch 320 will be implemented.

Furthermore, a plurality of the sections same as the section 40a which is composed of the latch circuits 305, 306, 311, 312, the AND gate 308, the OR gate 402 and the adder 313 can be successively coupled to the respective corresponding output terminals of the section 40b.

Referring now to Figure 5, a specifically illustrated example is given, in which a multiplicand X is of 12-bit data and a multiplier Y is of 5-bit. This embodiment shows an example of multiplying the two values X and Y.

For convenient explanation, a theory of converting from the straight code to the 2's complement code will be shortly described hereinbelow. When it is considered that a multiplicand R is multiplied by a multiplier F, both two

code values being 2's complement codes, the respective values can be rewritten as follows

$$F = F_{n-1},\ F_{n-2},\ \ldots.,\ F_2,\ F_1,\ F_0$$
(for example, F = 101001) .....(2)

$$R = G_{n-1},\ G_{n-2},\ \ldots.,\ G_2,\ G_1,\ G_0$$
(for example, R = 011001) .....(3)

The magnitudes (absolute values) of the above 2's complement codes are

$$||F|v = -F_{n-2} \cdot 2^{n-1} + \sum_{i=0}^{n-2} F_i 2^i \quad \ldots\ldots\ldots(4)$$

$$||R|v = -R_{n-1} \cdot 2^{n-1} + \sum_{i=0}^{n-2} R_i 2^i \quad \ldots\ldots\ldots(5)$$

In the above formulas, if the MSB bit is 0 the 2's complement code is positive, while if the MSB is 1 the 2's complement code is negative. Therefore, the values of the 2's complement codes can be calculated as follows

$$Fv = -2^5 + 2^3 + 1 = -23 \text{ (Negative)} \quad \ldots\ldots\ldots(6)$$

$$Rv = 2^4 + 2^3 + 1 = 25 \text{ (Positive)} \quad \ldots\ldots\ldots(7)$$

In a similar operation to that of Figure 4, the multiplier of Figure 5 having a pipeline structure, performs repeatedly the adding operation, as described above, so as to implement a desired multiplication. If the multiplicand F and the multiplier R are converted into the 2's complement codes, the values are calculated as follows:

$$|R \times |F = (|F) \times \{-R_{n-1} \cdot 2^{n-1} + \sum_{i=0}^{n-2} R_i 2^i\}$$

$$= |F(-R_{n-1}) \cdot 2^{n-1} + \ldots + |F \cdot R_1 2^1 + |F \cdot R_0 2^0 \quad \ldots (8)$$

where the R_i is of either 0 or 1. Therefore, in order to achieve the multiplication of two 2's compliment codes, the multiplicand F is repeatedly added to itself, after shifted at every addition, according to the value of R_i. As far as this operation is concerned, it is the same as the adding operation of the straight codes. However, due to a specific characteristic of the 2's complement codes, when operating the MSB bits, the multiplying operation of the 2's complement codes becomes different from that of the straight codes. Namely, if the value of the MSB bit, R_{n-1}, is 0, then there is no problem in implementing the multiplication of the 2's complement, as in case of implementing the straight codes. But, if the value of the MSB bit is 1, then it is necessary for the multiplier to

subtract the MSB bit, i.e., 1 in this case, from the partial sums added up to that time.

According to the basic theory of the 2's complement code, in order to get a 2's complement code, the entire bits of a code should be inverted first (which is a 1's complement), and then to the inverted result is added a binary 1. As shown in Figure 1B, therefore, the subtraction is carried out by inverting the entire bits of a multiplicand's code and by applying a carry signal of the logic 1 to the carry terminal Cin.

Based upon this principle, the correction of the 2's complement codes can be carried out by, as illustrated in Figure 4, NANDing the outputs of the latches 318, 319 at the NAND gate 322 and adding the output of the NAND gate 322 to the output of the latch 320.

Hereinbelow, in order to consider the correction of partial sums produced by the OR gates 401, 402 and 403, an example will be taken to 4-bit 2's complement codes. That is, if the 2's complement codes are given as F = 1100 and R = 1011,

$$Fv = -2^3 + 2^2 = -4$$

$$Rv = -2^3 + 2^1 + 2^0 = -5$$

Thus, the multiplication is

F x R = -4 x -5 = 20

```
          1 1 0 0
      x |   1 0 1 1
        |_____________
          1 1 0 0   ]
                    ] F
        0 0 0 0     ]
      ---------------
      1 1 0 0
    1 1 0 0
  ---------------
  X X X X X X X X
```

For this operation, the multiplier and the multiplicand are respectively logically multiplied by the AND gates 303, 304, respective outputs of the above gates being provided to the adder 310, so that the output added with each other may be produced by way of the latch 311.

Thus, by means of using the specific characteristics of the 2's complement codes, the multiplication can be implemented for two codes by adding the two codes. At this time, a sign extension should be made, which is remarked as F in the above formula. The sign extension can be expressed again

```
      ←┐
       1 1 1 0 0
    +| 0 0 0 0 0
```

```
└──────────
  1 1 1 0 0 ──────── Partial Sum
```

The result of the above calculation is added to the adder 313 so as to be added to the multiplicand logically ANDed with the second bit of the multiplier. Then, the MSB's of the latch 311 and the AND gate 308 are logically ORed by the OR gate 402 and the output of the OR gate 402 is provided to the latch 316, the output of which is

```
    1 1 1 1 0 0 ──────────Partial Sum
+|  1 1 0 0 0 0
 └──────────────
  1 1 0 1 1 0 0 ──────────Partial Sum
  └──────────────── Overflow
```

It is noted from the above exemplary operations that when the sign extension is made, an overflow is produced. However, since the overflow is used as it is at the next stage and the sign extension is naturally made, it is not necessary to correct the codes.

In this way, if n-bit multiplier and multiplicand are respectively applied to the latches 301 and 302, the LSB_0 (least significant bit) and LSB_1 of the multiplier respectively provided to the AND gates 303, 304 so as to be logically ANDed (multiplied) with the n-bit multiplicand, the outputs of the AND gates each being added with each other by the adder 310. In this embodiment, the OR gate 401 is used to correct the partial sum, produced by adding of the outputs from the AND gates

303, 304. The outputs of the adder 310 and the OR gate 401 are latched at the latch 311. Consequently, the same operation will be repeated until the MSB of the multiplier is processed. Finally, the MSB from the latch 318 and n-bit multiplicand passed through the latch 319 are logically NANDed by the NAND gate 322.

Next, the subtraction between the output of the NAND gate 322 and the output of the latch 320 will be implemented by adding the above outputs by means of the adder 323, while carry terminal Cin is receiving an inverting logic 1, contrarily to the inverting logics of carry terminals Cin of the adders 310, 313, 317. Therefore, as the result of the above operations, a multiplication of F x R will be successfully performed.

In contrast to the above case of not requiring for the correction of the 2's complement, another example requiring for the correction of the 2's complement will now be taken. If a partial sum is for example a binary 11XXXX (where X represents a "don't care"), there can exist two cases. That is, the one is a logic 1 which indicates a requirement of multiplication. The other is a logic 0 which indicates an unnecessariness of multiplication. In practice, when the logic is 0, there is no difficulty in multiplying the codes. However, when the logic is 1, the operation of "a partial sum + a value

to be multiplied" is expressed as follows, according to the characteristics of the multiplier.

```
      1 1 X X X X
+   | 0 0 X X X
    ---------------
  0 1 1 X X X
```

As can be understood from the above operation, the result of the partial sum operation has caused the sign of the code to be inverted from a negative number to a positive number. In order to correct this miscalculation, the OR gates 401, 402, 403 are used. Furthermore, if the result of the partial sum operation inverts the sign from the positive to the negative, the OR gates 401, 402, 403 again produce the MSB, by which the multiplication can be accomplished.

Referring to Figure 5, another example is shown in which the multiplicand is 12 and the multiplier is 5. Namely, in order to process the 2's complement codes, the inventive multiplier subtracts the output of the NAND gate 322 of the last stage, which is to process the MSB bit of the 2's complement code, from the output of the latch 320, the carry input terminal being provided with a given logic signal to allow the adder to perform the subtraction. In operation of the partial sum, in order to correct the overflow, the MSB of the partial sum is logically added to the MSB of the multiplicand which is provided through the

AND gates 304, 308, 315 according to the corresponding bits of the multiplier. Thus, the multiplication can be performed by the process of converting the MSB bit.

Although the above described example operations are enumerated by 2'S complement in a concrete way, other operations can be easily realized by those skilled in the art covering this field, if the NAND gate 322 of Figure 4 is turned into an AND gate as existing one and the carry Cin of the adder 323 is set at "0" when one is a 2'S complement code and the other is a straight code instead of two 2'S complement multiplication in the case where MSB is corrected in the digital signal processor.

Consequently, the above-described embodiment is effective in that it makes a correct calculation of 2'S complement code and achieves high-speed operation, making it possible to calculate 2'S complement, which is impossible in the known pipeline magnitude multiplier, by correcting the MSB of partial sum and the MSB of multiplying number.

While a preferred embodiment of the invention has been particularly shown and described above, it will be understood by those skilled in the art that modifications in detail may be made without departing from the scope of the invention.

CLAIMS:

1. A circuit arranged to multiply 2's complement codes, the codes comprising an n-bit multiplier and an m-bit multiplicand, the circuit comprising:

multiplication means for performing a multiplication by multiplying the m-bit multiplicand by each of the n bits of the multiplier;

a first stage of a partial sum formation means comprising: first adding means for adding together a first and second product, the first product being obtained by multiplying together the m-bit multiplicand and a first, least significant, bit of the multiplier, and the second product being obtained by multiplying together the m-bit multiplicand and a second bit of the multiplier; a first stage of a partial sum correcting means for receiving a most significant bit of the first product and a most significant bit of the second product and logically OR-ing them together; and a first corrected partial sum generating means for receiving an output from the first adding means and an output from the first stage of the partial sum correcting means, and generating a first corrected partial sum;

a second stage of the partial sum formation means comprising: second adding means for adding together the first corrected partial sum and a third product, the third product being obtained by multiplying together the m-bit multiplicand and a third bit of the multiplier; a second stage of a partial sum correcting means for receiving a most significant bit of the third product and a most significant bit of the first corrected partial sum and logically OR-ing them together; and a second corrected partial sum generating means for receiving an output from the second adding means and output from the second stage

of the partial sum correcting means and generating a second corrected partial sum;

where n is a number greater than 3, successive stages of the partial sum formation means are provided for the generation of third through to last corrected partial sums, each successive stage being substantially identical to the second stage; and

a 2's complement correcting means for subtracting the last corrected partial sum from a value produced by NANDing the most significant bit (MSB) of the multiplier and the m-bit multiplicand.

2. A circuit according to claim 1, wherein said 2's complement correcting means comprises;

means operably coupled to said multiplicand and said multiplier for selectively transferring said multiplicand on the basis of the MSB of said multiplier; and

means operably coupled to said transferring means, for subtracting the last corrected partial sum from the output of said transferring means.

3. A circuit substantially as herein described with reference to Figures 4 and 5.

4. A pipeline multiplier having a plurality of latch circuits and a plurality of adders, and including a circuit according to any of the preceding claims.

5. A digital signal processing system comprising a circuit according to any of claims 1 to 4.