Digital Optical Modified Signed-Digit Arithmetic
Based on Symbolic Substitution and Its Encoding Scheme

Ho-In Jeon and Mustafa A. G. Abushagur

Center for Applied Optics, and
Electrical and Computer Engineering Department
University of Alabama in Huntsville, Huntsville, AL 35899

Abstract

In this paper, we propose an encoding scheme suitable for digital optical modified signed-digit arithmetic processor based on symbolic substitution. The processor performs binary addition and subtraction in an extremely parallel manner, i.e., desired result is obtained in three steps of operations, independent of word and array size of the input operands. The encoding scheme proposed in this paper is designed to prevent the crosstalk problem.

1. Introduction

While electronic processing is sequential in nature and thus has ultimate fundamental speed limitation, optical systems have potential advantages such as high space-bandwidth and time-bandwidth products, inherent parallelism, nonplanar signal propagation, non-interfering and non-interacting properties in a linear media [1]. This makes it attractive to design parallel processors with optical systems. Optical systolic array processors and engagement processors for matrix-matrix and matrix-vector multiplication using acousto-optic devices are examples of these [2,3]. However, digital systems provide flexibility of operations and improved computation accuracy. Thus, it is desirable to combine the advantages of digital processing systems with those of optics. This is the basis for digital optical computing. Recently, there has been a lot of effort towards developing parallel processing architectures for digital optical computing (see [1] and references therein).

Symbolic substitution is one of the techniques which can be used to implement digital optical computing system by utilizing the parallelism of optics. Basically, symbolic substitution is a rule by which the patterns associated with input operands are replaced by new patterns associated with the results of the operation. The replaced patterns are combined and fed back into the input of the system and this process continues until the desired output is obtained. A system with pattern recognizers followed by pattern substituters and feedback can perform this type of operation.

A technique based on splitting an image, shifting the split image, superimposing the results, regenerating the superimposed image with an optical logic array, shifting the regenerated image, shifting the resulting images, and superimposing the shifted images is presented in References [4,5]. An optical implementation of symbolic substitution logic based on polarization control of the input data plane has been done in Reference [6]. Another implementation of digital optical processors based on symbolic substitution using holographic matched filters for pattern recognition, and space-invariant filters for pattern replacement has been presented in [7]. Yet another possibility is to use an optical digital truth-table look-up processors [8] for pattern recognizers.

The implementations of the digital optical arithmetic processors based on symbolic substitution described above rely essentially on the conventional ordinary binary number representation. The drawback of these systems is that the processing time for the case of binary addition and subtraction depends on the word size if not on the array size. It turns out that the modified signed-digit (MSD) ternary number representation provides better performance because of its carry propagation constrained only between adjacent digits due to the redundancy property of the number system.

In this paper, we propose an encoding scheme suitable for digital optical modified signed-digit arithmetic processor based on symbolic substitution. In section 2, binary arithmetic based on the modified signed-digit number representation is reviewed. In section 3, the modified signed-digit arithmetic operation based on symbolic substitution is presented and the associated symbolic substitution rules for these operations are derived. Section 4 describes optical implementations of the arithmetic operations and an encoding scheme for the symbols which represent the set of values \( \{1, 0, 1\} \). Finally, some applications and future work are discussed in the conclusion.
2. Binary Arithmetic with Modified 
Signed-Digit Number Representation 
The radix $r$ weighted number system is defined as

$$X = x_{n-1}r^{n-1} + x_{n-2}r^{n-2} + \cdots + x_1r + x_0,$$  \hspace{1cm} (1)

where $x_i \in \{0, 1, \cdots, r-1\}$, and $X$ is a non-negative integer number such that $0 \leq X \leq r^n - 1$. The number of digits required to express the dynamic range $0 \leq X \leq M - 1$ is given by

$$n = \lceil \log_r M \rceil,$$  \hspace{1cm} (2)

where $\lceil x \rceil$ denotes the smallest integer such that $\lceil x \rceil \geq x$.

When $r = 2$, the number system represents the conventional ordinary binary numbers with each digit taking value zero or one. There are tremendous number of algorithms to perform arithmetic operations based on this number system. The nature of this number system is sequential as far as the arithmetic operation is concerned. Therefore, it is extremely difficult to utilize the parallel processing technique to reduce the processing time. The symbolic substitution is one of the most powerful parallel processing algorithms which is good for big arrays of arithmetic operands with limited utilization of parallelism. When the radix $r$ is odd and $r = 2k + 1$, $k = 1, 2, \cdots, n$, each digit may take the value $x_i \in \{-k, -(k-1), \cdots, 0, \cdots, (k-1), k\}$, called the balanced or the symmetrical number representation. This sign symmetry makes the arithmetic operation powerful in that the sign conversion between positive and negative numbers is extremely difficult to utilize the parallel processing technique. When the radix $r$ is assumed to be zeros, and $s_i$ being the result of the binary addition operation. It is interesting to note that the result $s_i$ is an exact complement of $c_i$ resulting from the by-product of the operation. Therefore, $c_i$ can be used to check the validity of the answer. Example 1 illustrates the addition of two signed-digit integers $X = (-1)_{SD}$ and $Y = (3)_{SD}$ using the three steps shown above. It is shown that the result is a signed-digit number $S = (00010)_{SD} = (2)_{b}$ and that $C = S$. In this example, $\phi$ denotes a padded zero, needed to preserve the same input/output format for each stage of the signed digit addition.

**Signed-Digit Number Representation**

The radix $r$ weighted number system is defined as

$$X = x_{n-1}r^{n-1} + x_{n-2}r^{n-2} + \cdots + x_1r + x_0,$$  \hspace{1cm} (1)

where $x_i \in \{0, 1, \cdots, r-1\}$, and $X$ is a non-negative integer number such that $0 \leq X \leq r^n - 1$. The number of digits required to express the dynamic range $0 \leq X \leq M - 1$ is given by

$$n = \lceil \log_r M \rceil,$$  \hspace{1cm} (2)

where $\lceil x \rceil$ denotes the smallest integer such that $\lceil x \rceil \geq x$.

When $r = 2$, the number system represents the conventional ordinary binary numbers with each digit taking value zero or one. There are tremendous number of algorithms to perform arithmetic operations based on this number system. The nature of this number system is sequential as far as the arithmetic operation is concerned. Therefore, it is extremely difficult to utilize the parallel processing technique to reduce the processing time. The symbolic substitution is one of the most powerful parallel processing algorithms which is good for big arrays of arithmetic operands with limited utilization of parallelism. When the radix $r$ is odd and $r = 2k + 1$, $k = 1, 2, \cdots, n$, each digit may take the value $x_i \in \{-k, -(k-1), \cdots, 0, \cdots, (k-1), k\}$, called the balanced or the symmetrical number representation. This sign symmetry makes the arithmetic operation powerful in that the sign conversion between positive and negative numbers is easily performed by complementing each digit, various signed-digit arithmetic operations can be done without a special conversion technique, and rounding off corresponds to truncation.

The signed-digit number system proposed by Avizienis in 1961 [12] is one of the weighted number systems and is different from the balanced number system in that the radix $r$ is allowed to be either odd or even, and that the signed-digit number system is based on redundant representation, such that the carry propagation chain is limited to a small portion and highly parallel operation can be performed. It turns out that any number $X$ bounded as

$$-\frac{(k+1)(r^n-1)}{r-1} \leq X \leq \frac{(k+1)(r^n-1)}{r-1} \hspace{1cm} (3)$$

can be expressed by the appropriate choice of each digit. Moreover, the number $X$ is not unique in the signed-digit number system except when $X = 0$. For example, the decimal number 9 can be represented as the following 4-digit sequences in the radix-2 signed-digit number system:

$$\begin{align*}
(1000)_{SD} &= 1 \times 2^3 + 1 = (9)_{10} \\
(1011)_{SD} &= 1 \times 2^3 + 1 \times 2 + (1) = (9)_{10}
\end{align*}$$  \hspace{1cm} (4)

where $\overline{1}$ denotes $-1$.

A parallel addition of two signed-digit binary numbers $X = (x_{n-1} \cdots x_1 \cdots x_0)$ and $Y = (y_{n-1} \cdots y_1 \cdots y_0)$ can be performed by the following three steps for each digit regardless of the word size:

**Example 1:**

<table>
<thead>
<tr>
<th>$X$</th>
<th>$Y$</th>
<th>$S$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\overline{1}1111$</td>
<td>$0101$</td>
<td>$\overline{1}1100$</td>
</tr>
</tbody>
</table>

**Step 1:**

$$\begin{align*}
t_{i+1} &= \begin{cases} 
1 & \text{if } x_i + y_i \geq 1 \\
0 & \text{if } x_i + y_i = 0 \\
1 & \text{if } x_i + y_i \leq -1 
\end{cases} \\
w_i &= \begin{cases} 
1 & \text{if } x_i + y_i = 1 \\
0 & \text{if } x_i + y_i = 0, \text{ or } 2, \text{ or } -2 \\
1 & \text{if } x_i + y_i = -1 
\end{cases}
\end{align*}$$

<table>
<thead>
<tr>
<th>$w_i$</th>
<th>$w_i'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$</td>
<td>$0$</td>
</tr>
</tbody>
</table>

**Step 2:**

$$\begin{align*}
t_{i+1}' &= \begin{cases} 
1 & \text{if } t_i + w_i = 2 \\
0 & \text{if } t_i + w_i = 0, \text{ or } 2, \text{ or } -2 \\
1 & \text{if } t_i + w_i = -1 
\end{cases} \\
w_i' &= \begin{cases} 
1 & \text{if } t_i + w_i = 1 \\
0 & \text{if } t_i + w_i = 0, \text{ or } -2 \\
1 & \text{if } t_i + w_i = -1 
\end{cases}
\end{align*}$$

<table>
<thead>
<tr>
<th>$t_i'$</th>
<th>$w_i'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$</td>
<td>$0$</td>
</tr>
</tbody>
</table>

**Step 3:**

$$\begin{align*}
t_{i+1} &= \begin{cases} 
1 & \text{if } t_i' + w_i' \geq 1 \\
0 & \text{if } t_i' + w_i' = 0 \\
1 & \text{if } t_i' + w_i' \leq -1 
\end{cases} \\
w_i &= \begin{cases} 
1 & \text{if } t_i' + w_i' = 1 \\
0 & \text{if } t_i' + w_i' = 0, \text{ or } -2 \\
1 & \text{if } t_i' + w_i' = -1 
\end{cases}
\end{align*}$$

<table>
<thead>
<tr>
<th>$t_i'$</th>
<th>$w_i'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$</td>
<td>$0$</td>
</tr>
</tbody>
</table>

$S = (00010)_{SD} = (2)_{b}$ and that $C = S$. In this example, $\phi$ denotes a padded zero, needed to preserve the same input/output format for each stage of the signed digit addition.
As shown in Example 1, the carry propagation is always limited to one position to the left. This property of the signed-digit number system allows highly parallel operation, and the addition time is independent of the word size. Subtraction can be performed similarly by taking the complement the sign of a subtrahend and applying the same three steps to the subtraction operands.

3. Modified Signed-Digit Arithmetic with Symbolic Substitution

Symbolic substitution suggested by Huang at Bell Laboratory [4], as a means of utilizing the parallelism of optics, is a rule by which patterns associated with input operands are replaced by new patterns associated with the result of the operation. The replaced patterns are fed back into the input of the system and the process continues until the desired output is obtained. To perform these operations, the system may consist of pattern recognizers followed by pattern substituters and feedback. Fig. 1 shows the schematic diagram of a symbolic substitution system.

**Figure 1. Schematic Diagram of Symbolic Substitution.**

As shown in Fig. 1, the input data is split into P identical portions, and these copies of the input are fed into P different pattern recognizers, where P is the number of features that are embedded in the input data plane, and is determined according to the specific operation to be performed. Through these pattern recognizers and nonlinear operators, a particular pattern at each stage is recognized at all positions where the pattern to be detected occurs, and then replaced by new patterns via the pattern substituters. The replaced patterns are combined together to yield an intermediate or possibly final output. The part of this output is fed back into the input of the system and goes through the same operations until the desired output is obtained. Observe that the symbolic substitution described above can be implemented with either space-variant or space-invariant systems. Space-invariant systems, in some applications, provide higher performance in terms of the control complexity, the processing time, and the high space-bandwidth product requirement, while the applicable operations might be quite limited to some specific operations [13].

Due to the nature of parallelism, symbolic substitution allows binary addition operation independent of the array size when conventional ordinary binary number representation is used, while modified signed-digit number representation allows word size independent arithmetic operation. Therefore, it is not difficult to see that binary arithmetic operation with modified signed-digit number system based on symbolic substitution will exert a great advantage as far as parallelism is concerned. To perform the desired arithmetic operation with symbolic substitution, we need to find substitution rules for the operation. From the three steps for modified signed-digit addition, we can derive the substitution rules for this operation as shown in Table 1. Note that the procedure used in step 1 is exactly same as the one in step 3, and hence that the modified signed-digit substitution rule 3 for step 3 should be same as the MSD substitution rule 1.

**Table 1. Modified Signed-Digit Symbolic Substitution Rules.**

<table>
<thead>
<tr>
<th>MSD Substitution Rule 1 for Step 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 1 1 0 1 0 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>1 1 1 0 1 0 1 0 1 0 1 0 1 0 1 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MSD Substitution Rule 2 for Step 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
</tbody>
</table>

An arithmetic operation with a certain number of substitution rules requires same number of pattern recognizers, nonlinear operators, and pattern substituters. The optical system also suffers from optical power loss proportional to the number of substitution rules. It is therefore desirable to have as small number of substitution rules as possible. From Table 1, it is shown that MSD addition requires 27 pattern recognizers, nonlinear operators and pattern substituters. If, however, the input operands are conventional ordinary binary numbers, we don't need all of 9 rules in the MSD substitution rule 1; instead, we need only four. Therefore, the
27 substitution rules can be reduced to 22 even though the result is obtained in terms of MSD representation.

MSD subtraction can be performed by taking the complement of the subtrahend and adding it to the minuend, where 1 is a complement of 0, 0 is that of 1, and 0 is its own complement. This complementation of subtrahend can be achieved by using the substitution rule for the complementation as shown in Table 2. Therefore, if we want to perform MSD subtraction, we apply first the substitution rule for complementation and then those of MSD addition to the input operands.

Table 2. MSD Substitution Rule for Subtrahend Complementation

<table>
<thead>
<tr>
<th>MSD Substitution Rule for Complementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 1 1 0 0</td>
</tr>
<tr>
<td>0 0 0 0 1 1</td>
</tr>
<tr>
<td>1 1 1 1 0 0</td>
</tr>
<tr>
<td>1 1 0 0 1 1</td>
</tr>
<tr>
<td>1 1 1 1 1 1</td>
</tr>
</tbody>
</table>

4. Optical Implementation and an Encoding Scheme

An application of symbolic substitution to binary addition and subtraction based on conventional ordinary binary number system has been presented by Jeon and Abushagur et al. [7,9,10,11] using holographic matched and space-invariant filters. In the system, they addressed the crosstalk and suggested one way of solving the problem by either incorporating masks or introducing new encoding schemes. They also pointed out that the processing time depends on the word size but not on the array size when the binary arithmetic operation is incorporated with the conventional ordinary binary number system. To overcome this word size dependency, there have been several efforts to perform the arithmetic operation using highly parallel number system such as signed-digit number system.

Drake et al. [14] have proposed the use of holographic elements, prisms, and optical bistable devices to implement the signed-digit addition. This approach, however, is based on the parallel optical logic operation. Recently, Bocker et al. [15] have proposed an implementation of the SD addition and subtraction using optical symbolic substitution. They have first introduced substitution rules for MSD arithmetic operation. Ramamoorthy and Anthony [16] have suggested similar substitution rules for implementing the SD addition and subtraction using light polarization. Li and Eichmann [17] have introduced a variation of the substitution rules and a holographic content-addressable memory technique to implement the SD addition.

The modified signed-digit number system requires a set of three different digits \{1, 0, 1\} to represent its input operands. Lasher et al. [18] suggested three encoding schemes for a digital optical MSD arithmetic. These include three-position binary encoding using optical bistable technology, three-state polarization encoding, and three-state intensity encoding using optical tristable technology, even though these schemes are used to perform logical operations. In symbolic substitution, however, we need three different symbols which can prevent the crosstalk with minimum signal space. The combination of two (dark and bright) pixels to represent each logical value, of course, causes crosstalk which will necessarily require masks to be incorporated. Moreover, if the recognition is performed based on matched filtering, a correct recognition cannot be made due to the different optical power between the symbols.

One solution to this problem is to choose suitable symbols which can prevent the crosstalk. In designing those symbols, however, two conditions must be satisfied: (1) the symbols that represent logical values zero and one must have equal amounts of optical power to obtain correct recognition; (2) the spatial arrangement of the pixels must be same for a logical one as for a logical zero. Given these restrictions, we define new symbols which represent logical values \{1, 0, 1\} as shown in Fig. 2. It can be shown that the symbols shown in Fig. 2 does not cause the crosstalk to occur as long as we insert one row of dark isolation pixels in between vertically aligned pairs of operands' patterns.

![Figure 2. Symbols which represent logical values 1, 0, and 1.](image)

5. Conclusions

Due the nature of parallelism, symbolic substitution allows binary addition and subtraction operation independent of the array size when conventional ordinary binary number system is used, while modified signed-digit number system allows word size-independent arithmetic operation. Therefore, if we combine these two, we can harness higher degree of parallelism than one. The modified signed-digit addition and subtraction using optical symbolic substitution has been suggested by Bocker et al. [15]. In this paper, an encoding scheme suitable for digital optical modified signed-digit arithmetic processor based on symbolic substitution is proposed. The processor performs binary addition and subtraction in an extremely parallel manner, i.e., the processing time depends neither on the word size nor on the array size.
It is not too difficult to design a modified signed-digit digital optical multiplier based on symbolic substitution by using the concept of partial product generation and tree-like addition. The substitution rules for the binary multiplication will be available in the near future.

Acknowledgment

The authors would like to thank H. John Caulfield and R. Barry Johnson at the Center for Applied Optics for their encouragement and support. A special thanks should be directed to the SPIE Education Committee for their generous decision on supporting this research by awarding scholarships and grants to the first author of this paper, Ho-In Jeon.

References