Journal of the Optical Society of Korea, Vol. 9, No. 3, September 2005, pp. 111-118

Optical Look-ahead Carry Full-adder Using Dual-rail Coding

Sang Keun Gil*

Department of Electronic Engineering, The University of Suwon, Suwon P.O. Box 77, 445-743, KOREA

(Received February 14, 2005 : revised May 26, 2005)

In this paper, a new optical parallel binary arithmetic processor (OPBAP) capable of computing arbitrary n-bit look-ahead carry full-addition is proposed and implemented. The conventional Boolean algebra is considered to implement OPBAP by using two schemes of optical logic processor. One is space-variant optical logic gate processor (SVOLGP), the other is shadow-casting optical logic array processor (SCOLAP). SVOLGP can process logical AND and OR operations different in space simultaneously by using free-space interconnection logic filters, while SCOLAP can perform any possible 16 Boolean logic function by using spatial instruction-control filter. A dual-rail encoding method is adopted because the complement of an input is needed in arithmetic process. Experiment on OPBAP for an 8-bit look-ahead carry full addition is performed. The experimental results have shown that the proposed OPBAP has a capability of optical look-ahead carry full-addition with high computing speed regardless of the data length.

OCIS codes : 200.0200, 200.1130, 200.3760, 200.4560, 200.4660

I. INTRODUCTION

A digital optical computing is one of the interesting and promising new generation of computing systems [1,2]. To realize a parallel digital optical computing system, a variety of and a lots of problems must be taken into account. The fundamental problem solved most urgently is what kind of architecture or configuration is suitable for a digital optical computing system. Recently, All-optical information processing using semiconductor optical amplifier was proposed [3,4]. Digital optical computing advocates new and radically different computer architectures predicted upon the two- to multi-dimensionality and speed of optical systems. An another major attraction of using optical systems and technologies for digital computing is the ability to connect 2-D arrays of binary data achieving arbitrary connections among the elements of each array, processing the contents of the 2-D array in parallel and at high-speed.

As is well known, optical computing techniques have excellent features for large capacity information processing, such as massive parallelism, high-speed processing, interconnection capability free from crosstalk, and so on. Recent applications and advances in optical computing hold great promise in solving difficult problems for which electronics will be fundamentally limited in the future. Therefore, a number of investigations have been executed to develop a digital optical computer. Addition and subtraction are an elementary calculation in arithmetic computing, and optical implementation of an adder or a subtracter is necessary to develop an optical computer. Since binary addition, however, is an essential operation for any arithmetic unit, a variety of optical adders have been introduced. A carry look-ahead adder with SLM was suggested by Golshan and Bedi[5]. Merklein et al. [6] proposed the use of polarization encoding for optical shadow-casting. Jeon et al. [7] proposed also a digital optical full-adder based on symbolic substitution using holographic matched filter. Fukushima et al. [8] proposed an optoelectronics hybrid parallel carry look-ahead adder using content addressable memory. A single-stage optical adder/subtracter using modified signed digit (MSD) was proposed by Barua [9]. Sun and Weng[10] proposed recently that a butterfly free-space interconnection can be used as an n-bit parallel ripple carry full-adder. One-step digit-set-restricted modified signed-digit adder based on symbolic substitution using a shared content-addressable memory [11].

However, symbolic substitution method needs an intermediate process called a pattern matching, which is yet a difficult part and requires a nonlinear processing. The conventional shadow casting processor uses coded inputs. Since a decoding process is required at output. the difference between forms of input and output makes

it difficult to cascade these processors. To overcome these problems, therefore, a dual-rail encoding technique is adopted to design an optical parallel binary arithmetic processor (OPBAP) in this paper. This paper presents the design background of OPBAP. The OPBAP capable of performing look-ahead carry full addition is proposed and implemented in an optical way. The subtraction of two binary numbers may be accomplished by taking the complement of the subtrahend and adding it to the minuend. By this method, the subtraction operation becomes an addition operation requiring full-adders for its machine implementation. This processor is verified by experiment and simulation for a specific case of 8-bit addition.

Ⅱ. OPTICAL PARALLEL BINARY ARITHMETIC PROCESSOR

1. LOOK-AHEAD CARRY FULL-ADDER

An n-bit binary adder is, in principle, a two-level combinational logic circuit in which each of the n sum bits is expressed as a logical sum-of-products or product-of-sums of the input variables. A simple parallel adder can be formed by connecting n pieces of full-adders in the series or cascade arrangement. The schematic block diagram of this type parallel adder is shown in Fig. 1.

At the i-th stage, the i-th bits of operands x_i and y_i and a carry signal c_i from the proceeding full-adder stage are used to generate the i-th bit of s_i sum and the carry to the $(i+1)$st full-adder stage. The Boolean logical equations which describe the operation of the i-th full-addition stage are the same form as follows,

$$c_{i+1} = x_i \cdot y_i + (x_i + y_i) \cdot c_i \quad (1)$$

$$s_i = (x_i \oplus y_i) \oplus c_i \quad (2)$$

for $i = 0, 1, \cdots, n$ where x_i, y_i, and c_i are the inputs to the i-th full-adder stage s_i and c_{i+1} and are the sum and carry outputs, respectively. In Fig. 1, each full-adder stage supplies a carry bit to the next stage on its left side. A carry insertion to the input of a full-adder may cause it to generate a carry signal on its output of current full-adder stage, thus carry signals can propagate serially through the full-adder from right to left (or from LSB to MSB). So this type of full-adder is called a ripple carry full-adder.

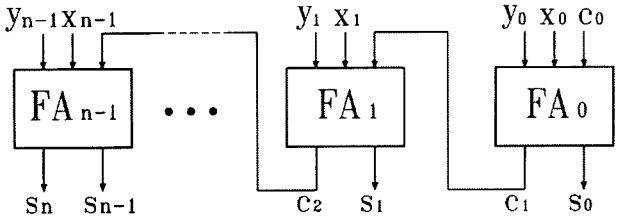


FIG. 1. Schematic diagram of a ripple carry full-adder.

The addition of two binary numbers in parallel implies that all the bits of the augend and the addend are available for computing process at the same time. As in any combinational logic circuit, the signal must propagate through the gates before the correct output is achieved in the output terminal. On this occasion the total propagation time is equal to the propagation time delay of a typical gate times the number of gate levels in the circuit. The longest propagation delay time in a parallel ripple carry full-adder is the time it takes the carry to propagate through the all full-adders. Since each bit of the sum output depends upon the value of the insertion carry, the value of the sum, s_i, in any given stage in the ripple carry full-adder will be in its steady-state final value only after the insertion carry to that stage has been propagated. The carry propagation time is a limiting factor on the speed with which two numbers of operand are added in parallel.

So as to reduce the carry propagation delay time, therefore, a carry look-ahead technique is very useful. The basic principle of look-ahead carry addition is described below. To understand the concept, refer to Eq. (1) and Eq. (2) which describe an operation of a ripple carry full-adder.

If two new binary variables are defined as

$$g_i = x_i \cdot y_i \quad (3)$$

$$p_i = x_i + y_i \quad (4)$$

for all digit positions i, the output sum and carry can be given by

$$c_{i+1} = g_i + p_i \cdot c_i \quad (5)$$

$$s_i = (x_i \oplus y_i) \oplus c_i \quad (6)$$

where g_i is called a carry-generate and it produces an output carry when both x_i and y_i are one, regardless of input carry. On the other hand, p_i is a carry-propagate because it is the term associated with the propagation of the carry from c_i stage to c_{i+1} stage.

Now, writing the Boolean functions for carry outputs of each stage while substituting for each c_i, from Eq.(1) through Eq. (6), yields

$$\begin{aligned}
c_1 &= g_0 + p_0 c_0 \\
c_2 &= g_1 + p_1 c_1 \\
&= g_1 + p_1 g_0 + p_1 p_0 c_0 \\
c_3 &= g_2 + p_2 c_2 \\
&= g_2 + p_2 g_1 + p_2 p_1 g_0 + p_2 p_1 c_0 \\
&\vdots \\
c_{i+1} &= g_i + p_i c_i \\
&= g_i + p_i g_{i-1} + p_i p_{i-1} g_{i-2} + \cdots + (p_i p_{i-1} \cdots p_1 p_0 c_0) \\
&\vdots
\end{aligned} \quad (7)$$

2. OPTICAL LOGIC GATE AND ARRAY PROCESSOR

In order to implement array system, two fundamental elements are required ; a logical gate and an interconnection element. An optical logic gate processing concept can be regarded as a free-space interconnected gate with logic. Therefore, free-space optical interconnections are important not only in communication systems but also in massive digital optical computing, and recently free-space optical interconnection is the focus of research because of its various advantages, such as its efficiency to implement logic functions and the complexity of the implementation system.

Space-variant optical logic gate processor (SVOLGP) is configured by Fig. 2, which shows the optical AND/OR operations. In this figure, M's represent mirrors and BS's beam splitters, respectively. The light is divided and then combined again in two Mach-Zender type interferometer schemes. It has three path-routes. One path is used for logic AND operation and the others for logic OR operation. In each path-route of the interferometer, two spatial interconnection logic filters (SILFs) are placed. The SILF is able to select a logic operation from AND or OR logic. That is, if the SILF is transparent aperture, the overall operation of this logic gate acts like the logical OR gate, while if the SILF is opaque aperture, the operation is equivalent to the AND logic. In other words, a pixel which blocks the light performs a logic AND operation.

The logic function on this operation can be represented as

$$F = x \cdot y + (S_1 \cdot x + S_2 \cdot y) \tag{8}$$

where S_1 and S_2 denotes SILF operation.

It can be realized that the SVOLGP uses spatial pixel apertures in SILF at spatial filtering planes to operate several optical logics. That is, optical logic gates can be implemented by using programmed data masks and spatial interconnection logic filters in the path of the route connecting the data mask to the other data mask. Two different AND/OR operations can be obtained in parallel applying two different filter functions of AND or OR logic in corresponding pixels of the same SILF. Therefore, each pixel in the SILF is capable of performing a logical AND or OR operation individually, i.e., one optical logic gate per pixel can be considered in this architecture. Two binary input data masks and two SIFLs can be considered to be replaced with real-time spatial light modulators (SLMs). The logic functions are, therefore, controlled by the on-off state of the illuminating light, which could be programmed into SLMs. Logic one and logic zero are achieved by passing a collimated light through a SLM in which those that correspond to the logic zeros are made opaque. Since the optical logic gate array is capable of performing more than one operation at a time, it may be considered as multiple instruction multiple data (MIMD) machine.

A dual-rail encoding method is probably a good breakthrough to solve the difficulties of input-output pattern match. In the type of dual-rail encoding, a bit of information is represented by the position of a bright spot rather than by its intensity or polarization. The advantages of the dual-rail encoding method is that no cell coding-decoding process in required and the output has the same format as the input. In the optical configuration of a logic-module, binary input variables are spatially dual-rail encoded. Consider input variables x and y divided into 1×4 pixel quadrants, which we call a logic-module, to apply a dual-rail encoding method. The coding principle of the dual-rail encoding is that two input variable x's are placed at the first and third rows and the two complements of x, i.e., $\bar{x}$, are placed at the second and fourth rows vertically. The other input variable is encoded in the similar manner, but two input variable y's are located at the first and second upper row and the two complements of y, i.e., $\bar{y}$, are located at the third and fourth lower row vertically. The two inputs x and y are segmented into four equal pixel quadrants, and each of the four quadrants is represented by a transparent or opaque coding characteristic. From Fig. 3 (a), the two spatially dual-rail encoded binary variable x and y are placed in cascade and an instruction-control unit, S's, is located behind the input stage. Finally in order to gather the four outputs into one position, a beam combining element is employed, but not displayed in the figure. The basic principle of the logic-module can be described in terms of the configuration of AND logic. In Fig. 3 (a), the spatial instruction-control filter (SICF) consists of four

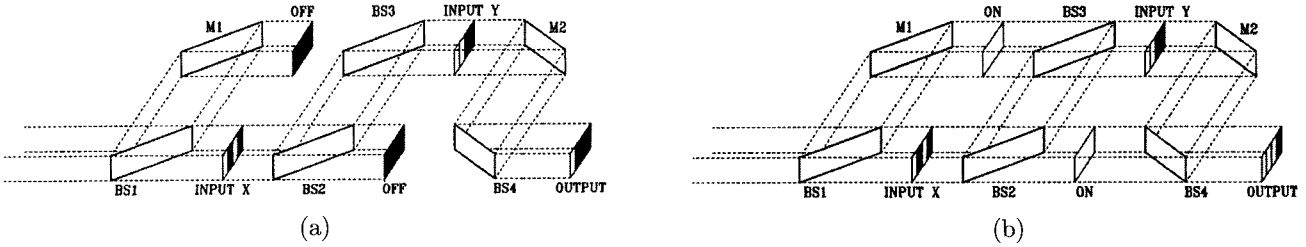


FIG. 2. Optical operations in SVOLGP (a) logic AND and (b) logic OR.

instruction-control signals written by S_1, S_2, S_3 and S_4. Two instruction-control states are available to be represented by S = 1(on state) and S = 0 (off state). The logic function on this configuration, which is the overlapped output intensity distribution, is now given by

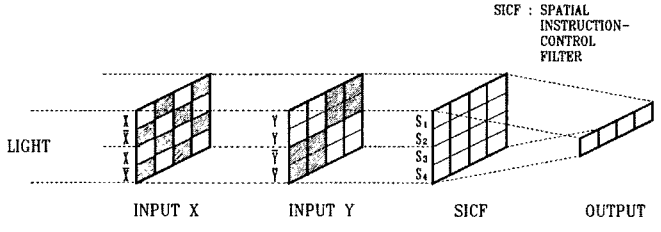
$$F = S_1(x \cdot y) + S_2(\bar{x} \cdot y) + S_3(x \cdot \bar{y}) + S_4(\bar{x} \cdot \bar{y}) \quad (9)$$

This Eq. (9) shows that the output results equivalently from the combination of four AND gates and an implied OR gate. Since the two dual-rail encoded binary variables placed in cascade only can generate the logical AND operation, four logic AND operation – xy, $\bar{x}y$, $x\bar{y}$, $\bar{x}\bar{y}$ – are obtained at the same time. The beam combining operation, which can be performed by a holographic element or a cylindrical lens, implies the logic OR operation. From Eq. (9), it is known that four output values of a required logic function are determined by the four instruction-control signals. For example, the logic XOR operation can be realized by setting $S_1 = S_4 = 0$ and $S_2 = S_3 = 1$. That is to say, the SICF performs a selective OR operation among xy, $\bar{x}y$, $x\bar{y}$, and $\bar{x}\bar{y}$ as an instruction. Therefore, all 16 Boolean logic functions can be performed for four pairs of binary input data by means of appropriate instruction selections. The logic-module using dual-rail encoding method essentially carries out all 16 Boolean logic functions based on the combination of logic NOT, AND, and OR functions, which is shown in Fig. 3 (b).

Since the distributed optical logic array can execute more than one operation at a time, it may be thought of as a multiple instruction multiple data (MIMD) machine. In this example, 4 instruction 4 multiple data were carried out by using 16 (44) logic AND gates and one logic OR gate. A real-time programmability can be obtained simply by changing the states of input variables and instruction-control signals. Some pixel area of SLM corresponds to an instruction-control path, and independently controllable pixel by pixel. Therefore, space-variant logic operation can also be executed by providing logical pixel areas with several different instructions.

3. SYSTEM ARCHITECTURE OF BINARY ARITHMETIC PROCESSOR

In designing an optical parallel binary arithmetic processor (OPBAP) capable of computing n-bit arithmetic, all of input data can be available simultaneously to the arithmetic processing unit that performs arithmetic operations in parallel. From this point of view the previously described logic array processors, i.e., space-variant optical logic gate processor (SVOLGP) and shadow-casting optical logic array processor (SCOLAP), are of use to execute a variety of logical operations different in space. In order to design a look-ahead carry full-adder, a look-ahead carry must be precomputed first. In this point of view, the proposed SVOLGP can be applicable to a carry-generating system. Because a look-ahead carry can be produced by the above mentioned SVOLGP, a sum is calculated by using the look-ahead carry. To perform an operation for sum, the previously proposed SCOLAP is promising and useful to implement these operations. The SCOLAP is conceptually analogous to an electronic logic array. If the input is replaced with a look-ahead carry and the combination of sum-of-product terms in controlled, a required sum can be obtained. To understand the procedure of summation, a configuration shown in Fig. 3 (a) is taken into account. From the figure, an expression of Eq. (9), if instruction-control signals(S_1, S_2, S_3, and S_4) are predetermined, that is, are replaced with look-ahead carry signals, a very interesting circumstance is observed.



(a)

SICF				X : □■□■ Y : □□■■	INPUTS
S1	S2	S3	S4	Fn.	16 BOOLEAN LOGIC FUNCTIONS	OUTPUT
off	off	off	off	F0	F = No operation	■■■■
on	off	off	off	F1	AND = XY	□■■■
off	on	off	off	F2	$X\bar{Y}$ = $X\bar{Y}$	■■□■
on	on	off	off	F3	X = XY + $X\bar{Y}$	□■□■
off	off	on	off	F4	$\bar{X}Y$ = $\bar{X}Y$	■□■■
on	off	on	off	F5	Y = XY + $\bar{X}Y$	□□■■
off	on	on	off	F6	XOR = $X\bar{Y}$ + $\bar{X}Y$	■□□■
on	on	on	off	F7	OR = XY + $X\bar{Y}$ + $\bar{X}Y$	□□□■
off	off	off	on	F8	NOR = $\bar{X}\bar{Y}$	■■■□
on	off	off	on	F9	XNOR = XY + $\bar{X}\bar{Y}$	□■■□
off	on	off	on	F10	$\bar{Y}$ = $X\bar{Y}$ +$\bar{X}\bar{Y}$	■■□□
on	on	off	on	F11	X+$\bar{Y}$ = XY + $X\bar{Y}$ + $\bar{X}\bar{Y}$	□■□□
off	off	on	on	F12	$\bar{X}$ = $\bar{X}Y$ + $\bar{X}\bar{Y}$	■□■□
on	off	on	on	F13	$\bar{X}$+Y = XY + $\bar{X}Y$ + $\bar{X}\bar{Y}$	□□■□
off	on	on	on	F14	NAND = $X\bar{Y}$ + $\bar{X}Y$ + $\bar{X}\bar{Y}$	■□□□
on	on	on	on	F15	T = XY + $X\bar{Y}$ + $\bar{X}Y$ + $\bar{X}\bar{Y}$	□□□□

(b)

FIG. 3. (a) Optical configuration of SCOLAP and (b) 16 Boolean logic functions obtained from SCOLAP.

$$s_1 = s_4 = c$$
$$s_2 = s_3 = \bar{c}$$

Let, then Eq.(9) can be rewritten as

$$\begin{aligned} F &= c \cdot (x \cdot y) + \bar{c} \cdot (\bar{x} \cdot y) + \bar{c} \cdot (x \cdot \bar{y}) + c(\bar{x} \cdot \bar{y}) \\ &= (\bar{x} \cdot y + x \cdot \bar{y}) \cdot \bar{c} + (x \cdot y + \bar{x} \cdot \bar{y}) \cdot c \\ &= (x \oplus y) \cdot \bar{c} + (\overline{x \oplus y}) \cdot c \\ &= (x \oplus y) \oplus c \end{aligned} \tag{10}$$

This expression is equal to Eq. (2) indicating a Boolean expression for sum. Therefore, a sum-generating system can be implemented by using the proposed SCOLAP.

A schematic architecture of OPBAP is proposed as shown in Fig. 4. In Fig. 4, the pattern splitter is used for a pattern feedback and the pattern shifter is used for a pattern shift and feedback to inputs. The pattern combiner produces dual-patterns supposed to be driven into sum-generating processor. The reason why dual-patterns are needed is that in the sum-generating processor two look-ahead carries are required. Therefore, the proposed system can perform optical parallel n-bit full addition by generating look-ahead carry and sum digits.

From this schematic diagram of OPBAP, a parallel binary arithmetic processing system can be designed with optical components such as mirrors, beam splitter, lenses, and spatial light modulators (SLMs). Fig. 5 shows an optical configuration of the proposed n-bit OPBAP. The system is composed of two layers ; one is for parallel look-ahead carry generation of all data bits, which is called the CGP (Carry-Generating Processor). The other is for parallel sum generation, called the SGP (Sum-Generating Processor). In the CGP, the proposed SVOLGP is operated as basic architecture. Two input operands of X and Y are displayed on SLMs with dual-rail encoding and spatial interconnection logic filters SILF1 and SILF2 are placed in the path of light to control logical operations. Mirrors, M3-M6, are used for an output pattern feedback. BS5 and M7 produce dual-output patterns for carry, i.e., two non-inverted carries and two inverted carries, and interconnect the CGP to SGP with M8-M10. The next processor, SGP, is made from the proposed SCOLAP where two input variables X and Y are also dual-rail encoded on SLMs. It is convenient and economical that we use a SLM to realize input of both CGP and SGP. Lastly the resultant output from SGP is combined by a cylindrical lens which is analogous to logical OR operation, and is detected as the final result.

In order to investigate operating principles of look-ahead carry full-adder, the schematic configuration of OPBAP shown in Fig. 6 is considered. With this configuration

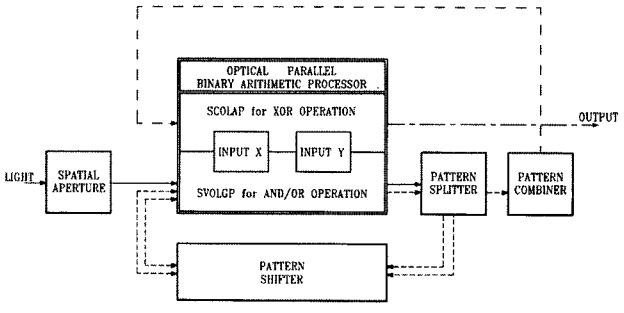


FIG. 4. Schemetic architecture of optical parallel binary arithmetic processor (OPBAP).

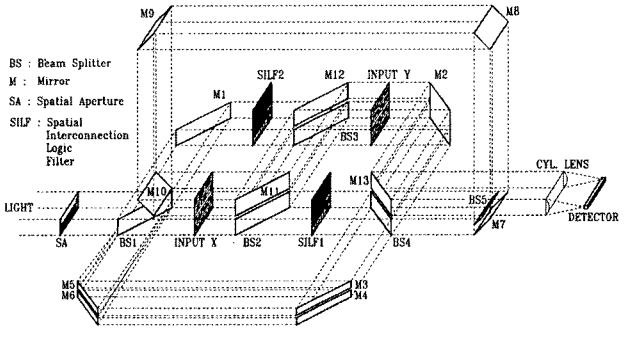
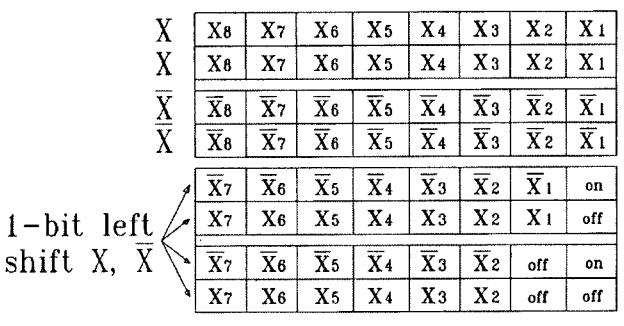
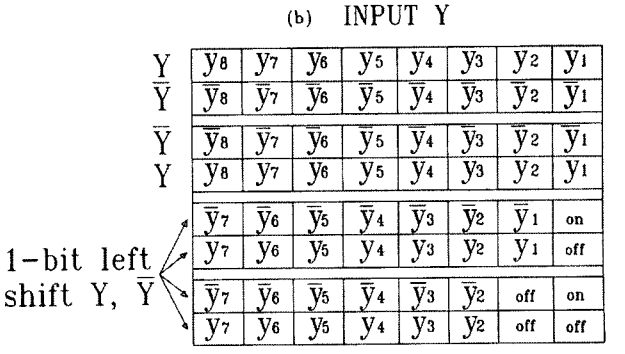


FIG. 5. Optical configuration of OPBAP.



(a)



(b)

FIG. 6. Dual-rail encoded operand expressions for (a) addend X, and (b) augend Y.

an 8-bit binary full addition is proposed. On this occasion each carry and sum bit for look-ahead carry full-adder is expressed as Eqs. (5) through (7). First input operands of addend X and augend Y are displayed on SLMs with dual-rail encoding rule. Fig. 6 shows the dual-rail encoded patterns for addend X and augend Y, respectively.

As shown in Fig. 6, an encoded input has two parts. The lower part is used for generating look-ahead carries, while the upper part for producing the final sums. The lower part has again four data arrays. A non-inverted data array with one-bit left shifted is placed at the second row, while an inverted data array is at the first row. Similarly one-bit shifted data arrays except x_1, $\overline{x_1}$, y_1, $\overline{y_1}$, are placed at the fourth and third row. And the on or off encoding means light-transparent or light-opaque, respectively, which is supposed to be used later for producing the first-bit sum. The upper part has four data arrays with dual-rail encoding in the same way. There is no bit-shift. Notices that addend X is arranged as $xx\bar{x}\,\bar{x}$ vertically, while augend Y arranged as $y\bar{y}\,\bar{y}y$. Second, an input spatial aperture (SA) and two SILFs are constructed, as shown in Fig. 7. SA is designed to transmit input light into some specific pixels of the data operands. In the pattern of SILF, transparent squares are operated as logical OR filter functions and opaque squares as logical AND filter functions.

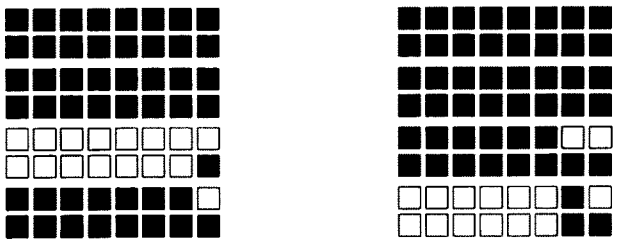


Fig. 7. Patterns for (a) input spatial apertures (SA), (b) spatial-interconnection logic filter (SILF).

The overall processing stages of look-ahead carry full addition are as follows:

[STEP 1] Assign input operands

$$X = [x_{n-1}x_{n-2}\cdots x_1x_0]$$
$$\overline{X} = [\overline{x_{n-1}}\,\overline{x_{n-2}}\cdots\overline{x_1}\overline{x_0}]$$
$$Y = [y_{n-1}y_{n-2}\cdots y_1y_0]$$
$$\overline{Y} = [\overline{y_{n-1}}\,\overline{y_{n-2}}\cdots\overline{y_1}\overline{y_0}]$$

where x_i, $\overline{x_i}$, y_i, $\overline{y_i}$ for $i = 0, 1, 2,\cdots, n\text{-}1$

[STEP 2] Interconnect input patterns bit-by-bit to operate *AND/OR* logic resulting in carry-generate

$$G^p = [g^p_n g^p_{n-1}\cdots g^p_2 g^p_1]$$
$$G^n = [g^n_n g^n_{n-1}\cdots g^n_2 g^n_1]$$

where $g^p_i = x_{i-1}\cdot y_{i-1}$ for $i = 0, 1, 2,\cdots, n$

$g^p_i = \overline{x_{i-1}} + \overline{y_{i-1}}$ for $i = 1$

$\overline{x_{i-1}}\cdot\overline{y_{i-1}}$ for $i = 2, 3,\cdots, n$

[STEP 3] Shift carry-generate output patterns to the left by one-bit and downward two-bits, and feedback to input operands resulting in carry-propagate

$$P^p_1 = [g^p_{(1),n} g^p_{(1),n-1}\cdots g^p_{(1),2} g^p_{(1),1}]$$
$$P^n_1 = [g^n_{(1),n} g^n_{(1),n-1}\cdots g^n_{(1),2} g^n_{(1),1}]$$

where

$P^p_{(1),i} = (x_{i-1} + y_{i-1})\cdot g^p_{i-1}$ for $i = 2, 3,\cdots, n$

$P^n_{(1),i} = (\overline{x_{i-1}} + \overline{y_{i-1}})\cdot g^n_{i-1}$ for $i = 2, 3,\cdots, n$

[STEP 4] Shift carry-propagate output patterns to the left by one-bit and feedback to input operands again, and iterate until steady-state

for $j = 2$ to n

$$P^p_j = [g^p_{(j),n} g^p_{(j),n-1}\cdots g^p_{(j),2} g^p_{(j),1}]$$
$$P^n_j = [g^n_{(j),n} g^n_{(j),n-1}\cdots g^n_{(j),2} g^n_{(j),1}]$$

where

$$P^p_{(1),i} = P^p_{(i-1),i} + (x_{i-1} + y_{i-1})\cdot P^p_{(j-1),i-1} \quad \text{for } i = 1, 2,\cdots, n$$
$$P^n_{(1),i} = P^n_{(i-1)i,} + (x_{i-1} + y_{i-1})\cdot P^n_{(j-1),i-1} \quad \text{for } i = 1, 2,\cdots, n$$

[STEP 5] Combine all the half-carries into look-ahead full-carry

$$c^{jp} = G^p + P^p_{(n)}$$
$$c^{jn} = G^n + P^n_{(n)}$$

[STEP 6] Make a pair of look-ahead carry and feedback to input operands of shadow-casting logic array processor

$$C = \begin{bmatrix} C^p \\ C^n \\ C^p \\ C^n \end{bmatrix} = \begin{bmatrix} C^{fp}|1 \\ C^{fn}|0 \\ C^{fp}|1 \\ C^{fn}|0 \end{bmatrix}$$

[STEP 7] Interconnect carry pattern bit-by-bit to

operate AND logic resulting in half-sums

$$S_1 = C^p \cdot X \cdot Y$$
$$S_2 = C^n \cdot X \cdot \overline{Y}$$
$$S_3 = C^p \cdot \overline{X} \cdot Y$$
$$S_4 = C^n \cdot \overline{X} \cdot Y$$

[STEP 8] Combine all the half-sums into final-sum output

$$S = \sum_{k=1}^{4} S_k$$

where

$$S_k = [s_{k,n} s_{k,n-1} s_{k,1} s_{k,0}] \text{ for } k = 1, 2, 3, 4$$

V. EXPERIMENT AND RESULTS

In order to compute a look-ahead carry full addition, three sequences of events must be followed: (1) X, Y input operandsare applied. (2) Look-ahead carry generator computes g and carry-propagate p, and then computes all carry outputs. (3) Each parallel adder computes the sum outputs. These sequences are an equivalent expression of Eqs. (5) and (6). But since output sum is given by $\overline{c_i}$ as well as c_i, it is necessary to calculate also the dual value $\overline{c_i}$ of the carry. To be able to compute the dual value $\overline{c_i}$ of the carry in full addition, dual-rail encoded input operands are used. An optical parallel look-ahead carry full adder was investigated and verified experimently for the case of 8-bit full addition as a specific example : (1) The input operands are two 8 bit binary numbers. (2) The output is a full precision 8-bit binary sum. (3) The MSB of input operands is set 0 intentionally. The reason is the we can observe carry propagation at final bit stage. For the case of addition the final carry shown in MSB represents sum digit. Fig. 8 shows dual-rail encoded input patterns for addition; X=[01001001] and Y=[01101011] which are 73 and 107 in decimal, respectively.

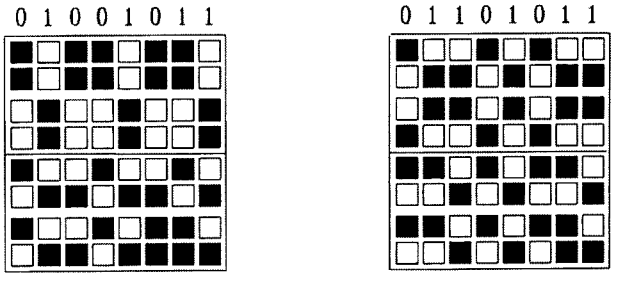


FIG. 8. Dual-rail encoded input patterns : (a) addend X=[01001001], (b) augend Y=[01101011]

In this experiment, He-Ne laser is used as a light source and pattern masks are used as input operands and spatial-interconnection logic filter. A pixel size of 2 mm × 2 mm square with 0.5 mm spaces between pixels was determined by considering diffraction effects. Selected output bits at final stage are combined by using a cylindrical lens in a single bit intensity that indicates the resultant value of the full addition.

The experimental results of the 8-bit binary full addition using look-ahead carry method are shown in Fig. 9. From Fig. 9 (f), the experiment shows that the final value intensity for sum is shown an S = [10110100] which is equivalent to the decimal number 180. This result concords with 73+107=180 because binary number X is 73 in decimal and Y is 107.

VI. CONCLUSION

In this paper, a new optical parallel binary arithmetic processor (OPBAP) is proposed and implemented. The implemented OPBAP consists of two parallel optical logic processors. The first one is a space-variant optical logic gate processor (SVOLGP) and the other one is a shadow-casting optical array processor (SCOLAP). SVOLGP can process logical AND and OR operations,

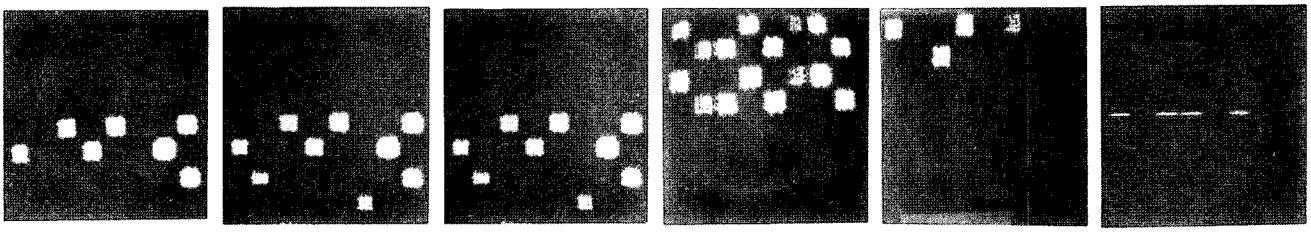


FIG. 9. Photographs of optical experiment output patterns for (a) STEP2 (b) STEP3 (c) STEP 4 and (d) STEP 6, (e) STEP 7 and (f) STEP 8, in look-ahead full addition.

while SCOLAP can execute the 16 Boolean logic functions. Both processors use linear optical components such as mirrors, beam splitters, and lenses for spatial interconnections. For these two processors to perform various logics in parallel, two types of spatial mask which control logic instruction signals are realized. One is a spatial-interconnection logic filter (SILF) used in SVOLGP and the other is a spatial instruction-control filter (SICF) used in SCOLAP. By using these two processors, an optical parallel binary arithmetic processor (OPBAP) to compute binary full addition is implemented. SVOLGP is used for constructing an architecture of a look-ahead carry-generating processor (CGP) and SCOLAP for realizing a sum-generating processor (SGP). In representing an input format, a dual-rall encoding technique is adopted because the complement of the input is required in realizing Boolean functions.

This paper shows that experiment on binary full addition are executed for a specific example ; 8-bit look-ahead carry full addition. In this experiment, it was shown that as for addition of input X = [01001001], 73 in decimal and Y = [01101011], 107 in decimal, the proposed OPBAP produced sum output S = [10110100], 180 in decimal. The experimental results have shown that the proposed OPBAP is able to process parallel arithmetic with high computing speed regardless of the data length.

*Corresponding author : skgil@suwon.ac.kr

ACKNOWLEDGMENTS

This research was partially supported by grant No. R01-2003-000-10528-0 (2005) from KOSEF.

REFERENCES

[1] A. A. Sawchuk and T. C. Strand, "Digital optical computing," Proc. IEEE, vol. 72, pp. 758-779, 1984.
[2] A. Huang, "Architectural considerations involved in the design of an optical digital computer," Proc. IEEE, vol. 72, pp. 780-786, 1984.
[3] J. H. Kim, B. K. Kang, Y. H. Park, Y. T. Byun, S. Lee, D. H. Woo, and S. H. Kim, "All-optical AND gate using XPM wavelength converter," Journal of Optical Society of Korea, vol. 6, no. 4, pp. 25-28, Mar. 2001.
[4] S. Lee, J. H. Kim, Y. I. Kim, Y. T. Byun, Y. M. Jhon, D. H. Woo, and S. H. Kim, "Various functionalities based on semiconductor optical amplifier for all-optical information processing," Journal of Optical Society of Korea, vol. 6, no. 4, pp. 165-171, Dec. 2002.
[5] R. Golshan and J. S. Bedi, "Implementation of carry look-ahead adder with spatial light modulators," Optics Comm., vol. 68, pp. 175-178, 1988.
[6] T. M. Merklein, W. Stork, and H. Yajima,, "Optical full adder," Appl. Opt., vol. 28, pp. 4313-4315, 1989.
[7] H. Jeon, M. Abushagur, A. A. Sawchuk, and B. K. Jenkins, "Digital optical processor based on symbolic substition using holographic matched filtering," Appl. Opt., vol. 29, pp. 2113-2125, 1990.
[8] A. Kostrzewski, D. H. Kim, Y. Li, and G. Eichmann, "Fast hybrid parallel carry look-ahead adder," Optics Lett., vol. 15, pp. 915-917, 1990.
[9] S. Barua, "Single-stage optical adder/subtractor," Opt. Eng., vol. 30, pp. 265-270, 1991.
[10] D. G. Sun and Z. H. Weng, "Butterfly interconnection implementation for an n-bit parallel ripple carry full adder," Appl. Opt., vol. 30, pp. 1781-1785, 1991.
[11] F. Quian, G. Li and M. S. Alam, "One-step digit-set-restricted modified signed-digit adder using an incoherent correlator based on a shared content-addressable memory," Opt. Eng., vol. 41, no. 10, pp. 2607-2612, 2002.