논문 2004-41SD-6-6

Triple rail-coding 입력과 기호치환을 이용한 변형부호화자리수 가산기 구현

(Implementation of the modified-signed digit(MSD) number adder using triple rail-coding input and symbolic substitution)

신 창 목*, 김 수 중*, 서 동 환**

(Chang Mok Shin, Soo Joong Kim, and Dong Hoan Seo)

요 약

본 논문에서는 입력패턴을 triple rail-coding 방식으로 표현한 후 입력의 직렬연결 방법으로 기호치환을 수행하는 광 병렬 변형부호화자리수 가산 시스템을 제안하였다. Triple rail-coding 방식으로 변형부호화자리수 입력을 표현할 때 중복연산 결과가 나오는 입력들은 동일한 패턴으로 전처리하여 기호치환과정의 규칙수를 줄였고, 광 구현시 공간 이동된 입력패턴을 직렬로 연결하여 광을 통과시킴으로써 공간 이동 연산, NOR 연산, 그리고 문턱치 연산과정이 필요 없는 광 가산기를 구현하였다.

Abstract

An optical parallel modified signed-digit(MSD) number adder system is proposed by using triple rail-coding input patterns and serial arrangement method of symbolic substitution. By combing overlapped arithmetic results, which are produced by encoding MSD input as triple rail-coding patterns, into the same patterns, symbolic substitution rules are reduced and also by using serialized and space-shifted input patterns in optical experiments, the optical adder without space-shifting operation, NOR operation and threshold operation is implemented.

Keywords: 기호치환(symbolic substitution), 광 가산기(optical adder), 직렬연결(serial arrangement), 변형부호화자리수(modified signed-digit number)

I. 서 론

광 컴퓨터 구현에 필요한 구성요소에는 입출력 장치, 처리 장치, 저장 장치 등이 있다. 이 중 처리 장치에서 가감산을 수행하는 광 병렬 가산기는 광 컴퓨터의 가장 기본적이고 중요한 요소 중의 하나이다. 이러한 광 병렬 가산기는 광의 병렬성과 고속성을 충분히 활용할 수 있도록 구현되어야 하며 이를 위해 적합한 수 체계(number systerm)의 도입과 입력 데이터의 효과적인 부호화 방법이 필수적이다.

광 병렬 가산기 구현에 사용되는 수 체계로는 이진 체계와 그 외 유수(residue number)나 변형부호화자리수(modified signed-digit, MSD)체계가 있다[1-3]. 이진 체계를 사용한 광가산기는 현재 비트 가산시 발생하는 출력 올림수가 상위 비트에 가산에 영향을 주는 구조로 되어있다. 이러한 자리수 사이의 의존성으로 인해 올림수 지연시간이 발생하고 그 결과 모든 자리수에 대한 병렬가산이 불가능하다. 유수를 사용한 유수 시스템은 여러 개의 상대소수계수(relative prime modulus)로 이루어져 있으며, 유수 연산으로 출력된 자리수가 각각

* 정회원, 경북대학교 전자전기컴퓨터공학부
(Optical Signal Processing Lab, School of Electrical Engineering and Computer Science, Kyungpook Nat'l University)
** 정회원, 한국해양대학교 전자전기공학부
(Division of Electrical and Electronics Engineering, Korea Maritime University)
접수일자: 2002년1월25일, 수정완료일: 2004년5월17일

독립적이므로 올림수를 발생시키지 않는다. 그러나 큰 수를 가산과 승산할 때 더욱 많은 소수계수 논리요소들이 필요하므로 실질적인 광학적 구현이 어렵다.

변형부호화자리수를 사용하면 입력이 잉여 이진수(1, 0, $\bar{1}$)로 표현되므로 자리수에 관계없이 병렬로 가감산을 수행할 수 있는 장점을 가지고 있다. 변형부호화자리수를 이용한 광병렬 가산기는 Huang과 Brenner 등이 제안한 기호치환 방법으로 구현할 있다[4-5]. 기호 치환 방법으로 구현한 변형부호화자리수 가산기는 가감산을 완전히 병렬로 수행할 수 있고 자리수에 관계없이 합을 구할 수 있음이 이미 제시되었다.

기존의 기호치환을 이용한 광 병렬 변형부호화자리수 가산기의 구현 과정은 인식 단계와 치환 단계가 있고 각 단계에 필요한 연산 과정은 공간 이동 연산, 중첩 연산, NOR 연산, 문턱치 연산 등의 많은 연산 과정이 필요하다. 변형부호화자리수를 입력으로 부호화할 때 서로 다른 3가지 형태로 부호화하여야 하므로 가산기 구현에 있어서 인식과 치환에 많은 규칙수가 필요하여 시스템이 커진다[6].

본 논문에서는 변형부호화자리수를 사용하여 입력을 나타내었고 3가지 형태의 부호화된 입력을 나타내기 위해 triple rail-coding 방식으로 부호화 한 후, 부호화된 입력을 직렬로 나열하여 광을 통과시킴으로써 가산결과를 얻는 기호치환 광 병렬 가산 시스템을 제안하였다. 기존의 기호치환 구현시 중복된 연산결과가 나오는 입력을 triple rail-coding 방식을 이용하여 동일하게 전처리하므로써, 전체 인식과 치환 과정에서의 규칙수를 줄였고 규칙수를 줄임으로써 전체 시스템의 크기를 줄일 수 있었다. 또한, 공간 이동된 입력 패턴을 직렬로 연결한 후 광을 통과시켜 인식과 치환과정을 수행하는 새로운 기호치환 방법을 사용함으로써 공간 이동 연산, 중첩 연산, NOR 문턱치 연산과정을 생략하여 광 병렬 가산 시스템의 크기를 줄일 수 있었고, 이를 광 실험을 통해 직접 확인하였다.

II. 두 단계 변형부호화자리수 가산기

1. 변형부호화자리수 가산기

변형부호화자리수에서 정수 A는

$$A = \sum_i a_i 2^i \quad (1)$$

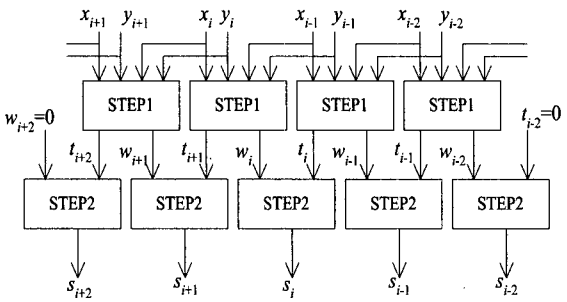


그림 1. 두 단계 변형부호화자리수 가산기의 구성도
Fig. 1. Block diagram for a two-stage MSD adder.

로 표현하고, a_i는 {1, 0, $\bar{1}$} 값을 나타낸다. $\bar{1}$는 -1을 의미한다.

변형부호화 자리수를 사용한 변형부호화 자리수 가산기는 최종연산결과를 구하기 위해 수행하는 단계의 수에 따라 세 단계, 두 단계, 한 단계 변형부호화 자리수 가산기가 있다[3],[8-9]. 세 단계를 적용할 경우 입력의 궤환으로 인해 처리속도가 늦어지며, 한 단계인 경우 많은 기호치환 규칙수로 인해 시스템 크기가 커진다는 문제점이 있다. 두 단계 가산기를 이용하면 위의 문제점을 보완할 수 있다.

2. 두 단계 변형부호화자리수 가산기

기호치환 규칙을 적용한 두 단계 변형부호화자리수 가산기의 구성도는 그림 1과 같다.

그림 1에서 기호치환 규칙을 이용한 첫 번째 단계에서의 중간 올림수와 중간 합인 t_{i+1}과 w_i는 현재 비트(x_i , y_i)과 바로 하위 비트(x_{i-1}, y_{i-1})의 값에 의해서 가산된다. 두 단계 변형부호화자리수 연산의 진리표는 표 1에 나타내었다.

표 1(a)는 첫 번째 단계, 1(b)는 두 번째 단계 진리표이다. 여기에서 'any possible combination'은 하위 비트(x_{i-1} , y_{i-1})가 두 변형부호화자리수의 조합으로 가능한 모든 9가지 경우를, 'both positive'는 하위 비트가 (0 , 1), (1 , 0), (1 , 1)인 3가지 경우, 'otherwise'는 하위 비트가 그 밖의 (0 , 0), (1 , $\bar{1}$), ($\bar{1}$, 1), (0 , $\bar{1}$), ($\bar{1}$, 0), ($\bar{1}$, $\bar{1}$)인 6가지의 경우를 표현한다. 표 1(a)에서 필요한 기호치환 규칙 수는 현재 비트 (x_i , y_i)가 (1 , 1), (0 , 0), (1 , $\bar{1}$), ($\bar{1}$, 1), ($\bar{1}$, $\bar{1}$)인 경우 하위 비트와 상관없이 기호치환되므로 5가지의 기호치환 규칙이, 현재 비트 (x_i , y_i)가 (0 , 1), (1 , 0), (0 , $\bar{1}$), ($\bar{1}$, 0)인 경우 하위 비트가 (0 , 1), (1 , 0), (1 , 1)인 'both

표 1. 변형부호화자리수 가산을 위한 (a) 첫번째 단계, (b) 두 번째 단계의 기호치환 규칙 진리표
Table 1. Truth table of symbolic substitution rules for (a) the first step, (b) the second step for MSD addition.

Input digits x_i y_i	Input digits at the Next lower order position x_{i-1} y_{i-1}	t_{i+1} w_i
1 1	any possible combination	1 0
1 0	both positive	1 $\bar{1}$
1 0	otherwise	0 1
0 1	both positive	1 $\bar{1}$
0 1	otherwise	0 1
0 0	any possible combination	0 0
0 $\bar{1}$	both positive	0 $\bar{1}$
0 $\bar{1}$	otherwise	$\bar{1}$ 1
$\bar{1}$ 0	both positive	0 $\bar{1}$
$\bar{1}$ 0	otherwise	$\bar{1}$ 1
$\bar{1}$ 1	any possible combination	0 0
1 $\bar{1}$	any possible combination	0 0
$\bar{1}$ $\bar{1}$	any possible combination	$\bar{1}$ 0

(a)

t_i	w_i	s_i
1	0	1
0	1	1
1	$\bar{1}$	0
0	0	0
$\bar{1}$	1	0
$\bar{1}$	0	$\bar{1}$
0	$\bar{1}$	$\bar{1}$

(b)

positive' 일 때는 4 × 3 = 12 가지, 그 외의 하위비트가 'otherwise' 인 경우 4 × 6 = 24 가지이다. 따라서 첫 번째 단계에서는 총 41가지의 기호치환 규칙이 필요하다. 표 1(b)에서 필요한 기호치환 규칙 수는 두 번째 단계의 현재 비트 (t_i , w_i)의 연산이므로 7가지가 된다.

이 기호치환을 병렬로 적용하는 과정인 그림 5에서는 첫 번째 단계 기호치환의 입력패턴 (x_0 , y_0)는 (t_1 , w_0)로, (x_1 , y_1)는 (t_2 , w_1)로, (x_2 , y_2)는 (t_3 , w_2)로, (x_3 , y_3)는 (t_4 , w_3)로 각각 치환되고 동일한 방법으로 두 번째 단계 기호치환됨을 나타내었다.

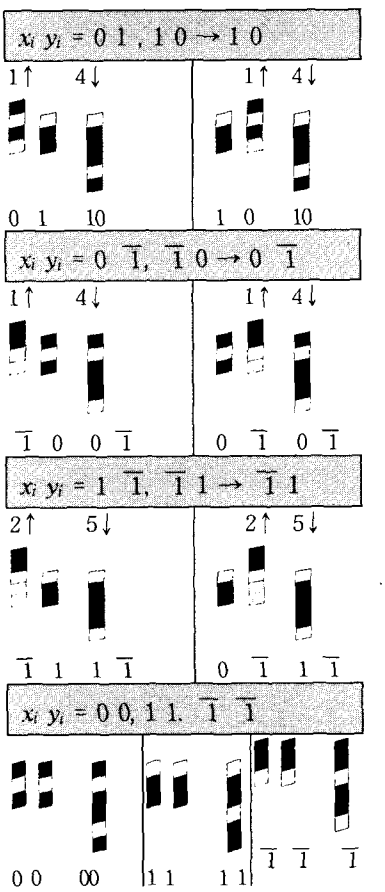
III. 제안한 두 단계 변형부호화자리수 가산기

1. 입력의 부호화

입력을 부호화하는 방법에는 빛의 편광각도를 다르게 하여 부호화하는 편광부호화 방법과 화소의 밝고 어둠을 이용하여 부호화하는 세기부호화 방법이 있다. 세기 부호화 방법 중에서 세 개의 화소를 사용하는 방법을 triple rail-coding 방법이라 한다.

본 논문에서는 세 가지 다른 수를 가지는 변형부호화자리수를 부호화하고 입력의 동일화 과정을 수행하기 위해서 세 개의 화소 중에서 밝은 화소가 가장 위쪽에 있을 때 1, 중간일 때 0, 아래쪽에 있을 때를 $\bar{1}$로 하여 입력을 부호화하는 triple rail-coding 방법을 사용하였다.

표 2. 공간 이동시킨 입력들의 직렬연결
Table 2. Serial arrangement of space shifted Inputs.



2. 중복된 연산값을 가지는 입력의 동일화

두 단계 변형부호화자리수 가산의 첫 번째 단계에서 사용하는 입력 (x_i , y_i)는 (0 , 0), (0 , 1), (1 , 0), (0 , $\bar{1}$), ($\bar{1}$, 0), (1 , $\bar{1}$), ($\bar{1}$, 1), (1 , 1), ($\bar{1}$, $\bar{1}$)이고 총 9가지이다. 가산을 수행할 경우 표 1(b)와 같이 동일한 가산결과 값을 가지는 입력을 두 번 중복하여 사용하므로, 기호치환 방법으로 변형부호화자리수 가산기를 구현할 경우 광의 병렬성을 이용할 수는 있지만 인식과 치환 규칙수가 많아져 광학적 구현이 어려워진다.

입력을 제안한 triple rail-coding 방법으로 부호화한 후 표 2와 같이 공간이동 한 입력 x_i와 입력 y_i을 직렬연결하여 광을 투영하면, 중복 입력인 (0 , 1), (1 , 0)을 (1 , 0)로 (0 , $\bar{1}$), ($\bar{1}$, 0)는 (0 , $\bar{1}$)로 (1 , $\bar{1}$), ($\bar{1}$, 1)은 ($\bar{1}$, 1)의 세 가지 입력으로 동일화할 수 있다.

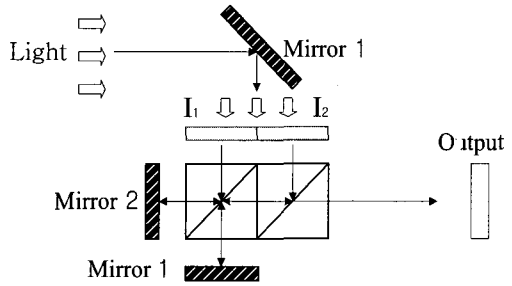


그림 2. 표 2와 4의 구현을 위해 제안한 기본적인 광 구성도

Fig. 2. Proposed basic optical diagram for Table 2 and 4.

표 3. 그림 3에서 각 거울의 이동 정도

Table 3. The distance of shift for each mirror in the Fig. 3.

거울	이동정도
M3	down 3
M4	down 4
M5	down 5

따라서 연산에 사용되는 입력 (x_i , y_i)는 (0 , 0), (0 , 1), (1 , 0), (0 , $\bar{1}$), ($\bar{1}$, 0), (1 , $\bar{1}$), ($\bar{1}$, 1), (1 , 1), ($\bar{1}$, $\bar{1}$)의 9가지에서 (0 , 0), (1 , 0), (0 , $\bar{1}$), ($\bar{1}$, 1), (1 , 1), ($\bar{1}$, $\bar{1}$)의 6가지로 줄어든다. 줄어든 입력을 두 단계 변형부호화자리수 연산에 사용하면 기호치환 과정의 인식과 치환 규칙수를 줄일 수 있다.

표 2을 수행하기 위한 기본적인 구성도는 그림 2와 같이 나타낼 수 있다.

입력영상 x_i와 y_i를 표 2와 같이 이동시킨 후 직렬연결하여 영상 I_1과 I_2를 만든다. 입력영상 x_i와 y_i의 이동 정도는 표 2에 나타나 있다. 거울 1과 거울 2는 입력을 반사하여 중첩하는 역할을 하며, 거울 2는 이동 역할도 수행하여 중복입력들을 동일화한 입력으로 바꾸어준다. 표 2에서 입력 (0 , 1), (0 , $\bar{1}$)과 (1 , 0), ($\bar{1}$ 0)은 이동하는 정도가 같으므로 그림 2의 입력 I_1 , I_2로 함께 쓰일 수 있고, 거울 2에 의해 아래로 4화소만큼 이동하는 정도도 같으므로 그림 2의 구성도를 사용하여 동일화한 입력 (1 , 0), (0 , $\bar{1}$)을 한꺼번에 얻을 수 있다. 마찬가지로 그림 2의 구성도을 이용하여 (1 , $\bar{1}$), ($\bar{1}$, 1)의 동일화한 입력 ($\bar{1}$, 1)를 얻을 수 있다. 입력 (0 , 0), (1 , 1), ($\bar{1}$, $\bar{1}$)은 동일화가 필요 없으므로 표 2와 같이 이동 없이 직렬연결하면 된다. 표 2에 대한 전체적인 광 구성도는 그림 3과 같다. 그림 3에서 거울의 이동 정도는 표 3에 나타내었다.

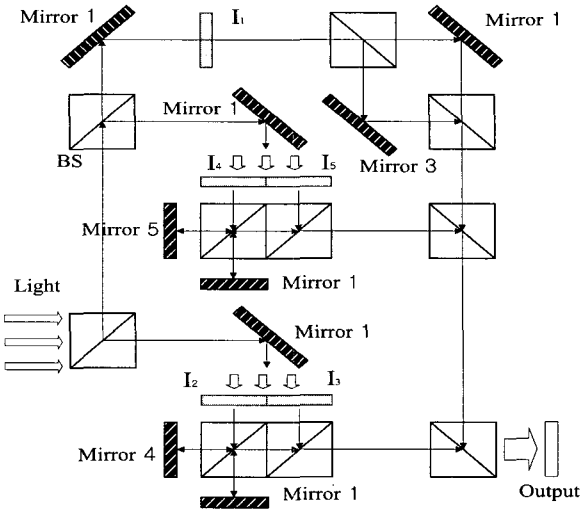


그림 3. 입력의 동일화를 위해 제안된 광 구성도

Fig. 3. Proposed optical Diagram for combing same inputs.

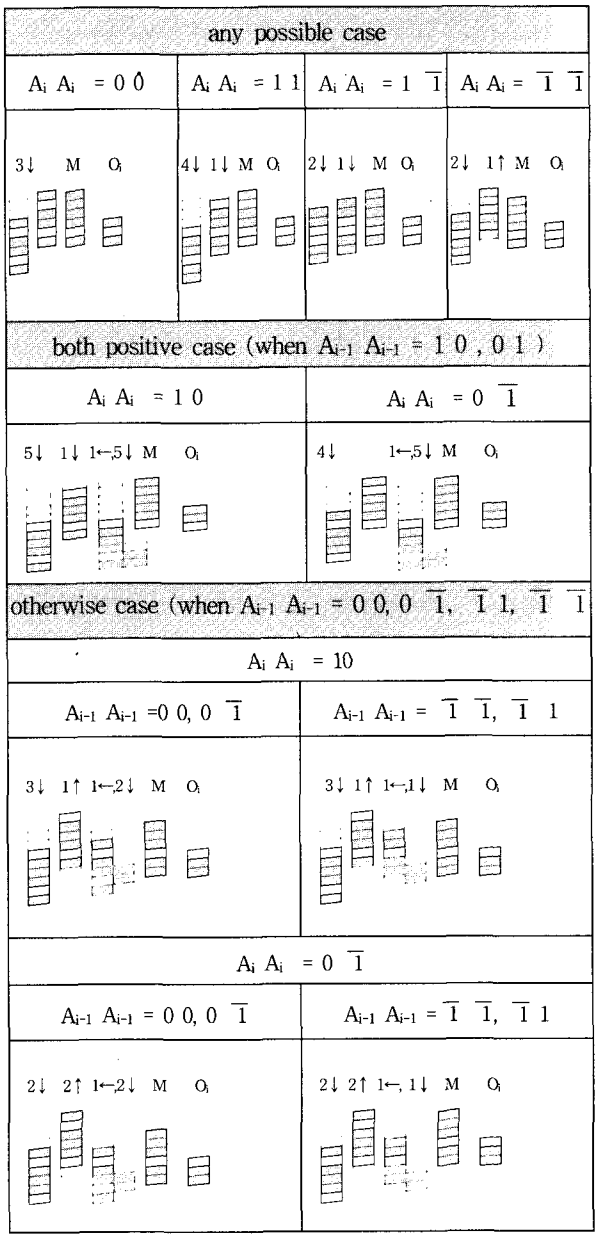
3. 입력의 직렬연결에 의한 첫 번째 단계 기호치환

그림 3에 의해 동일화 된 입력을 A_i라 한다면 표 4와 같이 이동한 입력 A_i들을 직렬연결하여 마스크 M를 통과시키면 기호치환의 인식결과 O_i를 쉽게 얻을 수 있다. 또한, 동일화한 입력의 직렬연결을 이용하여 기호치환을 수행할 경우, 두 단계 변형부호화자리수 가산에서 필요한 첫 번째 단계의 연산 규칙수를 크게 줄일 수 있다. 왜냐하면 동일화한 입력들을 이동하여 직렬연결한 경우 현재 비트 (x_i , y_i)가 (1 , 1), (0 , 0), ($\bar{1}$, 1), ($\bar{1}$, $\bar{1}$)인 경우 하위 비트와 상관없이 기호치환되므로 4가지의 기호치환 규칙이, 현재 비트 (x_i , y_i)가 (1 , 0), (0 , $\bar{1}$)인 경우 하위 비트가 (1 , 0), (1 , 1)인 'both positive' 이지만 동일화한 입력을 3개 사용하여 직렬연결하면 단 2가지의 기호치환 규칙이, 그 외의 하위 비트가 'otherwise' 인 경우도 동일화한 입력을 3개 사용하여 직렬연결하면 4가지의 규칙만이 필요하다. 결과적으로 첫 번째 단계 구현에 필요한 연산 규칙 수를 41가지에서 10가지로 줄일 수 있다.

표 4를 수행하기 위한 기본적인 광 구성도는 그림 2와 같다. 표 4는 공간이동 시킨 입력을 직렬연결하여 영상을 인식하는 과정을 나타낸 것이며 그림 2의 I_1 , I_2 부분 내에서 이루어진다. 인식된 입력 O_i는 거울 1과 거울 2의 반사와 공간 이동을 통해 중첩되어 중간입력 B_i로 만들어진다. 거울 2에 의해 공간이동 되는 정도가 같은 인식영상 O_i은 그림2의 입력영상으로 사용할 수 있다.

표 4. 공간 이동 시킨 입력들의 직렬연결방법을 이용한 첫 단계 기호치환 인식과정

Table 4. Recognition process of first step MSD Symbolic substitution' utilizing the serial arrangement me thod of space shifted Inputs.

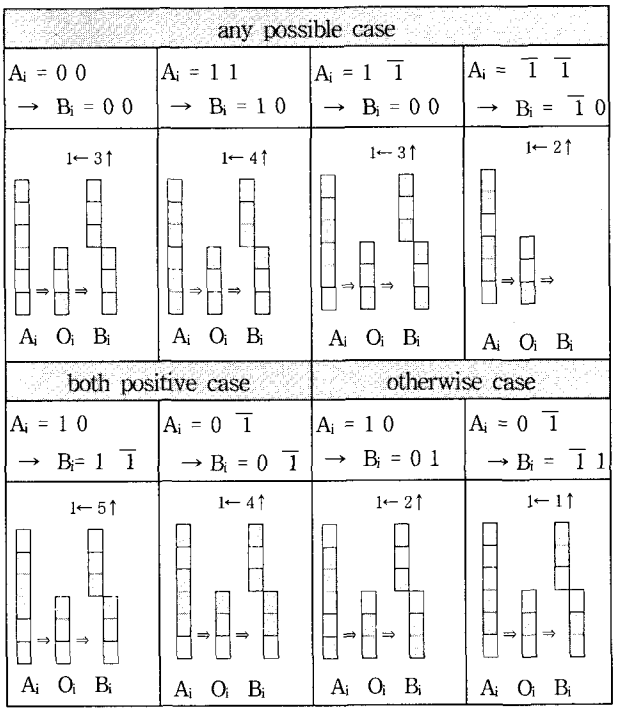


예를 들어 표 4의 입력 (A_i, A_i)가 (1 , $\bar{1}$), (0 , 0)인 경우 그림2의 거울 2에 의해 공간이동 되는 정도가 좌로 1화소, 위로 3화소와 같으므로 I_1, I_2의 입력으로 함께 사용할 수 있다. 동일한 방법으로 표 4의 나머지 부분들을 구현하면 두 단계 변형부호화자리수 가산기의 두 번째 단계의 입력인 중간입력 B_i을 얻을 수 있다.

그림 2을 통해 인식과 치환 과정을 거친 후 나타나는 첫 번째 단계의 최종출력 B_i은 표 5와 같다. 그림 2

표 5. 이동과 중첩에 의한 첫 단계 기호치환의 치환과정

Table 5. Substitution process of first step MSD Symbolic substitution' utilizing the space shift and super -position of input Oi.



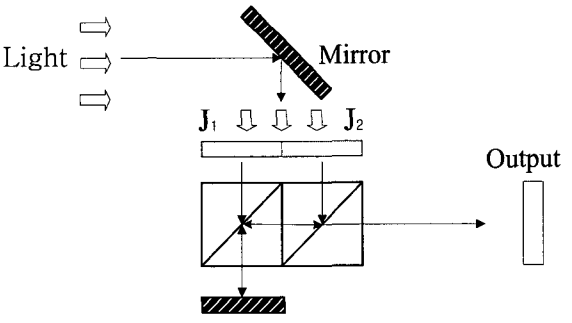


그림 4. 표 6의 구현을 위해 제안한 기본적인 광 구성도

Fig. 4. Proposed basic optical diagram for Table 6.

에서 거울 2는 I_1이나 I_2를 통해 나타난 인식영상을 다른 정도로 이동하고 중첩하는 역할을 수행하여 표 5의 최종출력 B_i들을 만든다.

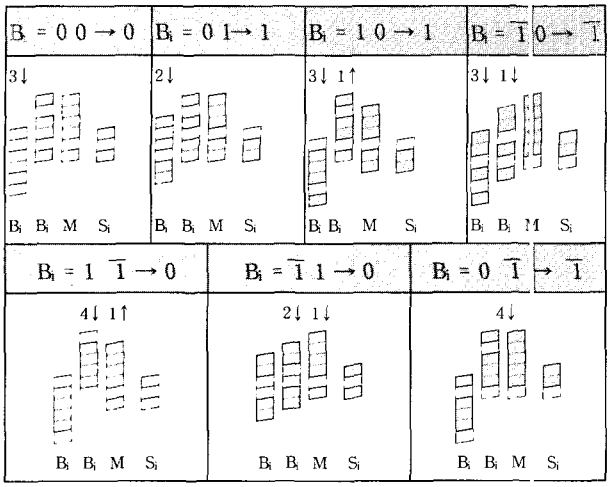
4. 입력의 직렬연결에 의한 두 번째 단계 기호치환

첫 번째 단계의 기호치환과 마찬가지로 두 번째 단계 기호치환 최종 출력 S_i는 공간이동 한 두 개의 입력 B_i을 마스크와 직렬연결한 광을 투영시켜 얻을 수 있다. 이 과정을 표 6으로 나타내었다.

표 6의 구현을 위한 기본적인 광 구성도는 그림 4와 같다.

표 6. 공간 이동 시킨 입력들의 직렬연결방법을 이용한 두 번째 단계 기호치환 결과

Table 6. Second step MSD Symbolic substitution' results of utilizing the serial arrangement method of space shifted Inputs.



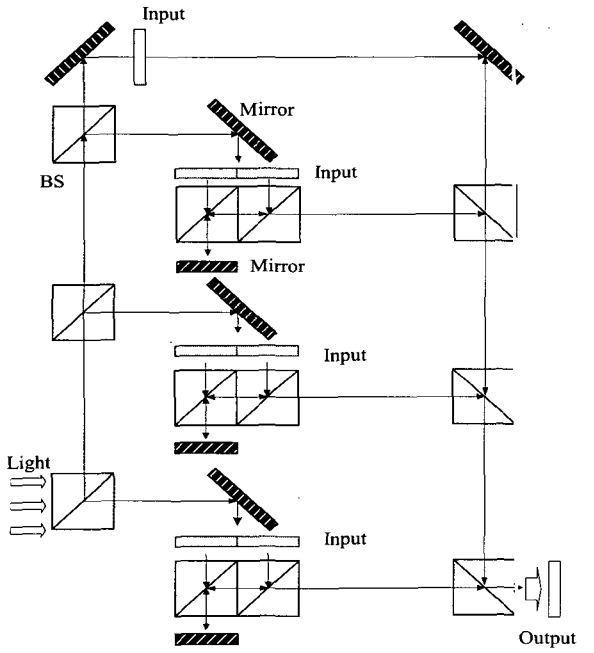


그림 5. 제안한 두 번째 단계 기호치환을 위한 전체적 광 구성도

Fig. 5. Proposed optical diagram for second step symbolic substitutions.

그림 4에서 입력 J_1이나 J_2속에는 표 6의 과정이 포함되어 있다. 거울은 반사를 이용하여 J_1과 J_2를 중첩시켜 최종 출력 S_i를 만든다. 동일한 방법으로 표 6의 나머지 부분들을 구현할 수 있다.

그림 4를 기본으로 하여 두 번째 단계 기호 치환의 전체적인 광 구성도는 구성하면 그림 5와 같다.

그림 5의 총 7개의 입력안에는 표 6에 해당하는 과정

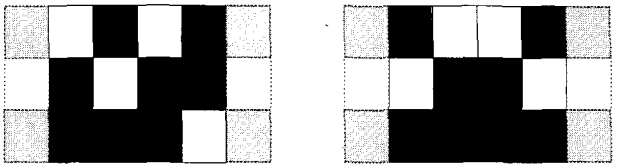


그림 6. Triple rail-coding 방법으로 부호화한 (a) 입력 X (b) 입력 Y

Fig. 6. Encoded inputs by triple rail-coding method (a) input X (b) input Y.

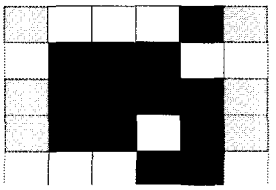


그림 7. 표 2의 의한 전처리된 입력

Fig. 7. Preprocessed input by table 2.

들을이 포함되어 있다. 제안한 두 번째 단계 기호치환 시스템은 인식 자체과 동시에 치환이 이루어지므로 별도의 기호치환의 별도의 치환과정이 필요없다. 따라서 최종출력(S_i)가 최종 가산결과값이 된다.

Ⅳ. 4비트 입력을 이용한 제안한 변형부호화자리수 가산기의 모의 실험

4 비트의 입력 X=1 0 1 $\bar{1}$(910)와 Y=0 1 1 0(610)를 사용하여 제안한 변형부호화자리수 가산기를 구현해 보았다. 그림 6은 두 입력 X와 Y를 제안한 방법으로 부호화한 것이다.

변형부호화자리수의 규칙수를 줄이기 위해 입력 X와 Y에 표 2의 입력동일화 규칙을 적용하여 새로운 입력을 만든다. 이를 수식으로 나타내면

$$\begin{matrix} x \\ +y \end{matrix} \Rightarrow \begin{matrix} \varnothing 1\ 0\ 1\ \bar{1}\ \varnothing \\ \varnothing 0\ 1\ 1\ 0\ \varnothing \end{matrix} \Rightarrow \begin{matrix} \varnothing 1\ 1\ 1\ 0\ \varnothing \\ \varnothing 0\ 0\ 1\ \bar{1}\ \varnothing \end{matrix} \quad (2)$$

와 같고, 그림 7과 같이 표현할 수 있다.

그림 7의 입력에 표 4와 표5의 인식, 치환 규칙들을 적용하여 첫 번째 단계 변형부호화자리수 연산을 할 수 있다. 우선 그림 7의 각각의 입력 비트에 대해 표 4의 규칙을 적용하여 구한 첫 번째 단계의 인식과정과 결과는 그림 8(a), 그림8(b) , 그림 9, 그림 10과 같이 나타낼 수 있다.

위의 인식된 패턴들을 표 5를 이용해 첫 번째 단계의 치환을 수행한 최종 결과는 그림 11(a)이다. 이 때 그림 11(b)와 같이 최상위 비트와 최하위 비트를 제거하면

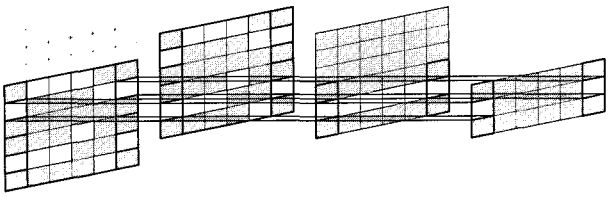


그림 8. any possible인 경우 첫 번째 단계의 인식
Fig. 8. First step MSD recognition process when any possible case.

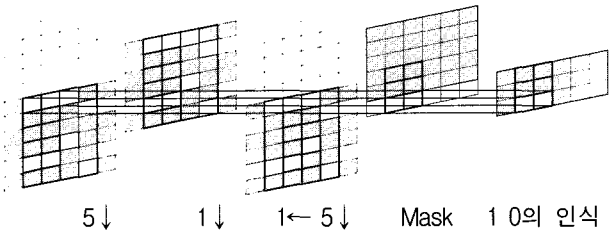


그림 9. both positive인 경우 첫 번째 단계의 인식
Fig. 9. First step MSD recognition process when both positive case.

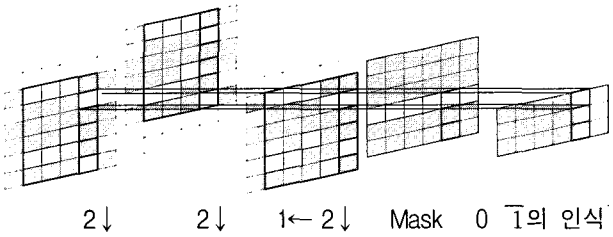


그림 10. otherwise 인 경우 첫 번째 단계의 인식
Fig. 10. First step MSD recognition process when otherwise case.

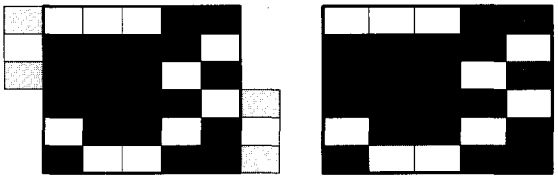
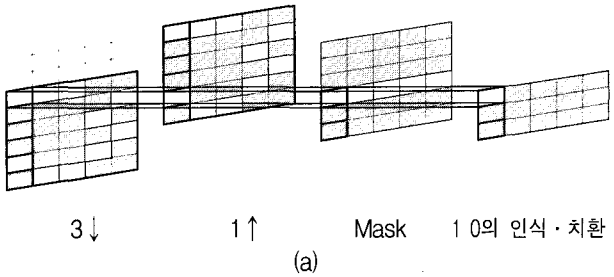


그림 11. (a) 첫 번째 단계 최종 출력패턴 (b) 두 번째 단계 입력패턴
Fig. 11. (a) First step MSD final result pattern (b) second step MSD input pattern.



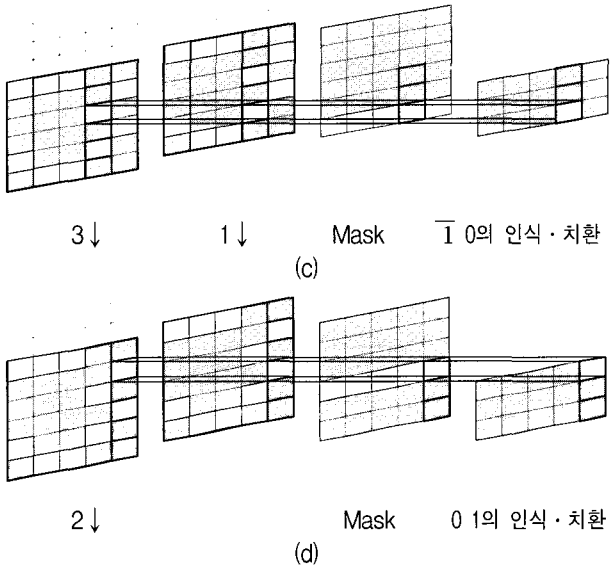


그림 12. 두 번째 단계 기호치환 과정
Fig. 12. Second step MSD symbolic substitution process.

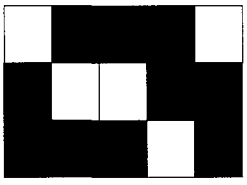
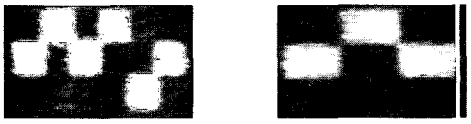


그림 13. 최종 연산결과 패턴
Fig. 13. Final arithmetic result pattern.

두 번째 단계의 입력 패턴으로 사용할 수 있다.

그림 12의 (a), (b), (c), (d)는 표 6의 규칙을 사용였을 때의 두 번째 단계 기호치환연산과정과 결과를 나타낸 것이다.

그림 12의 결과패턴들을 중첩한 입력 X, Y의 최종 연산결과 패턴은 그림 13이며 이 때, 최종 연산결과 패



(a) (b)

그림 14. Triple rail-coding 방법으로 부호화한 (a) 입력 X (b) 입력 Y

Fig 14. Encoded by triple rail-coding method (a) input X (b) input Y.

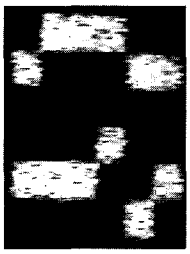


그림 15. 첫 번째 단계의 입력패턴 실험결과

Fig. 15. Experiment result of first step MSD input pattern.

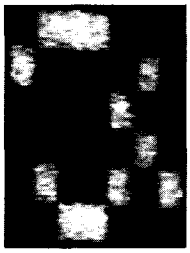


그림 16. 첫 번째 단계 최종출력 실험결과

Fig. 16. Experiment result of first step MSD final result.

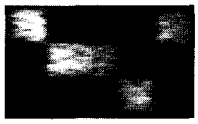


그림 17. 최종 연산결과 패턴

Fig 17. Final arithmetic result pattern.

턴은 입력 X와 Y의 합의 결과인 1 0 0 $\bar{1}$ 1 (1510)과 동일함을 확인 할 수 있다.

V. 실험결과 및 고찰

본 실험에서는 앞장의 예제로 제시된 입력 데이터로써 4 비트의 X=1 0 1 $\bar{1}$(910)와 Y=0 1 1 0(610)를 사용하였고 이를 triple rail-coding 방법으로 부호화 하여 CCD로 찍은 영상은 그림 14와 같다.

위 입력 패턴을 그림 3의 광 구성도의 입력으로 사용하여 동일화한 입력으로 만든 실험결과는 그림 15이며, 이는 그림 7의 시뮬레이션 결과와 동일하다.

그림 15의 결과를 그림 8의 광 구성도의 입력으로 이용하여 구한 첫 번째 단계의 출력은 그림 16이며, 이 역시 시뮬레이션 결과인 그림 11과 동일하다. 여기서 입력패턴과 마스크는 마이크로 필름을 사용하였다.

그림 17은 그림 16의 패턴을 그림 5의 광 구성도 입력으로 사용하여 나타낸 두 번째 단계의 최종출력이다. 시뮬레이션 최종 결과인 그림 13과 동일함을 확인할 수 있다.

실험에서 사각형이 조금 일그러지는 것은 그 원인이 빛의 회절 때문이라 여겨지며, 실험결과를 얻는데 크게 영향을 주지 않았다. 제안한 광 가산 시스템은 거울이나 광 분광기 같은 실험 소자의 크기로 인하여 전체 시스템이 다소 크지만 집적 광학 기술을 사용하여 소자를 칩으로 대체한다면 크기를 줄일 수 있을 것이다.

VI. 결 론

본 논문에서는 두 단계 변형 부호화자리수 가산기를 구현하기 위해 입력을 triple rail-coding 방법으로 부호화하고 이를 직렬연결하여 패턴을 인식하고 치환하는 새로운 기호치환 규칙을 제안하였다. 중복입력들의 형태를 통일하는 입력동일화 시스템과 공간 이동시킨 입력들의 직렬연결을 사용하여 두 단계 변형부호화 자리수 연산의 첫 번째 단계에 필요한 규칙수를 41가지에서 10가지로 줄였다. 이동시킨 입력들을 직렬연결하여 광을 투영하기만하면 인식과 치환이 되므로, 기호치환의 인식과 치환단계에서 필요한 공간이동 연산이나 NOR 문턱치 연산같은 부가적인 연산을 생략할 수 있으며, 부가적인 연산을 생략함으로써 광 가산기의 크기를 줄였고, 연산속도도 향상시켜 기호치환의 장점인 병렬성을 최대로 하였다. 본 논문에서는 4비트의 입력 데이터를 사용하여 모의 실험 및 광 실험을 통해 제안한 광 가산기의 구현과정을 확인하였다. 본 실험에서 입력 데이터와 마스크를 필름으로 제작하였으나 이 대신에 광 변조기를 사용한다면 실시간 처리가 가능한 광 가산기를 구현할 수 있을 것이다. 거울이나 광 분광기와 같은 광 수동소자의 소형화가 이루어진다면 제안한 광 가산기의 실질적인 이용이 가능해지리라 생각한다.

참 고 문 헌

[1] A. W. Lohmann and J. Weigelt, "Digital optical adder based on spatial filtering," *Applied Optics*, vol. 25, no. 18, pp. 3047-3053, September 1986.

[2] D. Psaltis and D. casasent, "Optical residue arithmetic: a correlation approach," *Applied Optics*, vol. 18, no. 2. pp. 163-171, January 1979.

[3] M. S. Alam, M. A. Karim, A. A. S. Awwal and J. J. Westerkamp, "Optical processing based on conditional high-order trinary modified signed -digit symbolic substitution," *Applied Optics*, vol. 31, no. 26, pp. 5614-5621, September 1992.

[4] A. Huang, "Parallel algorithms for optical digital computers," in Technical Digest, *IEEE* Tenth International Optical Computing Conference, pp. 13-17, 1983.

[5] K-H. Brenner, A. Huang, and N. Streibl, "Digital optical computing with symbolic substitution," *Applied Optics*, vol. 25, no. 18, pp. 3054-3060, September 1986.

[6] K-H Brenner, "New implementation of symbolic substitution logic," *Applied Optics*, vol. 25, no. 18, pp. 3061-3064, September 1986.

[7] H. Huang, M. Itoh, and T. Yatagai, "Modified signed-digit arithmetic based on redundant bit representation," *Applied Optics*, vol. 33, no. 26, pp. 6146-6156, September 1994.

[8] S. Barua, "Single-stage optical adder/subtracter," *Optical Engineering*, vol. 30, pp. 265-270, 1991.

저 자 소 개

신 창 목(정회원)
1997년 경북대학교 전자공학과 학사 졸업(공학사).
2000년 경북대학교 대학원 전자공학과 석사 졸업(공학석사).
2000년～현재 경북대학교 대학원 전자공학과 박사과정 재학중
<주관심분야: 광 컴퓨팅, 광 보안, 홀로그램 및 광 메모리>

김 수 중(정회원)
제40권 SD편 9호 참조

서 동 환(정회원)
제40권 SD편 9호 참조
현재 한국해양대학교 전자전기공학부 교수