Order Number 9108132

Optical arithmetic-logic processors based on location, content addressable and associative memories

Kostrzewski, Andrew Adam, Ph.D.

City University of New York, 1990

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

A

OPTICAL ARITHMETIC-LOGIC PROCESSORS BASED ON LOCATION, CONTENT ADDRESSABLE AND ASSOCIATIVE MEMORIES

by

ANDREW KOSTRZEWSKI

A dissertation submitted to the Graduate Faculty in Engineering in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York.

1990

This manuscript has been read and accepted for the Graduate Faculty in Engineering in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

9/5/90
Date

Chairman of Examining Committee

9/14/90
Date

Executive Officer

Prof. S. Ahmed

Prof. P. Karmel

Prof. R. Dornsinville

Dr. I. Kadar

The City University of New York

ABSTRACT

OPTICAL ARITHMETIC-LOGIC PROCESSORS BASED ON LOCATION, CONTENT ADDRESSABLE AND ASSOCIATIVE MEMORIES

by

Andrew Kostrzewski

Mentor: Dr. George Eichmann, Herbert G. Kayser Professor of Electrical Engineering

(Deceased, June 1990)

Co-mentor: Dr. Yao Li, Assistant Professor of Electrical Engineering

Final Mentor: Dr. Yao Li

Applications of content addressable, location addressable and associative memory for optical signal processing are studied in this thesis. The first two types of memory may use either coherent or noncoherent processing, but the last type uses only coherent processing.

A new real-time optical position coded processor is presented and experimentally implemented. The crossbar switch is implemented by using a single laser

source with two cascaded commercial liquid-crystal TVs, with operands represented as mutually orthogonal slits. The intersection of the slits represents an operation result and the collection of light output points constitutes a look-up table for a specific arithmetic-logic operation. A holographic mapping maps each truth-table point to a proper position coded result. Experimental results for a residue number multiplier are presented. An optical implementation of multiple-valued logic is also presented. Realizations of optical modified signed-digit as well as Post logic gates are included. Some experimental demonstrations are presented.

A new scheme for digital optical computing utilizing a non-holographic opto-electronic content addressable memory is discussed. Designs of optical binary carry look-ahead adder and modified signed digit adder are presented. Also implementations of binary number, a sign/logarithm number and residue number multiplier are included to illustrate the application of this arithmetic processor. This opto-electronic content addressable memory offers a number of practical advantages, such as fast processing speed and ease of optical implementation and alignment compare to other approaches. Active low and high spatial content addressable memory mask encoding techniques are discussed. Multioperation and multibit content addressable memory processors are described.

Optical register transfer micro-operations based on associative memory are proposed. A hybrid optical word-parallel, bit serial register transfer processor architecture based on an optical holographic associative symbolic substitution is described and preliminary experimental results are presented.

This research was supported in parts by the Air Force Office of Scientific Research under grant AFOSR 88-0260 and by National Science Foundation under grant MIP-8921337.

ACKNOWLEDGEMENT

This research was done with the guidance and support of Dr. George Eichmann, Herbert G. Kayser Professor of Electrical Engineering at The City College and at The Graduate School of the City University of New York. Professor Eichmann died suddenly in June 1990 when all but the final editing of the dissertation was completed. Dr. Eichmann's work will influence my main areas of interest for years to come and will be continued through my future research.

During the research work I also received substantial guidance from Professor Yao Li, who also stepped in as mentor at Professor Eichmann's death. Professor Samir Ahmed also assisted during these last months.

I would like to acknowledge financial support provided by the Air Force Office of Scientific Research under grant AFOSR 88-0260 and by National Science Foundation under grant MIP-8921337

To my wife Magda

and my daughter Emily Nicole

Table of Contents

Table of Figures

Table of Tables

I. INTRODUCTION

In recent years, to perform arithmetic operations, different optical schemes have been introduced. Due to the increasing demand for high speed arithmetic calculations, new, fast optical computational systems are required. For arithmetic operation, either a logic or a memory-based computation can be performed.

In a logic-based approach, the arithmetic operation is performed by using an algorithm that is executed via a logic gate circuit [1]. Here, depending on the employed number system, the key element for such an operation is a set of binary Boolean, multiple-valued, or residue, logic gates. To implement these logic operations, using state of the art optical technology, various acoustooptic, electrooptic, and nonlinear optical schemes have been investigated [2-4]. Among the various binary optical logic schemes, intensity-encoded bistable devices can be operated at GHz speed. On the other hand, using relatively slow optical two-dimensional (2D) spatial light modulators (SLMs) , optical binary pattern logic has shown both flexibility and a large degree of parallelism. In this category are optical theta-modulation [5], shadow-casting [6], triple-product [7], as well as scattering-based [8] optical logic processors. To increase the pattern logic's processing speed, recently, a picosecond optical-phase-conjugation-based pattern logic device has also been reported [9].

An alternative to a logic-based approach is the use of memory for arithmetic operations. In this case, all possible inputs and their outputs are tabulated and codified as truth-tables. For a given input set, the outputs can be found in two

different ways. In a straightforward truth-table look-up scheme, each stored output is assigned an address. To access this output, the given input set serves as its address code. For this reason, this method is also called a location-addressable memory (LAM). Because of the direct storage of all input-output pairs, for large dynamic-range arithmetic operation, large memory space is needed. However, since this method provides the fastest possible arithmetic operation speed, whenever the memory space permits, it is still the preferred scheme. To retain the speed advantage while improving the memory storage efficiency, a so-called content-addressable memory (CAM) scheme was suggested [12]. The difference between the CAM and the LAM is in the way the data is addressed. For a CAM, all possible inputs that can provide the same output are grouped and the redundant information contained in the group is reduced. Thus, depending on the particular arithmetic operation, two- to, even more than, ten-fold information reduction is possible. As a result, the storage efficiency is greatly increased. Although an additional redundant information reduction is required, since it is a precalculation, it does not affect the speed of the arithmetic operation.

A third type memory that can be applied for optical signal processing is an associative memory (AM). The main difference between AM and CAM is that while the former is designed to be able to cope with incomplete or imperfect input information, the latter is designed to deal with complete knowledge of its input. The associative memory-based computation is a two step process. First, the AM memory is generated based on all input -output pairs, next only an input is used as an address to access corresponding output. To implement AM-based processor, one-step holographic AM is employed [30].

1.1 Location Addressable Memory-Based Processing

The application of LAM-based processor for residue number (RN) arithmetic as well as for multi-value logic, including implementation of modified signed-digit (MSD) number system 3-level gates, has been studied. Unlike the logic-based approach, here for generating and matching the address code, optical switching rather than a complicated gating operation is used. Because there is no need for cascading, the advantage of fast optical switching is preserved.

A truth-table look-up operation has been described for either multi-digit binary or single digit multiple-valued logic (MVL) generation [11-12]. Also, the application of LAM-based processing for RN operations has been reported [17-19]. With this scheme, the arithmetic or logic operations are expressed as truth-tables. Using various optical encoding (i.e., position, intensity, polarization, etc.) methods, location addressable optical truth-tables can be implemented [11]. However, when using a binary number system, the truth-table size depends on the input's dynamic range. Thus for large numbers, this scheme may become impractical. Using a residue number system, there are methods to decompose an arithmetic operation into a set of parallel suboperations.

The truth-table look-up also can be used for single-digit MVL processing. The use of a MVL number system enhances the computation efficiency both by offering parallel processing capability and by processing/interconnecting more information through each element [20-22]. In this thesis a real-time holographic optical truth-table method for optical MVL generation is presented. In particular, the modified signed-digit (MSD) [23-27] and Post [28] logic are considered. Some

preliminary experimental results are also included.

1.2 Content Addressable Memory-Based Processing

Optical CAM can be implemented using either holographic or non-holographic techniques. While the former requires coherent light, the later can be implemented using either coherent or noncoherent optical processing. To record a CAM hologram, a relatively complicated 3-step process is required [11-12]. Since each reference pattern requires a phase multiplexed reference beam, for a large number of reference patterns the generation of a CAM hologram is a difficult task. An additional disadvantage of a holographic CAM is its low diffraction efficiency and crosstalk during the readout process. Here, because of the ease of alignment and implementation, in comparison to the holographic CAM, a non-holographic CAM approach is introduced [13-14]. Using a non-holographic CAM, any arithmetic operation representable as a Boolean sum of product terms can be implemented. To encode a logic function into a CAM mask, either active low or high level logic can be applied. For active low (high) logic, the minimum (maximum) light intensity transmitted through the CAM mask indicates that a match is found. With an active low logic, to provide an active high result [13], the thresholded output is inverted. For the active high case, by setting the threshold at a high intensity level, the final result can be obtained directly without any inversion.

As an example of the non-holographic CAM based arithmetic processing, the design of an optical carry look-ahead adder (CLA), binary number (BN),

sign/logarithm number (SLN) and residue number (RN) multipliers will be described. In the case of a CLA, regardless of the number of processed bits, the carry is precalculated and the final addition result is obtained in a single step. The existing optical CLA schemes, because of their use of spatial light modulators (SLMs) [15-16], allow only moderate processing speed. By using an array of laser diodes (LDs) to illuminate a non-holographic CAM mask, fast processing speed is anticipated. Experimental results for a 4-bit optical CLA are presented. By incorporating the input carry to an optical CLA, a cascade of a number of 4-bit adders can be achieved. Using the reduced truth tables stored in a non-holographic CAM, various BN, SLN and RN multipliers are also implemented. The design and implementation of a 3-bit BN multiplier is presented. Based on the moduli 3, 5 and 7, a RN multiplier, where the RNs were encoded using a BN representation, is designed. Also, experimental results for a 7-bit SLN multiplier are included.

1.3 Associative Memory-Based Processing

A combinatorial logic circuit is an interconnected array of logic gates or switches. However, for various arithmetic operations, iterative sequential computation is needed. To furnish feedback for combinatorial circuits, memory elements, such as flip-flops or registers, must be utilized. With this feedback, the overall logic circuit is a finite-state sequential logic machine. To generate a sequential logic circuit, a viable hybrid approach is to use optics for both fast parallel combinatorial logic, interconnect functions, high-speed bit-addressable

electronics for storage and feedback [28]. In this thesis, a hybrid sequential computing module, where an optical array processor performs the combinatorial logic and interconnect operations between high-speed electronic parallel addressed storage registers, is described. This hybrid system is able to perform fast optical register transfer micro-operations (ORTMOs), that represent the primitive operations required for a general-purpose optical digital computer. Using these operations, other, such as residue arithmetic operations can be constructed. This new system will be referred to as an optical register transfer processor (ORTP).

For an optical register transfer micro-operation (ORTMO) processor implementation, the recently developed symbolic substitution technique [29-32] can be used. Here, an optical holographic associative symbolic substitution (OHASS) technique, proposed by Yu et. al. [31-32] is employed.

1.4 Research Goal And Thesis Organization

The research goal was to use available optical elements and techniques, to implement various fast, fully parallel multi-bit optoelectronic arithmetic processors, and to develop efficient hardware, and software algorithms for these processors. For a dedicated arithmetic processor, capable of performing addition, subtraction, multiplication and division operations a comparison among a binary, residue, modified signed-digit and sign/logarithm number systems processors is performed. The number system conversion issue is also addressed. Specific research issues such as the source uniformity, the optical element alignment,

the required speed of the detectors, high resolution CAM mask fabrication, and the electronic threshold operation are also discussed.

The thesis has been divided into eight chapters. This introductory chapter provides background information concerning optical signal processing based on three types of optical memories. In the second chapter, review of binary, sign/logarithm, residue and modified signed-digit number system is included. Beginning with the third chapter, information concerning the use of three types of optical memories is presented. First, the location addressable memory-based processing is presented. Look-up table-based optical processors are described. The application of LAM for residue multiplier/adder MSD adder and multi-valued Post logic is presented. In chapter four, the non-holographic CAM, including active high and low, encoding is described. Implementation of the non-holographic CAM for BN, MSD, SLN, and RN processors are described. Design of single-operation as well as multi-bit, multi-operation processor is discussed. The experimental implementation of binary and MSD adders as well as binary, RN, SLN multipliers is included. In chapter five, based on associative memory, an optical register transfer micro-operation processor is described. Using a recently developed symbolic substitution technique, logic and transfer operations can be implemented. Chapter six contains the thesis summary and conclusions. Chapter seven and eight contain thesis related publications and bibliography, respectively.

1.5 References

[1] Z. Kohavi, Switching and Finite Automata Theory, McGraw-Hill, NY 1978.

[2] H. J. Caulfield, Optical Computing, Editor SPIE Milestones series, vol. 1142, 1989.

[3] A. W. Lohmann "What Classical Optics can Do for the Digital Optical Computing," Appl. Opt. 25, 1543 (1985).

[4] R. Arrathoon and S. Kozatis, "Architectural and Performance Consideration for a 10^7-Instructions/sec Optoelectronic Central Processing Unit," Opt. Lett. 12, 956 (1987).

[5] H. Bartelt, A. W. Lohmann and E. E. Sicre, J. Opt. Soc. Am. A, 1, 944 (1984).

[6] Y. Ichioka and J. Tanida, Proc. IEEE, 72, 787 (1984).

[7] T. Yatagai, Opt. Lett. 11, 260 (1986).

[8] A. W. Lohmann, and J. Weigelt, Opt. Comm. 54, 81 (1985).

[9] Y. Li, G. Eichmann, R. Dorsinville and R. R. Alfano, "Parallel Ultrafast Digital and Symbolic Optical Computation via Optical Phase Conjugation," Appl. Opt. 27, 2025 (1988).

[10] Y. Li, B. Ha, A. Kostrzewski, D. H. Kim, and G. Eichmann, "Optical Position-Coded Multiple-Variable Logic and Arithmetic Using Liquid-Crystal TVs and Holograms," Optics Comm. 70, 379 (1989).

[11] C. C. Guest and T. K. Gaylord, "Truth-Table Look-Up Optical Processing Utilizing Binary and Residue Arithmetic, " Appl. Opt. 19, 1201 (1980).

[12] M. M. Mirsalehi and T. K. Gaylord, " Logical Minimization of Multilevel Coded Functions," Appl. Opt. 25, 3078 (1986).

[13] Y. Li, D. H. Kim, A. Kostrzewski and G. Eichmann, "Content-addressable-memory-based Single-stage Optical Modified-signed-digit Arithmetic," Opt. Lett. 14, 1254 (1989).

[14] A. Kostrzewski, Y. Li, G. Eichmann and D. H. Kim, " Fast Hybrid Parallel Carry Look-Ahead Adder," Opt. Lett. 15, (1990).

[15] Y. Jin and F.T.S. Yu, "Optical Binary Adder Using Liquid Crystal Television", Opt. Comm. 65, 1 (1988).

[16] R. Golshan and J.S. Bedi, " Implementation of Carry Look-ahead Adder with Spatial Light Modulators," Opt. Comm. 68, 3, 175 (1988).

[17] A. Huang, Y. Tsunoda, J. W. Goodman and S. Ishihara, "Optical Computing Using Residue Arithmetic," Appl. Opt. 18, 149 (1979).

[18] A. Kostrzewski, Y. Li, D. H. Kim, B. Ha and G. Eichmann, " Optical Position Coded Residue Processor Using Inexpensive LCTV Devices," Appl. Opt. 28, 415 (1989).

[19] A. P. Goutzoulis, "Comparison of Laser Diode Electronic Residue Position-Coded Look-Up Tables," Opt. Eng. 28, 373 (1989).

[20] S. L. Hurst, IEEE Trans. Comput. C-33, 1160 (1984).

[21] T. T. Dao, E. J. McCluskey, and L. K. Russell, Opt. Eng. 25, 14 (1986).

[22] G. Eichmann, Y. Li, and R. R. Alfano, Appl. Opt. 25, 3113 (1986).

[23] A. Avizienis, IRE Trans. Electronic Computers, EC-10, 389 (1961).

[24] B. L. Drake, R. P. Bocker, M. E. Lasher, R. H. Patterson and W. J. Miceli, Opt. Eng. 25, 38 (1986).

[25] M. M. Mirsalehi and T. K. Gaylord, Appl. Opt. 25, (1986).

[26] Y. Li and G. Eichmann, Appl. Opt. 26, 2328 (1987).

[27] K. Hwang and A. Louri, "New Symbolic Substitution Algorithms for Optical Arithmetic using Signed-Digit Representation," Proc. SPIE, 880 (1988).

[28] R. Arrathoon, "Logic based spatial light modulators," Proc. SPIE, 881 230-239 (1988).

[29] K. H. Brenner, A. Huang, and N. Streibl, "Digital optical computing with symbolic substitutions," Appl. Opt. 25, 3054-3060 (1986).

[30] Y. Li, G. Eichmann, R. Dorsinville, and R. R. Alfano, "An AND operation-based symbolic pattern recognizer," Opt. Comm. 63, 375-379 (1987).

[31] F. T. S. Yu and S. Jutamulia, "Implementation of symbolic substitution logic using optical associative memories," Appl. Opt. 26, 2293-2294 (1987).

[32] F. T. S. Yu, C. Zhang and S. Jutamulia, "Applications of one-step holographic associative memory to symbolic substitution," Opt. Eng. 27, 399-402 (1988).

II. REVIEW OF SELECTED NUMBER SYSTEMS

In this chapter, a review of selected number systems that find applications in optical processors is presented. Among many, the binary number (BN) system, the modified signed-digit (MSD) system, the residue number (RN) and recently the sign/logarithm number (SLN) systems are studied for use in optical computational processors. The BN system, widely used in electronic applications, can be used in general purpose processors. Due to the carry propagation, the sequential computational algorithms limit the processing speed, although parallel algorithms (e.g. carry-look ahead addition) can be applied to improve the processing speed [1-2]. Applying other number systems, processing speed improvement can be achieved. The SLN system can provide high dynamic range and fast computation speed. The major advantage of this number system is product calculation through logarithm addition [3-4]. The MSD number system with its weak carry dependence is suitable for both parallel addition and subtraction operations [5-6]. Using tree structure fast multiplication operation can be implemented. For RN representation, an arithmetic operation can be decomposed into a set of independent suboperations, the RNs represent an effective parallel code. The high dynamic range and fast processing for addition, subtraction and multiplication operations are attractive features of this number system [7].

2.1 Binary number (BN) system

In general, a number in base r contains r digits 0, 1, 2, 3,..., r-1 and is expressed with a power series in r

$$A_n r^n + A_{n-1} r^{n-1} + \ldots A_0 r^0 + A_{-1} r^{-1} + A_{-2} r^{-2} + \ldots \quad (2.1.1)$$

when the number is expressed in positional notation, only the coefficients and the decimal point are written down:

$$A_n A_{n-1} \ldots A_0 \quad . \quad A_{-1} A_{-2} \ldots \quad (2.1.2)$$

The BN system, although widely employed in electronic computational devices, is suitable for a sequential type of processing. For all arithmetic operations, the carry propagation from digit to digit slows down the computational speed which is inversely proportional to the BN representation length. Some calculations can be evaluated using parallel algorithms such as carry look-ahead addition (CLA), where the carry propagation is precalculated and the resulting output can be represented as a set of Boolean functions. The hardware complexity increases rapidly with the representation length and only moderate dynamic range calculations can be implemented. On the other hand, the major advantage of the BN system is its compatibility with existing electronic devices based also on BNs.

2.2 Modified signed-digit (MSD) number system

In this section, some basic properties of the MSD number system are reviewed. The MSD number representation [1-2] is a redundant radix 2 trinary number system whose weights are 1, 0, and -1 ($\bar{1}$). Using the MSD representation, a decimal number N_{10}, where the subscript denotes base ten, is written as

$$N_{10} = (1,0,\bar{1})2^{p-1} + \ldots + (1,0,\bar{1})2^{1} + (1,0,\bar{1})2^{0}$$

$$N_{10} = \sum_{i} a_i 2^i \qquad (2.2.1)$$

where $a_i \in [-1,0,1]$. Positive as well as negative numbers can be represented in MSD number system. With this representation, decimal numbers A=9 and B=-9 can be expressed as

$$A = 1\bar{1}001 \qquad (2.2.2)$$

$$B = \bar{1}100\bar{1} \qquad (2.2.3)$$

It can be seen that by a parallel bit-wise logic inversion of nonzero digits, the sign of a number can be negated. One advantage of the MSD number system is its parallel addition and subtraction capability. The addition and subtraction operation can be performed using an array of four 2-bit MSD logic gates, e.g.

W, T, W^1 and T^1 gates. The truth tables for these gates are shown in Table 2.2.1. In Fig.2.2.1, a typical 5-bit MSD addition logic flow diagram is shown. Using this architecture, the carries generated from the first processing stage can only propagate 2-bits to the left. The addition result is independent of the number of bits to be added. A parallel subtraction of two MSD numbers, X - Y, can be achieved by negating Y followed by the MSD addition [1]. Multiplication and division operations can be decomposed into a set of MSD additions and subtractions, together with parallel shifts.

2.3 Sign/logarithm number (SLN) number system

SLN system is especially suitable for both multiplication and/or division operations. The addition (subtraction) operations can be also implemented using a SLN system [8]. To multiply (divide) two binary numbers (BNs) **a** and **b**, first the numbers are converted into a SLN system. The multiplication (division) is then obtained by adding (subtracting) appropriate logarithms. As a final multiplication step,

$$ab = antilog_2(\log_2 a + \log_2 b), \qquad (2.3.1)$$

a conversion from SLN to BN system is needed. The major advantage of an SLN system is that the product/quotient calculation is performed via fixed-point binary addition/subtraction. Because BN addition can be performed in a considerably shorter time than multiplication/division, the SLN system can achieve

a multiplication speed that is higher than with any other equivalent length fixed-point number system [9].

Since a logarithm of a BN may contain an infinite number of digits, a binary approximation of the logarithm mantissa is required. The multiplication accuracy depends mainly on the binary logarithm approximation. Using a linear approximation [3], the logarithm of a variable a between points $a=2^n$ and 2^{n+1} (where n is an integer) is approximated by a linear function of a. From [10], the absolute value of the logarithm approximation error E is

$$0 \leq E \leq 0.08639. \qquad (2.3.2)$$

The linear approximation for a logarithm can be implemented using shift operations performed on an n-bit BN. Shifting the BN until the most significant "one" is shifted to the left-most (to the most significant bit (MSB)) position, both the characteristic and the mantissa of the logarithm are obtained. If the number of shifts is equal to m, its characteristic is equal to n-m-1. The mantissa is represented by a BN that lies to the right (after the shift) of the MSB.

By truncating the mantissa at different bit positions, better multiplication accuracy is possible. In Table 3.2.1., multiplication error as a function of the mantissa length is shown. For each additional representation bit the accuracy increases by a factor of two.

2.4 Residue number (RN) system

To represent a number in RN system, a base of the representation must be chosen. A k-tuple of integers (set of relative prime integers m_1, m_2,... m_k called moduli) can be employed. To represent a number X, a set of integers r_1, r_2,...r_k that satisfy following condition

$$r_i = x / m_i, for \quad i = 1, 2, \ldots k \qquad (2.4.1)$$

where r_i denotes the least positive integer remainder of the division of x by m_i, is satisfied. As an example consider base of four moduli 3, 4, 5, 7. Two decimal numbers 23 and 15 are represented by a 4-tuple set of integers:
$23_{10} = 2_7\ 3_5\ 3_4\ 2_3 = 2332_R$ and $15_{10} = 1_7\ 0_5\ 3_4\ 0_3 = 1030_R$, where R stands for RN representation.

To obtain a unique number representation, the represented numbers should not exceed RN's dynamic range. The residue representation dynamic range D_m satisfies the following condition

$$0 \le D_m \le \prod_{i=1}^{k-1} (m_i) - 1 \qquad (2.4.2)$$

The RNs can be represented using multilevel representation or can be encoded using BN system. In the later case, each residue digit is encoded using its binary

equivalent. In Table 2.4.1, the RN dynamic range as a function of bit number of binary encoded RNs is shown.

The major advantage that RN system offers is its carry free arithmetic operations capability. Operations, such as addition, subtraction, and multiplication of two numbers A and B are performed according to

$$[A \odot B]_{m_i} = [[A]_{m_i} \odot [B]_{m_i}]_{m_i} \qquad (2.4.3)$$

where $\odot$ denotes one of the above mentioned arithmetic operations. In this case, an operation is decomposed into a set of k-suboperations, each performed independently and in parallel. To perform an arithmetic operation, a set of residue processors, each corresponding to a different modulus should be employed. By suitably combining the partial results generated by each processor, the final result can be obtained. As an example consider addition and multiplication operations performed on two RN 2332_R (decimal 23) and 1030_R (decimal 15)

$$\begin{array}{r} 2332 \\ +1030 \\ \hline 3322 \end{array} \qquad (2.4.4.)$$

$$\begin{array}{r} 2332 \\ \times 1030 \\ \hline 2010 \end{array}$$

Regardless of the RN major advantages, the disadvantages such as comparison problems, overflow detection and difficulties with division operation prevents the use of the RN system in a general type computational processor. Although in applications where for simple arithmetic operations high processing speed and high dynamic range are essential, the RN system can prove to be advantages.

2.5 References

[1] J. J. Cavanagh, Digital Computer Arithmetic. New York: McGraw-hill, 1984.

[2] I. Flores, The Logic of Computer Arithmetic. New York: Van Nostrsand, 1955.

[3] J. N. Michell, IRE Trans. Electron. Comput., pp 512-517, Aug. (1962).

[4] M. Combet, H. Van Zonneveld, and L. Verbeek, "Computation of the base two logarithm of binary numbers," IEEE Trans, Electronic Computers, vol. EC-14, 863, (1965).

[5] A. Avizienis, IRE Trans, Electronic Computers, EC-10,389 (1961).

[6] Y. Harata, Y. Nakamura, H. Nagase, M. Takigawa and N. Takagi, Proc. ICCD'84, Oct. 1984.

[7] N. S. Szabo and R. Tanada, Residue Arithmetic and Its Application to Computer Technology. New York: McGraw-Hill, 1967.

[8] H. Henkel, IEEE Trans. ASSP, 37, 301 (1989).

[9] G. L. Sicuranza, IEEE Trans. ASSP, 31, 877 (1983).

[10] T. A. Brubaker and J.C. Becker, IEEE Trans. Comput.,24, 761, (1974).

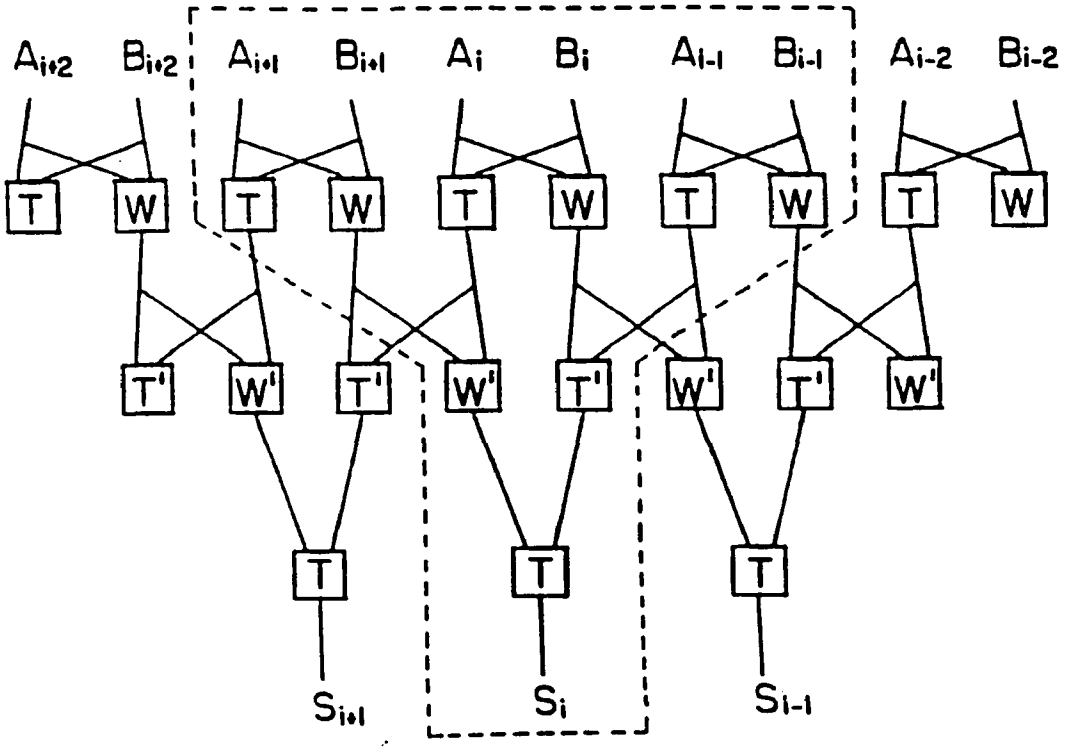


Fig. 2.2.1. 5-bit MSD addition logic flow diagram.

T

	1	0	$\bar{1}$
1	1	1	0
0	1	0	$\bar{1}$
$\bar{1}$	0	$\bar{1}$	$\bar{1}$

(a)

W

	1	0	$\bar{1}$
1	0	$\bar{1}$	0
0	$\bar{1}$	0	1
$\bar{1}$	0	1	0

(b)

T^1

	1	0	$\bar{1}$
1	1	0	0
0	0	0	0
$\bar{1}$	0	0	$\bar{1}$

(c)

W^1

	1	0	$\bar{1}$
1	0	1	0
0	1	0	$\bar{1}$
$\bar{1}$	0	$\bar{1}$	0

(d)

Table 2.2.1. Truth table for MSD (a) T, (b) W, (c) T^1, (d) W^1.

Mantissa length	Truncation error %
4-bit	8.1
5-bit	4.1
6-bit	2.02
7-bit	1.07
8-bit	0.53
9-bit	0.28
10-bit	0.13
11-bit	6.7 X 10-2
12-bit	3.3 X 10-2
13-bit	1.6 X 10-2
14-bit	8.4 X 10-3
15-bit	4.1 X 10-3
16-bit	2.1 X 10-3

Table 2.3.1. Multiplication error as a function of mantissa length.

NUMBER OF BITS	MODULI	RANGE (DECIMAL)	BINARY
3-BIT	5,6,7	210	2^7
4-BIT	11,13,14,15	30,030	2^{14}
5-BIT	17,19,23,29,30,31	200,360,130	2^{27}

Table 2.4.1. Residue number dynamic range for diffrent set of moduli.

III. LOCATION ADDRESSABLE MEMORY-BASED PROCESSING

3.1 Background.

During the past few years, an increased interest in applying optical parallel processing techniques to digital computation has been noted. Methods that are based on optical two-dimensional (2D) spatial light modulators (SLMs), due to their parallel data processing capability, can achieve a high throughput. For different optical logic and arithmetic operations, using either binary or multiple-valued logic , a number of optical truth-table processing schemes have been proposed [1-9]. By applying different optical encoding (i.e. position, density, polarization, etc.) techniques, either location- or content-addressable truth-tables can be implemented. The residue number (RN) system is well suited for both content-addressable (CAM) and location-addressable memory (LAM) look-up processors. In Table 3.1.1a (Table 3.1.1b) 2D truth table for modulo 5 multiplication (addition) is shown. Because with RNs arithmetic operations can be decomposed into a set of independently performed suboperations [1], RNs represent an effective parallel code. To access a memory location, the present position-coded RN (PCRN) look-up table processors utilize a 2D LED (or laser diode) array [5,7,8] (see Fig. 3.1.1). The hologram performs mapping of 2D truth-table elements to 1D result vector. The major disadvantage of this method is that, for a large modulus, a large number of LEDs is required (e.g. for a modulus m, m^2 LEDs are required). Although LAM processors that use smaller

number (for a modulus m, $2m$, or $4\sqrt{m}$) LEDs have been proposed, they require a real-time optical thresholding element not readily available at the present time [8].

In this thesis, a new real-time optical PCRN processor is proposed. The use of a single laser source together with two commercial liquid-crystal TVs (LCTVs) eliminates the complexity of the LED' arrangement. On the two cascaded LCTVs, the operands are represented as two mutually orthogonal light bars. Their intersection at a LCTV output plane represents a particular arithmetic result. For a specific arithmetic operation, the collection of light output points constitute a PCRN look-up table. A hologram, placed at the LCTV's output plane, maps each truth-table point to a proper PCRN result plane position. After presenting the details of this PCRN processor, an experimental PCRN optical multiplier will be described.

An optical implementation of multiple-valued logic (MVL) operators that employs an optical holographic look-up table addressed by a pair of position-encoded liquid-crystal TVs is proposed. This chapter includes realizations of optical modified signed-digit (MSD) as well as Post logic elements.

The MSD number system, originally proposed by Avizienis [10], was first introduced to optical computing community by Drake, et. al. [11]. Using a MSD, various either direct or multi-step optical logic processing schemes have been proposed [11-13]. Recently, to speed up the overall MSD arithmetic operation, an optical carry-free MSD to binary number conversion scheme has also been described [14]. One advantage of a MSD number system is its capability to

perform carry-free arithmetic operations. Another advantage is that its processing complexity is independent of the input's dynamic range. To build an N-bit MSD adder, the MSD logic elements are interconnected in three parallel stages [10-11,13]. Using a MSD adder together with an additional array of MSD logic inverters, a MSD subtraction can be performed. With a number of MSD adders, a fast MSD multiplier has also been proposed [13]. Thus, to perform optical MSD arithmetic, the key component is an optical MSD adder. To date, there are three, i.e. single-, double- and triple-step, MSD addition methods. The fast single- (double-) step method [18-19] uses an array of 6-bit (4-bit) optical content-addressable memory processors that are difficult to implement. On the other hand, the relatively slower triple-step method requires only 2-bit optical location-addressable memory elements. Using this method for an addition, only four different MSD logic elements i.e. W, T, W^1, and T^1, are required (see Table 2.2.1 for their definitions). The three-level interconnect of these four types of logic gates will generate the MSD addition result in a parallel fashion. To implement any of the four elements, in the Drake, et. al.'s approach [11] a cascade of one-dimensional elements (a grating, a prism and a bistable gate) is used. Using symbolic substitution truth-table generation method only two (a recognition and a scription) cascaded processors are required [13]. Next, it will be shown that single step 2D optical truth-table processor is sufficient.

3.2 Look-up processor.

Optical PCRN look-up table calculation is a two stage process. First, a look-up table is generated and stored. Next, during a read-out stage, operands x_i and y_i are used as an address to access an operation result. The optical PCRN look-up table processor requires that, for each x_i column and y_i row intersection pair, a light pulse be produced. In our approach, for each operand an inexpensive LCTV device is employed. The operation of a LCTV is based on the use of a twisted nematic liquid crystal cell [15-18]. First, the input light passes through a polarizer, next through a glass-liquid-crystal-glass cell, and finally, through an analyzer. Because at each LCTV's pixel, in the absence of an applied electric field, the incoming beam's polarization plane rotates by 90°, a parallel analyzer blocks the light at the output. In the presence of an appreciable dc electric field, there is a net polarization change resulting in an analyzer light output. By cascading two LCTVs, a logic AND operation of the two displayed images is performed. On $LCTV_1$, the first operand x_i is a vertical slit which position depends on the value of x_i. When x_i increases, its position is shifted to the right. On $LCTV_2$, the second operand y_i is a horizontal slit which position, for increasing y_i, is shifted downward. When the two LCTVs are transilluminated with a collimated laser beam, the output light represents the desired arithmetic result (see Fig. 3.2.1). To produce the two LCTV images, two computer graphics imaging boards are utilized. At the LCTV's output plane, in a second stage, a hologram-lens combination maps all the output (truth-table) plane points into position-coded result points. During the hologram recording stage, a collimated

input beam is separated by a beamsplitter into a signal and reference arms. With the optical 2D addressing element in the signal arm, the reference arm has an adjustable mirror that tunes the reference beam's direction. By utilizing different reference beam angles, for each result position, a hologram that consists of a 2D array of subholograms where each subhologram represents a particular arithmetic result is produced. During the hologram's read-out process, the light passing through the cascaded LCTVs illuminates specific subhologram. It is then deflected by the hologram-lens combination to one of the PCRN result positions (e.g. for a mod m multiplication there are m, from 0 to m-1, positions).

3.3 Residue number processor

In our experiment, two CASIO (TV 400) 2" diagonal LCTVs were used. To illuminate the entire LCTV screen, the 632.8 nm He-Ne laser beam was expanded and collimated. The set of four relative prime moduli was 3, 4, 5, 7. For a RN processor, the LCTV screens were divided into four sectors, with each sector corresponding to a different modulus LAM look-up table. In Fig.3.3.1, this LCTV's screen partition is depicted. Each multiplication operand was either a horizontal or a vertical slit (see Fig.3.3.2a and 3.3.2b). With the light passing through the common image points, particular operation results (the bright points in Fig.3.3.2c) were generated. For example, the upper-left hand corner point (3,0) represents a multiplication in mod 4, the upper-right hand corner point (2,2) a multiplication in mod 3, the lower-left hand corner point (3,4) a in mod 5 and the lower-right hand corner point (4,4) a multiplication in mod 7.

For each modulus, a lens-hologram combination deflects the beam to a result plane. The holograms were produced using Kodak 649F spectroscopic plates. The collimated input beam was first separated, by a beamsplitter, into a signal and reference arms. While the signal arm contains the LCTVs, the reference beam was directed to various angular directions. As an experimental example, a mod 3 multiplication holographic mapping is shown. The signal and reference beam intensity ratio was 1:3. The angular spacings between the signal and the first reference beam was 20° while the angular separation between two consecutive reference beams was 5°. For a mod 3 multiplication, to fabricate a nine element composite hologram, three different exposures with the three angular reference beams were taken. To generate a set of subholograms, for each angular reference beam, a particular mask was used (see Fig. 3.3.3 (a), (b), (c)). After a chemical processing, the composite hologram was positioned at the correct reference beam position. Since the nine subholograms diffract their input beams into three different output directions, to obtain the three position-coded results, a convex lens was employed. To collect the signal, in the lens back focal plane, either a photographic film or a photodetector can be used. As an example, in Fig.3.3.3d, the composite slightly defocused holographic outputs are shown. In each case, an over 5:1 contrast ratio was observed. For each moduli, a different subhologram array was generated and the four position-coded multiplication results were obtained.

3.4 Optical Position-coded Multiple-value Logic and Arithmetic

3.4.1 Optical MSD adder

To generate an optical MSD single-digit addition truth-table, an optical 2D addressing element followed by a hologram is used. The first variable's three-logic-levels are position-encoded, with the $\bar{1}01$ channels ordered from the top to the bottom, as the horizontal light bar channels, along the vertical direction. Similarly, the second variable is also position-encoded and distributed from the left to the right, as three vertical light bars. When the light passes through the two coded logic variable planes, the multiplication of two orthogonal light bars results, at the overlapped activated part of the two planes, in a light spot. For the nine possible two variable MSD logic combinations, nine position-encoded light spots will be detected. The two optical addressing elements access, upon receiving the two input values, one of the nine intersecting truth-table elements. To fulfill this task, two liquid-crystal TVs (LCTVs) driven by two digital electronic processors are used. Since the nine truth-table positions have to be mapped to the three output positions, an additional output mapping is incorporated. This mapping is performed with a composite hologram. It contains nine subholograms, each recorded at their corresponding truth-table positions. These subholograms diffract the nine LCTV's output dots into three marked output locations. Since a lens maps all identical angle input beams to a common focal point, in our approach, the nine subholograms are divided into three recording

groups. For each recording group, with a mask that selects the required truth-table points, a different angular reference beam is used. For example, to map three truth-table diagonal points to a particular output location, a holographic plate, covered by a spatial mask that blocks the other six truth-table pixels, is exposed to the interference pattern formed by the perpendicular LCTV signal and an off-axis reference beam. The three recorded subholograms, when used for reconstruction, are illuminated by the perpendicular LCTV's signal beams. The three diagonal subholograms will diffract the input beams to a common reference beam direction. By placing a lens after this composite hologram, these three reconstructed reference beams can be focused into a common point. With three different input masks, using three different exposures and three reference beam directions, various MSD holographic mapping elements can be implemented. Thus, with an increase in processing dimensions, optical MSD addition logic elements can be implemented.

3.4.2 Other Optical MVL Operations

Using this 2D optical holographic truth-table method, other two variable optical MVL operations can also be implemented. For example, for fast arithmetic computations, there are methods that use large radix either signed (redundant) or unsigned-number systems [10-11,20-21]. In either case, by transferring and processing more information through each connection per unit time, the use of larger radix number systems leads to an increase in

the logic power and packing density over their binary counterpart. Here, as specific examples, Post MVL operations are described.

Similar to the Boolean OR and AND logic operations, in Post MVL, the so-called maximum (MAX) and minimum (MIN) logic operations are used [20]. For the two input variables x, y, MAX (x,y) = x if x > = y and MIN(x,y) = x if x <= y. In Table 3.4.1 (a) and (b), for a ternary case, the two logic functions are tabulated. The Post logic MAX and MIN functions, together with other unary operators, form a complete logic set. Similar to a binary NAND logic operation, a universal Webb logic element [10] (see Table 3.4.1c) has also been defined. Since these and other two-variable MVL functions can be expressed as truth-tables, for their optical implementations, an identical process can be followed.

3.4.3 Experimental results

To demonstrate the proposed concepts, some preliminary experiments were performed. For the position-encoded MSD truth-table addressing, two cascaded LCTVs (Casio TV 400) were used. Each LCTV was programmed to display a position-encoded either horizontal or vertical slit. To fully utilize the LCTV's screens, each screen was partitioned into eight identical areas representing an 8-bit MSD number (as an example see Fig.3.4.1 (a) and (b)). The passage of the input beam through the two LCTVs results in eight position-encoded optical dots that represent the corresponding holographic memory address code of the MSD arithmetic operational result. In our

experiment, the observed LCTV display contrast was better than 5:1 (see Fig.3.4.1c).

In the holographic plate 1.5 X 1.5 cm^2 aperture, nine subholograms were recorded. To generate the nine element composite hologram, for the three different output entries, three exposures each with a different angular reference beam were taken. For optical MSD addition, after the chemical process, the reference beam illuminated composite hologram was placed at the original position. As examples, in Fig.3.4.2, the photographs of the four input masks and the composite holographic outputs (slightly defocused) for MSD addition gates W, T, W^1 and T^1 are shown. For the Post logic, the top figures (Fig.3.4.3 (a)-(b)) correspond to MAX, and the universal Webb operations, respectively. In each case, the nine truth-table points were divided into three groups and each being recorded with a particular angular reference beam. The bottom pictures of Fig.3.4.3(a)-(b) show the corresponding holographic mapping results. These results were obtained at slightly defocused output plane.

3.5 Summary and conclusions

In the conclusions, limitations of position coded look-up table processor are discussed. For a position coded residue number (PCRN) look-up table processor, the arithmetic operation result is mapped to a given truth-table position. The present commercially available LCTVs are limited to 500 X 500 pixels. The size of this display limits the dynamic range of an RN processor. By increasing

the number of neighboring LCTVs, e.g. using a set of four LCTVs, this limit can be increased to say 1000 X 1000 pixels. In this configuration, each LCTV displays one fourth of the whole look-up table. In addition, for k different residue numbers, each LCTV screen can be sub-divided into k parts, representing a particular modulus PCRN look-up table. For each modulus, a separate deflecting hologram (each consisting of a set of subholograms) must be generated.

For the existing LED/LD look-up processor, the light source complexity rapidly increases with the RN's dynamic range. On the other hand, the proposed LCTV-based processor requires, regardless of the RN's dynamic range, only a single light source. To increase the speed of this PCRN processor, a surface stabilized ferroelectric LC (SSFLC) plate may be used. It has been indicated [20] that a 1000 X 1000 SSFLC gate array (assuming $4\mu^2/gate$) will operate at 1 MHz frame rate.

This processor's physical dimensions and its addressing speed (milliseconds) are not competitive with a corresponding electronic counterpart. However, for MSD mumber-based lookup table processor, since a high resolution LCTV has 500 X 500 pixels, it is possible to partition these LCTV's into an array of identical 80 X 80 addressing cells each containing 6 X 6 (using a guard band between two addressing pixels) light pixels. Thus, for each processing frame (33 ms), a pair of 6400-bit MSD numbers can be processed. When a large array of such LCTV's are used, the processing throughput improves. To acquire a higher processing rate, other faster 2D modulators, such as optical bistable, optical phase conjugation, optical second-harmonic generation, must be used. To summarize, the use of transmission type LCTVs for a PCRN look-up table processor

has been described. An experimental four moduli optical PCRN multiplier processor has been presented. In our experiment, a slightly modified inexpensive LCTVs, (CASIO TV 400) were used. The proposed processor's dynamic range is limited by the number of available LCTV pixels. By combining a number of separate LCTVs, a high dynamic range optical multiplication can be achieved. To map the look-up truth-table results into PCRN output channels, holographic beam directors were used. An optical two-variable MVL hybrid processing scheme has been proposed. To implement a particular optical MVL operation, a holographic optical location-addressable truth-table look-up method has been used. The proposed device contains a 2D position-encoded either E-O or all-optical addressing element followed by a composite hologram. Examples of implementing optical MSD, Post as well as Webb logic elements were given. Preliminary experimental results were also presented.

3.6 References

[1] N. S. Szabo and R. Tanada, Residue Arithmetic and Its Application to Computer Technology (McGraw-Hill, New York, 1967).

[2] A. Huang, Y. Tsunida, J. Goodman and S. Ishihara, "Optical computation using residue arithmetic," Appl. Opt. 18, 149 (1979).

[3] A. M. Tai, I. Cindrich, J. R. Fienup, and C. C. Alexsoff, "Optical residue arithmetic computer with programmable computation modules," Appl. Opt. 18, 2182 (1979).

[4] C. C. Guest, and T. K. Gaylord, "Truth-table look-up optical processing utilizing binary and residue arithmetic," Appl. Opt. 19, 1201 (1980).

[5] A. P. Goutzoulis, D. K. Davis and E. C. Malarkey, "Prototype position-coded look-up table using laser diodes," Opt. Comm. 61, 302 (1987).

[6] M. M. Mirsalehi, and T. K. Gaylord, "Truth-table look-up parallel data processing using content-addressable memory," Appl. Opt. 25, 2277 (1986).

[7] R. I. McDonald, "Optoelectronic switch matrix as look-up table for residue arithmetic," Opt. Lett. 12, 787 (1987).

[8] A. P. Goutzoulis, "Complexity of residue position-coded lookup table array processor," Appl. Opt. 26, 4823 (1987).

[9] Y. Li, G. Eichmann, R. Dorsinville and R. R. Alfano, "Demonstration of a picosecond optical residue computation," Opt. Lett. 13, 178 (1988).

[10] A. Avizienis, IRE Trans. Electronic Computers, EC-10, 389 (1961).

[11] B. L. Drake, R. P. Bocker, M. E. Lasher, R. H. Patterson and W. J. Miceli, Opt. Eng. 25, 38 (1986).

[12] Y. Li and G. Eichmann, Appl. Opt. 26, 2328 (1987).

[13] K. Hwang and A. Louri, "New Symbolic Substitution Algorithms for Optical Arithmetic using Signed-Digit Representation," Proc. SPIE, 880 (1988).

[14] Y. Li, J. Zhu and G. Eichmann, "Optical On-the-fly Conversion of a Modified signed-digit (MSD) into Two's Complement Binary (TCB) Number Representation," Opt. Lett. 13, 294 (1988).

[15] F. T. S. Yu, S. Jutamulia and D. A. Gregory, "Optical parallel logic gates using an inexpensive liquid-crystal TV," Opt. Lett. 12, 1050 (1987).

[16] F. T. S. Yu, S. Jutamulia, and D. A. Gregory, "Real-time liquid crystal TV XOR- and XNOR-gate binary image subtraction technique," Appl. Opt. 26, 2738 (1987).

[17] K. D. Hughes, S. K. Rogers, J. P. Mills, and M. Kabrisky, "Optical preprocessing using liquid crystal televisions," Appl. Opt. 26, 1042 (1987).

[18] H. K. Liu, J. A. Davis, and R. A. Lilly, "Optical data processing properties of a liquid-crystal TV spatial light modulator," Opt. Lett. 10, 635 (1985).

[19] K. M. Johnson, M. R. Surette, and T. Shamir, "Optical interconnection network using polarization-based ferroelectric liquid crystal gates," Appl. Opt. 27, 1727 (1988).

[20] T. T. Dao, E. J. McCluskey, and L. K. Russell, Opt. Eng. 25, 14 (1986).

[21] G. Eichmann, Y. Li, and R. R. Alfano, Appl. Opt. 25, 3113 (1986).

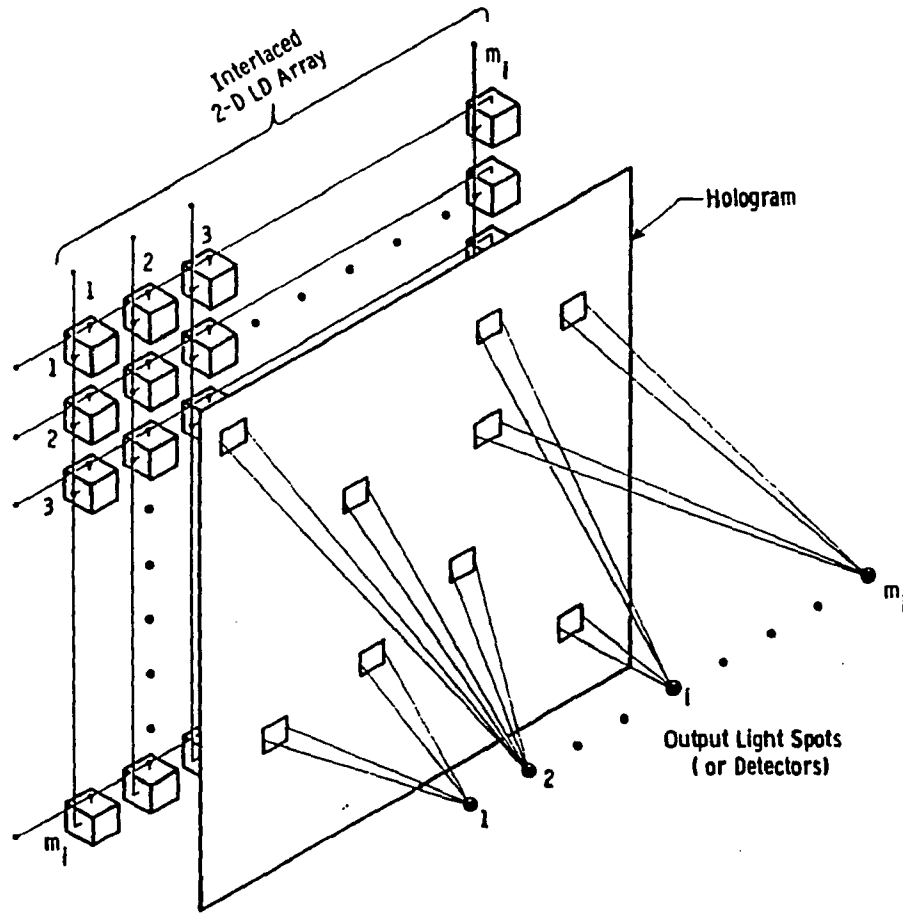


Fig. 3.1.1.Optical position coded lookup table processor architecture.

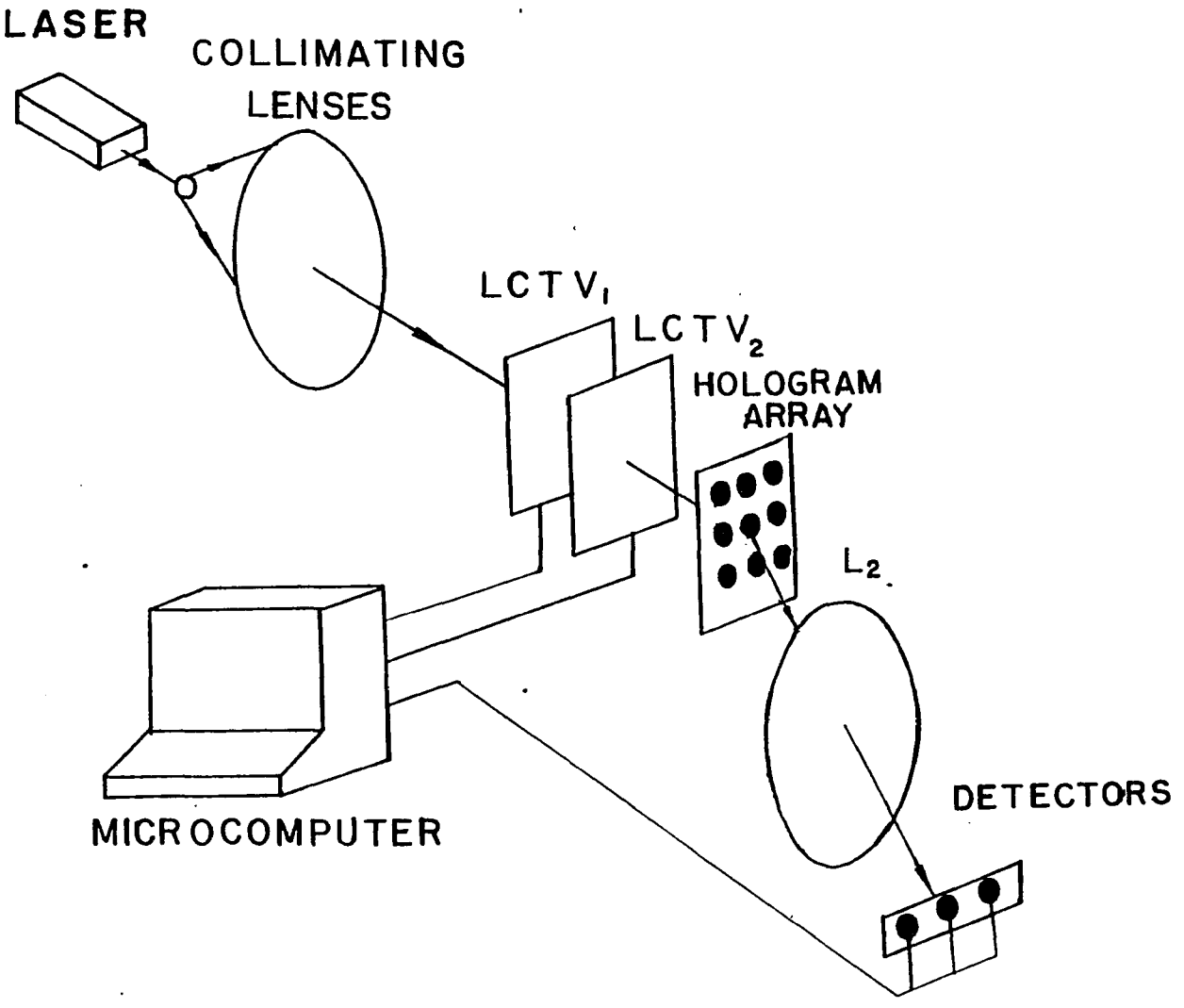


Fig. 3.2.1 Experimental setup for a single laser source optical position coded processor.

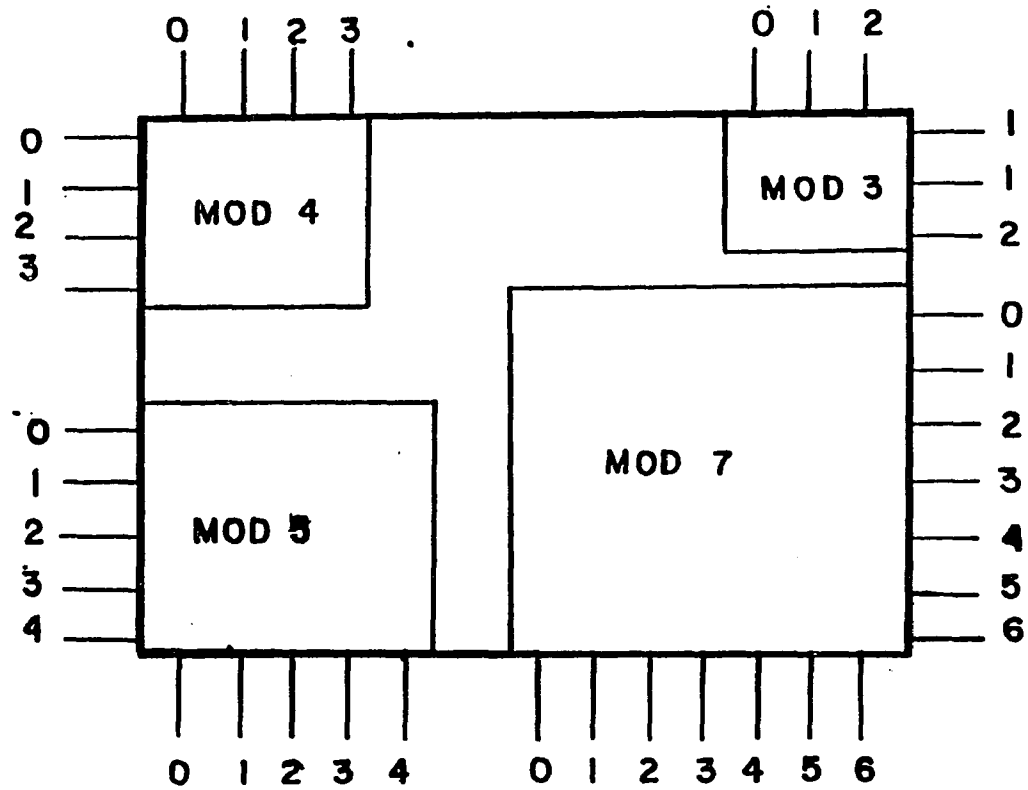


Fig. 3.3.1. LCTV screen partition for 4-digit RN processor.

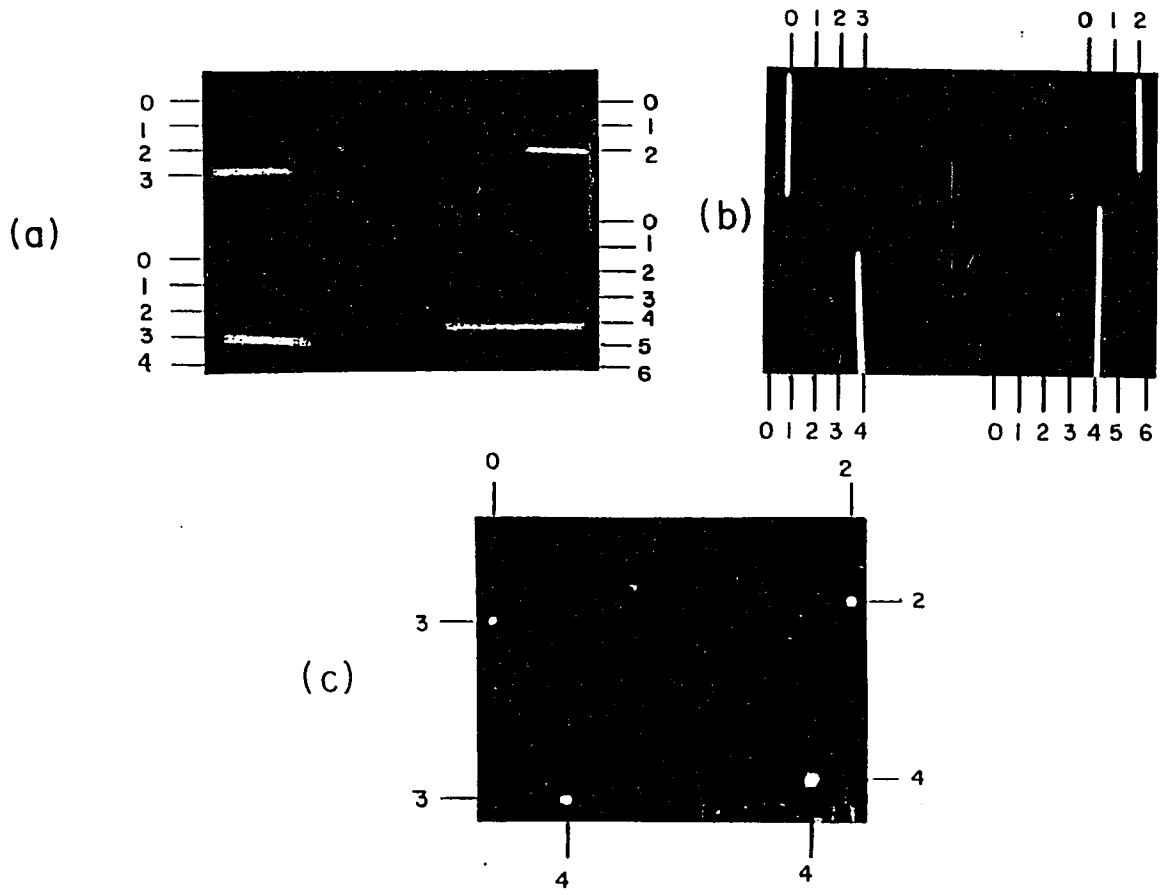


Fig. 3.3.2 Addressing of an experimental PCRN processor. (a) the four horizontal bars (the first operand) represent numbers 2, 3, 3, 4 for mod 4,3,5,7 multiplications, respectively; (b) the four vertical bars (the second operand) represent numbers 2, 0, 4, 4 for the moduli 4, 3, 5, 7, respectively: (c) the logic AND operation performed on the images of Fig. (a) and (b).

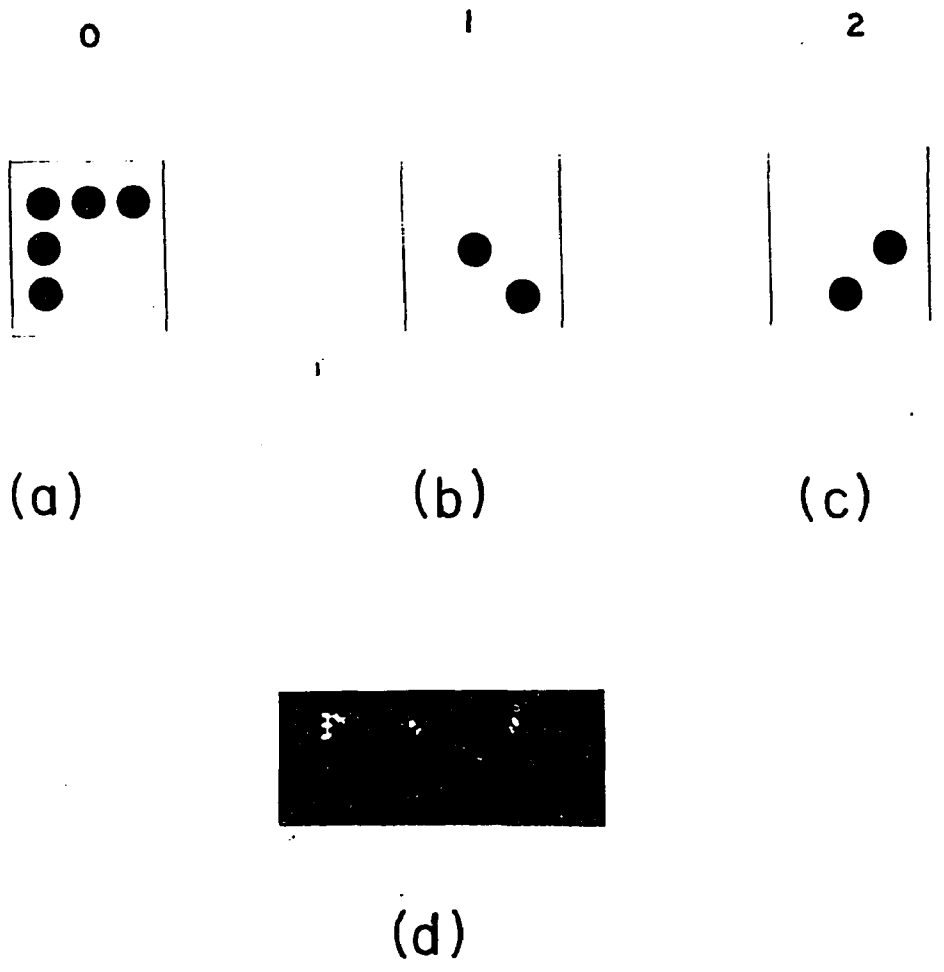


Fig. 3.3.3 The 3X3 hologram used to map the PCRN mod 3 multiplication truth table entries. In (a), (b) and (c), the hologram represents the PCRN level 0, 1 and 2, respectively, while in (d) the holographic mapping result at the processor's slightly defocused output plane is shown.

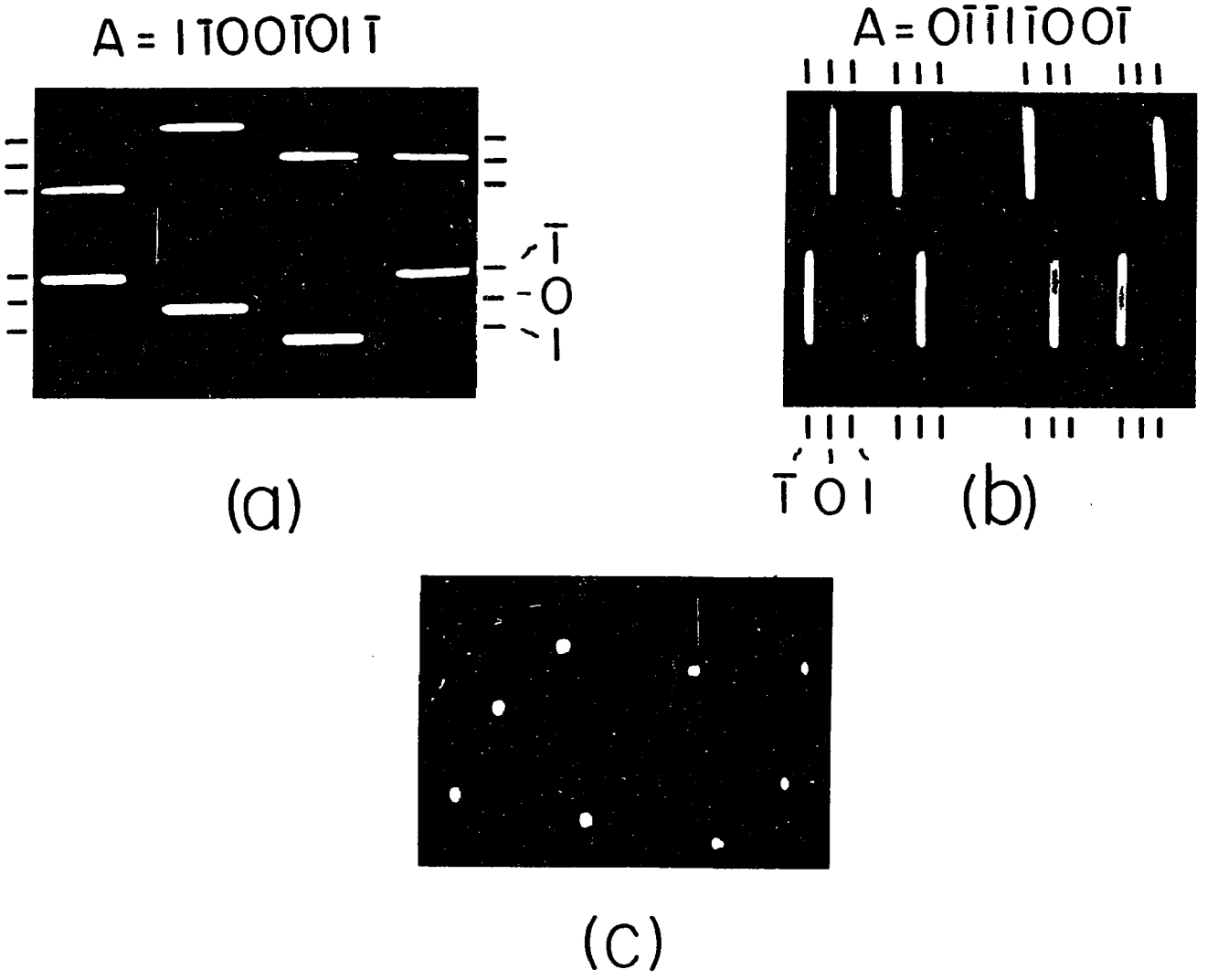


Fig. 3.4.1. The experimental result for MSD truth table addressing; (a), (b) The two LCTV screens are partitioned to display two position-encoded 8-bit MSD numbers. (c) The overlapping result of (a) and (b).

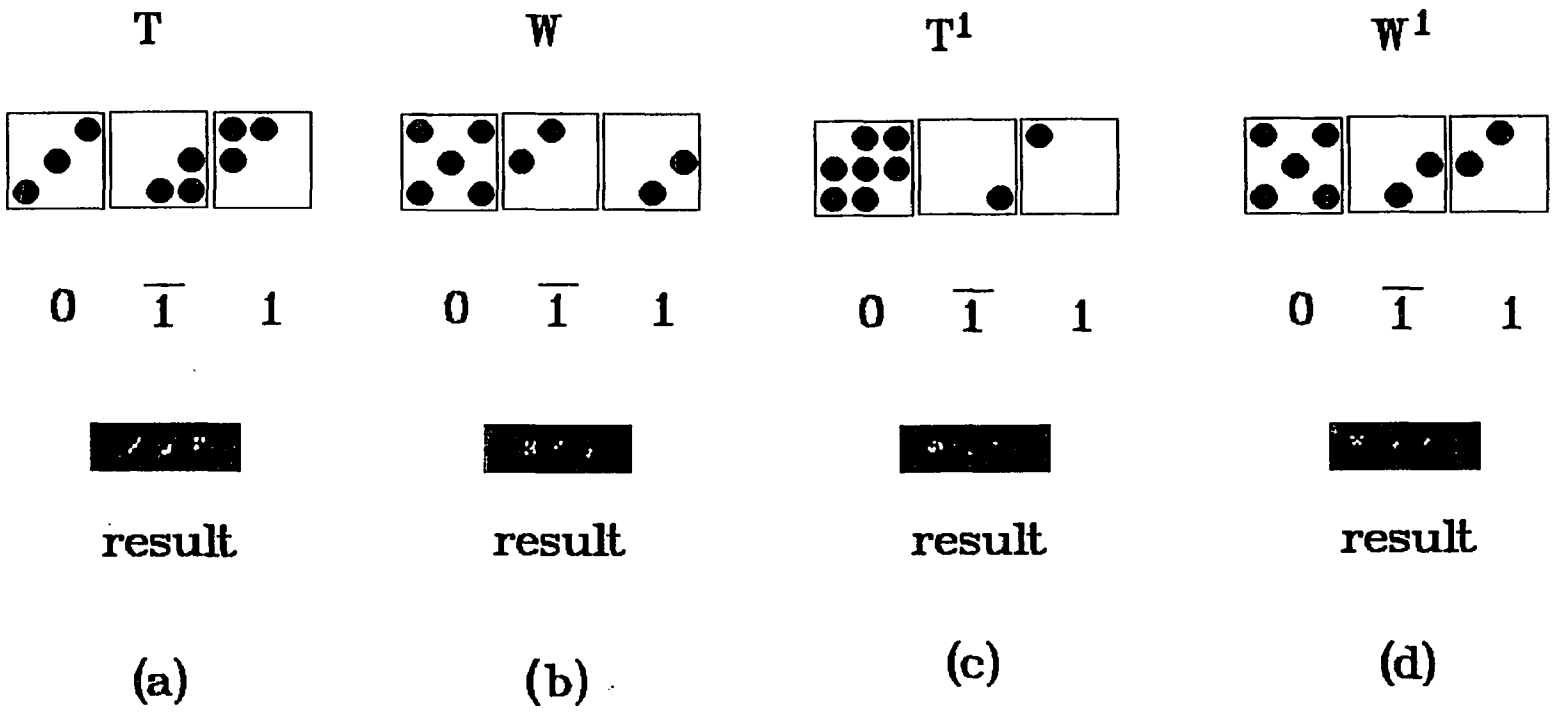


Fig. 3.4.2. The experimental holographic mapping for MSD gates, (a) W, (b) T, (c) W^1 and (d) T^1. Each of the nine subholograms is divided into three groups that are recorded with three exposures. With a lens, the mapping results are taken from a slightly defocused plane.

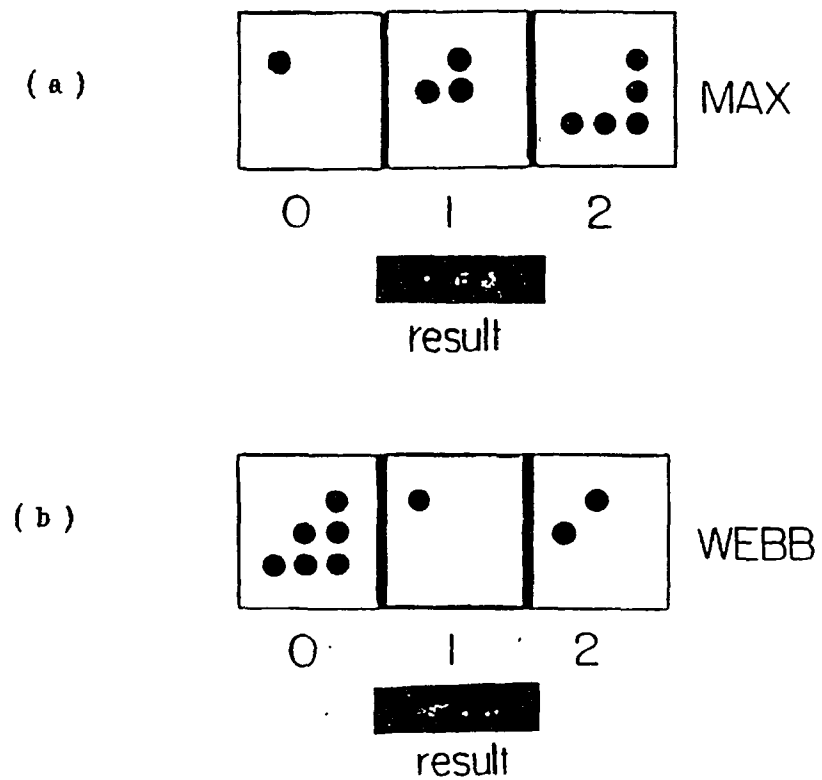


Fig. 3.4.3. Holographic mapping for Post logic gates (a) MAX and (b) WEBB

	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

(a) multipication

	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

(b) addition

Table 3.1.1. Truth table for mod 5 (a) multiplication, (b) addition.

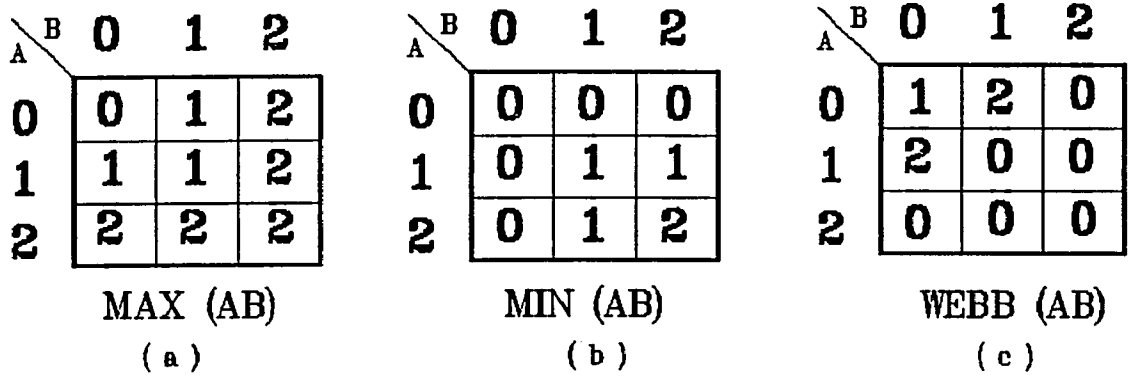


Table 3.4.1. Truth table for the ternary two-variable Post algebra (a) Maximum (MAX), (b) Minimum (MIN) and (c) universal Webb logic gates.

IV. CONTENT ADDRESSABLE MEMORY-BASED PROCESSING.

4.1 Background

The parallelism and interconnectivity of optical systems constitute a major advantage in massively parallel optical processing schemes. The need for high-speed data processing has brought a wide interest in memory-based Boolean functions implementation. Since any arithmetic-logic operation can be represented as a set of Boolean functions (expressed as a sum of product terms), by storing these product terms in a content addressable memory (CAM) fast optical arithmetic-logic processor can be implemented.

Optical content addressable memory (CAM) has been extensively studied for optical signal processing applications. Features such as memory efficiency (compared to LAM), fast processing speed and design flexibility attract the attention of the optical community. Holographic CAM-based processing applied for optical computing was introduced by T. Gaylord and et. al [1-3]. Here, a new concept of non-holographic CAM used to design a number of different arithmetic-logic processors is presented. This has a number of advantages over a holographic implementation, including (1) ease of alignment, i.e. no phase-matching between recording and reconstruction involved, (2) no zero-order diffraction loss and (3) no crosstalk.

Using non-holographic CAM-based processing, logic functions are encoded using a set of spatial symbols. For *a* m-level logic function, m orthogonal spatial

symbols are required. Thus, a spatially encoded logic function is represented by a matrix of spatial symbols where each product term is encoded as one column. To evaluate the logic function for a specific input sequence, the CAM mask is illuminated by a reconfigurable set of light sources. The transmitted light is summed along each column, detected and thresholded. Depending on the spatial encoding method, the output result can be active high or active low. In the first case, the maximum transmitted light indicates result "logic one", while in the second case, minimum light intensity corresponds to result "logic one". The spatial symbols used in active the high approach are negative of spatial symbol used in the active low case.

In this chapter, multilevel as well as binary variables are considered. Application of CAM for arithmetic-logic processors utilizing binary, residue, sign/log, and modified signed-digit number systems are presented. To successfully implement arithmetic-logic processors for non-binary number systems, forward and backward conversions between binary and other number systems are addressed.

The application of CAM is extended to so called higher-order symbolic substitution (SS), one of the frequently mentioned optical parallel computing techniques. Symbolic substitution, a powerful optical parallel digital computing, is a pattern replacement operation. The basic SS is a two step method: (1) the recognition of a particular pattern in the input data array, and (2) its replacement with another pattern according to a given substitution rule. The original SS binary addition rule was described by Huang [4] but other several-bit SS, also called higher-order SS, rules were suggested by Kozaitis [5]. Using dual-rail

(DR) encoding scheme (see Fig.4.1.1a), sixteen higher order substitution rules for 2-bit addition are shown in Fig.4.1.1b. In the case of conventional SS for binary numbers, two bits can be processed at the same time, implying n-stage computation for n-bit numbers. Using higher-order SS, more bits can be processed, which provides higher computational speed. For longer than 2-bit numbers, e.g. 4-bit numbers, applying some rules of Fig. 4.1.1b, addition result can be obtained in two steps (see Fig. 4.1.1c for addition of two binary numbers 0011 and 1010). Applying SS rules incorporating larger number of bits, further speed increase can be achieved. Applying higher-order SS rules, multi-variable logical functions can be calculated in a one step. Using SS, various image processing techniques, such as skeletonization, morphological processing, median filtering, syntactic pattern recognition, and local image processing, have also been suggested [6-10]. Here, higher-order SS can provide recognition and substitution of larger window elements, reducing number of steps required in case of conventional SS. In this thesis, symbol recognition based on CAM is presented and experimentally implemented.

4.2 Multilevel spatial encoding

To successfully construct a CAM-based processor (1) a logic function must be implemented using a CAM mask, and (2) the encoded function must be evaluated for different input sequences. For a logic function represented as a sum of product terms, the CAM mask is a fixed binary mask, where each mask column represents a different product term. In many cases, multilevel variables

are need to be encoded to perform an arithmetic-logic operations.

To encode logic variables either active low or high logic can be applied. For a high (low) level logic and for an m-level position coded variables only one element of the spatial symbol transmit (blocks) the incident light. To encode a "total don't care" ("A_x") element, for an active high (low) logic, an all pass (block) spatial symbol is used. Thus the spatial symbols for active high logic are the negative of spatial symbols for active low logic. In Fig. 4.2.1 a-d (4.2.2 a-d), for active high (low) logic, a set of spatial multi-rail position coded symbols, for 5-, 4-, 3- and 2-level variables is shown. To encode a product of logic variables the spatial symbols are aligned into one column. Fig. 4.2.3a (b) represents spatial active high (low) encoding of a product $a_1 b_3 c_0$ using 4-level variables. For an n-variable product term (where each logic variable assumes levels between 0 and m-1), the total number of elements K in one column is equal to

$$K = mn. \qquad (3.2.1)$$

For a function consisting of p-product terms, p-columns are included in the CAM mask and the total number of spatial symbols N_T incorporated into the CAM mask is equal to

$$N_T = Kp = nmp \qquad (3.2.2)$$

In Fig. 4.2.4a (b) active high (low) spatial encoding of a sum of three product

terms for 3-level logic variables

$$F(a,b,c) = b_0 c_3 + a_1 b_1 c_2 \qquad (3.2.3)$$

is shown.

To evaluate the logic function for a particular input sequence, the CAM mask is illuminated by a set of line sources. With each m-level logic variable, m light sources are associated, where one light source illuminates only one CAM mask's row.

Consider as a simple multiplication exampl, two decimal numbers A and B, each in the range from 0 to 3. In Table 4.2.1, the minimized truth table for the multiplication result is given. To represent possible multiplication results, seven functions F_0, F_1, F_2, F_3, F_4, F_6, F_9 corresponding to 0, 1, 2, 3, 4, 6, 9 output results, respectively, are required. In Fig. 4.2.5a (b), using position coded representation, the CAM mask corresponding to the functions F_i (i=0, 1, 2, 3, 4, 6, 9) for high (low) level logic case is shown. To perform a particular multiplication e.g. 2 X 3, row number 2 corresponding to the variable A and row number 3 corresponding to the variable B are illuminated. In Fig. 4.2.5c (d), for the active high (low) logic, transmitted light intensity is shown. To obtain the multiplication result, a high (low) level threshold operation is performed. For high (low) level logic, the maximum (minimum) transmitted light intensity represents the result. The number of functions F_i is equal to the number of all possible different output results. For a high dynamic range input numbers, a

large number of output functions is required. Using a binary encoded representation, the number of logic functions is reduced. In this case, a multilevel variable is substituted with a corresponding binary representation.

4.2.1 Spatial encoding for MSD based processor

MSD addition requires the encoding of 3-level logic variables, where each variable can assume any of the three values $1, 0, \bar{1}$. In this thesis, active low logic is applied for MSD variables encoding. In Fig. 4.2.6. a, b, c spatial symbols representing $1, 0, \bar{1}$, respectively, are shown. Since, one output digit is a function of six input variables, each of the six input variables is position-encoded into three spatial channels each of which represents one of the three logic levels $1, 0, \bar{1}$. These six variables span a 1D distribution of 18 channels (pixels). For a multi value logic (MVL), encoding of "total don't-care" (A_x) and "partial don't care" (A_{yz}), where x,y can be $10, 1\bar{1}, \bar{1}0$, is needed (see Fig. 4.2.6. D-g). As an example consider encoding of the reduced six-variable reference pattern $A_{1\bar{1}}^{n} A_1^{n-1} A_x^{n-2} B_0^{n} B_1^{n-1} B_{10}^{n-2}$. In particular, an 18 pixel CAM mask is designed to block the transmission of all possible light appearance positions corresponding to all the possible input combinations. For example, for the first variable $A_{1\bar{1}}^{n}$, the possible light appearance will be in the top and bottom channel pixels. Thus, the CAM mask for this digit is designed to block these two positions. The mask for the digits A_1 and A_x will block the light at the top and all the three positions, respectively (see Fig. 4.2.6h for encoding of the whole product term).

In section 4.3.2.1, it will be shown that to obtain a single output, 56 product terms are required to be stored into a CAM mask. With the design of this mask, the CAM write-process is completed. The mask is then placed in front of a lens. For a CAM readout, any of the twelve input light patterns corresponding to the stored compact reference pattern will not be transmitted and a dark output will be observed. The thus detected "zero" intensity is thresholded and for a bright-true logic matching result, electronically inverted. On the other hand, input light combinations that do not match the coded reference pattern positions will produce some residue light. This residue signal, after a threshold detection followed by a logic inversion, will yield a true "zero" indicating a pattern mismatch. Using this scheme for a multi-bit addition, a parallel 1D (horizontal) extension is possible. In addition, since a multiple-digit MSD adder is a parallel extension of a single-digit MSD adder, the MSD algorithm lends itself to be implemented as a multiplexed CAM.

4.3 Binary function encoding

With binary logic, a logic function can be spatially encoded using either an active low or high level logic. Using a dual-rail (DR) CAM mask encoding scheme (see Fig. 4.3.1), the input data is entered via two light sources. By activating the lower (upper) light source, a "1" ("0") is introduced. To encode a logic variable A into a CAM mask, a DR spatial symbol of Fig. 4.3.1a is used. Using the spatial symbol of Fig. 4.3.1b, its complement $\overline{A}$ is encoded. For active

high (low) logic, different "don't care" element ("x") encoding is required. For active high (low) logic, an "x" is encoded using the spatial symbol of Fig. 4.3.1c (d). As an example of an active high encoding scheme, assuming a 3-variable product term $ab\bar{c}$ sum of variables is encoded into a CAM mask

$$F = a + b + \bar{c}, \qquad (4.3.1)$$

using the spatial symbol of Fig. 4.3.1a, the variables a and b are encoded, while using a symbol of Fig. 4.3.1b, the variable $\bar{c}$ is encoded. In Fig. 4.3.2a, the spatially encoded CAM mask representing the term $ab\bar{c}$ is shown. To establish the product term logic value for a particular input sequence, the mask of Fig. 4.3.2a is illuminated by three pairs of light sources. But only one source from each pair is activated. This transillumination intensity pattern represents the logic product of the input source pattern and the CAM mask. If a particular input variable is equal to one (zero), the lower (upper) element of the spatial symbol is illuminated. For example, to evaluate the product term for the input a=1, b=0 and c=0, the elements 2, 3 and 5 are illuminated (see Fig. 4.3.2b). The light transmitted through the CAM mask is summed i.e.

$$F(1,0,0) = 1 x a + 0 x b + 0 x \bar{c}. \qquad (4.3.2)$$

where x represents the product between the input light pattern and the CAM

mask. To establish the logic value of the product term, the light intensity transmitted through the mask is thresholded. The transmitted intensity corresponding to a single variable e.g. a is equal to 1 for the cases $1xa, 0x\bar{a}$ and is equal to zero for $1x\bar{a}, 0xa$. Since for a three variable product term the sum of the transmitted light intensity can assume any one of the four levels (0, 1, 2, 3), by setting the threshold between intensity levels 2 and 3, the final result is obtained. The intensity above (below) the threshold level indicates that the product term equals one (zero). The intensities above the threshold are obtained for a=1, b=1, c=0 (see Fig. 4.3.2c). For the output logic function F, expressed as a sum of p products F_i (i=1, 2,.. p), the threshold logic operation can be expressed

$$F_{t\ \ high} = \sum_{i=1}^{p} T_h(F_i), \qquad (4.3.3)$$

where $\sum_{i=1}^{p}$ denote a p-variable binary OR and T_h represents a threshold operation at a high intensity level. In this case, each product term is encoded as a separate column of the CAM mask. Using an active high logic, a logical function (sum of products) is implemented by logically ORing the thresholded outputs. If at least one column intensity is above the threshold, the output function is equal to a one, i.e.

$$F = \begin{Bmatrix} 1 & if & \exists i(T_h(P_i) = 1) \\ 0 & if & \forall i(T_h(P_i) = 0) \end{Bmatrix}. \qquad (4.3.4)$$

On the other hand, for an active low logic, the sum of the complemented variables

$$F = \overline{a} + \overline{b} + c \qquad (4.3.5)$$

is encoded into the CAM mask (see Fig. 4.3.2d). Using an identical (to active high case) light source arrangement, the transillumination shown on Fig. 4.3.2e is represented as

$$F(1,0,0) = 1x\overline{a} + 0x\overline{b} + 0\overline{x}. \qquad (4.3.6).$$

By thresholding the output intensity at a low (between levels 0 and 1) intensity level, for an intensity below (above) threshold, the logical value of one (zero) is obtained. Since for the active low logic, the result is indicated by the lack of output light, to obtain an active high result, an inversion of the output intensity is required. If the logic function F is expressed as a sum of p products F_i, the threshold operation can be expressed

$$F_{t\ low} = \sum_{i=1}^{p} \overline{T_t(F_i)}, \qquad (4.3.7)$$

where T_t is the threshold operation performed at a low intensity level. If at least one output column intensity is below the threshold, the logic function is

equal to a one, i.e.

$$F = \begin{Bmatrix} 1 & if & \exists i(T_h(P_i) = 0) \\ 0 & if & \forall i(T_h(P_i) = 1) \end{Bmatrix}. \qquad (4.3.8)$$

As an example, consider the multiplication of two 2-bit BNs $A=A_0A_1$ and $B=B_1B_0$. The 4-bit multiplication result is represented by the four logic functions F_3, F_2, F_1, F_0. The minimized functions are

$$F_0 = A_0B_0, \qquad (4.3.9a)$$

$$F_1 = A_1\overline{A}_0B_0 + A_1B_0\overline{B}_1 + A_0\overline{A}_1B_1 + A_0\overline{B}_0B_1, \qquad (4.3.9b)$$

$$F_2 = \overline{A}_0A_1B_1 + A_1\overline{B}_0B_1, \qquad (4.3.9c)$$

$$F_3 = A_1A_0B_1B_0. \qquad (4.3.9d)$$

For each output logic function, each product term is encoded as a column in the CAM mask. Using DR spatial symbols, in Fig.4.3.3a, the corresponding CAM masks are shown. To evaluate the F_i functions for a specific input sequence, the corresponding CAM mask rows are illuminated. For example, for the inputs $A_0 = 0$, $A_1 = 1$, $B_0 = 1$, $B_1 = 1$, the rows 1, 4, 6 and 8 are illuminated. Along each column, the light transmitted through the mask is integrated. In Fig. 4.3.3b, the intensity integration result is shown. The highest intensity was detected for the

functions F_1 and F_2. If at least one total column intensity is above the threshold level, the logic function is equal to one, otherwise the result is a zero.

Assuming an n-variable product, the maximum transmitted light intensity through CAM column is equal to nI_t (where I_t represents light intensity transmitted through one DR spatial symbol). The light passing through the CAM mask is summed along each column, detected and thresholded. By setting the threshold level at

$$T_h = \left(n - \frac{1}{2}\right) I_t, \qquad (4.3.10)$$

intensity maxima are detected. In Fig. 4.3.3b, where the maximum transmitted intensity is $4I_t$, the threshold level is set at

$$T_h = \frac{7 I_t}{2}. \qquad (4.3.11)$$

For multiplication example of 2 X 3 (see Fig. 4.3.3b), the intensities corresponding to functions F_1 and F_2 are above the threshold thereby generating the result 0110 (F_3=0, F_2=1, F_1=1, F_0=0). In Fig. 4.3.3c, the CAM mask for the active low logic is shown. Here, the lowest transmitted intensity, below a single channel intensity, indicates that the value of the logic function is a one. In Fig. 4.3.3d, the transmitted intensity distribution corresponding to 2 X 3 is shown. For the

functions F_2 and F_1, the lowest intensity was detected. To establish a logic value, the output result is thresholded at a low intensity level and the result F_3=1, F_2=0, F_1=0 and F_0=1 was obtained. For a low level logic, regardless of the number of variables in a logic function, the threshold level is set to

$$T_l = \frac{I_t}{2}. \qquad (4.3.12)$$

4.4 Active low logic, active high logic comparison.

In this chapter, some differences between active high and active low approaches are presented. In the low level approach, an inversion step is necessary, while in the case of the active high logic, the inversion step is not required. These two approaches require detectors arrays with different characteristics. In case of the low level logic, the detector's dynamic range can be relatively low (only two levels are required). For the high level logic, the dynamic range has to be increased to accommodate the larger number of different intensity levels. For an n-variable function, the dynamic range of at least n levels is required. This requirement can be easily satisfied considering detectors currently available on the market with dynamic range ~50db.

An additional difference between the active low and active high approaches concerns the uniformity of light distribution for each channel. The active high approach requires each channel to produce light with identical intensities. To implement n-variable logical function, n light sources are activated. Due to

temperature changes, power fluctuation and difference in light source characteristics, each source produces a light beam with a slightly different intensity $I_t \pm \epsilon$, where ϵ is the maximum intensity deviation. For the active high logic, the threshold level should be set according to eq. (4.3.10). Due to intensity integration at the detector plane, the total intensity can be expressed as

$$I_T = nI_t + \sum_{i=1}^{n} A_i \epsilon \qquad (4.4.1)$$

where A_i can assume any value from interval (-1, 1) and n is the number of variables. Consider the worst case when A_i=-1 for i=1 to n. In order to detect output symbols, the following condition has to be satisfied

$$T_h = \left(n - \frac{1}{2}\right) I_t < nI_t - n\epsilon \qquad (4.4.2)$$

giving the condition for the maximum intensity deviation ϵ

$$\epsilon < \frac{I_t}{2n} \qquad (4.4.3)$$

The last equation indicates that for larger number of channels better LD's uniformity is required.

For the active low logic the threshold is set at low level 1/2 I_t. The worst

case is if one LD is activated and produces intensity below the threshold level. To properly detect the output logic value, the following condition must be satisfied

$$T_t = \frac{I_t}{2} > I_t - \epsilon \qquad (4.4.4)$$

The condition for maximum intensity deviation ϵ is then

$$\epsilon < \frac{I_t}{2} \qquad (4.4.5)$$

The intensity deviation for active low logic can be n times larger then for the active high logic, so better quality light sources are required for the active high case.

4.5 Optical arithmetic CAM processor architecture

4.5.1 Single operation processor.

In Fig. 4.5.1, an optical architecture for an m-bit CAM processor is shown. Let the two m-bit input BNs be $a=a_0a_1 \ldots a_{m-1}$ and $b=b_0b_1 \ldots b_{m-1}$. The CAM masks $M_0, \ldots M_{n-1}$ correspond to the output bits $s_0, \ldots s_{n-1}$. Each CAM mask is illuminated by an array of laser diodes (LDs). Because of the DR encoding two LDs are required to represent each binary variable. When the

input logic variable is a one (zero), the even (odd) row corresponding to the variable is illuminated. To provide a line source illumination, where each line source illuminates a single CAM mask row, an anamorphic optical system is employed. Assuming an n-variable product term, the maximum light intensity transmitted through a single CAM mask column is equal to nI_t, where I_t is the light intensity transmitted through a spatial symbol representing either a "0" or "1". To obtain the final logic result, the summed along columns intensity is detected and thresholded. If a single summed output intensity is above a threshold level, an output logic one, otherwise a logic zero is generated. For an n-variable logic function (active high logic), the detector's dynamic range of at least n levels is required. The threshold operation is performed electronically using an array of comparators. For this purpose, high speed operational amplifiers, such as National Semiconductor LH0032 or LH0033, can be used. An identical reference voltage V_{Th}, corresponding to threshold intensity level T_h is applied to all comparators. To obtain the final logic output functions, i.e. $s_0,\ldots s_{n-1}$, the outputs of the corresponding comparators must be OR-ed. If a single comparator generates an output signal, the corresponding logic output function is a one. For an active low logic, the threshold level is set at $1/2\ I_t$ and electronic inverters are placed at the output plane (see dashed line in Fig.4.5.1). The inverter outputs are connected to an array of OR gates where each OR gate output corresponds to a different output function.

To eliminate the electronic threshold operation, an optical bistable device such as a self-electro-optic-effect device (SEED) or a nonlinear

Fabry-Perot etalon can be applied [11-12]. In this configuration, using an active high logic, after the threshold, the OR operation can be performed using an array of cylindrical lenses. Each cylindrical lens' aperture must be equal to the aperture of the corresponding CAM submask. At the detector plane, the final logic result is obtained. The major advantage of the optical threshold is a reduction of required number of photodetectors (in this case, the number of photodetectors is equal to the number of output bits) and the ease in implementing the output OR gates. The single-operation processor using optical threshold is shown in Fig. 4.5.2.

The maximum number of product terms that can be stored in a CAM mask, using a laser diode (LD) with a beam divergence half-angle α and with the distance between LD and CAM mask equal to b, is

$$N_p = \frac{l}{p} = \frac{2b \operatorname{tg} \alpha}{p} \qquad (4.5.1)$$

where l is CAM mask dimension and p is its smallest mask aperture size (also referred to as a CAM pixel). If the diffraction-limited mask aperture p is $2\mu m$, and the dimension of the mask l is 130mm, the use of a typical LD's α as 30°, yields the total number of encoded product terms N_p ~66 X 10^4, which corresponds to a 16-bit processor. In comparison, the holographic CAM, although theoretically able to deliver larger then 66 X 10^4 storage capacity, the experimentally demonstrated hologram can store up to 500 single-pixel

reference patterns with low diffraction efficiency of 0.01% [13]. In the case of multi-pixel patterns, requiring 3-step recording process, this number will be drastically reduced.

4.5.2 Multi-operation processor.

To implement a multi-operation CAM based processor various CAM masks are placed on the mask plane. Using an electronically addressable SLM, different masks can be displayed. A major disadvantage of this approach, due to the SLM's limitations, is its slow reconfiguration speed. By replacing the reconfigurable mask, with a set of fixed masks, where each mask corresponds to a single operation, higher processing speed can be achieved. In Fig. 4.5.3, an architecture of a multioperation processor based on CAM is shown. To execute a particular operation, only one fixed mask is illuminated. For a CAM processor capable of executing k operations, k LD groups are needed, where each group of LDs is associated with a single CAM mask. The number of LDs in each group depends on the number of bits to be processed. For an input data multiplexing of a k-function processor, a fast decoder with $K = \log_2 k$ operation select lines is used. For an active high decoder, only one output line is at logic "one", which provides the enable signal for only one group of LDs. The operation select lines are connected to an operation sequencer which contains a sequence of instructions to be executed. In Fig. 4.5.3, each pair of LDs is controlled by the status of the data-out bus. If a particular bit in the data word is 1 (0), the lower (higher) LD from a

corresponding pair is activated. Two AND gates associated with each pair of LDs constitute a simple 2:1 demultiplexer. Using anamorphic optical system, each CAM mask is illuminated by a line sources. To integrate transmitted light intensities along all CAM mask columns, the detector can be superimposed with the CAM mask or an output anamorphic stage can be employed. In the later case, to provide further speed increase, detectors with small apertures can be employed. Although the intensity integration is performed for all masks, only one CAM mask is illuminated at a time, which corresponds to an execution of one operation. For a n-bit processor executing k different instructions, the total number of elements in a CAM column is equal to 4kn. For example, for an 8-bit processor performing addition, subtraction, multiplication and division, the number of column elements is 128. Since the output result is obtained by performing a logic OR operation on fixed number of product terms, all the submasks corresponding to the same output bit must have the same number of columns. In this case, some of the submasks, corresponding to the same output bit, will contain a redundant number of opaque columns. Additionally, different operations require different number of output functions, e.g. for an n-bit multiplication 2n output bits are needed, while for n-bit addition only n+1 bit output is generated. Since an operation that requires the largest number of output bits determines the maximum number of CAM masks, for the same number of the output bits, the operation that contains smaller number of output bits must be appended with all opaque masks.

4.5.3 An Angularly Multiplexed processor

The device shown in Fig. 2.2.1 is a six-input to one-output device, which generates in parallel a single optical stage MSD addition 1-bit result. To add two N-digit MSD numbers, N+1 output digits are needed. In a straightforward extension, N+1 such 1-bit devices should be used. However, this approach leads to an unacceptably large space-bandwidth product. To minimize the size of the CAM system, next, an efficient angular multiplexing method that drastically reduces the CAM processor's architectural complexity is proposed. Since all the N+1 free-space optical processors are identical, an angular multiplexing scheme is possible. In Fig.4.5.4, a proposed angularly multiplexed N-digit optical CAM based MSD adder architecture is depicted. Here, one 56 X 18 pixel CAM MSD addition mask and N+1 angularly multiplexed input patterns are used for the N+1 pattern matching operations. Here, as a 1D (vertical) array, the N+1 6-digit programmable optical input devices (SLMs) are positioned.

Each 1D SLM is connected to six electronic input ports. The dotted lines in Fig.4.5.4 indicate constant zero inputs. Past the input SLM array a 1D prism wedge array is placed. Each wedge element deflects its output to a common but shared CAM MSD mask. The matching results are then spatially integrated by a single cylindrical lens positioned immediately past the CAM mask. At the lens back focal plane, because of the angular multiplexing, the integrated results of the N+1 vertically displaced horizontal channels are

displayed. At this plane 56 X N+1 electronic threshold logic inverters are located. These results are connected to a device that outputs the N+1 bit MSD summation result where each bit has three position coded channels.

4.6 Arithmetic operations

In recent years, to implement optical processors various efficient number systems have been studied. In this section, for different number systems, a non-holographic CAM-based arithmetic processor implementation is described. In addition to binary number (BN), residue number (RN), here, the use of a sign/logarithm number (SLN) system for optical CAM computing is also proposed. To simplify hardware complexity, the given logic functions, specified as truth tables, must be minimized. For an ease of implementation, using a Quine-McCluskey minimization method, the total number of product terms are substantially reduced [14].

4.6.1 Binary arithmetic processor

To illustrate the capability of the non-holographic CAM method, first a design of a BN CLA adder and BN multiplier are presented. A CLA, where BNs to be added are (a_{n-1} ... a_2 a_1 a_0), (b_{n-1}... b_2 b_1 b_0) and the input carry bit c_0, is a (2n+1)-bit logic device generating an n-bit sum (s_{n-1} ... s_2 s_1 s_0) and an output carry c_{out} bit. In general, the minimized expression for the kth sum bit can be represented as [15]

$$s_k = a_k \oplus b_k \oplus c_k$$

$$= a_k b_k c_k + \bar{a}_k \bar{b}_k c_k + \bar{a}_k b_k \bar{c}_k + a_k \bar{b}_k \bar{c}_k, \quad (4.6.1)$$

where ⊕ and + represent the Exclusive-OR and the Inclusive-OR operations. The minimized expression for (k+1)th carry bit is

$$c_{k+1} = a_k b_k + a_k c_k + b_k c_k, \quad (4.6.2)$$

where k ranges from 0 to n-1. Each sum and the output carry bit can be expressed as a function of the input carry and the two input BNs. Using a CAM mask, in parallel, each output function bit s_{n-1}, ... s_2, s_1, s_0 and c_{out} is implemented. To estimate the size of the mask, we note that, the number of different reduced product terms P_k and C_k, for the kth sum and output carry, are

$$P_k = 4(2^{k+1} - 1), \quad (4.6.3)$$

$$C_k = 2^{k+1} - 1, \quad (4.6.4)$$

respectively [16]. The total number L_t of different product terms, for an

n-bit CLA, is equal to

$$L_t = \sum_{k=0}^{n-1} (P_k) + C_n = 10 \times 2^n - 4 \times n - 9. \quad (4.6.5)$$

For a 4-bit CLA, 135 product terms should be stored. For higher dynamic range computations a larger number of product terms is needed e.g. 2235 product terms for an 8-bit CLA.

By using a multiplication truth table, binary multiplication operation can be performed . For an n-bit multiplication, the result contains 2n output bits. With each output bit, a logic function (encoded as a CAM mask) is associated. Using the Quine-McCluskey minimization method, the number of product terms for each output logic function can be substantially reduced. For example, for a 3-bit multiplication, six output functions A_5, A_4, A_3, A_2, A_1 and A_0 need to be encoded as CAM masks. The unminimized expressions for these functions contain 6, 15, 22, 28, 24 and 16 product terms, respectively, for a total of 111 product terms. After minimization, the number of product terms was reduced to 3, 8, 10, 9, 4, 1, respectively, providing over 3-fold product term reduction. To implement multiplication of larger BNs, with each additional bit, the number of product terms increases rapidly. The direct binary multiplication CAM processing is therefore suitable for relatively small dynamic range calculations. For multiplications requiring higher dynamic range, other number systems such as MSD, RN or SLN systems can be applied.

4.6.2 Modified Signed-digit (MSD) processor

Recently, using various optical logic and memory processing techniques, optical MSD arithmetic operations have been proposed. However, in most existing optical scheme, a multiple-stage algorithm is utilized. For a multistage approach, because of optical cascading, additional signal energy loss and speed reduction is unavoidable. Here, a novel non-holographic opto-electronic CAM based fast MSD addition processing architecture is proposed. Based on this key opto-electronic CAM element, implementation of a more sophisticated MSD arithmetic operations is proposed. Finally, for the completion of a MSD arithmetic processing, the conversion between the conventional binary and MSD numbers, will also be proposed.

Despite the many and appealing MSD processing advantages, a fast and practical either electronic or optical implementation has been difficult to achieve. The traditional electronic MSD arithmetic scheme uses some binary logic gates to simulate the three-stage processing algorithm. The speed of this approach is very limited [17]. To implement an electronic CAM based MSD single-stage arithmetic a VLSI programmable logic array has to be used. For example, to obtain each additional output bit, about 500 logic elements, e.g. diodes and transistors, have to be used. Since a programmable logic array employs a mesh-type network, the densely packed conducting lines can cause serious interference problems. The speed of such an electronic CAM is also quite limited. For example, a kilobit CAM that has a minimum 140 ns cycle time can only operate at a maximum 5 MHz [18]. On the other hand, although

optical switching and free-space interconnect promises many speed and parallel processing advantages, up to now, only a few successful experimental efforts have been reported. For optical logic-based MSD processing, using a three-step MSD algorithm, a fast, compact and power-efficient MSD logic gate is desired. The satisfaction of these requirements has been proved to be difficult. For example, the scheme [19] that uses a combination of gratings, prisms, and etalons requires the alignment of 27 such elements per logic gate. For a symbolic substitution scheme [20-22], each logic gate requires nine recognition and substitution rule processors. Maintaining the required fan-out ratio will also be difficult. Even when such individual gate level processors are implemented, cascading them into three practical processing stages will be even more difficult to accomplish. The processing speed of such a processor will also be slow. The present use of both inefficient algorithms and their corresponding optical architectures form the major limitation of the present logic-based MSD arithmetic approach. The main difficulty with an optical memory-based approach, on the other hand, stems only from its optical architecture. The proposed optical memory-based optical MSD processing suggests the use of an optical content-addressable memory (CAM) [1-3, 23-24] to store MSD arithmetic computing results. Unlike location-addressable memory, where each stored result has an assigned memory address, a CAM groups all of the identical contents into a compact logic pattern and stores it.

4.6.2.1 MSD addition

By using a large number of "don't care" digits, the required storage capacity can drastically be reduced. It has been indicated that using a conventional three-stage MSD addition architecture, the MSD addition carry can only propagate two digits to the left. Since each carry digit is generated via an addition of two input digits, in principle, for each addition output digit s_i, three input digit pairs [2], e.g. $A_{i=1}$, B_{i+1}, A_i, B_i, A_{i-1}, B_{i-1}, should be used (see Fig.2.2.1). Since each MSD digit has three logic levels, to generate each output digit of the single-stage MSD addition, as many as $3^6 = 729$ different six-variable combinations have to be considered. In a previous work [2], it has been shown that out of 729 logic combinations, only 183 combinations are responsible for the generation of an output logic level 1 and $\bar{1}$ each, the remaining 363 terms are responsible for the output logic level 0. If all these cases are stored in a LAM, to generate a single output digit, a very large memory space has to be used. Using a CAM approach, a drastic logic term reduction before storing is possible. With the help of "total-don't-care" digits, denoted by A_x and "partial-don't-care" digits denoted by $A_{1\bar{1}}$, A_{10}, $A_{\bar{1}0}$, several six-variable logic terms can be combined into a compact logic expression. For example, the reduced logic expression $A_{1\bar{1}}A_1A_xA_0A_{10}$ represents the following twelve unreduced MSD input logic combinations: 11001, 11000, 11101, 11100, $11\bar{1}01$, $11\bar{1}00$, $\bar{1}1001$, $\bar{1}1000$, $\bar{1}1101$, $\bar{1}1100$, $\bar{1}1\bar{1}01$, $\bar{1}1\bar{1}00$. Using these reduced logic combinations, for a MSD addition, for the

numbers 1 and $\bar{1}$ each, only 28 terms (also called reference patterns) are required Since with a binary logic NAND gate based on 1 and $\bar{1}$, the output 0 digit can be formed, its CAM processing can be omitted. Thus, to generate each MSD addition output digit, a total of 56 six-variable reference patterns are required [2].

For a CAM readout, the input is used to match in parallel with all the stored information. A proper match will retrieve the required data. Because the CAM-based MSD addition uses a single-stage processing algorithm, the cascade and speed reduction problems are avoided. The remaining question is how to form an efficient optical CAM. The proposed volume holographic CAM [1-2] uses three recording steps for each hologram. To generate an addition 1-bit result, as many as 56 such holograms are required. Thus, even for obtaining a 1-bit result, both a sophisticated recording procedure and a holographic material with sufficient multiplexing capability form the major limiting factors of this approach. Because of these and other problems, no successful experimental CAM MSD processing has been reported.

In this section, based on the above-described opto-electronic (O-E) CAM-based MSD adder, the implementation of various other important O-E arithmetic processors are proposed. The goal is to implement the various necessary O-E supporting elements for a fast MSD arithmetic processing.

4.6.2.2 MSD subtraction

A MSD processing advantage is that a negative number can easily be obtained via a parallel bit-wise logic inversion from its positive counterpart. The logic inversion is defined to keep the 0 unchanged while there is an interchange between a 1 and $\bar{1}$. With this interchange, a MSD subtraction X - Y is obtained through the addition X + (-Y). With the previously proposed device, for a MSD subtraction, an additional exchange switch array is needed. This can be implemented either electronically or optically by an exchange switch between the two position-encoded Y input channels. As compared to MSD addition, a delay of only one time-cycle is needed.

4.6.2.3 MSD Multiplication Tree

A straightforward way to implement digital multiplication is through a series of addition steps [25]. Because of carry propagation, with a binary number system, a time-consuming multiplication operation is unavoidable. Since the MSD number representation provides a carry-free addition capability, in principle, the MSD numbers lend themselves for a fast multiplication. Various MSD fast multiplication algorithms were proposed. A MSD sequential multiplication method was first proposed by Avizienis [26]. A serial-parallel multiplier was suggested by Atkins [27]. Recently, Takagi et.al. [17] proposed a MSD adder-tree-based fast multiplication

scheme which uses a large degree of parallelism. Using this algorithm, in this section, an O-E MSD multiplication processor is proposed.

For multiplication of two N-bit MSD numbers, results in N X N MSD matrix array known as a partial product matrix are produced. To form the final 2N-bit MSD multiplication result, N-1 MSD string-pair additions must follow. It has been shown that using a parallel tree-decomposition, the N-1 string-pair additions can be performed through $\log_2 N$ steps [17]. This tree-addition algorithm is best explained via an example (see Fig.4.6.1.). Here, two 8-bit MSD numbers $X = 1\bar{1}\bar{1}0\bar{1}011$ and $Y = 10\bar{1}0\bar{1}1\bar{1}0$ are used. After forming an 8 X 8 partial product matrix, in the first stage, four parallel MSD string-pair additions are performed. The resulting four addition results are divided into two groups and summed separately to form the next step addition result. Using a third addition, the final multiplication (16-bit) result is obtained. Thus, in $\log_2 8$ steps, a 8-bit multiplication is performed. It can also be seen that, using this approach, for an N-bit multiplication, a total of N-1 MSD adders should be used. The advantage of this approach is its fast speed. In fact, this is the fastest MSD multiplication scheme available.

To optically implement this approach, an O-E iterative multiplication processor is proposed. In Fig.4.6.2, this processor's architecture is shown. For the multiplication of two N-bit MSD numbers, N/2 parallel O-E MSD adders are placed. These adders are used to perform the N/2 first-step MSD additions. The N/2 resultant strings are fed-back electronically using a fixed interconnect network (for an example of N=16 see Fig.4.6.3). With

this feedback, in the next step, only one half of the available adders are activated. In this fashion, after $\log_2 N$ iterative steps, the final MSD multiplication result is obtained.

4.6.2.4 Binary-MSD conversion

Another MSD processing bottleneck is the MSD to binary number conversion. At present, an optical MSD processor will most likely serve as a fast special-purpose arithmetic processor. Thus, an interface of this processor with other binary optical or electronic devices is necessary. Since the binary number is a subset of a MSD number system, no additional conversion from a binary to a MSD number is needed. However, after obtaining the MSD processing result, the conversion of a MSD number back into a binary representation raises a serious problem. The conventional number conversion algorithm [21-22] first separates the MSD number into a positive and negative parts. Next, a binary number subtraction of the two parts is performed. Since carry-free binary subtraction algorithm does not exist, the previously gained MSD processing speed may be partly or totally lost. Thus, for a successful optical MSD arithmetic processing, new and efficient number conversion schemes and their optical implementations are also necessary. Here CAM-based conversion method is discussed. First the conversion truth table is generated. Next, for each output bit a minimized logic function (sum of products) is implemented using CAM.

4.6.3 Sign/logarithm number based arithmetic processor

The sign/logarithm processor, although relatively easy to implement electronically, imposes problems for an optical implementation. The alternative method is to use a CAM for number system conversions as well as for multiplication or addition operation implementation. This approach takes advantage of the optical parallelism and the high CAM storage capacity.

A binary coded SLN is represented as

$$S \quad A_{n-1}A_{n-2}\cdots A_0 . A_{-1}\cdots A_{m-2}A_m \qquad (4.6.6)$$

where S is the sign of the BN number. The n digits A_{n-1} to A_0 represent the characteristic, while the m digits A_{-1} to A_{-m} represent the mantissa. The corresponding decimal number N is

$$N = S \quad 2^{A_{n-1}\cdots A_0} 2^{A_{-1}\cdots A_{-m}} \qquad (4.6.7)$$

For an integer, both the characteristic and the mantissa are positive numbers. To represent fractions, both the characteristic and the mantissa can be regarded in a two's complement format. In this case, a representation using negative powers is possible. Consider, as an example, the multiplication of 9X20 using a 4-bit each characteristic and mantissa. The integer numbers 9 and 20 are represented as

$$9 = 0 \quad 0011.0011 = 2^3 2^{\frac{1}{8}+\frac{1}{16}}, \qquad (4.6.8)$$

$$20 = 0 \quad 0100.0101 = 2^4 2^{\frac{1}{4}+\frac{1}{16}}. \qquad (4.6.9)$$

Adding these two SLNs, the binary coded logarithm 0 0111.1000, corresponding to the decimal number $2^7X2^{1/2}$ (181.0193), is obtained. Thus, the error due the mantissa truncation is equal to 1.0193. By increasing the mantissa length, a more accurate result can be obtained.

To implement a SLN multiplier, a 3-stage CAM is required. The first CAM performs the conversion from BNs to SLNs. To add the two logarithms, a second CAM performs a carry look-ahead (CLA) addition. Finally, a CAM does the conversion from the SLNs to BNs. The storage capacity needed for each CAM stage depends on the range of input numbers and the calculation accuracy.

Binary addition (subtraction) can be performed using a SLN system as well. The sum of two BNs A and B can be expressed as

$$A + B = A(1 + B/A). \qquad (4.6.10)$$

First, the ratio B/A is computed, to this ratio a constant unit is added, and finally, the multiplication by A is performed.

4.6.4 Residue number processor

The RNs can be encoded using multiple rail spatial symbols, but this approach is suitable for relatively small moduli. Using binary encoded RNs, large dynamic range can be obtained. For example, assuming 4-bit binary encoded RNs, with moduli 15, 13, 12, 11 and 7, can provide a dynamic range of 180180, which corresponds to 8-bit binary multiplication. Using the same set of moduli, a 16-bit addition can be implemented. For each modulus, a separate CAM mask must be implemented. Since multiplication (addition) is decomposed into a mod 15, mod 12, mod 13, mod 11 and mod 7 multiplication (addition), these results can be represented by 4-bits (3-bits for modulus 7). In order to provide an efficient use of CAM, all the nineteen output functions must be minimized.

The application of a CAM can be extended to the conversion of a BN to a RN system. A number of CAM conversion masks is equal to number of moduli used for a RN representation. For the example cited, five CAM masks are required. Also, the RN multiplication (addition) result must be converted to a BN system. For the RN to BN conversion the number of CAM masks is determined by the multiplication (addition) dynamic range. With each output bit a CAM mask is associated, e.g. seventeen masks for 16-bit addition and sixteen masks for 8-bit multiplication. For each modulus, the number conversions are performed in parallel. This approach provide a high dynamic range as well as high processing speed.

4.7 Experimental Results

To assure proper detection, uniform LD intensity distribution is necessary. In our experiment, as the source, an array of red light emitting diodes (LEDs) was used. By varying the power supply voltage to each LED, intensity equalization was achieved. Using a 40 mm focal length cylindrical and a 375 mm focal length spherical lens, an anamorphic optical stage was build. The CAM masks were first printed on a Hewlett-Packard laser jet printer and was reduced by a factor of twenty using an optical demagnification process. The final reduced mask was printed on a high contrast (Kodak 2556) film.

4.7.1 Binary carry look-ahead adder.

In this experiment, a 4-bit CLA was implemented. First, using Eq. (14-15), 135 product terms (i.e. P_0=4, P_1=12, P_2=28, P_3=60 C_4=31) are required. The output result contains five bits associated with the functions s_0, s_1, s_2, s_3, and c_{out}. In Fig. 4.7.1a, the five masks M_0, M_1, M_2, M_3 and M_4 used to store these reduced reference terms are shown. In the detector plane, the light transmitted through the masks was detected, by a CCD camera. The output image was compressed in vertical direction by a factor of 3. As an example, two BNs **a**=1011 and **b**=0110 with input carry equal to 1 were added. In Fig. 4.7.1b, the result of illumination of the five masks (rows 2, 4, 5, 8, 10, 11, 14, 16, 17) by nine line sources and the corresponding average intensity distribution along each column are shown. Since highest detected intensity

was equal to $9I_t$, therefore, the threshold level was set at 17/2 I_t. Since the intensity was above the threshold for s_1 and c_{out}, therefore, the result is c_{out}=1, s_3=0, s_2=0, s_1=1, s_0=0 (10010).

In Fig. 4.7.2a, for an active low logic example, the five masks M_0, M_1, M_2, M_3 and M_4 are also shown. Note, that for an active low logic, the masks are the complements of the active high counterparts. The threshold level is set to 1/2 I_t. For the example of Fig.4.7.2a, in Fig. 4.7.2b, the illumination of the five CAM masks and the corresponding average intensity distribution along each column is shown. For the logic functions corresponding to s_1 and c_{out} the intensities below the threshold indicate the result c_{out}=1, s_3=0, s_2=0, s_1=1, s_1=0.

4.7.2 Binary number multiplier.

Next, a 3-bit CAM multiplier was implemented. The multiplication result was represented by six binary logic functions A_0, A_1, A_2, A_3, A_4, A_5 where A_0 (A_5) is the least (most) significant bit. The unminimized expressions for A_0, A_1, A_2, A_3, A_4 and A_5 contain 16, 22, 24, 28, 15 and 6 product terms, respectively (a total of 111 terms). After minimization, the total number of product terms was reduced to 35 (1,4,9,10,8,3, respectively). In this experiment, an active low logic was used. In Fig. 4.7.3a, the six CAM masks M_5, M_4, M_3, M_2, M_1, M_0, corresponding to the output functions A_5, A_4, A_3, A_2, A_1, A_0 are shown. As a test example, the multiplication of two binary numbers a=110 and b=111 was performed. Rows 1, 4, 6, 8, 10, 12 of the CAM masks

were illuminated. In Fig. 4.7.3b, the illumination result together with its intensity integration is shown. With the threshold set at low level, the intensities below the threshold were detected for A_5, A_3 and A_1 which corresponds to the result 101010.

4.7.3 MSD processor

To prove the operational principle and the practicality of the proposed device, some first-order experiments were performed. To form the required CAM mask (see Fig.4.7.4a), first, a 56 X 18 pixel computer-generated mask was generated. This CAM system was tested by three groups of six digit inputs, e.g. $111111.01001\bar{1}$, and $1010\bar{1}1$ (see Figs.4.7.4(b), (e) and (h) for their encoded inputs). For the input of Fig.4.7.4(b), in the top and bottom parts of Fig.4.7.4(c) and 4.7.4(d), for the output channels 1 and $\bar{1}$ the integrated intensity is electronically thresholded and inverted. These results indicate that for this particular six-digit input, an output bit 1 was generated. In Figs.4.7.4(f), (g), (i) and (j), corresponding to the remaining input groups, the obtained $\bar{1}$ and 0 CAM summation results are also shown.

For the processing of the angularly-multiplexed inputs, a second proof-of-principle experiment was performed. Three spatially encoded input masks corresponding to the first, second and last six-digits of the two 5-digit MSD addends, e.g. $0\bar{1}\bar{1}01\bar{1}$, $\bar{1}01\bar{1}01$ and $1\bar{1}0111$, (see also the top part of Fig.4.7.5) were lined from the top to the bottom rows on an input plane device that was located at a distance of 15 cm from the shared CAM mask. The

three input groups were illuminated by three angularly multiplexed optical beams each of which was aligned to project its input on a common CAM mask. The masked results were spatially integrated and displayed at the lens back focal plane (see the bottom part of Fig.4.7.5). The three expected CAM results were 0, 1 and 1. Because of the available devices, an angular multiplexing of more input channels was not performed. However, since only a position- rather than a phase-matching, as is the case with a holographic approach, is required, the alignment of N channels is more straightforward than in other, for example a N channel holographic, schemes. With a higher quality device and a better precision control, we are confident that additional angular channel multiplexing can be done. In any event, to our knowledge, this is the first ever successful experiment which shows a complete MSD addition result.

4.7.4 Sign/logarithm number multiplier.

For each output bit, a separate truth table is needed. Using a Quine-McCluskey minimization method [14], the number of product terms was reduced to 111 (3, 3, 3, 10, 18, 21, 28, 25, respectively). To add two 8-bit logarithmic numbers, an 8-bit CLA whose CAM contains 2519 product terms is required. Although it is feasible to implement this size of CAM, in an alternative approach, two 4-bit CLAs operating in a ripple carry mode can be used. Each 4-bit CLA CAM requires only 135 product terms. A final, a third, CAM does the

conversion from a SLNs to a BNs. For a 7-bit multiplier, the final result is a 14-bit word (F_{13}, F_{12}, F_{11}, F_{10}, F_9, F_8, F_7, F_6, F_5, F_4, F_3, F_2, F_1, F_0). To implement each output bit, the number of product terms is 16, 22, 29, 36, 43, 53, 59, 68, 74, 81, 91, 97, 106, 116, respectively (for a total number of product terms of 891). Using a Quine-McCluskey minimization method, the total number of product terms was reduced from 891 to 329.

A SLN-based multiplier corresponding 7-bit BN multiplier was designed. Assuming 3-bit characteristic and 5-bit mantissa, the binary logarithm is represented by eight logic functions A_2, A_1, A_0, A_{-1}, A_{-2}, A_{-3}, A_{-4}, A_{-5}. The truth tables for the output bits A_2, A_1, A_0, A_{-1}, A_{-2}, A_{-3}, A_{-4}, A_{-5} contain 112, 76, 42, 71, 65, 63, 62, 114 product terms, respectively, for a total number of elements equal to 605. After minimization, the number of product terms was reduced to 111 (3 terms each for A_2, A_1, A_0, and 10 for A_{-1}, 18 for A_{-2}, 21 for A_{-3}, 28 for A_{-4}, and 25 for A_{-5}). In Fig. 4.7.6a, the eight CAM masks corresponding to A_2, A_1, A_0, A_{-1},A_{-2}, A_{-3}, A_{-4}, A_{-5} are shown. As an example, first the BNs a=61 and b=110 (a=0111101, b=11011010) were converted to SLN system. In Fig. 4.7.6b, the conversion results for the numbers 61 and 110 are shown. The intensities below the threshold level were detected for A_2, A_0, A_{-1}, A_{-2}, A_{-3}, A_{-5} (B_2, B_1, B_{-1}, B_{-2}, B_{-5}), which corresponds to SLNs 101.11101 (110.11001).

The addition result, represented by the eight logic functions (d_7, d_6, d_5, d_4, d_3, d_2, d_1, d_0), was equal to 1,1,0,0,1,0,1,1, respectively. This addition result when converted back to a BN system needs 14 output logic functions F_{13} - F_0. In Fig. 4.7.7a, the fourteen masks associated with output bits F_{13} - F_0 are

presented. In Fig. 4.7.7b, the illumination of CAM masks and corresponding intensity distribution is shown. The intensity below the threshold level appeared at F_{12}, F_{11}, F_8, F_7, F_6, F_2, F_0 corresponding to the binary result 01100111000101 (6597). Due to the mantissa approximation, the truncation error was 1.684%.

4.7.5 Residue number multiplier.

Next, a CAM-based RN multiplier that delivers range from 0 to 105 was realized. Three moduli 7, 5, 3 were chosen. To encode the multiplication operands as well as the multiplication result, active low logic was used. The binary input data is first converted to the RNs using a CAM mask shown in Fig. 4.7.8a. The unminimized truth table for the BN to RN conversion contains 30 product terms. After the minimization, this number was reduced to 23. In Fig. 4.7.8b (6c), the transillumination of the CAM mask by a sequence of bars corresponding to BN 9 (8) is shown. In the case of number 9, for mod 7 (mod 5), the intensity below the threshold was detected for $\mathbf{a}_1$ ($\mathbf{a}_2$). For the second operand, the number 8, the intensity below the threshold was detected for $\mathbf{a}_0$ (mod 7), $\mathbf{b}_0$, $\mathbf{b}_1$, (mod 5) and $\mathbf{c}_1$ (mod 3). To perform RN multiplication, for each moduli, an independent LD/LED input array is required. The binary output digits corresponding to the modulus 7 are denoted as A_2, A_1, A_0, for the modulus 5 are B_2, B_1, B_0 and for the modulus 3 are C_1, C_0. Using the Quine-McCluskey method, the number of product terms for each output function A_i, B_i and C_i was reduced. The unminimized output functions for

the moduli 3, 5 and 7 requires 4, 20 and 54 product terms, respectively (a total number of product terms of 78). After minimization, the number of product terms was reduced to 4, 15 and 18, respectively (a total of 37 product terms). In Fig. 4.7.9a, the three CAM masks are shown. The output result for the modulus 3 is a 2-bit number C_1 C_0, while for the modulus 5 (7) it is a 3-bit number B_2, B_1, B_0 (A_2, A_1, A_0). As an example, the multiplication of two decimal numbers 9X8 was performed. The decimal numbers 9 (8) in RN system are 240 (132). The multiplication operation was separated into three channels; 2X1 (mod 7), 4X3 (mod 5), and 0X2 (mod 3), respectively. In Fig. 4.7.9b, the three CAM masks transillumination together with the intensity integration along the CAM columns are shown. The threshold level was set at $1/2I_t$. For the output functions A_1 (for mod 7), B_1 (for mod 5), the below threshold intensity was detected. The result was A_2=0, A_1=1, A_0=0 ,B_2=0, B_1=1, B_0=0, C_1=0 and C_0=0, corresponding to the RN 220 (72 decimal). The third CAM performs RN to BN conversion. The unminimized conversion truth table contains 118 product terms, while after the minimization, only 59 product terms are required. In Fig. 4.7.10a, the conversion CAM mask is shown. In Fig. 4.7.10b, the transillumination result corresponding to a RN 220 is shown. The intensities below the threshold were detected for output bits F_6 and F_3 which corresponds to the decimal number 72.

4.8 CAM-based higher order symbolic substitution

An optical system for implementing SS must consist of two subsystems; a pattern recognizer that includes both optical linear and nonlinear devices that perform thresholding and complementing operations, respectively, and a pattern substituting device. Recently, various optical SS implementation techniques, such as associative memory, spatial filtering, phase-only hologram, optical correlation, grating, etc., have been suggested [28-39]. There are two basic SS techniques for recognizing patterns; an additive and a multiplicative logic techniques. With an additive logic technique, the location of the search pattern is ascertained by either a correlation or a superposition of the shifted data copies [30-36]. The additive logic technique, using either an optical intensity or polarization coding, requires an optical NOR operation. With a multiplicative logic technique, light is either transmitted through several transparencies using spatial filters or reflected using an optical cavity [37-39]. The multiplicative logic technique, again using either an intensity or polarization coding, is an optical AND operation. Unlike the multiplicative logic technique, the additive logic technique, because it is a NOR operation, need logic inversion. The required inversion is time-consuming and needs the support of additional optical or electronic devices.

4.8.1 Symbol recognition based on content-addresable memory

A CAM based processor compares the input data with all previously stored reference patterns. When the input matches a stored CAM pattern, an output is generated. In this section, a new opto-electronic CAM based symbol recognition scheme is introduced.

Consider, a 1D binary 12 bit string B (string of primitive elements)

$$B = 0011 \quad 0110 \quad 0111. \qquad (4.8.1)$$

Here, the objective is to find the locations of a 4-bit sequence, i.e. 0110 (the search symbol S) in the bit string B. First, using DR scheme, the primitive elements "0" ("1") in the string B and in the search symbol S are encoded as "01" ("10") producing substituted strings B_e and S_e (S_e=10010110). Complementing the string S_e and repeating it three times, a periodical copy P_e is generated. Performing the logic AND operation on the strings B_e and P_e a resulting string A_e is obtained. The position of the search symbol is indicated by an 8-bit string of all zeros (center set).

$$\begin{array}{l} B_e = 01011010 \quad 01101001 \quad 01101010 \\ \underline{P_e = 10010110 \quad 10010110 \quad 10010110} \\ A_e = 00010010 \quad 00000000 \quad 00000010 \end{array} \qquad (4.8.2)$$

To optically implement higher-order symbol recognition, using a CAM techniques, an input image is encoded using triple-rail spatial symbols shown on Fig 4.8.1. The encoded input data (shown in Fig.4.8.2a) is superimposed with a fixed binary mask (CAM) (see Fig.4.8.2b) resulting in the image shown on Fig.4.8.2c. This CAM approach utilizes a binary periodic mask which is a bit-wise complemented image of the search symbol. In order to established the locations of the search symbols, an integration of the light intensity over each symbol's aperture is performed. To perform this intensity integration, a lenslet array is employed. The size of the elemental lenslet aperture is identical to the size of the search symbol. In this case, the low intensity integration result indicates the search symbol locations. As a final step, to point the search symbols by means of high intensity pixels, light intensity inversion is performed.

4.8.2 Recognizer architecture

To recognize all possible 2^n symbols, in general, 2^n channels need to be used. Using an angularly multiplexed scheme, a number of different output channels (each channel corresponds to a different search symbol), where each channel equipped with a different CAM mask, can be incorporated. In Fig.4.8.3, the optical architecture for a three channel recognizer is shown. The three lenses (input lenslet array) together with a projection spherical lens will produce three collimated beams. The input data symbols are displayed on the SLM that is positioned in the front focal plane of lens L_4. The CAM

masks should be positioned in such a way that they do not overlap. This condition is satisfied when the spacing between two neighboring lenses in the input lenslet array is equal to the linear dimension **d** of the SLM. The longitudinal dimension of the system is equal to $f_1 + 3f_2 + f_3$ and the transverse dimension depends on the number of multiplexed channels (for **k** channels transverse dimension equals to **kd**). Because of rotational symmetry, and to more efficiently utilize the 3D space, the light point sources are to be arranged as a 2D array. To point the locations of the search symbols, an array of opto-electronic inverters (or an SLM) positioned at the output plane is used. The overall performance of the proposed system depends on the type of SLM used. The system's SNR is proportional to the SLM's contrast ratio. For the DR encoding technique half of the area corresponding to one search symbol is masked. Since each element of the output lenslet array preforms integration over the area equal to the area of a single search symbol (n by n pixels), the proper symbol detection is achieved for SLM with a contrast ratio c_r greater than $n^2/2$. Using DR encoding technique for SLMs with c_r equal to e.g. 500:1 the largest search symbols that can be detected should contain about 30 by 30 pixels. Additional limitation for the search symbol maximum size is caused by a crosstalk between symbols during integration by lenslet array. The crosstalk can be minimized by precise alignment of the output lenslet array.

4.8.3 Experimental implementation

For a CAM based symbol recognizer, using a TR encoding technique, the input data is encoded. In Fig.4.8.4a, the binary input data consisting of a 3 by 12 matrix is shown, while in Fig.4.8.4b the corresponding spatially encoded input data is shown. In this experiment, a search for the two 3-bit symbols 011 and 110 was performed. First, two masks corresponding to the search symbols 011 and 110 were generated. In Fig.4.8.4c (Fig.4.8.4d), the masks for the symbol 011 (110) are shown. They are the periodically repeated sequence 100 (001), that is the complement of the search symbol 011 (110). Using a two channel system, two search symbols recognition can simultaneously be performed. The search symbol masks are positioned according to the schematic shown in Fig.4.8.3. By illuminating the input data mask with two angularly multiplexed collimated beams, an AND operation between the input data and search masks was performed. In Fig.4.8.4e (f), for the channels 011 (110), the result of the AND operation is shown. Finally, using a 3 by 4 lenslet array, the intensity average of each 3 by 3 AND plane pixels was obtain. In Fig. 4.8.4g and h, the averaged and the thresholded version of the images of Fig. 4.8.4e and Fig. 4.8.4f are shown. The high intensity pixels represent the lack of search symbols while the low intensity pixels indicate the locations of the search symbols. The lenslet array, used in the experiment, consisted of 12 square aperture lenses. The images of Fig. 4.8.4g and h were recorded by a computer-linked TV camera.

4.9 Summary and conclusions.

The performance of the CAM processor is limited by the LD/LED maximum switching speed, detector response time and electronic comparator's propagation delay. The fastest reported LD switching rate is 14 GHz [26], the optical detector response time can be shorter than 200 ps [27] and ultra-fast comparators can have delay time as short as 500 ps [28]. Additional speed limitation is introduced by the travel time between the LD and the detector array. Distributing the LDs light through optical fibers and using a high resolution (E-beam technology) CAM mask, the longitudinal dimension of the system can be reduced substantially, forcing the travel time to be at least an order of magnitude smaller than the LD switching and the detector response time. The use of an all optical threshold can increase processing speed as well as can simplify the detection step. The advantage of the proposed scheme is that it's calculation speed is independent of the number of bits to be processed. A large number of product terms can be incorporated into a single CAM mask providing fast computation speed. A 4-bit CLA was design and experimentally implemented. To provide for the addition of large BNs, a ripple carry scheme must be employed. Here, the output carry from one stage is connected to input c_0 of the next stage. By connecting output carry from one 4-bit CLA to input carry of the next CLA, the applications of CLA to the addition of large BNs is possible. For an 16-bit CLA, the total summation result can be obtained in about 600 ps. The implementation of multiplication operation using BN, RN as well as SLN representations was presented.

The implementation of direct BN multiplication can deliver the fastest processing speed (one stage CAM), but because of hardware complexity, its implementation is limited to relatively small BNs. As alternative approach using either SLNs or RNs can be used to perform multiplication of high dynamic range multiplication. Here, however, a 3-stage CAM is used. When a number of sequential multiplication operations is to be performed, the final conversion stage is used less frequently, delivering faster overall processing speed. For all the examples, first, the truth tables were reduced and than implemented using a non-holographic CAM. A 3-bit BN, 7-bit SLN, and a three modulus RN multipliers were experimentally demonstrated.

4.10 References

[1] C. C. Guest and T. K. Gaylord, "Truth-Table Look-Up Optical Processing Utilizing Binary and Residue Arithmetic, " Appl. Opt. 19, 1201 (1980).

[2] M. M. Mirsalehi and T. K. Gaylord, " Logical Minimization of Multilevel Coded Functions," Appl. Opt. 25, 3078 (1986).

[3] R. Arrathoon and S. Kozaitis, Opt. Lett. 12, 956 (1987).

[4] A. Huang, "Parallel Algorithms for Optical Digital Computers," Proceeding of the Tenth International Optical Computing Conference, 13-17 (IEEE Computer Society, Los Angeles, 1983).

[5] S. P. Kozaitis, "Higher-Ordered Rules for Symbolic Substitution," Opt. Comm. 65, 339-342 (1988).

[6] G. Eichmann, J. Zhu and Y. Li, "Optical Parallel Image Skeletonization using Content-addressable Memory-based Symbolic Substitution," Appl. Opt. 27, 2905-2911 (1988).

[7] D. Casasent and E. Botha, "Optical Symbolic Substitution for Morphological Transformations," Appl. Opt. 27, 3806-3810 (1988).

[8] P. A. Ramamoorthy, S. Antony and T. A. Grogan, "Symbolic-Substitution-based Median Filters," Opt. Eng. 27, 409-412 (1988).

[9] S. D. Goodman and W. T. Rhodes, "Symbolic Substitution Applications to Image Processing," Appl. Opt. 27, 1708-1714 (1988).

[10] G. Eichmann and S. Basu, "Parallel Optical Syntactic Pattern Recognizer," Appl. Opt. 26, 1859-1865 (1987).

[11] A. L. Lentine, F. B. McCormick, R. A. Novotny, L. M. F. Chirovsky, L. A. D'Asaro, R. F. Kopf, J. M. Kuo and G. D. Boyd, "A 2 kbit Array of Symmetric Self-Electrooptic Effect Devices," Photonics Technology Letters, 2, 51 (1990).

[12] R. Jin, C. Hanson, A. Cavez-Prison, H. M. Gibbs, G. Khitrova, N. Peyghambarian, T. Bowen, F. Y. Juang, P. K. Bhattacharya, D. H. Weinberger, K. R. Evans, C. E. Stutz, R. L. Jones, "Direct Fiber-Etelon-Fiber Interfacing," Opt. Eng. 28, 340, (1989).

[13] E. S. Maniloff, K. M. Johnson, " Dynamic Holographic Interconnects Using Static Holograms, " Opt. Eng., 29 (3), 228-229, (1990).

[14] F. J. Hill and G.R. Peterson, Switching Theory and Logical Design, John Wiley & Sons, New York (1984).

[15] H. Taub, Digital Circuits and Microprocessors, McGraw-Hill, New York (1982).

[16] M. M. Mirsalehi, T. K. Gaylord, D. C. Fielder, and C. C. Guest, "Number representation effects in truth-table look-up processing," Appl. Opt. 28, 1931 (1989).

[17] N. Takagi, H. Yasuura, and S. Yajima, IEEE, Trans. Comput. C-34, 789 (1985).

[18] T. Ogura, S.-I. Yamada, and T. Nikaido, IEEE J. Solid State Circuits SC-20, 1277 (1985).

[19] B. L. Drake, R. P. Bocker, M. E. Lasher, R. H. Patterson, and W. J. Miceli, Opt. Eng. 25, 38 (1986).

[20] R. A. Bocker, B. L. Drake, M. Lasher, and T. B. Handerson, Appl. Opt. 25, 2456 (1986).

[21] P. A. Ramamoorthy and S. Antony, Opt. Eng. 26, 821 (1987).

[22] K. Hwang and A. Louri, Opt. Eng. 28, 364 (1989).

[23] Y. Li, A. Kostrzewski, D. H. Kim, and G. Eichmann, Opt. Lett. 13, 895 (1988).

[24] M. J. Murdocca, Ph.D dissertation, (State University of New Jersey at Rutgers, 1988) ch.6.

[25] K. Hwang, Computer Arithmetic/Principles, Architecture, and Design. (Wiley, New York, 1979).

[26] A. Avizienis, IRE Trans. Electronic Computers, EC-10, 389 (1961).

[27] D. E. Atkins, IEEE, Trans. Comput. C-19, 720 (1970).

[28] F. T. S. Yu, C. Zhang and S. Jutamula, "Applications of One-step Holographic Associative Memory to Symbolic Substitution," Opt. Eng. 27, 399-402 (1988).

[29] G. Eichmann, A. Kostrzewski, D. H. Kim and Y. Li, "Optical Parallel Register Transfer Micro-operations using Holographic Symbolic Substitution," Appl. Opt. 28, 3860-3863 (1989).

[30] M. J. Murdocca, "Digital Optical Computing with One-Rule Cellular Automata," Appl. Opt. 26, 682-688 (1987).

[31] J. N. Mait and K-H. Brenner, "Optical Symbolic Substitution: System Design using Phase-only Holograms," Appl. Opt. 27, 1692-1700 (1988).

[32] D. P. Casasent and E. C. Botha, "Multifunctional Optical Processor based on Symbolic Substitution," Opt. Eng. 28, 425-433 (1989).

[33] K. H. Hwang and A. Louri, "Optical Multiplication and Division using Modified-Signed-Digit Symbolic Substitution," Opt. Eng. 28, 364-372 (1989).

[34] K-H. Brenner, "Programmable Optical Processor based on Symbolic Substitution," Appl. Opt, 27 1687-1691 (1988).

[35] R. Thalmann, G. Pedrini, B. Acklin and R. Dandliker, "Optical Symbolic Substitution Using Diffraction Gratings," Proceeding SPIE 963, 635-641 (1988).

[36] J. N. Mait, "Design of Dammann Gratings for Optical Symbolic Substitution," Proceeding SPIE 963, 646-652 (1988).

[37] M. T. Tsao, L. Wang, R. Jin, R. W. Sprague, G. Gigioli, H. M. Kulcke, Y. D. Li, H. M. Gibbs and N. Peyghambarian, "Symbolic Substitution using ZnS Interference Filters," Opt. Eng. 26, 41-44 (1987).

[38] K-H. Brenner, A. W. Lohmann and T. M. Merklein, "Symbolic Substitution Implemented by Spatial Filtering Logic," Opt. Eng. 28, 390-395 (1989).

[39] Y. Li, G. Eichmann, R. Dorsinville and R. R. Alfano, "An AND Operation-based Optical Symbolic Recognizer," Opt. Comm. 63, 375-379 (1987).

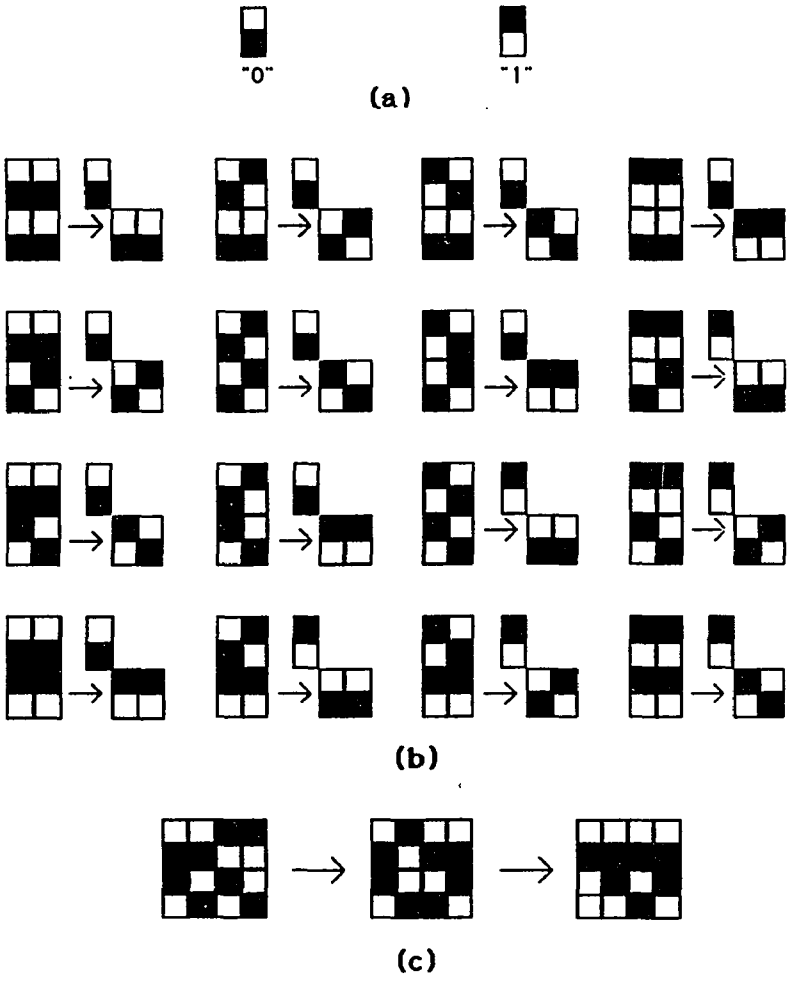


Fig. 4.1.1. (a) Spatial dual-rail symbols representing "0" and "1". (b) Higher-order SS rules for the addition of two 2-bit binary numbers. (c) Using the addition rules in (b) a two-step addition example for binary numbers 0011 and 1010.

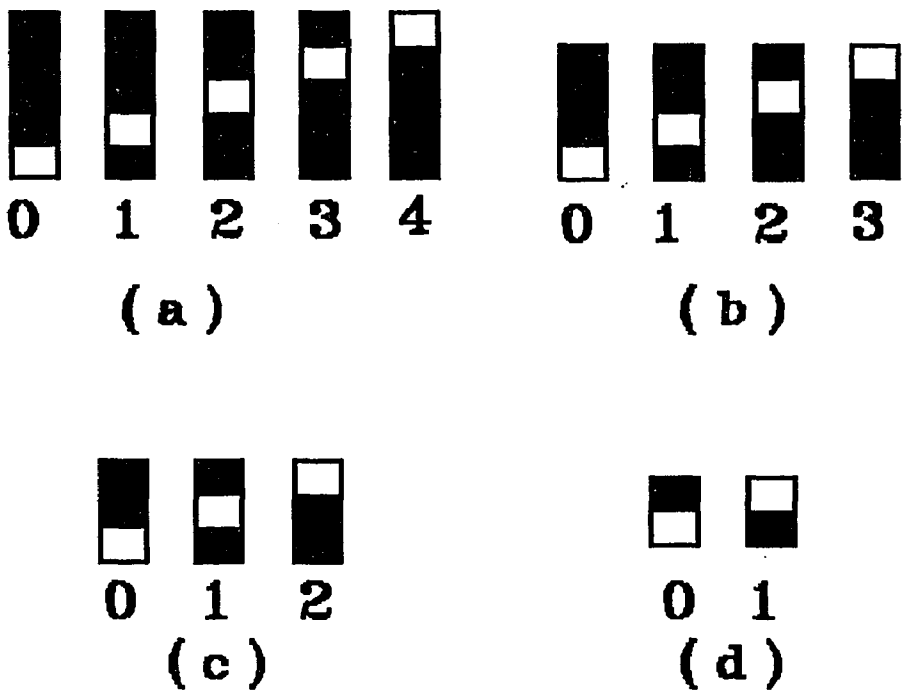


Fig. 4.2.1. Spatial, active high encoding of (a) 5-level, (b) 4-level, (c) 3-level and (d) 2-level multilevel logic variables.

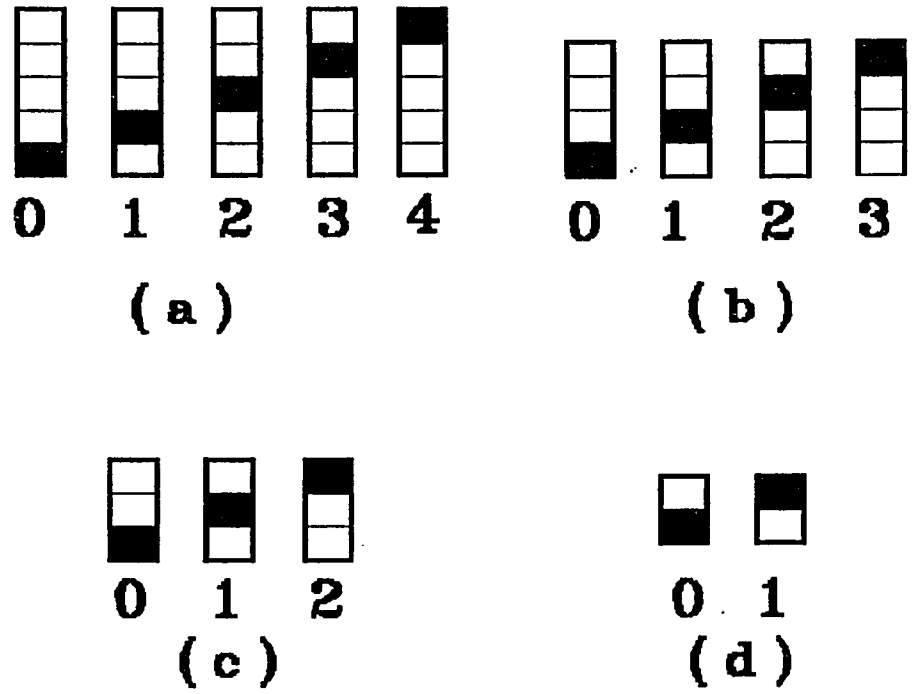


Fig. 4.2.2. Spatial, active low encoding of (a) 5-level, (b) 4-level, (c) 3-level and (d) 2-level multilevel logic variables.

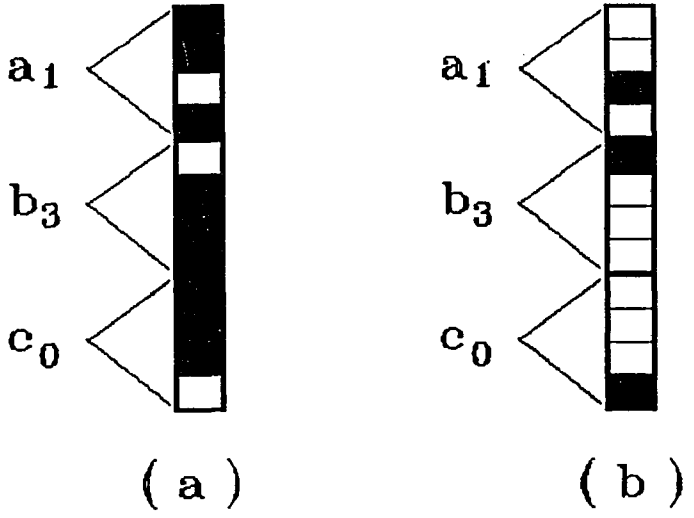


Fig. 4.2.3 Spatial active encoding of a product $a_1 b_3 c_0$ using 4-level variables. (a) using active high logic, (b) using active low logic.

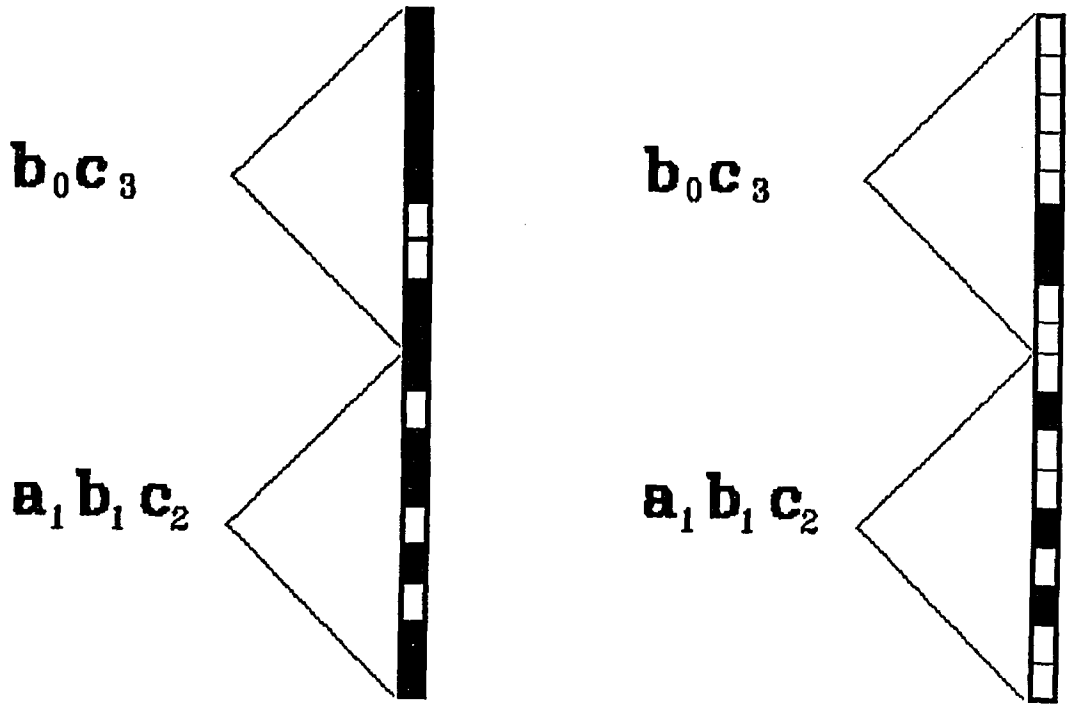


Fig. 4.2.4 Active high spatial encoding of a sum of two product terms for 3-level logic variables $F(a,b,c) = b_0c_3 + a_1b_1c_2$

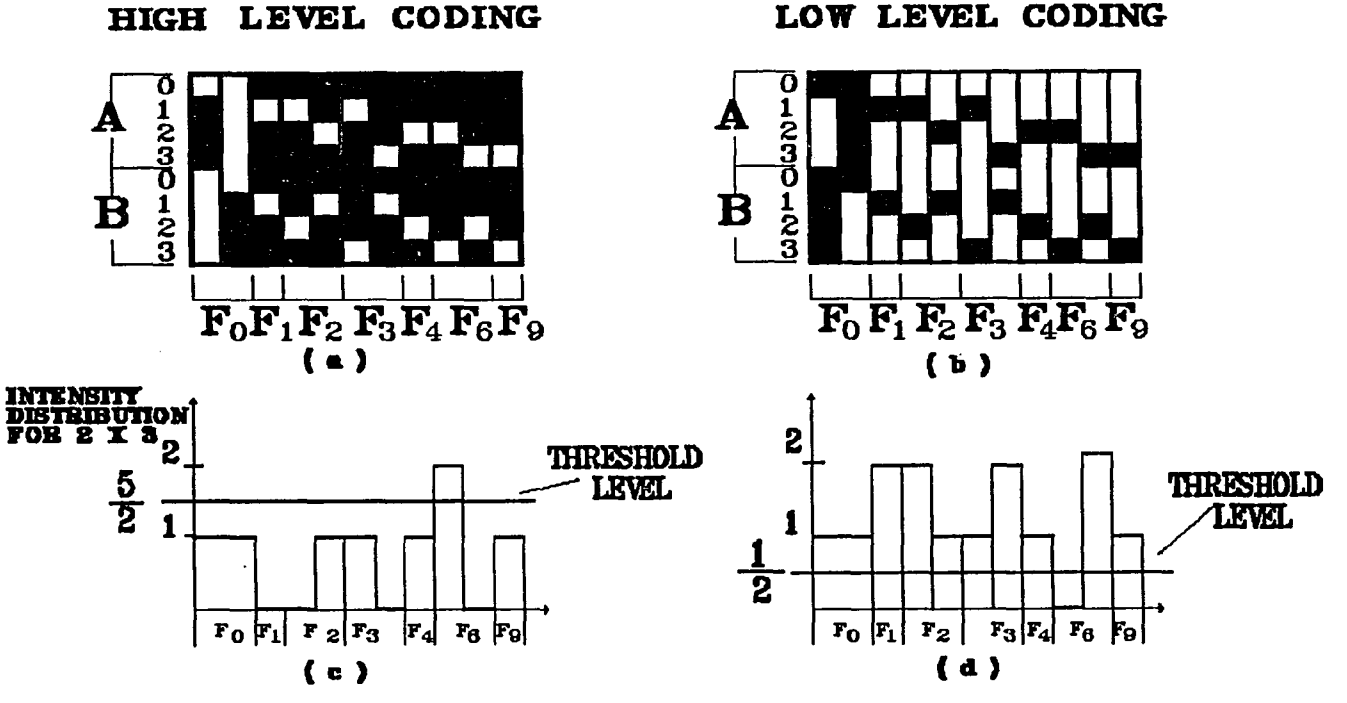


Fig. 4.2.5, Position coded CAM mask corresponding to the functions F_i for (a) high, (b) low level logic. Transmitted light intensity for the CAM multiplication mask (c) active high, (d) active low logic.

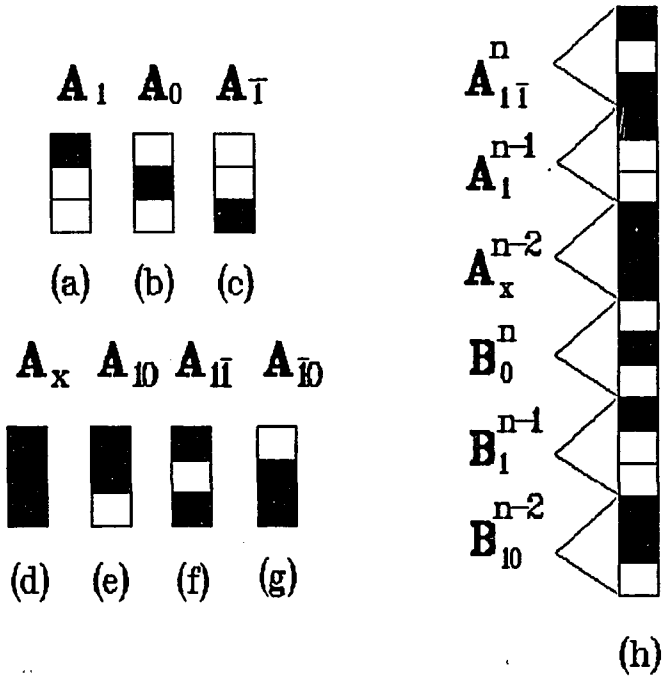
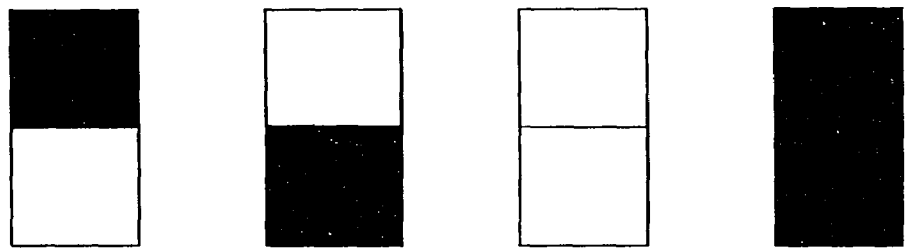


Fig. 4.2.6. (a), (b), (c) Spatial symbols representing MSD digits $1, 0, \bar{1}$, respectively. (d)-(f) Encoding of "total don't-care" (A_x) and "partial don't care" $X_{10}, X_{1\bar{1}}, X_{\bar{1}0}$.
(g) CAM mask representing product $A^n_{1\bar{1}} A^{n-1}_1 A^{n-2}_x B^n_0 B^{n-1}_1 B^{n-2}_{10}$.

logic value (input data)	"1"	"0"	active high logic "x" (CAM mask)	active low logic "x" (CAM mask)
logic variable (CAM mask)	d	$\bar{d}$		



(a) (b) (c) (d)

Fig. 4.3.1. The dual rail (DR) spatial symbol used to encode a active high (low) logic (a) one, (b) zero, (c), (d) don't care.

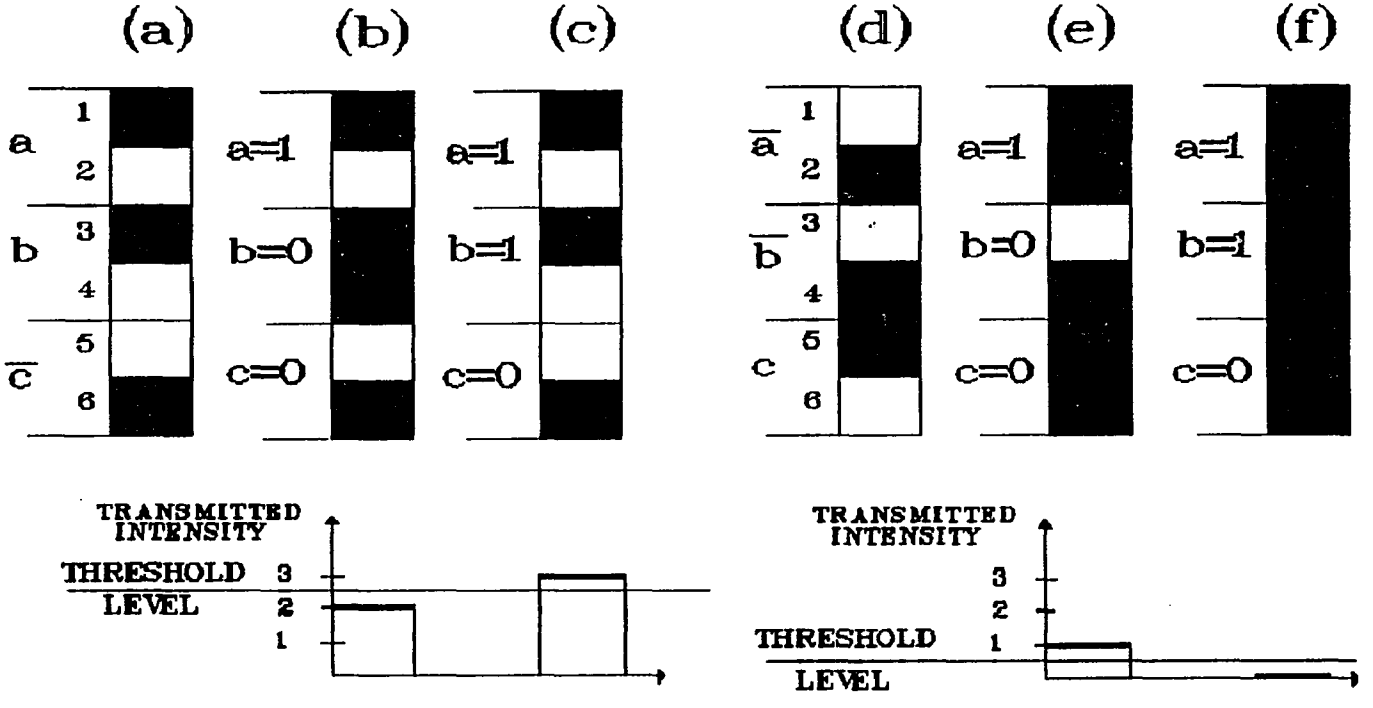


Fig. 4.3.2(a). Active high spatial encoding of a logic product $ab\bar{c}$.(b) The transillumination of the Fig. 2(a) mask by three light sources corresponding to input data a=1, b=0, c=0 and transmitted intensity distribution. (c) The transillumination for the input data a=1, b=1 and c=0 (d) Active low spatial encoding of the logic product $ab\bar{c}$ (e) The transillumination of the Fig. 2(d) mask by three light sources corresponding to input data a=1, b=0, c=0 and transmited intensity distribution. (f) The transillumination for the input data a=1, b=1 and c=0.

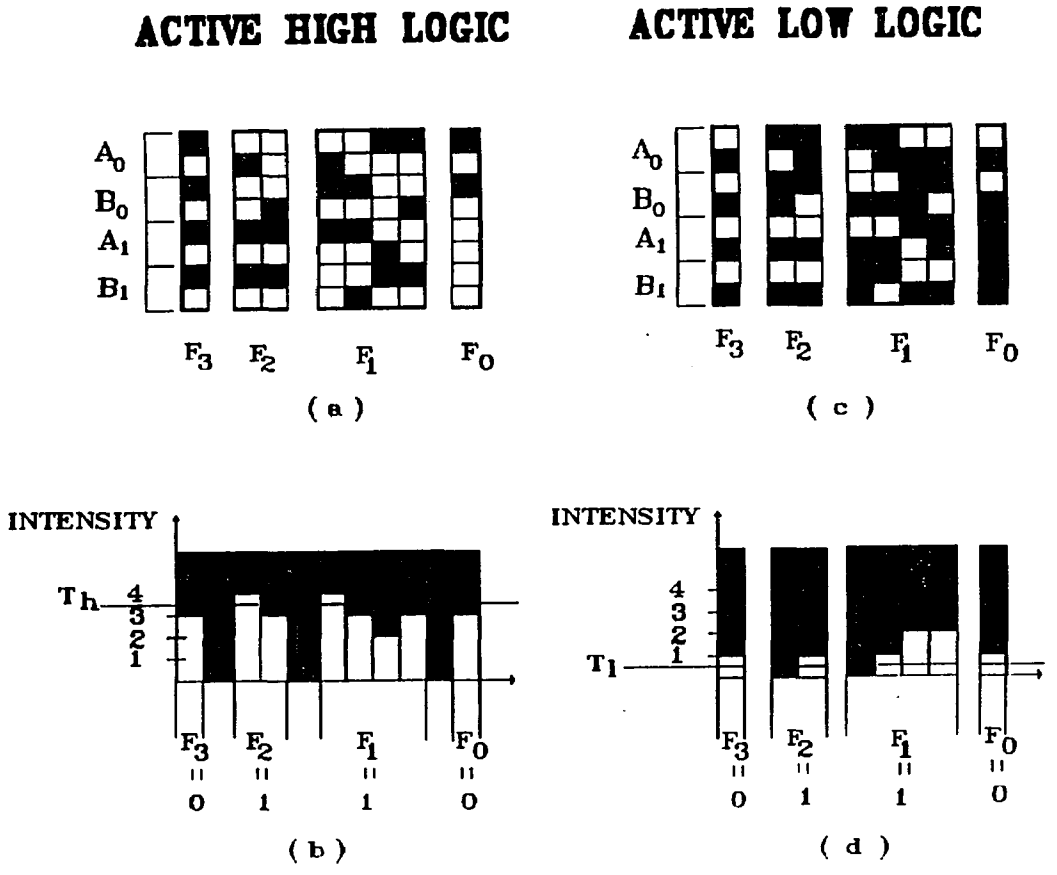


Fig.4.3.3. The two numbers to be multiplied are $A=A_1A_0$ and $B=B_1B_0$. The 4-bit result is represented by four output bits F_3, F_2, F_1, F_0. The CAM masks (a) for an active high, (b) for an active low logic. (c) For an active high logic, column-wise intensity distribution for a 2X3 multiplication. The intensity above threshold (set at $\frac{7}{2}I$) indicates output bit is equal to "1". The result is 0110 (the intensities corresponding to F_2 and F_1 are above threshold). (d) For an active low logic, the intensity below threshold (set at $\frac{1}{2}I$) indicates that the output bit is equal to "1".

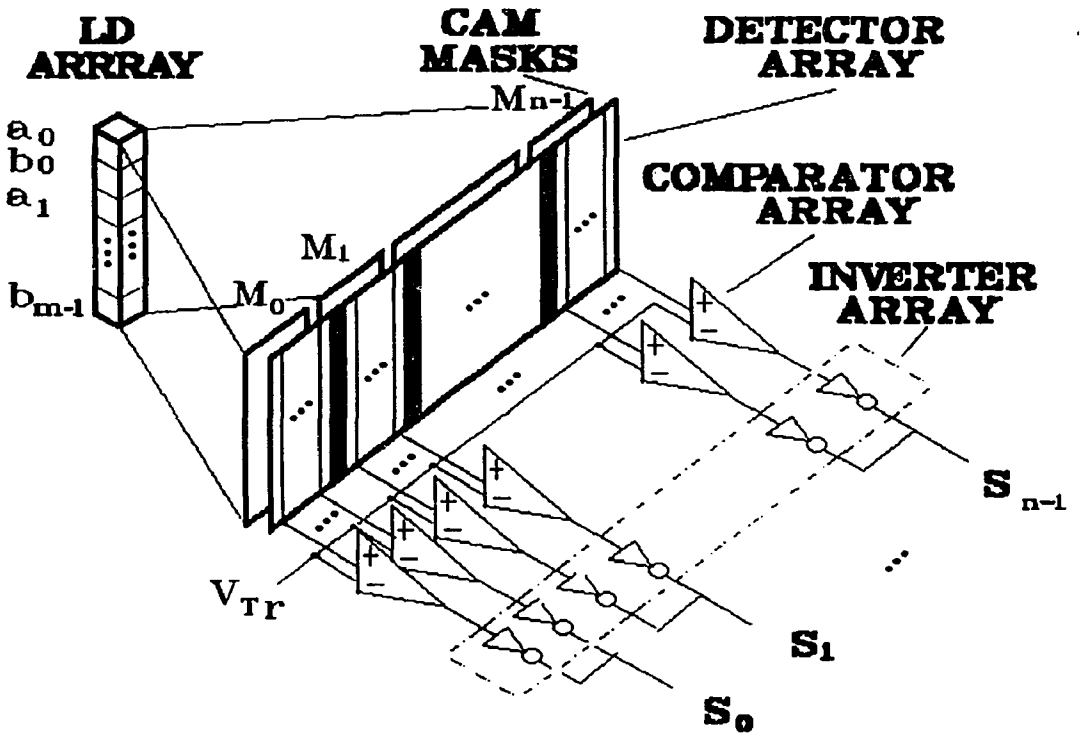


Fig. 4.5.1. Optical implementation of an m-bit CAM processor. For a single logic variable (channel), two laser diodes are used. The CAM masks M_0, M_1,... M_{n-1} are superimposed with a linear detector array. Using an array of comparators, where a comparator corresponds to one product term, threshold operation is performed. All comparators use an identical reference voltage V_{Tr}. For active low logic, an array of inverters is used (drawn by a dashed line).

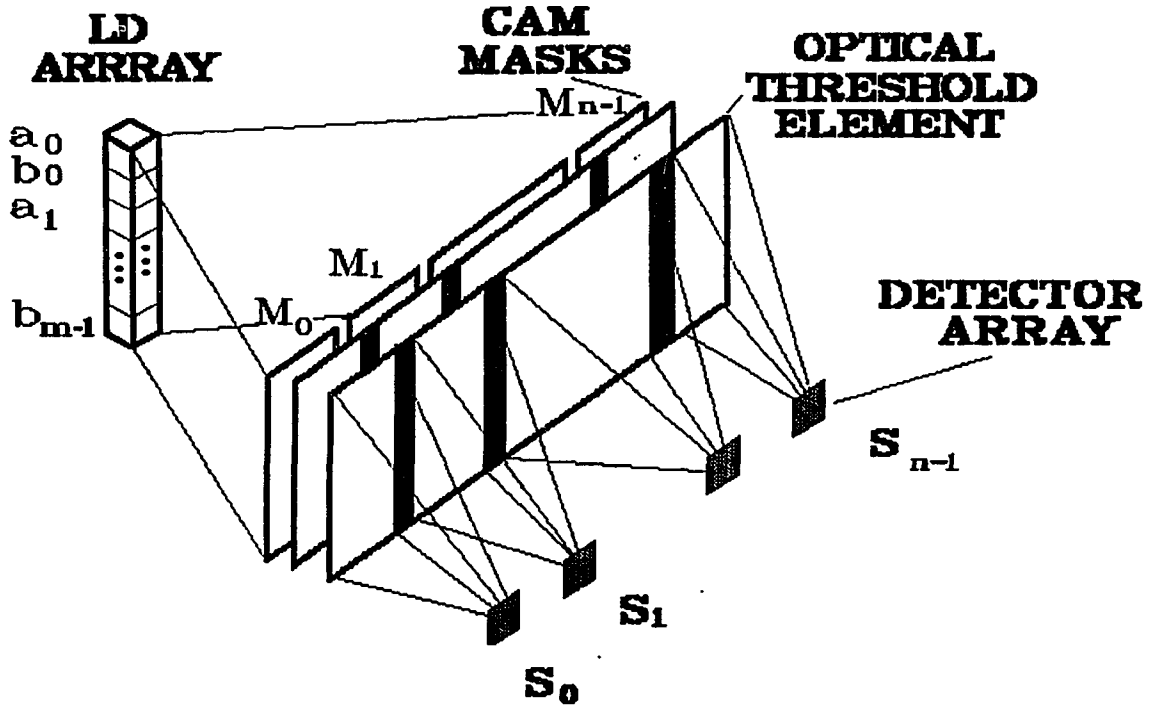


Fig. 4.5.2. Optical implementation of an m-bit CAM processor using optical threshold.

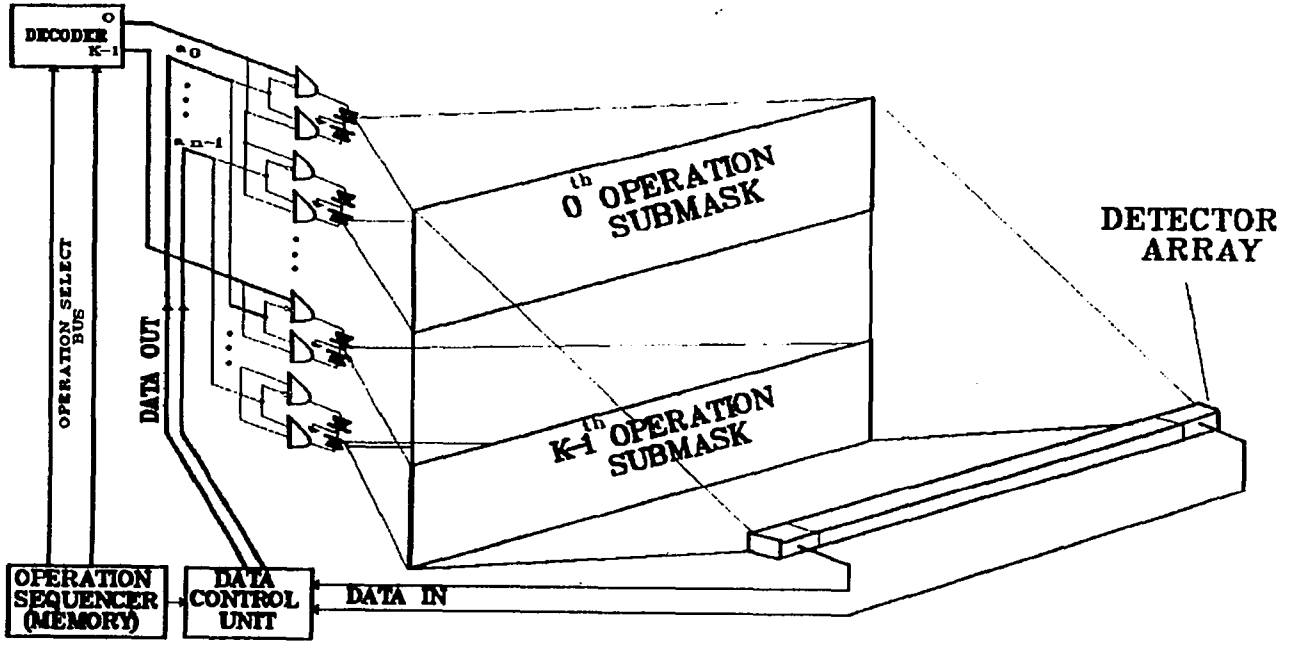


Fig. 4.5.3. A CAM based multioperation processor. A separate CAM mask and a LD array is associated with each operation. An anamorphic optical stage collects the light from each mask to a common detector array. Only one CAM mask is illuminated at a time.

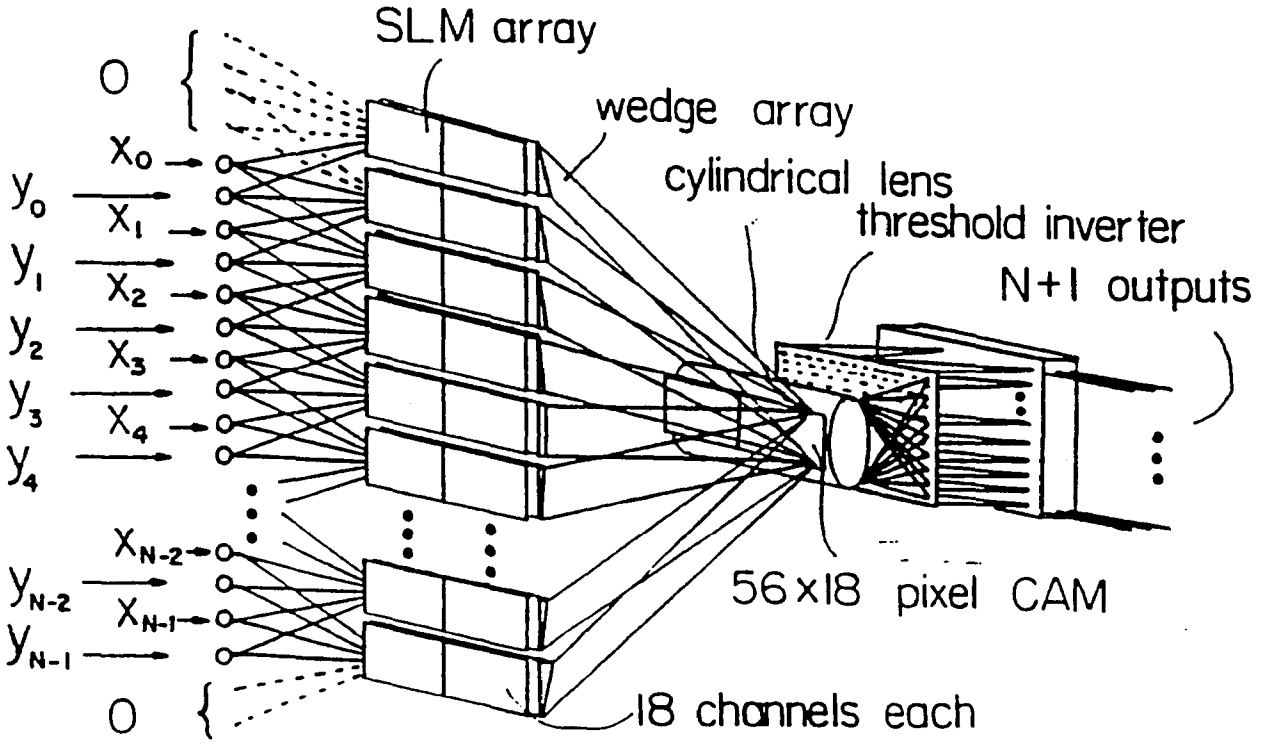


Fig.4.5.4. A proposed angularly multiplexed N-digit optical CAM based MSD adder architecture.

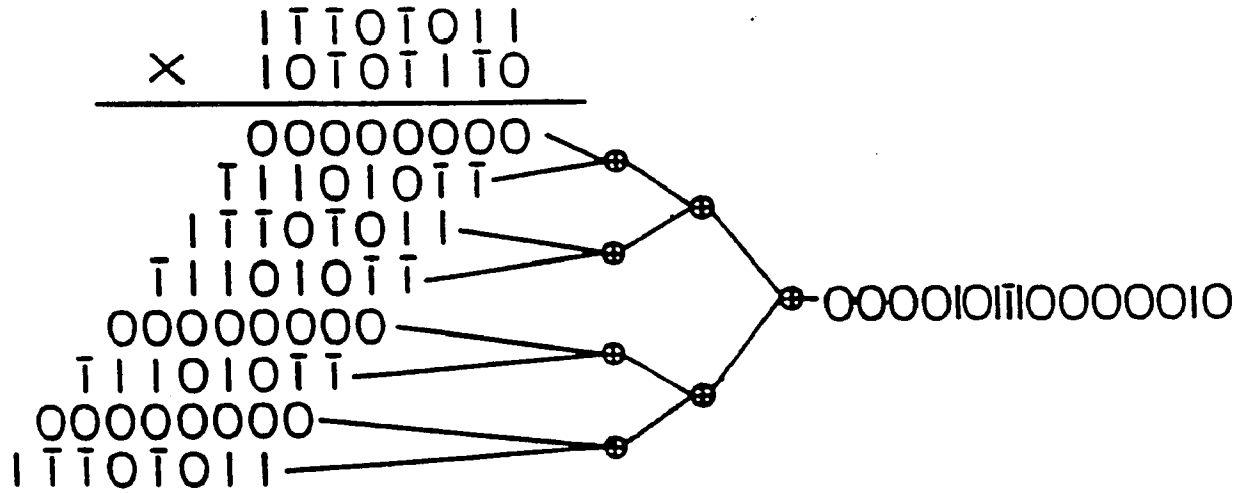


Fig. 4.6.1. Tree-addition algorithm for MSD number multiplication.

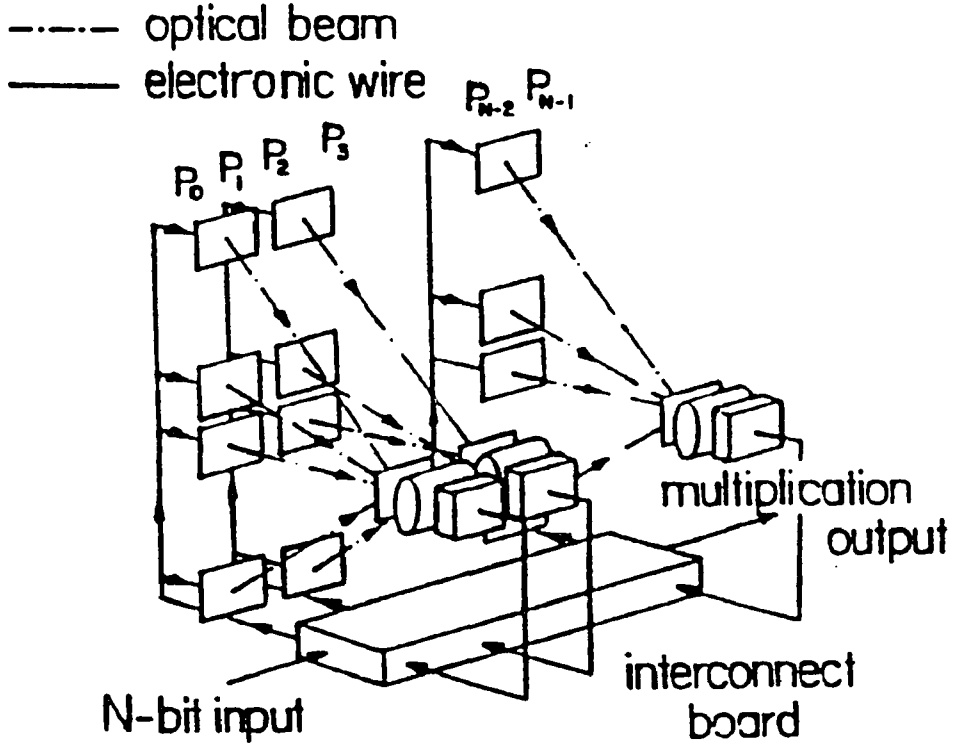


Fig. 4.6.2. A schematic of a CAM-based MSD multiplier.

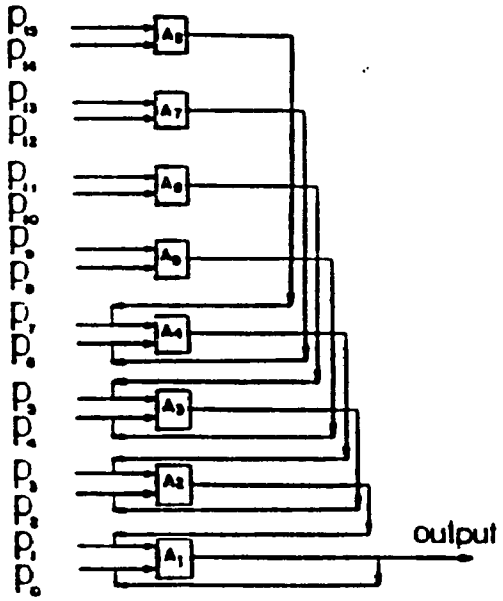


Fig. 4.6.3. An iteration loop for MSD multiplication.

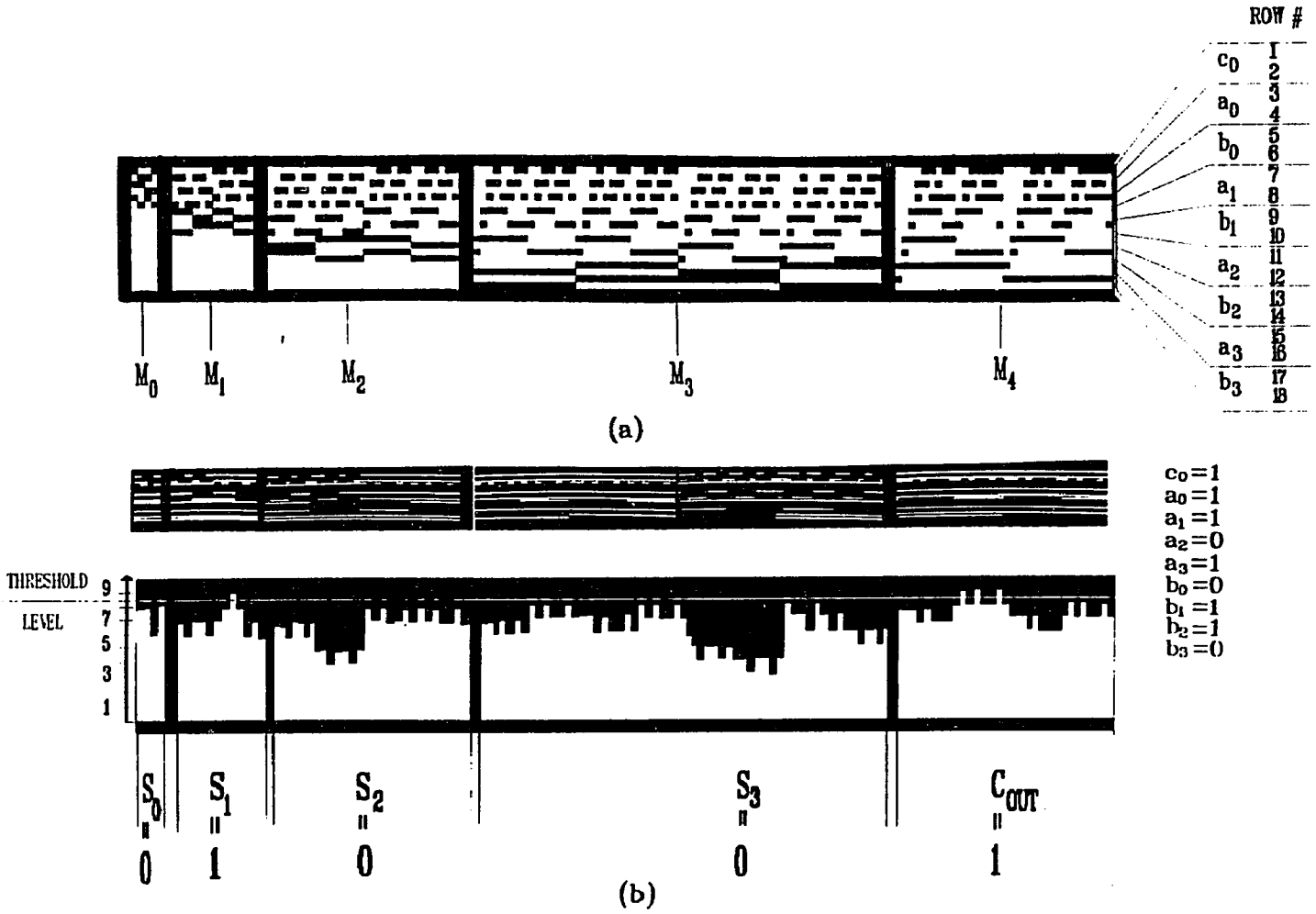


Fig. 4.7.1. (a) Experimental result of 4-bit carry look-ahead adder CAM for an active high logic. The masks M_0, M_1, M_2, M_3, M_4 encoded using dual-rail encoding technique represent the logic output s_0, s_1, s_2, s_3, and c_{out}. (b) The masks illuminated by a sequence of bars representing the binary numbers c_0 = 1, a = 1011 and b = 0110. For masks M_1 and M_5, the intensities are above the threshold level indicating the result 10010.

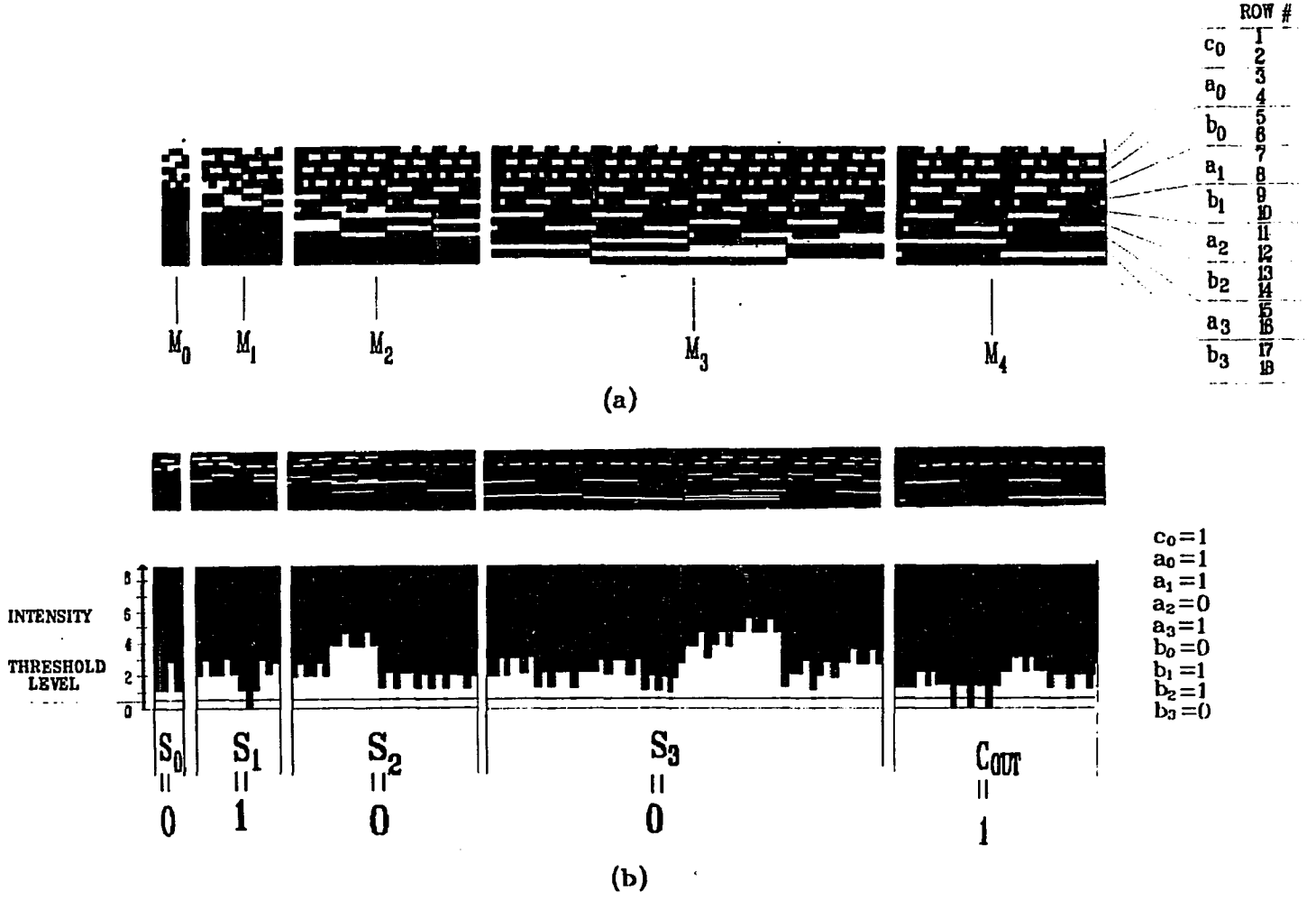


Fig. 4.7.2. For an active low logic, experimental result of a 4-bit CLA.
For masks M_1 and M_5, the intensities are below the threshold indicating the result 10010.

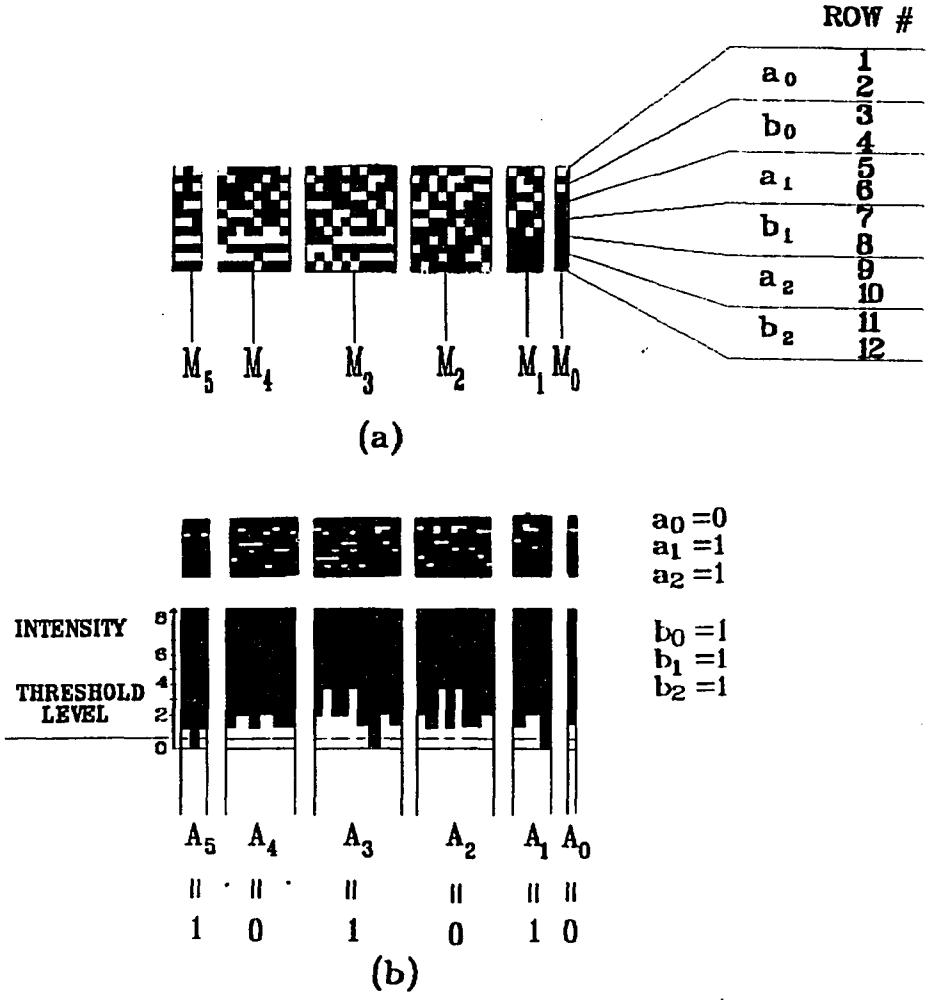


Fig. 4.7.3. (a) Experimental result of a 3-bit binary multiplier using active low logic. The masks M_0, M_1, M_2, M_3, M_4 and M_5 encoded using dual-rail encoding technique represent the multiplication result A_0, A_1, A_2, A_3, A_4 and A_5. (b) The six masks, all illuminated by a sequence of bars representing the binary numbers a = 110 and b = 111 are shown. The intensity below the threshold level are at the output bits A_1, A_3 and A_5 representing the result 101010.

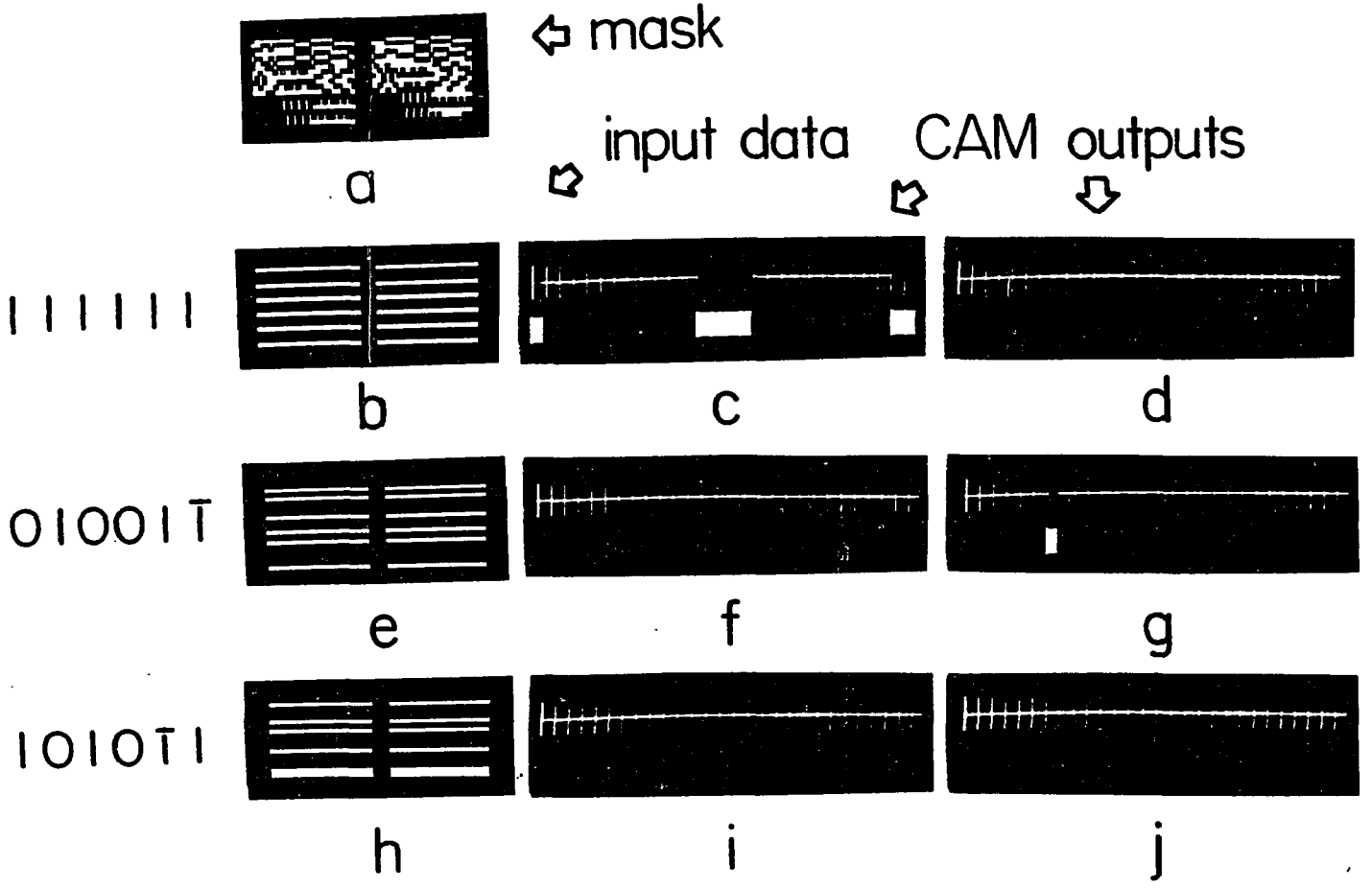


Fig. 4.7.4. (a) Composite CAM mask for a MSD addition [the left- (right-) hand-side submask is for the 1($\bar{1}$) output]. (b) A six-digit input data group. (c), (d) The corresponding masked intensity results for the output channels 1 and $\bar{1}$ in the lens back focal plane (top) and their thresholded and inverted outputs (bottom), respectively. (e)-(j) Two other input data groups and their corresponding results.

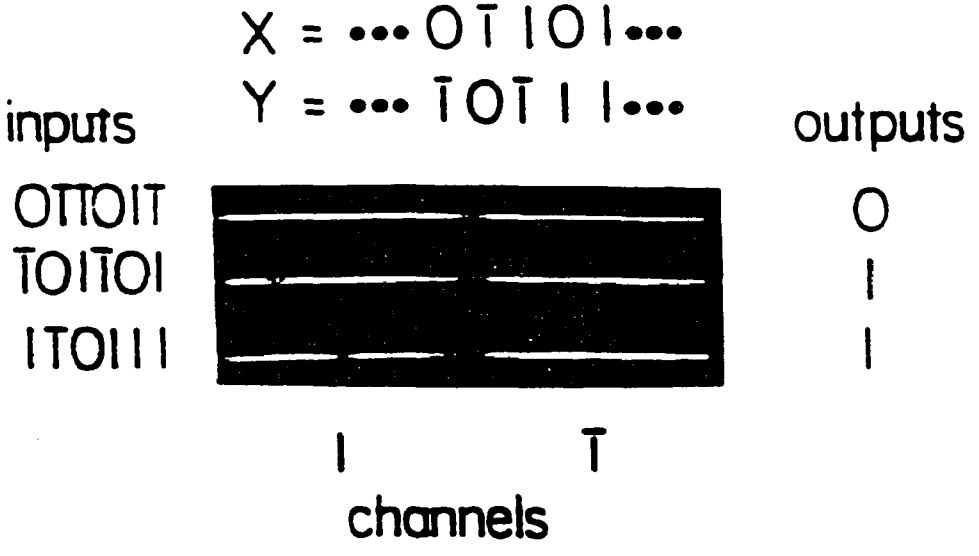
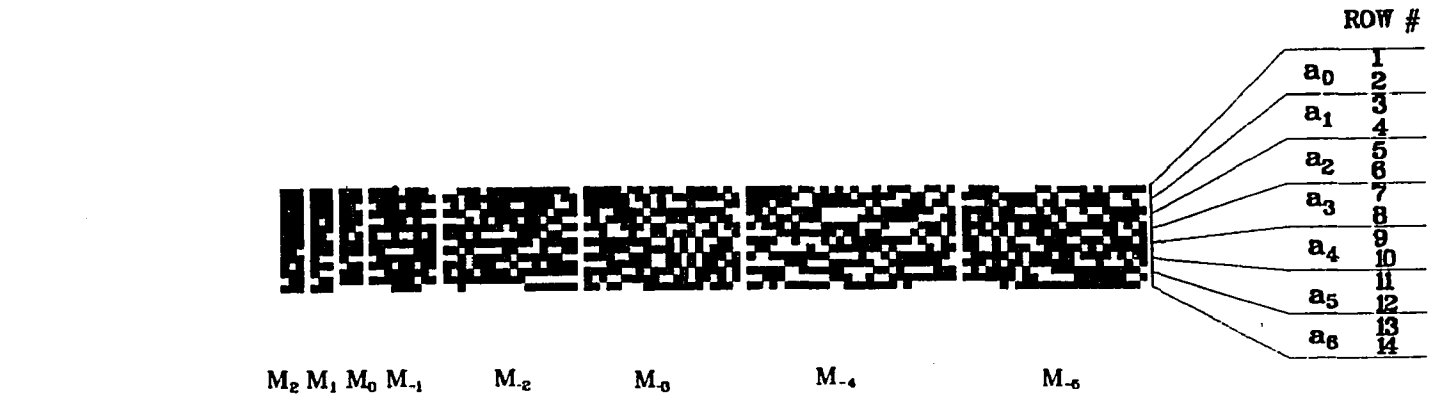
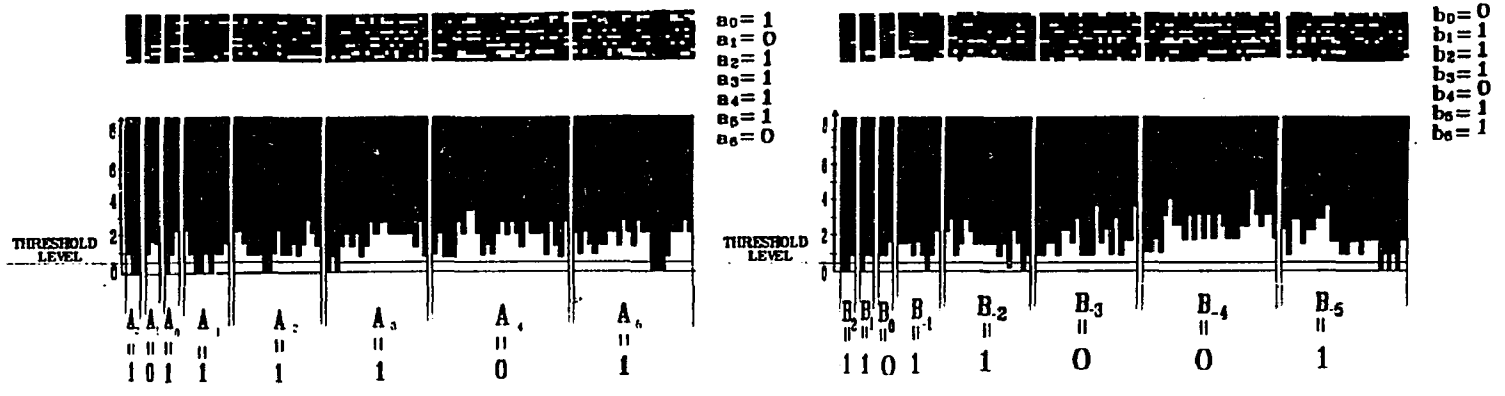


Fig. 4.7.5. Results of a three angularly multiplexed CAM processing. For this example, the two five-digit MSD numbers X and Y (top) are used. The inputs to the top, middle, and bottom CAM channels are the first, second, and last six-digit groups of X and Y, respectively. In the bottom part, the corresponding lens integrated results for the output channels 1 and $\bar{1}$ are shown. The three output digits are 0, 1 and 1, respectively.



(a)



(b)

Fig. 4.7.6. (a) The conversion masks for a 7-bit SLN multiplier (active low logic). (b) The CAM mask illumination and the corresponding intensity distribution. Two numbers, a=61 and b=110, are converted from a binary to an SLN representation.

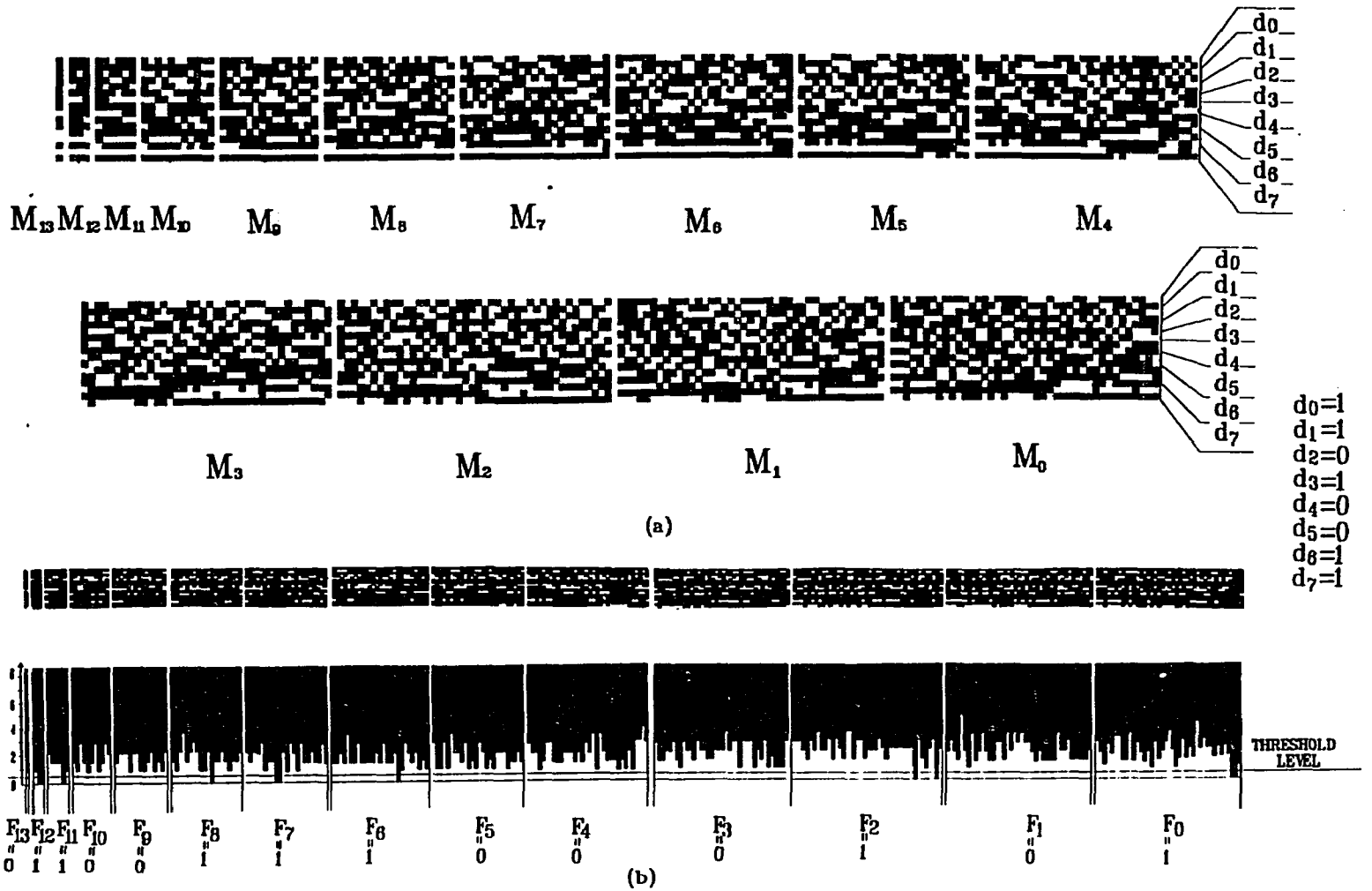


Fig. 4.7.7. (a) The CAM masks for an active low conversion from an 8-bit SLN (1100.1011) to a 14-bit BN representation. (b) The CAM mask illumination with the corresponding intensity distribution. For F_{12}, F_{11}, F_8, F_7, F_6, F_2, and F_0 corresponding to the binary result 01100111000101, the intensity below the threshold level is shown.

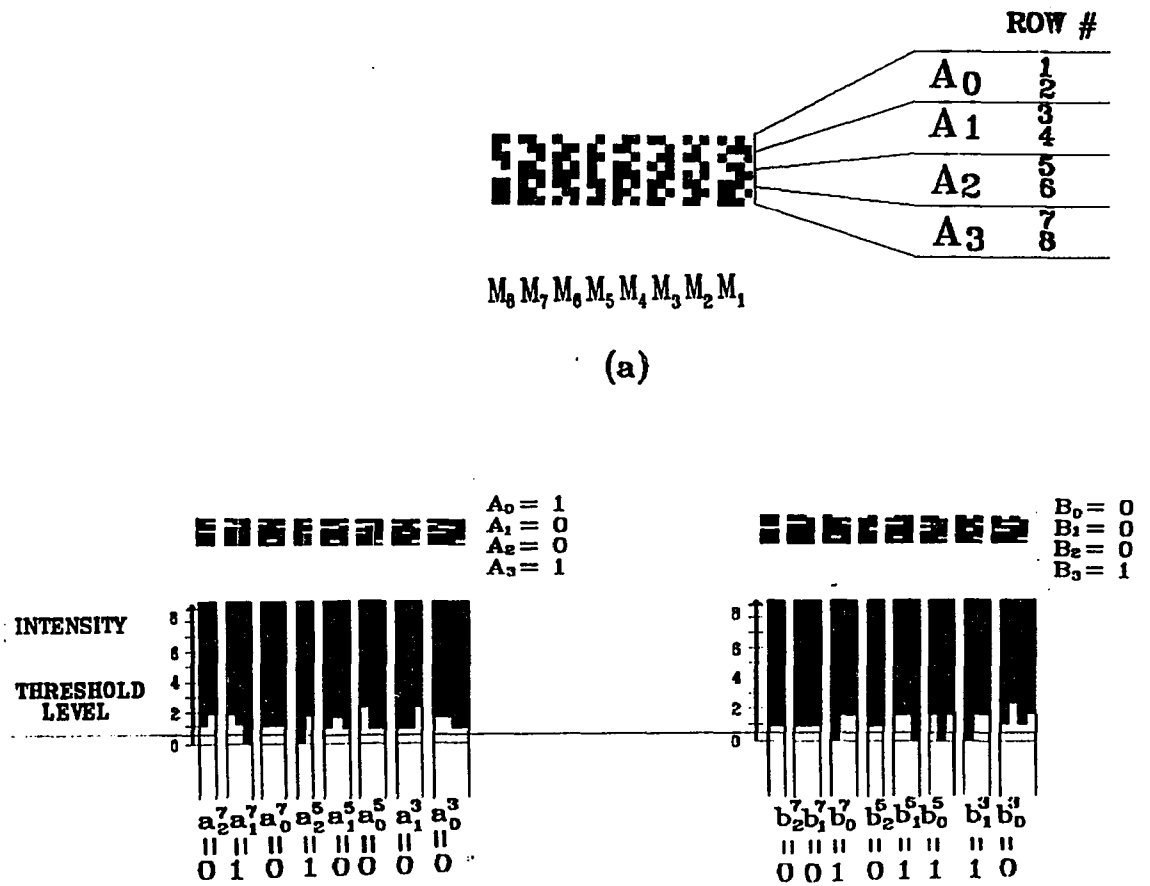


Fig. 4.7.8. (a) The conversion masks for a RN multiplier (active low logic) based on moduli 7, 5, 3. (b) The CAM mask illumination and the corresponding intensity distribution. Two numbers, a=9 and b=8, are converted from a binary to a RN representation.

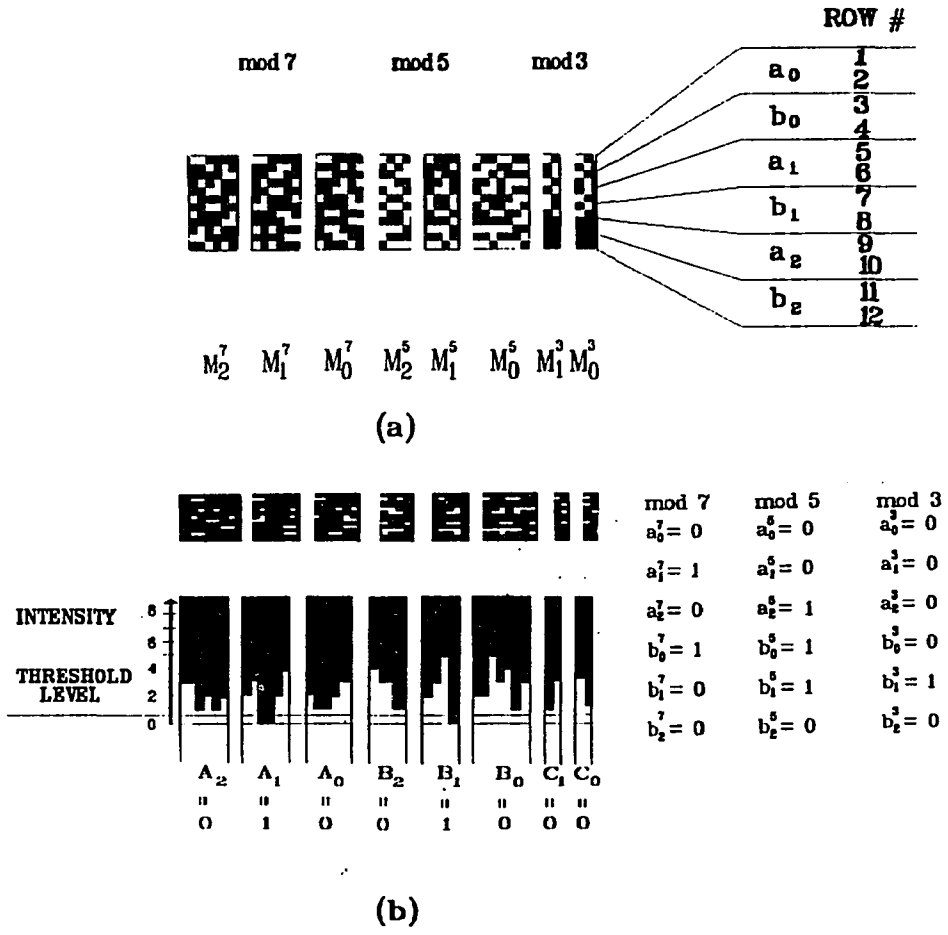


Fig. 4.7.9. (a) Experimental result of a residue multiplier using the moduli 3, 5 and 7. The masks encoded using dual-rail logic M_2^7, M_1^7, M_0^7 represent the modulus 7, masks M_2^5, M_1^5, M_0^5 represent modulus 5 while M_1^3, M_0^3 represent modulus 3 multiplication results. (b) The multiplication result for a 9X8. The eight masks were illuminated by a sequence of bars representing the binary numbers a = 010 and b = 001 in a mod 7, a=100 and b=011 in a mod 5, a=000 and b=010 in a mod 3 multiplications. The intensities are below the threshold level for the output bits A_1, B_1 corresponding to the RN 022 (decimal number 72).

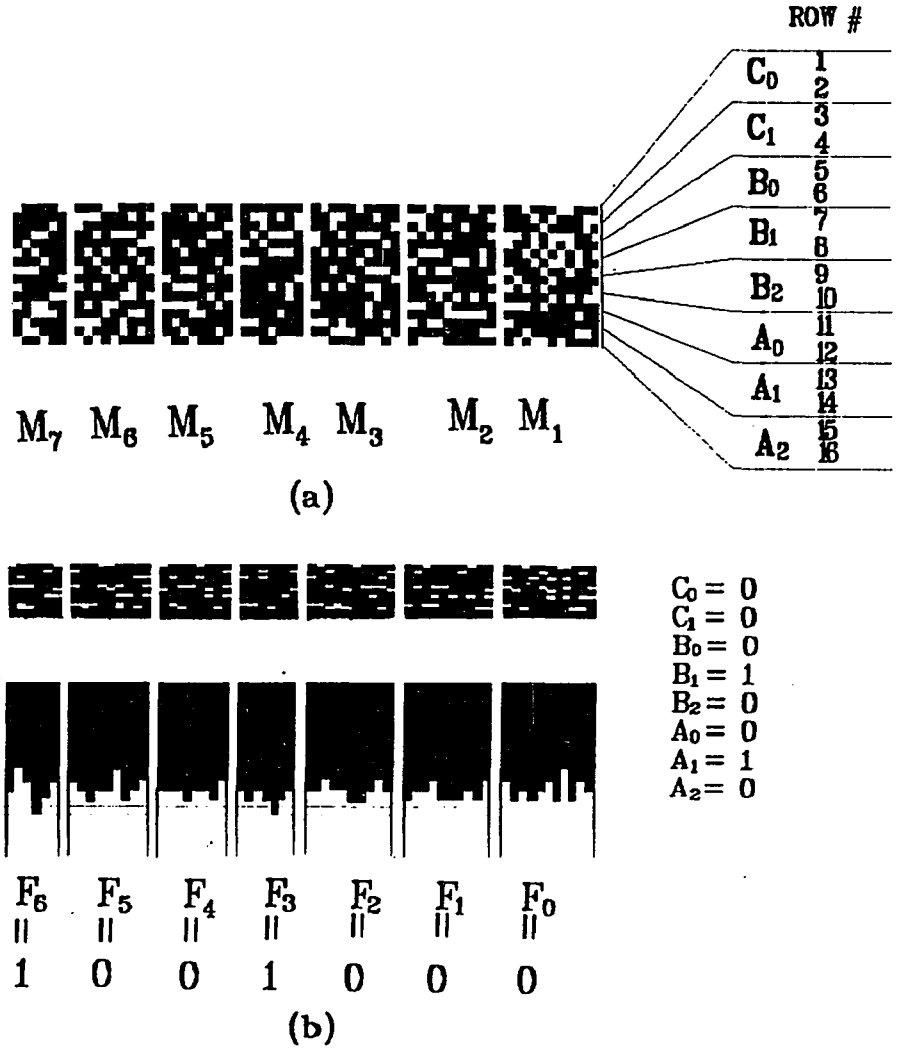


Fig. 4.7.10. (a) The conversion masks from RN representation to 7-bit BN representation. (b) The CAM mask illumination with the corresponding intensity distribution. For F_6 and F_3 corresponding to the binary result 1001000 (72 decimal), the intensity below the threshold level is shown.

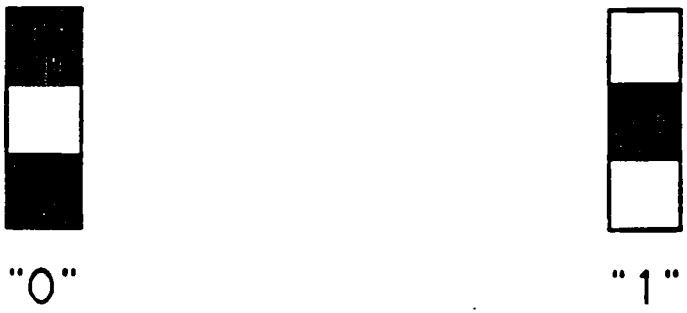


Fig.4.8.1 Spatially encoded primitive elements "0" and "1" using a Triple -rail (TR) encoding rule.

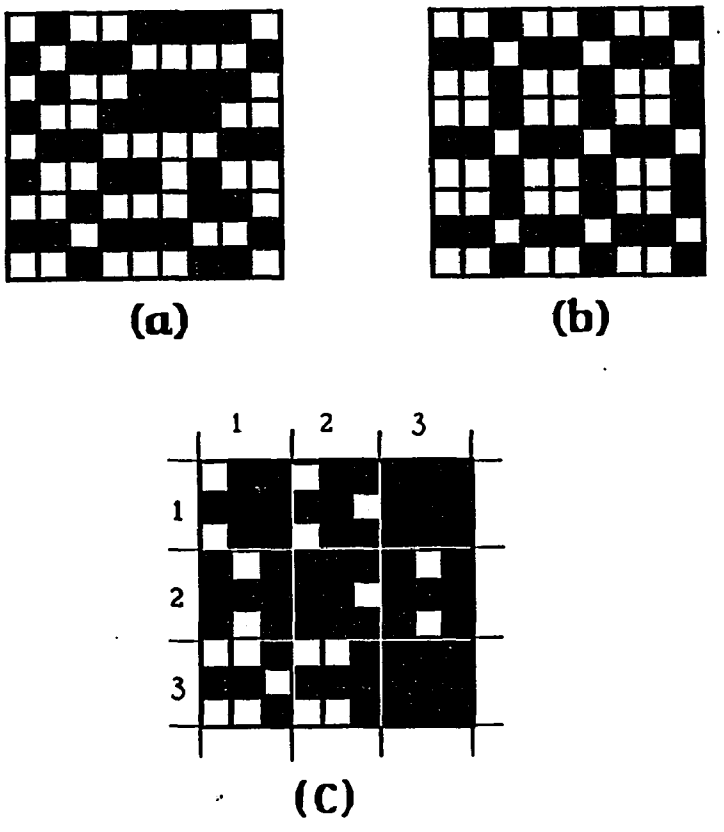


Fig.4.8.2. (a) Spatially encoded input data for a CAM based symbol recognition using TR encoding. (b) The corresponding CAM mask for the search symbol 001 consists of periodically repeated spatially encoded symbols 100 (inverse of the search symbol 001). (c) The result of a CAM recognizer. The location of the search symbol is indicated by the completely blocked area (3 by 3 pixels). The search symbol 001 is found in positions (1,3) and (3,3).

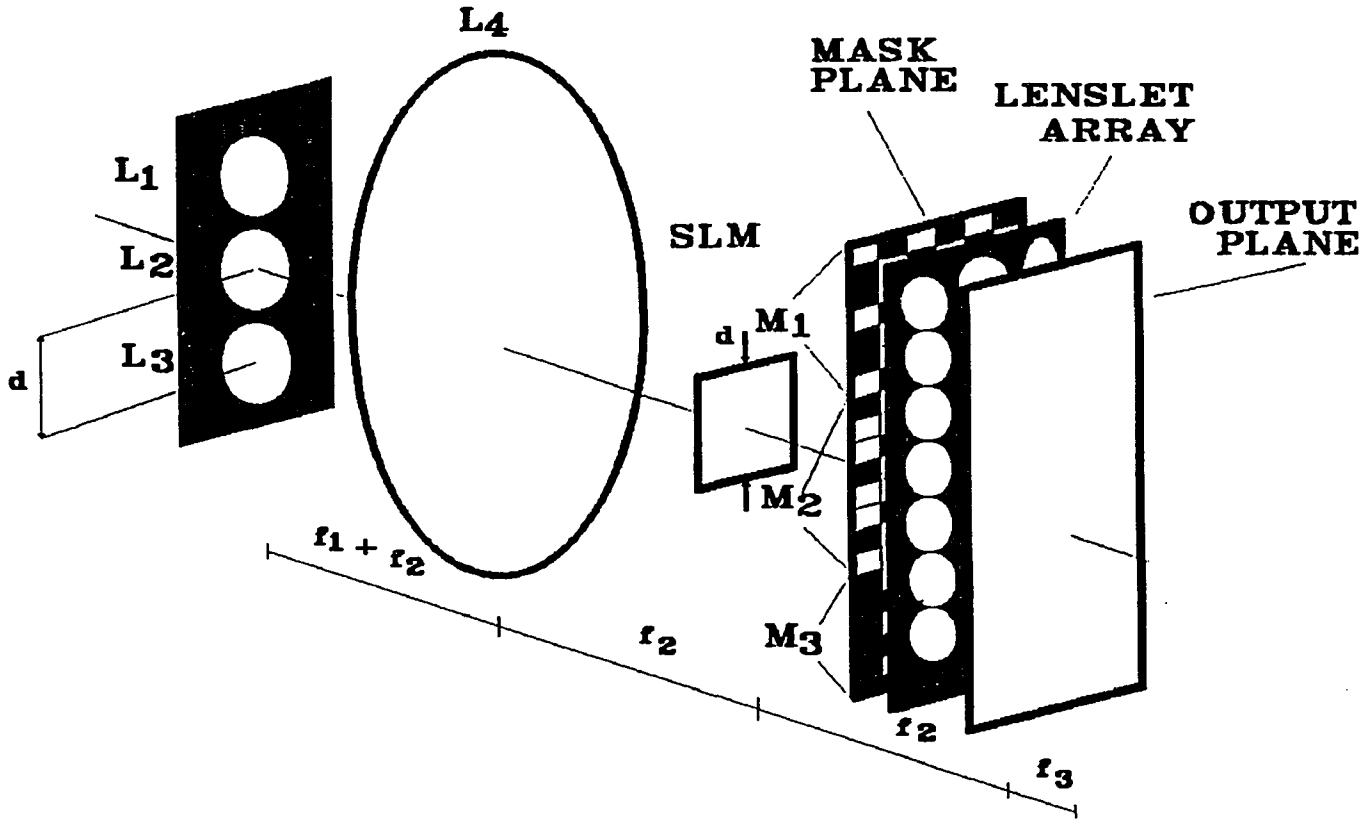


Fig.4.8.3. A CAM-based system for a three channel symbol recognition. L_1 - L_3 are lenses of a lenslet array. Lenses L_1 - L_3 together with projection lens L_4 generate three angle-multiplexed beams (each used to recognize a different search symbol). The input data is displayed on the SLM. The spacing d between two adjacent light sources is equal to the SLM dimension. To recognize three search symbols, three masks (M_1, M_2 and M_3) are placed on the mask plane. The lenslet array L_5 performs light intensity integration over each symbol's aperture.

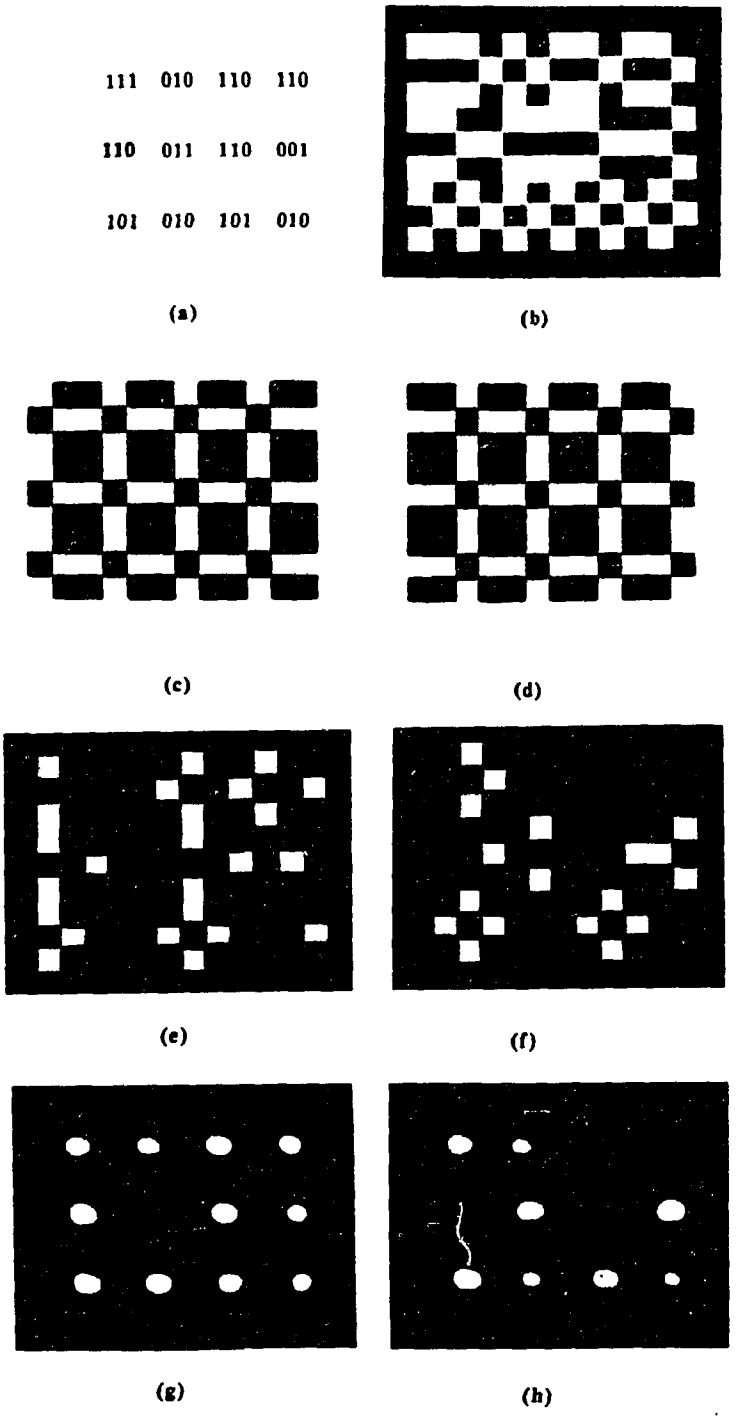


Fig. 4.8.4. Experimental results for a CAM based symbol recognition. (a) The binary input data. (b) The spatially encoded input data using a TR encoding rule. (c) The binary mask to recognize the search symbol 011. (d) the binary mask to recognize the search symbol 110. (E-f) A masked version of the input image used for the recognition of the search symbol 011 and 110, respectively. (G-h) The integration result for the case 011 and 110 for each of the 3 by 3 pixel areas. The low intensity pixels indicate the search symbol's position.

A	B	RESULT	FUNCTION
0	X	0	F_0
X	0	0	F_0
1	1	1	F_1
1	2	2	F_2
2	1	2	F_2
1	3	3	F_3
3	1	3	F_3
2	2	4	F_4
2	3	6	F_6
3	2	6	F_6
3	3	9	F_9

Table 4.2.1, The minimized truth for MVL variables, multiplication example.

V. ASSOCIATIVE MEMORY PROCESSING

5.1 Introduction

A combinatorial logic circuit is an interconnected array of logic gates or switches. However, for various arithmetic operations, iterative sequential computation is needed. To furnish feedback for combinatorial circuits, memory elements, such as flip-flops or registers, must be utilized. With this feedback, the overall logic circuit is a finite-state sequential logic machine. To perform fast combinatorial logic the use of optics was suggested [1-3]. However, for the proposed sequential logic schemes the efficient feedback generation is an active research area. To generate a sequential logic circuit, a viable hybrid approach is to use optics for fast parallel combinatorial logic and interconnect functions and also high-speed bit-addressable electronics for storage and feedback [1]. An example is the emitter-hologram-SLM-hologram-detector combination used for opto-electronic residue arithmetic processing [4]. In this thesis, a hybrid sequential computing module, where an optical array processor performs the combinatorial logic and interconnect operations between high-speed electronic parallel addressed storage registers, is described. This hybrid system is able to perform fast optical register transfer micro-operations (ORTMOs), operations that represent the primitive operations required for an general-purpose optical digital computer. Using these operations, other, such as residue arithmetic operations can be constructed. This new system will be referred to as an optical

register transfer processor (ORTP).

For the design of a digital computer, the so-called register transfer language (RTL) [5] plays an important role. Based on an interconnected set of logic gates, registers, etc., RTL serves as the most primitive language that links a physical digital machine and its programming environment. Any sophisticated iterative computation can be decomposed into a sequence of primitive logic and transfer operations. In Table 5.1, some typical register transfer micro-operations [5] are listed. Here, the source and destination registers are denoted as A and B, respectively. It can be shown [5] that for both a single bit and a full-length word, register parallel load, clear, rotate and shift as well as transfer operations are executable. The complete set of binary logic micro-operations (see Table 5.2) can be combined for other more sophisticated arithmetic operations, such as addition, subtraction, and multiplication [5].

5.2 Processor Architecture

For an optical register transfer micro-operation (ORTMO) implementation, the recently developed symbolic substitution technique [6-9] can be used. Here, an optical holographic associative symbolic substitution (OHASS) technique, proposed by Yu et. al. [8-9] is employed. The two logic states 0 and 1 are encoded as two spatial orthogonal symbols (see Fig.5.2.1). In the OHASS filter preparation stage, the interference pattern between the Fourier spectra of the input and the precalculated output symbols are recorded. To process either a single or two-variable ORTMO, at the correspondingly partitioned recording plane either

two or four exposures are affected. To generate an output, these input symbols are used as reference beams. The OHASS filter construction details can be found in Refs. [7-8].

For the various ORTMOs, in the input plane, a set of N-bit input optical symbols

$$c(x,y)=\sum_{i=0}^{N-1}\{a(x-ix_0,y-y_0)+b(x-ix_0,y-2y_0)+c^*(x-ix_0,y+y_0)\} \quad (5.1.1)$$

where (x_0,y_0) are spatial shifts, a, b and c^* are the two inputs and a reference pattern, respectively, is allowed to overlap with a pattern duplicating filter g(x,y). In the system's Fourier plane, the corresponding output pattern O is

$$\begin{aligned} O(x,n)&=C(x,\eta)oG(x,\eta)=\sum_{k=0}^{M-1}C(x,\eta-k\eta_1)\\ &=\sum_{i=0}^{N-1}\sum_{k=0}^{M-1}\{A(x-ix_0,\eta-k\eta_1)\exp[-jy_0(\eta-k\eta_1)]\\ &\quad+B(x-ix_0,\eta-k\eta_1)\exp[-j2y_0(\eta-k\eta_1)\\ &\quad+C^*(x-ix_0,\eta-k\eta_1)\exp[+jy_0(\eta-k\eta_1)\end{aligned} \quad (5.1.2)$$

where o denotes a convolution operation, A, B, C and C are the 1D Fourier transform of the input patterns. Eq.(5.1.2) represents a 1D (vertical) replica of a horizontal Fourier hologram array. The synthesis of a proper duplication filter G has been reported [10-12].

Based on the above-described OHASS and pattern duplication scheme, an optical register transfer processor (ORTP) can be implemented. In Fig.5.2.2, a schematic of an ORTP is shown. The four N-bit A, B, C, and C^* electronic registers are employed. To generate the optical inputs, input electronic registers

are used to drive a parallel 1D array of fast laser diode emitters. A fast detector array feeds into an ORTP output register. For iterative computing, an one-to-one electronic feedback loop connecting the registers A to C can be utilized. The register B also acquires input data from an electronic output port. In a learning phase, to load a reference pattern the register C^* is used. For the M possible N-bit micro-operations, using the M vertical Fourier spectrum replica, M consecutive associative holograms are prepared. To ensure that all the two-variable input combinations are available, for each hologram four OHASS bit-wise partitioned exposures are required. When all the M ORTMOs are encoded, the register C^* is deactivated. To control the ORTP's sequencing, located at the back of the hologram array, a 2D spatial light modulator (SLM) is programmed to select, one at a time, one of the M horizontal slices. The thus selected result, after passing through the second cylindrical lens L_2, is detected and stored in the register C.

Because an ORTP is a bit-serial word-parallel processor that processes in a sequential operation fashion two input variables at a time, optical fan-out is determined by the number of logic and transfer operations independent of the word length. It has been shown [13] that for a holographic interconnect to be superior to its electronic counterpart, the interconnect line-length, efficiency, rise-time, fan-out, and source threshold power must be optimized. It has been indicated [13] that for a given rise-time (e.g. 1 ns in a capacitance limited region), the use of long interconnect line-length (> 1 mm), large fan-out (> 1), low source threshold (< 2 mw) and high hologram efficiency (> 9 %) can provide an over-all superior interconnect performance for optics than for electronics.

For an existing optical semiconductor laser source with a maximum power of 50 mW operated with a 1 ns rise-time, a maximum fan-out of 67 is possible [13]. This fan-out places an upper limit on the number of ORTP micro-operations (more than what is required for primitive operations). Since all the registers store for a short period of time the parallel data and the intermediate results, and because no serial intra-register operations are required, fast GaAs-based GHz electronic registers together with a fast system clock can be used. For an all-optical ORTP, optical memory elements, such as the recently developed SEED [14] device, that can offer dynamic storage for as long as 30 sec., may be used. In addition, because all microoperations require an identical processing circuitry, clock synchronization is relatively easy. Finally, because of the ORTP speed is independent of the word length, by using a large space-bandwidth product optical system, the implementation of a multi-variable ORTP may also be possible.

5.3 Experimental Results

In our proof-of-principle ORTP experiment, various parallel OHASS bit transfer, logic complement and AND operations were performed. For the duplication of the Fourier spectrum into three laterally displaced spatial locations, two beam splitters and a mirror were used. These three Fourier spectra were used, for bit transfer, logic complement and AND micro-operations, respectively. For each operation there are k possible input bit configurations where for transfer $k=2$ and for logic $k=4$. Corresponding to a particular

operation, a hologram was partitioned into k sections. Depending on which operation, transfer or logic, is to be performed, a k step hologram generation process is used, where in each step the contents of registers A and C^* or A, B, and C^* are used. During the associative recall process, for either the transfer or logic, the entire hologram was illuminated by either A or A and B. Using the associative recall process, at the output register C plane, the result was retrieved. In Fig.5.3.1(a) and (b), the result of these transfers are shown. To select the desired microoperation, at the Fourier plane, a binary mask was employed. On the left-most Fourier spectrum, an optical bit transfer micro-operation was performed. Since there are two bit transfer cases, i.e the transfer of either a 0 or a 1, the hologram associated with these two transfers was divided into two vertical parts. At each exposure, two identical input symbols taken from the input registers A and C^* were used. For a bit complement logic operation, the central Fourier spectrum was used. In this case, for each exposure, a different pair of binary symbols were used. In Fig.5.3.2(a) and (b), the two OHASS logic complement results are shown. To perform a two-variable logic AND operation, the right-most Fourier spectrum was utilized. In this case, a four quadrant composite hologram was constructed. At each exposure, the three other spectral quadrants were covered. Also, for the four exposures, the four input symbol pairs were inserted. In Fig.5.3.3, an experimental OHASS logic AND results are shown.

5.4 Summary

In this thesis, an ORTP that promises fast, bit-serial and word parallel iterative operation on binary input data is proposed. For the ORTP architecture, an OHASS operation that is based on a coherent parallel Fourier processing system was described. The processor is based on holographic associative memory and symbolic substitution. The input data is encoded using orthogonal spatial symbols. Two stage process was proposed. First, the associative spatially multiplexed hologram was generated using all input-output pairs, next, the hologram was illuminated by input signal only. At the output plane both input and retrieved output symbols were present. The associative memory technique was used to design a bit-serial, word-parallel processor. The processor is capable of performing a set of transfer and logic operations. The limiting factors for the processing speed are LD maximum modulation rate, free space traveling time and detector response time. All these factor can be minimized to provide an overall processing speed <1ns. First order experimental results for transfer, complement and logic AND operations were also presented.

5.5 References

[1] R. Arrathoon, "Logic based spatial light modulators," Proc. SPIE, 881 230-239 (1988).

[2] P. S. Guilfoyle and W. J. Wiley, "Combinatorial logic based digital optical computing," Appl. Opt. 27, 1661-1673 (1988).

[3] Y. Li, A. Kostrzewski, D. H. Kim, and G. Eichmann, "A compact folded path free-space optical programmable logic array," Opt. Lett. 13, 895-897 (1988).

[4] S. F. Habiby and S. A. Collins, Jr., "Implementation of a fast digital optical matrix-vector multiplier using a holographic lookup table and residue arithmetic," Appl. Opt. 26, 4639-4651 (1987).

[5] M. M. Mano, Computer System Architecture, (Prentice-Hall, NJ, 1982) ch.4.

[6] K. H. Brenner, A. Huang, and N. Streibl, "Digital optical computing with symbolic substitutions," Appl. Opt. 25, 3054-3060 (1986).

[7] Y. Li, G. Eichmann, R. Dorsinville, and R. R. Alfano, "An AND operation-based symbolic pattern recognizer," Opt. Comm. 63, 375-379 (1987).

[8] F. T. S. Yu and S. Jutamulia, "Implementation of symbolic substitution logic using optical associative memories," Appl. Opt. 26, 2293-2294 (1987).

[9] F. T. S. Yu, C. Zhang and S. Jutamulia, "Applications of one-step holographic associative memory to symbolic substitution," Opt. Eng. 27, 399-402 (1988).

[10] L. P. Boivin, "Multiple imaging using various types of simple phase gratings," Appl. Opt. 11, 1782-1792 (1972).

[11] P. Matthijsse, "Multiple imaging with thin phase filters: a signal processing approach," J. Opt. Soc. Am. 68, 733-738 (1978).

[12] W. H. Lee, "High efficiency multiple beam gratings," Appl. Opt. 18, 2152-2158 (1979).

[13] M. R. Feldman, S. C. Esener, C. C. Guest, and S. H. Lee, "Comparison between optical and electronical interconnects based on power and speed considerations," Appl. Opt. 27, 1742-1751 (1988).

[14] G. Livescu, D. A. B. Miller, J. E. Henry, A. C. Gossard, and J. H. English, "Spatial light modulator and optical dynamic memory using a 6 X 6 array of self electro-optic-effect devices," Opt. Lett. 13, 297-299 (1988).

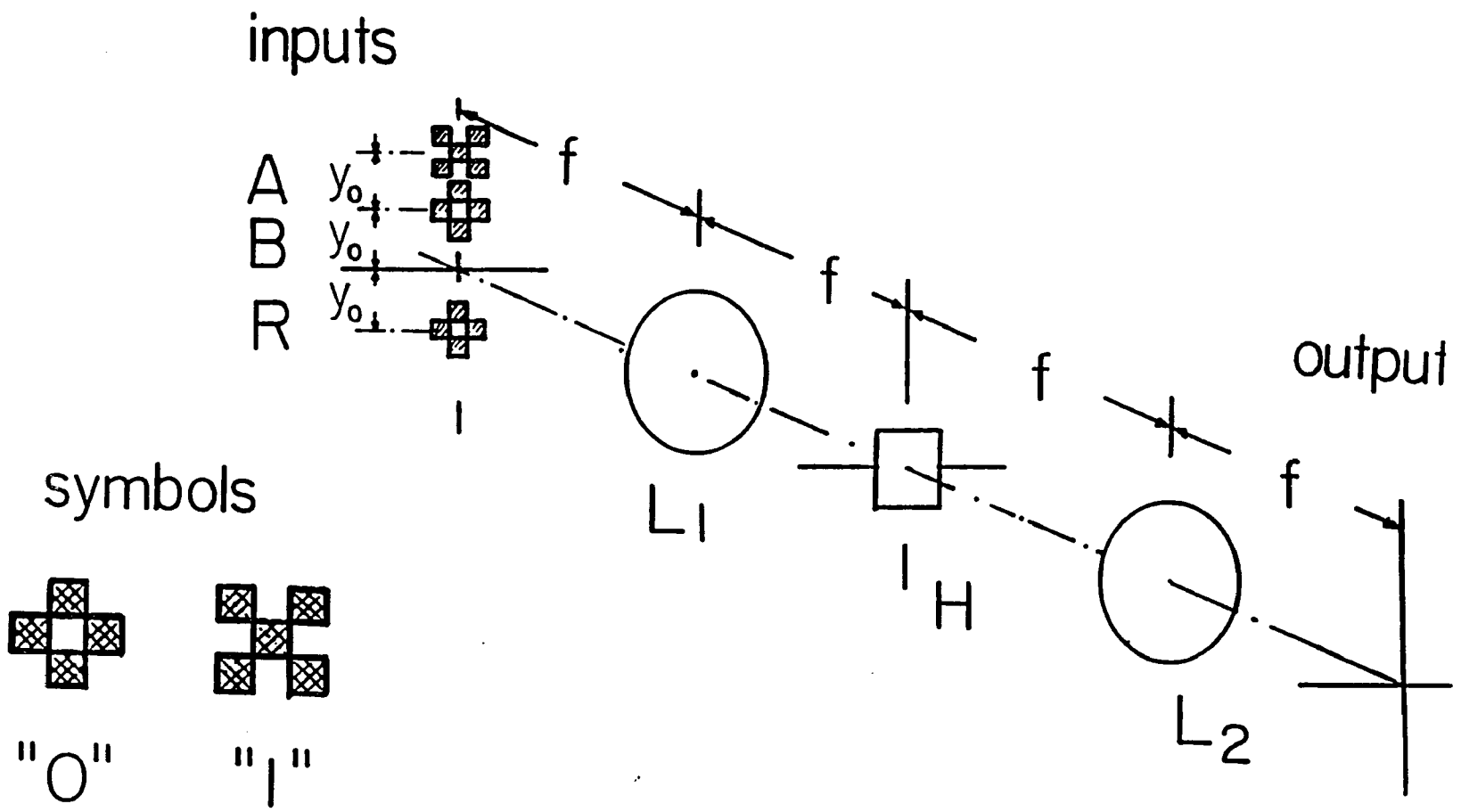


Fig. 5.2.1. A schematic of a 1-bit OHASS processor. Input binary symbols are encoded orthogonal patterns. L_1, L_2 and H, are two identical Fourier lenses and a hologram. The two logic and a reference symbols are inserted at the input plane. For the recording of the four associative subholograms, the Fourier spectrum is divided into four quadrants. For logic processing, depending on the inputs, at the system's output plane an associated output will be detected.

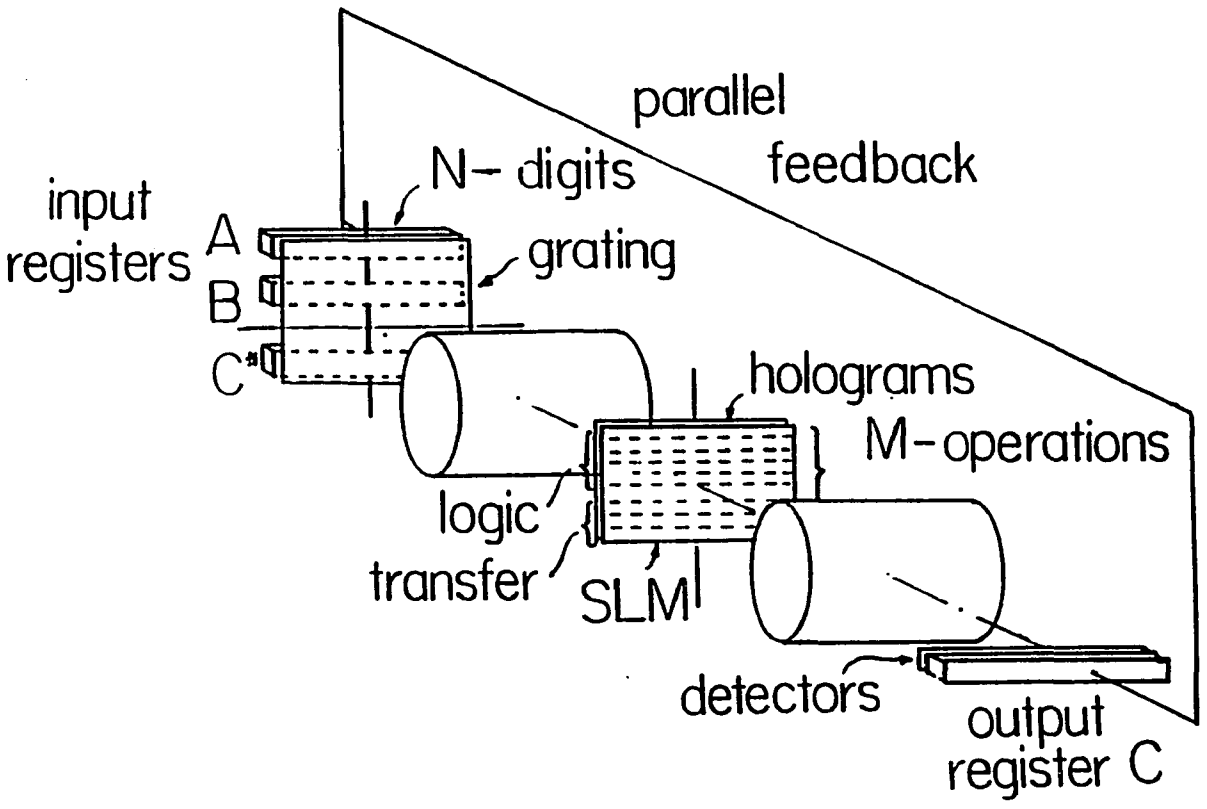


Fig. 5.2.2. A schematic of a N-bit OHASS iterative processor. A, B and C*, are three N-bit input registers driving channelized laser diodes; C, an N-bit output register storing the result of optical threshold detector array. In addition to the lenses, holograms, and an input duplication grating, a Fourier plane 2D SLM and a parallel electronic feedback are used.

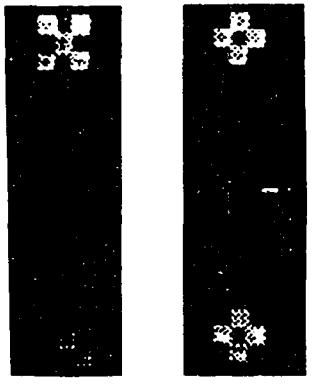


Fig. 5.3.1. Results of a 1-bit OHASS interregister microoperation, (a) and (b), the associative transfer of symbol 1 and 0, respectively. The top and bottom patterns a the input and output symbols, respectively.

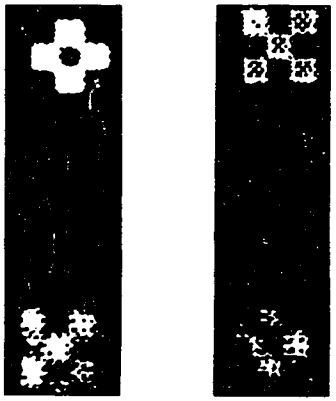


Fig. 5.3.2. Results of a 1-bit OHASS complement microoperation, (a) and (b), the associative complement of symbol 1 and 0, respectively. The top and bottom patterns a the input and output symbols, respectively.

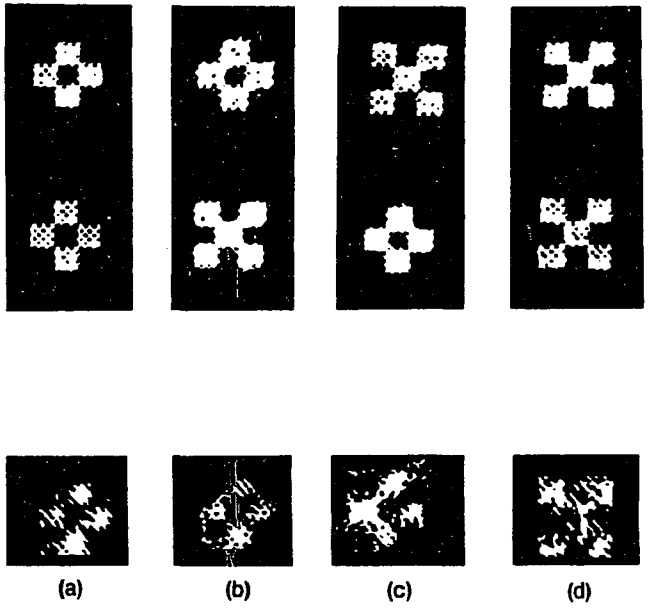


Fig. 5.3.3. Results of a 1-bit OHASS logic AND microoperation, (a) - (d), the associative logic AND operation result of the four input binary symbol pairs.

Microoperation	Explanation
$C \leftarrow A$	Transfer A into C
$C \leftarrow srA$	Shift A right by 1-bit and transfer into C
$C \leftarrow slA$	Shift A left by 1-bit and transfer into C
$C \leftarrow rrA$	Rotate A right by 1-bit and transfer into C
$C \leftarrow rlA$	Rotate A left by 1-bit and transfer into C
$C_i \leftarrow A_j$	Transfer j^{th} bit of A into i^{th} bit of C

Table 5.1 Inter-register transfer micro-operations.

Binary logic	Microoperation	Explanation
$O_0 = 0$	$C \leftarrow 0$	Reset
$O_1 = 1$	$C \leftarrow 1$	Set
$O_2 = A$	$C \leftarrow A$	A
$O_3 = B$	$C \leftarrow B$	B
$O_4 = \overline{A}$	$C \leftarrow \overline{A}$	Complement
$O_5 = \overline{B}$	$C \leftarrow \overline{B}$	Complement
$O_6 = A \bullet B$	$C \leftarrow A \bullet B$	AND
$O_7 = A \bullet \overline{B}$	$C \leftarrow A \bullet \overline{B}$	Inhibition
$O_8 = \overline{A} \bullet B$	$C \leftarrow \overline{A} \bullet B$	Inhibition
$O_9 = A + B$	$C \leftarrow A + B$	OR
$O_{10} = A + \overline{B}$	$C \leftarrow A + \overline{B}$	Implication
$O_{11} = \overline{A} + B$	$C \leftarrow \overline{A} + B$	Implication
$O_{12} = A + B$	$C \leftarrow A + B$	XOR
$O_{13} = A + B$	$C \leftarrow A + B$	XNOR
$O_{14} = \overline{A \bullet B}$	$C \leftarrow \overline{A \bullet B}$	NAND
$O_{15} = \overline{A + B}$	$C \leftarrow \overline{A + B}$	NOR

Table 5.2 Register logic micro-operations.

VI. SUMMARY

In this thesis, memory-based approach for optical computing was discused. Three types of optical memories: location addressable, content addressable and associative memories were used. Opto-electronic processors based on binary, residue, modified signed-digit and sign/logarithm number systems were presented. The application of LAM-based processor for residue number (RN) arithmetic as well as for multi-value logic (MVL), including implementation of modified signed-digit (MSD) number system 3-level gates, has been studied. Using this approach fast processing speed (LD matrix as a crossbar switch) or high dynamic range calculations (cascade of LCTVs) can be obtained. However, when using a binary number system, the truth-table size depends on the input's dynamic range. Thus for large numbers, this scheme may become impractical. Using a residue number system, there are methods to decompose an arithmetic operation into a set of parallel suboperations. The application of LAM-based processing for RN operations was described. The use of the LAM is limited to two variables only. For multi-variable function implementation, an optical noncoherent CAM can be used.

Using a non-holographic CAM, any arithmetic operation representable as a Boolean sum of product terms can be implemented. To encode a logic function into a CAM mask, either an active low or high level logic can be applied. The storage capacity is sufficient to implement 16-bit arithmetic-logic processor (32-variable processor). To utilize storage space efficiently, prior to the optical implementation,

the logic functions are minimized using Quine-McCluskey method. Single operation as well as multi-operation architectures were presented. For MSD number processor, angularly multiplexed architecture which utilizes only one CAM mask to generate all outputs, was presented. As an example of the non-holographic CAM based arithmetic processing, the design of an optical carry look-ahead adder (CLA); binary number (BN), sign/logarithm number (SLN) and residue number (RN) multipliers was described. The proposed CAM-based processor promises both large memory capacity and high processing speed (<1ns).

To perform various arithmetic operations in an iterative, sequential way a register-transfer processor was introduced. A combinatorial logic circuit is an interconnected array of logic gates or switches. To generate a sequential logic circuit, a viable hybrid approach is to use optics for both fast parallel combinatorial logic and interconnect functions and high-speed bit-addressable electronics for storage and feedback. In this thesis, a hybrid sequential computing module, where an optical array processor performs the combinatorial logic and interconnect operations between high-speed electronic parallel addressed storage registers, is described. This hybrid system is able to perform fast optical register transfer micro-operations, that represent the primitive operations required for a general-purpose optical digital computer. Based on the set of primitive operations, algorithms can be constructed to performs more complicated operations such as n-bit arithmetic-logic operations. Also, other, such as RN or MSD arithmetic operations can be constructed. To implement an optical register transfer micro-operation processor, associative memory, symbolic substitution technique was used.

VII. LIST OF THE THESIS RELATED PUBLICATIONS

Publications in Journals:

1. G. Eichmann, A. Kostrzewski, B. Ha and Y. Li, "Parallel Optical Pyramidal Image Processing", Opt. Lett., Vol. 13, 431 (1988).

2. Y. Li, A. Kostrzewski, D. H. Kim and G. Eichmann,"Free-Space Folded-Path Optical Programmable Logic Array", Opt. Lett., Vol. 13, 895 (1988).

3. A. Kostrzewski, D. H. Kim, Y. Li, B. Ha and G. Eichmann, "Optical Position Coded Residue Processor using Inexpensive LCTV Devices", Appl. Opt., Vol. 28, 415 (1989).

4. G. Eichmann, A. Kostrzewski, D. H. Kim and Y. Li, "Optical Parallel Register Transfer Micro-Operations using a Holographic Associative Symbolic Substitution", Appl. Opt., Vol. 28, 3860 (1989).

5. Y. Li, B. Ha, A. Kostrzewski, D. H. Kim and G. Eichmann, "Optical Position-Coded Multiple-Valued Logic and Arithmetic using Liquid-Crystal TVs and Holograms", Opt. Comm., Vol. 70, 379 (1989).

6. Y. Li, A. Kostrzewski, D. H. Kim and G. Eichmann,"A Liquid-Crystal-TV-Based White Light Optical Tracking Novelty Filter", Appl. Opt., Vol 28, 4861 (1989).

7. Y. Li, A. Kostrzewski, D. H. Kim and G. Eichmann,"A Compact Parallel Real-Time Programmable Optical Morphological Image Processor", Opt. Lett., Vol. 14, 981 (1989).

8. Y. Li, D. H. Kim, A. Kostrzewski and G. Eichmann,"Content Addressable Memory Based Single-Stage Optical Modified Signed Digit Arithmetic", Opt. Lett., Vol 14, 1254 (1989).

9. G. Eichmann, A. Kostrzewski, D. H. Kim and Y. Li, "Optical Higher-Order Symbolic Recognition", Appl. Opt., Vol. 29, 2135 (1990).

10. A. Kostrzewski, Y. Li, D. H. Kim and G. Eichmann, "Fast Hybrid Parallel Carry Look-Ahead Adder", Opt. Lett., 13, (1990).

11. A. Kostrzewski, G. Eichmann , D. H. Kim and Y. Li, "Fast Optical Binary Multiplication using a Sign/Logarithm Number System", Submitted to Opt. Lett. (1990).

12. A. Kostrzewski, G. Eichmann , Y. Li and D. H. Kim, "Parallel Optical Content Addressable Memory Arithmetic Multi-Processor", Submitted to Int. J. Of Optical Computing (1990).

Conference proceedings:

13. G. Eichmann, A. Kostrzewski and Y. Li,"Linear Associative Mapping for Optical Residue Computing", OSA Technical Digest, Vol. 22, (1987).

14. G. Eichmann, A. Kostrzewski, D. H. Kim and Y. Li,"Parallel Optical Pyramidal Image Processing", SPIE Vol. 939, Hybrid Image and Signal Processing, 93 (1988).

15. Y. Li, A. Kostrzewski, D. H Kim and G. Eichmann,"Compact Free-Space Optical Programmable Logic Array", OSA Technical Digest, (OPTCON' 88).

16. A. Kostrzewski, D. H. Kim, Y. Li, B. Ha and G. Eichmann, "Optical Liquid Crystal TV and Hologram-Based Position-Coded Multiple-Valued Logic and Arithmetic Operations", OSA Technical Digest, (OPTCON' 88).

17. G. Eichmann, A. Kostrzewski, D. H. Kim and Y. Li, "An Optical-Holographic-Associative-Memory-Based Parallel Register Transfer Processor", OSA Optical Computing 1989 Technical Digest Series Vol. 9, 240 (1989).

18. Y. Li, A. Kostrzewski, D. H. Kim and G. Eichmann, "Optoelectronic Content Addressable Memory-Based Modified Signed Digit Arithmetic", OSA Technical Digest, Vol. 18, (OPTICS'89)

19. Y. Li, A. Kostrzewski, D. H. Kim and G. Eichmann, "Real Time Programmable Optical Morphological Filter", OSA Technical Digest, Vol. 18, (OPTICS'89).

20. Y. Li, A. Kostrzewski, D. H. Kim and G. Eichmann, "Optical Pyramidal Processing Based-On Noncoherent Tracking Novelty Filter", SPIE Annual Meeting, (San Diego, CA., 1989), SPIE proceedings on Hybrid Image Processing.

21. Y. Li, A. Kostrzewski, D. H. Kim and G. Eichmann, "Real-Time Optical Morphological Image Processor Based-on Programmable Lenslet Array", Gordon Research Conference on Holography and Optical Information Processing (Plymouth, N.H., 1989).

22. Y. Li, A. Kostrzewski, D. H. Kim and G. Eichmann, "A Non-Coherent Optical Tracking Novelty Filter", Gordon Research Conference on Holography and Optical Information Processing (Plymouth, N.H., 1989).

23. G. Eichmann, A. Kostrzewski, D. H. Kim and Y. Li, "Hybrid higher order optical symbolic recognition", SPIE's Technical Symposium on High Power Laser and Optical Computing, Vol. 1215 (1990).

24. Y. Li, A. Kostrzeski, D. H. Kim and G. Eichmann, "Hybrid content addressable memory MSD arithmetic", SPIE's Technical Symposium on High Power Laser and Optical Computing, Vol. 1215 (1990).

25. Y. Li, A. Kostrzewski, G. Eichmann and D. H. Kim, "Non-Holographic Content Addressable Memory Based Arithmetic Processor", SPIE proceeding Vol. 1230-358, (1990).

26. A. Kostrzewski, Y. Li, D. H. Kim, B. Ha and G. Eichmann,"Fast Optical Digital Arithmetic Processors", SPIE Proceeding Vol. 1296, Advanced in Optical Information Processing IV, (1990).

27. G. Eichmann, Y. Li, G. Colef and A. Kostrzewski, "Hybrid Optical Computation Module for a Real-Time Microwave Adaptive Array", IEEE Princeton Section Sarnoff Symposium, (1990).

28. A. Kostrzewski, Y. Li, G. Eichmann and D. H. Kim, "A Sign/Logarithm Number System-based Fast Optical Multiplier", will be presented at the 1990 OSA Annual Meeting.

29. A. Kostrzewski, D. H. Kim, Y. Li and G. Eichmann, "An Optical Carry Look-Ahead Adder Based On a Content Addressable Memory", will be presented at the 1990 OSA Annual Meeting.

30. G. Eichmann, A. Kostrzewski, D. H. Kim and Y. Li, "Optical Higher-Order Symbolic Substitution", will be presented at the 1990 OSA Annual Meeting.

7.1 . BIBLIOGRAPHY

Arrathoon, R.,and S. Kozatis, "Architectural and Performance Consideration for a 10^7-Instructions/sec Optoelectronic Central Processing Unit," Opt. Lett. 12, 956 (1987).

Arrathoon, R.,"Logic based spatial light modulators," Proc. SPIE, 881 230-239 (1988).

Atkins, D. E., IEEE, Trans. Comput. C-19, 720 (1970).

Avizienis, A., IRE Trans. Electronic Computers, EC-10, 389 (1961).

Bartelt, H., A. W. Lohmann and E. E. Sicre, J. Opt. Soc. Am. A, 1, 944 (1984).

Bocker, R. A., B. L. Drake, M. Lasher, and T. B. Handerson, Appl. Opt. 25, 2456 (1986).

Boivin, L. P., "Multiple imaging using various types of simple phase gratings," Appl. Opt. 11, 1782-1792 (1972).

Brenner, K. H., "Programmable Optical Processor based on Symbolic Substitution," Appl. Opt, 27 1687-1691 (1988).

Brenner, K. H., A. Huang, and N. Streibl, "Digital optical computing with symbolic substitutions," Appl. Opt. 25, 3054-3060 (1986).

Brenner, K. H., A. W. Lohmann and T. M. Merklein, "Symbolic Substitution Implemented by Spatial Filtering Logic," Opt. Eng. 28, 390-395 (1989).

Brubaker, T. A., and J.C. Becker, IEEE Trans. Comput.,24, 761, (1974).

Casasent, D., and E. Botha, "Optical Symbolic Substitution for Morphological Transformations," Appl. Opt. 27, 3806-3810 (1988).

Casasent, D. P., and E. C. Botha, "Multifunctional Optical Processor based on Symbolic Substitution," Opt. Eng. 28, 425-433 (1989).

Caulfield, H. J., Optical Computing, Editor SPIE Milestones series, vol. 1142, 1989.

Cavanagh, J. J., Digital Computer Arithmetic. New York: McGraw-hill, 1984.

Combet, M., H. Van Zonneveld, and L. Verbeek, "Computation of the base two logarithm of binary numbers," IEEE Trans, Electronic Computers, vol. EC-14, 863, (1965).

Dao, T. T., E. J. McCluskey, and L. K. Russell, Opt. Eng. 25, 14 (1986).

Drake, B. L., R. P. Bocker, M. E. Lasher, R. H. Patterson and W. J. Miceli, Opt. Eng. 25, 38 (1986).

Eichmann, G., and S. Basu, "Parallel Optical Syntactic Pattern Recognizer," Appl. Opt. 26, 1859-1865 (1987).

Eichmann, G., Y. Li, and R. R. Alfano, Appl. Opt. 25, 3113 (1986).

Eichmann, G., A. Kostrzewski, D. H. Kim and Y. Li, "Optical Parallel Register Transfer Micro-operations using Holographic Symbolic Substitution," Appl. Opt. 28, 3860-3863 (1989).

Eichmann, G., J. Zhu and Y. Li, "Optical Parallel Image Skeletonization using Content-addressable Memory-based Symbolic Substitution," Appl. Opt. 27, 2905-2911 (1988).

Flores, I., The Logic of Computer Arithmetic. New York: Van Nostrsand, 1955.

Guilfoyle, P. S., and W. J. Wiley, "Combinatorial logic based digital optical computing," Appl. Opt. 27, 1661-1673 (1988).

Goodman, S. D., and W. T. Rhodes, "Symbolic Substitution Applications to Image Processing," Appl. Opt. 27, 1708-1714 (1988).

Goutzoulis, A. P., "Comparison of Laser Diode Electronic Residue Position-Coded Look-Up Tables," Opt. Eng. 28, 373 (1989).

Goutzoulis, A. P., D. K. Davis and E. C. Malarkey, "Prototype position-coded look-up table using laser diodes," Opt. Comm. 61, 302 (1987).

Goutzoulis, A. P., "Complexity of residue position-coded lookup table array processor," Appl. Opt. 26, 4823 (1987).

Guest, C. C., and T. K. Gaylord, "Truth-Table Look-Up Optical Processing Utilizing Binary and Residue Arithmetic, " Appl. Opt. 19, 1201 (1980).

Feldman, M. R., C. Esener, C. C. Guest, and S. H. Lee, "Comparison between optical and electronical interconnects based on power and speed considerations," Appl. Opt. 27, 1742-1751 (1988).

Habiby, S. H., and S. A. Collins, Jr., "Implementation of a fast digital optical matrix-vector multiplier using a holographic lookup table and residue arithmetic," Appl. Opt. 26, 4639-4651 (1987).

Harata, Y., Y. Nakamura, H. Nagase, M. Takigawa and N. Takagi, Proc. ICCD'84, Oct. 1984.

Henkel, H., IEEE Trans. ASSP, 37, 301 (1989).

Hill, F. J., and G.R. Peterson, Switching Theory and Logical Design, John Wiley & Sons, New York (1984).

Huang, A., "Parallel Algorithms for Optical Digital Computers," Proceeding of the Tenth International Optical Computing Conference, 13-17 (IEEE Computer Society, Los Angeles, 1983).

Huang, A., Y. Tsunida, J. Goodman and S. Ishihara, "Optical computation using residue arithmetic," Appl. Opt. 18, 149 (1979).

Hughes, K. D., S. K. Rogers, J. P. Mills, and M. Kabrisky, "Optical preprocessing using liquid crystal televisions," Appl. Opt. 26, 1042 (1987).

Hurst, S. L., IEEE Trans. Comput. C-33, 1160 (1984).

Hwang, K., Computer Arithmetic/Principles, Architecture, and Design. (Wiley, New York, 1979).

Hwang, K. H., and A. Louri, "New Symbolic Substitution Algorithms for Optical Arithmetic using Signed-Digit Representation," Proc. SPIE, 880 (1988).

Hwang, K. H., and A. Louri, "Optical Multiplication and Division using Modified-Signed-Digit Symbolic Substitution," Opt. Eng. 28, 364-372 (1989).

Hwang, K., and A. Louri, Opt. Eng. 28, 364 (1989).

Ichioka, Y., and J. Tanida, Proc. IEEE, 72, 787 (1984).

Jin, R., C. Hanson, A. Cavez-Prison, H. M. Gibbs, G. Khitrova, N. Peyghambarian, T. Bowen, F. Y. Juang, P. K. Bhattacharya, D. H. Weinberger, K. R. Evans, C. E. Stutz, R. L. Jones, "Direct Fiber-Etelon-Fiber Interfacing," Opt. Eng. 28, 340, (1989).

Jin, Y., and F.T.S. Yu, "Optical Binary Adder Using Liquid Crystal Television", Opt. Comm. 65, 1 (1988).

Johnson, K. M., M. R. Surette, and T. Shamir, "Optical interconnection network using polarization-based ferroelectric liquid crystal gates," Appl. Opt. 27, 1727 (1988).

Kohavi, Z., Switching and Finite Automata Theory, McGraw-Hill, NY 1978.

Kostrzewski, A., Y. Li, G. Eichmann and D. H. Kim, " Fast Hybrid Parallel Carry Look-Ahead Adder," Opt. Lett. 15, (1990).

Kostrzewski, A., Y. Li, D. H. Kim, B. Ha and G. Eichmann, " Optical Position Coded Residue Processor Using Inexpensive LCTV Devices," Appl. Opt. 28, 415 (1989).

Kozaitis, S. P., "Higher-Ordered Rules for Symbolic Substitution," Opt. Comm. 65, 339-342 (1988).

Lee, W. H., "High efficiency multiple beam gratings," Appl. Opt. 18, 2152-2158 (1979).

Lentine, A. L., F. B. McCormick, R. A. Novotny, L. M. F. Chirovsky, L. A. D'Asaro, R. F. Kopf, J. M. Kuo and G. D. Boyd, "A 2 kbit Array of Symmetric Self-Electrooptic Effect Devices," Photonics Technology Letters, 2, 51 (1990).

Li, Y., B. Ha, A. Kostrzewski, D. H. Kim, and G. Eichmann, "Optical Position-Coded Multiple-Variable Logic and Arithmetic Using Liquid-Crystal TVs and Holograms," Optics Comm. 70, 379 (1989).

Li, Y., A. Kostrzewski, D. H. Kim, and G. Eichmann, Opt. Lett. 13, 895 (1988).

Li, Y., D. H. Kim, A. Kostrzewski and G. Eichmann, "Content-addressable-memory-based Single-stage Optical Modified-signed-digit Arithmetic," Opt. Lett. 14, 1254 (1989).

Li, Y., G. Eichmann, R. Dorsinville and R. R. Alfano, "Parallel Ultrafast Digital and Symbolic Optical Computation via Optical Phase Conjugation," Appl. Opt. 27, 2025 (1988).

Li, Y., and G. Eichmann, Appl. Opt. 26, 2328 (1987).

Li, Y., G. Eichmann, R. Dorsinville and R. R. Alfano, "Parallel Ultrafast Digital and Symbolic Optical Computation via Optical Phase Conjugation," Appl. Opt. 27, 2025 (1988).

Li, Y., G. Eichmann, R. Dorsinville and R. R. Alfano, "An AND Operation-based Optical Symbolic Recognizer," Opt. Comm. 63, 375-379 (1987).

Li, Y., G. Eichmann, R. Dorsinville and R. R. Alfano, "Demonstration of a picosecond optical residue computation," Opt. Lett. 13, 178 (1988).

Li, Y., J. Zhu and G. Eichmann, "Optical On-the-fly Conversion of a Modified signed-digit (MSD) into Two's Complement Binary (TCB) Number Representation," Opt. Lett. 13, 294 (1988).

Liu, H. K., J. A. Davis, and R. A. Lilly, "Optical data processing properties of a liquid-crystal TV spatial light modulator," Opt. Lett. 10, 635 (1985).

Livescu, G., D. A. B. Miller, J. E. Henry, A. C. Gossard, and J. H. English, "Spatial light modulator and optical dynamic memory using a 6 X 6 array of self electro-optic-effect devices," Opt. Lett. 13, 297-299 (1988).

Lohmann, A. W., "What Classical Optics can Do for the Digital Optical Computing," Appl. Opt. 25, 1543 (1985).

Lohmann, A. W., and J. Weigelt, Opt. Comm. 54, 81 (1985).

Mait, J. N., "Design of Dammann Gratings for Optical Symbolic Substitution," Proceeding SPIE 963, 646-652 (1988).

Mait, J. N., and K. H. Brenner, "Optical Symbolic Substitution: System Design using Phase-only Holograms," Appl. Opt. 27, 1692-1700 (1988).

Maniloff, E. S., K. M. Johnson, " Dynamic Holographic Interconnects Using Static Holograms, " Opt. Eng., 29 (3), 228-229, (1990).

Mano, M. M., Computer System Architecture, (Prentice-Hall, NJ, 1982) ch.4.

Matthijsse, P., "Multiple imaging with thin phase filters: a signal processing approach," J. Opt. Soc. Am. 68, 733-738 (1978).

McDonald, R. I., "Optoelectronic switch matrix as look-up table for residue arithmetic," Opt. Lett. 12, 787 (1987).

Michell, J. N., IRE Trans. Electron. Comput., pp 512-517, Aug. (1962).

Mirsalehi, M. M., and T. K. Gaylord, " Logical Minimization of Multilevel Coded Functions," Appl. Opt. 25, 3078 (1986).

Mirsalehi, M. M., and T. K. Gaylord, "Truth-table look-up parallel data processing using content-addressable memory," Appl. Opt. 25, 2277 (1986).

Mirsalehi, M. M., T. K. Gaylord, D. C. Fielder, and C. C. Guest, "Number representation effects in truth-table look-up processing," Appl. Opt. 28, 1931 (1989).

Murdocca, M. J., "Digital Optical Computing with One-Rule Cellular Automata," Appl. Opt. 26, 682-688 (1987).

Murdocca, M. J., Ph.D dissertation, (State University of New Jersey at Rutgers, 1988) ch.6.

Ogura, T., S.-I. Yamada, and T. Nikaido, IEEE J. Solid State Circuits SC-20, 1277 (1985).

Ramamoorthy, P. A., S. Antony and T. A. Grogan, "Symbolic-Substitution-based Median Filters," Opt. Eng. 27, 409-412 (1988).

Ramamoorthy, P. A., and S. Antony, Opt. Eng. 26, 821 (1987).

Sicuranza, G. L., IEEE Trans. ASSP, 31, 877 (1983).

Szabo, N. S., and R. Tanada, Residue Arithmetic and Its Application to Computer

Technology. New York: McGraw-Hill, 1967.

Tai, A. M., I. Cindrich, J. R. Fienup, and C. C. Alexsoff, "Optical residue arithmetic computer with programmable computation modules," Appl. Opt. 18, 2182 (1979).

Takagi, N., H. Yasuura, and S. Yajima, IEEE, Trans. Comput. C-34, 789 (1985).

Taub, H., Digital Circuits and Microprocessors, McGraw-Hill, New York (1982).

Thalmann, R., G. Pedrini, B. Acklin and R. Dandliker, "Optical Symbolic Substitution Using Diffraction Gratings," Proceeding SPIE 963, 635-641 (1988).

Tsao, M. T., L. Wang, R. Jin, R. W. Sprague, G. Gigioli, H. M. Kulcke, Y. D. Li, H. M. Gibbs and N. Peyghambarian, "Symbolic Substitution using ZnS Interference Filters," Opt. Eng. 26, 41-44 (1987).

Yatagai, T., Opt. Lett. 11, 260 (1986).

Yu, F. T. S,. and S. Jutamulia, "Implementation of symbolic substitution logic using optical associative memories," Appl. Opt. 26, 2293-2294 (1987).

Yu, F. T. S., C. Zhang and S. Jutamulia, "Applications of one-step holographic associative memory to symbolic substitution," Opt. Eng. 27, 399-402 (1988).

Yu, F. T. S., S. Jutamulia and D. A. Gregory, "Optical parallel logic gates using an inexpensive liquid-crystal TV," Opt. Lett. 12, 1050 (1987).

Yu, F. T. S., S. Jutamulia, and D. A. Gregory, "Real-time liquid crystal TV XOR- and XNOR-gate binary image subtraction technique," Appl. Opt. 26, 2738 (1987).