TABLE I
LIST OF n, p, r, AND PERIOD FOR $m = 5, 7, 13, 17$, AND 19

n	p	r (order of p)	period $(2^n-1)r$
***** M = 5			
2	6	6	18
3	18	15	105
4	1	1	15
***** M = 7			
2	6	126	378
3	87	21	147
4	118	126	1890
5	119	14	434
6	126	2	126
***** M = 13			
2	6	910	2730
3	5040	1638	11466
4	8154	1365	20475
5	7268	8190	253890
6	5086	210	13230
7	4347	8190	1040130
8	5804	1170	298350
9	1123	1638	837018
10	7711	1365	1396395
11	2087	585	1197495
12	1	1	4095
***** M = 17			
2	6	131070	393210
3	5040	21845	152915
4	103431	43690	655350
5	6958	43690	1354390
6	45204	43690	2752470
7	85423	131070	16645890
8	25027	1285	327675
9	41703	43690	22325590
10	113021	43690	44694870
11	104297	21845	44716715
12	117473	4369	17891055
13	20133	13107	107359437
14	52447	2570	42104310
15	64611	131070	4294770690
16	131070	2	131070
***** M = 19			
2	6	524286	1572858
3	5040	37449	262143
4	354035	87381	1310715
5	64318	262143	8126433
6	135841	262143	16515009
7	142484	27594	3504438
8	315302	87381	22282155
9	124189	87381	44651691
10	406212	262143	268172289
11	202951	262143	536606721
12	282298	174762	715650390
13	494755	524286	4294426626
14	509505	262143	4294688769
15	194210	524286	17179279362
16	266965	4599	301395465
17	449113	262143	34359345153
18	1	1	262143

An interesting topic for future research is the exponentiation in extension field $GF(p^m)$ where $p \neq 2$. Using a normal basis $\{\alpha, \alpha^p, \alpha^{p^2}, \cdots, \alpha^{p^{m-1}}\}$ may generate a long sequence of pseudorandom numbers.

ACKNOWLEDGMENT

The authors would like to thank Prof. J. K. Omura for his stimulation leading to this topic and his encouragement during the course of this work.

REFERENCES

[1] J. L. Massey and J. K. Omura, "Computational method and apparatus for fine field arithmetic," U.S. Patent Application, in 1981.

[2] C. C. Wang *et al.*, "VLSI architecture for computing multiplications and inverses in $GF(2^m)$," *IEEE Trans. Comput.*, vol. C-34, no. 8, Aug. 1985.

[3] R. C. Dixon, *Spread Spectrum Systems.* New York: Wiley, 1976.

[4] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes.* New York: North-Holland, 1977.

[5] M. Perlman, "Periodic binary sequence generators: Very large scale integrated (VLSI) circuits considerations," JPL Publ. 85-7, Dec. 1984.

[6] W. W. Peterson and E. J. Weldon, Jr., *Error-Correcting Codes.* Cambridge, MA: MIT Press, 1972.

[7] S. Berkovits, J. Kowalchuk, and B. Schanning, "Implementing public key scheme," *IEEE Commun. Mag.*, vol. 17, pp. 2-3, May 1979.

On the Complexity of Mod-2 Sum PLA's

TSUTOMU SASAO AND PHILIPP BESSLICH

Abstract—**We consider the realization of logic functions by using PLA's with an Exclusive-OR (EXOR) array, where a function is represented by mod-2 (EXOR) sum-of-products (ESOP's), and both true and complemented variables are used. First, we propose a new PLA structure using an EXOR array. Second, we derive upper bounds on the number of products of this type of PLA, which are useful for estimating the size of a PLA as well as for assessing the minimality of the solutions obtained by heuristic ESOP minimization algorithms. Computer simulation using randomly generated functions shows that PLA's with EXOR array require, on the average, fewer products than conventional PLA's. For symmetric functions, we conjecture that the PLA's with an EXOR array require at most as many products as the conventional PLA's. The proposed PLA's can be made easily testable by adding a small amount of hardware.**

Index Terms—**Complexity, easily testable circuits, Exclusive-OR sum-of-products, logic minimization, programmable logic array, symmetric functions.**

I. INTRODUCTION

The complexity of various types of programmable logic arrays (PLA's) has been studied in detail [1], [2]. Logic design for standard PLA's is based on multiple-output (quasi-)minimization of the conventional sum-of-products expression (SOP) of switching functions. However, there is a conjecture that PLA's consisting of inverters, an AND array, and an Exclusive-OR (EXOR) array may offer certain advantages. "It has long been conjectured that the realization of switching functions as a mod-2 SOP is more economical than the conventional SOP's." This sentence appears at least three times in the literature [3]-[5]. Since there has been neither an exact minimization method nor a reliable quasi-minimization procedure, the conjecture has never been confirmed except for the case of $n = 4$ variables [3]. In this paper, we consider the complexity of PLA's using EXOR (or mod-2) SOP's.

Motivations for this research are twofold: First, there is the challenge to prove or disprove the conjecture mentioned above. Since no exact minimization algorithm for EXOR SOP's (ESOP's) exists, we developed several quasi-minimization algorithms for ESOP's. In order to assess the performance of these algorithms, there is a need for upper bounds on the number of products and for functions for benchmark testing. The second motivation is to investigate other possibilities for the realization of switching functions in logic arrays. There have also been proposals to employ modulo sum addition in

Manuscript received March 13, 1987; revised March 21, 1989. Part of the results were obtained during P. Besslich's visit to Osaka University in September 1986, jointly sponsored by JSPS (Japan Society for the Promotion of Science) and DAAD (German Academic Exchange Service). This work was also supported by The Mazda Foundation's Research Grant (T. Sasao).

T. Sasao is with the Department of Computer Science and Electronics, Kyushu Institute of Technology, Iizuka 820, Japan.

P. Besslich is with the Department of Electrical Engineering, University of Bremen, D-2800 Bremen, Federal Republic of Germany.

IEEE Log Number 8930171.

0018-9340/90/0200-0262$01.00

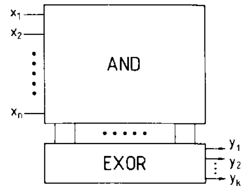

Fig. 1. AND–EXOR PLA without input decoders.

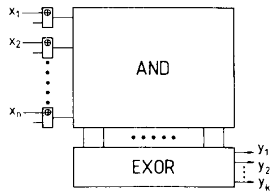

Fig. 2. AND–EXOR PLA with input EXOR's.

multiple-valued PLA's [6], [7]. Because of the page limitation, all the proofs of the theorems are omitted. A full length paper is available from the authors. A preliminary version of this paper has been presented as [8].

II. PLA Structure for MOD-2 Sum

A. AND-EXOR PLA Without Input Decorders

An arbitrary switching function can be represented by a mod-2 SOP of uncomplemented variables [9], [10]. The PLA structure shown in Fig. 1 (an AND–EXOR PLA without input decoders) realizes logic functions based on the positive polarity Reed–Muller canonical expansion (RME). In this PLA, since only uncomplemented variables are employed, the number of rows in the AND array is half of those of a standard PLA (an AND–OR PLA with one-bit decoders) shown in Fig. 4. However, as shown in Lemma 2.2, the number of columns (or products) of PLA's having this structure is, on the average, at least as twice as large as that of standard PLA's. This is undesirable for VLSI implementation.

Lemma 2.1: The necessary and sufficient number of products to realize the function $f(x) = \bar{x}_1\bar{x}_2 \cdots \bar{x}_n$ by a PLA shown in Fig. 1 is 2^n.

Note that a standard PLA requires only one column to realize this function.

Lemma 2.2: The average number of products to realize n-variable functions by PLA's with structure shown in Fig. 1 is 2^{n-1}.

In the case of standard PLA's, the average number of products to realize n-variable functions is less than 2^{n-2} [1].

B. AND-EXOR PLA with Input EXOR's

In the PLA structure shown in Fig. 2, each input variable can either be uncomplemented or complemented. It realizes the fixed polarity Reed–Muller canonical representation [11], [12]. However, as Lemma 2.3 shows, this PLA requires more columns than a standard PLA for simple functions. Therefore, this PLA structure is also unsuitable for VLSI implementation.

Lemma 2.3: The necessary and sufficient number of products to realize the function $f(x) = x_1x_2 \cdots x_n \vee \bar{x}_1\bar{x}_2 \cdots \bar{x}_n$ by a PLA shown in Fig. 2, is $2 \cdot (2^r - 1)$, where $n = 2r$.

C. AND-EXOR PLA with One-Bit Decoders

The PLA structure shown in Fig. 3 uses both uncomplemented and complemented variables. It realizes an ESOP without any restrictions. As will be shown in Section V, this PLA structure requires, on the average, fewer columns than standard PLA's.

Example 2.1: Let $f(x) = x_1x_2x_3x_4 \vee \bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4$. The positive polarity RME for this function requires 15 products. From Lemma 2.3, fixed polarity RME needs six products. If the condition of fixed polarities in the RME products is dropped (i.e., mixed or free polarity RME [5] is used), we need only four products. Finally, if the restrictions imposed by the canonical RME's are disregarded, we need only two products as follows:

$$f(x) = x_1x_2x_3x_4 \oplus \bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4. \text{ (End of Example)}$$

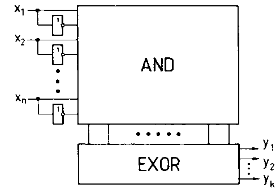

Fig. 3. AND–EXOR PLA with one-bit decoders.

III. Upper Bounds on the Number of Products in ESOP's

In this section, we show upper bounds on the number of products in ESOP's for arbitrary functions, symmetric functions, and adders. These bounds correspond to the upper bounds on the number of columns for PLA's having the structure of Fig. 3.

A. Bound for Arbitrary Functions

Let $t_e(f)$ be the necessary and sufficient number of products to represent the function f by an ESOP. Theorem 3.1 shows that PLA's with an EXOR array require fewer products than conventional PLA's to realize arbitrary functions.

Theorem 3.1: Let f_n be an arbitrary function of n variables. Then, $t_e(f_n) \leq 3/4 \cdot 2^{n-1}$, where $n \geq 3$.

Note that in order to represent a parity function of n variables by an SOP, 2^{n-1} products are necessary.

B. Bounds for Symmetric Functions

In this part, we consider upper bounds on the number of products in ESOP's of symmetric functions.

Theorem 3.2: Suppose $n = 2r + 1$. Let $X = (X_1, X_2, \cdots, X_{r+1})$ be a partition of $x = (x_1, x_2, \cdots, x_n)$, where $X_i = (x_{2i-1}, x_{2i})(i = 1, 2, \cdots, r)$, and $X_{r+1} = (x_{2r+1})$. If $f_n(x)$ is partially symmetric with respect to X_i $(i = 1, 2, \cdots, r)$, then $t_e(f_n) \leq 3^r$.

Theorem 3.3: Suppose $n = 2r$. Let $X = (X_1, X_2, \cdots, X_r)$ be a partition of $x = (x_1, x_2, \cdots, x_n)$, where $X_i = (x_{2i-1}, x_{2i})(i = 1, 2, \cdots, r)$. If $f_n(x)$ is partially symmetric with respect to X_i $(i = 1, 2, \cdots, r)$, then $t_e(f_n) \leq 2 \cdot 3^{r-1}$.

These bounds are equal to the ones given in [3]. They apply, however, to a larger class of functions. While Even, Kohavi, and Paz [3] prove the bounds only for completely symmetric functions, Theorem 3.3 applies to all functions in which at least two variables may be interchanged. The number of totally symmetric functions of n variables is 2^{n+1}, while the number of the partially symmetric functions of n variables is 2^{3^r}, where $n = 2r$.

In order to find a tighter upper bound on the number of products in ESOP's, we define another class of functions.

Definition 3.1: Let $x = (x_1, x_2, \cdots, x_n)$.

Then the functions $E_k^n(x)$ are defined as follows:

$$E_0^n = 1$$

$$E_1^n = \sum_{i=1}^{n} \oplus x_i$$

$$E_2^n = \sum_{1 \leq i < j \leq n} \oplus x_i x_j$$

$$\cdots\cdots\cdots\cdots$$

$$E_k^n = \sum_{1 \leq i < j < \cdots < m \leq n} \oplus x_i x_j, \cdots, x_m$$

$$E_n^n = x_1 \cdot x_2 \cdot x_3 \cdot \cdots \cdot x_n.$$

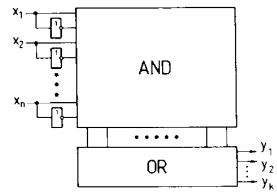

Fig. 4. AND–EXOR PLA with one-bit decoders (standard PLA).

Note that $E_k^n(\boldsymbol{a})$ is 1 iff at least k of the variables are 1 and the minterm $\boldsymbol{a}$ is covered an odd number of times.

Let $T(n, k) = t_e(E_k^n)$. By Definition 3.1, we have the following.

Theorem 3.4: $T(n, k) \leq \binom{n}{k}$.

We will show recurrence relations, which derive an upper bound on the number of products in the minimum ESOP of E_k^n functions.

Theorem 3.5: $T(k, 0) = 1, T(k, 1) = k, T(k, k-1) = k, T(k, k) = 1$, and $T(n, k) \leq T(n-1, k) + T(n-1, k-1)$.

When the upper bounds for E_k^{n-1} and E_{k-1}^{n-1} are known, Theorem 3.5 provides an upper bound for the E_k^n function. It often gives a tighter bound than the bound given by Theorems 3.2, 3.3, or 3.4. This fact makes the E_k^n functions a suitable class of functions for benchmark testing of heuristic ESOP minimizers.

Example 3.1: Let $\boldsymbol{x} = (x_1, x_2, x_3, x_4)$. By Definition 3.1, we have $E_2^4(\boldsymbol{x}) = x_1x_2 \oplus x_1x_3 \oplus x_1x_4 \oplus x_2x_3 \oplus x_2x_4 \oplus x_3x_4$.

This function is equivalent to the symmetric function $S^4_{\{2,3\}}$. Although [3] claimed that this function required six products, it can be represented by five products: $E_2^4(\boldsymbol{x}) = x_1x_2 \oplus x_1x_4 \oplus x_2\bar{x}_3x_4 \oplus \bar{x}_2x_3\bar{x}_4 \oplus \bar{x}_1x_3$.

Therefore, $T(4, 2) \leq 5$. Because $T(4, 1) = 4$, we have an upper bound for E_2^5 function as follows: $T(5, 2) \leq T(4, 2) + T(4, 1) = 9$. (End of example)

C. Bound for Adders

The number of products to realize an n-bit adder by a standard PLA (Fig. 4) is $6 \cdot 2^n - 4n - 5$ [2]. Theorem 3.6 shows that PLA's with EXOR array require about one-third as many columns as standard PLA's.

Theorem 3.6: The number of products which is sufficient to realize an n-bit adder by a PLA with EXOR array (Fig. 3) is $2^{n+1} - 1$.

IV. Minimization of MOD-2 Sums of Products

A Pascal program which obtains near minimal ESOP's of up to 12 variables on a personal computer has been developed. Because we are only interested in the complexities of PLA's, we did not try to reduce computation time.

The minimization program consists of three independent minimization algorithms and it chooses the best solution of the three. The first algorithm detects single-variable EXOR patterns by an adaptive filter operation [13]. An adaptive threshold is introduced to decide whether or not an extraction of those patterns is rewarding. After extracting a single-variable EXOR pattern, the residual minterms are iteratively covered by a heuristic EXOR covering algorithm. Since these decisions are taken on present state information only, the minimization is suboptimal. DON'T CARE minterms, if any, will be (sub-)optimally allocated. This algorithm is particularly suitable for functions having distinct "parity function character," as well as for the type of functions which exhibit "smaller" zero-products within "larger" one-products. The algorithm is essentially based on radix-2 type in-place processing of data. For a detailed description, refer to [13] and [14].

The second algorithm is based on an iterative improvement of covering. Several rules are used to replace a pair of products with another one. By using these rules, the given cover is modified iteratively without changing the function represented by the cover. This algorithm uses both ideas of the program developed by Even, Kohavi, and Paz [3], as well as those of MINI [15]. Similar algorithms have been developed independently [16]–[18].

The third algorithm is same as the second one except that the complement of the function is realized instead of the given function. The constant 1 is added to obtain the proper output.

TABLE I
Average Number of Products for All the Four-Variable Functions

MINTERMS	1	2	3	4	5	6	7	8
AND-EXOR	1.000	1.733	2.371	2.738	3.132	3.350	3.702	3.696
AND-OR	1.000	1.733	2.371	2.905	3.370	3.730	4.053	4.273
MINTERMS	9	10	11	12	13	14	15	16
AND-EXOR	3.912	3.864	4.088	3.732	3.371	2.733	2.000	1.000
AND-OR	4.457	4.537	4.546	4.426	4.200	3.733	4.000	1.000

TABLE II
Minimization Results of Randomly Generated Functions

n = 6

d=u/64	1/8	2/8	3/8	4/8	5/8	6/8	7/8
u	8	16	24	32	40	48	56
E (n, u)	5.7	8.8	11.1	11.7	11.6	9.8	7.4
S (n, u)	6.0	9.4	12.9	13.2	13.1	12.2	9.5

n = 7

d=u/128	1/8	2/8	3/8	4/8	5/8	6/8	7/8
u	16	32	48	64	80	96	112
E (n, u)	11.0	16.9	20.0	22.1	20.4	18.1	12.0
S (n, u)	12.1	18.6	21.5	24.2	23.1	21.9	15.8

n = 8

d=u/256	1/8	2/8	3/8	4/8	5/8	6/8	7/8
u	32	64	96	128	160	192	224
E (n, u)	21.4	32.6	38.7	41.6	38.4	32.2	22.4
S (n, u)	21.5	35.1	40.6	42.8	41.9	37.9	27.7

n:number of the input variables.
u:number of the minterms.
E (n, u):average number of products in ESOP's.
S (n, u):average number of prime implicants in the minimum SOP's
(the average of ten randomly generated functions)

V. Experimental Results

A. Minimization of All the Four-Variable Functions

In order to obtain the minimum ESOP's and SOP's for all functions of four variables, we minimized each representative function of the NP-equivalence classes [19], [20]. To find absolute minimum ESOP's, we developed a special program [21], which is similar to [22]. To find absolute minimum SOP's, we used the Quine–McCluskey algorithm. Table I shows the average number of products in ESOP's and SOP's for logic functions with u minterms ($u = 1, 2, \cdots, 16$). Table I shows that ESOP's, on the average, require fewer products than SOP's. Note that both ESOP's and SOP's are absolute minimum.

B. Minimization of Randomly Generated Functions

We also obtained statistical data for the functions of $n = 6$, 7, and 8 variables. We generated ten random functions by a pseudorandom number generator for each density, where density denotes the fraction of minterms which are mapped into 1, i.e., $d = u/2^n$. Table II com-

TABLE III
AVERAGE NUMBER OF PRODUCTS TO REALIZE SYMMETRIC FUNCTIONS

n	ESOP	SOP
2	1.375	1.375
3	2.250	2.625
4	4.031	5.156
5	6.969	10.125
6	12.093	20.055
7	21.418	39.563

TABLE IV
NUMBER OF PRODUCTS FOR E^n_k FUNCTIONS OBTAINED BY HEURISTIC MINIMIZER

	k								
n	0	1	2	3	4	5	6	7	8
1	1	1							
2	1	2	1						
3	1	3	3	1					
4	1	4	5	4	1				
5	1	5	9	8	5	1			
6	1	6	14	16	13	6	1		
7	1	7	18	29	26	17	7	1	
8	1	8	25	49	42	45	25	8	1

pares the average number of products in ESOP's to those of SOP's. In this case, we used the heuristic minimization program mentioned in Section IV to obtain ESOP's, and the Quine–McCluskey algorithm to obtain SOP's. Again, this table shows that ESOP's require, on the average, fewer products than SOP's.

C. Minimization of Symmetric Functions

We compared the number of products in ESOP's and SOP's for symmetric functions of $n = 2$ to 7 variables. Table III shows that ESOP's require many fewer products than SOP's. Note that ESOP's are near minimal, but SOP's are absolute minimum. The surprising fact is that, for every symmetric function of up to seven variables, the number of products in an SOP is equal to or greater than that of the near minimal ESOP. We conjecture this property holds for $n \geq 8$, and have the following.

Conjecture 5.1: Let $t_e(f)$ and $t_o(f)$ be the number of products in the minimum ESOP and SOP for f, respectively. If f is symmetric, then $t_e(f) \leq t_o(f)$.

D. Minimization of E^k_n Functions

Table IV shows the number of products for E^n_k functions, which are obtained by the heuristic minimization program. Unfortunately, we recognize that some of the solutions obtained by the program are nonminimum.

Example 5.1: We know that some of the solutions obtained by the heuristic minimization program are not minimum. From Theorem 3.2, we can see that $T(7, 3) \leq 3^3 = 27$. However, the corresponding entry in Table IV is 29, which shows the heuristic minimizer failed to produce the optimal solution for E^7_3. Also, from Table IV, we can see that $T(7, 2) \leq 18$ and $T(7, 3) \leq 29$. Therefore, by Theorem 3.5, we have $T(8, 3) \leq T(7, 2) + T(7, 3) \leq 47$. On the other hand, the corresponding entry in Table IV is 49, which shows that the heuristic minimizer failed to obtain the optimal solution for E^8_3. (End of the Example).

VI. CONCLUSION

In this paper, we considered the number of products in PLA's having inverters, an AND array, and an EXOR array. The results of theoretical and experimental studies are as follows.

1) We conjecture that, on the average, PLA's with EXOR arrays require fewer products than standard PLA's. This conjecture is based on a) exhaustive minimization of all the four-variable functions, b) minimization of randomly generated functions of $n = 6, 7$, and 8 variables, and c) Theorem 3.1.

2) We conjecture that, for symmetric functions, PLA's with EXOR arrays require many fewer products than standard PLA's. This conjecture is based on d) exhaustive minimization of all the symmetric functions of $n = 2$ to 7 variables, and on e) Theorems 3.2 and 3.3.

3) Upper bounds on the number of products of PLA's with EXOR arrays are shown in Table V.

4) We derived a special class of symmetric functions, E^n_k functions, which is useful for assessing the minimality of solutions obtained by heuristic ESOP minimization algorithm.

The experimental results obtained by the heuristic ESOP minimizer confirm the conjecture made in [2]–[4]. Benchmark tests of the heuristic minimizers show that further improvements of the heuristic algorithms may be possible.

The disadvantage of the conventional SOP's (POS's) of switching functions is the nonlinear nature of the OR (AND) connective, which may make the testing of circuits rather involved. The test of circuits using the EXOR addition (i.e., the complete system consisting of AND/EXOR/UNITY) is considered to be less expensive [23]. Since the cost of testing can be a decisive factor in VLSI production, ESOP's may be more economical than the usual SOP's. Recently, it has been suggested to provide additionally one row, one column, and a cascade of EXOR gates to facilitate the testing of PLA's [24], [25]. These schemes may be modified so as to use the same EXOR gates for combining of products as well as for testing [26].

TABLE V
NUMBER OF PRODUCTS TO REALIZE VARIOUS FUNCTIONS BY THREE TYPES OF PLA'S

	PLA with EXOR array	Standard PLA	PLA with two bit decoders
Arbitrary Functions	$(3/4) \cdot 2^{n-1}$	2^{n-1}	$(1/2) \cdot 2^{n-1}$
Symmetric Functions	$2 \cdot 3^{r-1}$	2^{n-1}	3^{r-1}
Parity Functions	n	2^{n-1}	2^{r-1}
n-bit Adders	$2^{n+1}-1$	$6 \cdot 2^n - 4n - 5$	$n^2 + 1$

REFERENCES

[1] T. Sasao, "Multiple-valued decomposition of generalized Boolean functions and the complexity of programmable logic arrays," *IEEE Trans. Comput.*, vol. C-30, pp. 635–643, Sept. 1981.

[2] —, "Input variable assignment and output phase optimization of PLA's," *IEEE Trans. Comput.*, vol. C-33, pp. 879–894, Oct. 1984.

[3] S. Even, I. Kohavi, and A. Paz, "On minimal modulo-2 sums of products for switching functions," *IEEE Trans. Electron. Comput.*, vol. EC-16, pp. 671–674, Oct. 1967.

[4] A. Mukhopadhyay and G. Schmitz, "Minimization of Exclusive OR and logical equivalence of switching circuits," *IEEE Trans. Comput.*, vol. C-19, pp. 132–140, 1970.

[5] J. P. Robinson and C.-L. Yeh, "A method for modulo-2 minimization," *IEEE Trans. Comput.*, vol. C-31, pp. 800–801, 1982.

[6] P. Tirumalai and J. T. Butler, "On the realization of multiple-valued logic functions using CCD PLA's," in *Proc. 14 Int. Symp. Multiple-Valued Logic,* May 1984, pp. 33–42.

[7] G. W. Dueck and D. M. Miller, "A 4-valued PLA using the MODSUM," in *Proc. 15 Int. Symp. Multiple-Valued Logic,* May 1986, pp. 232–240.

[8] T. Sasao and P. W. Besslich, "On the complexity of PLA's with EXOR arrays," IECEJ Tech. Rep., FTS 86-17, Nov. 1986.

[9] I. S. Reed, "A class of multiple-error-correcting codes and the decoding scheme," *IRE Trans. Inform. Theory,* vol. PGIT-4, pp. 38–49, 1954.

[10] D. E. Muller, "Application of Boolean algebra to switching circuit design and to error detection," *IRE Trans. Electron. Comput.*, vol. EC-3, pp. 6-12, 1954.
[11] G. Bioul, M. Davio, and J. P. Deschamps, "Minimization of ring-sum expansions of Boolean functions," Philips Res. Rep., vol. 28, pp. 17-36, 1973.
[12] D. H. Green and I. S. Taylor, "Multiple-valued switching circuit design by means of generalized Reed-Muller expansions," *Digital Processes,* vol. 2, pp. 63-81, 1976.
[13] P. W. Besslich and H. Bassmann, "Synthesis of Exclusive-OR logic functions," Berichte Elekrotechnik, ISSN 0724-1933, No. 3.87, Universitat Bremen, FB-1, P.O. Box 330 440, D-2800 Bremen 33, West Germany.
[14] P. W. Besslich, "Spectral processing of switching functions using signal-flow transformations," *Spectral Techniques and Fault Detection*, in M. Karpovsky, Ed. Orlando, FL: Academic, 1985, pp. 91-141.
[15] S. J. Hong, R. G. Cain, and D. L. Ostapko, "MINI: A heuristic approach for logic minimization," *IBM J. Res. Develop.*, vol. 18, pp. 443-458, Sept. 1974.
[16] H. Fleisher, M. Tavel, and J. Yeager, "A computer algorithm for minimizing Reed-Muller canonical forms," *IEEE Trans. Comput.*, vol. C-36, no. 2, pp. 247-250, Feb. 1987.
[17] M. Helliwell and M. Perkowski, "A fast algorithm to minimize multi-output mixed-polarity generalized Reed-Muller forms," in *Proc. 25th Design Automat. Conf.*, June 1988, pp. 427-432.
[18] T. Sasao and M. Higashida, "On a design algorithm for AND-EXOR PLA's with input decoders," *The 20th FTC Workshop,* Jan. 1989 (in Japanese).
[19] M. A. Harrison, *Introduction to Switching and Automata Theory*. New York: McGraw-Hill, 1965.
[20] S. Muroga, *Logic Design and Switching Theory*. New York: Wiley, 1979.
[21] N. Koda and T. Sasao, "Table for minimum exclusive-OR sum-of-products for 4-variable functions," in preparation.
[22] L. Hellerman, "A catalog of three-variable OR-invert and AND-invert logical circuits," *IEEE Trans. Electron. Comput.*, vol. EC-12, pp. 198-223, June 1963.
[23] S. M. Reddy, "Easily testable realization for logic functions," *IEEE Trans. Comput.*, vol. C-21, pp. 1083-1088, 1972.
[24] H. Fujiwara and K. Kinoshita, "A design procedure of programmable logic arrays with universal tests," *IEEE Trans. Comput.*, vol. C-30, pp. 823-828, Nov. 1981.
[25] K. A. Hua, J.-Y. Jou, and J. A. Abraham, "Built-in tests for VLSI finite-state machines," in *Proc. IEEE 14th Fault-Tolerant Comput. Syst. Conf.*, 1984, pp. 292-297.
[26] T. Sasao and H. Fujiwara, "A design of AND-EXOR PLA's with universal tests," IECEJ Tech. Rep., FTS 86-26, Feb. 1987 (in Japanese).

On the Design of a Unidirectional Systolic Array for Key Enumeration

FERNG-CHING LIN AND KUNG CHEN

Abstract—**Key enumeration is to compute the rank of each key in a sequence of keys. This paper introduces a new systolic linear array to enumerate *n* keys in $3n - 1$ time steps. This array has unidirectional data flow and achieves maximum data pipelining rate. Modifications of the array for solving the closest neighbor problems in computational geometry are also presented.**

Index Terms—**Data dependence graph, key enumeration, nearest neighbor problems, systolic array.**

Manuscript received March 26, 1987; revised August 4, 1988. This work was supported in part by the National Science Council of the Republic of China under Contract NSC76-0408-E002-5.

The authors are with the Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, Republic of China.

IEEE Log Number 8930172.

I. Introduction

The dramatic development of very large scale integration (VLSI) technology has made it possible to implement algorithms directly in hardware and hence promoted great interest in designing algorithmically specialized processing components. Following Kung's *systolic concept* [1]-[3], many computing arrays have been proposed to handle various compute-bound problems. These array processors generally consist of a regular array of simple and identical processing elements (PE's) in which data are communicated locally and operated on rhythmically. The simplicity, regularity, and locality of the systolic arrays render them suitable for VLSI implementation. High performance is achieved by the concurrent use of a large amount of PE's in the arrays.

The purpose of this paper is to introduce a new systolic array for the problem of *key enumeration.* The well-known enumeration sort (sorting by counting) [4] is composed of a ranking process and a rearranging process. The ranking process inputs a sequence of keys $k_1, k_2, \cdots, k_n$ and outputs a sequence of ranks $r_1, r_2, \cdots, r_n$ to represent that k_i is the $(r_i + 1)$th smallest key in the input sequence. Then in the rearranging process the records are rearranged according to the ranks of their keys. In this paper, we are only concerned with the ranking process, i.e., key enumeration.

Yasuura *et al.* [5] first proposed a linear array equipped with two global I/O buses to compute the ranks. The propagation delay along the long wires limits the clock speed at which the system can run reliably. Su [6] later proposed a linear array without global communication buses. It, however, requires duplication of input data sequence. Lin and Wu [7] then presented a systolic linear array to which only one set of input keys is serially fed. The array uses an extra PE at one end to reverse the flow direction of the data stream. A bidirectional data move typically cannot achieve maximum data pipelining rate. In that design, data in the stream have to be separated by two time units. Recently, Chen and Nussbaum [8] proposed an array of triangularly connected PE's with two sets of data moving orthogonally. Such a two-dimensional structure is area-demanding and furthermore the problem-size-dependent number of I/O ports makes it even more impractical, due to the packaging limitation.

The new systolic array presented in this paper circumvents all the shortcomings of those previous designs. The way of synthesizing this array, which will be explained in detail in Sections II and III, can be outlined as follows. First, we write down a usual sequential algorithm to compute the ranks and model it as computation activities on an abstract index set [9], [10]. From the indexed computations we identify the data dependencies and represent them as a data dependence graph. In order to map it into a regularly operated one-dimensional array in more alternative ways, the data dependence graph is modified according to a broadcast normalization concept introduced in [7]. Then a proper space-time transformation is chosen to map the graph into the desired systolic array.

All-nearest-neighbors and *closest-pair* problems in computational geometry are, by nature, closely related to the enumeration problem. In Section IV, we shall show that the proposed array for key enumeration can easily be modified to solve these problems efficiently. Some practical considerations for the design will be discussed in the final section.

II. Data Dependence Graph

Here is straightforward sequential algorithm to enumerate keys:

For $i := 1$ to n do
  For $j := 1$ to n do
    if $k_j > k_i$ then $r_j := r_j + 1$;