On the Optimal Design of Multiple-valued PLA's

Tsutomu SASAO

Department of Electronic Engineering
Osaka University
Osaka 565, Japan

Abstract: In this paper, first, two types of multiple-valued PLA's are presented: 1) PLA's with Min and Max arrays, and 2) PLA's with AND and OR arrays followed by encoders. The optimal set of literals for p-valued PLA's is presented. Logical complexities and capabilities of two types of PLA's are compared. Then, a logic design of the second type of PLA's is considered in detail. In that PLA's, each p-valued input is converted into a set of two-valued signals. Optimal input and output encoding problems are presented. Adders are designed for 5 different PLA realizations. The PLA's proposed in this paper requires much smaller arrays than previously published ones.

I. Introduction

One of the most pressing problems in present-day two-valued system is interconnection complexity, both in-chip and between chips[HUR 84]. It is evident that multi-valued logic (MVL) is useful for reducing interconnections. Thus, various MVL systems have been proposed for many years.

When we design multiple-valued VLSI(MV-VLSI), we encounter the same problems as in two-valued systems. The first problem is the enormous design complexity of VLSI's. As the number of the elements in a chip increases, design time increases exponentially. Because logic design of multiple-valued systems is usually much more complicated than two-valued systems, this problem is more important in MV-VLSI's. In order to reduce the design time and errors, automatic design is indispensable in MV-VLSI's. However, even in two-valued system, automatic design of random logic circuit is very difficult. The only two-valued circuits which are successfully designed by a complete automatic system are programmable logic arrays(PLA)[SAS 86a].

The second problem is the testability of the VLSI's. In the modern VLSI's, testing cost often dominates the total production cost[DAN 85]. In order to overcome the design complexity and testability problems, circuits having regular structure such as PLA's, ROM's and RAM's are extensively used in the many of the VLSI's. For example, recent VLSI microprocessors such as BELLMAC-32A[LAW 82] and Motorola MC68020[DAN 85] use PLA's extensively in the control part of the processors. PLA's can be used to implement complex MVL circuits. PLA's are the most promising approach to the design of complex MVL circuits.

In this paper, the author proposes a multiple-valued PLA (MVPLA) which is easily implemented by (static or dynamic) MOS/CMOS circuits. The MVPLA consists of literal generators (which convert multiple-valued signals into two-valued signals), an AND array, an OR array, and output encoders (which convert two-valued signals into multi-valued signals). Because the AND and the OR arrays are same as those of two-valued PLA's with decoders[SAS 81], we can use various existing PLA design tools such as MINI[HON 74], MINI-II[SAS 84] and ESPRESSO-MV[RUD 85]. Logical capability and logical complexity analysis show that the proposed MVPLA requires much smaller arrays than previously published MVPLA's[KUO 85],[IMM 85].

II. PLA with Min and Max Arrays.

2.1 Logical Implementation

Fig.2.1 shows an n-input m-output p-valued PLA with a Min and a Max arrays. We will call this PLA a Type 1 PLA. This structure represents a MVL expression:

$$f=0\cdot g_0 \vee 1\cdot g_1 \vee \ldots \vee (p-1)\cdot g_{p-1}, \quad \text{--------} \quad (2.1)$$

where $\vee$ denotes the Max operator and $\cdot$ denotes the Min operator. Similar structures can be found in [HUR 84] or [KUO 85].

In realizing (2.1), g_0 is usually omitted. (p-1), which is the largest value, can also be omitted from (2.1). Therefore, (2.1) can be rewritten as

$$f= 1\cdot g_1 \vee \ldots \vee (p-2)\cdot g_{p-2} \vee g_{p-1}. \quad \text{--------} \quad (2.2)$$

Thus, (p-2) different constants are used in the Min-array in Fig.2.1. Each sub-function g_i (i=1,2,...,p-1) of (2.2) can be represented as the sum-of-products expression:

$$g_i = \vee \quad X_1^{S_1}\cdot X_2^{S_2}\cdot \ldots\ldots \cdot X_n^{S_n}, \quad \text{--------} \quad (2.3)$$

where $S_j \subseteq P$, and $P=\{0,1,2,\ldots,p-1\}$. A literal X^S takes a value 0 if $X \notin S$ and a value (p-1) if $X \in S$. There are 2^p literals. In Fig.2.1, each p-valued signal X_i is converted into a set of literals. There are many ways to choose the set of literals. We choose the minimum universal set of literals, which can represent any literal by a logical product of some of literals in the set, and contains the minimum number of elements.

As shown in the Appendix, the minimum universal set of literals contains p elements. Thus, the number of rows in the Min array is $H_1=np+(p-2)$.

Theorem 2.1: Let W be the number of columns necessary to realize an arbitrary p-valued function in a Type 1 PLA, then $W \leq (p-1)p^{n-1}$.

(Proof) It is clear that each column of Type 1 PLA realizes a product

$(j)\cdot X_1^{S_1}\cdot X_2^{S_2}\cdot \ldots X_n^{S_n}$. Therefore, the number of columns is equal to the number of products in (2.2). Each sub-function g_i can be realized with at most p^{n-1} products because g_i can be written as

$g_i = \vee\, G(X_1)X_2^{a_2}\cdot X_3^{a_3}\cdot \ldots\ldots \cdot X_n^{a_n}$, where $a_k \in P$ and k=2,...,n. Therefore, the total number of products to represent (2.2) is at most $(p-1)p^{n-1}$. (Q.E.D)

Logic design of a Type 1 PLA can be done as follows:

Algorithm 2.1:

1) Obtain a minimum sum-of-products expression for g_{p-1}.

2) For each g_k,(k=p-2,...,1), obtain a minimum sum-of-products expression for g_k. In this case, g_r $(k+1 \leq r \leq p-1)$ can be used as don't care sets.

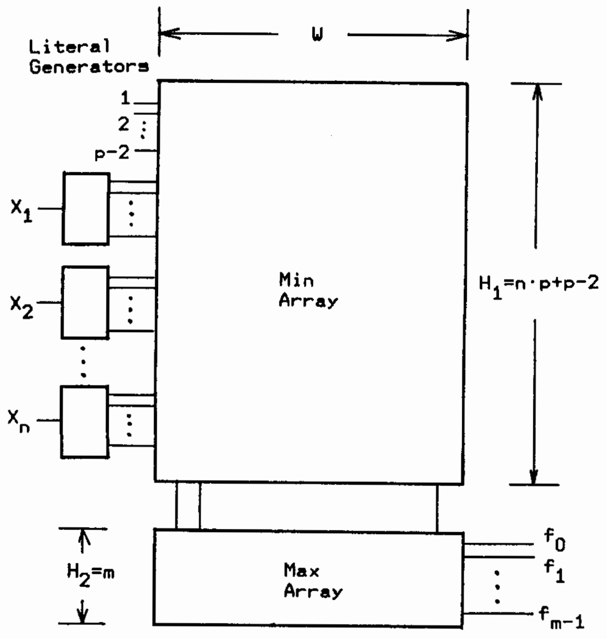


Fig.2.1 p-valued PLA with Min and Max Array (Type 1 PLA)

Table 2.1 Truth Table for 4-valued Adder

Input		Output	
X_1	X_2	Sum	Carry
0	0	0	0
0	1	1	0
0	2	2	0
0	3	3	0
1	0	1	0
1	1	2	0
1	2	3	0
1	3	0	1
2	0	2	0
2	1	3	0
2	2	0	1
2	3	1	1
3	0	3	0
3	1	0	1
3	2	1	1
3	3	2	1

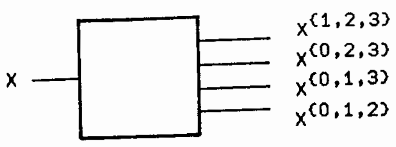


Fig. 2.2 Literal Generator

X_1 \ X_2	0	1	2	3
0		1	2	3
1	1	2	3	
2	2	3		1
3	3		1	2

(a) Sum

X_1 \ X_2	0	1	2	3
0				
1				1
2			1	1
3		1	1	1

(b) Carry

X_1 \ X_2	0	1	2	3
0				3
1			3	
2		3		
3	3			

(c) g_3

X_1 \ X_2	0	1	2	3
0			2	x
1		2	x	
2	2	x		
3	x			2

(d) $2 \cdot g_2$

X_1 \ X_2	0	1	2	3
0		1	x	x
1	1	x	x	
2	x	x		1
3	x		1	x

(e) $1 \cdot g_1$

Fig.2.3 Maps for Adder using Type 1 PLA

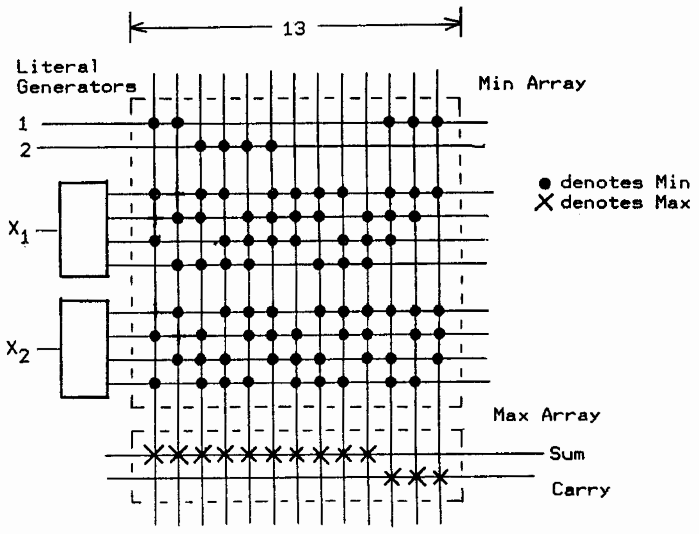


Fig. 2.4 Adder using Type 1 PLA

Minimization of logical expression can be done by a K-cover method[SMI 77] which includes MINI, MINI-II and ESPRESSO-MV.

Example 2.1: Let's design an adder of 4-valued logic shown in Table 2.1. In this case, the minimum universal set of literals is generated by a literal generator shown in Fig.2.2. By using the map shown in Fig. 2.3, we can obtain the minimum sum-of-products expressions for Sum and Carry functions as follows:

Sum $=1\cdot g_1 \vee 2\cdot g_2 \vee g_3$, where

$$g_1=X_1^{\{1,3\}}\cdot X_2^{\{0,2\}} \vee X_1^{\{0,2\}}\cdot X_2^{\{1,3\}},$$

$$g_2=X_1^{\{2\}}\cdot X_2^{\{0\}} \vee X_1^{\{1\}}\cdot X_2^{\{1\}} \vee X_1^{\{0\}}\cdot X_2^{\{2\}} \vee X_1^{\{3\}}\cdot X_2^{\{3\}},$$

$$g_3=X_1^{\{3\}}\cdot X_2^{\{0\}} \vee X_1^{\{2\}}X_2^{\{1\}} \vee X_1^{\{1\}}\cdot X_2^{\{2\}} \vee X_1^{\{0\}}\cdot X_2^{\{3\}}.$$

Carry $=1\cdot g_4$, where

$$g_4=X_1^{\{3\}}X_2^{\{1\}} \vee X_1^{\{2,3\}}\cdot X_2^{\{2\}} \vee X_1^{\{1,2,3\}}X_2^{\{3\}}.$$

Fig.2.4 shows the PLA realizing the adder. Note that 13 products are used in this PLA. (End of example).

2.2 Physical Implementation

The Max and the Min arrays are easily implemented by bipolar technology, but they require many transistors if realized by MOS technology. Therefore, this structure is unsuitable for MOS implementation.

III. PLA with AND-OR Arrays Followed by Output Encoders.

3.1 Logical Implementation

Fig.3.1 shows an n-input m-output p-valued PLA with AND-OR arrays followed by output encoders. We call this PLA a Type 2 PLA. Similar to Type 1 PLA, each p-valued signal X_i is converted into the minimum universal set of literals. Then, these literals are used in the AND and the OR arrays to realize mr two-valued functions $h_0, h_1, \ldots, h_{mr-1}$, where $r=\lceil \log_2 p \rceil$, and $\lceil \log_2 p \rceil$ denotes the least integer greater than or equal to $\log_2 p$. Finally, these two-valued signals are converted into p-valued signals by the output encoders.

For simplicity, suppose that m=1 and $p=2^r$. Then Fig.3.1 represents a MVL function:

$$f=0\cdot g_0 \vee 1\cdot g_1 \vee \ldots \vee (p-1)g_{p-1}. \quad \text{--------} \quad (3.1)$$

The sub-functions $g_0, g_1, \ldots, g_{p-1}$ are represented by

$$\begin{aligned}
g_0 &= \bar{h}_0\cdot\bar{h}_1\cdot\ldots\cdot\bar{h}_{r-2}\cdot\bar{h}_{r-1} \quad ,\\
g_1 &= \bar{h}_0\cdot\bar{h}_1\cdot\ldots\cdot\bar{h}_{r-2}\cdot h_{r-1} \quad ,\\
g_2 &= \bar{h}_0\cdot\bar{h}_1\cdot\ldots\cdot h_{r-2}\cdot\bar{h}_{r-1} \quad ,\\
g_3 &= \bar{h}_0\cdot\bar{h}_1\cdot\ldots\cdot h_{r-2}\cdot h_{r-1} \quad ,\\
&\ldots\ldots\ldots\ldots\ldots \qquad\qquad \text{-----}(3.2)\\
g_{p-2} &= h_0\cdot h_1\cdot\ldots\cdot h_{r-2}\cdot\bar{h}_{r-1} \quad ,\\
g_{p-1} &= h_0\cdot h_1\cdot\ldots\cdot h_{r-2}\cdot h_{r-1} \quad ,
\end{aligned}$$

where $\bar{h}_i=(p-1)-h_i$. Each of sub-sub functions $h_0, \ldots, h_{r-1}$ is represented by an expression:

$$h_j= \vee X_1^{S_1}\cdot X_2^{S_2}\cdot\ldots\cdot X_n^{S_n}, (j=0,1,\ldots,r-1) \text{---}(3.3)$$

where $S_i \subseteq \{0,1,2,\ldots,p-1\}$.

In Type 2 PLA, each column realizes a product $X_1^{S_1}\cdot X_2^{S_2}\cdot\ldots\cdot X_n^{S_n}$.

The output encoder accepts $h_0, h_1, \ldots, h_{r-1}$, and generates a p-valued signal according to (3.1) and (3.2).

Theorem 3.1: Let W be the number of columns necessary to realize an arbitrary p-valued function in a Type 2 PLA, then $W \leq \lceil \log_2 p \rceil \cdot p^{n-1}$.

(Proof) The number of columns of a Type 2 PLA is equal to the distinct number of the products in the sub-sub functions. It is clear that each sub-sub function h_j can be realized by at most p^{n-1} products. Hence, we need at most $r\cdot p^{n-1}$ products to represent(3.1). (Q.E.D.)

Example 3.1: Let's design the adder of 4-valued logic shown in Table 2.1. Suppose that the 4-valued output signal is represented by a pair of 2-valued signals as shown in Table 3.1. Then the function to be realized by the arrays can be represented as Table 3.2. By using the maps shown in Fig.3.2, we can obtain the minimum sum-of-products expressions for s_1, s_0 and c_0 as follows:

$$s_1 =X_1^{\{2,3\}}\cdot X_2^{\{0\}} \vee X_1^{\{1,2\}}\cdot X_2^{\{1\}} \vee X_1^{\{0,1\}}\cdot X_2^{\{2\}} \vee X_1^{\{0,3\}}\cdot X_2^{\{3\}},$$

$$s_0 =X_1^{\{1,3\}}\cdot X_2^{\{0,2\}} \vee X_1^{\{0,2\}}\cdot X_2^{\{1,3\}} \quad ,$$

$$c_0 =X_1^{\{3\}}\cdot X_2^{\{1\}} \vee X_1^{\{2,3\}}\cdot X_2^{\{2\}} \vee X_1^{\{1,2,3\}}\cdot X_2^{\{3\}}.$$

Fig.3.3 shows the PLA realizing the adder. Note that 9 products are used in this PLA. (End of example).

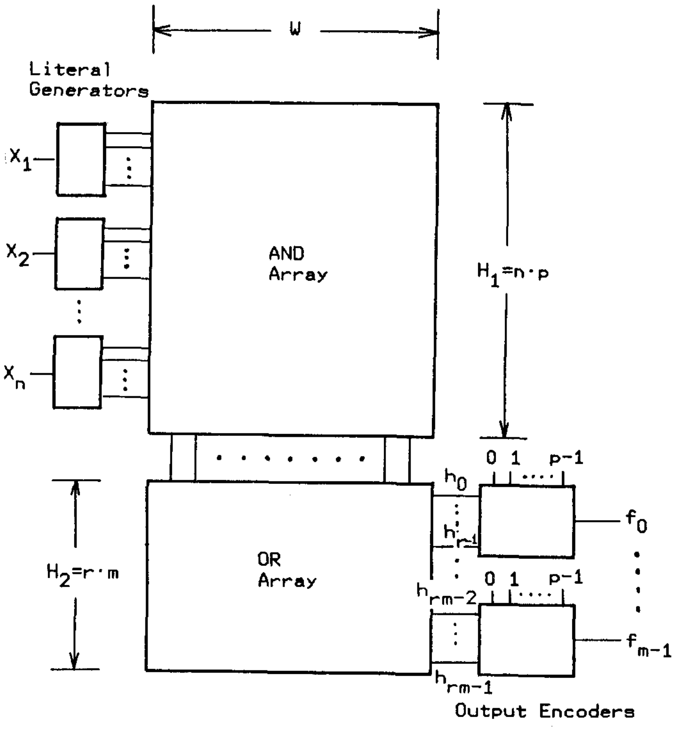


Fig.3.1 p-valued PLA with Output Encoders (Type 2 PLA)

Table 3.1 Output Encoding for 4-valued Adder

4-valued signal	2-valued signals
0	0 0
1	0 3
2	3 0
3	3 3

(a) S_1

X_1 \ X_2	0	1	2	3
0			3	3
1		3	3	
2	3	3		
3	3			3

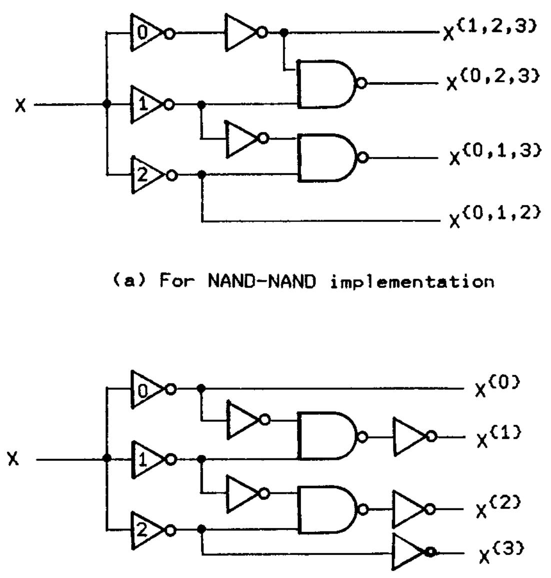
(b) S_0

X_1 \ X_2	0	1	2	3
0		3		3
1	3		3	
2		3		3
3	3		3	

(c) C_0

X_1 \ X_2	0	1	2	3
0				
1				3
2			3	3
3		3	3	3

Fig.3.2 Maps for Adder using Type 2 PLA



(a) For NAND-NAND implementation

(b) For NOR-NOR implementation

Fig. 3.4 Literal Generator

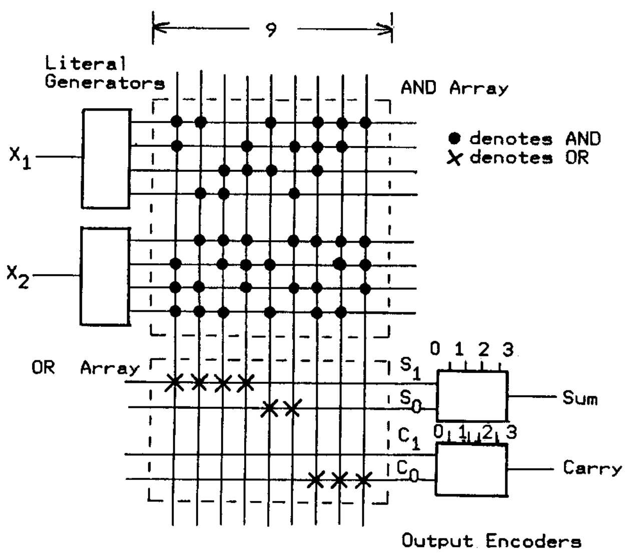


Fig.3.3 Adder using Type 2 PLA

Table 3.2 Truth Table for 4-valued Adder

Input		Output			
		Sum		Carry	
X_1	X_2	S_1	S_0	C_1	C_0
0	0	0	0	0	0
0	1	0	3	0	0
0	2	3	0	0	0
0	3	3	3	0	0
1	0	0	3	0	0
1	1	3	0	0	0
1	2	3	3	0	0
1	3	0	0	0	3
2	0	3	0	0	0
2	1	3	3	0	0
2	2	0	0	0	3
2	3	0	3	0	3
3	0	3	3	0	0
3	1	0	0	0	3
3	2	0	3	0	3
3	3	3	0	0	3

Table 3.3 Invertors with Various Thresholods

Input	Output			
	0	1	2	
0	3	3	3	3
1	0	3	3	-
2	0	0	3	-
3	0	0	0	0

3.2 Physical Implementation

The AND and the OR arrays are easily implemented by both bipolar and MOS technology. When we realize a large PLA, dynamic CMOS circuit is the most attractive technology[LAW 82]. In this case, an NOR-NOR structure is used to implement the AND and the OR arrays. When we need an extremely low power system, a static CMOS circuit is also feasible[POW 84]. In this case, an NAND-NAND structure is used to implement the AND and the OR arrays to take advantage of the n-channel device in the serial device path.

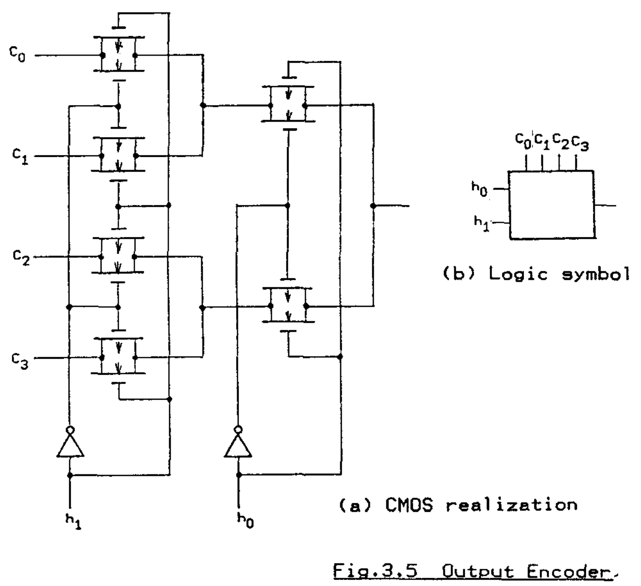
For example, when p=4, Type 2 PLA can be implemented as follows:

1) The literal generator is implemented as shown in Fig.3.4. For the NAND-NAND structure, which is logically equivalent to the AND-OR structure, we use Fig.3.4(a). While, for the NOR-NOR structure, which is logically equivalent to the OR-AND structure, every binary signal must be complemented and so we use Fig.3.4(b). The input-output relations of the inverters having different thresholds are shown in Table 3.3. These inverters can be realized either by an ion implantation technique for n-MOS[KAM 85], or CMOS [ZUK 85], or by voltage divider circuits using transistors[MCC 80].
2) The output encoder using a CMOS circuit is shown in Fig.3.5(a) and denoted by the symbol shown in Fig.3.5(b). For an NAND-NAND structure, we set $C_0=0$, $C_1=1$, $C_2=2$, and $C_3=3$, and for an NOR-NOR structure, we set $C_0=3$, $C_1=2$, $C_2=1$, and $C_3=0$.

3.3 Comparison of Type 1 PLA with Type 2 PLA.

Table 3.4 compares Type 1 PLA's with Type 2 PLA's. Because Type 2 PLA is easily implemented by MOS/CMOS technology, it is more suitable for VLSI than Type 1 PLA. Bounds on the number of distinct functions realized by both types of PLA's are derived in Appendix, and summarized in the table. Table 4.4 compares the size of PLA's for randomly generated functions, and shows that Type 2 PLA requires fewer products.
Although these results suggests that Type 2 PLA's usually require smaller arrays than Type 1 PLA's, it needs further study to verify it. Indeed, there is a function whose Type 1 PLA realization requires smaller arrays than Type 2 PLA. (see Addendum of [TIR 84] distributed at ISMVL-84). Comparison of the complexities of these PLA's is quite interesting. Similar study can be found in [BEN 85].

From the next section, we will consider the design of Type 2 PLA in detail.



(a) CMOS realization

(b) Logic symbol

Fig.3.5 Output Encoder.

Table 3.4 Comparison of Type 1 PLA with Type 2 PLA to Realize p valued Functions

		Type 1 PLA	Type 2 PLA
Structure		Literal generators Min-array,Max-array	Literal generators AND-array,OR-array Output encoders
Signals in array		p-valued	two-valued
Technology		Bipolar	MOS/CMOS/Bipolar static/dynamic
Array size	H_1	np+p-2	np
	H_2	m	rm
	W *	$\leqq (p-1)p^{n-1}$	$\leqq r\cdot p^{n-1}$
Number of realizable functions	UB	$(p-1)^W\cdot t^{nW}$	$t^W\cdot t^{nW}$ +
	LB	$(p-1)^W\cdot t^{W(n-r)}$	$t^W\cdot t^{W(n-r)}$ +
Design method		g_0 can be omitted. $g_i(i=1,2,\ldots,h_{r-1})$ are minimized by using g_s as don't care sets. $(k+1\leqq s\leqq p-1)$	g_i $(i=0,1,\ldots,p-1)$ are realized by $h_0,h_1,\ldots,h_{r-1}$. These expressions can be minimized simultaneously. Optimal output encodig often reduces array size

n:number of input variables
m:number of output variables
W:number of columns for PLA
*:when m=1
UB:upper bound
LB:lower bound
+:$p=2^r, p\geqq 4$
$t=2^p-1, r=[\log_2 p]$
[a] denotes the least interger equal to or greater than a

IV. Output Encoding Problem

4.1 Optimal Output Encoding for Adder

The concept of the optimal output encoding problem for Type 2 PLA is illustrated by the following.

Example 4.1: In realizing the adder in Example 3.1, we assign a pair of two-valued signals to represent a 4-valued signal as shown in Table 3.1. However, if we assign signals as shown in Table 4.1, we have Table 4.2. By using maps shown in Fig.4.1, we have the minimum sum-of-products expression for h_1,h_0, and c_0 as follows:

$$h_1=X_1^{\{0,2\}}\cdot X_2^{\{0,2\}}\vee X_1^{\{1,3\}}\cdot X_2^{\{1,3\}},$$
$$h_0=X_1^{\{0,1\}}\cdot X_2^{\{0\}}\vee X_1^{\{0,3\}}\cdot X_2^{\{1\}}\vee X_1^{\{2,3\}}\cdot X_2^{\{2\}}\vee X_1^{\{1,2\}}\cdot X_2^{\{3\}},$$
$$C_0=X_1^{\{2,3\}}\cdot X_2^{\{2\}}\vee X_1^{\{1,2\}}\cdot X_2^{\{3\}}\vee X_1^{\{3\}}\cdot X_2^{\{1,3\}}.$$

Fig. 4.2. shows the Type 2 PLA for this function. Note that the first two terms of C_0 are shared with h_0. In this PLA, only 7 columns are used to realize the function. In this case, we need to permute the connection of constants in the output encoder to obtain the proper output values.

(End of example).

Table 4.1 Optimum Output Encoding of Adders for Type 2 PLA

(a) Encoding for Sum

4-valued signal	2-valued signals	
0	3	3
1	0	3
2	3	0
3	0	0

(b) Encoding for Carry

4-valued signal	2-valued signals	
0	0	0
1	0	3
2	3	0
3	3	3

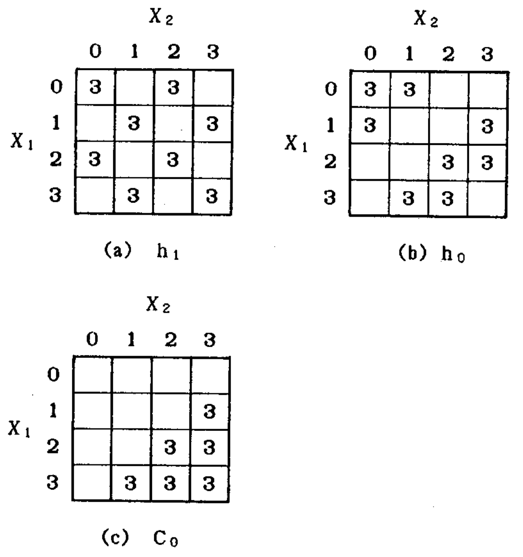


Fig.4.1 Maps for Adder using Type 2 PLA with optimal output encoding

Table 4.2 Truth Table for Adder (Output encoding Optimum)

Input		Output			
		Sum		Carry	
X_1	X_2	h_1	h_0	C_1	C_0
0	0	3	3	0	0
0	1	0	3	0	0
0	2	3	0	0	0
0	3	0	0	0	0
1	0	0	3	0	0
1	1	3	0	0	0
1	2	0	0	0	0
1	3	3	3	0	3
2	0	3	0	0	0
2	1	0	0	0	0
2	2	3	3	0	3
2	3	0	3	0	3
3	0	0	0	0	0
3	1	3	3	0	3
3	2	0	3	0	3
3	3	3	0	0	3

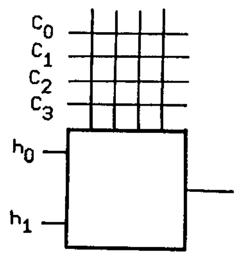


Fig. 4.3 Programmable Output Encoder(4-valued)

4.2 Optimal Output Encoding Problem for MVPLA

As was illustrated in the Example 4.1, different output encodings derive PLA's with different complexities. Suppose that we can use programmable output encoders shown in Fig.4.3. In such a case, we can use any output encoding for each output.

Definition 4.1: The optimal output encoding of a Type 2 PLA is a set of encodings which makes the size of the arrays minimum.

For a p-valued single-output function, there are p! different ways of encodings. The exhaustive way to find an optimum output encoding requires p! minimizations. For p=4, the number is 4!=24. This value can be reduced to 12 by considering the symmetry of the sub-sub functions. Table 4.3 lists the 12 essentially different output encodings.

As for the optimum output encoding problem for an m-output function, the exhaustive search requires $(p!/\lceil \log_2 p\rceil)^m$ minimizations, which is impractical for large problems. By using a heuristic method similar to [SAS 84], we can obtain the encoding shown in Table 4.1. This encoding has been verified to be optimum by the exhaustive examination by using a computer program.

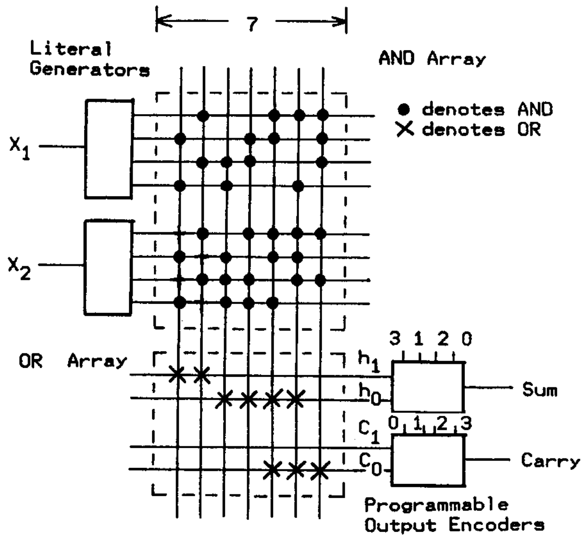


Fig.4.2 Adder using Type 2 PLA with Optimal Output Encoding

4.3 Computer Simulation

Table 4.4 compares the numbers of products of Type 1 PLA's, Type 2 PLA's with original output encodings, and Type 2 PLA's with optimum output encoding. Randomly generated functions were used to compare the complexites of PLA's. For each function, the number of input combinations which are mapped into i (i=0,1,2,3) are equal to 4^{n-1}, where n is the number of the input variables. The optimum output encodings were obtained by the exhaustive method. Minimization of the expressions were done by QM (modified Quine-McCluskey method[SAS 86b]) for n=2 and 3 and byMINI-IIfor n=4 and 5. When n=5, output encoding optimum PLA's require on the average 3.7% fewer products than output encoding original PLA's. In most cases, Type 1 PLA's require more products than Type 2 PLA with output encoding original.

Table 4.3 Essentially Different Output Encodings

Value	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12
0	00	00	00	03	03	03	03	03	03	33	33	33
1	03	03	33	00	00	30	30	33	33	00	03	03
2	30	33	03	30	33	00	33	00	30	03	00	30
3	33	30	30	33	30	33	00	30	00	30	30	00

Table 4.4 Average Number of Products 4-valued PLA's

	Type 1	Type 2 PLA	
	PLA	Original Encoding	Optimum Encoding
n=2	6.4	6.1	5.3
n=3	20.0	19.1	17.4
n=4	67.4	64.4	61.0
n=5	251.5	244.3	235.3

Average of 10 randomly generated functions. The number of input combinations are mapped into i (i=0,1,2,3) are equal.

V. Optimal Input Encoding Problem

In a two-valued PLA with two-bit decoders, the size of the arrays can be reduced by considering the assignment of the input variables to the decoders[SAS 81],[SAS 84]. In a MVPLA having the structure shown in Fig.5.1, the size of the array can be reduced by using the similar technique.

Example 5.1: Suppose that the adder shown in Table 2.1 is realized by the PLA having a structure shown in Fig.5.1. In this case, each literal generator generates two literals as shown in Fig.5.2. In addition, we use the two-bit decoder shown in Fig.5.3. Between the literal generators and the two-bit decoders, we use a permutation network. Now, introduce 4 independent two-valued variables y_1, y_2, y_3, and y_4, to represent X_1 and X_2.

Let $y_1=X_1^{\{2,3\}}$, $y_2=X_1^{\{1,3\}}$, $y_3=X_2^{\{2,3\}}$, and $y_4=X_2^{\{1,3\}}$.

Then, $\bar{y}_1=X_1^{\{0,1\}}$, $\bar{y}_2=X_1^{\{0,2\}}$, $\bar{y}_3=X_2^{\{0,1\}}$, and $\bar{y}_4=X_2^{\{0,2\}}$.

By using the new variables y_1, y_2, y_3, and y_4, the adder can be represented as shown in Table 5.1. Next, introduce two super variables $Y_1=(y_1,y_3)$ and $Y_2=(y_2,y_4)$. Then, S_1, S_0 and C_0 can be represented by maps shown in Fig.5.4(a)-(c). From these maps, we have the minimum sum-of-products expression:

$$S_1=Y_1^{\{03,30\}}\cdot Y_2^{\{00,03,30\}} \vee Y_1^{\{00,33\}}\cdot Y_2^{\{33\}},$$
$$S_0=Y_2^{\{03,30\}},$$
$$C_0=Y_1^{\{33\}} \vee Y_1^{\{03,30\}}\cdot Y_2^{\{33\}}.$$

The PLA realizing these functions requires only 5 products.

By optimizing the output encodings, the size of the PLA is further reduced. When we use the output encodings shown in Table 5.2, we have the PLA with only 4 columns. In this case, $\bar{C}_0$ is realized intead of C_0.

As shown in Fig.5.4(d), $\bar{C}_0$ can be written as

$$\bar{C}_0 = Y_1^{\{03,30\}}\cdot Y_2^{\{00,03,30\}} \vee Y_1^{\{00\}}$$

Fig.5.5. shows the PLA realizing a 4-valued adder. Note that the first term of $\bar{C}_0$ is shared with S_1.

(End of example)

Table 5.3 compares the size of the 4-valued adders. Type 2 PLA's require smaller arrays than Type 1 PLA's. Although Type 2 PLA's with two-bit decoders require additional hardware, the size of the arrays are much smaller than one's without two-bit decoders.

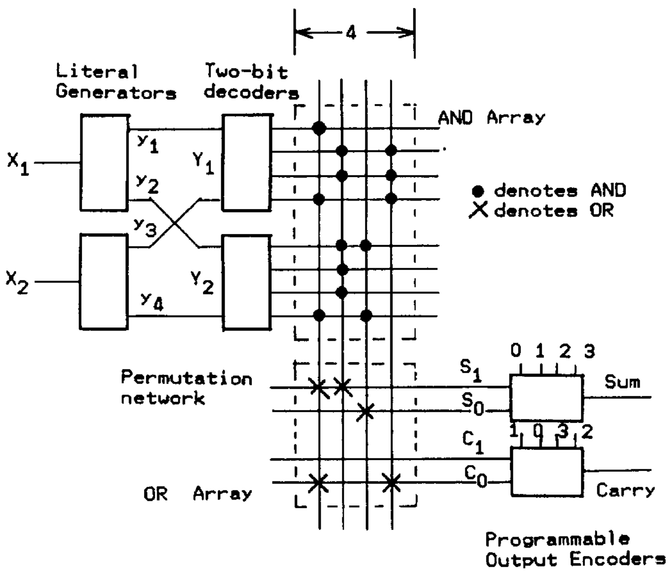


Fig.5.5 Adder using Type 2 PLA

Table 5.1 Truth Table of Adder for Type 2 PLA with Two-bit Decoders

X_1		X_2		Sum		Carry	
y_1	y_2	y_3	y_4	S_1	S_0	C_1	C_0
0	0	0	0	0	0	0	0
0	0	0	3	0	3	0	0
0	0	3	0	3	0	0	0
0	0	3	3	3	3	0	0
0	3	0	0	0	3	0	0
0	3	0	3	3	0	0	0
0	3	3	0	3	3	0	0
0	3	3	3	0	0	0	3
3	0	0	0	3	0	0	0
3	0	0	3	3	3	0	0
3	0	3	0	0	0	0	3
3	0	3	3	0	3	0	3
3	3	0	0	3	3	0	0
3	3	0	3	0	0	0	3
3	3	3	0	0	3	0	3
3	3	3	3	3	0	0	3

Table 5.3 Comparision of Size of Adders

		Type 1 PLA	Type 2 PLA		Type 2 PLA with two-bit decoders	
			encoding original	encoding optimum	encoding original	encoding optimum
One-figure adder X_0 +) Y_0 $Z_1 Z_0$	H_1	10	8	8	8	8
	H_2	2	4	4	4	4
	W	13	9	7	5	4
	S	156	108	84	60	48
Two-figure adder X_1X_0 +) Y_1Y_0 $Z_2Z_1Z_0$	H_1	18	16	16	16	16
	H_2	3	6	6	6	6
	W	79	63	52	17	14
	S	1659	1386	1144	374	308

$S=W\cdot(H_1+H_2)$

Table 5.2 Optimal Output Encoding for Adder for Type 2 PLA with Two-bit Decoders

(a) Encoding for Sum

4-valued signal	2-valued signals	
0	0	0
1	0	3
2	3	0
3	3	3

(b) Encoding for Carry

4-valued signal	2-valued signals	
0	0	3
1	0	0
2	3	3
3	3	0

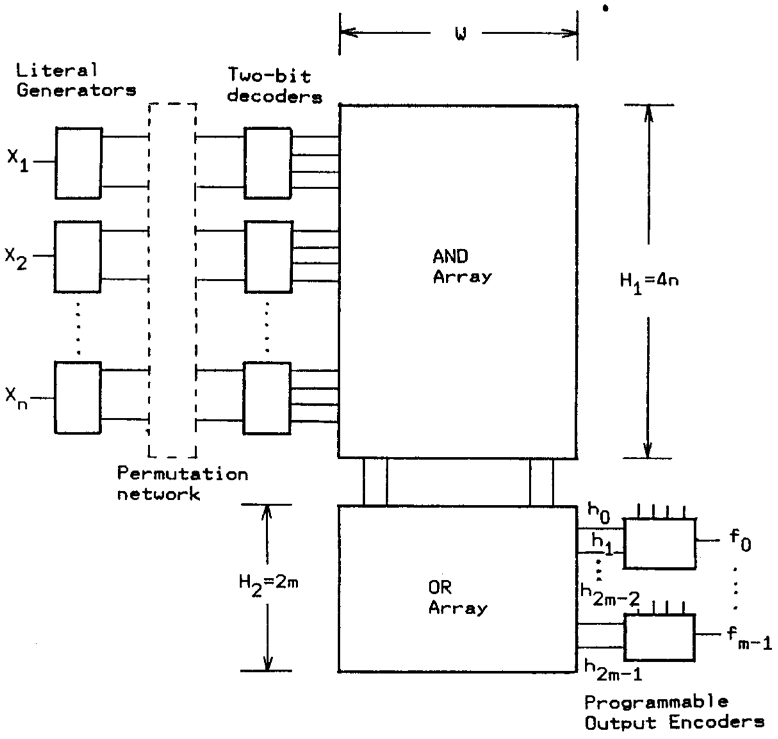
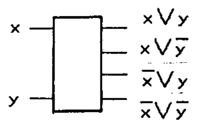
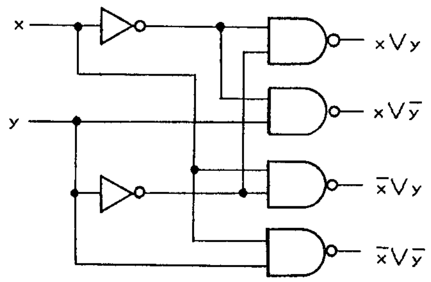


Fig.5.1 4-valued PLA with Two-Bit Decoders and Programmable Output Encoders

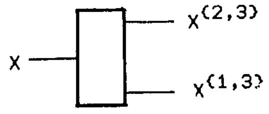


(a) Logic Symbol

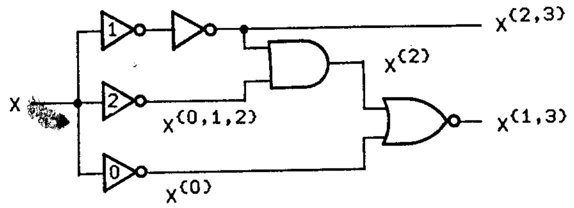


(b) Circuit Realization

Fig.5.3 Two-bit Decoder



(a) Logic Symbol



(b) Circuit Realization

Fig.5.2 Literal Generator

$Y_2=(y_2,y_4)$ across, $Y_1=(y_1,y_3)$ down

	00	03	30	33
00				3
03	3	3	3	
30	3	3	3	
33				3

(a) S_1

$Y_2=(y_2,y_4)$ across, $Y_1=(y_1,y_3)$ down

	00	03	30	33
00		3	3	
03		3	3	
30		3	3	
33		3	3	

(b) S_0

$Y_2=(y_2,y_4)$ across, $Y_1=(y_1,y_3)$ down

	00	03	30	33
00				
03				3
30				3
33	3	3	3	3

(c) C_0

$Y_2=(y_2,y_4)$ across, $Y_1=(y_1,y_3)$ down

	00	03	30	33
00	3	3	3	3
03	3	3	3	
30	3	3	3	
33				

(d) $\overline{C_0}$

Fig.5.4 Maps for Adder using Type 2 PLA with Two-bit Decoders

VI. Conclusion and Comparison with Other Methods

In this paper, the author proposed two types of PLA's: Type 1 PLA and Type 2 PLA. . Because both PLA's use the minimum universal set of literals, they require the minimum number of literal lines.

The array structure for 4-valued logic proposed by [IMM 85] uses 14 literals for each input. On the other hand, the method in this paper uses only 4 literals. Thus, the height of the Min array in this paper is about 29% of [IMM 85]. The 4-valued PLA which can be obtained by extending[KUO 85] uses only 4 literals ($X^{\{0\}}, X^{\{1\}}, X^{\{2\}}, X^{\{3\}}$). However, the Min array using these literals cannot be minimized at all. On the other hand, Type 1 PLA proposed in this paper uses the minimum universal set of literals

$$(X^{\{1,2,3\}}, X^{\{0,2,3\}}, X^{\{0,1,3\}}, X^{\{0,1,2\}}),$$

and the Min array can usually be minimized into the half of [KUO 85] or even smaller. Indeed, Table 2 of [IMM 85] implies that the number of columns for Type 1 PLA is ,on the average, 50% to 60% of [KUO 85]. Thus, the Type 1 PLA proposed in this paper usually requires much smaller arrays than previously published ones[KUO 85],[IMM 85].

Although, Type 1 PLA is easy to implement by bipolar technology, it is unsuitable for MOS realization. Type 2 PLA, which is also proposed in this paper, is suitable for MOS/CMOS implementation. Complexity analysis and logical capability analysis suggest that Type 2 PLA's are, on the average, smaller than Type 1 PLA's. However, we need further study on which realization reqires smaller arrays. This is an interesting open problem, which is similar to the one discussed by [TIR 84].

[ACKNOWLEDGEMENT]

The author is grateful to the referees for their detailed and constructive comments.
Part of this work was supported by Grant in Aid for Scientific Research of the Ministry of Education, Science, and Culture of Japan.

[Reference]

[BEN 85] E.A.Bender, J.T.Butler, and H.G.Kerkhoff, 'Comparing the sum with the max for use in four-valued PLA's,' ISMVL-85, pp.30-35,May 1985.

[DAN 85] R.G.Daniels and W.C.Bruce, 'Bulit-in self-test trends in Motorola microprocessors,' IEEE Design and Test, pp.64-71, April 1985.

[FLE 83] H.Fleisher,'The implementation and use of multivalued logic in a VLSI environment,' ISMVL-83, pp.138-143, May 1983.

[FRE 84] D.A.Freitas and K.W.Current,'CMOS circuit for quaternary encoding and decoding,' ISMVL-84, pp.164-168, May 1984.

[HON 74] S.J.Hong, R.G.Cain, and D.L.Ostapko,'MINI: A heuristic approach for logic minimization,'IBM J. Res. and Develop. pp.443-458, Sept. 1974.

[HUR 84] S.L. Hurst, 'Multiple-value logic: Its status and its future,' IEEE Trans. Comput. Vol.C-33, No.12, pp.1160-1179, Dec. 1984.

[IMM 85] M.Imme and C.A.Papachristou,'Simplification of MVL functions and implementation via a VLSI array structure,'ISMVL-85, pp.242-248, May 1985.

[KAM 85] M.Kameyama, T.Hanyu, M.Esashi, T.Higuchi, and T.Ito,'Implementation of quaternary NMOS integrated circuits for pipelined image processing,' ISMVL-85, pp.226-232, May 1985.

[KUO 85] H-L Kuo and K-Y Fang,'The multiple-valued programmable logic array and its application in modular design,' ISMVL-85, pp.10-18, May 1985.

[LAW 82] H.F.Law and M.Shoji,'PLA design of the BELLMAC-32A microprocessor,' ICCC-82, PP.161-164, 1982.

[MCC 80] E.J.McCluskey,'Logic design of MOS ternary logic,'ISMVL-80,pp.1-5,June 1980.

[POW 84] S.Powell, E.Iodice, and E.Friedman, 'An automated, low power, high speed complementary PLA design system for VLSI application,'ICCD-84, pp.314-319, 1984.

[RUD 85] R.Rudell and A.L.M.Sangiovanni-Vincentelli,'ESPRESSO-MV:Algorithms for multiple-valued logic minimization,' 1985 Custom Integrated Circuits Conference, pp.230-234, May 1985.

[SAS 81] 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.

[SAS 84] T.Sasao,'Input variable assignment and output phase optimization of PLA's,'IEEE Trans. Comput. Vol. C-33, No.10,pp.879-894, Oct. 1984.

[SAS 86a] T.Sasao, Programmable Logic Array: How to make and How to Use, NIKKAN KOGYOU Pub. Co.Tokyo, April 1986.(to be published in Japanese).

[SAS 86b] T.Sasao, 'MACDAS: Multi-level AND-OR circuit synthesis using two-variable function generators,' 23-rd Design Automation Conference, June 29-July 02, 1986.(to be published).

[SMI 77] W.R.Smith III,'Minimization of multivalued functions,' in Computer Science and Multiple-Valued Logic, D.C.Rine Ed. North Holland Publi. Co. 1977.

[TIR 84] P. Tirumalai and J.T.Butler,'On the realization of multiple-valued logic functions using CCD PLA's,' ISMVL-84,pp.33-42, May 1984 and Addendum distributed at ISMVL-84.

[ZUK 85] C.Zukeran, C.Afuso, M.Kameyama, and T.Higuchi,'Design of new low-power quaternary CMOS logic circuits based on multiple ion implants,' ISMVL-85, pp.84-90, May 1985.

APPENDIX

A.1 Minimum Universal Set of Literals for MVPLA's.

Definition A.1: Let $L=\{X^{S_0}, X^{S_1}, \ldots, X^{S_{k-1}}\}$ be a set of literals of X. L is said to be universal if any other literal of X can be represented by an AND (or a Min) operation among the elements in L. L is said to be minimum if k is the minimum.

Theorem A.1: $L=\{X^{S_0}, X^{S_1}, \ldots, X^{S_{p-1}}\}$ is the minimum universal set of X, where $S_i=\overline{\{i\}}=P-\{i\}$, $i=0,1,\ldots,p-1$ and $P=\{0,1,\ldots,p-1\}$.

(Proof) L is universal: It is sufficient to show that any literal X^A of X can be represented by the AND operation of the elements in L. X^A can be represented by

$$X^A=\bigwedge_{i\in B} X^{S_i}, \text{ where } B=\overline{A}=P-A, \text{ and } P=\{0,1,\ldots,p-1\}. \qquad \text{-------(A.1)}$$

L is the minimum: There are 2^p literals including constant zero and constant p-1. In order to represent all the literals in the forms of (A.1), we need at least p different elements. To show that L is unique, assume, on the contrary, that there is another universal set L' which is different from L. Thus, there is i such that $X^{S_i}\notin L'$. Since L' is universal, it realizes X^{S_i} itself (as the product of literals in L'). That is,

$X^{S_i}=X^{S'_1}\cdot X^{S'_2}\cdot\ldots\cdot X^{S'_m}$ where $S'_j\in L'$. Because $S_i=S'_1\cap S'_2\cap\ldots\cap S'_m$ and $S_i=P-\{i\}$, every S'_j $(j=1,2,\ldots,m)$ contains all the elements in $P-\{i\}$, and at least one $S'_{j'}$ does not contain the element i. But, it follows that $S'_{j'}=S_i$, a contradiction. It must be that L is unique. (Q.E.D.)

A.2 On the Number of Functions Realized by a PLA with W Columns

Theorem A.2: Let A(n,p,W) be the number of distinct n-variable p-valued single output function realized by a Type 1 PLA with W columns. When $W=p^u$, $(p-1)^W\cdot t^{W(n-u)}<A(n,p,W)<(p-1)^W\cdot t^{nW}$, where $t=2^p-1$.

(Proof) Lower Bound: Consider a function f which can be represented by the expression:

$$f(X_1,X_2,\ldots,X_n)=\bigvee_{\underline{a}\in P^u}(i)\cdot g(\underline{a},X_{u+1},\ldots,X_n)\cdot X_1^{a_1}\cdot X_2^{a_2}\cdot\ldots\cdot X_u^{a_u}, \qquad \text{----(A.2)}$$

where $\underline{a}=(a_1,\ldots,a_u)$, and $i=1,2,\ldots,p-1$.

For every $\underline{a}\in P^u$, a function $(i)g(\underline{a},X_{u+1},\ldots,X_n)$ can be represented as

$$(i)g(\underline{a},X_{u+1},\ldots,X_n)=(i)X_{u+1}^{S_{u+1}}\cdot\ldots\cdot X_n^{S_n}, \qquad \text{---(A.3)}$$

where $S_k\subseteq P$ $(k=u+1,\ldots,n)$. Note that the total number of products in (A.2) is $W=p^u$. The number of distinct non-zero functions realized by (A.3) is $(p-1)\cdot t^{(n-u)}$, because for each set $(S_{u+1},\ldots,S_n)$ and for each i, there exists a unique function, and there are $t=2^p-1$ possible way to choose a subset S_i of P. The total number of distinct non-zero functions realized by (A.2) is at least $(p-1)^W\cdot t^{W(n-u)}$, because for each a every combination of i and S_k $(k=u+1,\ldots,n)$ in (A.3) will make distinct functions.

Upper Bound: f can be represented as follows:

$$f(X_1,X_2,\ldots,X_n)=\bigvee_{(S_1,S_2,\ldots,S_n)} g(S_1,S_2,\ldots,S_n)\cdot X_1^{S_1}\cdot X_2^{S_2}\cdot\ldots\cdot X_n^{S_n}, \qquad \text{----(A.4)}$$

where $g(S_1,S_2,\ldots,S_n)$ is $1,2,\ldots,$or $(p-1)$, and $S_k\subseteq P$ $(k=1,2,\ldots,n)$.

It is clear that the number of distinct functions realized by (A.4) with W products is at most $(p-1)^W\cdot t^{nW}$. (Q.E.D.)

Lemma A.1: Let B(n,p,W) be the number of distinct n-variable p-valued input 2-valued output functions represented by the expression

$$f(X_1,X_2,\ldots,X_n)=\bigvee X_1^{S_1}\cdot X_2^{S_2}\cdot\ldots\cdot X_n^{S_n} \qquad \text{------(A.5)}$$

with W products.

When $W=p^u$, $t^{W(n-u)}<B(n,p,W)<t^{Wn}$, where $t=2^p-1$.

(Proof) Lower Bound: Consider the expression which has the following form:

$$f(X_1,X_2,\ldots,X_n)=\bigvee_{\underline{a}\subseteq P^u} g(\underline{a},X_{u+1},\ldots,X_n)\cdot X_1^{a_1}\cdot X_2^{a_2}\cdot\ldots\cdot X_u^{a_u} \qquad \text{------(A.6)}$$

where $\underline{a}=(a_1,a_2,\ldots,a_u)$. A function $g(\underline{a},X_{u+1},\ldots,X_n)$ can be represented as

$$g(\underline{a},X_{u+1},\ldots,X_n)=X_{u+1}^{S_{u+1}}\cdot\ldots\cdot X_n^{S_n}, \qquad \text{-----(A.7)}$$

where $S_k\subseteq P$ and $k=u+1,\ldots,n$.

Note that the number of products in (A.6) is $W=p^u$. The number of distinct non-zero functions realized by (A.7) is $t^{(n-u)}$. Thus, the total number of distinct non-zero functions represented by (A.6) is at least $t^{W(n-u)}$ because for each $(S_{u+1},\ldots,S_n)$ in (A.7), we have distinct function.

Upper Bound: It is clear that the number of distinct functions realized by (A.5) with W products is at most t^{Wn} (Q.E.D.)

Theorem A.3: Let C(n,p,W) be the number of distinct p-valued single-output function realized by a p-valued Type 2 PLA with W columns. Then $C(n,p,W)=B(n+1,p,W)$, where $p=2^r$ and $p\geq 4$.

(Proof) The n-input r-output function realized by a Type 2 PLA can be represented as

$$f(X_1,X_2,\ldots,X_n,X_{n+1})=\bigvee_{(S_1,\ldots,S_{n+1})} X_1^{S_1}\cdot X_2^{S_2}\cdot\ldots\cdot X_n^{S_n}\cdot X_{n+1}^{S_{n+1}}, \qquad \text{---(A.8)}$$

where x_{n+1} denotes a variable for the outputs and it takes p values. It is clear that (A.8) also represents a (n+1)-input single output function. Therefore, the number of the functions realized by a Type 2 PLA with W columns is equal to the number of (n+1)-variable functions represented by an expression (A.8) with W products. (Q.E.D.)