Multiple-Valued Logic and Optimization of Programmable Logic Arrays

Tsutomu Sasao

Kyushu Institute of Technology

Since their introduction in 1970, programmable logic arrays (PLAs) have been used in large-scale-integration/very-large-scale-integration (LSI/VLSI) chips for pocket calculators, watches, and toys. Because PLAs have a regular structure, their design, test, and modification are quite easy. Modern 32-bit microprocessors such as the Motorola MC68020, Intel 80386, AT&T WE32100, and IBM Micro/370 use PLAs extensively in their control sections. Minimizing the number of columns in a PLA is important because it reduces not only the chip area but also the propagation delay and power dissipation.

Multiple-valued logic is used to design two-valued PLAs. A *standard PLA* (or a two-level PLA) has a memory-like structure, as shown in Figure 1. It realizes switching functions in sum-of-products form. Figure 2 shows an example of a standard PLA. The vertical lines realize the products (ANDs) of the input variables or their complements, and the horizontal line realizes the sum (OR) of the products. We can reduce the size of the PLAs by adding decoders to their input. A PLA with two-bit input decoders (see Figure 3a) is an extension of the standard PLA.[1] Figure 3b shows a schematic diagram of a two-bit input decoder. The PLA with two-bit decoders in Figure 4 realizes the same function as the standard PLA in Figure 2, but the PLA with two-bit decoders requires only three columns while the standard PLA requires seven columns. In general, a PLA with two-bit decoders requires 10 to 20 percent fewer columns than a standard PLA.

Multiple-valued input, two-valued output functions simplify the task of optimizing programmable logic arrays with input decoders.

This article will show a method of designing PLAs using *multiple-valued input, two-valued output functions* (MVITVOFs). A MVITVOF is an extension of the two-valued logic function. An expression for a MVITVOF directly represents a multiple-output PLA with decoders. Each product of the expression corresponds to each column of the PLA, so the number of products in the expression equals the number of columns of the PLA. The array size of the PLA is proportional to the number of products. Thus, we can minimize the PLA by minimizing the expression. Without a MVITVOF, minimizing multiple-output PLAs with decoders is difficult.

New optimization techniques for PLAs

Two new optimization techniques are the (1) assignment of the input variables to the decoders, and (2) output phase optimization. As an example of the first technique, consider the PLA in Figure 4. The variables x_1 and x_2 are connected to the first decoder, and x_3 and x_4 are connected to the second decoder. However, as shown in Figure 5, if x_1 and x_3 are connected to the first decoder, and x_2 and x_4 are connected to the second decoder, then the PLA requires only two columns to realize the same function. The specific assignment of the input variables to the decoders influences the size of the PLA. As an example of output phase optimization, consider the PLA in Figure 6. We can reduce this PLA by realizing the complement of f_0 and adding an inverter to the output f_0, as shown in Figure 7.

0018-9162/88/0400-0071$01.00 © 1988 IEEE

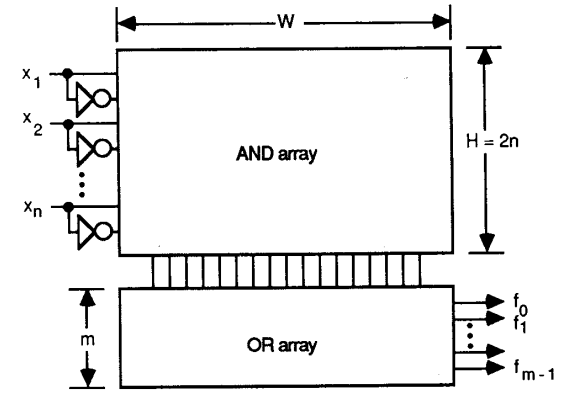


Figure 1. Standard PLA.

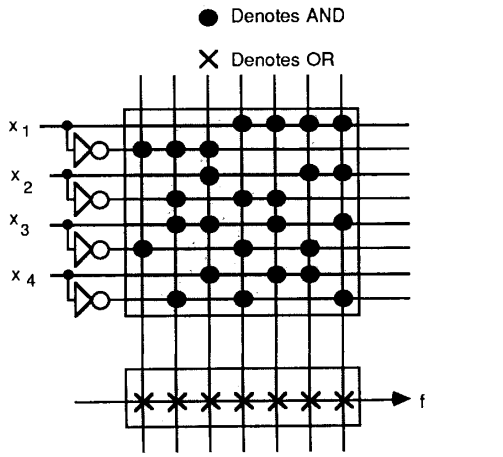
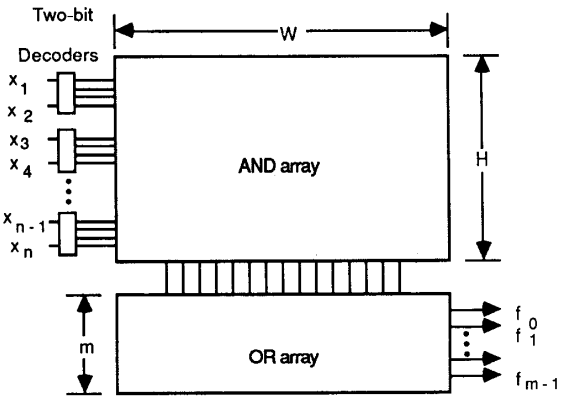
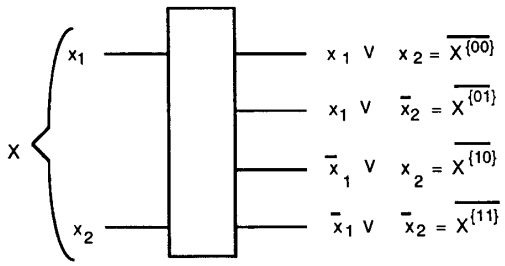


Figure 2. Standard PLA for table "Switching function" in sidebar "Example of PLAs with two-bit decoders."



(a)



(b)

Figure 3. (a) PLA with two-bit decoders. (b) Two-bit decoder.

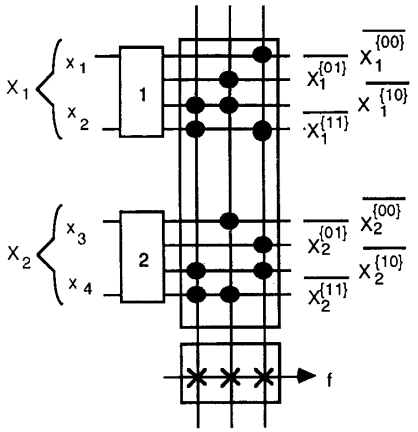


Figure 4. PLA with two-bit decoders for table "Switching function" in sidebar "Example of PLAs with two-bit decoders."

A standard PLA consists of the input inverters, the AND array, and the OR array, as shown in Figure 1. The *array size* of the PLA is defined as $W \cdot (H+m)$, where W is the number of columns of the PLA, H is the number of literal lines, and m is the number of outputs.

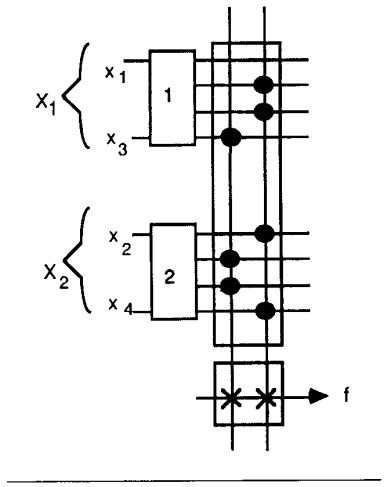


Figure 5. PLA with two-bit decoders for table "Switching function" in sidebar "Example of PLAs with two-bit decoders."

PLAs with decoders

When we replace the inverters of a standard PLA with decoders, we have a *PLA with decoders* (see Figures 3a and 3b).

In general, a t-bit decoder has 2^t outputs and produces all the maxterms of the input variables. The standard PLA is a special case of a PLA with decoders, being a PLA with one-bit decoders.

A PLA with t-bit decoders is represented by an expression of a MVITVOF, $F: P_1 \times P_2 \times \ldots \times P_n \rightarrow \{0, 1\}$, where $P_i = \{0, 1, \ldots, p-1\}$, and $p = 2^t$. When $t = 1$, F is an ordinary two-valued logic function. A minimum sum-of-products expression corresponds to a minimum two-level AND-OR network, or a minimum standard PLA. When $p = 2^t$, where $t \geq 2$, F represents a PLA with t-bit decoders, and a minimum sum-of-products expression for F corresponds to a minimum PLA with t-bit decoders.[2]

PLAs with decoders never require more columns than standard PLAs. In the case of a PLA with two-bit decoders, the number of literal lines equals that of a standard PLA. So, the array size of a PLA with two-bit decoders never exceeds that of a stan-

Table 1. Number of columns to realize n-variable functions by PLAs (n is even).

	Standard PLA	PLA with Two-Bit Decoders
Arbitrary Function (worst case)	2^{n-1}	2^{n-2}
Symmetric Function (worst case)	2^{n-1}	$3^{(n-2)/2}$
Parity Function (worst case)	2^{n-1}	$2^{(n-2)/2}$
Random Function of 10-Variable (average)	163	120

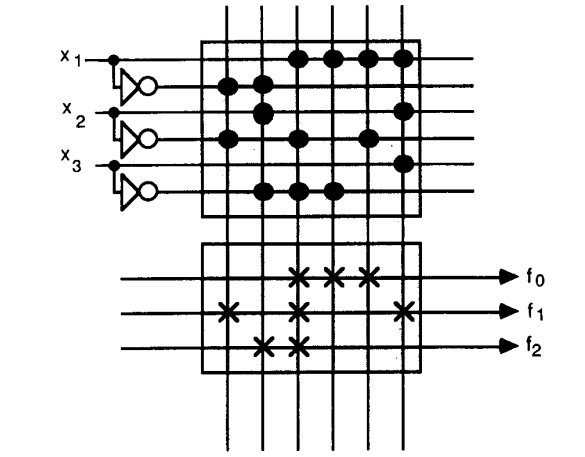


Figure 6. Standard PLA for table "Three-input three-output function" in sidebar "MVITVOF and its expressions."

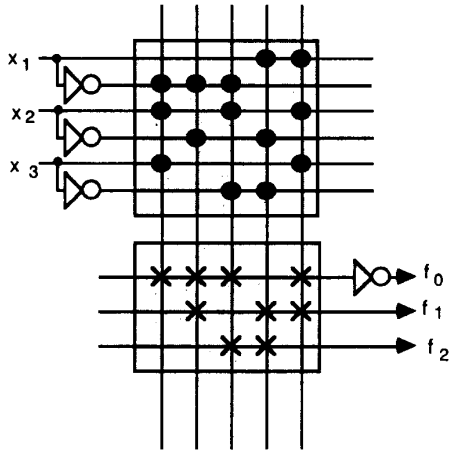


Figure 7. Output phase optimized PLA for table "Three-input three-output function" in sidebar "MVITVOF and its expressions."

dard PLA. Table 1 shows the number of columns needed to realize various classes of functions. When the function has a symmetric property, PLAs with two-bit decoders require much smaller arrays than standard PLAs. However, in the case of a PLA with t-bit decoders, where $t \geq 3$, the number of literal lines is greater than that of a standard PLA. So, a PLA with t-bit decoders ($t \geq 3$) can be larger than a standard PLA even if the number of columns is smaller.

The cost of the decoders is excluded from the PLA's array size; it depends on the technology of the circuit realizing the decoder as well as the size of the PLA. In a dynamic complementary metal oxide semiconductor (CMOS) PLA with 200 columns, the area for one-bit decoders is equivalent to 14 columns while the area for two-bit decoders is equivalent to 29 columns. The cost of a three-bit decoder is at least three times greater than that of a two-bit decoder for a CMOS realization. Thus, we have to add the area for the decoders to compare the actual chip size.

Assignment of input variables to decoders

The specific assignment of the input variables to the decoders often influences the size of the PLA with decoders. An optimum assignment minimizes the size. We can find the optimum assignment by considering all possible assignments. For a PLA with two-bit decoders, the number of combinations is $(n!)/(2^k \cdot k!)$, where $n = 2k$ is the number of input variables. When $n = 8$, the number of combinations is 105, and it is possible to obtain the optimum solution. I minimized many randomly generated functions of eight variables and obtained statistical data.[2] When the assignment of the input variables is nonoptimized, the array size of PLAs with two-bit decoders is 24 percent smaller than that of standard PLAs. When the assignment is optimum, the array size is 32 percent smaller.

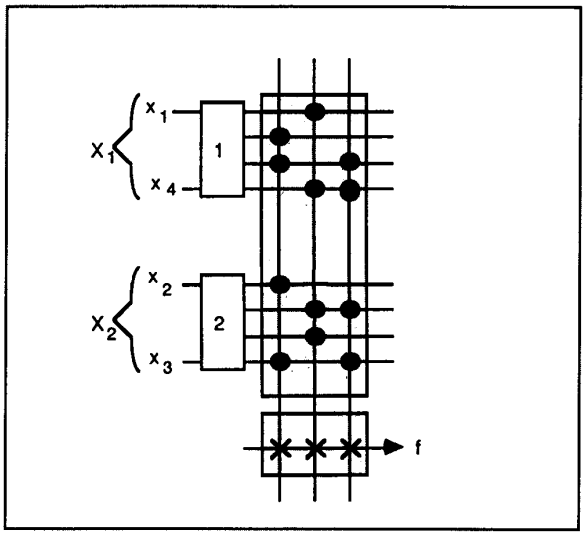
The exhaustive method takes too much computation time for functions with many input variables, so I have developed a heuristic algorithm that quickly obtains good solutions.[3] Table 2 shows the number of

(Continued on p. 78)

Example of PLAs with two-bit decoders

First, we will represent the switching function shown in the table below by using the conventional two-valued logic. Let $X_1 = (x_1)$, $X_2 = (x_2)$, $X_3 = (x_3)$, and $X_4 = (x_4)$ be variables of the MVITVOF. The switching function can be represented as follows:

$$\begin{aligned} f(X_1, X_2, X_3, X_4) = {} & X_1^{\{0\}}X_2^{\{0\}}X_3^{\{0\}}X_4^{\{0\}} \vee X_1^{\{0\}}X_2^{\{0\}}X_3^{\{0\}}X_4^{\{1\}} \vee \\ & X_1^{\{0\}}X_2^{\{0\}}X_3^{\{1\}}X_4^{\{0\}} \vee X_1^{\{0\}}X_2^{\{1\}}X_3^{\{0\}}X_4^{\{0\}} \vee \\ & X_1^{\{0\}}X_2^{\{1\}}X_3^{\{0\}}X_4^{\{1\}} \vee X_1^{\{0\}}X_2^{\{1\}}X_3^{\{1\}}X_4^{\{1\}} \vee \\ & X_1^{\{1\}}X_2^{\{0\}}X_3^{\{0\}}X_4^{\{0\}} \vee X_1^{\{1\}}X_2^{\{0\}}X_3^{\{1\}}X_4^{\{1\}} \vee \\ & X_1^{\{1\}}X_2^{\{1\}}X_3^{\{0\}}X_4^{\{1\}} \vee X_1^{\{1\}}X_2^{\{1\}}X_3^{\{1\}}X_4^{\{0\}} \end{aligned} \quad (1)$$



PLA with two-bit decoders for table "Switching function."

Note that in the conventional two-valued notation, $X_i^{\{0\}}$ is written more concisely as $\overline{X}_i$, and $X_i^{\{1\}}$ as X_i.

Equation 1 shows that f is equal to one if $\{X_1 = 0, X_2 = 0, X_3 = 0$, and $X_4 = 0\}$ or $\{X_1 = 0, X_2 = 0, X_3 = 0$, and $X_4 = 1\}$ or . . . or $\{X_1 = 1, X_2 = 1, X_3 = 1$, and $X_4 = 0\}$. Equation 1 can be simplified as follows:

$$\begin{aligned} f(X_1, X_2, X_3, X_4) = {} & X_1^{\{0\}}X_2^{\{0,1\}}X_3^{\{0\}}X_4^{\{0,1\}} \vee X_1^{\{0\}}X_2^{\{0\}}X_3^{\{1\}}X_4^{\{0\}} \vee \\ & X_1^{\{0\}}X_2^{\{1\}}X_3^{\{1\}}X_4^{\{1\}} \vee X_1^{\{1\}}X_2^{\{0\}}X_3^{\{0\}}X_4^{\{0\}} \vee \\ & X_1^{\{1\}}X_2^{\{0\}}X_3^{\{1\}}X_4^{\{1\}} \vee X_1^{\{1\}}X_2^{\{1\}}X_3^{\{0\}}X_4^{\{1\}} \vee \\ & X_1^{\{1\}}X_2^{\{1\}}X_3^{\{1\}}X_4^{\{0\}} \end{aligned} \quad (2)$$

Equation 2 shows that f becomes one if $\{X_1 = 0$ and $(X_2 = 0$ or $X_2 = 1)$ and $X_3 = 0$ and $(X_4 = 0$ or $X_4 = 1)\}$ or . . . or $\{X_1 = 1, X_2 = 1, X_3 = 1$, and $X_4 = 0\}$. Equation 2 is essentially the same as for an ordinary switching function. Figure 2 (main text) is a realization of the standard PLA for the table at left. In this PLA, each column of the AND array corresponds to each product of Equation 2.

Next, we will consider the four-valued input, two-valued output function. Let $X_1 = (x_1, x_2)$ and $X_2 = (x_3, x_4)$ be the variables of the function. Note that X_1 and X_2 are super-variables and take 00, 01, 10, or 11. $X_1 = 00$ means $x_1 = 0$ and $x_2 = 0$ at the decoder with X_1, and so on.

The expression for the function in the table is

$$\begin{aligned} f(X_1,X_2) = {} & X_1^{\{00\}}X_2^{\{00\}} \vee X_1^{\{00\}}X_2^{\{01\}} \vee X_1^{\{00\}}X_2^{\{10\}} \vee X_1^{\{01\}}X_2^{\{00\}} \vee \\ & X_1^{\{01\}}X_2^{\{01\}} \vee X_1^{\{01\}}X_2^{\{11\}} \vee X_1^{\{10\}}X_2^{\{00\}} \vee X_1^{\{10\}}X_2^{\{11\}} \vee \\ & X_1^{\{11\}}X_2^{\{01\}} \vee X_1^{\{11\}}X_2^{\{10\}} \end{aligned} \quad (3)$$

Equation 3 shows that f becomes one if $\{X_1 = 00$ and $X_2 = 00\}$ or $\{X_1 = 00$ and $X_2 = 01\}$ or . . . or $\{X_1 = 11$ and $X_2 = 10\}$. Note that

$$\begin{aligned} X_1^{\{00\}}X_2^{\{00\}} \vee X_1^{\{00\}}X_2^{\{01\}} &= X_1^{\{00\}} \cdot (X_2^{\{00\}} \vee X_2^{\{01\}}) \\ &= X_1^{\{00\}}X_2^{\{00,01\}} \text{ etc.} \end{aligned} \quad (4)$$

Table 2. Number of columns for arithmetic PLAs.

	Input Data			Standard PLA		PLA with Decoders Worst Input Assignment	PLA with Decoders Optimum Input Assignment	
Function	# In n	# Out m	# Cube	Output Phase Originally	Output Phase Optimized	Output Phase Originally	Output Phase Originally	Output Phase Optmized
$X+Y$	8	5	255	75	61	63	17	14
Log (X)	8	8	255	123	111	113	93	89
$X*Y$	8	8	225	121	108	107	86	73
SQRT (X^2+Y^2)	8	5	255	120	101	101	70	64
$5X+1$(mod 256)	8	8	255	76	76	59	47	47
SQRT (X)	8	5	255	57	48	52	37	32
X^2	8	16	255	180	165	159	142	134
ΣXi	8	4	255	255	186	54	54	38

Equation 3 is minimized to

$$f(X_1,X_2) = X_1^{\{00,01\}}X_2^{\{00,01\}} \vee X_1^{\{00,11\}}X_2^{\{01,10\}} \vee X_1^{\{01,10\}}X_2^{\{00,11\}} \quad (5)$$

Equation 5 shows that f becomes one if $\{(X_1 = 00 \text{ or } 01) \text{ and } (X_2 = 00 \text{ or } 11)\}$, or $\{(X_1 = 00 \text{ or } 11) \text{ and } (X_2 = 01 \text{ or } 10)\}$, or $\{(X_1 = 01 \text{ or } 10) \text{ and } (X_2 = 00 \text{ or } 11)\}$. Figure 4 (main text) shows the PLA with two-bit decoders, where the AND array corresponds to Equation 5. In fact, the first column realizes the first product because

$$\overline{X_1^{\{10\}}} \cdot \overline{X_1^{\{11\}}} \cdot \overline{X_2^{\{10\}}} \cdot \overline{X_2^{\{11\}}} = X_1^{\{00,01\}} \cdot X_2^{\{00,01\}} \quad (6)$$

In a similar way, the second column realizes the second product, and so on. Note that the PLA with two-bit decoders uses only three columns, while the standard PLA uses 7 columns.

Assignment of the input variables

Consider a PLA with two-bit decoders realizing the switching function in the table. There are three ways to assign four input variables to two two-bit decoders.

(1) When the input variables are assigned as $X_1 = (x_1, x_2)$ and $X_2 = (x_3, x_4)$ (see Figure 4), the minimum sum-of-products expression is

$$f(X_1, X_2) = X_1^{\{00,01\}}X_2^{\{00,01\}} \vee X_1^{\{00,11\}}X_2^{\{01,10\}} \vee X_1^{\{01,10\}}X_2^{\{00,11\}} \quad (7)$$

Three columns are necessary in this assignment.

(2) When the input variables are assigned as $X_1 = (x_1, x_3)$ and $X_2 = (x_2, x_4)$ (see Figure 5 in main text), the minimum sum-of-products expression is

$$f(X_1, X_2) = X_1^{\{00,01,10\}}X_2^{\{00,11\}} \vee X_1^{\{00,11\}}X_2^{\{01,10\}} \quad (8)$$

Two columns are necessary in this assignment.

(3) When the input variables are assigned as $X_1 = (x_1, x_4)$ and $X_2 = (x_2, x_3)$ (see the figure at left), the minimum sum-of-products expression is

$$f(X_1, X_2) = X_1^{\{00,11\}}X_2^{\{01,10\}} \vee X_1^{\{01,10\}}X_2^{\{00,11\}} \vee X_1^{\{00,01\}}X_2^{\{00,10\}} \quad (9)$$

Three columns are necessary in this assignment.

Therefore, when we assign the input variables as shown in Figure 5, the array is minimized.

Switching function.

x_1	x_2	x_3	x_4	f
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

MVITVOF and its expressions

The table below shows an example of a MVITVOF, where X_1 takes two-values, X_2 takes four-values, and X_3 takes three-values. Like ordinary two-valued logic functions, MVITVOFs can be represented by expressions. The function in the truth table is represented as a sum-of-products expression consisting of minterms:

$$F(X_1, X_2, X_3) = X_1^{\{0\}}X_2^{\{0\}}X_3^{\{1\}} \vee X_1^{\{0\}}X_2^{\{1\}}X_3^{\{1\}} \vee X^{\{0\}}{}_1X_2^{\{2\}}X_3^{\{2\}} \vee X_1^{\{1\}}X_2^{\{0\}}X_3^{\{0\}} \vee X_1^{\{1\}}X_2^{\{0\}}X_3^{\{1\}} \vee X_1^{\{1\}}X_2^{\{0\}}X_3^{\{2\}} \vee X_1^{\{1\}}X_2^{\{1\}}X_3^{\{0\}} \vee X_1^{\{1\}}X_2^{\{2\}}X_3^{\{0\}} \vee X_1^{\{1\}}X_2^{\{3\}}X_3^{\{1\}} \quad (10)$$

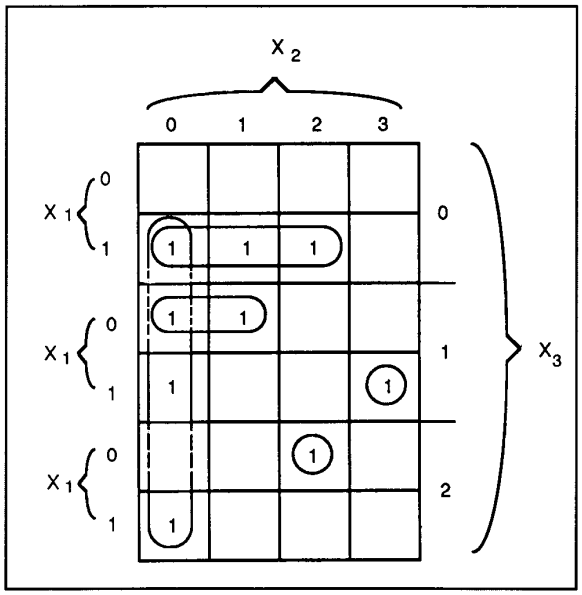
Equation 10 shows that F becomes one when $\{X_1 = 0, X_2 = 0,$ and $X_3 = 1\}$ or $\{X_1 = 0, X_2 = 1,$ and $X_3 = 1\}$ or . . . or $\{X_1 = 1, X_2 = 3,$ and $X_3 = 1\}$. Equation 10 is minimized by using the map for the truth table of MVITVOF (see figure at lower left):

$$F(X_1, X_2, X_3) = X_1^{\{0\}}X_2^{\{0,1\}}X_3^{\{1\}} \vee X_1^{\{0\}}X_2^{\{2\}}X_3^{\{2\}} \vee X_1^{\{1\}}X_2^{\{0\}}X_3^{\{0,1,2\}} \vee X_1^{\{1\}}X_2^{\{0,1,2\}}X_3^{\{0\}} \vee X_1^{\{1\}}X_2^{\{3\}}X_3^{\{1\}} \quad (11)$$

Equation 11 shows that F becomes one when $\{X_1 = 0$ and $(X_2 = 0$ or $X_2 = 1)$ and $X_3 = 1\}$ or $\{X_1 = 0, X_2 = 2,$ and $X_3 = 2\}$ or . . . or $\{X_1 = 1, X_2 = 3,$ and $X_3 = 1\}$.

Truth table of MVITVOF.

x_1	x_2	x_3	F	x_1	x_2	x_3	F
0	0	0	0	1	0	0	1
0	0	1	1	1	0	1	1
0	0	2	0	1	0	2	1
0	1	0	0	1	1	0	1
0	1	1	1	1	1	1	0
0	1	2	0	1	1	2	0
0	2	0	0	1	2	0	1
0	2	1	0	1	2	1	0
0	2	2	1	1	2	2	0
0	3	0	0	1	3	0	0
0	3	1	0	1	3	1	1
0	3	2	0	1	3	2	0

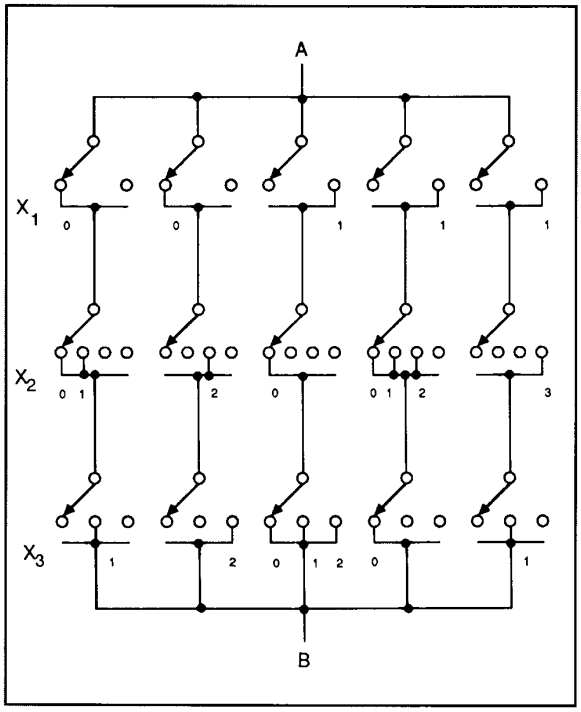


Map for table "Truth table of MVITVOF."

Contact networks with multipositional switches

Expressions for two-valued input logic functions represent contact networks with two-positional switches. In a similar way, expressions for MVITVOFs represent contact networks with multipositional switches.[6] For example, the contact network with multipositional switches shown in the figure below corresponds to Equation 11. In the top row of the network, two-positional switches correspond to X_1; in the second row, four-positional switches correspond to X_2; and in the bottom row, three-positional switches correspond to X_3. Note that the first column of the network realizes the first product of Equation 11. There is a path in the first column only if $\{(X_1 = 0)$ and $(X_2 = 0$ or $X_2 = 1)$ and $(X_3 = 1)\}$. Similarly, the second column realizes the second product, and so on. There is a path between terminals A and B when the value of the expression is one.

The number of products in the sum-of-products expression is equal to the number of columns of the contact network. Furthermore, a minimum sum-of-products expression corresponds to a contact network with a minimum number of columns.



Contact network with multipositional switches.

An example of multiple-output PLA

Consider the three-input, three-output function shown in the table at upper right, and the characteristic function of the multiple-output function shown in the table below it. Note that the variable X representing the outputs takes three values. The minimized expression for the table "Characteristic function" is

$$f(X_1, X_2, X_3, X_4) = X_1^{\{0\}}X_2^{\{0\}}X_3^{\{0,1\}}X_4^{\{1\}} \vee X_1^{\{0\}}X_2^{\{1\}}X_3^{\{0\}}X_4^{\{2\}} \vee X_1^{\{1\}}X_2^{\{0\}}X_3^{\{0\}}X_4^{\{0,1,2\}} \vee X_1^{\{1\}}X_2^{\{0,1\}}X_3^{\{0\}}X_4^{\{0\}} \vee X_1^{\{1\}}X_2^{\{0\}}X_3^{\{0,1\}}X_4^{\{0\}} \vee X_1^{\{1\}}X_2^{\{1\}}X_3^{\{1\}}X_4^{\{1\}} \quad (12)$$

Equation 12 shows the same permitted combinations as the characteristic function. Figure 6 (main text) shows a multiple output PLA realizing this expression. Each column including the OR part of the PLA corresponds to a product of the expression, so the number of columns of a PLA is equal to the number of the products in a sum-of-products expression for the characteristic function.

Multiple-output PLA with decoders

An expression of a MVITVOF represents a multiple-output PLA with decoders. For example, consider the realization of the three-input, three-output function (see table at upper right) by using a PLA with a one-bit decoder and a two-bit decoder. Let $X_1 = (x_2)$ and $X_2 = (x_2, x_3)$ denote the input variables, and X_3 denote the outputs. Note that X_1, X_2, and X_3 take two, four, and three values, respectively. The truth table of the characteristic function is the same as the truth table of MVITVOF, except that in X_2, 0 is replaced by 00, 1 by 01, 2 by 10, and 3 by 11. Therefore, the minimized expression obtained from Equation 11 is

$$F(X_1, X_2, X_3) = X_1^{\{0\}}X_2^{\{00,01\}}X_3^{\{1\}} \vee X_1^{\{0\}}X_2^{\{10\}}X_3^{\{2\}} \vee X_1^{\{1\}}X_2^{\{00\}}X_3^{\{0,1,2\}} \vee X_1^{\{1\}}X_2^{\{00,01,10\}}X_3^{\{0\}} \vee X_1^{\{1\}}X_2^{\{11\}}X_3^{\{1\}} \quad (13)$$

The figure at right shows the PLA with a one-bit and a two-bit decoder.

Minimization of expressions for MVITVOF

Minimization algorithms for expressions of MVITVOFs[2,4,6,7] are similar to the Quine-McCluskey method for two-valued input logic functions. The major difference is that the input variables may take on more than two values. MVITVOFs with "don't cares" can be treated similarly to two-valued cases. For multiple-valued functions, the number of prime implicants is much larger than that of the two-valued input functions. Therefore, minimization of expressions for MVITVOFs is more time-consuming than for two-valued cases.

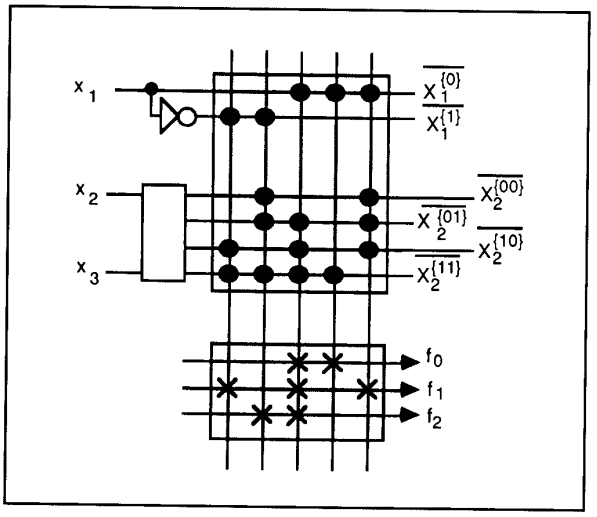
It is impractical to find absolute minimum solutions for large problems because computation time and storage requirements for the minimization programs increase exponentially with the number of prime implicants. MINI,[8] MINI-II,[3] and ESPRESSO-MV[4] are heuristic programs that obtain near-minimum sum-of-products expressions for MVITVOFs. In these programs, expand, reduce, and other operations iteratively simplify the expressions. Because the expand is the most time-consuming operation in the heuristic programs, a complement of a MVITVOF is used to perform the expand operation efficiently.[9]

Three-input, three-output function.

x_1	x_2	x_3	f_0	f_1	f_2
0	0	0	0	1	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	0	0	0
1	0	0	1	1	1
1	0	1	1	0	0
1	1	0	1	0	0
1	1	1	0	1	0

Characteristic function.

X_1	X_2	X_3	X_4	F
0	0	0	0	0
0	0	0	1	1
0	0	0	2	0
0	0	1	0	0
0	0	1	1	1
0	0	1	2	0
0	1	0	0	0
0	1	0	1	0
0	1	0	2	1
0	1	1	0	0
0	1	1	1	0
0	1	1	2	0
1	0	0	0	1
1	0	0	1	1
1	0	0	2	1
1	0	1	0	1
1	0	1	1	0
1	0	1	2	0
1	1	0	0	1
1	1	0	1	0
1	1	0	2	0
1	1	1	0	0
1	1	1	1	1
1	1	1	2	0



PLA with decoders for table "Three-input, three-output function."

(Continued from p. 74)
columns for various arithmetic PLAs. I have optimized these PLAs by exhaustive and heuristic methods. I also designed 16 control circuits for microprocessors. When we optimize the assignment of the input variables, the array size of PLAs with two-bit decoders is, on the average, 20 percent smaller than that of standard PLAs.[3]

We can also consider a PLA with multibit decoders, where each decoder may

Multiple-valued PLAs and encoding problems

A four-valued PLA adder[11] (see figure below) consists of literal generators, the AND array, the OR array, and programmable output encoders. In this PLA, external input and output lines take four different physical states, say 0, 1, 2, and 3 volts. However, the body of the PLA is two-valued and takes only 0 and 3 volts.

The literal generator (see table) converts a four-valued signal into two-valued signals. The CMOS realization of the literal generator (see facing figure) uses inverters having three different thresholds (see table).

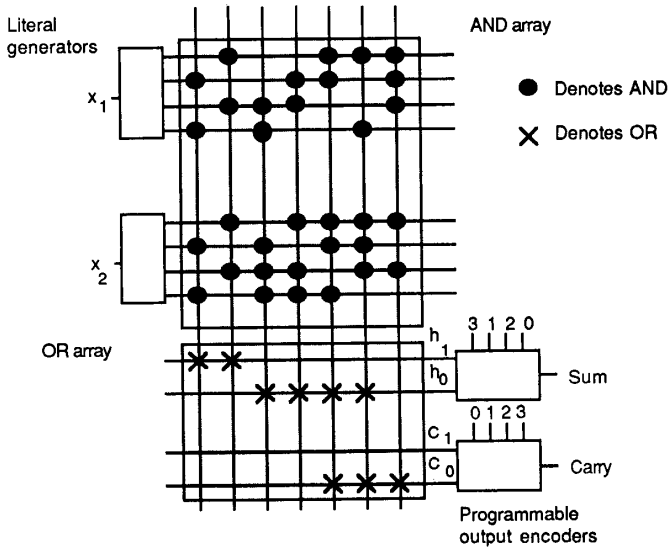
The programmable output encoders convert two-valued signals into four-valued ones, as shown in the tables "Output encoding for Sum" and "Output encoding for Carry." Note that the encodings for Sum and Carry are different. In a CMOS realization of a programmable output encoder (see the facing figure) four different voltage sources (0, 1, 2, and 3 volts) are connected to C_0, C_1, C_2, and C_3. We can permute the voltage sources to have different output encodings. Note that there are 4! = 24 different output encodings. In the figure below, the output encoding is chosen to minimize the array size. If the encoding for Sum were the same as that for Carry, the PLA would require nine columns.

Literal generator.

Four-Valued Signal	Two-Valued Signals			
0	0	3	3	3
1	3	0	3	3
2	3	3	0	3
3	3	3	3	0

Inverters with different thresholds.

Input	Outputs			
	▷○	▷○	▷○	▷○
0	3	3	3	3
1	0	3	3	—
2	0	0	3	—
3	0	0	0	0



Four-valued PLA adder.

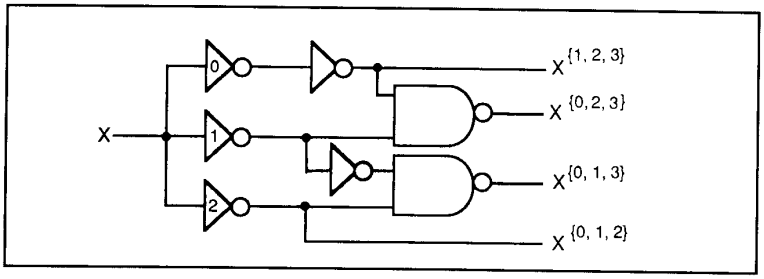
Output encoding for Sum

Four-Valued Signal	Two-Valued Signals	
0	3	3
1	0	3
2	3	0
3	0	0

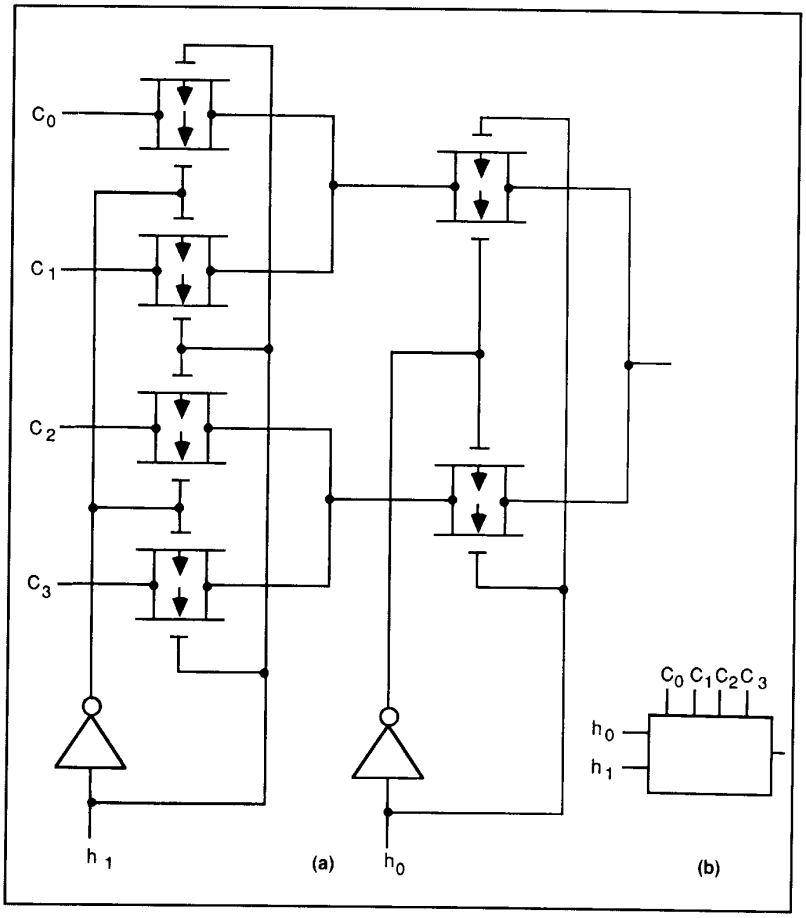
Output encoding for Carry.

Four-Valued Signal	Two-Valued Signals	
0	0	0
1	0	3
2	3	0
3	3	3

have a different number of inputs, and at least one decoder has more than two inputs. For adders, multipliers, and some control circuits, the array size of such PLAs is smaller than that of PLAs with two-bit decoders. In a PLA with multibit decoders, the number of literal lines is greater than that of a PLA with two-bit decoders. Thus, the design of such PLAs is more complex than that of PLAs with two-bit decoders. (See the sidebar "Example of PLAs with two-bit decoders.")



CMOS realization of a literal generator.



(a) CMOS realization of a programmable output encoder. (b) Logic symbol.

Multiple-output PLA

PLAs usually have multiple outputs. In designing multiple-output PLAs, independent minimization of each output does not always produce the full minimization. We have to consider all outputs at the same time. The literature of logical design often shows an extension of the Quine-McCluskey method, which uses multiple-output prime implicants. However, this method is complicated when treating the output part. A more elegant method is to use a characteristic function of the multiple-output function.[3] The characteristic function contains only and all the permitted combinations of inputs and outputs. In this function, we use an augmented variable X_{n+1} that shows the output part $(f_0, f_1, \ldots, f_{m-1})$ in addition to the input variables $x_1, x_2, \ldots, x_n$. The minimization algorithm for a characteristic function is more uniform and in many cases appears to be faster than the one that treats each output separately.[4,5] (See the sidebar "MVITVOF and its expressions.")

Output phase optimization

When realizing a multiple-output function by a PLA, we often have the option to realize f_i or $\bar{f}_i$ for each output. For each PLA output, we need a buffer either logically inverting or noninverting. We often can decrease the PLA's array size by appropriately selecting the phase (polarity) of each output.

For example, when the complement of f_0 is realized and an inverter is inserted into the output in Figure 6, we have a smaller PLA, as shown in Figure 7. In this case, the complementing of the first output reduces the array size.

The optimum output phase minimizes the total number of columns in the PLA. For an m-output function, there are 2^m possibilities of choosing the output phase. The exhaustive method for output phase optimization takes 2^m minimizations. This is impractical for large problems. I have developed a heuristic method that obtains good solutions in a reasonable computation time.[3] The heuristic algorithm uses the expressions for MVITVOFs. Table 2 compares the number of

PLA columns with and without consideration of the output phase. Intel used output phase optimization technique to reduce some of the PLAs used by its 80386 microprocessor.[10]

Encoding Problems

We can often reduce the size of PLAs by changing the encoding of the inputs or outputs.[4] Consider as an example of input encoding a PLA used for a processor's instruction decoder. In this case, we can choose the codes for the instruction to minimize the size of the PLA. As an example of output encoding, consider the generation of the control signals for a data path. In this case, we have to encode each action in the data path uniquely, but there is no reason to favor any particular encoding. Another example of output encoding is in a multiple-valued PLA.[11]

We can use MVITVOFs to solve the input encoding problem. An approximation to the state-assignment problem for finite-state machines can be solved by using MVITVOFs.[12] The output encoding problem is more difficult, but current techniques for solving the problem also use MVITVOFs. (See the sidebar "Multiple-valued PLAs and encoding problems.")

We can reduce industrial PLAs by 20 to 30 percent by using these new optimization techniques for multiple-output PLAs with decoders. For large PLAs, the use of two-bit decoders usually reduces the actual chip area even if we account for the decoder area.

Minimization of expressions for MVITVOFs is indispensable in these PLA optimizations. Modern microprocessors use PLAs extensively in their control parts, so high-quality minimizers for MVITVOFs, such as MINI-II and Espresso-MV, have become very important. Continuing efforts are being made to improve these optimization algorithms to work faster for larger PLAs.

Logic minimization is a basic tool for VLSI logic design. Encoding problems, optimization of multilevel combinational networks, and decomposition of a large PLA into small linked PLAs are future applications of MVITVOFs. □

Acknowledgments

I wish to thank Jon T. Butler and all seven referees for refining this article, and S.D. Hamilton for reading the preliminary versions. This work is partially supported by a Grant in Aid for Scientific Research from the Ministry of Education, Science, and Culture of Japan, and a research grant from the Mazda Foundation.

References

1. H. Fleisher and L.I. Maissel, "An Introduction to Array Logic," *IBM J. Research and Development*, Vol. 19, No. 2, March 1976, pp. 98-109.
2. T. Sasao, "Multiple-Valued Decomposition of Generalized Boolean Functions and the Complexity of Programmable Logic Arrays," *IEEE Trans. Computers*, Sept. 1981, pp. 635-643.
3. T. Sasao, "Input Variable Assignment and Output Phase Optimization of PLAs," *IEEE Trans. Computers*, Oct. 1984, pp. 879-894.
4. R.L. Rudell and A. Sangiovanni-Vincentelli, "Multivalued Minimization for PLA Optimization," *IEEE Trans. CAD*, Sept. 1987, pp. 727-750, and also in *Proc. ISMVL-87*, May 1987, pp. 198-208.
5. R.K. Brayton et al., *Logic Minimization Algorithms for VLSI Synthesis*, Kluwer Academic Publishers, 1984.
6. P.L. Tison, "An Algebra for Logic Systems: Switching Circuit Application," *IEEE Trans. Computers*, Mar. 1971, pp. 339-351.
7. M. Davio, J.-P. Deschamps, and A. Thayse, *Discrete and Switching Functions*, Georgi Publishing Co., St.-Saphorin, Switzerland, 1978.
8. S.J. Hong, R.G. Cain, and D.L. Ostapko, "MINI: A Heuristic Approach for Logic Minimization," *IBM J. Research and Development*, Sept. 1974, pp. 443-458.
9. T. Sasao, "An Algorithm to Derive the Complement of a Binary Function With Multiple-Valued Inputs," *IEEE Trans. Computers*, Feb. 1985, pp. 131-140.
10. P.P. Gelsinger, "Design and Test of the 80386," *IEEE Design and Test*, June 1987, pp. 42-50.
11. T. Sasao, "On the Optimal Design of Multiple-Valued PLAs," *ISMVL-86*, May 1986, pp. 214-223, and also to be published in *IEEE Trans. Computers*.
12. G. De Micheli, R.K. Brayton, and A. Sangiovanni-Vincentelli, "Optimal State Assignment for Finite-State Machines," *IEEE Trans. CAD*, 1985, pp. 269-285.

Tsutomu Sasao is an associate professor of computer science and electronics at Kyushu Institute of Technology, Japan. His current interests include PLA design, automatic logic design systems, and complexity theory of logic circuits.

Sasao was the Asia Program Chairman of the 1984 International Symposium on Multiple-Valued Logic (ISMVL-84). He received the NIWA Memorial Award in 1979, and the Distinguished Contribution Award for ISMVL-86 in 1987.

He received his BS, MS, and PhD in electronic engineering from Osaka University in 1972, 1974, and 1977, respectively.