Adder and Comparator Synthesis with Exclusive-OR Transform of Inputs

James Jacob and P.S. Sivakumar
Indian Institute of Science
Bangalore 560 012, India
james@ece.iisc.ernet.in

Vishwani D. Agrawal
Bell Labs, Lucent Technologies
Murray Hill, NJ 07974, USA
va@research.bell-labs.com

Abstract

An exclusive-OR transform of input variables significantly reduces the size of the PLA implementation for adder and comparator circuits. For n bit adder circuits, the size of PLA for transformed functions is $O(n^2)$. In comparison, when the complete truth-table of an adder is minimized, the PLA size will be $O(2^{n+2})$. Similarly, for an n bit comparator, the size of the PLA is reduced from $O(2^{n+1})$ to $O(n)$. These implementations require additional transform logic of complexity $O(n)$, consisting of exclusive-OR gates.

1 Introduction

For arithmetic functions, such as adders and comparators, the size of the minimized two-level form grows exponentially with the number of inputs [1]. We show that when an exclusive-OR transform is applied to inputs, the number of product terms in the two-level sum-of-products drastically reduces. The transform requires only n exclusive-OR gates for an n bit parallel adder or comparator [2]. This technique makes the optimization and implementation of adders and comparators possible either as a programmable logic array (PLA) or as a programmable logic device (PLD).

2 Adder PLAs

Consider an n-bit parallel adder for two n-bit words $x_n x_{n-1} \ldots x_2 x_1$ and $y_n y_{n-1} \ldots y_2 y_1$. The $n+1$ bit output is $f_{n+1} f_n \ldots f_2 f_1$. Consider an input transform T that transforms only one half of the variable positions and retains other half of the variable positions, i.e., the transform function (applied to each minterm) is: $T(x_n x_{n-1} \ldots x_2 x_1 y_n y_{n-1} \ldots y_2 y_1) = X_n X_{n-1} \ldots X_2 X_1 y_n y_{n-1} \ldots y_2 y_1$, where $X_i = x_i \oplus y_i$ for $1 \leq i \leq n$. Similar transforms, in general, may be devised using spectral methods [3].

The transform based realization is shown in Figure 1. Obtaining f^T from the minterm form becomes impractical for large values of n as the truth table has 2^{2n} entries for an n bit adder or comparator. However, we observe a recursive regularity in the pattern of the minimized cubes of the transformed functions of sizes 2 to 8, whose truth tables can be easily handled by a logic minimization program. Taking advantage of this regularity, we directly obtain the minimized cubes of the transformed adders and comparators of any bit-width n without using the logic minimizer [2].

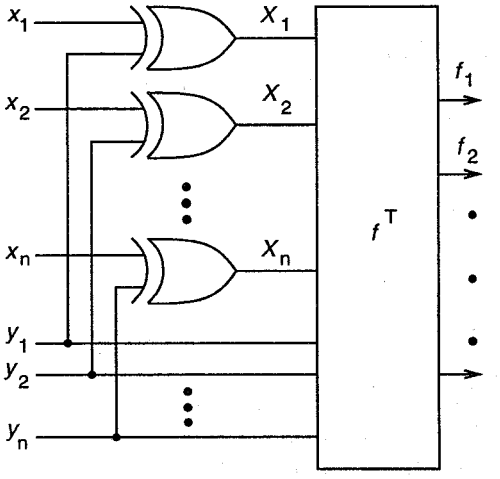

Figure 1: Input transformed n-bit adder/comparator

The overall cost of the transform is n XOR gates for an n-bit parallel adder. It also has a delay penalty of 1 unit. Experimental results for this transform are given in Table 1. Columns 2 and 3 give the number of inputs and outputs for the adders. Output functions of the original and transformed adders were minimized using Espresso in the default mode [4]. The number of minimized cubes are shown in columns 4 and 5. For $n > 8$, since it was not practical to minimize the truth tables by Espresso, the number of cubes are given only for the transformed PLA, as obtained by our method that directly generates the minimized cubes [2]. The saving in the PLA area, directly proportional to the reduction

0-8186-7755-4/96 $05.00 © 1996 IEEE

Table 1: Results of input transform on adders

Circuit name	No. of inputs	No. of outputs	Minimized Cubes		Red. in Product Terms %
			Orig. PLA	Trans. PLA	
adder2	4	3	11	6	45.5
adder3	6	4	31	12	61.2
adder4	8	5	75	20	73.3
adder5	10	6	167	30	82.0
adder6	12	7	355	42	88.2
adder7	14	8	735	56	92.4
adder8	16	9	1499	72	95.2
adder16	32	17	-	272	>99.9
adder32	64	33	-	1056	>99.9
adder64	128	65	-	4160	>99.9

Table 2: Results of input transform on comparators

Circuit name	No. of inputs	No. of outputs	Minimized Cubes		Red. in Product Terms %
			Orig. PLA	Trans. PLA	
comp2	4	2	7	3	57.1
comp3	6	2	15	4	73.3
comp4	8	2	31	5	83.8
comp5	10	2	63	6	90.5
comp6	12	2	127	7	94.5
comp7	14	2	255	8	96.9
comp8	16	2	511	9	98.2
comp16	32	2	-	17	>99.9
comp32	64	2	-	33	>99.9
comp64	128	2	-	65	>99.9

in number of product terms is given in the last column. Notice that for an n bit adder the number of original PLA product terms increases as an exponential function of n. Empirically, the PLA size is $O(2^{n+2})$. With the input transform, the number of product terms reduces to $n^2 + n$ or $O(n^2)$. The cost of the transform, consisting of n exclusive-OR gates is $O(n)$. Clearly, the advantage of the input transform is enormous as the size of the adder increases.

We should point out that our design differs from the three-stage carry look-ahead (CLA) adder [5]. In that design, the first stage produces the *generate* ($G_i = x_i.y_i$) and *propagate* ($P_i = x_i \oplus y_i$) signals. In the second stage, internal carry signals C_i are obtained as sum of product expressions involving G_i and P_i variables. In the third stage, the sum bits S_i are obtained as exclusive-OR of the corresponding C_i and P_i signals. As shown in Figure 1, ours is a two-stage design. The S_i signals are generated by sum of product expressions of the X_i, which are the same as P_i, and y_i variables. We save on the AND gates that generate G_i signals and the output exclusive-OR gates at the cost of an increased size of the PLA that is twice that of the PLA in the CLA design.

3 Comparator PLAs

Consider an n-bit parallel comparator that compares two n-bit words $x_1x_2 \ldots x_n$ and $y_1y_2 \ldots y_n$ giving the two output bits G and E :

G	E	*Meaning*
0	0	$x_1x_2 \ldots x_n$ is less than $y_1y_2 \ldots y_n$
1	0	$x_1x_2 \ldots x_n$ is greater than $y_1y_2 \ldots y_n$
0	1	$x_1x_2 \ldots x_n$ is equal to $y_1y_2 \ldots y_n$
1	1	Don't Care (combination never occurs)

The results of Table 2 are for comparators designed with the input transform of Section 2.

The comparator PLAs were minimized using Espresso and the numbers of cubes in the original and transformed comparators are given in columns 4 and 5, respectively. For $n > 8$, we have given the number of minimized cubes in the transformed PLA obtained by our direct method [2]. The last column lists the savings in area due to the reduction in number of product terms. The input transform brings down the cost of the PLA in proportion to $n + 1$ product terms for an n-bit parallel comparator. Once again we see that the number of cubes for the original PLA of an n-bit comparator is $O(2^{n+1})$, which reduces to $O(n)$ with the input transform, at a cost of an $O(n)$ overhead for the transform exclusive-OR gates.

4 Conclusion

The novel idea of input transforms is effective in the synthesis of functions such as adders and comparators, which are known to be difficult functions for two-level implementation. The cost of the transform is small relative to the reduction obtained in the PLA size.

References

[1] S. Devadas, A. Ghosh and K. Keutzer, *Logic Synthesis*, McGraw-Hill, Inc., New York, 1994.

[2] P. S. Sivakumar. *Boolean Transforms in Logic Synthesis*, Master's Project Report, E.C.E Department, Indian Institute of Science, Bangalore, India, June 1994.

[3] J. P. Hansen and M. Sekine, "Synthesis by Spectral Translation Using Boolean Decision Diagrams", *in Proc. 33rd Design Automation Conf.*, June 1996, pp. 248-253.

[4] R. K. Brayton, G. D. Hachtel, T. McMullen and A. L. Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis*, Kluwer Academic Publishers, Boston, 1984.

[5] F. J. Hill and G. R. Peterson, *Digital Systems Hardware Organization and Design*, Second Edition, John Wiley & Sons, New York, 1978.