SYNTHESIS OF LPGA FROM LINEAR DECISION DIAGRAMS

P. DZIURZAŃSKI
Department of Computer Science, Technical University of Szczecin, Poland
dziurzan@jota.ps.pl

Abstract

The goal of this paper is to present a new approach for synthesis of channeled gate array circuit, including LPGA (ang. *laser programmable gate array*), from word-level linear decision diagrams.Next it will be shown that exchanging of NAND circuit with a 1-bit adder in a structure similar to popular circuits form Chip Express QYH500 serie, can decrease an amount of required cells and levels, which can result with lower dissipated power and circuit delay.

I. INTRODUCTION

Laser programmable gate array (LPGA) circuits are better than field programmable gate array (FPGA) with respect to capacity and speed. Advantages of their applications there are short time-to-market time, flexibility and low production cost.

One of more interesting solution in a LPGA field is presumably a technology developed by Chip Express and Lucent Technologies, which uses in LPGA circuits an architecture known from channeled gate array.

In Chip Express QYH500 channels with NAND gates are used. NAND gate has two important property, namely

- function NAND is a full set, i.e. each boolean function can be represented with circuit including NAND gates only,
- NAND gate has very simple implementation in CMOS and TTL technologies.

In sequel we will show that structure similar to QYH500 circuits, including full 1-bit adders instead of NAND gate, can lead to decreasing of the number of needed cells. Although 1-bit adder has a bigger area than NAND gate, smaller number of cells has a few assets, such as smaller circuit delay, lower power dissipation, better reliability etc.

The proposed approach of synthesis from linear arithmetic decision diagrams is based on an idea of linearization of decision trees presented in [4], [5] and [6]. A usage of word-level Binary Moment Diagrams (BMDs) were firstly used in [2], and next developed in [3].

The proposed approach of building of linear decision diagrams were implemented in *linDD* program, and parameters of synthesized circuits from standard benchmark libraries were estimated with a package for multilevel synthesis SIS [7].

The paper is organized as follows. In the chapters II and III the essential theoretical basics are presented and an algorithm for a building of linear decision diagrams, respectively. In the next chapter IV the synthesis from a set of linear decision diagrams is outlined, in chapter V we present our experiments, which proved the effectiveness of proposed method. The last chapter VI concludes the paper.

II. BASICS OF THE LINEARIZATION OF BOOLEAN FUNCTION.

The proposed approach to synthesis of LPGA circuit is based on *word-level* representation of switching functions. In this paper we briefly describe possibilities and limitations of this representation.

An arbitrary switching function can be described by a corresponding arithmetical polynomial [**?**] (AR), for instance

$$\begin{aligned}
\overline{x}_1 &= 1 - x_1, \\
x_1 \vee x_2 &= x_1 + x_2 - x_1 x_2, \\
x_1 \wedge x_2 &= x_1 x_2, \\
x_1 \oplus y_1 &= x_1 + y_1 - 2x_1 y_1.
\end{aligned}$$

TABLE I

LAR EXPRESSION CORRESPONDING TO STANDARD LOGIC GATES

Gate	LAR	Gate	LAR
OR	$\Xi_2^2\{1+x+y\}$	AND	$\Xi_2^2\{x+y\}$
EXOR	$\Xi_2^1\{x+y\}$	EXNOR	$\Xi_2^1\{1+x+y\}$
NOT	$\Xi_2^1\{1+x\}$		

Linearization it is a transformation of non-linear, in a general case, AR expression to linear AR (LAR) in form

$$d_0 + d_1x_1 + \cdots + d_nx_n.$$

In the other words, the task of linearization is to find a linear representation, which includes no more than $n+1$ non-zero coefficients.

Our goal is to represent an arbitrary circuit by a set of linear BMD (LBMD). In proposed method circuits levels are analysed independently, whereas in other approach for linearization, for example in [5], authors focused on a description of the whole network.

In proposed method we take advantage of *masking operator* and word-level decision diagrams with a *positive Davio* (pD_A) expansion for representing the given, i-th level of the circuit. As a result, the whole circuit is described by the set of linear decision diagrams.

Another advantages of proposed method is the small amount of memory, which LBMDs require, and linear time of decision diagram building.

A proposed approach has two important properties:

- with a set of LBMDs it is possible to describe an arbitrary multi-level circuit,
- a number of nodes in a LBMD set is equal to $O(gate)$, where $gate$ is the number of gates in the circuit.

We use *masking* technique, which allows us to chose one bit of a *word-level* representation.

Definition 1: Masking operator $\Xi_m^j\{f\}$ extracts the j-th output, $j = 1, \ldots, m$, from the m-output function f.

Example 1: A single-output function

$$x_1 \oplus x_2 = x_1 + x_2 - 2x_1x_2$$

can be linearized by expanding to a 2-input function with outputs

$$\begin{aligned} f_1 &= x_1 \oplus x_2, \\ f_2 &= x_1x_2. \end{aligned}$$

We could extract the given function f from its *word-level* representation with a masking operator

$$\Xi_2^1\{x_1 + x_2\}.$$

This operator extracts a position that corresponds to f_1 function.

In Table I we show LAR expression for typical logic gates. Based on these LAR expressions, it is possible to describe a circuit level in following way. Each LAR described in Table I needs two bit for keeping a value of its output. Thus by multiplying the of a LAR corresponding with k-th gate in the level by a weight $2^{2(k-1)}$, we obtain k disjointed functions. After adding of these LAR, we obtain a LAR for the whole level.

Example 2: Let us consider a circuit level IN, which consists of three gates

$$\begin{aligned} y_1 &= x_1 \vee x_2, \\ y_2 &= x_2 \oplus x_3, \\ y_3 &= x_1x_3. \end{aligned}$$

Based on Table I we could write polynomials for these gates

$$\begin{aligned} LAR(y_1) &= x_1 + x_2 + 1, \\ LAR(y_2) &= x_2 + x_3, \\ LAR(y_3) &= x_1 + x_3. \end{aligned}$$

By multiplying them by appropriate weights, we obtain a LAR for the whole level

$$\begin{aligned} LAR(IN) &= 2^0 LAR(y_1) + 2^2 LAR(y_2) + 2^4 LAR(y_3) \\ &= 2^0(x_1 + x_2 + 1) + 2^2(x_2 + x_3) + 2^4(x_1 + x_3), \end{aligned}$$

where

$$\begin{aligned} y_1 &= \Xi_6^2\{LAR(IN)\}, \\ y_2 &= \Xi_6^4\{LAR(IN)\}, \\ y_3 &= \Xi_6^6\{LAR(IN)\}. \end{aligned}$$

III. Circuit representing by a set of linear decision diagrams

Since functions from Table I form system funkcjonalnie pełny, there is a possibility of describing an arbitrary logic circuit by a set of LARs, doing a linear combination of LAR corresponding with single gates, with a masking operator. Obtained LAR could be depicted in LBMD form.

Definition 2: LBMD is a decision diagram, whose nodes realize arithmetic positive Davio expansion

$$f = 1 \cdot f_0 + x_i(-f_0 + f_1),$$

where $f_0 = f_{x_i=0}$, $f_1 = f_{x_i=1}$, for each variable of the function f.

In Fig. 1 we show LBMD representation for selected gates form Table I.

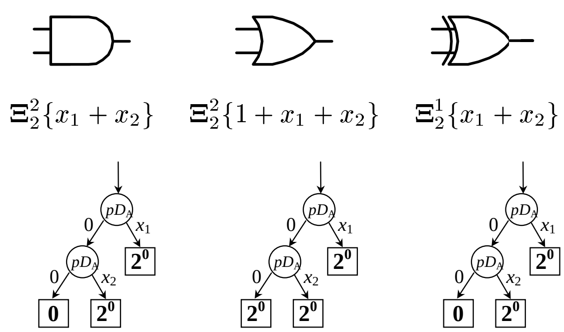

Fig. 1. LBMD for selected 2-input gates

Example 3: A circuit including 4-levels IN_i, $i = 1, 2, 3, 4$, depicted in Fig. 3, can be described by four LARs

$$\begin{aligned} IN_1 &= 2^0 x_2 + 2^2 x_3 + 2^0 + 2^2 \\ IN_2 &= (2^0 + 2^4)x_1 + 2^2 x_2 + 2^0 \overline{x}_2 + (2^2 + 2^4)\overline{x}_3, \\ IN_3 &= 2^0 y_1 + 2^0 y_2 + 2^0, \\ IN_4 &= 2^0 z_1 + 2^0 z_2. \end{aligned}$$

Let us calculate a value of function f for value $x_1 x_2 x_3 = 101$. Since $IN_1 = 2^2 + 2^0 + 2^2 = 9 = 1001_2$,

$$\begin{aligned} \overline{x}_2 &= \Xi_4^1\{IN_1\} = \Xi_4^1\{100\mathbf{1}\} = 1, \\ \overline{x}_3 &= \Xi_4^3\{IN_1\} = \Xi_4^1\{1\mathbf{0}01\} = 0, \end{aligned}$$

```
linDD algorithm
Input: a logic circuit
Oputput: a set of LBMD describing an input circuit

 for(∀ level i in the circuit)
 {
   Create LBMD for the i-th level;
   for(∀ gate g in the i-th level)
       for(∀ input in in the g-th gate)
           if(input in exists
              in LBMD for i-th level) then
              add a new node with a variable
              corresponding with input in;
           add a new value for existing terminal node
           corresponding with input in;

 }
```

Fig. 2. Algorithm for representing r-level circuit with a set of linear decision diagrams

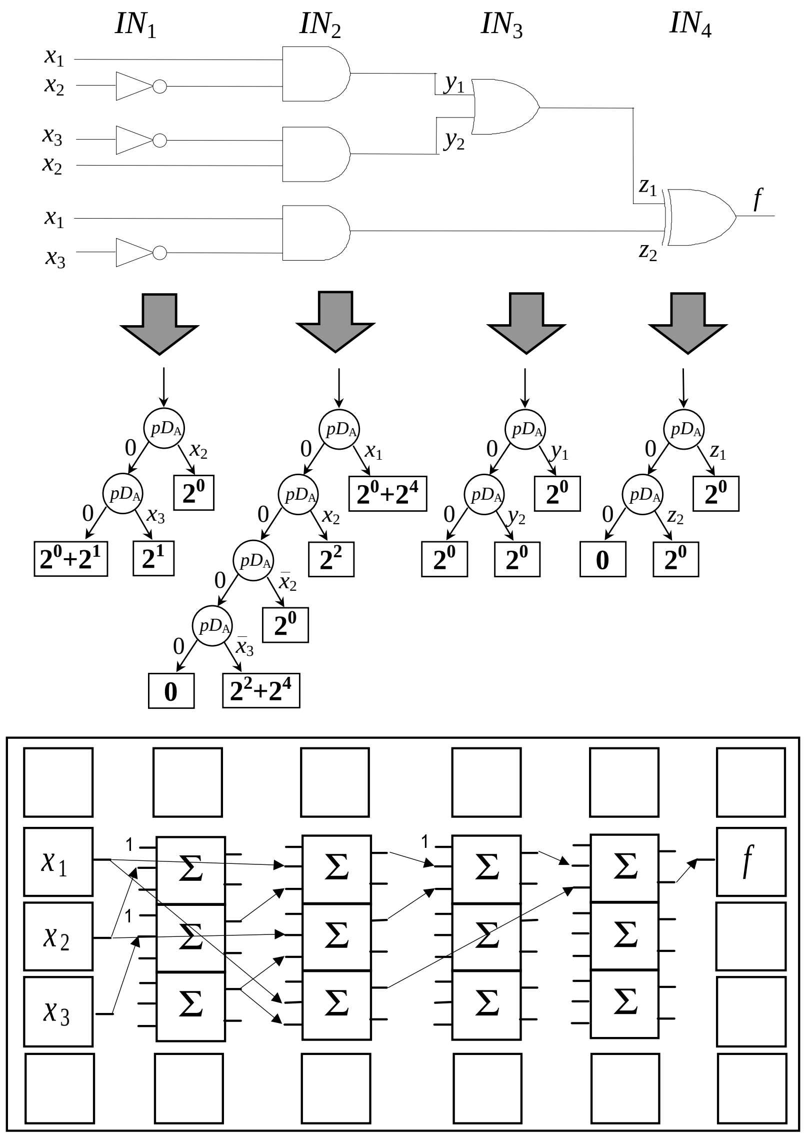

Fig. 3. A circuit, corresponding set of LBMD and CGA structure

$IN_2 = 2^0 + 2^4 + 2^0 = 18 = 010010_2$,

$$\begin{aligned} y_1 &= \Xi_6^2\{IN_2\} = \Xi_6^2\{0100\mathbf{1}0\} = 1, \\ y_2 &= \Xi_6^4\{IN_2\} = \Xi_6^5\{01\mathbf{0}010\} = 0, \\ z_2 &= \Xi_6^6\{IN_2\} = \Xi_6^6\{\mathbf{0}10010\} = 0, \end{aligned}$$

For the third level, $IN_3 = 2^0 + 2^0 = 2 = 10_2$, thus

$$z_1 = \Xi_2^2\{IN_3\} = \Xi_2^2\{\mathbf{1}0\} = 1,$$

Finally, $IN_4 = 2^0 = 1 = 01_2$ i

$$f = \Xi_2^1\{IN_4\} = \Xi_2^1\{0\mathbf{1}\} = 1.$$

IV. Logic circuit synthesis from a set of linear decision diagrams

Channeled gate array (CGA) is a semi-custom ASIC circuit, in which gates are grouped in rows, among which tere are routing channels. An example of CGA is presented in Fig. 4. Boundary squares are I/O blocks, whereas vertical stripes represent rows of gates (cells).

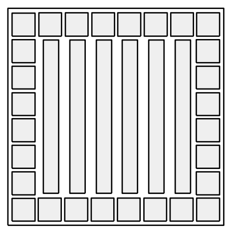

Fig. 4. A structure of channeled gate array

$MapLBMD\_CGA$ ***algorithm***
Input: *a set of LBMD*
Output: *netlist of CGA*

for(*∀ LBMD i in the set)*
{
for(*∀ output g in the i-th diagram)*
{
for(*∀ variable in of a function with output g)*
connect an empty input of g-th cell from the i-th row with appropriate output of an appropriate cells of previous rows;
if(*in an expression there is a coefficient with a weight 2^{2g-1})* **then**
connect output $carry_{in}$ with logic "1";
connect remaining outputs with logic "0";
}
}

Fig. 5. An algorithm of mapping of a linear diagrams set into channeled gate array structure

A mapping of a circuit given by a linear diagrams set into CGA maps one LBMD into one CGA row, composed of adders. Appropriate CGA cells are connected with classical routing algorithms, described for example in [8]. A sketch of proposed approach is outlined in Fig. 5.

Example 4: (Continuation of example 3) As the circuit consists of four LBMDs, it requires four CGA cells. A structure of CGA connections which realizes the function f is depicted in Fig. 3.

V. Experimental results

In order to verify proposed approach, it was tested on a set of standard benchmarks ISCAS'85[1]. In our research we use Pentium III 650 MHz with 256 MB of RAM memory.

TABLE II

Comparison of number of nodes in ROBDDs, K*BMDs and proposed LBMDs, and nubmer of cells in synthesis into channeled gate array with NANDs and 1-bit Adders

TEST		ROBDD	K*BMD	LBMD			Cells	
	I/O	#	#	L	#	Time [s]	NAND	Adders
c432	36/7	1064	1209[2]	17	336	0.00	–	–
c499	41/32	25866	29561[2]	11	365	0.01	675	435
c880	60/26	4053	4048[2]	24	605	0.01	557	381
c1355	41/32	29561	29651[2]	24	993	0.04	–	–
c1908	33/25	5526	5944[2]	40	1464	0.07	1004	655
c2670	233/140	1850	3939[2]	32	2026	0.14	1406	1011
c3540	50/22	23828	–	47	2760	0.22	2101	1503
c5315	178/123	1719	2504[2]	49	4156	0.47	3322	2378
c6288	32/32	–	–	124	4318	0.23	–	–
c7552	207/108	2213	–	64	4627	0.49	–	–
Total		>95680	>76856		21650		9071	6373

In the Table II we have shown the number of nodes into nonlinear bit-level ROBDD (3rd column) [1], nonlinear word-level K*BMD (4th column) [3], and proposed word-level linear decision diagrams (6th column), for functions from ISCAS'85 benchmark sets. From these types, only linear decision diagrams was able to represent all the benchmarks. Moreover, in comparison with typical ROBDD and K*BMD, number of nodes in proposed linear decision diagrams is smaller in 80%.

Synthesis of circuits were performed with SIS [7] package. This system was able to synthesis 8 from 10 benchmarks, into structure similar to Chip Express QYH500 (8th column) and proposed channeled gate array with adders (9th column). The results of experiment shown that usage of adders instead of NAND gates save about 30% of cells.

VI. Conclusion

In the paper it was presented synthesis of ASIC channeled gate array circuit from linear decision diagrams LBMDs. Experiment proved, that this type of synthesis is more optimal in comparison with popular, nonlinear word-level and bit-level decision diagrams. The difference is caused by smaller number of nodes - its complexity can be estimated as $O(n)$ with respect to memory and time of building. Among analized types of DDs, only proposed LBMDs allow to represent all circuit from ISCAS'89 set.

Experiment proved, that channeled gate array with adders can include less cells (about 30%) that widely used channeled gate array with NAND cells.

References

[1] R. Bryant, Graph - based algorithm for Boolean function manipulation, *IEEE Trans. on Computers*, 1986, vol. C-35, no. 8, pp. 667-691

[2] R. Bryant, Y. Chen, Verification of Arithmetic Functions Using Binary Moment Diagrams, *Proc. Design Automation Conf.*, 1995, pp. 535-541.

[3] R. Drechsler, B. Becker, S. Ruppertz, The K*BMD: A Verification Data Structure, *IEEE Design and Test of Computers*, 1997, vol. 14, no. 2, pp.51-59

[4] V. Malyugin, Realization of Boolean Function's Corteges by Means of Linear Arithmetical Polynomial, *Automation and Remote Control (USA)*, 1984, vol.45, no. 2, Pt. 1, pp. 239-245

[5] S. Muroga, T. Ibaraki, Design of Optimal Switching Networks by Integer Programming, *IEEE Trans. on Computers*, 1972, vol. C-21, no. 6, pp. 573-582

[6] M. Nakajima, M. Kameyama, Design of Highly Parallel Linear Digital System for ULSI Processors, *IEICE Trans*, 1993, vol. 7, no. E76-C, pp. 1119-1125.

[7] SIS Release 1.2, *UC Berkeley Soft. Distr*, http:/www.cbl.ncsu.edu/www/, 1994.

[8] M. J. S. Smith, Application-Specific Integrated Circuits, *Addison-Wesley*, 1997.

[1] ISCAS'85 benchmarks set is available at *www.cbl.ncsu.edu/CBL_Docs/iscas85.html*.