DBpia

A Study on the Exclusive-OR-based Technology Mapping Method in FPGA

Seok-Bum Ko ***Regular Member***

ABSTRACT

In this paper, we propose an AND/XOR-based technology mapping method for field programmable gate arrays (FPGAs). Due to the fixed size of the programmable blocks in an FPGA, decomposing a circuit into sub-circuits with appropriate number of inputs can achieve excellent implementation efficiency. Specifically, the proposed technology mapping method is based on Davio expansion theorem to decompose a given Boolean circuit. The AND/XOR nature of the proposed method allows it to operate on XOR intensive circuits, such as error detecting/correcting, data encryption/decryption, and arithmetic circuits, efficiently.

We conduct experiments using MCNC benchmark circuits. When using the proposed approach, the number of CLBs (configurable logic blocks) is reduced by 67.6 % (compared to speed-optimized results) and 57.7 %(compared to area-optimized resutls), total equivalent gate counts are reduced by 65.5 %, maximum combinational path delay is reduced by 56.7 %, and maximum net delay is reduced by 80.5 % compared to conventional methods.

Key Words: Parity Prediction Functions, Davio Expansion, AND/XOR Expressions, FPGA, Logic Synthesis, Technology Mapping

요 약

본 논문에서는 FPGA (Field Programmable Gate Array)에 사용될 수 있는 AND/XOR기반의 기술적인 매핑 기법이 제안되었다. FPGA에서는 프로그램 블록들의 숫자가 정해져 있기 때문에 적절한 수의 입력을 가진 블록으로 회로를 나눌 수 있으면 효과적인 구현이 가능하다. Davio Expansion에 기반한 제안된 기법은 Davio Expansion 자체가 AND/XOR의 성질을 가지고 있기 때문에 XOR를 많이 포함하고 있는 에러 검출/수정, 데이터 암호/해독, 산술 회로 등을 구현하기 매우 용이하다.

본 논문에서는 제안된 기법을 이용할 때 구현되는 면적뿐만 아니라 속도도 현저히 저하될 수 있음을 MCNC 벤치마크를 이용하여 증명하였다. 면적이 줄어듦을 보이기 위하여 CLB (Configurable Logic Block) 숫자와 총 게이트 숫자가 이용되었다. CLB 숫자는 67.6 % (속도로 최적화된 결과)와 57.7 % (면적으로 최적화된 결과) 만큼 감소되었고 총 게이트 숫자는 65.5 %만큼 감소되었다. 속도관련 결과를 확인하기 위해 사용된 최대 Path Delay는 현재 사용되고 있는 방법들에 비해 56.7 %만큼 감소되었고 최대 Net Delay는 80.5% 만큼 감소되었다.

I . Introduction

RAM-based or re-programmable FPGAs have been used for reconfigurable computing and for design prototyping. Due to its unique characteristics, this type of FPGAs has been subject of fault-tolerant studies[1,2]. Techniques have been proposed[3] to identify faults and defects in such FPGAs. Also, on-line testing technique[4] has been proposed to exercise the

* Department of Electrical Engineering, University of Saskatchewan, Saskatoon, SK S7N 5A9, Canada (seokbum.ko@usask.ca)

논문번호 : 030248-0613, 접수일자 : 2003년 6월 13일

www.dbpia.co.kr
Copyright (C) 2003 NuriMedia Co., Ltd.

FPGA just before each reconfiguration occurs. Defective blocks are then avoided in the next configuration. Note that this technique differs from the CED technique discussed here.

An FPGA is made from a fixed number of fix-sized programmable or configurable blocks. In most cases, these configurable blocks consist of look-up tables or programmable universal logic gates. Therefore, when implementing a logic function, the realization complexity (the number of configurable blocks being used) has a stronger correlation to the number of inputs to the function than to the Boolean expression complexity. For instance, a configurable logic block (CLB) in a Xilinx's XC4010[5] has two first stage 4-input look-up tables (LUTs) and a 3-input LUT at the second stage. Therefore, when implementing in a XC4010, a 4-input Boolean function with one product term has the same realization complexity as a 4-input Boolean function with 10 product terms since both must fit one 4-input LUT. We remark here that such observation has been incorporated in most FPGA design automation tools. However, our problem is slightly different since we are dealing with the exclusive-OR(XOR) of multiple Boolean functions, i.e. the parity prediction function.

Previous studies, e.g. [6,7], showed that parity code is very effective in protecting logic circuits. We will concentrate on the techniques for an efficient implementation of parity prediction circuit in FPGA. The key issue here is the effective decomposition of the parity prediction circuit such that an efficient realization in FPGA is possible. Specifically, we investigate the potential benefit of manipulating the parity prediction function in the form of AND/XOR expression. The superiority of such approach is quite intuitive since parity prediction is inherently XOR intensive. However, the existing EDA tools do not handle the AND/XOR expressions well. In fact, a typical EDA tool will first translate an AND/XOR expression into an AND/OR one before further processing. Therefore, we derive here a process that will handle the AND/XOR expressions directly and independent of the existing EDA tools.

The Davio expansion theorem is applied here to the technology mapping method for FPGA. We design three different approaches: (1) Direct Approach, (2) AND/XOR Direct, and (3) proposed approach and conduct experiments using MCNC benchmark circuits to demonstrate the effectiveness of the proposed approach. We formulate the parity prediction circuits for the MCNC benchmark circuits. The proposed approach is superior to the conventional methods for parity prediction circuits in terms of both speed and area.

This paper is organized into five chapters. We provide an introduction in Chapter 1. The technical background is introduced in Chapter 2. Technology mapping for AND/XOR expressions is introduced in Chapter 3. In the following Chapter 4, we present the benchmark results and discussions. Finally, Chapter 5 draws conclusions.

II. Technical Background

The design of parity prediction circuit for any random logic circuit has been explained in several previous works[6,7,8]. The efficient implementation techniques of parity prediction circuits in FPGA are also introduced[9,10]. In this paper, we will consider the following approach. First, we assume that the original circuit is described in VHDL. The VHDL description of the parity prediction circuit is then formed by simply changing the original outputs to the internal signals and then exclusive-ORing these internal signals to produce the parity bit. After logic synthesis process, these internal signals will ideally be minimized and completely removed from the circuit. Such approach ties the parity prediction function directly to the original output functions. Any discrepancy that may arise due to the don't-care conditions is avoided.

Many researchers defined various classes of AND/XOR expressions and we use term ESOP because it is the most generic one. Arbitrary

www.dbpia.co.kr
Copyright (C) 2003 NuriMedia Co., Ltd.

product terms combined by XORs are called an Exclusive-or-Sum-of-Products Expression (ESOP). The ESOP is the most general AND/XOR expression. No efficient minimization method is known, and iterative improvement methods are regularly used to obtain near minimal solutions[11]. This is yet another reason that typical EDA tools do not incorporate ESOP type considerations.

Parity prediction circuits can be realized with many fewer gates if XOR gates are available. Such circuits can be derived from AND/XOR circuits. So the minimization of ESOP, which corresponds to the minimization of AND/XORs, is very important. ESOP requires fewer products than SOPs to realize randomly generated functions and symmetric functions[12].

The ESOP represents a Boolean function in an AND/XOR relation rather than the typical AND/OR format. In other words, instead of sum-of-products, the Boolean functions are expressed in exclusive-OR-sum-of-products. Given an n-input Boolean function $f(x_1, x_2, \cdots, x_n)$, a function can be expressed in ESOP as follows:

$$f_i(x_1, x_2, \cdots, x_n) = p_{i,1} \oplus p_{i,2} \oplus \cdots \oplus p_{i,q} = \sum_{t=1}^{q} p_{i,t} \quad (1)$$

where $p_{i,t}$ is the t^{th} product terms of $f_i(x_1, x_2, \cdots, x_n)$. Then, the parity prediction function of the logic circuit can be written as:

$$P(x_1, x_2, \cdots, x_n) = \sum_{i=1}^{m} f_i(x_1, x_2, \cdots, x_n) = \sum_{i=1}^{m} \sum_{t=1}^{q} p_{i,t} \quad (2)$$

The most remarkable feature of Equation (2) is that the boundaries between the original output, f_i have been removed. The parity function is now a function of all the product terms. In [13], Even, Kohavi and Paz proposed an algorithm that minimized single output completely specified equations by the repeated application of four rules that linked product terms. The four rules are Merging: $ab \oplus \bar{a}b = b$, Exclusion: $ab \oplus b = \bar{a}b$, Increase of Order: $ab \oplus \bar{a} = 1 \oplus a\bar{b}$ and Bridging: $ab \oplus \bar{a}\bar{b} = a \oplus b$. Their strategy is to apply the rule which give the greatest benefit first: if any merging is possible then *merging* is applied, if not then *exclusion*, if neither of these then *increase of order*, and if none of these then *bridging*. We implemented this algorithm in C language since it is especially suitable for parity prediction circuit (single output function) and extended it to handle multiple outputs.

There are many reasons that ESOP algorithms are not as popular in commercial EDA tools. As mentioned earlier, the lack of efficient minimization is a big issue. The minimizer algorithm applied here will give a solution in parity prediction circuit but not a guaranteed minimum solution. We note that XOR is not a primitive Boolean operator, such as AND, OR and NOT. Consequently, the hardware realization of XOR gate is not straightforward. In the CMOS realization, 2-input XOR gate needs about twice the area than that of a 2-input AND or an NOR gate. However, such disadvantage in hardware efficiency does not exist in FPGA implementation.

In this section, we describe the organization of the Xilinx XC 4000 FPGA family. Although our experimental results will base on this FPGA family, extension to other FPGA families, even from different vendors, may be easily obtained. Xilinx XC4000E/XL[14] consists of an array of CLBs embedded in a configurable interconncet structure and surrounded by configurable I/O blocks. The Xilinx XC4000 family consists of ten members. The family members differ in the number of CLBs, (ranging from 8×8 to 56×56), and I/O blocks, (ranging from 64-448). The typical capacity varies from 1,600 to 85,000 equivalent gates.

Ⅲ. Technology Mapping for AND/XOR Expressions

Technology mapping is the logic synthesis task

www.dbpia.co.kr
Copyright (C) 2003 NuriMedia Co., Ltd.

that is directly concerned with selecting the circuit elements used to implement the minimized circuit. Previous approaches to technology mapping have focused on using circuit elements from a limited set of simple gates. However, such approaches are inappropriate for complex logic blocks where each logic block can implement a large number of functions[15]. A K-input LUT can implement 2^{2^K} different functions. For values of K greater than 3, the number of different functions becomes too large for conventional technology mapping. Therefore, different approaches to technology mapping are required for LUT-based FPGAs.

Technology mapping is accomplished by translating the given Boolean expressions into hierarchical ones. The transformed expression will consist of sub-expression that uses only a subset of all inputs. Of course, a sub-expression can be made from sub-expressions that use even smaller subset of inputs. The proposed is so the lowest level sub-expression, which is still a Boolean function, can be mapped directly to the smallest realizable device. In this paper, the smallest realizable device is set a CLB which can realize any 5-input, or less, Boolean function. The goal here is obviously to transform the given Boolean functions into hierarchical expression where all the lowest level sub-expressions have five or less inputs.

An arbitrary logic function $f(x_1, x_2, \cdots, x_n)$ can be expanded as

$$f = \overline{x_1} f_0 \oplus x_1 f_1, \quad (3)$$

$$f = f_0 \oplus x_1 f_2, \quad (4)$$

$$f = f_1 \oplus \overline{x_1} f_2, \quad (5)$$

where $f_0 = f(0, x_2, \cdots, x_n)$, $f_1 = f(1, x_2, \cdots, x_n)$ and $f_2 = f_0 \oplus f_1$.

Equations (3), (4) and (5) are called the Shannon expansion, the positive Davio expansion, and the negative Davio expansion, respectively.

3.1 XOR-Extension of Positive Davio Expansion Theorem

Given an n-input Boolean function, $f(x_1, x_2, \cdots, x_n)$, the x_i-residue of the function is $f_{x_i}(x_1, \cdots, x_i = 1, \cdots, x_n)$ and the $\overline{x_i}$-residue is $f_{\overline{x_i}}(x_1, \cdots, x_i = 0, \cdots, x_n)$ for $1 \le i \le n$. Let us now consider a circuit with n inputs and two outputs, $f_1(x_1, x_2, \cdots, x_n)$ and $f_2(x_1, x_2, \cdots, x_n)$. To predict the parity of these two outputs we wish to derive a function $P(x_1, x_2, \cdots, x_n) = f_1(x_1, x_2, \cdots, x_n) \oplus f_2(x_1, x_2, \cdots, x_n)$. According to positive Davio theorem, we may write:

$$P = (f_{1,\overline{x_i}} \oplus f_{2,\overline{x_i}}) \oplus x_i (f_{1,x_i} \oplus f_{2,x_i} \oplus f_{1,\overline{x_i}} \oplus f_{2,\overline{x_i}}). \quad (6)$$

In fact, such relation can be expanded to multiple functions. Let $f_l(x_1, x_2, \cdots, x_n)$ be the functions of a logic circuit with n inputs and m outputs such that $1 \le l \le m$. The parity prediction function of this circuit is then:

$$P = (\sum_{l=1}^{m} f_{l,\overline{x_i}}) \oplus x_i (\sum_{l=1}^{m} f_{l,x_i} \oplus \sum_{l=1}^{m} f_{l,\overline{x_i}})$$

for any $1 \le i \le n$. (7)

In the above equation, $\sum_{l=1}^{m} f_{l,x_i}$ represents the exclusive-OR sum of all the m x_i-residues. Note that the XOR-extension of the positive Davio expansion theorem can be recursively applied such that:

www.dbpia.co.kr
Copyright (C) 2003 NuriMedia Co., Ltd.

$$P = (\sum_{l=1}^{m} f_{l,\overline{x_i}\overline{x_j}}) \oplus x_j(\sum_{l=1}^{m} f_{l,\overline{x_i}x_j} \oplus \sum_{l=1}^{m} f_{l,\overline{x}_i x_j}) \oplus x_i(\sum_{l=1}^{m} f_{l,\overline{x_i}x_j} \oplus \sum_{l=1}^{m} f_{l,x_i\overline{x_j}}) \oplus x_i x_j(\sum_{l=1}^{m} f_{l,x_i x_j} \oplus \sum_{l=1}^{m} f_{l,x_i\overline{x_j}} \oplus \sum_{l=1}^{m} f_{l,\overline{x}_i x_j} \oplus \sum_{l=1}^{m} f_{l,\overline{x_i}\overline{x_j}}) \quad (8)$$

3.2 XOR-Extension of Negative Davio Expansion Theorem

Negative Davio expansion stated that $f = f_1 \oplus \overline{x_1} f_2$ in equation (5). Let us consider a circuit with n inputs and two outputs, $f_1(x_1, x_2, \cdots, x_n)$ and $f_2(x_1, x_2, \cdots, x_n)$. To predict the parity of these two outputs we wish to derive a function $P(x_1, x_2, \cdots, x_n) = f_1(x_1, x_2, \cdots, x_n) \oplus f_2(x_1, x_2, \cdots, x_n)$. According to negative Davio theorem, we may write:

$$P = (f_{1,\overline{x_i}} \oplus f_{2,\overline{x_i}}) \oplus x_i(f_{1,\overline{x_i}} \oplus f_{2,\overline{x_i}} \oplus f_{1,x_i} \oplus f_{2,x_i}) \quad (9)$$

In fact, such relation can be expanded to multiple functions. Let $f_1(x_1, x_2, \cdots, x_n)$ be the functions of a logic circuit with n inputs and m outputs such that $1 \le l \le m$. The parity prediction function of this circuit is then:

$$P = (\sum_{l=1}^{m} f_{l,\overline{x_i}}) \oplus \overline{x_i}(\sum_{l=1}^{m} f_{l,\overline{x_i}} \oplus \sum_{l=1}^{m} f_{l,x_i}) \quad \text{for any } 1 \le i \le n. \quad (10)$$

In the above equation, $\sum_{l=1}^{m} f_{l,\overline{x_i}}$ represents the exclusive-OR sum of all the m $\overline{x_i}$-residues. Note that the XOR-extension of the negative Davio expansion theorem can be recursively applied such that:

$$P = (\sum_{l=1}^{m} f_{l,\overline{x_i}\overline{x_j}}) \oplus \overline{x_j}(\sum_{l=1}^{m} f_{l,\overline{x_i}\overline{x_j}} \oplus \sum_{l=1}^{m} f_{l,\overline{x_i}x_j}) \oplus \overline{x_i}(\sum_{l=1}^{m} f_{l,\overline{x_i}\overline{x_j}} \oplus \sum_{l=1}^{m} f_{l,\overline{x}_i\overline{x_j}}) \oplus \overline{x_i}\,\overline{x_j}(\sum_{l=1}^{m} f_{l,\overline{x_i}\overline{x_j}} \oplus \sum_{l=1}^{m} f_{l,\overline{x_i}x_j} \oplus \sum_{l=1}^{m} f_{l,\overline{x}_i\overline{x_j}} \oplus \sum_{l=1}^{m} f_{l,\overline{x_i}\overline{x_j}}) \quad (11)$$

3.3 Proposed Technology Mapping Method Based on Davio Expansion

As shown in equations (4) and (5) and in Sections 3.1 and 3.2, the Davio expansions can be seen as manipulations of the Shannon's expansion utilizing the XOR property. Based on our observation, $f_2 = (f_0 \oplus f_1)$ is usually no more complex than f_0 or f_1 itself. Therefore, the Davio expansions constantly lead to solutions with less literal counts. The resulting functions from the Davio expansions are, of course, AND/XOR expressions.

Based on the Davio expansions, we derived the following greedy-type algorithm. This algorithm attempts to decompose the given logic circuit into blocks or sub-functions, each with five or less inputs. This number is due to the Xilinx XC 4000 FPGA organization. For other FPGA families, different number may be used. Figure 1 shows the Davio-Decompose algorithm.

A given logic circuit, f, can be identified that contains product terms, $p_1, p_2, \cdots, p_t$, where $t \ge 1$, then we define $f = \sum_{i=1}^{t} p_i$, where $\sum$ represents XOR-sum operation. We will use f_i^0, f_i^1, and f_i^2, where $f_i^0 = f(x_1, x_2, \cdots, x_i = 0, \cdots, x_n)$, $f_i^1 = f(x_1, x_2, \cdots, x_i = 1, \cdots, x_n)$, and $f_i^2 = f_i^0 \oplus f_i^1$ for our method.

Step 1:

if (# of input variables of $f \le 5$) **exit;**

/* f can be fit into a CLB of XC 4000 */

www.dbpia.co.kr
Copyright (C) 2003 NuriMedia Co., Ltd.

else go to **Step 2**;

Step 2: Choose the least frequently used input variable leading to the simplest residual functions in f and let it be x_i.

Step 3: Decompose f into $f = f_i^0 \oplus x_i f_i^2$.

Step 4:

if (# of input variables in $f_i^0 > 5$) replace f with new f_i^0 and go to **Step 2**;

else go to **Step 5**;

Step 5:

if (# of input variables in $f_i^2 > 5$) replace f with new f_i^2 and go to **Step 2**;

else go to **Step 6**;

Step 6: Update the input list of f after performing the Davio expansion, and replace f with the new f_i^0 and f_i^2.

Step 7: exit.

Figure 1. Proposed Davio-Decompose algorithm

Ⅳ. Benchmark Results and Discussions

We examine the proposed Davio-Decompose algorithm on the parity prediction circuits. The experiments are conducted on the MCNC benchmark circuits. The target is the Xilinx's XC4010 which has 400 CLBs with 7-20K equivalent gates. The software is the Xilinx's Foundation 2.1i with Synopsys' FPGA Express.

The parity prediction circuits are obtained consequently after converting the original MCNC benchmark circuits described in PLA format into VHDL. We design three different approaches for experiments as follows: (1) direct approach: In case each obtained parity prediction circuit is directly applied to Xilinx Foundation and FPGA, it is termed direct approach. (2) AND/XOR direct: Each parity prediction circuit is converted into AND/XOR expressions and then applied to Xilinx Foundation and FPGA. We call this AND/XOR direct here. (3) the proposed approach: The converted AND/XOR expression is decomposed into sub-circuits which have five or less input variables by the proposed Davio-Decompose algorithm and finally applied to Xilinx Foundation and FPGA. We call it proposed approach here.

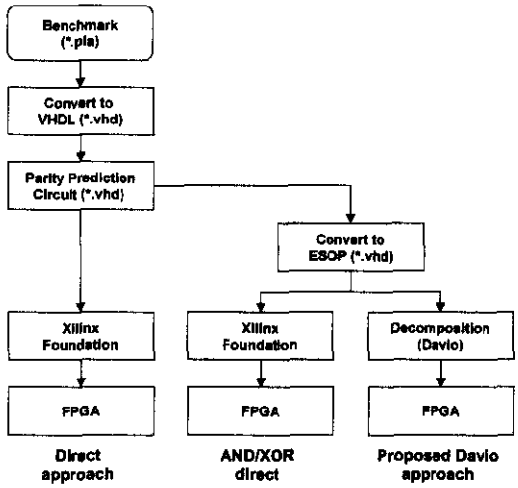

Figure 2. Design flow for parity prediction circuit of MCNC benchmark circuit

To get an experimental result for the proposed approach, we did some pre-processing works. To keep the optimized AND/XOR expression, we made a separate VHDL code for each sub-circuits so that it is forced to be implemented in a CLB of Xilinx XC4010. Recall that conventional EDA tools cannot handle AND/XOR expressions efficiently.

Figure 2 shows the three different design paths for the following experiments. Table 1 lists the results of parity prediction circuits of MCNC benchmark circuits when applying direct approach, AND/XOR direct, and the proposed approach, respectively. Here we list the number of CLBs since these are the direct evidence of the hardware efficiencies of these techniques.

4.1 Number of CLBs

By direct approach, we convert the VHDL description of the original circuit into the VHDL description for the parity prediction circuit as shown in Figure 2. In some cases, the results are quite good, e.g., ***bw***, ***squar5***, and ***rd53*** as shown in Table 1. The XC4010 CLB can realize any

www.dbpia.co.kr
Copyright (C) 2003 NuriMedia Co., Ltd.

5-input functions, but also some 6-, 7-, 8-, and even 9-input functions[14,16]. The complexity of the function is usually the key to realization efficiency. Hence, it is not surprised to see that such straightforward approach can achieve efficient implementation sometimes.

Table 1 Results of parity prediction circuits in terms of # of CLBs

circuits \ stat.	# in	# out	# product terms	Original		Direct		AND/XOR direct		proposed approach
				A*	B*	A*	B*	A*	B*	
misex1	8	7	32	11	10	7	5	3	5	3
sqrt8	8	4	40	13	10	7	6	5	6	3
5xp1	7	10	75	18	15	33	28	17	17	5
inc	7	9	34	22	20	15	8	10	8	5
con1	7	2	9	3	2	3	2	3	4	3
bw	5	28	87	32	31	1	1	1	1	1
rd53	5	3	32	3	3	1	1	1	1	1
squar5	5	8	32	6	6	1	1	1	1	1
Total	52	71	341	108	97	68	52	41	43	22
Average	6.5	8.9	42.6	13.5	12.1	8.5	6.5	5.1	5.4	2.75

A: the number of CLBs when optimized for speed in Xilinx Foundation 2.1i
B: the number of CLBs when optimized for area in Xilinx Foundation 2.1i

Next, we applied the AND/XOR direct to the parity prediction circuits of MCNC benchmark circuits. The AND/XOR expression can break the boundary of each internal signal which is the original output. After breaking the boundary, it can be minimized by different techniques[13,17]. We implemented here the minimization method described in Chapter 2 in C language. The minimized AND/XOR expression in VHDL is then processed by Synopsys's FPGA Express and Xilinx Foundation 2.1i. We observe that the number of CLBs is still related to the complexity of the original logic function. Compared to direct approach, the AND/XOR direct shows better results except one case, circuit ***con1***.

The results of this AND/XOR direct is somewhat surprising. We observed that the AND/XOR expression of the parity function can be rather small after the minimization. However, such advantage was lost after the processing by the Xilinx's Foundation 2.1i with Synopsys' FPGA Express. Note that both packages use AND/OR-based approaches. Therefore, the minimized AND/XOR expressions were converted back to AND/OR by these packages before the mapping results were obtained. Hence, the good results shown in Table 1 for the AND/XOR direct can be better if the technology mapping is also done with an AND/XOR-based approach.

Finally, we applied the proposed approach as shown in Figure 2 after getting the minimized AND/XOR expressions. In all eight benchmark cases, the proposed approach shows the best result. The proposed approach only needs 22 extra CLBs for eight benchmark circuits. It takes only on average 2.75 extra CLBs or 20 % of the original area. We remark here that the complexity of the parity prediction function is proportional to its original function. The complexity of 8 benchmark circuits can be seen from the number of inputs, outputs and product terms, and the number of CLBs columns in Table 1. In conclusion, the proposed method can indeed provide the most efficient realization of parity function in all cases here. This is mainly accomplished by bypassing the AND/OR type mapping and with dedicated AND/XOR Davio expansion for technology mapping.

4.2 Total Equivalent Gate Counts

We collect the information about the total number of equivalent count for three different approaches as shown in Table 2. In the case of ***5xp1***, for example, direct approach requires 453 equivalent gates and AND/XOR direct requires 208 equivalent gates, but the proposed approach requires only 54 equivalent gates. In conclusion, The proposed approach shows the best result in terms of total equivalent gate count by showing 35.13 equivalent gates on average while direct approach takes 108.25 equivalent gate counts on average and AND/XOR direct takes 64.75 equivalent gate counts on average. In other words, the same function can be implemented with the least number of total equivalent gate counts when

www.dbpia.co.kr
Copyright (C) 2003 NuriMedia Co., Ltd.

using the proposed approach.

Table 2 Results of parity prediction circuits in terms of total equivalent gate counts

stat. / circuits	# in	# out	# product terms	direct approach	AND/ XOR direct	proposed approach
misex1	8	7	32	75	34	33
sqrt8	8	4	40	82	63	43
5xp1	7	10	75	453	208	54
inc	7	9	34	178	129	66
con1	7	2	9	30	36	37
bw	5	28	87	16	16	16
rd53	5	3	32	16	16	16
squar5	5	8	32	16	16	16
Total	52	71	341	866	518	281
Average	6.5	8.9	42.6	108.25	64.75	35.13

Table 3. Results of parity prediction circuits in terms of maximum combinational path delay & maximum net delay

stat. / circuits	direct approach		AND/ XOR direct		proposed approach	
	A*	B*	A*	B*	A*	B*
misex1	28.576	22.594	18.824	3.724	6.765	5.417
sqrt8	25.173	21.559	30.782	7.204	5.862	3.802
5xp1	49.317	27.354	35.672	9.181	18.381	5.896
inc	35.452	28.762	25.868	6.409	27.885	6.086
con1	21.634	25.334	23.494	4.030	22.196	3.567
bw	14.123	15.642	15.656	2.557	1.909	2.396
rd53	14.455	15.713	15.713	2.742	2.241	2.742
squar5	16.836	17.110	16.836	3.737	3.737	4.116
Total	205.566	174.068	182.845	39.584	88.976	34.022
Average	25.696	21.759	22.856	4.948	11.122	4.253

A: Maximum Combinational Path Delay (ns)
B: Maximum Net Delay (ns)

4.3 Maximum Combinational Path Delay and Maximum Net Delay

We also collect maximum combinational path delay and maximum net delay as shown in Table 3 to see if the proposed method in Chapter 3 is appropriate for the point of view of speed consideration. In the case of maximum combinational path delay, direct approach shows 25.696 ns on average and AND/XOR direct shows 22.856 ns on average. The proposed approach shows the shortest maximum combination path delay by showing 11.122 ns. In the case of maximum net delay, direct approach shows 21.759 ns and AND/XOR direct shows 4.948 ns on average. The proposed approach shows 4.253 ns on average.

In conclusion, the proposed approach shows the best performance in terms of speed and area. For both direct approach and AND/XOR direct, the results are obtained directly from the package. However, for the proposed approach, we have to do some pre-processing to get the results from the package since the package is based on AND/OR expressions and is not able to manipulate the AND/XOR expressions.

V. Conclusions

We have presented here AND/XOR-based minimization and technology mapping techniques for the efficient realization of XOR-intensive functions in FPGA. The efficiency came from the fact that the proposed Davio expansion-based technology mapping method is an AND/XOR-based method. Specifically, the proposed method is shown to reduce the number of CLBs up to 57.7 % (compared to area-optimized results) and 67.6 % (compared to speed-optimized results), respectively. The proposed method can also reduce the total number of equivalent gate counts. Further, the proposed method can reduce the maximum combinational path delay by 56.7 % and the maximum net delay by 80.5 %. Indeed, the proposed method is far superior to the conventional method for the parity functions.

We expect similar results, as reported above for the parity functions, could be obtained for other XOR-intensive functions. As many XOR-intensive functions, such as error detecting/correcting, data encryption/ decryption, and computer arithmetic circuits, are emerging, the proposed method is an important step forward.

www.dbpia.co.kr
Copyright (C) 2003 NuriMedia Co., Ltd.

References

[1] R. Cuddapah and M. Corba, *Reconfigurable Logic for Fault Tolerance*, Springer-Verlag, 1995.

[2] F. Hanchek and S. Dutt, "Methodologies for Tolerating Logic and Interconnect Faults in FPGAs," *IEEE Trans. on Computers*, Vol. 47, No. 1, pp. 15-33, Jan. 1998.

[3] W. K. Huang, F. J. Meyer, X. Chen, and F. Lombardi, "Testing Configurable LUT-Based FPGAs," *IEEE Trans. on VLSI Systems*, Vol. 47, No. 6, pp. 276-283, June 1998.

[4] M. Abramovici, C. Stroud, S. Wijesuriya, C. Hamilton, and V. Verma, "Using Roving STARs for On-Line and Diagnosis of FPGAs in Fault-Tolerant Applications," *Proc. ITC*, pp. 973-982, Oct. 1999.

[5] Xilinx Inc., http://www.xilinx.com.

[6] N. A. Touba, and E. J. McCluskey, "Logic Synthesis of Multilevel Circuits with Concurrent Error Detection," *IEEE Transactions on Computer-Aided Design*, Vol. 16, No. 7, pp. 783-789, Jul. 1997.

[7] C. Bolchini, F. Salice and D. Sciuto, "A Novel methodology for Designing TSC Networks based on the Parity Bit Code," *Proc. European Design & Test Conf.*, pp. 440-444, March 1997.

[8] J. C. Lo, M. Kitakami and E. Fujiwara, "Reliable Logic Circuits using Byte Error Control Codes," *Proc. Int'l Symposium on Defect and Fault Tolerance in VLSI Systems*, pp. 286-294, October 1996.

[9] S. B. Ko, T. Xia and J. C. Lo, "Efficient Error Prediction in FPGA," *IEEE Int'l Symposium on Defect and Fault Tolerance in VLSI systems*, pp. 176-181, Oct. 2001.

[10] S. B. Ko and J. C. Lo, "Efficient Decomposition Techniques for FPGAs," *Lecture Notes in Computer Science (IEEE International Conference on High Performance Computing)*, Vol. 552, pp. 630-639, December 2002, Springer-Verlag

[11] T. Sasao, "Logic Synthesis and Optimization," *Kluwer Academic Publishers*, 1998

[12] T. Sasao and P. Besslich, "On the complexity of MOD-2 Sum PLAs," *IEEE Transactions on Computers*, Vol. 32, No. 2, pp. 262-266, Feb. 1990.

[13] S. Even, I. Kohavi and A. Paz, "On minimal modulo-2 sums of products for switching functions," *IEEE Transactions on Electronic Computers*, EC-16:671-674, Oct. 1967

[14] Xinlinx Inc., Xilinx Data Book: XC4000E and XC4000X Series, May 1999.

[15] J. Cong and Y. Ding, "Combinational Logic Synthesis for LUT Based Field Programmable Gate Arrays," *ACM Transactions on Design Automation of Electronic Systems*, Vol. 1, No. 2, pp. 145-204, April 1996

[16] J. Cong and Y. -Y. Hwang, "Boolean Matching for Complex PLBs in LUT-based FPGAs with Application to Architecture Evaluation," *Proc. ACM 6th Int'l Symposium on FPGA*, pp. 27-34, Feb. 1998.

[17] M. Helliwell, and M. Perkowski, "A Fast Algorithm to Minimize Multi-Output Mixed-Polarity Generalized Reed-Muller Forms," *Proc. ACM/IEEE Design Automation Conf.*, pp. 427-432, 1988.

Biography

Seok-Bum Ko received his Ph. D in computer engineering from University of Rhode Island in 2002. He is currently working as an assistant professor at the Department of Electrical Engineering, University of Saskatchewan, Canada. His research interests include computer architecture, fault-tolerant computing, and VLSI testing. Dr. Ko is a member of IEEE Computer Society.

www.dbpia.co.kr
Copyright (C) 2003 NuriMedia Co., Ltd.