TLU형 FPGA를 위한 새로운 다출력 함수 기술 매핑 알고리즘

박 장 현† · 김 보 관††

요 약

본 논문에서는 최근에 관심을 모으고 있는 Table Look-Up형의 FPGA를 위한 다출력 함수로직 합성 알고리즘에 대해 기술한다. 본 고에서 제안하는 TLU형 FPGA를 위한 다출력 함수 로직 합성 방법은 기능적 분해 방법을 사용하였으며, 이 방법을 이용한 2가지의 새로운 알고리즘을 설명한다. 첫번째는 한 출력에 적용한 Roth-Karp 알고리즘을 다출력에 응용할 수 있도록 확장하였으며, 두 번째는 분해과정에서 공통 분해 함수를 찾는 효과적인 알고리즘을 제안한다. 기술 매핑의 최적화 대상은 CLB 개수를 고려했으며, 벤치마크 테스트를 통한 일반적인 회로에 적용성 검증, 기존 알고리즘과의 성능 비교 및 개선에 대해 연구하였다. 논리 설계 합성기 구성 과정에서 새로운 알고리즘을 구현하여 실험한 결과를 기존의 다출력 함수 분해 방법과 비교하면 CLB의 개수, 네트 수 등 성능과 수행 시간에서 매우 만족할 만한 결과를 얻었다.

New Technology Mapping Algorithm of Multiple-Output Functions for TLU-Type FPGAs

Jang-Hyun Park† · Bo-Gwan Kim††

ABSTRACT

This paper describes two algorithms for technology mapping of multiple output functions into interesting and popular FPGAs(Field Programmable Gate Arrays) that use look-up table memories. For improvement of technology mapping for FPGA, we use the functional decomposition method for multiple output functions. Two algorithms are proposed. The one is the Roth-Karp algorithm extended for multiple output functions. The other is the novel and efficient algorithm which looks for common decomposition functions through the decomposition procedure. The cost function is used to minimize the number of CLBs and nets and to improve performance of the network. Finally we compare our new algorithm with previous logic design technique. Experimental results show significant reduction in the number of CLBs and nets.

1. 서 론

FPGA(Field Programmable Gate Array)는 Carter [1]등에 의해 제안된 가장 새로운 프로그램 가능한 부품으로서, 기존의 MPGA(Mask Programmable Gate Array)와 같이 배선 영역을 사이에 둔 로직 셀의 2차원적 배열이나, PLD(Programmable Logic Devices)와 같이 로직 셀의 기능과 셀들 간의 상호연결을 사용자가 임의로 결정할 수 있도록 되어있다. 이와 같

† 정 회 원:한국전자통신연구원 선임연구원
†† 정 회 원:충남대학교 전자공학과 부교수
논문접수:1997년 5월 6일, 심사완료:1997년 10월 8일

이 기존의 두 semi-custom 구조를 혼합함으로써 FPGA는 짧은 제작 시간, 저렴한 제작 가격, 제작과 테스트 용이성 및 고집적도 등 PLD와 MPGA의 장점을 두루 갖추고 있어, 시스템의 시제품 제작 및 소량생산에 매우 적합한 소자이다.

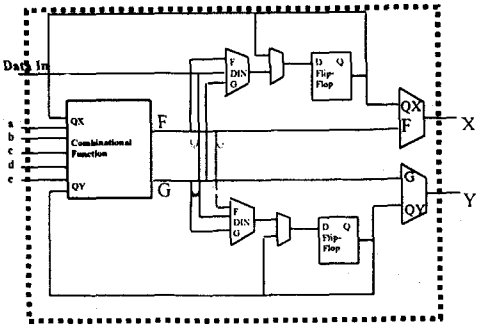
FPGA 구조에 회로를 구현하기 위한 연결은 정해진 배선 영역을 사용하여 프로그램되어야 하며, 블럭 구조에 따라 TLU(Table Look-Up)형과 MB(Multiplexor Based)형으로 분류한다. TLU형 구조에서 로직의 기본 단위를 CLB(Configurable Logic Block)[2]라고 부른다. TLU형의 기본 블럭은 m개의 입력으로 어떤 함수도 구현 가능토록 되어있으며, TLU 구조가 결정되면 m은 항상 고정되어 있다. 이 구조의 전형적인 예는 Xilinx(XC3000)[2] 구조로서 이 경우 m=5이다.

CLB의 어레이 구조는 사용자 로직을 구현 가능하도록 기능 요소들을 제공한다. (그림 1)에 Xilinx 3090 CLB[2]의 구조를 보여주고 있으며, 각 CLB는 조합회로 부분과 2개의 플립플롭, 내부 제어 부분으로 구성되어 있다. 입력 신호는 5개의 변수 입력과 한 개의 데이타 입력이 있으며, 또한 CLB 외부로 연결되는 2개의 출력신호를 갖고 있다. (그림 1)에 나타난 Xilinx 3090 CLB의 조합논리 함수에 대한 주요 기능은 다음과 같다.

1)7개의 입력 변수 (a, b, c, d, e, Qx, Qy)중 5개를 입력하여 어떤 함수(F 혹은 G)도 구현이 가능하다.

2)각 4개 이하의 변수를 입력하는 경우 두 조합함수 (F와 G)를 구현할 수 있다. 이때 조건은 한 입력 변수(a)는 두 함수에 공통으로 들어가고, 두 번째 변수는 b, Qx, Qy 중 하나이고, 세 번째는 c, Qx, Qy 중 하나이고, 마지막 네 번째 변수는 d 혹은 e 중 하나이다.

3)각 CLB는 2개의 출력(X와 Y)을 가진다. 출력 X는 Qx 혹은 F 중 하나이고, 출력 Y는 Qy 혹은 G 중 하나이다. 각 플립플롭의 출력 Qx, Qy는 한 CLB내의 TLU 입력으로 사용 가능하다.



(그림 1) Xilinx 3090 CLB 구조
(Fig. 1) Xilinx 3090 CLB Architecture

MISII[3]나 GLSS[4] 등의 임의 논리회로 구현을 대상으로 하는 다단계논리 합성 도구에서는 면적의 평가로서 게이트의 수나 불리안 회로의 노드에서 문자 수를 사용하였다. 그러나, 이것은 CLB의 수를 최소화하는 평가 방법에는 적합치 않다. 왜냐하면 아무리 복잡한 표현식을 갖는 함수일지라도 최대 입력변수의 수에 대한 조건만 만족된다면 하나의 CLB로 구현될 수 있기 때문이다. 예를 들면, m=5 이고, f_1 = abcdeg, f_2 = abc + b′de + a′e′ + c′d′라 하자. 함수 f_1과 f_2는 각각 6개, 10개의 문자 수를 가지고 있다. 함수 f_1는 그것의 최적 구현에서 2개의 CLB가 요구되나, f_2는 5개의 변수만 갖고 있으므로 한 개의 CLB로 구현이 가능하다. 이와 같이 목적함수는 함수를 구현하는 실제적인 로직 보다도 입력의 수에 중점을 두어야만 한다. 전체 회로에서 입력이 m보다 작은 중간 노드의 수가 이 회로에 필요한 CLB 수의 상한 개수이다. 기본적인 문제는 논리식에 의한 회로가 주어지면 목표 FPGA 구조의 기본 블럭(CLB)을 사용하여 회로를 구현하는 것이다.

서론에 이어 2장에서는 FPGA 특성을 이용한 다출력 함수 로직합성에 대한 기존 알고리즘을 분석하고, 3장에서는 2종류의 새로운 알고리즘을 이용한 다출력 함수 로직 합성 알고리즘에 대해서 설명한다. 4장에서는 제안된 로직 합성 알고리즘에 대한 실험 결과를 보여주며, 마지막 5장에서는 결론 및 앞으로 할 일에 대해서 서술한다.

2. 기존 알고리즘 분석 [6, 7, 8, 9, 10, 11]

로직 합성을 한다는 것은 불리안 함수들을 주어진 목표 기술의 네트 리스트로 변환하는 것이다. 그것의 복잡성 때문에 로직합성은 일반적으로 두 단계로 분리한다. 첫번째 단계로서 목표 기술에 독립적인 것으로 주어진 회로를 공동 부함수로 동일화와 추출을 통하여 다단계 회로를 만드는 것이다. 로직 합성의 두 번째 단계는 다단계로 최적화된 회로를 주어진 목표

```
CNU_MO{
(1)  Find all equivalence classes for each output
(2)  Find global partition for the given function
(3)  Check the condition of common sub-functions
(4)  Find the worst condition for every output
     repeat
(5)     Check the decomp. fn of the selected output
(6)     Check the decomposition function of the others
     until (a decomposition found);
} * end CNU_MO
```

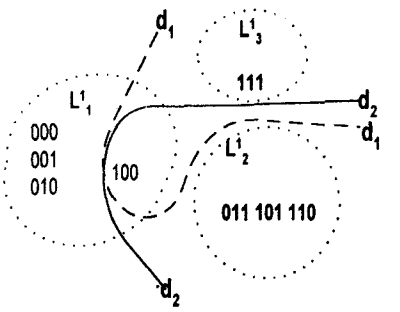
(그림 3) 다출력 함수의 기능적 분해 알고리즘
(Fig. 3) Functional Decomposition Algorithm of Multiple Output Function

분해표를 사용하여 동등 클래스를 찾는다. f_1의 함수가 〈표 1〉과 같이 주어지면, 이 함수는 3개의 동등 클래스를 갖고 있다. 즉 $L_1=\{000, 001, 010, 100\}$, $L_2=\{011, 101, 110\}$, $L_3=\{111\}$이다. 이 동등 클래스가 주어지면 분해 함수 d(x)를 구하여 주어진 함수를 분해할 수 있다. 이때 최소의 분해함수 개수는 $c=\lceil \log_2 L \rceil$ 이다. 〈표 1〉에서는 $L=3$ 이므로 $c=2$이다. (그림 4)에 표시된 분해함수 $d_1(x)=x_1x_2x_3+x_1x_2'x_3'$, $d_2(x)=x_1x_3'+x_1'x_2x_3+x_1x_2'x_3$을 선택하면 각 동등 클래스 $X\in\{000, 001, 010\}$에 대하여 d(x)=(00), $X\in\{011, 101, 110\}$에 대하여 d(x)=(01), $X\in\{111\}$에 대하여 d(x)=(10), $X\in\{100\}$에 대하여 d(x)=(11)로 표현 가능하다.

함수 f_2가 〈표 2〉의 함수 분해표로 주어진다면 f_2의 동등 클래스는 아래와 같이 표시된다.

〈표 1〉 함수 분해표(f_1)
〈Table 1〉 Function Decomposition Chart(f_1)

X_1 X_2 X_3								Y_1 Y_2
000	011	010	011	100	101	110	111	
0	0	0	1	0	1	1	1	00
1	1	1	1	1	1	1	0	01
1	1	1	1	1	1	1	0	10
0	0	0	1	0	1	1	0	11



(그림 4) 분해함수 선택 방법
(Fig. 4) Selection Method of Decomposition Function

$L_1^2=\{000\}$, $L_2^2=\{001\ 010\ 100\ 110\}$, $L_3^2=\{011\ 101\}$, $L_4^2=\{111\}$

〈표 2〉에서 동등 클래스 $L^2=4$이므로 최소분해 함수 개수 $c_2=2$이다.

〈표 2〉 함수 분해표(f_2)
〈Table 2〉 Function Decomposition Chart(f_2)

X_1 X_2 X_3								Y_1 Y_2
000	001	010	011	100	101	110	111	
0	0	0	1	0	1	0	1	00
0	1	1	1	1	1	1	0	01
0	1	1	1	1	1	1	0	10
1	1	1	0	1	0	1	0	11

(2) 다출력 함수의 전체 분할 그룹을 구한다.

〈표 1〉과 〈표 2〉에 주어진 두 함수의 전체분할 그룹을 구하면 $\{G_1\ G_2\ G_3\ G_4\ G_5\}\Rightarrow\{000\}\ \{001\ 010\ 100\}\ \{110\}\ \{011\ 101\}\ \{111\}$이다.

(3) 전체 분할 그룹으로부터 공통 분해함수가 존재할 조건을 조사한다.

전체 분할 그룹이 5개이므로 주어진 두 함수를 표현하기 위해서는 최소 3개의 분해함수가 필요하다. 즉, 3개의 분해함수로 f_1과 f_2를 모두 표현하면 최적의 해가 된다.

(4) 각 출력에 대해 최악의 조건 함수를 구한다.

3개의 분해함수로 f_1과 f_2를 표현해야 하므로 한 분해함수는 공통으로 사용되어야 한다. 함수 f_1의 경우

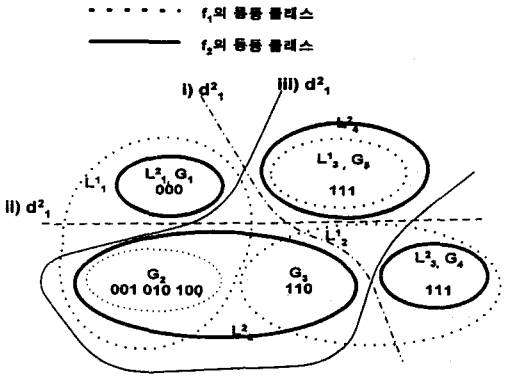
2개의 분해함수로 3개의 동등 클래스를 표현하면 되고, 함수 f_2의 경우 2개의 분해함수로 4개의 동등 클래스를 표현해야만 한다. 이 경우 f_2가 최악의 조건 분해 함수가 된다.

(5) 선택된 출력 분해 함수를 조사한다.

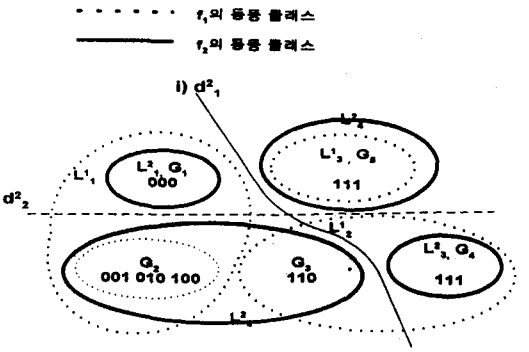
함수 f_2가 최악의 조건함수로 선택되면 함수 f_1과 f_2에 공통 사용될 분해함수(d_1^2)를 찾는다. 이때 함수 f_2에 주어진 조건을 만족하는 경우는 아래와 같이 3가지 경우가 있다(그림 5 참조).

i) d_1^2가 (L_1^2, L_2^2)과 (L_3^2, L_4^2)로 분리
ii) d_1^2가 (L_1^2, L_3^2)과 (L_2^2, L_4^2)로 분리
iii) d_1^2가 (L_1^2, L_4^2)과 (L_2^2, L_3^2)로 분리

조건 i)인 경우 d_2^2가 (L_1^2, L_3^2)과 (L_2^2, L_4^2)로 분리 되면 d_1^2와 d_2^2으로 함수 f_2의 표현이 가능하다(그림 6 참조).



(그림 5) 공통 사용될 분해 함수 찾기
(Fig. 5) Look for Common Decomposition Function



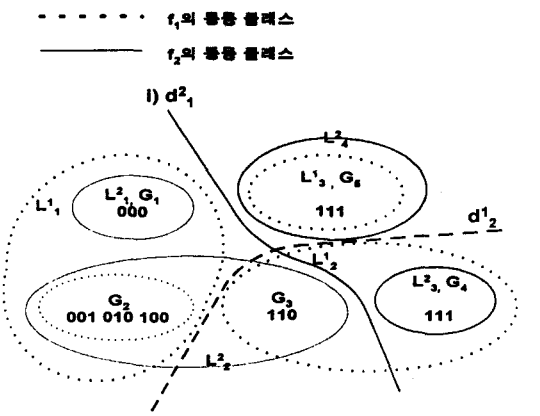
(그림 6) 선택된 출력 함수 분해 찾기
(Fig. 6) Look for Decomposition Function of Selected Function

6) 선택 안된 출력 분해 함수를 조사한다.

만약 d_1^2을 공통 분해함수로 사용하여 (그림 7)에서 표시한 바와 같이 적당한 분해함수 d_2^1를 구하여 함수 f_1만 표현하면 아래와 같이 최적의 해를 구할 수 있다.

$$d_1^2(x) = x_1x_3 + x_2x_3 \text{ (공동 사용)}$$
$$d_2^2(x) = x_1x_2x_3 + x_1'x_2'x_3'$$
$$d_2^1(x) = x_1'x_2x_3 + x_1x_2x_3' + x_1x_2'x_3$$
$$f_1(x, y) = g_1(d_1^2(x_1, x_2, x_3), d_2^1(x_1, x_2, x_3), y_1, y_2)$$
$$f_2(x, y) = g_2(d_1^2(x_1, x_2, x_3), d_2^2(x_1, x_2, x_3), y_1, y_2)$$

만약 5)항의 조건 i)가 만족하지 못하면 조건 ii)로 넘어간다. 이렇게 조건이 만족할 때까지 루프를 돈다. 위의 예제에서는 조건 i)와 iii)는 만족하고, 조건 ii)는 f_1을 표현할 d_2^1를 찾을 수 없으므로 공통분해 함수 조건을 만족 하지 않는다.



(그림 7) 선택 안된 출력 함수 분해 찾기
(Fig. 7) Look for Decomposition Function of Unsel

4. 실험 결과

4.1 Roth-Karp 분해확장 결과

〈표 3〉에서 첫번째 횡은 테스트 회로와 입출력을 나타내며, 두 번째 횡은 공통 인수 분해 결과이며, 세 번째 횡은 Roth-Karp 분해 결과이고, 네 번째 횡은 다출력 함수를 위한 Roth-Karp 분해 확장 결과이다. 세 경우 모두 SIS-pga의 표준 스크립트 12단계 중 처음 3단계만 수행하고 〈표 3〉에 나타난 3가지 방법을 수행한 것이다. 〈표 3〉에서 나타나듯이 Roth-Karp 분

기술에 매핑하는 것이다. 만약 한 회로가 FPGA에서 널리 사용하고 있는 m 입력 TLU로 제한되어진다면, Ashenhurst[5], Roth-Karp[6]에 의해 이론화된 종래의 함수 분해방법을 많이 사용하였다[7]. 이 방법은 한 출력 함수와 입력이 제한 변수(x)와 자유변수(y)로 분할하여 주어진다면, 기능적 분해로 함수 d(x)와 g(z, y)를 찾아서 f(x, y)=g(d(x), y)를 만든다. 또한 제한 변수의 개수가 m으로 정해지면 각 분해함수 d_i는 하나의 TLU로 구현이 가능하다. 이렇게 함수 분해를 위해 Binary Decision Diagram(BDD)를 이용한 효율적인 기능적 함수 분해 방법[8]이 제안되었다. 또한 x-꼭지점의 최적화된 엔코딩으로 단순한 g 함수를 만들어 면적을 줄이는 방법[9]도 연구되었다.

그러나 이러한 방법을 다출력 함수에 적용하려면 각 출력을 개별적으로 적용해야하므로 여러 출력간의 공통 부함수를 찾을 수가 없다. 그래서 다출력 함수의 기능적 분해 방법은 지금까지 만족 할만한 결과를 보여주지 못하고 있다. 종래의 분리된 2단계 로직 합성 방법에 다출력 함수의 기능적 분해방법을 첨가할 수 있다면 TLU형 FPGA 구조에 매우 효과적일 것이다.

2출력 함수분해 방법은 Karp[6]에 의해서 처음으로 제안되었다. 이것은 오직 2출력 함수에서만 가능하지만 하나의 동등 클래스에 여러 개의 코드를 사용할 수 있는 장점을 갖고 있다. 만약 한 동등 클래스에 한 코드만 할당하면 다출력 함수의 모든 공동 분해 함수를 찾을 수가 없다. 한 동등 클래스에 여러 개 코드를 할당하는 방법으로서 EV(Edge-Valued)BDD를 이용한 기능 분해 방법[10]이 제안되었다. EVBDD는 정수 함수(Integer Function)를 표현하고 처리 조작하는 DAG(Directed Acyclic Graph)이다. 이것은 OBDD가 불리안 함수를 처리조작 하는 것과 같은 효과를 가진다. EVBDD는 불리안 함수와 산술함수 사이의 동등함을 보여주는 로직 검증(Verification) 방법으로 사용되어진다.

B. Wurth등에 의해서 제안된 preferable 분해 함수 개념을 사용한 방법[11]도 있다. 이 방법은 많은 분해 함수를 모두 처리하는 것이 아니라 그 중에서 유용한 것들만 가려낸다. 여기에서는 유용한 분해 함수만 찾아내는 세련되고 효율적인 개념(Preferable Decomposition)을 도입하였다.

본 고에서 제안하는 TLU형 FPGA를 위한 다출력 함수 로직 합성 방법은 기능적 분해 방법을 사용하였다. 첫번째는 한 출력에 적용한 Roth-Karp 알고리즘을 다출력에 응용할 수 있도록 확장하였으며, 두 번째는 분해과정에서 공통 분해 함수를 찾는 효과적인 알고리즘을 제안한다.

3. 다출력 함수 분해 알고리즘

(정의) 다출력 함수 로직 분해 : 한 출력 함수 f가 주어지고 이 함수의 n개 입력변수가 제한변수 $X=\{x_1,...,x_b\}$와 자유변수 $Y=\{y_1,...,y_{n-b}\}$로 분리 되어 주어진다면 기능적 함수 분해는 분해함수 $d_1,...d_c$와 합성함수 g를 결정하여 $f(x, y)=g(d_1(x_1,...,x_b),...,d_c(x_1,...,x_b), y_1,...,y_{n-b})$ (단 $c<b$)로 표현한다. 다출력 함수 $F=\{f_1,...,f_m\}$가 주어지고 이 함수들의 기능적 함수 분해는 분해함수 $D_1,...,_q$와 합성함수 $G=\{g_1,...,g_m\}$를 결정하여 $F(x, y)=G(D_1(x_1,...,x_b),...,D_q(x_1,...,x_b), y_1,...,y_{n-b})$로 표현하는 것을 다출력 함수 로직 분해라고 한다.

3.1 다출력 함수를 위한 Roth-Karp 분해 확장[13]

다단계 논리구조 합성의 전처리 과정과 TLU로의 분해 변환에서 사용한 대수학적 방법의 단점 보완을 주목적으로 SIS-pga[13]에서는 두 번의 불리안 분해 변환을 시도한다. 즉 지금까지의 모든 논리합성 결과를 무시하고 네트웍을 2단계로 완전히 펼친 후 Roth-Karp 방법과 공통 인수분해 방법의 두 가지 불리안 분해변환만으로 각각 설계하여 대수학적 방법에 의한 결과를 포함한 세 가지 결과 중에서 제일 좋은 결과를 선택한다. 이러한 불리안 방법들은 대칭 함수를 포함한 특별한 성질의 함수에 대해서 매우 효과적이다. 그러나, 불리안 방법은 대수학적 방법에 비해 기본적으로 계산시간이 많이 요구되기 때문에 주입력의 개수가 일정 규모 이하인 회로에 대해서만 적용한다.

그러나 SIS-pga에서 채택한 방법은 각각의 노드에 대해 개별적으로 분해 변환을 행하기 때문에, 공통된 중간 노드의 생성을 보장하지 못한다. 이러한 단점을 보완하기 위하여 본 연구에서 제안한 다출력 Roth-Karp 분해변환은 Roth-Karp 방법을 다출력 함수에 적용할 수 있도록 다음과 같이 확장한 것이다.

Given a set of functions $f_i(X, Y_i)$'s and
d_{max} (d_i : Decomposition function)
Find a minimum set of d-functions $d_1(X), d_2(X),...,$
$d_j(X)$ with $j = < max$
and the corresponding disjoint decomposition for each f_i
$f_i(X, Y_i) = f_i(d_1(X), d_2(X),..., d_j(X), Y_i)$

여기에서 X는 함수 f_i들 간에 공통된 입력변수 또는 이의 부분집합이다. 적절한 또는 최적의 제한 집합 X를 미리 예측하기는 어려우나, 작은 회로에 대해서는 여러 가지 경우를 시도해 볼 수 있다.

```
Mul_out_decomp{
    N = {n an infeasible node}
    P = {(N',F') N'⊆ N, F'= fanins common to N'}
    repeat
        pick a maximal p=(N',F') (P with max N' );
        for (each subset F ⊆ F', F =< 5){
            decompose N' by F with d_max < F
            N' = (d1(F),..., d_max(F), );
            if (possible){
                change the network;
                break;
            } /* end if */
        } /* end for */
    until (a decomposition found);
} /* end mul_out_decomp */
Repeat this when a decomposition is found.
if (still not feasible) do split_network().
```

(그림 2) Roth-Karp 분해 확장
(Fig. 2) Roth-Karp Decomposition Extended

(그림 2)은 Roth-Karp 알고리즘 확장 과정을 설명한 것이다. 먼저 매핑 불가능한 노드 중에서 공통의 입력신호를 갖는 함수들을 찾는다. 관련 함수가 많고 공통변수의 수가 많은 집합부터 분해변환의 존재여부를 조사하며, 만일 존재하면 이 변환에 의한 이득을 계산하여 임시 저장하고, 추후에 발견되는 또 다른 분해변환과 비교하여 최적의 것을 찾을 수 있도록 한다. 일단 하나의 분해변환이 선택되어 수행됨으로써 네트웍이 변화되면 매핑 불가능성 등 모든 상황이 변화되므로 다시 공통변수를 갖는 매핑 불가능한 노드 집합을 조사하여야 한다. 이러한 과정을 바람직한 Roth-Karp 분해 변환이 발견되지 않을 때까지 반복 수행하고, 마지막으로 네트웍이 여전히 매핑 불가능할 때에는 적절한 방법을 통해 매핑 가능한 네트웍으로 변환한다. 이와 같은 알고리즘에서 중요한 것은 최적의 분해 변환을 찾는 과정의 계산속도를 줄이기 위한 적절한 데이타 구조, 정보 변환 방법 및 분해변환을 찾는 과정의 필요성 유무에 대한 판단이다. 큰 회로에 대한 계산속도를 제한하기 위하여 주어지는 선택 조건은 다음과 같다.

- -. Test all maximal (common variables, nodes) sets, or
  Test only the largest (common variables, nodes) sets.
  For each selected pair
- -. Test all bound sets to pick the best, or
  Take just the one possible decomposition for each bound size, or
  Take just the first possible bound set

분해변환에 의한 이득은 그 변환에 의한 엣지의 감소로써, gain = [common variables − (# of d-functions)] *(# of nodes)이다.

중간 노드를 찾는 과정에서의 불필요한 계산을 줄이기 위해서, 현재까지 발견된 최적의 분해변환(# of nodes, # of d-functions)으로부터, 현재 조사 중인 제한 집합에서 d-함수의 수에 대한 상한 제한을 계산하고, 이러한 제약조건을 만족하는 분해변환의 존재여부만을 조사토록 한다.

3.2 다출력 함수의 기능적 분해 방법

본 고에서 제안하는 TLU형 FPGA를 위한 다출력 함수 로직 합성 방법은 기능적 분해 방법을 사용하였다. 기능적 분해과정에서 공통 분해 함수를 찾는 효과적인 알고리즘을 제안한다. 다출력 함수의 공통적인 분해함수를 찾는 새로운 알고리즘을 (그림 3)에 나타내었다.

다음은 (그림 3) 다출력 함수의 기능적 분해 알고리즘에 대한 공통 분해함수를 찾는 방법을 알고리즘의 번호 순서대로 설명한다.

(1) 다출력 함수의 각 출력마다 모든 동등 클래스를 찾는다.

함수 f가 주어지고, 제한변수 X가 정해지면, 함수

해 확장 결과에서 CLB와 네트 수에서 2% 정도의 비용함수 향상을 보였으며, 지연시간에 관계되는 레벨 수는 약간 증가하였다.

〈표 3〉 Roth-Karp 분해 확장 결과
〈Table 3〉 Result of Roth-Karp Decomposition Extended

Circuits (Ins, OUTs)	Cofactoring CLBs	NETs	LEVs	Roth-Karp CLBs	NETs	LEVs	Multiple-Outputs CLBs	NETs	LEVs
z4ml (7,4)	13	63	3	8	38	2	10	46	3
misex1 (8,7)	18	85	3	14	63	2	10	45	3
5xp1 (7,10)	21	101	3	17	80	2	11	51	3
9symml(9,1)	24	118	5	7	35	3	6	20	3
rd84 (8,4)	37	184	4	12	58	3	7	28	3
rd73 (7,3)	17	83	3	7	34	2	5	21	2
f51m (8,8)	23	110	4	20	98	4	10	45	4
alu2 (10,6)	96	449	5	83	378	17	78	374	16
inc (7,9)	23	112	3	27	128	2	20	99	3
con1 (7,2)	4	18	2	4	16	2	3	13	2
Total	**276**	**1323**	**35**	**199**	**928**	**39**	**160**	**742**	**42**

4.2 다출력 함수 기능적 분해 결과

본 연구에서는 입력이 5개인 TLU로 만들어진 CLB를 가지는 Xilinx XC3000 구조를 목표로 하였다. 실험결과는 제 3.2절에서 설명한 TLU형 FPGA를 위한 다출력 함수의 변환 알고리즘을 사용하여 CLB 개수, 네트의 수를 동시에 최적화 하도록 하였다. 실험에 사용된 회로는 MCNC 벤치마크 회로로써, 기존의 알고리즘과의 비교를 위하여 각 문헌에서 사용한 회로 중 표준 벤치 마크에 있는 것을 모두 취합한 것이며, 최종 실험 결과는 〈표 4〉와 같다.

〈표 4〉의 첫번째 횡은 테스트 회로와 입출력의 개수를 나타내며, 두번째 횡은 다출력 함수를 각 출력마다 계산한 값이다. 세 번째 횡은 참고문헌[11]에서 인용한 IMODEC의 결과이며, 네 번째 횡은 참고문헌[12]에서 인용한 FGMap의 결과이며, 마지막 다섯번째 횡은 제안된 알고리즘을 이용한 결과이다. 면적을 평가할 CLB의 개수와, 배선에 큰 영향을 주는 네트의 개수, 성능의 지연시간과 관계되는 레벨의 수, SUN 스파크20에서 수행한 CPU 시간을 초단위로 표시하였다.

〈표 4〉의 실험 결과에서 보듯이 다출력 함수를 각 출력마다 계산한 것과 다출력 함수를 고려했을 때를 비교하면 CLB의 개수가 약 34% 향상됨을 볼 수 있다. 또한 IMODEC과 비교했을 때는 CLB의 개수가 2% 향상 되었으며, FGMap보다는 약 18% 향상 되었다. 면적을 평가하는 CLB의 개수 뿐만아니라 네트의 수나 레벨의 수, 수행시간 등도 매우 만족할 만한 결과를 보여주고 있다.

〈표 4〉 다출력 함수 기능적 분해 결과
〈Table 4〉 Functional Decomposition Result of Multiple Output Function

Circuits (IN,OUT)	Single	IMO DEC	FG-Map	CNU_MO CLB	NET	LEV	CPU
5xp1(7,10)	15	9	15	9	48	3	1.2
9symml(9,1)	7	7	7	7	20	3	0.8
alu2(10,4)	47	46	53	44	214	13	8.4
apex7(49,37)	61	41	47	41	193	6	7.2
clip(9,5)	19	12	20	12	78	5	3.3
count(35,16)	35	26	24	26	82	8	2.4
f51m (8,8)	13	8	11	8	35	4	1.4
misex1(8,7)	11	9	8	8	36	3	1.3
misex2(25,18)	34	21	21	21	88	4	2.4
rd73 (7, 3)	7	5	7	5	21	2	0.7
rd84 (8 ,4)	11	8	12	7	28	3	1.0
sao2(10,4)	24	17	27	16	66	6	2.0
vg2(25,8)	64	19	23	19	93	5	2.2
z4ml(7, 4)	4	4	5	4	17	3	0.6
Total	**352**	**232**	**280**	**227**			

5. 결 론

본 연구에서는 최근에 관심을 모으고 있는 Table Look-Up형의 FPGA를 위한 다출력 함수 로직 합성 알고리즘에 대해 설명하였다. 본 고에서 제안하는 TLU형 FPGA를 위한 다출력 함수 로직 합성 방법은 기능적 분해 방법을 사용했다. 기술 매핑의 최적화 대상은 CLB 개수를 고려했으며, 벤치마크 테스트를 통한 일반적인 회로에 적용성 검증, 기존 알고리즘과의 성능 비교 및 개선에 대해 연구하였다. 논리 설계 합성기 구성 과정에서 새로운 알고리즘을 구현하여 CLB 개수, 네트의 수를 최적화하도록 하였으며, 기

존 알고리즘과 성능을 비교 하였다. 성능 비교 결과 다출력 함수 분해 방법이 매우 효과적이며, 4장의 실험결과에서 보듯이 기존의 다출력 함수 분해 방법과 비교하면 CLB의 개수, 네트의 수, 성능과 수행 시간에서 매우 만족할 만한 결과를 얻었다.

아울러 앞으로 계속되어야 할 연구과제는 다음과 같다.

(1) 지연 시간을 고려한 변환: 회로의 일부 변경이 CLB 및 NET의 수, 즉 면적에 미치는 영향은 상대적으로 국지적이고 비교적 쉽게 정량적 판단이 가능하나, 임계경로의 지연 시간에 미치는 영향은 간단하지 않다. 그렇지만, 고성능을 요구하는 최근의 설계요구조건을 고려할 때 지연시간을 고려한 설계 기법의 개발이 필요하다.

(2) 입력변수 분할 알고리즘 개발: 지금까지의 모든 연구는 입력변수가 제한변수와 자유 변수로 분할된 경우에만 취급하였다. 그러나 입력변수의 분할방법에 따라 면적 및 성능에 큰 영향을 주므로 이에 대한 알고리즘도 연구가 필요하다.

참 고 문 헌

[1] W. Carter et. al., "A User Programmable Reconfigurable Gate Array", Proc. 1986 CICC, 1986, pp. 233-235.

[2] The Programmable Gate Array Data Book, Xilinx Co., 1996.

[3] R. K. Brayton, et al, "MIS: A multiple-level logic optimization system", in IEEE Trans. CAD ,pp 1063-1081,1987.

[4] B. G. Kim and D. L. Dietmeyer, "Multilevel Logic Synthesis with Extended Arrays", IEEE Trans. on CAD, Vol. 11, No. 2 1992.

[5] R.L. Ashenhurst, "The Decomposition of Switching Function", Ann. Computation Lab. of Harvard Univ. vol. 29, pp. 74-116, 1959.

[6] J. P. Roth and R. M. Karp, "Minimization Over Boolean Graphs", IBM Journal, April 1962, pp. 227-238.

[7] R.Murgai, et al, "Logic Synthesis for Programmable Gate Arrays", 27th DAC, 1990, pp. 620-625.

[8] Y.T. Lai et al, "BDD Based Decomposition of Logic Functions with Application to FPGA Synthesis", 30th DAC, 1993, pp. 642-647.

[9] R. Murgai, et al, "Optimum Functional Decomposition Using Encoding", 31th DAC, 1994, pp. 408-414.

[10] Y.T. Lai et al, "EVBDD-Based Algorithm for Integer Linear Programming, Spectrral Transformation, and Function Decomposition", IEEE Trans. On CAD, Vol. 13, no. 8, 1994.

[11] B. Wurth, et al, "Functional Multiple-Output Decomposition: Theory and an Implicit Algorithm", 32th DAC, 1995, pp. 54-59.

[12] Y.T. Lai, et al, "FGMap: A Technology Mapping Algorithm for Look-Up Table Type FPGAs Based on Function Graphs", International Workshop on Logic Synthesis IWLS, May, 1995, pp. 9b1-9b4.

[13] R. K. Brayton, et al, "SIS: A System for Sequential Circuit Synthesis", Technical Memo No. UCB/ERL M92/41, 1992.

박 장 현

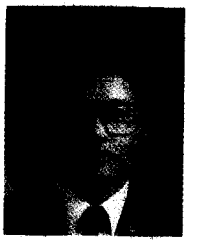
1983년 서강대학교 전자공학과 (공학사)

1985년 AIT 컴퓨터공학과 (공학석사)

1997년 충남대학교 전자공학과 (공학박사)

1985년~현재 한국전자통신연구원 선임연구원

관심분야: VLSI 설계, 설계 자동화



김 보 관

1976년 서울대학교 전자공학과 (공학사)

1978년 한국과학기술원 전기 및 전자공학과 (공학석사)

1989년 Univ. of Wisconsin Madison 전기 및 컴퓨터공학과(공학박사)

1980년~1991년 금오공대 전자공학과 조교수

1991년~현재 충남대학교 전자공학과 부교수

관심분야: Logic Synthesis, Hardware/Software Codesign, 설계 자동화