論文93-30B-9-1

이단계 Reed-Muller 회로의 최소화에 관한 새로운 접근

(A New Approach to the Minimization of Two-level Reed-Muller Circuits)

張峻榮*, 李貴相*

(June Young Chang and Guee Sang Lee)

要約

본 논문에서는 이단계 리드뮬러회로의 최소화에 관한 새로운 방법을 제시한다. 리드뮬러회로를 최소화하기 위해 두개의 큐브를 묶는 기존의 Xlinking 방법에 비해 본 논문에서 제안한 최소화 방법은 주어진 함수의 ON-set를 커버할 때까지 한번에 하나씩 큐브를 선택해 나가는 방법이다. 적절한 큐브를 선택하기 위해 간단한 휴리스틱이 사용되었다. 즉, 가장 큰 큐브부터 시작하여 가장 작은 큐브까지 차례로 시도하면서 남아있는 항의 수를 줄이는 큐브를 선택한다. 일단 어떤 큐브가 선택되어지면 새로운 큐브를 선택할 때, 이미 선택된 큐브는 고려하지 않기 때문에 반복적인 최소화 과정이 필요하는 기존의 방법보다 시간이 적게 걸린다. 실험결과 다른 문헌에 나타난 가장 좋은 결과보다 여러가지 경우에서 개선된 결과를 보여주었다.

Abstract

In this paper, a new approach to the minimization of two-level Reed-Muller circuits is presented. In contrast to the previous method of using Xlinking operations to join two cubes for minimization, Cube selection method tries to select cubes one at a time until they cover the ON-set of the given function. A simple heuristic for selecting appropriate cubes is presented. In this heuristic, simply all cubes from the largest to the smallest are tried and whenever they decrease the number of remaining terms they are accepted. Since cubes once selected are not considered for a new selection, our method takes less time than other methods that need repetitive optimization process. The experimental results turned out to be improved in many cases compared to the best results in the literature.

Ⅰ. 서론

VLSI 설계자동화중 논리함수를 최적화하여 레이아웃 합성단계의 입력을 생성하는 논리합성에 관한 연구가 활발히 추진되고 있다.[1,5,7,8] 논리합성분야에는 생성되어지는 회로의 구조에 따라 이단계회로(Two-level circuits)와 다단계회로(Multi-level circuits)로 구분할 수 있고, 합성하고자 하는 회로가 AND/OR 회로인지 AND/XOR 회로인지에 따라 다시 구분할 수 있다. 그리고 AND/OR회로 논리합성의 경우 EXPRESSO[7]와 MISII[8] 등과 같은 논리합성도구 개발을 비롯한 많은 진전이 있었다. 그러나 AND/XOR 합성의 경우, 이제 본격적인 연구의 시작단계

* 正會員, 全南大學校 電算學科
(Dept of Computer Science Chonnam Nat'l Univ.)
接受日字 1993年 1月 8日

라 할 수 있다.[3,6] 본 논문에서는 이단계 리드뮬러회로의 최소화에 대해서 살펴보기로 한다.

본 논문에서 사용하는 리드뮬러회로라 함은 일반적으로 AND/XOR 회로를 말하고, XOR게이트를 이용한 회로의 표현에 리드뮬러형식을 이용한다. 리드뮬러형식은 논리함수의 리터럴(literals)에 따라서 세가지 형태로 나누어진다.[12] 첫번째로, 양극성을 갖는 리드뮬러형식(PRM : Positive-polarity Reed Muller forms)은 식(1)과 같이 표현되고 사용되는 모든 리터럴 x_i는 양극성만을 갖는다.

$$f(x_1,x_2,\cdots,x_n) = \{a_0 \oplus a_1x_1 \oplus a_2x_2 \oplus \ldots \oplus a_{2n-1}x_1x_2\ldots x_n\}, a_i \in \{0,1\} \quad (1)$$

두번째로, 음극성을 갖는 리드뮬러형식(NRM : Negative-polarity Reed Muller forms)은 식(2)와 같이 표현되고 사용되는 모든 리터럴 x_i는 음극성만 갖는다.

$$f(x_1,x_2,\cdots,x_n) = \{a_0 \oplus a_1x_1' \oplus a_2x_2' \oplus \ldots \oplus a_{2n-1}x_1'x_2'\ldots x_n'\}, a_i \in \{0,1\} \quad (2)$$

세번째로, 일반화된 리드뮬러형식(Generalized Reed Muller forms)은 양극성과 음극성의 일반적인 경우로 하나의 표현식으로 두가지 경우에 모두 이용되고 식(3)과 같이 표현된다. 따라서 식(3)의 리터럴이 모두 양극성만을 가질때 PRM형식이라 하고, 리터럴이 음극성만 가질때 NRM형식이라 한다.

$$f(x_1,x_2,\cdots,x_n) = \{a_0 \oplus a_1\hat{x}_1 \oplus a_2\hat{x}_2 \oplus \ldots \oplus a_{2n-1}\hat{x}_1\hat{x}_2\ldots\hat{x}_n\}, a_i \in \{0,1\} \quad (3)$$
$$\hat{x}_i = x_i \text{또는} \ x_i'$$

식(3)의 모든 리터럴이 두가지의 극성중 하나의 극성만으로 구성되어지면 고정극성(Fixed-polarity)이라 하고, 각각의 리터럴이 동시에 양극성과 음극성을 갖는 형식을 혼합-극성 (Mixed-polarity)이라 한다. 일반적으로 혼합-극성 일반화된 리드뮬러형식(GRM : Mixed-polarity Generalized Reed Muller forms)이 최소의 면적으로 회로 구현이 가능하므로 더 효율적인 것으로 평가되고 있다.[12] 본 논문에서는 혼합-극성 GRM형식으로 표현된 리드뮬러회로를 고려하기로 한다. 최근들어, 이러한 리드뮬러형식을 이용하여 논리함수를 설계하고 구현하는데 많은 관심이 기울여지고 있다.[1,3,4,6]

리드뮬러형식은 논리표현의 간결성과 회로를 쉽게 테스트 할 수 있는 장점에도 불구하고 자주 언급되지 않았다. 그 이유는 리드뮬러형식에 의해 구현된 AND/XOR회로가 상대적으로 곱의 합(SOP: Sum of Product)형식에 의해 실현된 AND/OR회로에 비해 XOR게이트의 특성상 비용이 많이 들기 때문이었다. 그러나 최근에 VLSI 기술의 발달로 인해 XOR게이트가 다른 게이트와 비교해서 가격과 속도면에서 동등한 구현이 가능하게 되었다.[1]

리드뮬러형식을 이용한 AND/XOR구현의 주된 응용은 산술, 통신회로, 에러 제어를 위한 코딩구조, 동기화 시스템, 테스팅등과 같은 응용분야에서 찾아볼 수 있다.[4,6] 이러한 응용분야의 회로들은 AND/XOR회로로 구현하는것이 AND/OR회로로 구현하는 것보다 더 적은 회로들로 구현이 가능하기 때문에 더욱 경제적이다. 예를 들면, 패리티 함수(Parity function)는 AND/OR 회로로 실현할 경우 2^{n-1}의 항이 필요한 반면 AND/XOR회로로 실현할 경우 n개의 항만 필요하다.[3]

따라서, 논리함수의 간결한 표현과 높은 회로 검증도는 리드뮬러회로의 가장 중요한 장점들로 지적할 수 있다. XOR게이트로 테스트 가능한 회로를 설계하면 가능한 작은 테스트의 집합으로 회로의 결함을 쉽게 검출할 수 있는 장점이 있다. 동일한 회로를 구현하는데 있어서 XOR게이트를 이용하는 것이 회로의 검증도를 향상시키는데 유리하다는 것이 여러 논문에서 제시되었다.[1,2,6,9,14,15]

리드뮬러형식을 사용하는데 있어서 가장 먼저 고려해야 할 사항은 XOR게이트를 이용하는 회로의 논리함수를 어떻게 간결하게 표현하느냐 하는 것이다. 논리함수의 간결성은 리드뮬러회로의 논리합성을 이용하여 항과 리터럴의 수를 최소화하는 것을 말한다. 현재 대다수의 논리합성도구는 논리함수의 AND/OR 논리합성에만 이용되어 지고, AND/XOR처리에 필요한 논리합성도구가 빈약한 형편이기 때문에 AND/XOR회로에서 잘 동작하지 않는다.[1] 이단계 논리합성도구인 EXPRESSO[7] 와 MISII[8] 나 BOLD[5] 와 같은 다단계 논리합성도구에도 AND/XOR논리합성을 위한 구조를 포함하고 있지 않다.[7] 본 논문에서는 기존의 AND/XOR회로의 최소화 방법을 소개하고, 리드뮬러형식을 이용한 AND/XOR회로의 최소화 방법에 대한 새로운 접근방법을 제시한다.

SOP형식의 최소화 방법으로는 카르노 맵을 이용하는 방법이나 퀸-맥클러스키방법이 많이 이용되었다.[6] 그러므로 이 방법을 리드뮬러형식의 최소화 방법으로 확장하여 사용하는 것은 쉽지 않다. 최근의 리

드뮬러형식 최소화 방법에 관한 연구중에서 Perkowski [5] 와 Saul [3] 의 연구가 대표적이라 할 수 있다. Perkowski [6] 의 Xlinking 방법은 두개의 큐브(cube)를 묶는 연산 과정이 자주 반복됨으로 복잡할 뿐만 아니라 처음에 어떤 큐브를 선택하는가에 따라서 결과가 달라지는 문제점을 내포하고 있다. Saul [3] 은 Xlinking 방법의 큐브선택의 문제점을 개선하기 위해 Unlink방법을 제안 하였는데, Unlink방법은 큐브의 수를 늘어나게 하는 민텀(minterms)들을 원상태로 복귀시키는 방법이다. 즉, 초기상태의 큐브 수보다 연결 과정중에서 생성된 큐브 수가 많아지게 되면 원상태로 복귀시키는 방법이다. Xlinking 방법에 부가적인 Unlink루틴이 추가 되므로 민텀들을 커버하는 적절한 큐브들이 선택되어지고 따라서 처리결과의 향상을 가져왔다. 그러나 최적의 큐브를 선택하기위한 과정이 반복되므로 처리시간이 증가하는 문제점을 내포하고 있다.

본 논문에서는 혼합-극성 일반화된 리드뮬러회로(GRM : Mixed-polarity Generalized Reed Muller circuits)의 최소화에 관한 새로운 접근방법을 제시한다. 이 새로운 접근방법을 본 논문에서는 Cube selection 방법이라 부르고자 한다. 이 방법은 기존의 *Xlinking* 방법과는 전혀 다른 새로운 방법으로 서로 다른 큐브가 한번에 하나씩만 선택되어지기 때문에 불필요한 최소화 연산을 계속 반복할 필요가 없는 장점을 가지고 있다.

다음 Ⅱ장에서는 리드뮬러형식의 최소화에 관한 기존의 연구를 살펴보고, Ⅲ장에서는 새로운 리드뮬러회로의 최소화 접근 방법인 Cube selection 방법에 대해 기술한다. Ⅳ장에서는 간단한 예제회로들을 통해서 본 논문에 제시된 방법을 기존의 결과와 비교하고, benchmark회로를 이용한 실험 결과를 제시한다.

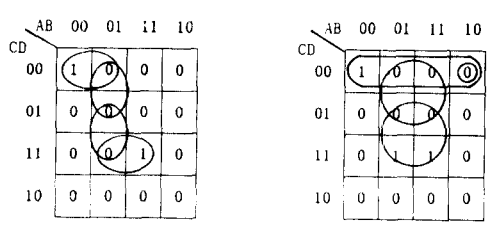
Ⅱ. Xlinking연산에 의한 최소화 방법

Perkowski [6] 는 논리함수를 구성하는 큐브의 수를 최소화하기위해 두개의 큐브를 묶는 과정인 *Xlinking* 방법을 제안하였다. 여기서 큐브는 논리함수를 구성하는 민텀 또는 곱항(product terms)을 말하고, 논리함수를 표현하기위해 카르노 맵(karnaugh map)을 이용하고, 연결과정은 두개의 큐브를 묶어서 논리함수를 최소화하는 과정이다.

리드뮬러형식을 최소화하는데 있어 논리함수의 리드뮬러형식은 일반적인 SOP형식의 특성과 매우 다르다. SOP형식에서는 주어진 함수를 카르노 맵으로 표현하여 2' 개의 인접된 ON-set를 묶음으로 큐브를 최소화하는 방법인데 반해 리드뮬러형식의 중요한 특징은 함수의 ON-set을 실현하는 큐브의 집합을 만들 때 인접된 OFF-set까지 확장하여 큐브를 만들 수 있다는 점이다. 왜냐하면 부울대수의 XOR(Exclusive-OR)성질에 의해 함수의 OFF-set를 짝수번 커버하면(A⊕A= 0) 원래의 함수에 영향을 미치지 않기 때문이다. 그러므로 최소화 과정을 유도하는 큐브는 함수의 ON-set뿐만 아니라 필요한 경우 OFF-set까지 포함할 수 있다.

*Xlinking*방법은 Primary *Xlinking*과 Secondary *xlinking*으로 구성되어 있다. 연결하고자 하는 큐브의 집합중 리터럴의 위치가 같은 큐브중에서, Primary *xlinking*은 리터럴의 수가 같은 두개의 큐브에 적용하는 규칙이며, Secondary xlinking은 리터럴의 수가 1개 다른 두개의 큐브를 위한 규칙이다. Perkowski [6] 의 논문에 나타난 그림 1은 Primary xlinking과 Secondary xlinking이 어떻게 적용되는가를 보여 준다. 그림 1(a)에서 두개의 큐브 [A 'B' C 'D'] 와 [ABCD] 는 각각 [A 'C' D '] 와 [BCD] 로 큐브를 만들고, 이들은 [A' BC '] 과 [A' BD] 에 의해 연결된다. [A 'B' C 'D'] ⊕ [ABCD] => [A 'C' D '] ⊕ [A' BC '] ⊕ [A' BD] ⊕ [BCD] 와 같이 수행된다. 그림 1(b)는 Secondary *xlinking*을 설명하는데 두개의 큐브 [A 'B' C 'D'] 와 [BCD] 는 [C 'D'] 와 [BD] 로 큐브를 만들고 이들을 [BC '] 과 [AB' C 'D'] 으로 연결한다. [A 'B' C 'D'] ⊕ [BCD] => [C 'D'] ⊕ [BC '] ⊕ [AB' C 'D'] ⊕ [BD] 와 같이 큐브가 생성된다.

Perkowski [6] 는 큐브의 수를 최소화하기 위해 *Xlinking* 방법을 더이상 적용할 수 없을 때까지 반복적으로 적용하였다. *Xlinking* 방법은 선택되어지



(a) Primary *xlinking* of [a 'b' c 'd'] and [abcd]

(b) Secondary *xlinking* of [a 'b' c 'd'] and [bcd]

그림 1. Xlinking 예제
Fig. 1. Xlinking example.

는 큐브에 따라 결과가 달라지기 때문에 큐브를 선택할 때 주의해야 한다. 다시말해, 일련의 큐브들이 최적의 과정으로 선택되지 않으면 최적의 해를 구할 수 없기 때문이다. Saul [3] 은 큐브의 수를 최소화하기 위해 좀 더 좋은 큐브들을 선택하는 방법을 제안함으로서 *Xlinking* 방법을 개선하였고, *Xlinking* 방법을 다중출력함수를 효율적으로 처리할 수 있는 방법으로 확장하였다. 이 후에 Saul은 *Xlinking* 방법을 다단계 리드뮬러 합성에 관한 연구로 확장하였다. [4]

Ⅲ. Cube selection 방법

본 논문에서는 Perkowski [6] 의 *Xlinking* 방법과는 다른 새로운 최소화 방법인 Cube selection 방법을 제시한다. *Xlinking* 방법은 큐브의 집합을 변화시키면서 최소화를 하는 반면, Cube selection 방법은 처음부터 ON-set의 크기가 가장 빨리 줄어들도록 큐브를 선택하는 방법이다. Cube selection 방법은 여러 가지 휴리스틱을 적용할 수 있으나, 본 논문에서는 가장 큰 큐브부터 가장 작은 큐브까지 차례대로 조건에 맞는 적절한 큐브를 선택하는 휴리스틱을 이용한다. *Xlinking* 방법은 처음에 어떤 큐브를 선택하느냐에 따라 결과가 크게 달라질 수 있고, 큐브의 수를 최소화하는데 필요한 큐브가 멀리 떨어져 있는 경우 불필요한 항들이 늘어나는 문제점을 가지고 있다.

Xlinking 최소화 방법에 비해 Cube selection 방법은 불필요한 큐브들이 늘어나는 문제점이 없으며, Saul [3] 의 Unlink방법에서 잘못 선택된 큐브를 원상태로 복구하는 반복과정을 제거할 수 있고, 여러가지 경우의 논리함수에 대해 최소의 큐브 집합을 쉽게 찾을 수 있는 매우 간단한 방법이다. Cube selection 방법은 가장 큰 큐브부터 시작해서 가장 작은 큐브까지 차례대로 이득(gain)을 계산하여 선택된 큐브에 보수를 취하고 다시 선택되지 않은 큐브를 선택해 감으로 결과적으로 최소화된 큐브를 찾아내는 방법이다. 다음은 Cube selection 방법에 의해서 최소 큐브를 선택하는 알고리즘을 설명한다.

1. 용어의 정의

논리함수가 주어지면, 함수를 카르노맵(karnaugh map)으로 표현한다. 논리함수의 크기, 즉 논리함수의 ON-set의 크기는 논리함수의 ON-set의 민텀의 최대 갯수이다. 논리함수에 관한 큐브의 이득은 함수를 커버하기 위해 선택된 큐브의 ON-set의 수에서 OFF-set의 수를 감산한 결과이다. 이것을 식으로 표현하면 다음과 같다.

이득(C) = 크기(f)- 크기(f ⊕ C)

2. Cube Selection 알고리즘

1) 아직 시도되지 않은 큐브 중에서 가장 큰 큐브를 선택
2) 선택된 큐브의 이득을 계산
3) 이득이 양수이면 큐브 를 ACCEPT 하고, f = f ⊕ C 0 이거나, 음수이면 REJECT
4) ON-set의 수가 0 이면 끝내고, 그렇지 않으면 1)의 과정부터 반복

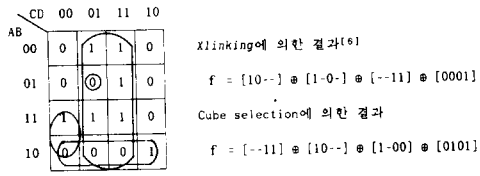


그림 2. K-map과 함수 의 최소화 결과

Fig. 2. K-map and optimization results of function.

(a) 선택된 큐브

(b) f=f ⊕ C에 의해 변해가는 함수 맵

그림 3. Cube selection에 의한 리드뮬러 최소화의 예제

Fig 3. Example of Reed Muller minimization by Cube selection.

3. Cube Selection 방법의 설명

Perkowski [6] 의 논문에 나타난 예제를 통해서 리드뮬러 형식을 가진 회로를 최소화하기 위해 큐브를 선택하는 과정은 다음과 같다. 먼저, 가장 큰 큐브인, 1 혹은 [……] 를 선택한다. 큐브 [……] 의 이득은 크기(ON-set)-크기(OFF-set) 이므로 7-9=-2, 따라서 Cube selection 알고리즘에 의해 선택되지 않는다. 다음은 큐브의 크기가 8인 […1] 을 선택하고,

이득이 5-3=2이므로 선택한다. [···1] 이 선택되어지고, 함수 의 K map이 f= f⊕C에 의해 변화된다. 이 때 큐브 C= [···1] 이다. 다음으로 변화된 함수 맵에서 양수의 이득을 가진 가장 큰 큐브를 찾아가면서 이득이 3-1=2인 큐브 [10···] 가 선택되고, 함수 맵이 변한다.

이 방법에 따라 그림 3과 같이 큐브가 선택되어지면 함수 맵을 변화시키고, 다음 큐브를 변화된 맵에서 찾는다. 함수 맵의 민텀의 수가 0 일때까지 함수 맵은 변화한다. 즉, 함수 맵의 민텀의 수가 0 일때까지 함수 맵은 변화한다. 그림 2의 최소화 결과와 같이 항의 수와 리터럴의 수가 동일하다. 따라서 이 예제에서 Cube selection 방법이 *xlinking* 방법과 같이 잘 동작함을 알 수 있다.

4. *Xlinking*과 Cube selection의 대비 설명

다음은 간단한 예제를 통해서 Cube selection과 *Xinking*을 비교해서 설명한다. 그림 4는 Cube selection 방법에 의하면 주어진 논리함수의 최소화된 결과를 쉽게 찾을 수 있음을 보여준다. 즉 가장 큰 큐브인 [···1] 를 선택하고, 다음으로 큰 큐브 [10···] 를 선택하면 주어진 함수의 ON-set가 완전히 커버된다. 그러나 *Xinking* 방법을 적용하면 최소화 결과를 예측할 수 없다. 왜냐하면 묶어지는 순서에 따라 선택되어지는 큐브 쌍의 조합이 매우 많기 때문이다. 일련의 큐브 쌍에 *Xinking* 방법을 적용하면 최적의 결과를 얻을 수는 있으나 정확한 일련의 순서를 짓하는 것은 쉽지 않다. 따라서 적절한 큐브 쌍을 선택하는 것을 *Xinking* 방법에서 찾아 내기는 매우 어렵다. 각 민텀으로 부터 시작해서 적절한 큐브쌍을 선택하기 위해서는 거대한 탐색공간이 생기며, 이것은 최소화 과정을 복잡하게 만드는 요인이 된다. 이 예제로부터 *Xinking*보다 Cube selection의 최소화 과정이 매우 간단하다는 것을 알 수 있다.

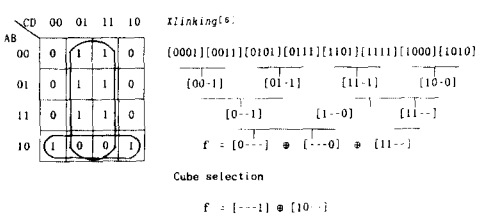
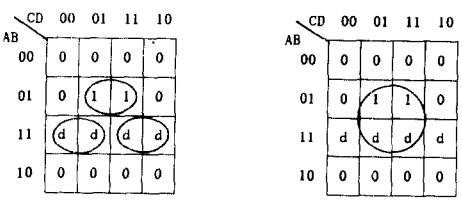


그림 4. *Xlinking* 방법과 Cube selection 방법의 비교

Fig 4. Comparison of *Xinking* and Cube selection.

5. 불완전하게 표현된 함수

Xlinking 방법을 이용하여 don't care항을 처리하기가 쉽지 않다. 왜냐하면, 이 방법은 큐브를 형성하는 don't care항이 다음의 *Xinking*과정에 영향을 미치기 때문이다. *Xlinking* 방법은 제약조건을 가지고 두개 큐브를 묶는 것에 유의해야 한다. 예를 들면, 이 방법은 연결하고자 하는 큐브의 집합중 리터럴의 위치가 같은 큐브중에서, 리터럴의 수에 따라 두개의 큐브를 묶는 것에 대한 제약조건이 있다. 따라서 부적절한 방법으로 don't care항을 묶을 수 없고 *Xinking* 방법에 의한 최소화를 위해 don't care항을 사용할 수 없다. 그림 5(a)는 *Xinking* 방법에서 don't care항 사용의 어려운 점을 보여주고 있다. 분명히 큐브 [01-1] 는 don't care항 [11-1] 과 묶어질수 있음에도 불구하고 *Xinking* 방법의 제약조건에 맞지않으므로 묶어질수 없다. 왜냐하면 두개의 큐브가 묶어질 경우 남아있는 don't care항 [1100] 과 [1110] 와의 리터럴 수가 2 차이나므로 don't care항이 두개의 원으로 묶어지고 큐브 [01-1] 은 don't care항을 이용할 수 없다. 만일 Cube selection 방법을 사용한다면 위의 문제점을 해결할 수 있다. 큐브의 크기가 4일때 그림 5(b)를 고려하면 먼저 큐브 [-1-1] 가 선택된다. 왜냐하면 가장 좋은 이득(ON-set의 수 - OFF-set 수)= 2 이기 때문이다. 단, don't care항은 계산에 포함되지 않는다. 이 때, 큐브의 크기가 4인 큐브 [0 1] 를 선택하면 이득이 0 이다. 따라서 새로운 큐브를 위해 [-1-1] 이 선택된다. 큐브를 선택하기 위한 과정에 don't care항을 위해 어떠한 고려도 필요하지 않다. 즉 Cube selection 방법에서 don't care 항은 이득 계산에서 제외 된다.



(a)Don't care trems in *Xinking* (b)Don't care trems in Cube selection

그림 5. *Xlinking*과 Cube selection에서 don't care 항 사용 예제

Fig 5. Example of using don't care terms in xlinking and Cube selection.

Ⅳ. 실 험 결 과

본 논문에서 제안한 Cube selection 알고리즘은 NeXT Station에서 C 언어로 작성되었다. 표 1의 benchmark회로들에 대한 실험결과는 Saul [4] 의 결과와 리터럴과 항의 수(XOR게이트 수)의 관점에서 비교한 것인데, 특히 항의 수를 감소하는데 목표를 두었다. 실험결과, Cube selection 방법이 몇가지 회로에 대해서는 항의 수를 감소시키지 못하는 어려움이 있으나 5xpl, 9sym, sao2와 같은 회로에 대해서는 리터럴과 XOR게이트 수를 크게 감소시키는 좋은 결과를 보여주고 있다. 특히, 9sym회로는 지금까지 이론적으로 알려진 최소화된 XOR게이트 수가 61개인데 Cube selection 방법을 적용한 결과 XOR게이트 수가 94개에서 68개로 대폭 감소하여 거의 최적의 결과를 보였다. [3] sao2회로와 5xpl회로와 같이 다중출력을 갖는 회로는 각각의 출력에 대해 Cube selection 방법을 적용하여 각 출력에 대한 최소화된 XOR게이트 수의 합계로 계산하였는데, 결과에 나타난 것처럼 XOR게이트 수를 상당히 감소시켰다는 것을 볼 수 있다.

표 1. Benchmark회로를 이용한 실험 결과

Table 1. Experimental results using Benchmark circuits.

			Saul[4]의 결과		Cube Select의 결과		Time
Circuit	PI	PO	literal	XOR gates	literal	XOR gates	CPU(sec)
5xpl	7	10	223	61	206	* 49	1.3
9sym	9	1	507	94	548	* 68	1.9
ba	5	28	404	92	358	* 83	1.7
b2ml	7	2	99	7	16	13	1.0
clip	9	5	812	123	962	161	9.6
misex1	8	7	99	35	190	43	3.4
sao2	10	4	752	103	548	* 59	30.7
f51m	8	7	132	36	199	46	15.5

* 모든 시간은 NeXT Station에서의 CPU second

Perkowski [6] 의 논문에 나타난 예제 회로들에 대해 *Xinking* 방법와 Cube selection 방법의 실험결과를 비교한 내용이 표 2에 제시 되었다. 실험결과 근사한 결과를 보여 주었고 표 2의 예제 18 회로와 같이 입력의 수가 많고, 민텀의 수가 많은 회로에서는 효과적으로 항과 리터럴의 수가 감소됨을 알 수 있다. 특히 프로그램에 의한 결과보다는 수작업으로 하는 방법이 결과가 좋은 것을 알 수 있다. 그 이유는 프로그램에 의한 방법은 선택되지 않는 가장 큰 큐브부터 시작해서 가장 작은 큐브로 순차적으로 시도하는 반면, 수작업으로 하는 방법은 ON-set의 크기가 가장 빨리 줄어드는 큐브를 효과적으로 찾기 때문이다. 표 2의 예제 5와 같은 회로는 격자 형태로 나타나는 함수인데, 이 경우에 어떤 큐브를 선택해도 이득이 0 이 되므로 주어진 함수의 항이 감소되지 않는다. 이러한 경우는 Cube selection 방법의 문제점으로 지적되며 앞으로 개선하여 해결해야 할 문제로 보여진다. 표 2에서 빠진 부분은 Perkowski [6] 의 논문에 논리함수에 대해 정확하게 정의되어 있지않아서 비교할 수 없었다.

표 2. Perkowski [6] 의 xlinking 실험결과와 비교

Table 2. Comparison of Perkowski's [6] xlinking results.

				Perkowski[6]		Cube selection의 실험결과			
EXAM	PI	PO	MINTE	의 실험결과		by PROGRAM		by MANUAL	
NO			-RM	term#	lit#	term 수	literals 수	term 수	literals 수
1	3	1	4	2	4	3	7	3	7
2	3	1	5	4	6	4	9	4	9
3	3	1	5	2	4	3	4	3	4
4	5	1	10	4	14	6	20	6	20
5	3	1	4	3	3	4	12	4	12
10	3	1	4	3	6	3	7	3	7
12	6	6	49	19	64	32	83	25	83
18	6	12	63	40	117	51	110	35	110

Ⅴ. 결 론

본 논문에서는 일반화된 리드뮬러회로를 최소화하기 위한 새로운 접근인 Cube selection 방법을 제안하였으며, 이 방법은 NeXT Station에서 C 언어로 작성되었다. Cube selection 방법은 남아 있는 민텀의 수가 매우 빠르게 감소되도록 적절한 일련의 큐브들을 하나씩 찾아가는 방법이다. 적절한 큐브들을 찾아내기 위한 여러 가지 간단한 휴리스틱이 있는데, 본 논문에서는 가장 큰 큐브부터 차례대로 선택하여 큐브의 수를 줄이는 방법을 채택하였고, 간단한 예제 함수와 benchmark회로를 통해서 실험 결과를 제시하였다. Benchmark 실험결과, 여러가지 경우에 Cube selection 방법이 기존의 방법들보다 특히 좋

은 결과를 보여 주었다.

본 논문에서 제안된 방법은 모든 두개의 큐브들을 연결하기 위해 복잡한 반복과정을 반복하는 *Xinking* 방법에 비해, 함수의 ON-set를 커버할 때까지 한번에 하나씩 큐브를 선택하는 점이 새로운 것이다. *Xlinking*방법은 조건을 만족하는 큐브중에서 가장 가까운 큐브부터 선택해서 최소화 과정을 시행하므로 최적의 일련의 큐브를 선택하는 것이 매우 어렵고 일련의 큐브 선택이 최적의 과정을 밟지 않으면 최적의 결과를 얻기 어려운 반면 Cube selection 방법은 민텀의 수가 가장 빨리 감소하도록 큐브를 선택함으로 최적에 가까운 결과를 얻을 수 있다. 따라서 어떠한 큐브를 선택하던지 최적의 큐브를 선택하기 위한 반복 과정이 없고 처리과정이 매우 단순하며 큐브의 수를 최소화 하는데 따른 시간이 매우 적게 걸린다.

본 논문에서 제안한 방법이 몇가지의 benchmark 회로에서 좋은 결과를 보였으나, 큐브를 선택하는 과정에 개선해야 할 점이 남아있다. 왜냐하면, 큐브의 선택 순서에 따라 마지막 결과에 영향를 줄 수 있으므 최적의 일련의 큐브를 선택하는 적절한 방법을 생각해야 한다. 본 논문에 제시된 큐브 선택과정은 근사적인 방법에 해당한다. 따라서 최적에 가까운 결과를 생성하기 위해 선택되는 큐브에 우선 순위를 평가하는 체계적인 방법이 개발되어져야 하겠다. 또한 Cube selection 방법과 *Xinking* 방법을 연결하여 개발할 경우, 리드뮬러형식을 위한 효과적인 최소화 도구가 개발될 것으로 예상된다.

參 考 文 獻

[1] A. Sarabi and M.Perkowski, "Fast Exact and Quasi-Minimal Minimization of Highly Testable Fixe-Polarity AND/XOR Canonical Networks," Design Auto mation Conference, pp.30-35, 1992.

[2] D.K.Pradhan, "Universal Test Sets for Multiple Fault Detection in AND EXOR Arrays", *IEEE trans. on Computers*, vol. C-27, no. 2, PP. 181-187, 1978.

[3] J. Saul, "An Improved Algorithm for the Minimization of Mixed Polarity Reed-Muller Representation," ICCD, pp.372-375, 1990.

[4] J. Saul, "An Algorithm for the Multi-level Minimization of Reed-Muller R epresentations," ICCD 1991.

[5] K. Bartlett et al., "BOLD: A Multi-level Logic Optimization System," ICCAD, 1987.

[6] M. Helliwell and M. Perkowski, "A Fast Algorithm to Minimized Multi-out put Mixed-Polarity Generalized Reed-Muller Forms," Design Automation Conference, pp.427-432, 1988.

[7] R.k.Brayton et al., "Logic Minimization Algorithms for VLSI synthesis," Kluwer Academic Publishers.

[8] R.K. Brayton et al., "MIS: A Multi-level Logic Optimization System," *IEEE transaction on CAD*, pp. 1062-1081, Nov. 1987.

[9] S.M. Reddy, "Easily Testable Realization for Logic Functions", *IEEE trans. on Computers*, vol C-21, no. 11, pp. 1183-1188, 1972.

[10] S.Even,I.Kohavi, A.Paz, "On minimal module-2 sums of products for switching functions," *IEEE transaction on Electronic Computers*, vol. EC-16 , pp. 671-674, Oct. 1967.

[11] T.Sasao and P. besslich, "On the complexity of Mod-2 Sum PLA's," *IEEE transaction on Computers*, vol.39 nO.2, pp. 262-265, Feb. 1990.

[12] X.Wu, X Chen, S.L.Hurst, "Mapping of Reed-Muller coefficients and the minimisation of exclusive OR-switching functions," *IEE PROC*, vol.129,Pt.E, no.1 Jan. 1982.

[13] 장준영, 이 화, 이귀상, "리드뮬러회로 합성을 위한 새로운 방법", 전자공학회 추계 학술발표 회지,제 15권 제2호, pp.565-567, 1992년, 11월.

[14] G.S.Lee, M.J.Irwin, and R.M. Owens, "Test Generation in circuit const ructed by input decomposition," ICCD, pp.107-111, 1990.

[15] G.S.Lee, M.J.Irwin, and R.M. Owens, "Polinomial-time Testability of Circuits Generated by Input Decomposition," accepted to *IEEE Transactions on Computer*, 1993.

著者紹介

李 貴 相(正會員)

1958年 2月 1日生. 1980年 2月 서울대학교 전기공학과 졸업. 1982年 2月 서울대학교 컴퓨터공학과 졸업(석사). 1991年 8月 Pennsylvania State University (Ph.D) 1982年 2月 ~ 1983年 3月 금성통신연구소 연구원. 1983年 4月 ~ 현재 전남대학교 전산학과 조교수. 주관심분야는 VLSI설계 자동화, 테스팅, 논리합성, 뉴럴네트워크 등임.

張 峻 榮(正會員)

1962年 8月 20日生. 1985年 2月 전남대학교 전산학과 졸업. 1987年 8月 중앙대학교 전산학과 졸업(석사). 1992年 3月 ~ 현재 전남대학교 전산학과 박사과정 재학중. 주관심분야는 VLSI 설계자동화, 테스팅, 논리합성, 뉴럴네트워크 등임.