스펙트럴 행렬에서 보정부분 탐색에 의한 분해식 산출

권오형*
*한서대학교 항공컴퓨터전공
e-mail:ohkwon@hanseo.ac.kr

Factorization by Searching for Correction Part in Spectral Method

Oh-Hyeong Kwon*
*Dept. of Aeronautic Computer Engineering, Hanseo University

요 약

인수분해를 통해 논리식을 간략화 하기 위한 방법을 제안한 것이다. 이때 인수분해를 위한 방법은 스펙트럴 행렬을 이용하였으며, 기존의 스펙트럴 방법과 달리 논리식이 간략화 될 것으로 예상되는 보정 부분을 찾고 이 보정 부분을 포함하는 식을 산출해서 분해식을 산출하는 방법을 제안한다. 실험 결과 보정 부분 탐색으로 수행시간을 늘어났지만 논리식을 구성하는 리터럴 개수를 줄이는 효과 얻었다.

1. 서론

반도체 부품을 설계하는 단계에서 논리합성은 간략화된 논리식 산출을 목표로 하며 이 결과를 논리회로로 변환하는 방법을 취하고 있다. 특히 다단 논리식이 이단 논리식보다 간략화된 결과를 산출하는 것은 잘 알려진 사실이나 완벽한 다단 논리식을 산출하는 방법이 정립되어 있지 않아 여전히 연구가 진행 중에 있다. 지금까지 진행된 연구 중에서 다단 논리식을 산출하기 위한 방법 중의 한 가지로는 논리식 내에서 존재하는 공통 인수를 찾아 논리식을 다단의 형태로 만드는 인수분해 방법이다. 인수분해에 의해 산출된 논리식을 분해식(factored form)이라 부른다. 분해식은 $F = QD + R$ 형태가 되고, Q, D, R은 반복해서 분해식을 표현이 될 수 있다[1-4].

본 논문은 분해식을 산출하기 위한 새로운 방법을 제안한 것으로 스펙트럴 행렬(Spectral Matrix)[5-8]을 활용하였다. 제안한 방법에 의해 산출된 분해식은 AND, OR, NOT, XOR(Exclusive-OR) 연산자로 구성되며, 스펙트럴 행렬로는 하다마드(Hadamard) 행렬을 사용한다. 특히 2개 입력을 갖는 XOR 게이트는 3개의 트랜지스터로 구현[9]이 될 수 있고, 또한 논리식에 XOR 연산자를 활용하면 더욱 간략화된 논리식을 산출하는 장점을 갖기 때문에 XOR 연산자를 AND, OR. NOT 연산자와 함께 사용해서 분해식을 산출하는 방법을 제안한 것이다. 또 제안하는 방법은 기존의 스펙트럴 방법과 달리 보정 부분을 활용해서 분해식을 산출하는 것으로 차별화된 방법이다.

논문의 구성은 다음과 같다. 2장에서는 본 논문 서술에 필요한 배경 지식으로 용어 정의와 스펙트럴 행렬에 대해 서술하고, 3장에서 제안하는 방법에 대해서 소개한다. 4장에서 실험결과를 보이고, 5장에서 결론을 제시한다.

2. 배경지식

정의 1: 변수(variable)는 부울 공간(Boolean space)에서 한 좌표를 나타내는 문자다. 리터럴(literal)은 변수 그 자체 또는 그의 보수(complement)다. 항(term) 또는 큐브(cube)는 리터럴들의 집합으로 만일 리터럴 a가 존재하면, 그의 보수 리터럴 a'을 포함하지 않는다. 단순식(expression 또는 sum-of-products(SOP) form)은 항 또는 큐브들의 집합이다.

예 1: 문자 a는 변수이며, a와 a'은 리터럴이다. 리터럴 집합 $\{a,b\}$는 큐브이나 집합 $\{a,a'\}$은 큐브가 아니다. $\{\{a,b'\},\{b,c\}\}$는 단순식이다.

본 논문에서는 항과 큐브를 같은 의미로 사용한다. 또한 큐브와 단순식을 표현하는 경우 집합 표기와 보편적으로 사용되는 논리식 표기를 모두 사용한다. 따라서 큐브 $\{a,b\}$는 ab와 동일한 표현이며, $\{\{a,b'\},\{b,c\}\}$는 $ab'+bc$와 동일한 표현이다.

정의 2: 논리식 F의 서포트(support)는 논리식 F를 구성하는

변수들 집합으로$sup(F)$로 표현한다. 논리식을 구성하는 모든 큐브들 간에 공통으로 사용되는 리터럴을 갖지 않은 경우 그 논리식은 큐브면제(cube-free) 되었다고 한다.

예 2: 논리식 $F=a+bc'$의 경우, $sup(F)=\{a, b, c\}$. 논리식 $ab+c$ 는 큐브 면제된 경우이나, 논리식 $ab+ac$ 및 abc는 큐브 면제된 것이 아니다.

정의 3: 하다마드(Hadamard) 행렬 H의 각 원소는 다음과 같이 재귀적 수식에 의해 산출된다. 하다마드 행렬의 원소값은 1과 -1로 표현된다. 즉, 논리값 0은 하다마드 행렬에서 1로, 논리값 1은 -1로 변경한다. 여기서 n은 입력 변수의 개수를 나타낸다.

$$H^n=\begin{bmatrix} H^{n-1} & H^{n-1} \\ H^{n-1} & -H^{n-1} \end{bmatrix}$$

$$H^0=1$$

예 3: $n=1$인 경우와 $n=2$에 대한 각 하다마드 행렬은 다음과 같은 모습이 된다.

$$H^1=\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$H^2=\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

정의 4: 하다마드 행렬의 각 행은 논리식에 대응된다. 논리식에 사용된 입력 변수를 x_i, (i=2^n-1,..., 0)라 하자. 그러면 하다마드 행렬의 각 행에 대응되는 논리식은 첫 번째 행은 0이 되고, 두 번째 행은 x_0, 세 번째 행은 x_1, 네 번째 행은 $x_0 \oplus x_1$, 다음 다섯 번째 행은 x_2, 다음 2개의 행은 각각 $x_0 \oplus x_2$, $x_1 \oplus x_2$와 같이 논리식이 대응된다. 나머지 행에 대해서도 동일한 방식으로 논리식이 대응된다.

예 4: 입력 변수가 c, b, a 경우 8x8 하다마드 행렬과 각 행에 대응되는 논리식을 행렬의 왼쪽에 나타내면 다음과 같다.

$$\begin{array}{r} 0 \\ a \\ b \\ a\oplus b \\ c \\ a\oplus c \\ b\oplus c \\ a\oplus b\oplus c \end{array}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}$$

3. 제안 방법

3.1 보정 부분 산출

기존의 스펙트럴 방식에 의해 논리식을 산출하는 경우, 스펙트럴 행렬과 주어진 진리표를 곱해서 산출된 행렬에서 절대값이 가장 큰 행을 선택하고, 이 행에 대응되는 논리식을 산출한다. 이 때, 산출된 논리식이 주어진 진리표에 일치하지 않는 경우 일치하지 않는 부분을 수정하는 방법으로 논리식을 산출하게 된다. 다음 예는 스펙트럴 방식에 의해 논리식이 산출되는 기존 방법을 보인 것이다. 이러한 방식을 본 논문에서는 절대값 우선 방식에 의한 스펙트럴 방법이라 부르겠다.

예 5: 입력 변수가 3개인 진리표가 다음과 같다고 하자.

입력			출력
c	b	a	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

그러면 하다마드 행렬은 8x8의 크기가 되고 출력 행렬은 진리표의 출력 부분에서 0은 1로 1은 -1로 대치해서 만든다. 그리고 다음과 같이 하다마드 행렬과 출력 행렬의 곱셈을 산출한다.

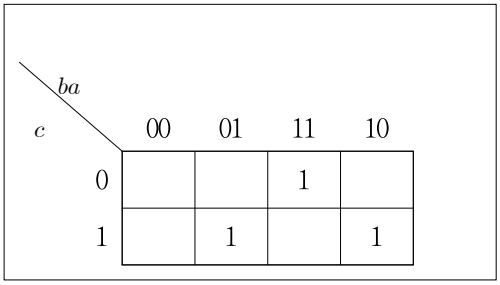
$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}\begin{bmatrix} 1 \\ -1 \\ 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \end{bmatrix}=\begin{bmatrix} 2 \\ 2 \\ -2 \\ 6 \\ 2 \\ 2 \\ -2 \\ -2 \end{bmatrix}$$

식의 오른쪽 변에 있는 행렬에서 6이 가장 큰 절대값이 되며, 이는 네 번째 행이 되며 이는 하다마드 행렬에서 $a \oplus b$가 된다. 그러면 $a \oplus b$는 $a'bc'$가 1이 되는 것으로 포함하기 때문에 이를 제거하기 위한 보정 부분이 필요하게 되어 논리식은 $F=(a \oplus b) \oplus a'bc'$ 가 된다.

예 5에서 보인 기존의 스펙트럴 방식에 의한 논리식 산출은 출력이 1이 되는 부분을 가능한 많이 포함하는 스펙트럴 행렬의 행에 대응되는 논리식을 산출하는 것이다. 반면에 본 논문에서 제안하는 방법은 스펙트럴 방법에 의해 산출된 행렬에서 보정부분을 중심으로 논리식을 산출하는 보정 방식에 의한 스펙트럴 방법으로 기존의 방법과 차별된 방법을 제안한 것이다. 다음 예는 제안하는 방법에 따라 논리식이 산출되는 과정을 보인 것이다.

예 6: 그림 1과 같이 주어진 카르노 맵에 대응되는 논리식 F

를 산출하는데 하다마드 행렬을 이용하자.



[그림 1] 논리식 $abc+ab'c+a'bc$ 의 카로노 맵

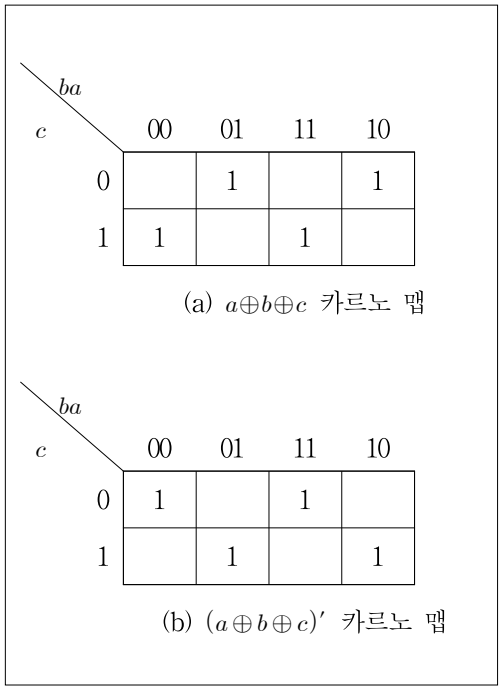
그러면 다음과 같은 행렬 곱셈 결과를 산출하게 된다.

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ -1 \\ 1 \\ -1 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ -6 \end{bmatrix}$$

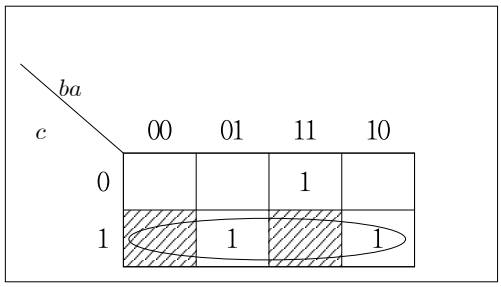
기존 방식에 의해 논리식을 산출할 경우는 절대값이 가장 큰 -6에 해당하는 8번 째에 해당하는 논리식인 $a \oplus b \oplus c$가 선택된다. 이 논리식에 해당하는 카로노 맵은 그림 2과 같다. 이 경우 그림 1의 카르노 맵으로 변경하기 위해서는 그림 2(b)의 $a \oplus b \oplus c$에 NOT을 적용한 $(a \oplus b \oplus c)'$를 구하고 다시 $a'b'c'$ 해당하는 항을 제거하는 과정이 필요하다. 반면에 제안한 방법에 의하면, 행렬 산출에서 5번 째 행에 해당하는 논리식 c를 선택한 경우 보정할 부분은 $a'b'c'$과 abc이다. 그림 3은 주어진 출력과 함께 c를 선택한 경우를 보인 것으로 빗금친 부분이 보정할 곳이다. 그림 3에서 묶음으로 표시된 부분이 논리식 c가 된다. 그러면 산출된 논리식은 $F = c \oplus a'b'c \oplus ab$가 된다. 이를 정리하면 다음과 같은 논리식이 산출된다.

$$\begin{aligned} F &= c(1 \oplus a'b') \oplus ab \\ &= c(a'b')' \oplus ab \\ &= c(a+b) \oplus ab \end{aligned}$$

제안한 방식에 의하면 최종적으로 5개의 리터럴로 구성된 논리식을 산출한다.



[그림 2] $a \oplus b \oplus c$ 와 $(a \oplus b \oplus c)'$의 카르노 맵



[그림 3] 논리식 c 의 카르노 맵과 보정 부분 표시

본 논문에서는 보정 부분 선택하기 위해서 다음과 같은 규칙을 제안한다.

선험 규칙: 논리식을 간략화 할 것으로 예상되는 보정 부분은 보정부분은 출력이 1인 부분들 중에서 해밍거리가 1이 가장 많이 나타나는 것을 선택한다.

예 7: 그림 3의 카르노 맵에서 1에서 해당하는 최소항 abc', $ab'c$, $a'bc$과 스펙트럴 대상이 되는 항들에 포함되는 0인 부분들과의 해밍거리가 1인 회수를 산출하자. 만일 c를 선택하면 그림 3의 abc에 해당하는 0인 부분과 최소항들의 해밍거리 회수를 다음과 같다. 1=HD(abc, abc'), 1=HD(abc, $ab'c$), 1= HD(abc, $a'bc$)가 되기 때문에 abc과 관련된 회수는 3이 된다.

3.2 알고리즘

알고리즘을 4단계로 구성되며 단계 1은 입력 개수

에 따라 하다마드 행렬의 크기가 결정되고, 하다마드 행렬과 주어진 진리표의 출력에 대한 행렬 곱셈을 산출한다. 이 때, 0은 1로 1은 -1로 변경한 출력 행렬을 만들어 곱셈을 한다. 단계 2에서는 보정 부분이 출력이 1인 부분과 해밍거리가 1인 회수를 산출해서 회수가 가장 많은 부분을 포함하는 보정 부분을 선택한다. 단계 3에서는 선택된 보정 부분을 포함하는 하다마드 행렬에 대응되는 식을 선택한다. 마지막 단계 4는 위의 과정을 반복해서 수행해서 다단 논리식을 산출하게 된다.

Algorithm : 보정 부분 선택에 의한 분해식 산출
입력: 각 출력 별 진리표
출력: 분해식
단계 1. 하다마드 행렬과 출력에 대한 행렬 곱셈을 산출
단계 2. 선험 규칙에 따라 보정 부분을 선택
단계 3. 보정 부분을 포함하는 논리식 산출
단계 4. 단계 3에서 산출된 논리식의 일부분에 대해서 단계1부터 3을 적용한 분해식 산출

3.3 실험 결과

[표 1] 실험 결과

Circuit Name	Hurst		Proposed method	
	# of literals	time (sec)	# of literals	time (sec)
rd53	82	0.1	73	0.2
rd73	172	0.1	164	0.2
con1	22	0.1	19	0.1
z4ml	76	0.1	65	0.2
cmb	69	0.1	67	0.2
C17	13	0.1	9	0.1

3.3GHz i3 CPU가 장착된 Linux 운영체제의 컴퓨터에서 C언어로 제안한 방법을 구현하였다. 본 실험에서는 Microelectronics Center of North Carolina (MCNC) 벤치마크 회로[10]를 대상으로 수행 시간과 리터럴 개수에 대해서 [5-7]에서 제시하는 방법 결과를 비교하였다. 표 1은 실험 결과를 정리한 것이다. 실험 대상의 벤치마크 회로에 대해서 제안한 방법으로 산출된 논리식이 보다 간결함을 보였다. 반면에 보정 부분 산출에 소요되는 시간의 증가로 수행시간이 기존 방법보다 늘어났다.

4. 결론

분해식을 산출하는데 하다마드 행렬을 적용한 방법을 제안하였다. 기존 방법과 달리 보정 부분 탐색을 통해 보다 간략화된 분해식을 산출하는 방법을 본 논문에서 제안하였다. 실험 결과 수행시간을 기존 방법보다 다소 늘어났지만 논리식을 구성하는 리터럴 개수를 줄일 수 있었다.

참고문헌

[1] R. K. Brayton and C. McMullen, “The Decomposition and Factorization of Boolean Epressions,” *Proc. ISCAS*, pp. 49-54, 1982.

[2] R. K. Brayton, R. Rudell, A. Sangiovanni-Vincentelli, and A. R. Wang, “ MIS: A Multiple-Level Logic Optimization System,” *IEEE Trans. CAD*, Vol. 6, No. 6, pp. 1062-1081, 1987.

[3] E. M. Sentovich, K. J. Singh, C. Moon, H. Savoj, R. K. Brayton, R. K., and A. Sangiovanni-Vincentelli, "Sequential Circuit Design Using Synthesis and Optimization," *Proc. ICCD*, pp. 328-333, 1992.

[4] O.-H. Kwon and B.T. Chun, “Boolean Factorization Using Two-cube Non-kernels", *Journal of Korean Academia-industrial cooperation Society*, Vol. 11, No. 11, pp. 4597-4603, 2010.

[5] S. L. Hurst, D. M. Miller, and J. C. Muzio, *Spectral Techniques in Digital Logic*.. San Diego, CA: Academic Press, 1985.

[6] C. R. Edwards and S. L. Hurst, "A Digital Synthesis Procedure Under Function Symmetries and Mapping Methods," *IEEE Trans. Computer*, Vol. c-27, No. 11, pp. 985-997, 1978.

[7] M. G. Karpovsky, R. S. Stankovic, and J. T. Astola, *Spectral Logic and Its Applications for the Design of Digital Devices*.. Hoboken, New Jersey: John Wiley & Sons, 2008.

[8] L. Diaz-Olavarrieta, K. Illanko, and S. G. Zaky, "Goal-Oriented Decomposition of Switching Functions," *IEEE Trans. CAD*, Vol. 12, No. 5, pp. 655-665, 1993.

[9] J. B. Kim, S. J. Hong, and J. Kim, "New Circuits for XOR and XNOR Functions," *International Jonrnal of Electronics,* Vol. 82, No. 2, pp. 131-144, 1997.

[10] S. Yang, "Logic Synthesis and Optimization Benchmarks User Guide Version 3.0," Technical Report, Microelectronics Center of North Carolina, 1991.