논문번호 03-02-03

다치 논리 함수 연산 알고리즘에 기초한 MOVAG 구성과 T-gate를 이용한 회로 설계에 관한 연구

A Study on the Constructions MOVAGs based on Operation Algorithm for Multiple Valued Logic Function and Circuits Design using T-gate

尹 炳 熙*, 朴 秀 珍*, 金 興 壽*
Byoung-Hee Yoon*, Soo-Jin Park*, Heung-Soo Kim*

요 약

본 논문에서는 Honghai Jiang에 의해 제안된 OVAG(Output value array graphs)를 기초로 MOVAG(Multi output value array graphs)를 이용한 다치논리함수의 구성방법을 제안하였다. D.M.Miller에 의해 제안된 MDD(Multiple-valued Decision Diagram)는 주어진 다변수의 함수에서 회로 설계까지 많은 처리시간과 노력이 요구되므로 본 논문에서는 MDD의 단점을 보완하여 데이터 처리시간의 단축과 적은 복잡도를 갖도록 MOVAG를 설계하였다. 또한 MOVAG의 구성 알고리즘과 입력행렬선정 알고리즘을 제안하고 T-gate를 사용하여 다치 논리 회로를 설계, 모의 실험을 통해 그 결과를 검증하였다.

Abstract

In this paper, we proposed MOVAG(Multi Output Value Array Graphs) based on OVAG by Honghai Jiang to construct multiple valued logic function The MDD(Muliple-valued Decision Diagra) needs many processing time and efforts in circuit design for given multi-variable function by D.M.Miller, and we designed a MOVAG which has reduce the data processing time and low complexity. We propose the construction algorithm and input matrix selection algorithm and we designed the multiple-valued logic circuit using T-gate and verified by simulation results.

Key word : OVAG, MOVAG, MDD, T-gate

I. 서 론

최근 2진논리에 근거한 집적회로 기술의 발전으로 회로의 형태가 VLSI, ULSI화되어 단일 칩상에 방대한 양의 회로를 집적할 수 있게 되었다. 그러나 단일 칩상에 방대한 양의 회로를 집적하기 위해서는 칩상의 상호 결선의 복잡성, 외부 단자의 증가와 연산속도의 제한성, 정보전송량의 방대함에 따른 정보전송시간지연등의 문제점들이 대두되기 시작했다.

이러한 문제점들을 해결하기 위해 단일 결선에서 더 많은 양의 정보를 보낼 수 있는 다치논리이론의 연구

* 인하대학교 전자공학과
(Dept. of Electronic Eng., InHa Univ.)
接受日:2003年 7月 21日, 修正完了日:2004年 6月 25日

가 1970년대 초반부터 활발히 진행되고 있다.[1,2,3] 2진 시스템에서는 n개의 정보선에 2^n개의 정보를 전송할 수 있지만 p치시스템에서는 p^n개의 정보를 전송하여 상호 결선의 복잡성, 외부 단자의 증가와 같은 문제점들을 해결할 수 있으며 처리 능력 및 신뢰성을 향상시키고 비용을 현저하게 줄일 수 있다.

S.B.Akers[4]는 방향성 비순환 그래프(Directed Acyclic Graphs: DAG) 형태의 데이터 구조인 결정도(Decision Diagram: DD)의 개념을 정립하였으며, R.E.Bryant[5]는 S.B.Akers의 2치결정도(Binary Decision Diagram : BDD)에 기초하여 부울함수를 방향성 비순환그래프(Directed Acyclic Graph : DAG) 형태의 데이터 구조로 표현한 후 부울함수를 간략화하는 방법을 제안하였다. D.M.Miller[6]는 R.E.Bryant가 제안한 순서화된 2치결정도에 대한 연구를 최초로 다치논리에 적용하여 확장한 다치 결정도(Multiple-valued Decision Diagram)를 제안하였다. MDD는 지금까지 OMDD(Ordered MDD)와 ROMDD(Reduced OMDD)등과 같은 간략화 알고리즘이 제안되어 회로 설계를 효과적으로 할 수 있었다. 그러나 이러한 많은 응용에도 불구하고 BDD와 MDD는 다변수 함수에서는 절점수의 무한 증가에 따른 너무나 느린 계산 능력과 노력이 요구되는 단점이 있다. 즉, 주어진 다변수 함수로부터 회로 설계까지의 과정에서 결정도를 이용한 함수구성방법은 OMDD와 ROMDD등 여러 단계를 거쳐야 소자의 수를 축약시킬 수 있으므로 결정도를 이용한 방법은 더욱 복잡해지고 증가하는 소자에 따라 비용도 증가한다.

이러한 단점을 보완하기 위하여 Honghai Jiang[7]은 2치시스템에서의 입력과 출력 사이의 효과적인 관계를 도식적으로 표현하는 출력값배열그래프(Output value array graph: OVAG)개념을 제안하였다. OVAG란 문자 그대로 입력에 따른 출력값 생성의 일련의 과정을 무시하고 단지 출력값만을 그래프에 배열해 놓은 것이다. 이 때 출력값의 생성은 주어진 함수식에 최적의 입력행렬을 선정, 입력행렬의 각각의 레벨을 고려하여 출력값을 생성하게 된다. BDD로 구성시에 나타나는 절점수 증가에 따른 처리시간의 증가는 OVAG의 변환을 통해 감소시킬 수 있다.

본 논문에서는 OVAG의 개념을 확장하여 다치논리함수 구현에 적용한 MOVAG (Multi-OVAG)를 제안하였고 회로 설계를 통해서 MOVAG를 도출하는 알고리즘을 제안하였으며 다치조합논리함수에 대한 회로를 MOVAG 변환을 통해 MDD와 비교·검토하여 절점의 개수 및 처리시간이 감소함을 보였다.

II. 다치결정도

Miller는 Akers의 BDD를 다치논리에 적용하여 MDD를 제안하였다. MDD는 하나의 절점에 p치의 수만큼 가지가 생성된다. 그러므로 BDD는 2^n-1개의 절점수를 갖지만 MDD는 $(p^n-1)/(p-1)$개의 많은 절점수를 갖는다. 즉, BDD보다는 MDD를 구성하는데 있어 더 많은 시간과 노력이 소요되므로 Miller는 더 복잡한 MDD를 축소화하는 알고리즘을 제안하였는데 MDD의 절점들을 최적의 순서로 배열한 후 함축하는 알고리즘으로 그 단계는 순서화된 MDD(Ordered MDD: OMDD)와 함축순서화 MDD(Reduced Ordered MDD: ROMDD)로 이루어진다.

2.1 다치결정도의 응용 및 한계성

다치결정도를 사용하는 최후의 목적은 회로 설계에 있다. 하나의 절점을 MUX나, T-게이트[11] 등의 소자를 사용하여 회로 설계를 보다 용이하게 사용할 수 있도록 한다. 그러나 다치결정도를 응용하여 회로 설계까지는 과정이 복잡하고 많은 처리 과정이 소요된다.

다음 그림 1에서는 함수 $F=x_1 \wedge x_2$의 다치결정도를 나타냈다.

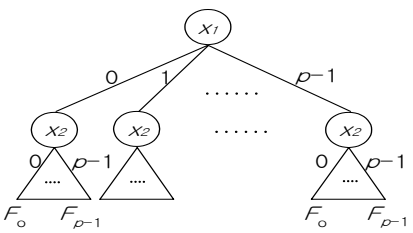


그림 1. 함수 $F=x_1 \wedge x_2$의 MDD

Fig. 1. MDD of function $F=x_1 \wedge x_2$

그림 1에서 본 바와 같이 2변수의 MDD에서 다치 다변수 일수록 절점수가 무한 증가하는 것을 알 수 있다. 즉, 다변수에서는 $(p^n-1)/(p-1)$개의 절점이 생성된다. 그림 1의 MDD에서 각 절점을 T-게이트로 구현하면 다음과 같은 그림 2와 같이 구성할 수 있다. 그림 2와 같은 회로에 사용된 다치 T-gate의 기본 블록도는 그림 3과 같다.

다치 T-게이트의 구조는 p개의 입력 단자와 하나의 출력 단자로 구성된다. T-게이트의 입력 단자는 MDD에서의 종단 절점에 해당된다. MDD는 최상위 절점으로부터 종단 절점으로 출력을 찾아 내려가지만 T-게이트에서는 종단 절점에 해당하는 입력단으로부터 출력값을 찾아간다. 또한 각각의 T-게이트를 제어하는 Xi는 0,1,2,…,p-1일 때 그에 해당하는 단자의 출력값을 생성한다.입력에 따른 출력값을 나타내는 T-게이트의 기본 수식은 식(1)과 같다.

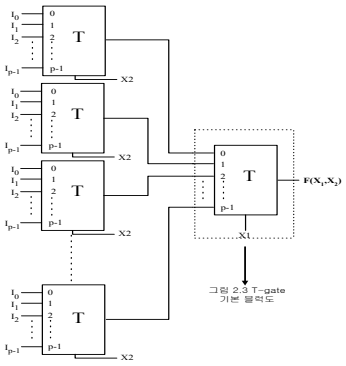


그림 2. 함수 $F=x_1 \wedge x_2$의 회로 구현도
Fig. 2. Circuit design of function $F=x_1 \wedge x_2$

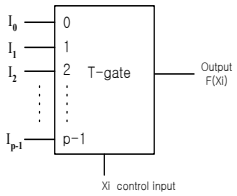


그림 3. T-게이트의 기본 블록도
Fig. 3. Basic block diagram of T-gate

$$\begin{aligned} F(Xi) &= 0 \quad if \quad Xi = 0 \\ &= 1 \quad if \quad Xi = 1 \\ &= 2 \quad if \quad Xi = 2 \\ &\quad \vdots \\ &= p-1 \quad if \quad Xi = p-1 \end{aligned} \tag{1}$$

지금까지 MDD의 회로 구현에 대해 논하였다. 그러나 다변수의 함수에서는 그 절점수가 $(p^n-1)/(p-1)$로 증가함에 따라 회로를 구현하기 위해서는 $(p^n-1)/(p-1)$개의 소자가 사용된다.

Miller에 의해서 MDD를 OMDD, ROMDD의 처리 과정을 거쳐 좀 더 축소된 절점수를 갖는 결정도를 구할 수 있어 절점에 해당하는 소자를 줄일 수 있다. 그러나 주어진 다변수 함수에서 회로 구현까지의 그 과정이 복잡할 뿐더러 많은 노력이 요구되는 단점을 갖고 있다.

2.2 다치논리 함수표현을 위한 기본 연산자

다치논리함수를 수식으로 표현하는데는 여러 표현 방법이 있으나 본 논문에서는 Postian 대수를 이용하여 2진논리연산을 그대로 확장시킨다는 관점에서 논리적(conjunction), 논리합(disjunction) 연산자들을 사용하여 다치논리함수를 표현한다.[9,10]

1) 논리적과 논리합

$x, y \in \{0, 1, 2, \cdots, p-1\}$인 두 변수에 대한 논리적과 논리합은 다음 식(2)와 (3)과 같이 정의한다.

$$x \wedge y = \mathrm{MIN}(x, y) \tag{2}$$
$$x \vee y = \mathrm{MAX}(x, y) \tag{3}$$

여기서, 논리적 MIN(x, y)는 변수 x, y의 최소치인 변수값이고 논리 합 MAX(x, y)는 변수 x, y의 최대값을 나타낸다.

위의 식(2)와 식(3)에 3치와 4치를 적용하여 진리표로 나타내면 다음의 표 1과 표 2와 같이 나타낼 수 있다.

표 1. 3치 논리적과 논리합의 진리표
Table 1. Truth table of ternary disjunction and conjunction

x \ y	0	1	2
0	0	0	0
1	0	1	1
2	0	1	2

(a) x∧y = MIN(x,y)

x \ y	0	1	2
0	0	1	2
1	1	1	2
2	2	2	2

(b) x∨y = MAX(x,y)

표 2. 4치 논리적과 논리합의 진리표
Table 2. Truth table of quaternary disjunction and conjunction

x \ y	0	1	2	3
0	0	0	0	0
1	0	1	1	1
2	0	1	2	2
3	0	1	2	3

(a) x∧y = MIN(x,y)

x \ y	0	1	2	3
0	0	1	2	3
1	1	1	2	3
2	2	2	2	3
3	3	3	3	3

(b) x∨y = MAX(x,y)

위의 표 1과 표 2에 따라 다음과 같은 예의 함수식이 정의된다.

예 2.1) 3치 연산의 예

$x \wedge 0 = 0 \wedge x = 0$ $\quad x \wedge 2 = 2 \wedge x = x$

$x \vee 0 = 0 \vee x = x$ $\quad x \vee 2 = 2 \vee x = 2$

$1 \wedge x \neq x$ $\quad 1 \wedge 2 = 2 \wedge 1 = 1$

위의 예에서 살펴보면 다치논리연산의 Postian 대수에서도 부울함수와 같이 교환법칙이 성립함을 알 수 있다.

2) 부정연산자(negation operator)

본 논문에서는 Postian 대수를 사용하여 다음 표 3과 같이 부정연산자를 표현하였다.

표 3. p치 부정연산자
Table 3. p-value negation operator

X^0	X^1	X^2	·····	X^{p-1}
0	1	2	·····	$p-1$
1	2	3	··· $p-1$	0
2	3	···	·· $p-1$ 0	1
⋮	⋮	⋮	⋮	⋮
$p-1$	0	1	·····	$p-2$

III. MOVAG

본 장에서는 MOVAG의 정의와 특성에 대해 다루었고 MOVAG 구성방법에 대한 알고리즘을 제안하였다. 그리고 다치논리 함수 중에서 3치 논리함수를 예를 들어 설명하였다.

3-1 수학적 배경[7]

다치조합논리함수 F의 입력 변수의 수를 N이라 하면 이들 N개의 입력변수는 집합 n으로 나눌 수 있으며 각각의 집합은 입력 변수 $m(i)(i=1,2,\cdots,n)$로 구성된다. 이를 식으로 표현하면 다음 식(4)와 같다.

$$N = \sum_{i=1}^{n} m(i) \tag{4}$$

본 논문에서는 $(x_{i1},x_{i2},\cdots,x_{im(i)})=(0,0,\cdots,0)$과 $(x_{i1},x_{i2},\cdots,x_{im(i)})=(p-1,p-1,\cdots,p-1)$를 나타내기 위해 $X_i=0$, $X_i = p^{m(i)}-1$로 표현하였다.

위 내용으로부터 입력행렬(input matrix) X를 표현하면 다음 식(5)와 같다.

$$X = \begin{bmatrix} x_{11}, x_{12},\cdots, x_{1m(1)} \\ x_{21}, x_{22},\cdots, x_{2m(2)} \\ \vdots \\ x_{i1}, x_{i2},\cdots, x_{im(i)} \end{bmatrix} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{bmatrix} \tag{5}$$

여기서, $x_{ij}\in\{0,1,2\ \cdots,\ p-1\}$ 이다.

표1, 표2와 표3으로부터 변수배열 Xi에 대한 다치논리함수의 Shannon 확장식[11]을 정의하면 다음 식(6)과 같다.

$$\begin{aligned} f(Xi) = {} & x_{i1}^{0} f(0, x_{i2},\cdots, x_{im(i)}) \\ & \vee x_{i2}^{1} f(0, x_{i2},\cdots, x_{im(i)}) \vee \cdots \\ & \vee x_{i1}^{p-1} f(p-1, x_{i2},\cdots, x_{im(i)}) \end{aligned} \tag{6}$$

여기서, Xi=k-1 ($1\le k\le p^{m(i)}$)이고 x_{ij}^{0}, x_{ij}^{1}, $\cdots$ x_{ij}^{p-1}은 표3의 부정연산자에 따른다.

위 식(6)으로부터 다음과 같은 성질(1)을 얻을 수 있다.

[성질 1]

(1) k에 대하여 $F(X)|_{Xi=k-1}\equiv p-1$이면, $F(X)=p-1$이다.

(2) 이외의 경우에는 $F(X)\equiv F(X)|_{Xi=k-1}$이다.

3-2 MOVAG 기본 성질

본 절에서는 MOVAG의 기본 성질을 다음과 같이 정의한다.

[정의 1]

MOVAG는 방향성 그래프이고 절점들의 집합을 δ라 하면, 레벨값배열은 0,1,2,…,p-1,*로 구성된다. 만일 레벨값배열의 요소가 절점 V_k의 자손이라면 child(V_k)$\in\delta$이다. 그러므로 종단절점이 아닌 절점은 적어도 하나의 *을 포함하고 종단절점은 0,1,2,…,p-1의 값을 갖는다.

[정의 2]

MOVAG는 $V=(v_1,v_2,\cdots,v_p m)$으로 표현되는 최상위 절점을 가지며 그에 대한 레벨변수배열이 $Xv=(x_1,x_2,\cdots,x_m)$이고 이때 다치함수의 출력값은 아래의 식(7)과 같다.

$$f_v = x_1^{p-1}x_2^{p-1}\cdots x_m^{p-1}f_{v1}\vee x_1^{p-1}x_2^{p-1}\cdots x_m^{p-2}f_{v2}\vee\cdots \vee x_1^{0}x_2^{0}\cdots x_m^{1}f_{v(p-1)}{}^{m}\vee x_1^{0}x_2^{0}\cdots x_m f_{vp}{}^{m} \quad (7)$$

여기서, x_i^0, x_i^1, … x_i^{p-1}(i=1,2,…,m)는 표 3의 부정연산자에 따른다. 위 식(7)에서 $Xv=k$-1(k=1,2,,…,p^m)이고 다음과 같은 성질(2)를 갖는다.

[성질 2]

(1) 만일 $v_k=p$-1이면, $fv=fv_{ki}=p$-1이다.

(2) 만일 v_k=*이면, $fv=fv_{ki}=f_{child}(v_{ki})$이다.

[정의 3]

MOVAG의 하나의 레벨에서 두 개의 절점 V'와 V''의 각각의 요소가 같다면 동일한 절점(배열)이라고 하고 동일한 요소의 조건은 다음과 같은 성질 (3)을 갖는다.

[성질 3]

$v_k'\in V'$, $v_k''\in V''$의 조건은 1:1 동일함을 의미하며

(1)$v_k'=v_k''$이면 $v_k'\neq$* 또는 $v_k''\neq$*이다.

(2)$fv_k'=fv_k''$이면 v_k'=* 또는 v_k''=*이다

[정의 4]

MOVAG의 크기는 주어진 입력행렬에 의해 영향을 받으므로 입력행렬의 순서에 따라 MOVAG 절점들의 최대수를 다음 표 4와 같이 나타낼 수 있다.

표 4. 입력행렬 순서에 따른 MOVAG 절점의 최대수
Table 4. Max. number of vertices

레벨	최대 절점수
1	$@N_1=1$
2	$@N_2=MIN(\ p^{\,p^{m_2}},\ @N_1 * p^{\,m_1})$
3	$@N_3=MIN(\ p^{\,p^{m_3}},\ @N_2 * p^{\,m_2})$
:	:
n	$@Nn=MIN(\ p^{\,p^{m_n}},\ @N_{n-1} * p^{\,m_{n-1}})$

3-3 MOVAG의 구성

3.3.1 입력행렬 선정 알고리즘

MOVAG는 입력행렬에 따라 MOVAG의 크기가 결정되므로 입력변수를 최적의 순서로 배열하는 것이 주요 관점이다. 본 절에서는 입력행렬선정 알고리즘을 제안하고 MDD와 MOVAG의 관계를 논의하였고 MOVAG 구성에 필요한 알고리즘을 제안하고 이에 바탕을 둔 MOVAG를 구성하는 방법을 제안하였다.

1)입력행렬선정 알고리즘

[Step1] 다치함수의 입력변수의 수만큼 각각의 입력행렬 레벨에 입력 변수를 할당한다. 이때 i레벨에서는 변수의 수에 따라 절점수가 p^m으로 확장된다.

[Step2] i레벨의 입력행렬에서 레벨에 삽입될 입력변수를 고려한다.

[Step3] 고려된 입력 변수에 해당하는 레벨에서 다치조합논리함수의 출력값이 최대 값을 가질 수 있도록 입력변수를 선택한다.

[Step4] 나머지 레벨에 대해서도 단계3을 적용한다.

2) MDD와 그의 MOVAG의 관계

본 절에서는 MDD와 그의 MOVAG 사이의 관계를 논의하기 위해 간단한 예를 제시한다. p치 2입력 변수를 갖는 다치조합논리함수식 $F(X)=x_1\wedge x_2$에 대한 MDD는 그림 4와 같고 그에 대한 MOVAG를 도출하면 그림 5와 같다. 여기서, 위의 입력행렬 선정 알고리즘에 의해 다음과 같은 입력행렬을 선정한다.

예) 입력행렬 : $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

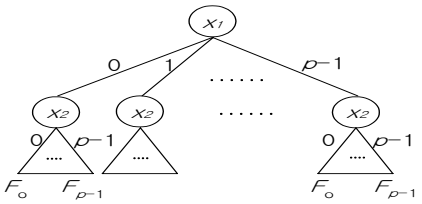


그림 4. $F(X)=x_1 \wedge x_2$의 MDD
Fig. 4. MDD of $F(X)=x_1 \wedge x_2$

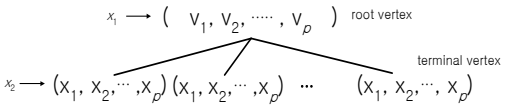


그림 5. $F(X)=x_1 \wedge x_2$에 대한 MOVAG
Fig. 5. MOVAG of $F(X)=x_1 \wedge x_2$

그림 5에서 최상위 절점의 요소는 0,1,…,p-1,* 이고 종단절점의 요소는 0,1,…,p-1이다. 만약, 최상위 절점의 요소가 모두 *으로 이루어졌다면 MOVAG의 크기는 최대가 될 것이다.

3.3.2 MOVAG 구성 알고리즘

위의 3.3.1 내용을 토대로 최종 MOVAG 도출 알고리즘을 제안하면 다음과 같다.

[Step1] 주어진 다치조합논리함수의 입력 변수로부터 최적의 입력행렬을 선정한다.

[Step2] 선정된 입력행렬의 i레벨변수에서 절점을 p^m으로 확장한다.

[Step3] 입력행렬을 레벨별로 나누어서 각각의 레벨에 해당하는 MOVAG의 절점을 단계2를 토대로 도시한다. 이 때 주어진 다치논리함수식에서 첫 번째 레벨의 입력 변수만을 고려하여 출력값이 나오면 0,1,2,…,p-1를 도출한다.

[Step4] 이외의 경우에는 성질 3-2와 같이 *로 표시하여 다음 레벨의 입력 변수와 상관하여 비교한다.

[Step5] 마지막 레벨에서는 상위 레벨변수배열에 의해 축약된 함수식을 고려하여 다치조합논리함수 출력값을 도출한다.

[Step6] 최종 MOVAG를 구성한다.

위의 MOVAG 도출 알고리즘을 흐름도로 나타내면 다음의 그림 6과 같다.

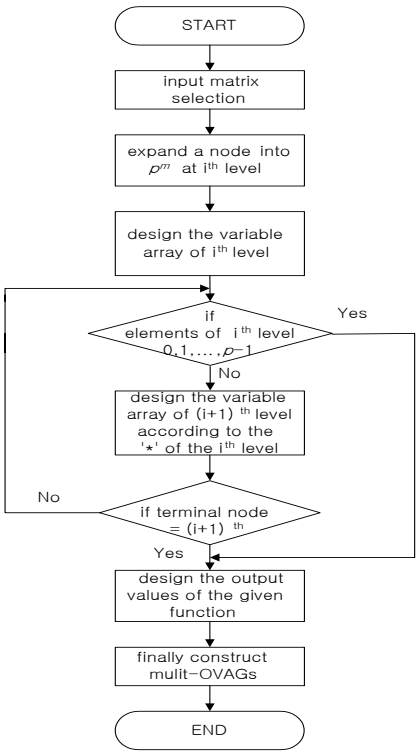


그림 6. MOVAG 구성 알고리즘의 흐름도
Fig. 6. A flowchart of the MOVAG construction algorithms

3-4 MOVAG의 다치논리 함수 구성

다음 예에 주어진 간단한 3치 3변수 조합논리함수를 진리표와 TDD(Ternary DD), ROTDD(Reduced ordered TDD)로 표현하고 제안한 알고리즘에 적용하여 MOVAG를 구성하면 다음과 같다.

예1) 3치 조합논리함수 $F=x \vee y \vee z$

입력행렬 : $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$

(a) 주어진 3치함수를 아래의 그림 7과 같은 TDD로

나타내었고 TDD를 그림 8과 같은 ROTDD로 다시 변환하여 절점의 수를 최대한으로 축약시켰다.

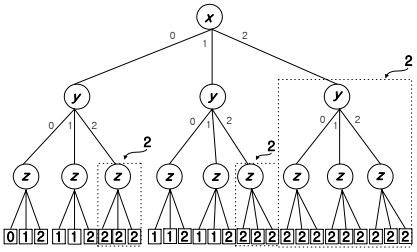


그림 7. 3치 함수 $F=x \vee y \vee z$의 TDD
Fig. 7. TDD of ternary function $F=x \vee y \vee z$

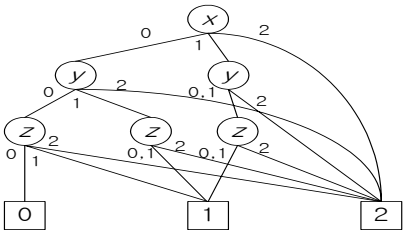


그림 8. 3치 함수 $F=x \vee y \vee z$의 ROTDD
Fig. 8. ROTDD of ternary function $F=x \vee y \vee z$

(b) 주어진 3치함수의 입력행렬을 선정한 후 다음의 그림 9와 같이 MOVAG를 구성하였다.

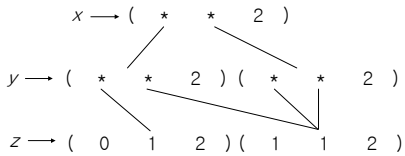


그림 9. 3치 함수 $F=x \vee y \vee z$의 MOVAG
Fig. 9. MOVAG of ternary function $F=x \vee y \vee z$

위 예의 결과 TDD의 절점수는 13개, ROTDD의 절점수는 6개, MOVAG의 절점수는 5개로 절점이 감소된 것을 볼 수 있다.

예2) 3치 조합논리함수 $F=(x \vee y) \wedge z$

입력행렬 : $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$

(a) 주어진 3치함수 $F=(x \vee y) \wedge z$를 그림 10과 같은 TDD로 나타내었고 TDD를 그림 11과 같은 ROTDD로 다시 변환하여 절점의 수를 축약시켰다.

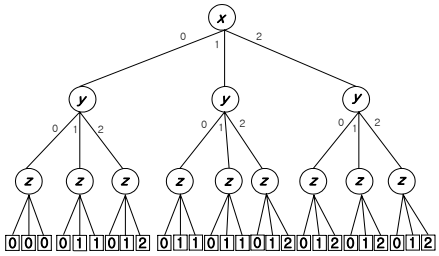


그림 10. 3치 함수 $F=(x \vee y) \wedge z$의 TDD
Fig. 10. TDD of ternary function $F=(x \vee y) \wedge z$

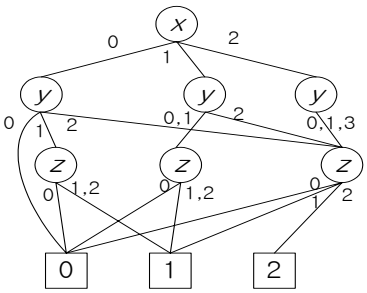


그림 11. 3치 함수 $F=(x \vee y) \wedge z$의 ROTDD
Fig. 11. ROTDD of ternary function $F=(x \vee y) \wedge z$

(b) 주어진 3치 3변수 함수의 입력행렬을 선정한 후 다음의 그림 12와 같이 MOVAG를 구성하였다.

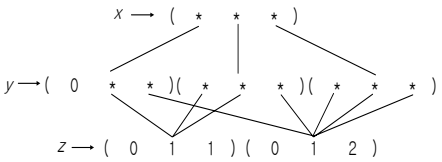


그림 12. 3치 함수 $F=(x \vee y) \wedge z$의 MOVAG
Fig. 12. MOVAG of ternary function $F=(x \vee y) \wedge z$

위 예2의 결과를 볼 때 TDD의 절점수는 13개이고 ROTDD의 절점수는 7개이다. 그러나 MOVAG의 절점수는 6개로 절점이 감소된 것을 볼 수 있다. 특히, 다변수 함수에서는 절점수의 감소 효과는 더욱 향상된다. 다음 예3)과 예4)에서는 3치 4변수의 함수를 예를 들어 변수가 증가했을 때의 절점 수 감소 정도를 나타내었다.

예3) 3치 조합논리함수 $F=(w \vee x) \wedge (y \vee z)$

(a) 진리표를 통해 TDD를 구성하면 $(p^n-1)/(p-1)$에 의해 40개의 절점을 갖는다. TDD의 절점수를 함축하여 ROTDD로 변환하면 다음의 그림 13과 같다.

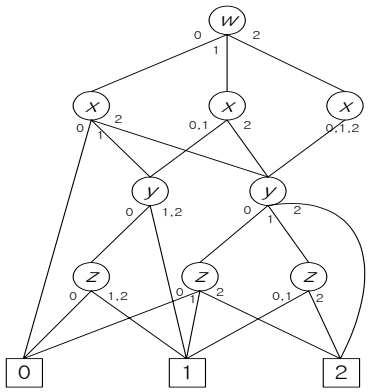


그림 13. 3치 함수 $F=(w \vee x) \wedge (y \vee z)$의 ROTDD
Fig. 13. ROTDD of ternary function
$F=(w \vee x) \wedge (y \vee z)$

(b) 3치함수 $F=(w \vee x) \wedge (y \vee z)$를 MOVAG로 구성하기 위해 다음과 같이 입력행렬을 선정하였고 그림 14와 같이 MOVAG를 구성하였다.

입력행렬 ; $\begin{bmatrix} w & x \\ y & z \end{bmatrix}$

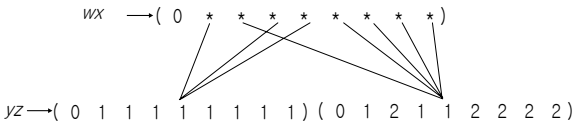


그림 14. 3치함수 $F=(w \vee x) \wedge (y \vee z)$의 MOVAG
Fig.14. MOVAG of ternary function
$F=(w \vee x) \wedge (y \vee z)$

예4) 3치 조합논리함수 $F=(w \wedge x) \vee (y \wedge z)$

(a) 진리표를 통해 TDD를 구성하면 $(p^n-1)/(p-1)$에 의해 40개의 절점을 갖는다. TDD를 ROTDD로 변환하면 다음의 그림 15와 같다.

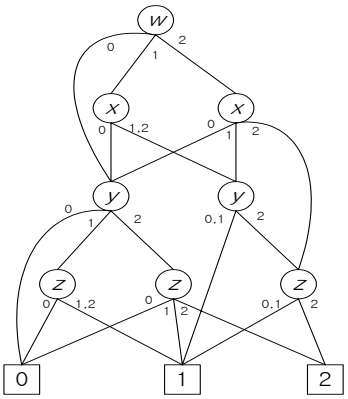


그림 15. 3치함수 $F=(w \wedge x) \vee (y \wedge z)$의 MOVAG
Fig.15. MOVAG of ternary function
$F=(w \wedge x) \vee (y \wedge z)$

(b) 3치함수 $F=(w \wedge x) \vee (y \wedge z)$를 MOVAG로 구성하기 위해 다음과 같이 입력행렬을 선정하였고 그림 16과 같이 MOVAG를 구성하였다.

입력행렬 ; $\begin{bmatrix} w & x \\ y & z \end{bmatrix}$

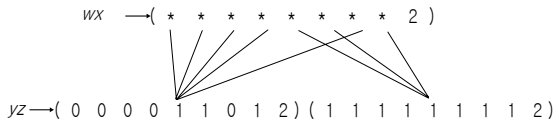


그림 16. 3치함수 $F=(w \wedge x) \vee (y \wedge z)$의 MOVAG
Fig.16. MOVAG of ternary function
$F=(w \wedge x) \vee (y \wedge z)$

위의 예3)과 예4)에서 ROTDD와 MOVAG의 절점수의 관계는 예1)과 예2)의 경우보다 많은 차이가 있다. 그 이유는 예3과 예4의 MOVAG는 입력행렬선정 알고리즘에 의해 최적의 입력행렬을 선정하였기 때문이다. 즉, 하나의 레벨에 2변수를 할당하므로 TOVAG의 크기는 총 2개의 레벨로 구성된 ROTDD에 비해 절점수의 감소율이 더 높다고 말할 수 있다.

3-5 MOVAG의 회로 설계

지금까지 주어진 3치 논리 함수를 결정도, 그리고 MOVAG로 구성한 후 각각의 구성 방법을 통한 절점수의 감소를 비교하였다.

MOVAG의 회로 설계에 사용되는 3치 T-게이트의 내부 회로는 그림 17과 같다.

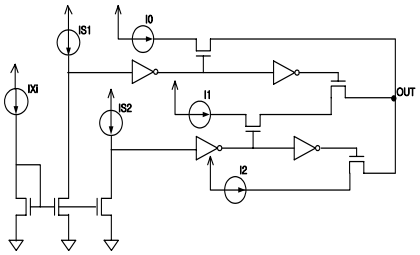


그림 17. 3치 T-게이트의 기본 회로
Fig. 17. Basic circuit of ternary T-gate

3치 T-게이트를 사용하여 결정도와 MOVAG를 회로로 구현하였다.

예5) 예1의 (b)MOVAG의 회로 설계.
주어진 3치 함수로부터 MOVAG 구성알고리즘을 이용하여 구성하였고 다음의 그림 18과 같이 T-게이트를 사용하여 회로를 설계하였다.

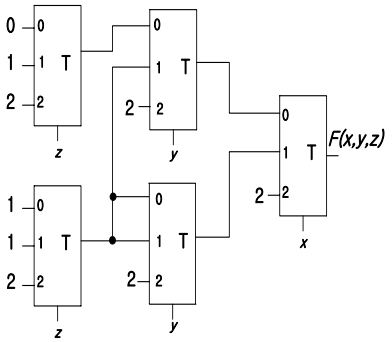


그림 18. 예1)(b)의 회로 설계
Fig. 18. Circuit design of *ex*1)(b)

예6) 예2의 (b)MOVAG의 회로 설계
예2)에서 주어진 함수를 아래의 그림 19와 같이 T-게이트를 사용하여 회로를 설계하였다.

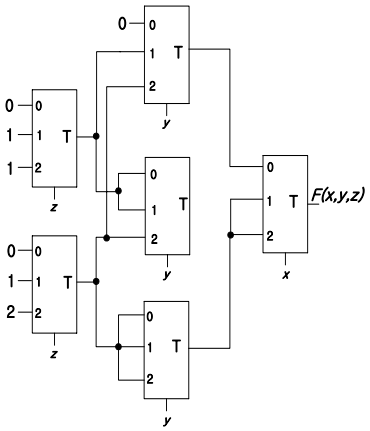


그림 19. 예2)(b)의 회로 설계
Fig. 19. Circuit design of *ex*2)(b)

위의 예3과 예4의 회로는 MOVAG 구성방법을 사용하여 회로를 설계하였다. MDD와 ROMDD, 그리고 MOVAG의 절점수 만큼의 T-게이트 소자가 사용되므로 회로 설계 시 MOVAG의 구성방법이 유용하다.

IV. 비교 및 검토

본 장에서는 본 논문에서 제안한 MOVAG를 기존의 MDD와 비교하여 각각의 특성을 살펴보고 MOVAG 구성 알고리즘을 이용한 다치논리함수 구성 방법과 MDD를 이용한 구성 방법의 장단점을 서술하였다. 그리고 주어진 예1)과 예2)의 회로를 0.35um 2-poly 4-metal CMOS parameter를 이용하여 모의 실험한 결과를 나타내었다.

5-1 MOVAG과 MDD의 비교

최근의 회로 설계 과정에서 가장 중요한 것은 설계자가 주어진 함수를 통해 기존의 방법을 가지고 회로 설계를 할 때 시간 단축과 설계 비용의 절감이라고 말할 수 있다.

이 때 기존의 MDD와 본 논문에서 제안한 MOVAG를 비교해 보면 절점수가 무한정 늘어나는 다치논리함수에 대한 MDD를 회로 설계하기란 설계자에게 있어 무척 어려운 작업이다. 그러므로 MDD의 절점수를 최대한으로 축소시켜야 할 필요가 있다. 그래서 절점을 순서화 시켜 변수의 순서에 따른 복잡도를 최대한 감

소시킨 후에 각 절점의 출력값이 동일한 것을 축약시켜 ROMDD를 구성해야 한다. ROMDD를 구성한 후 회로 설계를 하게 된다. 즉, MDD에 의한 회로 설계 방법은 매우 복잡한 과정을 거쳐야 한다. 그러나 본 논문에서 제안한 MOVAG는 주어진 다치논리함수를 입력행렬선정 알고리즘에 의해 입력행렬을 선정한 후 입력행렬의 각각의 레벨에 절점을 할당한 후 출력값 배열 그래프를 이용하여 주어진 다치논리함수에 대한 MOVAG를 구성알고리즘에 의해 구성한다. 그리고 바로 MOVAG를 회로 설계한다.

흔히, 다변수함수에서 ROMDD로 변환시 문제시되던 MDD의 절점 수의 무한 증가와 MDD를 함축시키는 과정에서 설계자는 많은 시간과 노력을 감수해야만 했다. 그러나 MOVAG는 다변수함수에서 MDD의 변환이 불가능한 상황에서 변환을 가능하게 했다. 즉, MOVAG는 다변수함수에서 더욱 유용하다고 말할 수 있다.

5-2. 모의 실험 결과

예1)과 예2)의 회로를 모의 실험한 결과 파형을 다음의 그림 20, 21, 22에 나타내었다. 공정은 0.35um 2-poly 4-metal CMOS parameter 공정을 이용하였다.

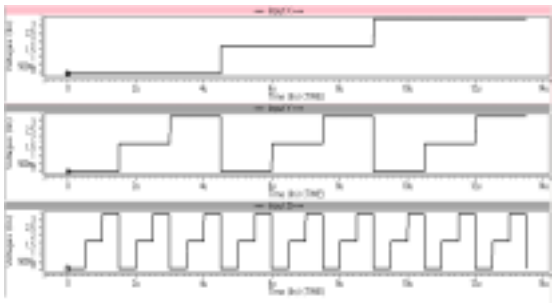


그림 20. 예1)과 예2)의 함수 입력 파형(x, y, z)
Fig. 20. Function input wave of ex1) and ex2)(x, y, z)

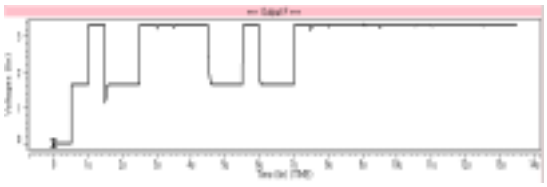


그림 21. 예1) $F=x \vee y \vee z$의 출력 파형
Fig. 21. Output wave of ex1) $F=x \vee y \vee z$

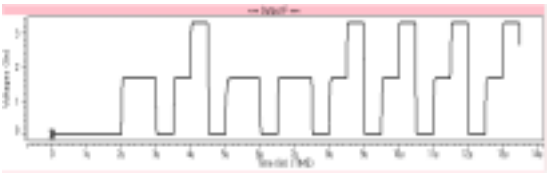


그림 22. 예2) $F=(x \vee y) \wedge z$의 출력 파형
Fig. 22. Output wave of ex2) $F=(x \vee y) \wedge z$

5-3 각각의 설계된 회로의 비교 및 검토

지금까지 예제에서 다루었던 다치조합논리함수로부터 구성된 MDD, ROMDD, MOVAG의 절점수를 다음 표 5에 각각 나타내었다.

표 5. 구성 방법에 의한 절점수(T-gate) 비교
Table 5. Comparison of node numbers(T-gate) by each construction method

	MDD	ROMDD	MOVAG
예1)	13	6	5
예2)	13	7	6
예3)	40	9	3
예4)	40	8	3

위의 표 5의 결과로부터 얻을 수 있는 것은 다변수함수의 회로 설계 시 함수구성방법 중 하나인 결정도를 사용한다면 MDD에서 ROMDD를 구성하는 과정이 복잡하기 때문에 설계자는 많은 시간이 필요하게 되므로 본 논문에서 제안한 MOVAG를 사용하면 절점수의 감소뿐 아니라 시간과 노력도 절감할 수 있게 된다.

V. 결 론

본 논문에서는 기존의 다치논리시스템에 도입되고 있는 결정도의 단점을 보완한 새로운 형태의 그래프 즉, MOVAG를 제안하여 다치논리시스템에 적용, T-게이트를 사용하여 회로 설계까지의 과정을 논하였다.

기존의 방법의 단점들을 보완하기 위해 그래프 이론의 한 방향인 MOVAG를 제안하여 다변수함수에서도

복잡도의 영향을 받지 않으면서 소자를 적게 사용하여 회로를 설계할 수 있도록 하였다.

본 논문에서의 MOVAG에서 가장 핵심이 되는 알고리즘인 주어진 함수로부터 입력행렬을 선정하는 입력행렬선정 알고리즘과 MOVAG를 구성하는 MOVAG 구성 알고리즘을 제안하여 순서 의존도에 영향을 주는 입력행렬을 최적의 순서화로 선정하여 절점을 축약시킬 수 있었고 MOVAG 구성 알고리즘으로 좀 더 쉽게 MOVAG를 구성할 수 있다.

MOVAG의 구성 알고리즘과 3장에서 논의하였던 MOVAG의 축약 알고리즘, 그리고 다치논리 연산에 의해 보다 적은 절점을 사용하여 함수를 구성할 수 있었다. 이렇게 다치논리시스템에 적용된 MOVAG를 4장에서 T-게이트를 사용하여 회로를 설계하여 MOVAG의 함수구성방법이 복잡도나 비용면에서 MDD보다 우수하다는 타당성을 입증하였다. 추후 연구 과제는 본 논문에서 제안한 MOVAG는 다치논리시스템의 수학적으로 Post 대수의 개념에만 적용되었으므로 Galois 연산이나 모듈러 연산의 개념을 적용하여 다치논리시스템의 전반적인 영역에 적용하는 연구가 지속적으로 필요하다.

참고문헌

[1]K.C.Smith, "The Prospects for Multi-valued Logic : A Technology and Applications View", IEEE Trans. on Comput. vol. C-30. pp. 619-634, 1981.

[2]T.Hanyu, M.Kameyama, T.Higuchi, "Prospects of Multiple-Valued VLSI Processors", IEICE Trans. Electron, vol. E76-C, No.3, pp.383-392, March 1993.

[3]K.C.Smith, "Multiple-valued logic : A Tutorial and Appreciation", COMP. mag., pp. 17-27, April. 1988.

[4]S.B.Aker, "Binary Decision Diagrams", IEEE Trans. Comput. vol. C-27, no.6, pp.509-516, Jun. 1978.

[5]R.E.Bryant, "Graph-Based Algorithms for Boolean Function manipulations", IEEE Trans. Comput. vol. C-35, no.8, pp.677-601, Aug. 1986.

[6]D.M.Miller, "Multiple-Valued Logic Design Tools", IEEE Proc. of Symp. on MVL, Sacramto, California, pp.2-11, May. 1993.

[7]Honghai Jiang, Jay C. Majithia, "Suggestion for a New Representation for Binary Function", IEEE Trans. Comput. vol.45, no.12, pp.1445-1449, December. 1996.

[8]Masahiro FUJITA, "Evaluation and Improvements of Boolean Comparison Method Based on Binary Decision Diagrams." IEEE Proc. CAD. pp.2-5, September,1998.

[9]D.Green, *"Modern logic Design"*, Addison-Wesley publishing Company, pp. 211-215. 1988.

[10]George Epstein, *"Mulitple-valued logic design an introduction"*, Institute of Physics Publishing, pp. 29-35. 1993.

[11]Zvi Kohavi, *"Switching and Finite Automata Theory"*, McGraw-Hill Book Company, Inc. pp.53-57. 1978.

저 자 소 개

윤 병 희 (尹 炳 熙, Byoung-Hee Yoon)

제 7 권 제 1호 논문 03-01-05 참조

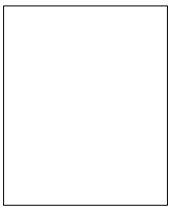
인하대학교 전자공학과 박사수료

김 흥 수(金 興 壽, Heung-Soo Kim)

제 7 권 1호 논문 03-01-05 참조

인하대학교 전자공학과 교수

박 수 진 (朴 秀 珍, Soo-Jin Park)



2002년 2월 : 건양대학교 정보통신공학과 졸업(공학사)

2004년 2월 : 인하대학교 대학원 전자공학과 졸업(공학석사)

2004년 2월 ~ 현재 : 인하대학교 대학원 전자공학과 박사과정

주관심분야 : PLL설계, ADC/DAC설계, Fractional-N 설계