論文97-34C-12-6

Reed-Muller 전개식에 의한 범용 논리 모듈 U_f의 다치 논리 회로의 최적 합성에 관한 연구

(A Study on Optimal Synthesis of Multiple-Valued Logic Circuits using Universal Logic Modules U_f based on Reed-Muller Expansions)

崔 在 碩 *, 韓 永 煥 **, 成 賢 慶 **

(Jai Sock Choi, Young Hwan Han, and Hyeon Kyeong Seong)

요 약

본 논문에서는 3변수 3치 Reed-Muller 전개식에 기초한 만능 논리 모듈(universal logic modules; ULM) U_f를 사용하여 다치 논리 회로의 최적 합성 알고리듬을 제시하였다. 제시된 다치 논리 회로의 최적 합성 알고리듬은 Reed-Muller 전개식의 계수에 대하여 모든 변수의 차수를 검사하여 ULM U_f 모듈의 수를 최소화하는 최적의 제어 입력 변수의 순서를 결정한다. 최적 제어 입력 변수의 순서는 회로 비용 행렬을 사용하여 Reed-Muller 전개식에 기초한 ULM U_f 모듈로 구성되는 다치 논리 회로의 실현에 사용된다. 제시된 방법은 최적 제어 입력 변수를 찾는데 유일하게 단위 시간내에 수행되며, 컴퓨터 프로그램이 가능하고, 프로그래밍 수행 시간이 $O(p^n)$이다.

Abstract

In this paper, the optimal synthesis algorithm of multiple-valued logic circuits using universal logic modules(ULM) U_f based on 3-variable ternary Reed-Muller expansions is presented. We check the degree of each variable for the coefficients of Reed-Muller expansions and determine the order of optimal control input variables that minimize the number of ULM U_f modules. The order of optimal control input variables is utilized the realization of multiple-valued logic circuits to be constructed by ULM U_f modules based on Reed-Muller expansions using the circuit cost matrix. This algorithm is performed only unit time in order to search for the optimal control input variables. Also, this algorithm is able to be programmed by computer and the run time on programming is $O(p^n)$.

I. 서 론

오늘날 사용되고 있는 디지털 시스템은 대부분 2진 논리 이론을 기초로 하고 있으며, 반도체 집적기술의 발달로 인하여 칩의 집적도가 비약적으로 증가하고 회로의 복잡도가 날로 높아가고 있다. 이러한 VLSI 시스템에 심각하게 대두되고 있는 단자수 제한 문제, 단자간 상호 연결 문제, 보다 많은 정보량의 처리 문제와 연산 속도의 제한성이라는 근본적인 문제에 직면하게 되었으며, 이러한 문제점을 해결하는 가장 효과적인 방법 중의 한가지 방법이 다치 논리 회로가 될 것으로 예상된다[1,2,3].

* 正會員, 仁荷大學校 電子工學科
(Dept. of Electronic Eng., In Ha University)
** 正會員, 尙志大學校 電算學科
(Dept. of Computer Science, Sang Ji University)
接受日字:1997年4月12日, 수정완료일:1997年11月27日

VLSI 설계에서 모듈 구조와 규칙적 상호 연결이 중요한 설계 개체이다. 다치 논리 회로 합성에 대한 연구가 지난 십 수년간 수행되어 왔으나 불행하게도 이들 회로 합성들은 불규칙한 회선 경로 선택, 복잡한 제어 문제, 비모듈화 구조 및 병발성의 부족 때문에 VLSI 구조의 설계에 부적합하였다[4].

다치 논리 회로는 함수적 완전성, 유연성, 회로의 실현을 만족하여야 한다. Reed-Muller 전개식(Reed-Muller Expansion; RME)은 이러한 성질을 만족하므로, Reed-Muller 전개식을 이용한 다치 논리 회로의 설계에 관한 연구가 활발히 진행되고 있다[5-9]. 대부분의 많은 다치 논리 회로 합성에 관한 연구는 규칙적인 셀 배열과 간단하고 규칙적인 상호 연결 형식을 갖고 논리 회로를 합성 할 수 있는 동일한 기본 빌딩 블록을 사용하여 논리 함수를 실현하고 있으며, 대표적으로 만능 논리 모듈(universal logic modules; ULM)을 사용한다. 만능 논리 모듈은 논리적의 논리합(sum of product; SOP)에 대응되는 T 게이트와 Reed-Muller 전개식에 대응되는 ULM U_f 모듈의 두가지 종류가 있다[10].

기본 모듈인 ULM을 이용하는 다치 논리 회로의 합성은 나무 구조를 가지며, 이러한 다치 논리 회로 합성은 다치 논리 함수의 스위칭 함수에 효과적으로 적용할 수 있다. 나무 구조 형식에서 요구되는 최대 ULM U_f 모듈의 수가 $(p^n-1)/(p-1)$이며, 제어 입력 변수의 순서를 어떻게 선택하는가에 따라서 ULM U_f 모듈의 수를 상당히 줄일 수 있으므로 최적의 제어 입력 변수의 순서를 선택하는 것이 매우 중요하다. 그러므로 선택된 제어 입력 변수의 순서에 따라서 다치 논리 회로를 합성하면 Reed-Muller 전개식에 기초한 ULM U_f 모듈의 간단한 나무 구조의 다치 논리 회로를 실현할 수 있다[11,12].

여러 가지 다치 논리 회로의 합성 방법이 제안되었으나 이러한 방법들은 다치 논리 회로의 합성 방법에서 약간의 단점들을 가지고 있다. Tokmen과 Hurst[11]가 제시한 방법은 계산되는 역 행렬의 원소수가 줄어 데이터를 저장하는데 필요한 메모리 공간을 감소시키며, Reed-Muller 전개식의 계수를 계산하는데 있어서 가산의 수가 적어지나 각 제어 입력 변수에 대한 Reed-Muller 전개식의 계수를 효과적으로 선택하는 것이 복잡하며, 최적 제어 입력 변수의 순서를 할당하는 것이 어려워서 다치 논리 회로의 설계가 복잡해지는 단점이 있다. Hu와 Wu[12]가 제시한 알고리듬은 Kronecker 적을 사용하므로 변환 행렬을 유도하는 것이 복잡하며, Reed-Muller 전개식의 간략화에 있어서 계수 맵을 사용하기 때문에 3변수 이상인 경우에는 비효과적이다.

Hong과 Fei[13]는 3치 Reed-Muller 전개식에서 변수의 수가 많을 때 그레이 코드 순서에서 Green의 단계적 흐름 그래프의 계산보다 더 적은 계산 비용을 가지나 병렬 계산에서 순환 알고리듬을 사용하므로 변환 행렬이 복잡하고, 기수가 높은 Reed-Muller 전개식에서는 다치 논리 회로 합성 방법이 어렵게 되고 회로가 복잡해 진다. 또한 Fei와 Hong[14]는 Reed-Muller 전개식에 대하여 Kronecker 적을 사용하여 계수를 찾는 방법을 제시하였다. 그러나 Kronecker 적을 사용하므로 계산 비용이 높고 다변수 일 경우에는 회로가 매우 복잡해지고 회로 비용이 높다.

Drechsler 등[15]은 Reed-Muller 전개식의 부울 함수를 순서화된 함수적 결정도(ordered functional decision diagrams; OFDDS)를 사용하여 AND/EXOR 표준 표현식으로 간략화하는 방법을 제시하였으며, 이 방법은 다출력의 회로를 취급할 수 있으나 다치 Reed-Muller 전개식으로 확장할 수 없다. 또한, Dubrov와 Muzio[16]는 2보다 큰 소수(素數)인 m을 갖는 논리적의 논리합(sum of product; SOP) 형의 m치 함수를 실현하는 일반화된 Reed-Muller 회로의 테스트 가능성을 연구하였다. 이들은 오류 검출에 요구되는 테스트의 수와 테스트의 일반성인 두가지 문제점을 고려하였다.

실제로 나무 구조를 갖는 ULM U_f의 다치 논리 회로의 합성은 제어 입력 변수의 최적 할당을 선택하는 것이다. 일반적으로 n 변수 함수는 n! 종류의 할당 방법이 있다. 그러므로 n의 값이 크면 '조합 급증'이 발생한다.

본 연구는 제어 입력 변수의 최적 순서를 할당하는 한가지 알고리듬을 제시한다. 이 알고리듬은 Reed-Muller 전개식의 계수를 갖는 모든 변수의 차수를 변수 차수표를 사용하여 각 변수의 차수를 조사하므로서 최적의 제어 입력 변수의 순서를 선택한다. 이러한 알고리듬에 의해 선택된 제어 입력 변수의 최적 순서를 Fei와 Zhuang[17]이 제시한 회로 비용 행렬을 이용하여 최적의 나무 구조를 갖는 ULM U_f 모듈의 다치 논리 회로를 실현한다.

Ⅱ. ULM U_f 모듈을 사용한 3치 Reed-Muller Expansion의 실현

1. ULM U_f 모듈의 기본적 개념

본 절에서는 다치 논리 회로의 합성에 필요한 수학적 배경을 설명하며, 다치 논리 회로에서 논리값이 집합 $R=\{0,1,2,\ldots,p\}$이면 다음과 같은 연산 함수를 표현할 수 있으며, 이를 이용한 만능 논리 모듈인 ULM U_f 모듈에 대하여 설명한다.

(1) mod-p 가산 함수

p치 2변수 x와 y에 대하여 mod-p 가산을 수행하는 함수는 다음과 같이 표현할 수 있으며, 그림 1은 mod-p 가산기를 나타낸다.

$$x \oplus y = (x+y)_{\bmod p} \tag{1}$$

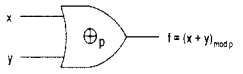

그림 1. mod-p 가산기
Fig. 1. mod-p adder.

(2) mod-p 승산 함수

p치 2변수 x와 y에 대하여 mod-p 승산을 수행하는 함수는 다음과 같이 표현할 수 있으며, 그림 2는 mod-p 승산기를 나타낸다.

$$x \cdot y = (x \times y)_{\bmod p} \tag{2}$$

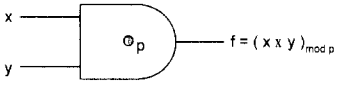

그림 2. mod-p 승산기
Fig. 2. mod-p multiplier.

(3) 만능 논리 모듈 ULM U_f

p치 단일 변수 x에 대하여 mod-p 가산 함수와 mod-p 승산 함수를 이용한 만능 논리 모듈 ULM U_f 함수의 표현식이 다음과 같으며, 그림 3(a)는 mod-p 가산기와 mod-p 승산기를 이용하여 구성한 ULM U_f 모듈의 회로이며, 그림 3(b)는 ULM U_f 모듈의 기호이다. $c_i=\{0,1,2,\ldots,p\}$이다.

$$U_f(c_0,c_1,c_2,x) = c_0 \oplus c_1 \cdot x \oplus c_2 \cdot x^2 \tag{3}$$

2. 3치 Reed-Muller 전개식의 실현

다치 논리 시스템에서 n변수 p치 함수는 다음과 같이 일반적인 Reed-Muller 전개식으로 표현할 수 있다.

$$f(x_1,x_2,\ldots,x_n)=\sum_{i=0}^{p^n-1} c_i\left[\prod_{j=1}^{n} x_j^{i\cdot j}\right] \tag{4}$$

여기서 $c_i \in GF(p)$이고 $x_j^{i\cdot j}$는 변수 x_j의 $i\cdot j$의 승수(power)이다.

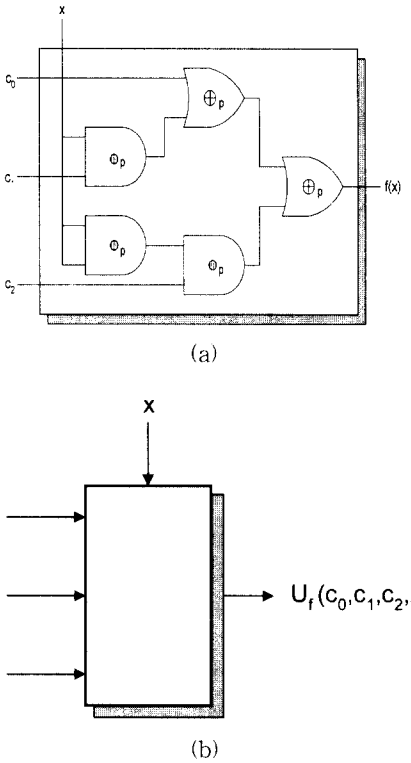

그림 3. 만능 논리 모듈 ULM U_f
(a) ULM Uf의 회로 (b) ULM Uf의 기호
Fig. 3. The universal logic module ULM U_f.
(a) circuit of ULM U_f (b) symbol of ULM U_f

식 (4)에 의해서 n변수 3치 함수는 다음과 같이 Reed-Muller 전개식으로 나타낼 수 있다.

$$f(x_1,x_2,\ldots,x_n)=c_0\cdot x_1^0\cdot x_2^0\cdot\ldots\cdot x_n^0\oplus c_1\cdot x_1^0\cdot x_2^0\cdot\ldots\cdot x_n^1\oplus\ldots. \\ \oplus c_j\cdot x_1^{e_1}\cdot x_2^{e_2}\cdot\ldots\cdot x_n^{e_n}\oplus c_{3^n-1}\cdot x_1^2\cdot x_2^2\cdot\ldots\cdot x_n^2 \tag{5}$$

여기서, $x_i^0=1, x_i^1=x_i, x_i^2=x_i\cdot x_i\ (i=1,2,\ldots,n)$,

$c_j\in\{0,1,2\},\ (j=0,1,2,\ldots,3^n-1)$,

$e_i\in\{0,1,2\}$,

$(j)_{10}=(e_1e_2\ldots e_n)_3$

식 (5)의 n변수 3치 Reed-Muller 전개식은 그림 3에서 보인 것처럼 만능 논리 모듈 ULM U_f 모듈을 사용하여 나무 구조 형식으로 실현할 수 있으며, 그림 3에서 요구되는 최대 ULM U_f 모듈의 수가 $(p^n-1)/(p-1)$에서 p=3이므로 $(3^n-1)/(3-1)$이다[12,17]. 그림 4에서 보인 것처럼 n번째 단을 나무의 뿌리로서 정의한다.

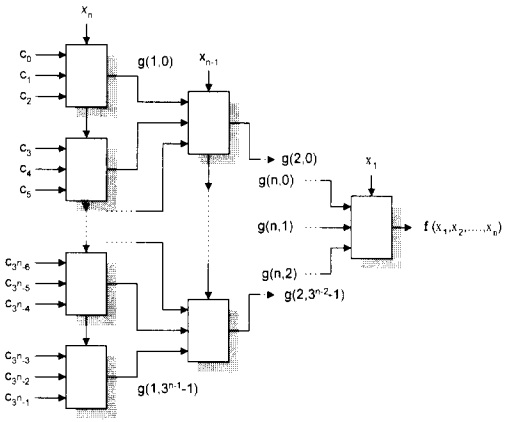

그림 4. ULM U_f 모듈을 사용한 n 변수 3치 함수의 실현

Fig. 4. Realization of n-variable ternary functions using ULM U_f modules.

Ⅲ. Reed-Muller 전개식의 계수에 의한 회로 비용 행렬

[정의 1] n단으로 구성된 나무 구조의 다치 논리 회로 합성에서 임의의 변수 x_k가 i-번째 중간 함수의 ULM U_f 모듈의 출력 $g_k(i,j)$ $(i=0,1,2,\ldots,n$ 와 $j=0,1,2,\ldots 3^n-1)$를 정의하며, $g_k(0,j)=c_j$ 이다[17].

x_n이 첫 번째 제어 입력 변수라고 가정하자. 그러면 식 (5)는 다음과 같이 표현할 수 있다.

$$f(x_1,x_2,\ldots,x_n)=(c_0\oplus c_1\cdot x_n\oplus c_2\cdot x_n^2)\oplus(c_3\oplus c_4\cdot x_n\oplus c_5\cdot x_n^2)x_{n-1}\oplus\ldots\ldots\oplus(c_{3^n-3}\oplus c_{3^n-2}\cdot x_n\oplus c_{3^n-1}\cdot x_n^2)x_1^2\cdot x_2^2\cdot\ldots\cdot x_{n-1}^2 \tag{6}$$

[정의 1]에 따라서 식 (6)은 다음과 같이 표현할 수 있다.

$$f(x_1,x_2,\ldots,x_n)=g_n(1,0)\oplus g_n(1,1)x_{n-1}\oplus g_n(1,2)x_{n-1}^2\oplus\ldots\ldots\oplus g_n(1,3^{n-1}-1)x_1^2\cdot x_2^2\cdot\ldots\cdot x_{n-1}^2 \tag{7}$$

식 (6)과 식 (7)을 비교하여 다음과 같은 표현식을 얻을 수 있다.

$$\begin{aligned}g_n(1,0)&=c_0\oplus c_1\cdot x_n\oplus c_2\cdot x_n^2\\ g_n(1,1)&=c_3\oplus c_4\cdot x_n\oplus c_5\cdot x_n^2\\ &\cdots\cdots\\ g_n(1,3^{n-1}-1)&=c_{3^n-3}\oplus c_{3^n-2}\cdot x_n\oplus c_{3^n-1}\cdot x_n^2\end{aligned} \tag{8}$$

식 (8)에 의해 Reed-Muller 전개식의 계수들에 대한 회로 비용 행렬을 다음과 같이 표현할 수 있다.

$$[\,g_n(1,0)\;g_n(1,1)\;\ldots\ldots\;g_n(1,3^{n-1}-1)\,]=[1\;\;x_n\;\;x_n^2]\begin{bmatrix}c_0 & c_3 & \ldots\ldots & c_{3^n-3}\\ c_1 & c_4 & \ldots\ldots & c_{3^n-2}\\ c_2 & c_5 & \ldots\ldots & c_{3^n-1}\end{bmatrix} \tag{9}$$

[정의 2] 임의의 변수 x_m이 q-번째 제어 입력 변수이면, $[\,g_m(q,0)\,g_m(q,1)\,g_m(q,3^{n-1}-1)\,]$를 갖는 중간 함수 $[G_m(q)]$를 정의하며, 이것은 회로 비용 행렬 $[G_m(q-1)]$로부터 구하여진다.

[정의 2]에 의해 만약 임의의 변수 x_i가 첫 번째 제어 입력 변수이면, Reed-Muller 전개식의 계수의 중간 함수를 표현할 수 있고 다음과 같은 회로 비용 행렬을 얻을 수 있다.

$$\begin{aligned}[G_i(1)]&=[g_i(1,0)\,g_i(1,1)\ldots\ldots g_i(1,3^{n-1}-1)]\\ &=[1\;x_i\;x_i^2]\begin{bmatrix}g(0,L^{(0)}) & g(0,L^{(1)}) & \ldots & g(0,L^{(k)}) & \ldots & g(0,L^{(3^{n-1}-1)})\\ g(0,M^{(0)}) & g(0,M^{(1)}) & \ldots & g(0,M^{(k)}) & \ldots & g(0,M^{(3^{n-1}-1)})\\ g(0,N^{(0)}) & g(0,N^{(1)}) & \ldots & g(0,N^{(k)}) & \ldots & g(0,N^{(3^{n-1}-1)})\end{bmatrix}\\ &=[1\;x_i\;x_i^2][G_i(0)]\end{aligned} \tag{10}$$

여기서,

$$\begin{aligned}(k)_{10}&=(e_1\;e_2\;e_3\;\ldots\ldots\;e_{i-1}\;e_{i+1}\ldots\ldots\;e_n)_3\\ (L^{(k)})_{10}&=(e_1\;e_2\;e_3\;\ldots\ldots\;e_{i-1}\;0\;e_{i+1}\ldots\ldots\;e_n)_3\\ (M^{(k)})_{10}&=(e_1\;e_2\;e_3\;\ldots\ldots\;e_{i-1}\;1\;e_{i+1}\ldots\ldots\;e_n)_3\\ (N^{(k)})_{10}&=(e_1\;e_2\;e_3\;\ldots\ldots\;e_{i-1}\;2\;e_{i+1}\ldots\ldots\;e_n)_3\end{aligned}$$

만약 임의의 변수 x_j가 두 번째 제어 입력 변수이면, 연속적으로 다음과 같은 표현식을 얻을 수 있다.

$$\begin{aligned}[G_j(2)]&=[g_j(2,0)\,g_j(2,1)\ldots\ldots g_j(2,3^{n-2}-1)]\\ &=[1\;x_j\;x_j^2]\begin{bmatrix}g(1,L^{(0)}) & g(1,L^{(1)}) & \ldots & g(1,L^{(k)}) & \ldots & g(1,L^{(3^{n-2}-1)})\\ g(1,M^{(0)}) & g(1,M^{(1)}) & \ldots & g(1,M^{(k)}) & \ldots & g(1,M^{(3^{n-2}-1)})\\ g(1,N^{(0)}) & g(1,N^{(1)}) & \ldots & g(1,N^{(k)}) & \ldots & g(1,N^{(3^{n-2}-1)})\end{bmatrix}\\ &=[1\;x_j\;x_j^2][G_j(1)]\end{aligned} \tag{11}$$

여기서,

$$\begin{aligned}(k)_{10}&=(e_1\;e_2\;\ldots\ldots\;e_{j-1}\;\;e_{j+1}\;\ldots\ldots e_{i-1}\;e_{i+1}\ldots\ldots\;e_n)_3\\ (L^{(k)})_{10}&=(e_1\;e_2\;\ldots\ldots\;e_{j-1}\;0\;\;e_{j+1}\;\ldots\ldots e_{i-1}\;e_{i+1}\ldots\ldots\;e_n)_3\\ (M^{(k)})_{10}&=(e_1\;e_2\;\ldots\ldots\;e_{j-1}\;1\;\;e_{j+1}\;\ldots\ldots e_{i-1}\;e_{i+1}\ldots\ldots\;e_n)_3\\ (N^{(k)})_{10}&=(e_1\;e_2\;\ldots\ldots\;e_{j-1}\;2\;\;e_{j+1}\;\ldots\ldots e_{i-1}\;e_{i+1}\ldots\ldots\;e_n)_3\end{aligned}$$

유사하게 만약 임의의 변수 x_u가 i-번째 제어 입력 변수이면 다음과 같은 일반적인 회로 비용 행렬을 얻을 수 있다.

$$[G_u(i)] = [g_u(i,0)\, g_u(i,1) \ldots\ldots g_u(i,\ 3^{n-i}-1)]$$
$$= [1\, x_u\, x_u^2] \begin{bmatrix} g(i-1,L^{(0)})\, g(i-1,L^{(1)}) \ldots g(i-1,L^{(k)}) \ldots g(i-1,L^{(3^n-1)}) \\ g(i-1,M^{(0)})\, g(i-1,M^{(1)}) \ldots g(i-1,M^{(k)}) \ldots g(i-1,M^{(3^n-1)}) \\ g(i-1,N^{(0)})\, g(i-1,N^{(1)}) \ldots g(i-1,N^{(k)}) \ldots g(i-1,N^{(3^n-1)}) \end{bmatrix}$$
$$= [1\, x_u\, x_u^2][G_u(i-1)] \qquad (12)$$

Ⅳ. ULM U_f 모듈에 의한 3치 Reed-Muller 전개식의 합성 알고리듬

ULM U_f 모듈를 사용하는 다치 논리 회로의 합성은 나무 구조를 구성하며, 이러한 다치 논리 회로 합성은 효과적으로 스위칭 함수에 적용할 수 있다. 이러한 나무 구조의 최적 다치 논리 회로 합성에 대하여 제어 입력 변수의 순서를 선택하는 것이 매우 중요하며, 선택된 최적 제어 입력 변수의 순서는 최적 다치 논리 회로 합성에 대하여 간단한 나무 구조를 구성할 수 있다. 그러므로 n변수 3치 Reed-Muller 전개식에 대하여 선택된 최적 제어 입력 변수의 순서가 나무 구조를 갖는 다치 논리 회로의 합성에 할당되기 때문에 ULM U_f 모듈를 사용하는 다치 논리 회로 설계에서 회로 비용이 감소될 수 있다. 회로 비용을 감소하기 위한 최적 제어 입력 변수의 순서를 선택하는 알고리듬이 다음과 같다.

[1] Reed-Muller 전개식의 계수들을 갖는 모든 변수의 차수에 관한 변수 차수표(variable degree table; VDT)를 만든다. 예를 들면, 만약 2변수 3치 논리 함수이면 이 함수의 최대 계수의 수가 9이고, 만약 3변수 3치 논리 함수이면, 이 함수의 최대 계수의 수가 27이다.

[2] Reed-Muller 전개식의 모든 계수에 대하여 각 변수의 차수를 검사하여 변수 차수표에 해당되는 차수의 열을 "γ" 기호로 표시한다. 만약 2변수이면 각 계수 열에 2개의 "γ" 기호를 가지며, 3변수이면 각 계수 열에 3개의 "γ" 기호를 갖는다.

[3] 변수 차수표에 표시된 "γ" 기호의 수를 각 변수의 차수에 대하여 계산하고, 변수 차수표의 아래에 계산된 수치를 나타낸다. 각 변수의 차수에 대하여 계산된 수의 합이 동일하다.

[4] 변수 차수표에서 차수가 0인 변수(x_i^0)들에서 "γ" 기호가 가장 많은 열의 변수를 첫 번째 제어 입력 변수로 선택할 수 있으며, 표시된 수에 따라서 제어 입력 변수가 결정될 수 있다.

[5] 표시된 기호의 수가 모든 변수에 대하여 동일하다면, 차수 1인 변수(x_i^1)에서 가장 많이 표시된 열의 변수를 첫 번째 제어 입력 변수로 할당할 수 있다.

[6] 다음의 제어 입력 변수는 선택된 변수를 제외한 모든 변수에 대하여 단계 [4]와 [5]를 반복하여 제어 입력 변수의 순서를 결정한다.

[7] 선택된 제어 입력 변수의 순서에 따라서 식 (12)의 회로 비용 행렬을 사용하여 ULM U_f 모듈의 회로를 실현할 수 있다.

위의 최적 제어 입력 변수의 순서를 선택하는 알고리듬에 대한 C++프로그램이 다음과 같다.

```
OptimalControlVariables(int n, int p) // n 변수 p치 함수의 최적 제어 입력 변수
{
   CreateArray(n); // n차원의 배열 생성
   InputCoefficient(p); // 계수 입력

   // n=3이고, p=3이라 가정하면
   for ( i=0; i<p; I++ )
        for ( j=0; j<p; j++ )
             for ( k=0; k<p; k++ )
             {
                if ( X[i][j][k] != 0 )
                {
                    switch ( i )
                    { case 0 : cost_x10++; break;   //
                      변수 x_1^0에 대한 차수 계산
                       case 1 : cost_x11++; break; //
                          변수 x_1^1에 대한 차수 계산
                       case 2 : cost_x12++; break;  //
                          변수 x_1^2에 대한 차수 계산
                    }
                    switch ( j )
                    { case 0 : cost_x20++; break;   //
                      변수 x_2^0에 대한 차수 계산
                       case 1 : cost_x21++; break;  //
                          변수 x_2^1에 대한 차수 계산
                       case 2 : cost_x22++; break;  //
                          변수 x_2^2에 대한 차수 계산
                    }
```

```
                switch ( k )
                { case 0 : cost_x30++; break;   //
                  변수 x3^0에 대한 차수 계산
                    case 1 : cost_x31--; break;   //
                      변수 x3^1에 대한 차수 계산
                    case 2 : cost_x32++; break;   //
                      변수 x3^2에 대한 차수 계산
                }
            }
        }
    if ( cost_x10 != cost_x20 != cost_x30 )
        FindMaxDegree( cost_x10, cost_x20, cost_x30
        ); // 0차수의 최대값을 찾음
    else if ( cost_x11 != cost_x21 != cost_x31 )
        FindMaxDegree( cost_x11, cost_x21, cost_x31
        ); // 1차수의 최대값을 찾음
    else
        FindMaxDegree( cost_x12, cost_x22, cost_x32
        ); // 3차수의 최대값을 찾음
}
```

위에서 제시한 Reed-Muller 전개식에 대하여 최적 제어 입력 변수의 순서를 선택하는 알고리듬을 사용하여 ULM U_f 모듈에 의한 다치 논리 회로의 합성 방법을 예를 들면 다음과 같다.

[예 1] 3변수 3치 Reed-Muller 전개식인 다음과 같은 다치 논리 함수 $f(x_1, x_2, x_3)$를 ULM U_f 모듈을 사용하여 다치 논리 회로를 실현한다.

$$f(x_1,x_2,x_3) = 2\cdot x_3 \oplus x_3^2 \oplus x_1 \oplus x_1\cdot x_3 \oplus x_1\cdot x_3^2 \oplus 2\cdot x_1\cdot x_2\cdot x_3 \oplus 2\cdot x_1\cdot x_2\cdot x_3^2 \oplus x_1\cdot x_2^2\cdot x_3 \oplus x_1\cdot x_2^2\cdot x_3^2 \oplus x_1^2\cdot x_3 \oplus x_1^2\cdot x_2\cdot x_3 \oplus 2\cdot x_1^2\cdot x_2^2\cdot x_3 \tag{13}$$

식 (13)의 3변수 3치 Reed-Muller 전개식의 계수 c_j의 벡터열을 다음과 같이 표현할 수 있다.

$$[C] = [\,0\,2\,1\;\;0\,0\,0\;\;0\,0\,0\;\;1\,1\,1\;\;0\,2\,2\;\;0\,1\,1\;\;0\,1\,0\;\;0\,1\,0\;\;0\,2\,0\,]^T \tag{14}$$

위에서 논한 최적 제어 입력 변수의 순서를 선택하는 알고리듬을 사용하여 ULM U_f 모듈에 의한 다치 논리 회로를 합성하는 절차가 다음과 같다.

[1] 식 (13)으로 부터 3변수 3치 Reed-Muller 전개식의 첫 번째 항 $2\cdot x_3$에서 변수 x_1과 변수 x_2의 차수가 각각 0이고, 변수 x_3의 차수가 1이므로 변수 차수표에서 변수 x_1의 0 차수항, 변수 x_2의 0 차수항, 변수 x_3의 1 차수항에 기호 "γ"를 표시한다.

[2] 식 (13)의 두 번째 항인 x_3^2에서 변수 x_1과 변수 x_2의 차수가 역시 각각 0이고, 변수 x_3^2의 차수가 2이다. 그러므로 변수 차수표에서 변수 x_1의 0 차수항, x_2의 0 차수항, 변수 x_1의 2 차수항에 기호 "γ"를 표시한다.

[3] Reed-Muller 전개식의 모든 계수들에 대하여 반복적으로 이러한 절차를 수행한다. 이러한 과정의 결과가 표 1에서 보인다.

표 1에서 변수 x_1의 0 차수항에서 표시된 "γ" 기호의 수가 2이고 1 차수항에서 표시된 "γ" 기호의 수가 7이고, 2 차수항에서 표시된 "γ" 기호의 수가 3이다. 그리고 변수 x_2에 대하여 0 차수항에서 표시된 "γ" 기호의 수가 6이고, 1 차수항에서 표시된 "γ" 기호의 수가 3이고, 2 차수항에서 표시된 "γ" 기호의 수가 3이다. 변수 x_3에 대하여 0, 1, 2 차수항에 표시된 "γ" 기호의 수가 각각 1, 7, 4이다. 그러므로 첫 번째 제어 입력 변수가 x_2이고, 두 번째 제어 입력 변수가 x_1이고, 세 번째 제어 입력 변수가 x_3이다.

선택된 제어 입력 변수의 순서 (x_2, x_1, x_3)에 의해서 회로 비용 행렬을 사용하여 ULM U_f 모듈의 입력 계수를 결정할 수 있다. 식 (14)의 계수 벡터열에서 회로 비용 행렬 $[\,G_1(0)\,]$, $[\,G_2(0)\,]$, $[\,G_3(0)\,]$을 다음과 같이 구할 수 있다.

$$[G_1(0)] = \begin{bmatrix} 0\,2\,1 & 0\,0\,0 & 0\,0\,0 \\ 1\,1\,1 & 0\,2\,2 & 0\,1\,1 \\ 0\,1\,0 & 0\,1\,1 & 0\,2\,0 \end{bmatrix} \tag{15}$$

$$[G_2(0)] = \begin{bmatrix} 0\,2\,1 & 1\,1\,1 & 0\,1\,0 \\ 0\,0\,0 & 0\,2\,2 & 0\,1\,0 \\ 0\,0\,0 & 0\,1\,1 & 0\,2\,0 \end{bmatrix} \tag{16}$$

$$[G_3(0)] = \begin{bmatrix} 0\,0\,0 & 1\,0\,0 & 0\,0\,0 \\ 2\,0\,0 & 1\,2\,1 & 1\,1\,2 \\ 1\,0\,0 & 1\,2\,1 & 0\,0\,0 \end{bmatrix} \tag{17}$$

표 1에서 선택된 첫 번째 제어 입력 변수가 x_2이므로 변수 x_2에 대하여 ULM U_f 모듈의 입력 계수를 다음과 같이 결정할 수 있다.

$$[G_2(1)] = [1\;\;x_2\;\;x_2^2]\,[G_2(0)]$$
$$[1\;\;x_2\;\;x_2^2]\begin{bmatrix} 0\,2\,1 & 1\,1\,1 & 0\,1\,0 \\ 0\,0\,0 & 0\,2\,2 & 0\,1\,0 \\ 0\,0\,0 & 0\,1\,1 & 0\,2\,0 \end{bmatrix}$$

$$= [0\ 2\ 1\ 1\ \ U_f(1,2,1,x_2)\ U_f(1,2,1,x_2)\ 0\ U_f(1,1,2,x_2)\ 0] \quad (18)$$

식 (18)에서 $[\ G_1(1)\ ]$ 을 구하면 다음과 같다.

$$[G_1(1)] = \begin{vmatrix} 0 & 2 & 1 \\ 1 & U_f(1,2,1,x_2) & U_f(1,2,1,x_2) \\ 0 & U_f(1,1,2,x_2) & 0 \end{vmatrix} \quad (19)$$

표 1. 식 (13)의 검사된 변수 차수표
Table 1. The variable degree table of Eq. (13) checked.

c_i	x_1			x_2			x_3		
	0	1	2	0	1	2	0	1	2
2	√			√				√	
1	√			√					√
1		√		√			√		
1		√		√				√	
1		√		√					√
2		√			√			√	
2		√			√				√
1		√				√		√	
1		√				√			√
1			√	√				√	
1			√		√			√	
2			√			√		√	
*	2	7	3	6	3	3	1	7	4
	12			12			12		

표 1에서 두 번째 제어 입력 변수가 x_1이므로 변수 x_1에 대하여 ULM U_f 모듈의 입력 계수를 다음과 같이 구할 수 있다.

$$[G_1(2)] = [1\ \ x_1\ \ x_1^2]\ [G_1(1)]$$
$$[1\ \ x_1\ \ x_1^2] \begin{vmatrix} 0 & 2 & 1 \\ 1 & U_f(1,2,1,x_2) & U_f(1,2,1,x_2) \\ 0 & U_f(1,1,2,x_2) & 0 \end{vmatrix}$$
$$= [x_1\ \ U_f(2,\ U_f(1,2,1,x_2),\ U_f(1,1,2,x_2),\ x_1)\ \ U_f(1,\ U_f(1,2,1,x_2),0,x_1] \quad (20)$$

그러므로 ULM U_f 모듈을 사용하여 식 (20)에 대한 다치 논리 회로를 합성하면 그림 5와 같은 회로를 실현할 수 있다.

그림 5에서 식 (13)과 같은 3변수 3치 다치 논리 함수를 회로 실현하면 ULM U_f가 5개 필요하다. 표 2는 식 (13)의 3변수 3치 Reed-Muller 전개식에서 6가지 제어 입력 변수의 순서 선택에 대하여 다치 논리 회로를 실현하는데 필요한 ULM U_f의 수를 나타낸다. 표 2에서 보인 것처럼 제어 입력 변수의 순서를 (x_1, x_2, x_3)로 선택하면 8개의 ULM U_f의 수가 필요하며, 제어 입력 변수의 순서를 (x_1, x_3, x_2)로 선택하면 9개의 ULM U_f의 수가 필요하다. 만약에 제어 입력 변수의 순서를 (x_3, x_1, x_2)로 선택하면 9개의 ULM U_f의 수가 필요하며, 제어 입력 변수의 순서를 (x_3, x_2, x_1)으로 선택하면 8개의 ULM U_f의 수가 필요하다. 또한 제어 입력 변수의 순서로 (x_2, x_1, x_3)를 선택하면 5개의 ULM U_f의 수가 필요하며, 제어 입력 변수를 (x_2, x_3, x_1)의 순서로 선택하면 5개의 ULM U_f의 수가 필요함을 알 수 있다. 그러므로 최적의 제어 입력 변수의 순서로 (x_2, x_3, x_1)를 선택하면 ULM U_f의 수가 가장 적음을 알 수 있다.

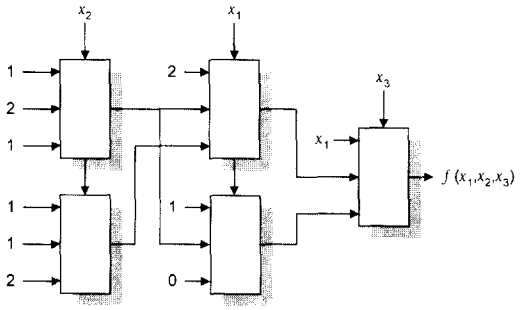

그림 5. ULM U_f 모듈을 사용한 식 (13)의 논리 회로 실현
Fig. 5. Realization of logic circuits of Eq. (13) using ULM U_f modules.

표 3은 식 (13)을 회로 실현하는데 필요한 ULM U_f, mod-3 가산기, mod-3 승산기의 수를 기존의 방법과 비교하였으며, 본 논문은 Fei와 Zhuang[17]이 제시한 방법과 동일한 결과를 가진다. 그러나 최적의 입력 제어 변수의 순서를 선택하는데 Fei와 Zhuang의 방법은 회로 비용 행렬을 이용하여 각 변수에 대하여 회로 비용을 계산하며, 최소의 회로 비용을 갖는 변수를 첫 번째 제어 입력 변수로 선택하고, 다음에 나머지 변수를 가지고 다시 회로 비용 행렬을 이용하여 제어 입력 변수를 선택한다. Fei와 Zhuang는 회로 비용 행렬을 이용하여 한 변수에 대한 최소 회로 비용을 계산하는 시간을 탐색 단위 시간으로 가정하였다. 그러므로 만약 3변수(n=3)의 경우 Fei와 Zhuang이 제시한 방법은 먼저 3변수에 대하여 각각 회로 비용을 계산하여 회로 비용이 적은 변수를 첫 번째 제어 입력 변수로 선택하고, 나머지 변수들에 대하여 다시 회로 비용을 계산하여 두 번째 제어 입력 변수를 선택한다.

그러므로 전체 변수에 대한 회로 비용의 탐색 단위 시간이 (n+2)(n-1)/2이므로 5번의 탐색 단위 시간이 소요된다. 그러나 본 논문은 변수 차수표(VDT)를 검사하는 시간만이 필요하므로 이것을 탐색 단위 시간이라 한다면 한번의 탐색 단위 시간이 소요하게 되며, 컴퓨터 프로그래밍으로 탐색할 경우 Fei와 Zhuang의 방법은 회로 비용 행렬을 계산하기 때문에 {$(p^n)(n+2)(n-1)/2$}의 수행 시간이 소요되며, 본 논문의 경우 (p^n)의 수행 시간이 소요되므로 탐색 시간이 빠른 장점을 갖는다.

표 2. 식 (13)에 대한 3! 제어 입력 변수의 선택

Table 2. Selection of 3! control input variables for Eq. (13)

	1st 단	2nd 단	3rd 단	ULM U_f 수
제어 입력 변수	x_1	x_2	x_3	8
	x_1	x_3	x_2	9
	x_2	x_1	x_3	5
	x_2	x_3	x_1	5
	x_3	x_1	x_2	9
	x_3	x_2	x_1	8

표 3. 식 (13)에 대한 타 논문과의 비교

Table 3. Comparision with other papers for Eq. (13)

	Tokmen & Hurst [11]	Hu & Wu [12]	Fei & Zhuang [17]	본 논문
ULM Uf	8	8	5	5
mod-3 가산기	16	16	10	10
mod-3 승산기	24	24	15	15

[예 2] 3변수 3치 Reed-Muller 전개식인 다음과 같은 다치 논리 함수 $f(x_1, x_2, x_3)$를 ULM U_f 모듈을 사용하여 다치 논리 회로를 실현한다.

$$f(x_1,x_2,x_3) = x_2 \oplus 2\cdot x_2\cdot x_3 \oplus 2\cdot x_2\cdot {x_3}^2 \oplus x_1 \oplus x_1\cdot x_3 \oplus x_1\cdot {x_3}^2 \oplus x_1\cdot x_2 \oplus 2\cdot x_1\cdot x_2\cdot x_3 \oplus 2\cdot x_1\cdot x_2\cdot {x_3}^2 \oplus {x_1}^2\cdot x_2\cdot x_3 \oplus 2\cdot {x_1}^2\cdot x_2\cdot {x_3}^2 \oplus 2\cdot {x_1}^2\cdot {x_2}^2 \oplus 2\cdot {x_1}^2\cdot {x_2}^2\cdot x_3 \oplus 2\cdot {x_1}^2\cdot {x_2}^2\cdot {x_3}^2 \quad (21)$$

식 (21)의 3변수 3치 Reed-Muller 전개식의 계수 c_i의 벡터열을 다음과 같이 표현할 수 있다.

$$[C] = [\,0\,0\,0\;\;1\,2\,2\;\;0\,0\,0\;\;1\,1\,1\;\;1\,2\,2\;\;0\,0\,0\;\;0\,0\,0\;\;0\,1\,2\;\;2\,2\,2\,]^T \quad (22)$$

앞에서 논한 최적 제어 입력 변수의 순서를 선택하는 알고리듬을 사용하여 변수 차수표를 구한 것이 표 4와 같다. 표 4에서 변수 x_1의 0 차수항에서 표시된 "√" 기호의 수가 3이고 1 차수항에서 표시된 "√" 기호의 수가 6이고, 2 차수항에서 표시된 "√" 기호의 수가 5이다. 그리고 변수 x_2에 대하여 0 차수항에서 표시된 "√" 기호의 수가 3이고, 1 차수항에서 표시된 "√" 기호의 수가 8이고, 2 차수항에서 표시된 "√" 기호의 수가 3이다. 변수 x_3에 대하여 0, 1, 2 차수항에 표시된 "√" 기호의 수가 각각 4, 5, 5이다. 그러므로 첫 번째 제어 입력 변수가 x_3이고, 두 번째 제어 입력 변수가 x_2이고, 세 번째 제어 입력 변수가 x_1이다.

선택된 제어 입력 변수의 순서 (x_3, x_2, x_1)에 의해서 회로 비용 행렬을 사용하여 ULM U_f 모듈의 입력 계수를 결정할 수 있다. 식 (22)의 계수 벡터열에서 회로 비용 행렬 [$G_1(0)$], [$G_2(0)$],

표 4. 식 (21)의 검사된 변수 차수표

Table 4. The variable degree table of Eq. (21) checked.

c_i	x_1			x_2			x_3		
	0	1	2	0	1	2	0	1	2
1	√				√		√		
2	√				√			√	
2	√				√				√
1		√		√			√		
1		√		√				√	
1		√		√					√
1		√			√		√		
2		√			√			√	
2		√			√				√
1			√		√			√	
2			√		√				√
2			√			√	√		
2			√			√		√	
2			√			√			√
*	3	6	5	3	8	3	4	5	5
	14			14			14		

[$G_3(0)$] 을 다음과 같이 구할 수 있다.

$$[G_1(0)] = \begin{bmatrix} 0\,0\,0 & 1\,2\,2 & 0\,0\,0 \\ 1\,1\,1 & 1\,2\,2 & 0\,0\,0 \\ 0\,0\,0 & 0\,1\,2 & 2\,2\,2 \end{bmatrix} \quad (23)$$

$$[G_2(0)] = \begin{bmatrix} 0\,0\,0 & 1\,1\,1 & 0\,0\,0 \\ 1\,2\,2 & 1\,2\,2 & 0\,1\,2 \\ 0\,0\,0 & 0\,0\,0 & 2\,2\,2 \end{bmatrix} \quad (24)$$

$$[G_3(0)] = \begin{bmatrix} 0\,1\,0 & 1\,1\,0 & 0\,0\,2 \\ 0\,2\,0 & 1\,2\,0 & 0\,1\,2 \\ 0\,2\,0 & 1\,2\,0 & 0\,2\,2 \end{bmatrix} \quad (25)$$

표 4에서 선택된 첫 번째 제어 입력 변수가 x_3이므로 변수 x_3에 대하여 ULM U_f 모듈의 입력 계수를 다음과 같이 결정할 수 있다.

$$[G_3(1)] = [1 \;\; x_3 \;\; x_3^2]\,[G_3(0)]$$
$$[1 \;\; x_3 \;\; x_3^2] \begin{bmatrix} 0\,1\,0 & 1\,1\,0 & 0\,0\,2 \\ 0\,2\,0 & 1\,2\,0 & 0\,1\,2 \\ 0\,2\,0 & 1\,2\,0 & 0\,2\,2 \end{bmatrix}$$
$$= \begin{bmatrix} 0 & U_f(1,2,2,x_3) & 0 & U_f(1,1,1,x_3) & U_f(1,2,2,x_3) & 0 \\ 0 & U_f(0,1,2,x_3) & U_f(2,2,2,x_3) \end{bmatrix} \quad (26)$$

식 (26)에서 $[G_2(1)]$을 구하면 다음과 같다.

$$[G_2(1)] = \begin{bmatrix} 0 & U_f(1,1,1,x_3) & 0 \\ U_f(1,2,2,x_3) & U_f(1,2,2,x_3) & U_f(0,1,2,x_3) \\ 0 & 0 & U_f(1,1,2,x_2) \end{bmatrix} \quad (27)$$

표 4에서 두 번째 제어 입력 변수가 x_2이므로 변수 x_2에 대하여 ULM U_f 모듈의 입력 계수를 다음과 같이 구할 수 있다.

$$[G_2(2)] = [1 \;\; x_2 \;\; x_2^2]\,[G_2(1)]$$
$$= [1 \;\; x_2 \;\; x_2^2] \begin{bmatrix} 0 & U_f(1,1,1,x_3) & 0 \\ U_f(1,2,2,x_3) & U_f(1,2,2,x_3) & U_f(0,1,2,x_3) \\ 0 & 0 & U_f(1,1,2,x_2) \end{bmatrix}$$
$$= [\,U_f(0, U_f(1,2,2,x_3), 0, x_2) \;\; U_f(U_f(1,1,1,x_3), U_f(1,2,2,x_3), 0, x_2)\, U_f(0, U_f(0,1,2,x_3), U_f(2,2,2,x_3), x_2)\,] \quad (28)$$

그러므로 ULM U_f 모듈을 사용하여 식 (28)에 대한 다치 논리 회로를 합성하면 그림 6과 같은 회로를 실현할 수 있다.

그림 6에서 식 (21)과 같은 3변수 3치 다치 논리 함수를 회로 실현하면 ULM U_f가 8개 필요하다. 표 4는 식 (21)의 3변수 3치 Reed-Muller 전개식에서 6가지 제어 입력 변수의 순서 선택에 대하여 다치 논리 회로를 실현하는데 필요한 ULM U_f의 수를 나타낸다. 표 4에서 보인 것처럼 제어 입력 변수의 순서를 (x_1, x_2, x_3)로 선택하면 9개의 ULM U_f의 수가 필요하며, 제어 입력 변수의 순서를 (x_1, x_3, x_2)로 선택하면 9개의 ULM U_f의 수가 필요하다. 만약에 제어 입력 변수의 순서를 (x_3, x_1, x_2)로 선택하면 8개의 ULM U_f의 수가 필요하며, 제어 입력 변수의 순서를 (x_3, x_2, x_1)으로 선택하면 8개의 ULM U_f의 수가 필요하다. 또한 제어 입력 변수의 순서로 (x_2, x_1, x_3)를 선택하면 10개의 ULM U_f의 수가 필요하며, 제어 입력 변수를 (x_2, x_3, x_1)의 순서로 선택하면 역시 10개의 ULM U_f의 수가 필요함을 알 수 있다. 그러므로 최적의 제어 입력 변수의 순서로 (x_3, x_2, x_1)를 선택하면 ULM U_f의 수가 가장 적음을 알 수 있다.

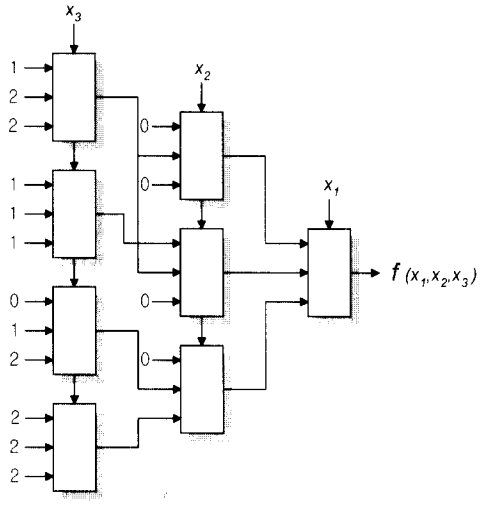

그림 6. ULM U_f 모듈을 사용한 식 (21)의 논리 회로 실현

Fig. 6. Realization of logic circuits of Eq. (21) using ULM U_f modules.

표 4. 식 (21)에 대한 3! 제어 입력 변수의 선택

Table 4. Selection of 3! control input variables for Eq. (21)

	1st 단	2nd 단	3rd 단	ULM U_f 수
제어 입력 변수	x_1	x_2	x_3	9
	x_1	x_3	x_2	9
	x_2	x_1	x_3	10
	x_2	x_3	x_1	10
	x_3	x_1	x_2	8
	x_3	x_2	x_1	8

표 5는 식 (21)을 회로 실현하는데 필요한 ULM U_f, mod-3 가산기, mod-3 승산기의 수를 Stankovic[18] 등이 제시한 방법과 비교하였으며, Stankovic 등이 제시한 회로는 9개의 ULM U_f가 필요한 반면 본 논문에서는 8개의 ULM U_f가 필요함을 알 수 있다.

표 5. 식 (21)에 대한 타 논문과의 비교
Table 5. Comparision with other papers for Eq. (21)

	Stankovic et al[18]	본 논문
ULM U_f	9	8
mod-3 가산기	18	16
mod-3 승산기	27	24

V. 결 론

본 논문에서는 3변수 3치 Reed-Muller 전개식에 기초한 만능 논리 모듈(universal logic modules; ULM) U_f를 사용하여 다치 논리 회로의 최적 합성 알고리듬을 제시하였다. 제시된 다치 논리 회로의 최적 합성 알고리듬은 Reed-Muller 전개식의 계수에 대하여 모든 변수의 각 차수를 변수 차수표에 표시하고, 각 변수의 0 차수열이 가장 많이 표시된 변수를 탐색하므로서 최적의 제어 입력 변수의 순서를 결정한다. 또한 제시된 알고리듬에 의해 결정된 최적의 제어 입력 변수의 순서를 Reed-Muller 전개식에 기초한 ULM U_f 모듈을 이용하여 간략화된 나무 구조 모양의 다치 논리 회로를 실현하였다. 따라서 기존에 제시된 다치 논리 회로의 합성 방법보다 향상된 다치 논리 회로의 실현을 보였다.

본 연구는 Fei와 Zhuang[17]의 방법과 동일한 결과를 보였으며, Stankovic[18] 등이 제시한 방법보다 회로가 다소 간략화됨을 보였다. 최적의 입력 제어 변수의 순서를 선택하는데 Fei와 Zhuang의 방법은 회로 비용 행렬을 이용하여 각 변수에 대하여 회로 비용을 계산하며, 최소의 회로 비용을 갖는 변수를 첫 번째 제어 입력 변수로 선택하고, 다음에 나머지 변수를 가지고 다시 회로 비용 행렬을 이용하여 제어 입력 변수를 선택한다. Fei와 Zhuang는 회로 비용 행렬을 이용하여 한 변수에 대한 최소 회로 비용을 계산하는 시간을 탐색 단위 시간으로 가정하였으므로 Fei와 Zhuang이 제시한 방법은 $(n+2)(n-1)/2$의 탐색 단위 시간이 소요되는 반면에 본 논문은 변수 차수표(VDT)를 검사하는 시간만이 필요하므로 이것을 탐색 단위 시간이라 한다면 유일하게 한번의 탐색 단위 시간이 소요하게 된다. 또한 컴퓨터 프로그래밍으로 탐색할 경우 Fei와 Zhuang의 방법은 회로 비용 행렬을 계산하기 때문에 $\{(p^n)(n+2)(n-1)/2\}$의 수행 시간이 소요되며, 본 논문의 경우 n^p의 수행 시간이 소요되므로 탐색 시간이 빠른 장점을 갖는다. 다만 변수와 치수가 증가하면 계산의 요소가 (p^n)으로 증가하므로 최적의 제어 입력 변수를 선택하는데 제한성을 갖는다.

참 고 문 헌

[1] K. C. Smith, "The prospects for multivalued logic; a technology and applications view," IEEE Trans., Comput., vol. C-30, pp. 619-634, Sept. 1981.

[2] S. L. Hurst, "Multiple-valued logic - its status and its future," IEEE Trans., Comput., vol. C-33, pp. 1160-1179, Dec. 1984.

[3] M. Kameyama, "Toward the age of beyond binary electronics and systems," IEEE Proc. 20th ISMVL, pp. 162-166, May 1990.

[4] D. Etiemble, "On the performance of multivalued integrated circuits: past, present and future," IEICE Trans. Electron., vol. E76-C, No. 3, pp. 364-371, Mar 1993.

[5] Z. Zilic and Z. G. Vranesic, "New interpolation algorithms for multiple-valued Reed-Muller forms," IEEE Proc. 26th ISMVL, pp. 16-23, May 1996.

[6] X. Chen and X. Wu, "The synthesis of ternary functions under fixed polarities and ternary I^2L circuits," IEEE Proc. 13th ISMVL, pp. 424-429, May 1983.

[7] H. Wu, M. A. Perkowski, X. Zeng and N. Zhuang, "Generalized partially-mixed polarity Reed-Muller expansion and its fast computation," IEEE Trans. Comput., vol. 45, no. 9, pp. 1084-1088, Sept. 1996.

[8] V. H. Tokmen, "Disjoint decomposablity

of multi-valued functions by spectral means," IEEE Proc. 10th, ISMVL, pp. 88-93, May 1980.

[9] B. Harking and C. Moraga, "Efficient derivation of Reed-Muller expansions in multiple-valued logic systems," IEEE Proc. 22nd ISMVL, pp. 436-441, May 1992.

[10] T. Higuchi and M. Kameyama, "Synthesis of optimal T-gate networks in multiple-valued logic ," IEEE Proc. 9th ISMVL, pp. 190-195, May 1979.

[11] V. H. Tokmen and S. L. Hurst, "A consideration of universal logic modules for ternary synthesis based upon Reed-Muller coefficient," IEEE Proc. 9th ISMVL, pp. 248-256, May 1979.

[12] Z. Hu, X. Wu,"The logic synthesis using ternary universal logic module U_{hs}," IEEE Proc. 17th ISMVL, pp. 250-259, May 1987.

[13] Q. Hong and B. Fei, "Fast synthesis for ternary Reed-Muller expansion," IEEE Proc. 23rd ISMVL, pp. 14-16, May 1993.

[14] B. Fei and Q. Hong, "Calculation of ternary mixed polarity function vector," IEEE Proc. 23rd ISMVL, pp. 236-238, May 1993.

[15] R. Drechsler, M. Theobald and B. Becker, "Fast OFDD-based minimization of fixed polarity Reed-Muller expressions," IEEE Trans. Comput., vol. 45, no. 11, pp. 1294-1299, Nov. 1996.

[16] E. V. Dubrova and J. C. Muzio, "Testability of generalized mulitple-valued Reed-Muller Circuits," IEEE Proc. 26th ISMVL, pp. 56-61, May 1996.

[17] B. Fei, Z. Zhuang, "Fast logic synthesis based upon ternary universal logic module U_f," IEEE Proc. 22nd ISMVL, pp. 401-407, May 1992.

[18] R. S. Stankovic, M. Stankovic, C. Moraga and T. Sasao, "Calculation of Reed-Muller-Fourier coefficients of multiple-valued functions through multiple-place decision diagrams," IEEE Proc. 24th ISMVL, pp. 82-88, May 1994.

저 자 소 개

成 賢 慶(正會員)

1955년 12월 21일생. 1982년 2월 인하대학교 전자공학과 졸업(공학사). 1984년 2월 인하대학교 대학원 전자공학과 졸업(공학석사). 1991년 2월 인하대학교 대학원 전자공학과 졸업(공학박사). 1989년 3월 ~ 1991년 8월 부천전문대학 전자계산과 조교수. 1991년 9월 ~ 현재 상지대학교 전산학과 조교수. 주관심 분야는 Multiple-Valued Logic Design, Computer Architecture & VLSI Design, Fuzzy Control, Digital Signal Processing 등임

韓 永 煥(正會員) 第 33卷 B編 第12號 參照
현재 상지대학교 전산학과 전임강사

崔 在 碩(正會員)

1964년 6월 8일생. 1988년 2월 인하대학교 전자공학과 졸업(공학사). 1990년 2월 인하대학교 대학원 전자공학과 졸업(공학석사). 1997년 8월 인하대학교 대학원 전자공학과 졸업(공학박사). 1990년 3월 ~ 1995년 4월 유니온 시스템 연구소 연구원. 1997년 1월 ~ 현재 (주)정오정보통신 선임연구원. 주관심 분야는 Multiple-Valued Logic Design, Logic Minimization, Fuzzy Control, CAD/CAE 등임