論文 89-26-5-17

상태합성기 설계를 위한 상태 CHDL 기술 및 기호최소화 알고리듬개발

(The State CHDL Description and Symbolic Minimization Algorithm Development for State Machine Synthesizer)

金 熙 碩*

(Hi Seok Kim)

要 約

상태합성기를 설계하기 위해 상태 CHDL 기술 및 기호간소화 알고리듬을 제안하였다.

상태 CHDL은 PLA에 의한 FSM 설계에 매우 적합하며 제안된 기호간소화 알고리듬은 단일큐브포함, 1거리병합 알고리듬을 이용하였다.

BOLD 논리최소화 tool을 이용한 상태합성의 절차를 교통신호제어기 등의 예를 들어 설명하였다.

Abstract

A Symbolic cover Minimization Algorithm and State CHDL Description for Finite State Machine Synthesizer are Presented.

State CHDL are used for design of PLA based finite state machine, also the symbolic cover minimization algorithms are based upon single cube containment and distance 1 merging algorithms.

The procedure for state machine synthesizer has been applied to practical example, including traffic light controller by using Boulder Optimal Logic Design System.

I. 서 론

최근 VLSI/LSI 디지틀시스템 설계시 회로의 복잡도로 인하여 논리합성(logic synthesis) tool을 이용한 설계자동화 연구가 활발히 진행되고 있다. 논리합성 tool을 이용한 디지틀시스템의 설계자동화를 하기 위해서는 설계시스템의 상위단위설계[1,10] (highlevel design) 즉 기능설계(behavioral design)가 우선되어야 한다.

디지틀시스템 기능설계는 설계하고자 하는 시스템의 동작특성을 상위단계 언어로 기술하여 제어용 부분을 기술하는 언어와 논리조합용(combinational logic) 언어로 분리되어 netlist 즉 부울회로(boolean network)를 산출한다.

제어용 부분의 기술언어는 순서논리회로(sequential logic circuit)인 Finite State Machine 형태로 기술되어야 한다.

현재 FSM 회로를 기술하는 여러형태의 FSM 합

*正會員, 淸州大學校 電子工學科

(Dept. of Elec. Eng., Chongju Univ.)

接受日字 : 1988年 11月 19日

(※본 연구는 1987년 문교부의 IBRD 차관 연구기금에 의해 조성되었음.)

성 tool[5,6,7](synthesizer)이 개발되고 있으며 FSM 회로의 설계시간을 단축하기 위해 규칙성이 높은 논리회로인 RAM, ROM, PLA 등을 많이 사용하고 있다. 일반적으로 PLA는 논리구조가 AND-OR 2단이라는 단순한 구조를 하고 있으므로 설계자동화를 하기가 쉽다.

그러나 PLA에 의한 FSM 합성 tool은 논리최소화 알고리듬 및 상태할당이 매우 중요시 되고 있으며 현재 PLA에 의한 상태할당(state assignments)에 대한 연구가 활발히 진행중에 있다. 최적할당을 하기 위해서는 먼저 PLA의 면적을 최소화하는 기호최소화(symbolic minimization) 알고리듬이 개발되어야 한다.

따라서 본 논문에서는 종래의 FSM 합성 tool과는 다른 기호최소화 알고리듬을 개발하였으며 FSM 기술언어는 상태 CHDL[2,4](c based hardware description language)을 정의 기술하였으며 논리최소화 tool인 BOLD(boulder optimal logic design) tool에 적용 PLA 최적화의 효율성을 입증하였다.

Ⅱ. PLA에 의한 FSM 회로설계

PLA는 규칙적인 구조이기 때문에 설계자동화가 용이하며 논리최소화 tool, PLA folding 알고리듬을 이용하면 PLA의 면적과 전달지연시간 등을 최적화 할 수 있다.

그러므로 PLA에 의한 FSM 회로는 설계자가 요구하는 설계사양(specification)에 매우 적합하게 설계될 수 있다.

일반적으로 순서회로의 동기방식으로는 동기식(synchronous)과 비동기식 방식이 있으며 PLA에 의한 FSM 회로 설계시 사용되는 방식은 시스템 CLOCK과 동기되는 동기식 방식을 사용한다.

동기식 FSM 순서회로는 조합회로 부분과 메모리 부분으로 나누어지며 조합회로 부분은 PLA로 설계되며 메모리 부분은 여러형태(JK, D, RS…)가 있으나 회로의 간략화와 PLA에 의한 FSM 회로의 경우 D Flip-Flop을 사용한다.

그림 1에 PLA에 의한 FSM 회로를 도시하였다. FSM 회로는 Primary 입력과 현재상태에 의해 출력과 다음상태가 결정되므로 식(1)과 같이 표현된다.

$$Z = \lambda(P, X)$$
$$Q = \delta(P, X) \qquad (1)$$

$X = \{X1, X2, X3, \cdots Xn\}$ primary 입력집합
$P = \{P1, P2, P3, \cdots Pn\}$ 현재상태 집합
$Q = \{Q1, Q2, Q3, \cdots Qn\}$ 다음상태 집합

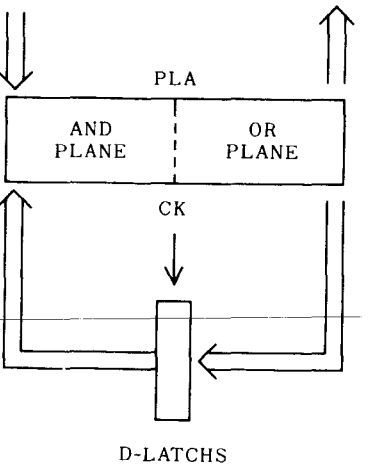


그림 1. PLA에 의한 FSM 회로
Fig. 1. PLA based finite state machine.

$Z = \{Z1, Z2, Z3, \cdots Zn\}$ 출력상태 집합

식(1)에서 X 즉 primary 입력이 존재하고 출력 Z, Q가 존재하지 않는다면 incompletely-specified 상태로 정의된다.

식(1)을 상태그래프(state graph), 상태천이표(state transition table)로 전개할 수 있으므로 FSM 회로설계를 용이하게 하기 위해서 고급언어를 사용한 자동설계를 필요로 한다.

따라서 본 논문에서는 상태그래프, ASM 도표[9]로부터 입력기술 언어의 상태 CHDL(state CHDL)을 기술하고 상태천이표를 구한 다음 구한 상태기호 테이블(state symbolic table)을 간략화하여 상태할당을 한 후 BOLD 논리최소화 tool에 입력하여 최소화된 FSM 회로를 설계하기로 하며 그림 2에 전체시스템 흐름을 표현하였다.

Ⅲ. C입력기술언어 입력

현재 BOLD 논리최적화 tool에 사용되고 있는 CHDL은 조합논리를 기술하는 CHDL이며 FSM 회로를 기술할 수 있는 CHDL의 개발이 필요하다. FSM 회로를 기술하는 CHDL은 상태기술을 할 수 있어야 하므로 상태 CHDL로 정의하고 상태 CHDL의 최종 출력형태는 BOLD 논리최소화 tool의 입력형태인 LIF(logic interchange format)이어야 한다. 상태기술을 하기 위해 특정한 언어기술을 하였으므로 상태 CHDL을 토큰(token)으로 분석하기 위한 어휘분석(lexical analysis) 알고리듬과 상태천이표를 만들기 위

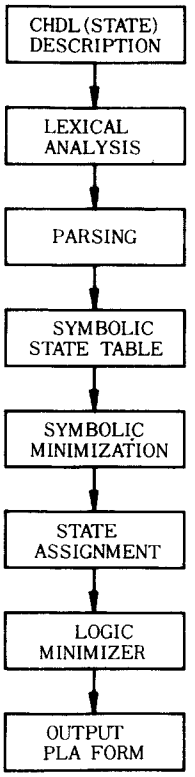


그림 2. 전체순서도
Fig. 2. State machine synthesizer flow chart.

한 구문분석(parsing) 알고리듬이 개발되어야 하므로 본 논문에서 제안한 어휘분석, 구문분석 알고리듬은 다음과 같이 표현된다.

〔단계 1〕 Keyword를 INPUT, OUTPUT, MODEL, BEGIN, PSTATE, NSTATE, IF, THEN, ELSE, END 등으로 정한다.

〔단계 2〕 상태기술용 CHDL을 lookup 테이블을 사용하여 FSM 방식으로 Keyword, identifier, 숫자…등으로 어휘를 구분한다.

〔단계 3〕 순환재귀적(recursive) 구문분석 알고리듬를 이용하여 심벌테이블(symbol table) 즉 상태천이표를 구성한다.

〔단계 4〕 구성된 상태천이표로 부터 오퍼레이터 AND (&), OR(¦), NOT(!)를 이용하여 PLA 입력코드를 생성한다.

〔단계 5〕 PLA 입력코드 생성에 따라 현상태(present state)와 다음상태의 위치를 입력코드와 일치하게 재배치 한다.

〔단계 6〕 생성된 입력코드에 일치하도록 출력테이블을 생성한다. 상태 CHDL은 상태그래프, ASM 도표로 부터 쉽게 기술할 수 있도록 IF, THEN, ELSE의 문장구조(syntax)를 갖고 있으며 상태 CHDL의 조건에 따라 상태 CHDL의 제어구조문(control statement)이 변하게 된다. 상태 CHDL의 예로서 MEAD & Conway[1,9]의 교통신호제어회로(traffic light controller)를 그림 3의 상태그래프로 부터 그림 4와 같은 상태 CHDL로 기술된다. 교통신호제어회로를 기술한 상태 CHDL은 어휘분석, 구문분석 알고리듬에 의해 그림 5와 같은 상태천이테이블을 생성한다. 상태천이 테이블에서 입력변수, 입력변수조건값, AND, OR, NOT, 오퍼레이터에 의해 PLA의 입력코드를 생성한다.

입력코드 생성 알고리듬은 그림 6과 같이 표현된다. 입력코드 생성후 현상태와 다음상태 위치를 상태천이 테이블에서 생성된 입력코드와 일치하도록 재배치 하여야 한다. 그림 7에 교통신호제어회로의 PLA 입력코드와 재배치된 현재상태, 다음상태와 출력이 다출력(multiple output)인 경우 출력변수값을 출력변수 위치와 일치하도록 정하는 위치큐브방식(positional cube)으로 정하여 표현하였다. 일반적으로 상태그래프, ASM 도표는 입력변수가 여러개일 경우가 있다. 이런경우 상태 CHDL의 조건식 표현을 순환재귀적 구조로 표현 기술하여야 한다. 그림 8에서 자동판매기(automatic vending machine)의 상태 CHDL을 일부 기술하였으며 조건식을 순환재귀적인 구조로 기술하였

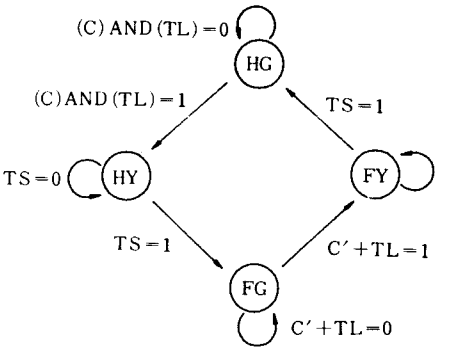


그림 3. 교통신호제어기 상태그래프
Fig. 3. Traffic light controller state diagram.

```
MODEL (traffic)
INPUT : C, t1, ts :
OUTPUT : st, h1, f1 :
BEGIN {
PSTATE = hg :
IF ( !(C = 1 & t1 = 1)) THEN {
      NSTATE = hg ;
      out1 = green ;
      out2 = red ;
      out3 = no ;
      }
PSTATE = hg ;
IF (C = 1 & t1 = 1) THEN {
      NSTATE = hy ;
      out1 = green ;
      out2 = red ;
      out3 = yes ;
      }
PSTATE = hy ;
IF (ts = 0) THEN {
      NSTATE = hy ;
      out1 = yellow ;
      out2 = red ;
      out3 = no ;
      }
      ELSE {
      NSTATE = fg ;
      out1 = yellow ;
      out2 = red ;
      out3 = yes ;
      }
PSTATE = fg ;
IF ( ! (C = 0 | t1 = 1)) THEN {
      NSTATE = fg ;
      out1 = red ;
      out2 = green ;
      out3 = no ;
      }
PSTATE  fg ;
IF (C = 0 | t1 = 1) THEN {
      NSTATE = fy ;
      out1 = red ;
      out2 = green ;
      out3 = yes ;
      }
PSTATE = fy ;
IF (ts = 0) THEN {
      NSTATE = fy ;
      out1 = red ;
      out2 = yellow ;
      out3 = no ;
      }
      ELSE {
      NSTATE = hg ;
      out1 = red ;
      out2 = yellow ;
      out3 = no ;
      }
END
```

그림 4. 교통신호제어기 상태 CHDL
Fig. 4. Traffic light controller state CHDL.

Symbolic transition table

C	0	hg	hg	¦	green
t1	0	hg	hy		red
C	1	hy	hy	&	no
t1	1		fg		green
ts	0	fg	fg		red
ts	1	fg	fy		yes
C	1	fy	fy	&	yellow
t1	0		hg		red
C	0			¦	no
t1	1				yellow
ts	0				red
ts	1				yes
					red
					green
					no
					red
					green
					yes
					red
					yellow
					no
					red
					yellow
					yes

그림 5. 상태천이 테이블
Fig. 5. State transition table.

```
PROCEDURE encode( )
/* S, K는 입력변수 집합 및 입력변수 갯수 */
/* I, l는 상태천이 테이블에서의 입력변수 및입력변수 갯수 */
/* op는 operator 집합 */
/* sop는 상태천이 테이블의 operator */
/* opv는 상태천이 테이블의 operator 값 */
/* dopv는 don't care 상태 */
BEGIN
¦K¦ ← ¦S¦, 1 ← ¦sop¦ ;
if (I ∈ S, sop ∈ op) {
  for (k = 0, 1, ..... 입력변수) {
    for (1 = 0, 1, ..... 상태입력변수) {
      if (SK ≡ S1) {
        code = code ∪ opv[k];} ;
      else {code = code ∪ dopv[k] ;
          }
      }
    }
  }
}
END
```

그림 6. 입력코드 생성 알고리듬
Fig. 6. Input code generation algorithm.

다. 현재 개발된 상태 CHDL은 입력변수가 10개까지 가능하며 더 많은 입력변수를 기술하기 위한 상태CHDL을 연구 진행중이다.

Symbolic cover table

0	–	–	hg	hg	00101
–	0	–	hg	hg	00101
1	1	–	hg	hy	10110
–	–	0	hy	hy	11001
–	–	1	hy	fg	11010
1	0	–	fg	fg	10101
0	–	–	fg	fy	10110
–	1	–	fg	fy	10110
–	–	0	fy	fy	11001
–	–	1	fy	hg	11010

그림 7. 기호테이블
Fig. 7. Symbolic table.

```
MODEL (Vendingmachine)
INPUT : xa, xb, r, c ;
OUTPUT : za, zb, zc ;
BEGIN {
PSTATE=sa ;
IF (xa=0 & xb=0) THEN {
  IF (r=0 & c=0) THEN {
    NSTATE=sa ;
    out1=0 ;
    out2=0 ;
    out3=0 ;
    }
  }
PSTATE=sa ;
IF (xa=1 & xb=0) THEN {
  IF (r=0 & c=0) THEN {
    NSTATE=sb ;
    out1=0 ;
    out2=0 ;
    out3=0 ;
    }
  }
PSTATE=sa ;
IF (xa=1 & xb=0) THEN {
  IF (r=0 & c=0) THEN {
    NSTATE=sf ;
    out1=0 ;
    out2=0 ;
    out3=0 ;
    }
  }
```

그림 8. 자동판매기 CHDL
Fig. 8. Automatic vending machine CHDL.

Ⅳ. 기호테이블 최소화와 상태할당

일반적으로 PLA(특히 FPLA)의 면적, 지연시간 등을 최적화 하기 위해서는 상태천이 테이블에서 재배치된 기호테이블(symbolic cover table) 최소화와 상태할당이 매우 중요한 문제이다.

만약 기호테이블에서 2행의 동등상태(equivalent state)가 존재하고 현행이 다음행을 포함하는 관계가 성립한다면 동등상태와 포함되는 행은 1행으로 병합(merge)될 수 있으며 상태할당 방식에 따라 최적화된 PLA가 생성된다. 근래에 Amstrong,[8] Stearns, Hartmanis 등은 상태분할(partition), Reduced Dependency 알고리듬을 이용 FSM 회로의 상태할당을 하였다. 이러한 상태할당법은 기호테이블에서 변수가 don't care 상태를 고려하지 않아 규모가 큰 FSM 상태할당에는 적합하지 못하다는 단점이 있다.

최근 Demichel[6,7]는 이러한 단점을 보완한 조건코드 방식에 의한 Kiss(keep internel state synthesis) 알고리듬을 개발 제안하였다.

그러나 Kiss 알고리듬은 2레벨 PLA의 상태할당만 가능하다는 단점이 있다. 현재 PLA 면적을 효율적으로 사용하기 위한 다단 PLA(multi-level PLA)의 사용이 증대되고 있다. 이에 따라 상태할당 방식도 2단 상태할당 알고리듬에서 다단 상태할당 알고리듬으로 변환되어야 한다. 최근 다단 상태 알고리듬인 Mustang 상태할당 알고리듬이 S.Devadas에 의해 개발되었다.

Mustang[7] 상태할당 알고리듬은 FSM 면적을 최소화 하기위한 공통큐브식(common cube expression)을 기호테이블의 입력코드와 현재상태에서 추출하는 Fanin Oriented 알고리듬과 다음상태의 출력에서 추출하는 Fanout Oriented 알고리듬으로 구성하여 다단 상태할당을 가능하게 하였다. 그러나 Mustang 상태할당 알고리듬은 기호테이블의 최소화를 고려하지 않아 기호테이블에서 동등상태와 행의 포함관계가 있는 경우도 상태할당을 하고 있다.

따라서 본 논문에서는 기호테이블에서 행을 서로 비교하여 동등상태와 현행이 다음행을 포함(cover)할 때 입력조건식과 출력조건식을 고려하여 단일큐브포함[1](single cube containments) 알고리듬과 1거리 병합(distance 1 merging) 알고리듬을 사용 다음과 같은 정리를 제안하였다.

〔정리 1〕 상태군(state group) A={iA, SA, SA', OA}, B={iB, SB, SB', OB} (i는 primary 입력, S는 현상태, S'는 다음상태, O는 출력)일 때 SA⊇SB, SA'⊇SB'이고 iA⊇iB, OA⊇OB이면 상태 A, 상태 B는 동등상태 또는 현행이 다음해을 포함하므로 1개 상태로 병합할 수 있다.

〔증 명〕 OA=λ(iA, SA), OB=λ(iB, SB) 이므로 SA⊇SB, OA⊆OB, iA⊇iB이면 OA와 OB는 OA로 병합되므로 정리 1은 증명이 된다.

정리 1에서 사용된 커버(cover) 알고리듬은 단일큐브포함 알고리듬이며 현행과 다음행의 큐브포함관계와 입출력큐브가 n개일 때 포함관계를 탐색하여 최소큐브커버(minimal cube cover)를 구할 수 있으므로 그림 9와 같이 표현된다. 정리 1과 단일큐브포함 알고리듬을 이용하면 다음과 같은 예제가 성립한다.

```
/* C, Ci+1은 큐브 행수 */
/* F는 CUBE의 COVER 행렬 */
PROCEDURE contain(Ci, Ci+1)
BEGIN
for(i=0, ..... 최대변수) {
switch(Ci, Ci+1∈F) {
  case(0, 1) {입출력 큐브값이 0, 1이면 다음 입출력
              큐브값 비교 if(contained) contained
              =TRUE break}
  case(2, 3, 4, ..... n) {기호테이블의 현재상태 행과
                          다음상태 행의 포함관계조사
                          if(contained) cobtained=
                          TRUE break}
  don't care : break ;
 }
}
return(contained)
END
```

그림 9. 단일큐브포함 알고리듬
Fig. 9. Single cube containment algorithm.

〔예 제〕 기호테이블에서 현재상태 P와 다음상태 Q가 서로같은 상태와 현상태 P가 다음상태 Q를 포함하는 상태를 함께 정리하면 다음과 같이 된다고 가정한다.

```
0 0 0 0   SA   SA   0 0 0
- - - 1   -    SA   0 0 0
- - 0 1   SA   SA   0 0 0
```

위의 기호테이블에서 $---1 \supseteq --01$ 이므로 단일큐브포함 알고리듬에 의해 3행 입력코드는 제거된다. 또한 1행 입력코드 0000은 단일큐브포함 및 1거리 병합(distance 1 merging) 알고리듬에 의해 000- 로 변환된다. 따라서 정리 1에 의해 생성된 테이블은 아래와 같다.

```
0 0 0 -   SA   SA   0 0 0
- - - 1   SA   SA   0 0 0
```

위 예에서 제안된 단일큐브포함 알고리듬과 큐브의 거리(distance)가 1일 때 병합 알고리듬인 1거리 병합 알고리듬을 이용하면 기호테이블에서의 동등상태 행제거와 열의 변수도 don't care 상태로 줄일 수 있어 FSM 회로의 면적을 줄일 수 있다. 제안된 알고리듬은 행의 포함관계를 조사하고 난 후 1거리 병합 알고리듬을 반복수행하므로 최소의 기호테이블을 생성한다.

따라서 제안된 기호최소화 알고리듬은 그림10과 같다. 상태 CHDL에서 생성된 기호테이블을 최소화한 다음 기호로 표기된 현재상태, 다음상태는 2진 코드로 상태할당이 되어야 한다.

```
PROCEDURE symbolic minimize(IN, PS, NS, OUT)
/* PS, NS는 현재, 다음상태 */
/* IN, OUT는 입출력큐브 */
BEGIN
ncube←|Fi|
for(i=0, ..... ncube-1) {
  for(k=i+1, ..... ncube) {
    if(현상태 PSi⊇PSk) {
      if(다음상태 NSi⊇NSk) {
        if(INi⊇INk∪1거리병합(INi, INk))∩
          (OUTi⊇OUTk∪1거리병합(OUTi, OUTk)) {
          contained Fk=TRUE
          Fi=병합(행i, 행K)
          } return Fi
        else if{(INk⊇INi∪1거리병합(INi, INk))∩
                (OUTk⊇OUTi∪1거리병합(OUTk, OUTi))
                contained Fi=TRUE
                Fi=병합(행k, 행i)
               } return Fi
        }
      }
    } return Fi /* 줄일수 없는 큐브행렬 */
  }
END
```

그림10. 기호최소화 알고리듬
Fig. 10. Symbolic minimize algorithms.

FSM 회로를 상태할당 할 수 있는 여러 알고리듬에서 본 논문에서 이용한 상태할당 알고리듬은 Mustang[7] 알고리듬을 사용하였다.

표 1에 PLA를 이용한 FSM 회로의 기호테이블을 표현하였다.

표 1의 기호테이블에서 1행 2행은 동등상태이므로 기호최소화 알고리듬을 사용하면 1개의 행으로 병합된다. 즉 기호테이블을 22행에서 21행으로 줄여 Mustang 알고리듬에 입력한다.

기호테이블을 최소화 한 후 Mustang 알고리듬을 사용하여 상태할당을 하여 표 2에 표현하였다. 상태할당이 되면 PLA 면적을 최적화하여야 한다. 현재 PLA 면적을 최소화 하기 위한 논리최소화 tool이 개

표 1. PLA에 의한 FSM 회로의 테이블
Table 1. PLA based finite state machine table.

--1--	init0	init0	110000
--1--	init0	init0	110000
--0--	init0	init1	110000
--00-	init1	init1	110000
--01-	init1	init2	110001
--0--	init2	init4	110100
--01-	init4	init4	110100
--00-	init4	iowait	000000
0000-	iowait	iowait	000000
1000-	iowait	init1	110000
01000	iowait	read0	101000
11000	iowait	write0	100010
01001	iowait	rmack	100000
11001	iowait	wmack	100000
--01-	iowait	init2	110001
--0-0	rmack	rmack	100000
--0-1	rmack	read0	101000
--0-0	wmack	wmack	100000
--0-1	wmack	write0	100010
--0--	read0	read1	101001
--0--	read1	iowait	000000
--0--	write0	iowait	000000

발 사용되고 있다. 현재 사용되고 있는 논리최소화 tool은 ESPRESSO Ⅱ, MINI, MIS…등이 사용되고 있으며 규칙적 구조인 PLA 설계 및 최소화에 매우 효율적임이 입증되었다.

PLA 최소화 설계를 하기 위해 본 논문에서 사용한 논리최소화 tool은 BOLD 논리최소화 tool이며 BOLD 논리최소화 tool의 특징은 ESPRESSO Ⅱ 논리최소화 tool의 기본 알고리듬인 확대(expand), 축소(reduce), irredundant cover 알고리듬을 확장 정립한 ESPRESSO MLT (multi-level tautology checking) 방법을 사용하므로 다른 논리최소화 tool보다 효율적인 PLA 논리최소화를 할 수 있다.

이러한 BOLD 논리최소화 tool을 이용하면 부울회로의 최적화(최소화)를 할 수 있으며 특히 PLA의 경우 AND-OR 2단 PLA를 인수분해 (factorization)와 공통큐브추출 알고리듬에 의해 다단 PLA로 변환되므로 최소화된 PLA를 구현할 수 있다.

따라서 종래의 2단 PLA 설계보다 PLA 면적 지연시간 등이 감소하게 되어 대규모 FSM 회로의 PLA화가 가능하다. 이에 따라 다단 PLA의 변환과정은 매우 중요한 것으로 다단 PLA 변환과정을 BOLD 논리최소화 tool을 사용 다음과 같이 표현하였다.

표 3에 부울변수(boolean literals)가 43개인 식 f1, f2, f3를 2단 PLA 테이블인 LIF 입력화일로 기술하여 BOLD 논리최소화 tool에 입력하여 다단 PLA 변환 결과를 표 4에 구현하였다.

표 2. 상태할당된 PLA
Table 2. State assignment PLA.

```
# STATE ASSIGNED FINITE AUTOMATION
# init0    0000   0000
# init1    0011   0011
# init2    0111   0111
# init4    0010   0010
# iowait   0001   0001
# read0    0100   0100
# write0   1000   1000
# rmack    1001   1001
# wmack    0101   0101
# read1    0110   0110
.i 9
.o 10
.type fr
--1--    0000    0000    110000
--0--    0000    0011    110000
--00-    0011    0011    110000
--01-    0011    0111    110001
--0--    0111    0010    110100
--01-    0010    0010    110100
--00-    0010    0001    000000
0000-    0001    0001    000000
1000-    0001    0011    110000
01000    0001    0100    101000
 11000   0001    1000    100010
01001    0001    1001    100000
11001    0001    0101    100000
--01-    0001    0111    110001
--0-0    1001    1001    100000
--0-1    1001    0100    101000
--0-0    0101    0101    100000
--0-1    0101    1000    100000
--0--    0100    0110    101001
--0--    0110    0001    000000
--0--    1000    0001    000000
```

표 4의 결과에서 43개의 입력 부울변수가 29개의 부울변수로 최소화 되었으며 다단 PLA로 설계가 되었으며 그림11에 다단 PLA인 4단 PLA를 부울회로로 변환하여 표시하였다.

지금까지 논한 BOLD 논리최소화 tool과 상태할당 알고리듬을 이용 교통신호제어기의 상태할당을 표 5에 표현하였으며 표 6에 다단 PLA로 변환될 경우 생성된 부울변수의 갯수를 표현하였다. 생성된 부울변수(literal)를 다른 방법과 비교하면 Random Kiss 알고리듬보다 변수가 적게 생성되어 효율적임이 입증

표 3. PLA식 f1, f2, f3
Table 3. PLA f1, f2, f3 boolean equetion.

f1=ABCD+ABCE+ABF+ABG+HLM
f2=AICD+AICE+AIF+AIJ+HKLM
f3=AHLM+BHLM

표 4. PLA LIF 출력화일
Table 4. PLA LIF output file.

```
#LIF description of 2-level PLA after MLMIN.
. circuit 2-level PLA. pwmlm
.nv 13 3 20
.io 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
.elements
.cover 6 1 3 1 2 7 12 18 19 14
     - - - 1 1 - 1
     1 1 1 - - - 1
     - 1 - - - 1 1
.cover 7 1 3 1 9 10 11 12 18 19 15
     - - - 1 1 1 - 1
     1 1 1 - - - - 1
     - 1 - - - - 1 1
.cover 3 1 1 12 17 18 16
     1 1 1 1
.cover 2 1 2 1 2 17
     - 1 1
     1 - 1
.cover 2 1 1 8 13 18
     1 1 1
.cover 4 1 2 1 3 6 20 19
     1 1 - 1 1
     1 - 1 - 1
.cover 2 1 2 4 5 20
     - 1 1
     1 - 1
. end
#*******MLMIN V1. 0. e (6/11/87)*******
#A : 2-level PLA. pwdiv filename=2-level PLA. pwdiv
#B : 29 original number of literals=29
```

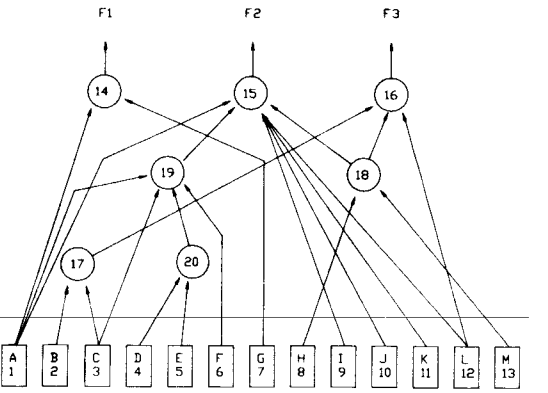


그림 11. 다단 PLA의 부울신호
Fig. 11. The boolean network of multilevel PLA.

표 5. 교통신호제어기 PLA 테이블
Table 5. Traffic light controller PLA table.

```
# STATE ASSIGNED FINITE AUTOMATION
# hg  00  00
# hy  10  10
# fg  01  01
# fy  11  11
.i 5
.o 7
.type fr
0--  00  00  00101
-0-  00  00  00101
11-  00  10  10110
--0  10  10  11001
--1  10  01  11010
10-  01  01  10101
0--  01  11  10110
-1-  01  11  10110
--0  11  11  11001
--1  11  00  11010
```

되었다.

또한 FSM2, FSM3, FSM4 예에서 기호테이블 최소화 알고리듬을 적용하므로서 기호테이블의 행과 열의 입출력 코드중 일부변수를 don't care 상태로 처리 기호테이블 최소화를 할 수 있었다.

또한 표 6에서 다른 여러종류의 FSM 회로도 실현하였으며 상태 CHDL에서 기호최소화 알고리듬까지는 약 3500Line 정도의 C언어로 구성되어 있다. 구현된 C프로그램은 VAX/780 UNIX OS상에서 실현된다.

이와 같이 최소화된 기호테이블을 Mustang 알고리듬에 적용비교하면 원래의(original) 기호테이블보다 Mustang 알고리듬 수행시간이 적었으며 PLA 면적도 감소된다.

그러나 기호테이블 간소화 알고리듬을 사용하므로서 상태 CHDL로 부터 생성된 기호태이블에서 동등상태 및 현행이 다음행을 커버하는지 여부를 조사하는 동등상태검사(equivalance check) 시간이 많이 든다는 단점이 있다.

표 6. PLA 최소화 결과
Table 6. PLA minimization result.

	I/O	P	P'	S	M literal	CHDL excution time	t	t'
FSM1(traffic)	3/5	10	10	2	37	17.2sec	1.6	1.6
FSM2(PLA controller)	5/6	22	21	2	253	16.3sec	2.4	2.2
FSM3(auto-vending)	4/3	43	40	4	2322	25.6sec	14.57	14.02
FSM4(black jack)	5/10	17	16	3		29.7sec	3.29	3.20
FSM5(10bit conuter)	2/1	21	21	4		18.1sec	2.2	2.2

I/O : 입력 / 출력
P : 기호테이블 행수
P' : 줄인 기호테이블 행수
S : 상태수
M literal : 다단 PLA literal 수
t' : 줄인 기호테이블을 Mustang 알고리듬에 적용한 시간
t : 기호테이블을 Mustang 알고리듬에 적용한 시간

따라서 동등상태검사를 하는 효율적 알고리듬이 필요로 하며 현재 Tautology 검사법을 이용 연구 진행중이다.

V. 결 론

본 논문에서는 BOLD 논리최적화 tool에 응용할 수 있는 상태 CHDL을 연구하였으며 생성된 상태천이 테이블을 기호최소화 테이블로 변환되는 알고리듬을 제안하였다. 또한 BOLD 논리최소화 tool을 이용 다단 PLA 설계도 가능함을 입증하였다.

앞으로 본 논문의 연구과제로는 BOLD 논리최소화 tool에 이용할 수 있는 상태 CHDL을 효율적으로 설계하는 것과 다단 PLA에도 적용되는 최적상태할당(optimal state assignment) 알고리듬을 개발하는 것이다.

또한 CHDL을 상태 CHDL과 조합(combination) CHDL로 분리하는 cycle break에 관해서도 연구할 예정이다.

參 考 文 獻

[1] In Proceedings, NATO ASI on Logic Synthesis and Silicon Compilation for VLSI, P. Antognetti, G. DeMicheli and A. Sangiovanni-Vincentelli (editor), Kluwer, Dordrecht, The Nethrland, 1987.

[2] K.A. Bartlett, D.G. Bostick, G.D. Hachtel, R.M. Jacoby, M.R. Lightner, P.H. Moceyunas, C.R. Morrison and D. Ravenscroft, "BOLD: A multi-level logic optimization system", ICCAD87, 1987.

[3] R. Brayton, R. Rudell, A. Sangiovanni-Vincentelli and A. Wang, "MIS: A multi-level logic optimization system," ICCAD87, 1987.

[4] C.T. Bye, M.R. Lightner and D.L. Ravenscroft, "A functional modeling and simulation environment based on ESIM and c," : ICCAD-84, pp. 51-54, 1984.

[5] C.J. Tseng et. al "A verstail finite state machine synthesizer," ICCAD-86 Digest, pp. 206-209, Nov. 1986.

[6] G. DeMicheli, R.K. Brayton and A.S. Vincentelli, "Optimal state assignments of finite state machines," ICCAD87, pp. 269-285, 1985.

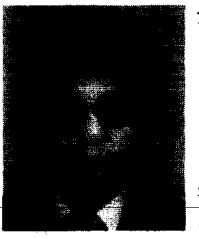
[7] S. Devadas, H. Tongma el. at "MUSTANG: State assignment of finite state machines for optimal multi-level logic implementations," ICCAD-87, pp. 16-19, 1987.

[8] T.A. Dolotta, A.J. McClusl-ey, "The coding of interal state of sequential machines," *IEEE Tras on Electronic Computer*, vol. EC-13, Oct. 1964.

[9] 조중휘, "VLSI의 논리설계 자동화를 위한 SDL에 관한 연구", Phd, 논문, 한양대학교 6. 1986.

[10] 엄성용, 전주식, "상위단계의 HDL에 관한 연구", 한국정보과학회지, vol 14, 5. 1987. *

著 者 紹 介



金 熙 碩(正會員)

1954年 12月 23日生. 1977年 한양대학교 전자공학과 공학사학위 취득. 1980年 한양대학교 대학원 전자공학과 공학석사학위 취득. 1985年 한양대학고 대학원 전자공학과 공학박사학위 취득. 1987年 9月~1988年 9月 미국 University of Colorado at Boulder VLSI 설계실에서 연구. 현재 청주대 전자공학과 부교수. 주관심분야는 실리콘콤파일러 분야중 HDR, 논리합성 및 PLD 설계분야 등임.