論文 89-26-1-12

VLSI 설계자동화 시스템을 위한 순서회로의 최적상태코드 할당 알고리듬

(An Optimal State-Code Assignment Algorithm of Sequential Circuits for VLSI Design Automation Systems)

林 載 允,* 林 寅 七**

(Jae Yun Lim and In Chil Lim)

要 約

순서회로를 PLA로 실현하기 위한 설계자동화 방법에 대해 논했으며, PLA로 회로 실현시 면적최소화를 위한 최적 상태코드할당 알고리듬을 제안하였다. 회로기술이 용이하고 합성력이 강한 하드웨어 기술언어인 DASL(design automation support language)[8]을 이용하여 회로를 기술한 후 이를 컴파일하여 그 결과를 PLA로 자동설계하는 시스템을 제안한다. 순서회로의 각 상태에 대한 코드할당시 출력 및 다음상태를 동시에 고려하여 코드할당을 수행함으로서 종래의 할당법에 비해 평균 10%의 PLA 면적을 감소 시킬 수 있었다. 본 시스템은 마이크로 인스트럭션, FSM, VLSI의 제어부설계등에 확장 적용할 수 있게 구성하였다.

Abstract

A design automation method for sequential circuits implementation by mans of PLA is discussed, and an optimal state-code assignment algorithm to minimize the PLA area is proposed. In order to design sequential circuit automatically, DASL (Design Automation Support Language) [8] which is easy to describe and powerful to synthesize, is proposed and used to describe sequential circuit. An optimal statecode assignment algorithm which considers next states and outputs simultaneously is proposed, and by adopting this algorithm to various examples, the area of PLA is reduced by 10% comparing privious methods. This system is constructed to design microinstruction, FSM, VLSI control part synthesis.

*正會員, 濟州大學校 通信工學科
(Dept. of Telecomm. Eng., Cheju Nat'l Univ.)
**正會員, 漢陽大學校 電子工學科
(Dept. of Elec. Eng., Hanyang Univ.)
接受日字 : 1988年 4月 18日

I. 서 론

최근 VLSI의 집적도가 증가하고 설계의 복잡도가 증가됨에 따라 되어 자동 설계시스템에 대한 필요성이 증가하게 되었다.[1]

이러한 자동설계 시스템은 규칙적이고 구조적인

특성을 갖으며 설계 효율을 높이기 위해 계층적 설계를 수행할 필요가 있고, 가능한한 최소의 면적으로 회로를 실현하여야 한다.[8]

이런 요구에 부응하는 회로 설계방식으로 PLA (programmable logic array)가 널리 사용되며, 이러한 PLA의 출력 일부에 래치를 부가하여 입력측으로 귀환시켜 주 클럭 신호에 의해서 동작시킴으로서 일반 순서 회로의 설계에 널리 사용된다.

특히 순서회로 설계는 현상태와 다음 상태에 따른 최적 설계가 요구되는 분야로서 디지탈 컴퓨터의 제어부 설계나 일반 제어 시스템 설계 등으로 사용된다. 최근 이러한 복잡한 순서회로 설계를 하드웨어 기술 언어로 기술한 후, 이를 컴파일하여 회로에 대한 마스크 패턴까지를 자동생성하려는 실리콘 컴파일러에 대한 연구가 활발히 진행되고 있다. 이러한 실리콘 컴파일러에 대한 목적은 설계 전과정에 대한 자동화를 기함과 동시에, 최소의 면적으로 주어진 기능에 맞는 회로를 실현하는 것으로서 이러한 순서회로에 대한 하나의 설계방법으로 PLA가 널리 사용되고 있다.[2,3]

이러한 순서회로를 PLA로 실현할 경우, 면적의 최소화는 PLA의 적항선을 최소화하는 것으로 이는 주어진 함수를 최소의 적항선으로 실현하는 함수 최소화 기법과 이러한 함수 최소화를 도모하기 위하여, 순서회로의 각 상태에 대한 최적 코드 할당등으로 대별할 수 있다. 여기서 함수 최소화기법에 대해서는 이미 각종 알고리듬이 개발되어 실회로의 설계에 널리 사용되고 있으나, 상태에 대한 코드 할당에 대해서는 현재 개발도중에 있다.

종래의 상태 코드 할당에서는 가중치에 의한 Gray 코드 할당, 또는 각 상태에 대한 제한 조건만을 고려하여 각 상태에 따른 행렬 계산에 의한 상태별 코드 할당을 수행하였으나, 이는 다음 상태 또는 출력에 대한 정보를 고려하지 않고 제한조건만을 고려한 상태 코드 할당을 수행함으로서 최종 회로 실현시 면적의 최소화를 기할 수 없게 되었다.[2,4]

이는 함수 최소화 기법에 대한 개념을 고려하지 않음으로서 PLA의 적항선이 최적화 되지 않아 최종 회로 면적의 증가를 초래하게 된다.

따라서 본 논문에서는 순서회로를 PLA로 실현할 경우, 회로의 면적을 최소화하기 위하여 함수최소화 기법을 고려하여, 상태 천이표의 다음 상태 및 출력을 동시에 고려한 각 상태에 대한 최적 코드 할당법을 제안한다. 우선 순서회로를 자동설계하기 위하여, 본 저자들이 개발한 하드웨어 기술언어인 DASL[8]를 사용하여 PLA로 순서 회로를 자동설계하는 시스템을 구성하고 면적 최소화를 위한 상태 코드할 당법에 관해 기술한다.

또한 본 알고리듬을 프로그램하여 여러 회로에 적용한 후 기존의 방법들과 비교 검토함으로서 알고리듬이 면적 및 수행시간이 단축됨을 입증한다.

Ⅱ. DASL을 이용한 순서회로의 자동설계

디지탈 시스템의 제어부의 설계시 순서회로가 중요한 역할을 하며 이러한 순서 회로는 여러 모델로 표현할 수 있으며 이중 Finite State Machine이 일반적으로 많이 사용되고 있다.

이러한 FSM은 일반적으로 조합회로 및 기억장치로 구성되며, 기억장치는 특정 시간에 회로의 상태를 일시 저장시키며 조합회로는 입력 및 회로상태에 따른 주 출력으로 표시할 수 있다.[1]

한편 VLSI 시스템에서 순서회로를 실현키 위해 CAD 도구에 의해 설계될 수 있는 규칙적 구조를 갖는 것이 필요하며 또한 칩면적의 최소화를 기해야 한다. 이에 가장 적합한 회로로서 PLA가 일반적으로 많이 사용되며, 이때 기억 장치로서는 각 상태의 결과를 일시 저장키 위해 D래치 또는 J-K래치를 많이 사용하며 주 클럭에 의해서 입력으로 귀환되어 작동시킨다. 이러한 특성으로 인해 순서회로의 실현 방법으로 PLA가 널리 사용되고 있으며 이에 대한 일반구조를 나타내면 그림 1과 같다.

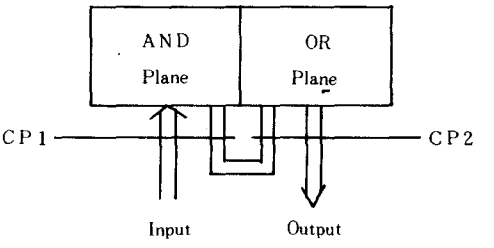

그림 1. PLA를 이용한 순서회로 실현
Fig. 1. Sequential circuit implementation by PLA.

이렇게 PLA로 순서회로를 실현할 경우 최종 목적은 최소의 면적내에 주어진 기능에 맞는 회로를 실현하는 것으로 PLA의 적항선의 수를 최소화 하는 것과도 일치한다.

이러한 순서회로의 설계과정은 설계하고자 하는 회로의 기능에 대한 순서도를 작성한 후 이에 대한 상태표를 작성하여 각 상태에 코드값을 할당한 후 전상태 및 입력의 조합에 의한 다음 상태 및 출력의

함수간소화를 수행한 후 이를 바탕으로 PLA 마스크 패턴을 작성하는 것이다.

최근 이를 자동화하기 위한 방편으로 순서도에 의한 기능을 하드웨어 기술언어로 기술한 후 이에 대한 각 기능을 함수화하여 실행시킴으로서 설계 전 과정을 자동화하려는 실리콘 컴파일러에 관한 연구가 활발히 진행되고 있다.[1,6]

이러한 하드웨어 기술언어는 주어진 회로를 정확히 기술해야 하며 설계 정보가 자동설계 시스템에 의해 레이아웃 정보로 변환이 가능한 합성력을 가져야 하고, 분석 시스템에 필요한 정보 제공이 가능한 분석력등을 갖추어야 한다.

이에 따라 본 구문 규칙이 간단하고 기술이 용이하며 레이아웃 정보로의 변환이 용이한 하드웨어 기술언어인 DASL를 자체 개발하여 각종 회로의 자동설계에 사용하고 있다.[8] 한 예로서 그림 2는 [2]의 예에 대한 상태도를 나타낸 것이고 그림 3은 이를 DASL로 기술한 것이다.

이렇게 기술된 회로는 하드웨어 컴파일러를 거쳐 각 상태별 상태표를 작성한 후 각 상태에 따른 코드 할당을 하고 각 상태 및 출력에 대한 함수 최소화를 수행하여, 이를 PLA 자동생성기의 입력으로 사용하여 이 함수에 대한 PLA 마스크 패턴을 얻는다.

표 1은 그림 3의 하드웨어 컴파일러를 거친후의 상태표이며 표 2는 각 상태에 [2]의 알고리듬에 의한 코드할당을 한 결과이며 이에 따른 PLA에 대한 회로도를 나타내면 그림 4와 같다. 여기서 상태에 대한 don't care항은 111이며, 총 적항수는 10개이다.

이때 PLA의 면적을 최소화 하기 위해선 AND 항의 적항선의 수를 최소화할 필요가 있으며 이는 함

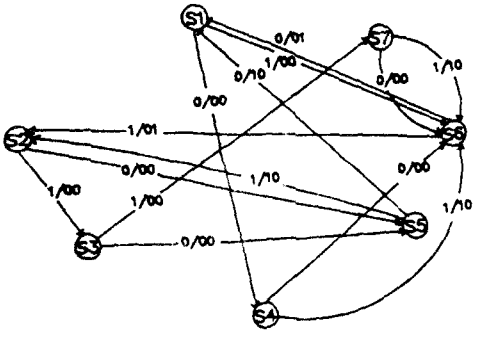

그림 2. [2]의 예에 대한 상태도
Fig. 2. State diagram for [2]'s example.

```
〈MODULE : example〉
MODIN    : bit in;
MODOUT   : register out[0, 1];
OUTNAME  : register 00, 01, 10;
MBEGIN
  S 1 {case in.   out[0, 1]=00;goto S6;
       case ! in. out[0, 1]=00;goto S4;
       }
  S 2 {case in.   out[0, 1]=00;goto S3;
       case ! in. out[0, 1]=00;goto S5;
       }
  S3  {case in.   out[0, 1]=00;goto S7;
       case ! in. out[0, 1]=00;goto S5;
       }
  S 4 {case in.   out[0, 1]=10;goto S6;
       case ! in. out[0, 1]=00;goto S6;
       }
  S 5 {case in.   out[0, 1]=10;goto S2;
       case ! in. out[0, 1]=10;goto S1;
       }
  S 6 {case in.   out[0, 1]=01;goto S2;
       case ! in. out[0, 1]=01;goto. S1;
       }
  S 7 {case in.   out[0, 1]=10;goto S6;
       case ! in. out[0, 1]=00;goto S6;
       }
MEND
```

그림 3. 그림 2에 대한 DASL 기술
Fig. 3. DASL description for fig. 2.

표 1. 그림 2에 대한 상태표
Table 1. State table of fig. 2.

IN	PSTATE	NSTATE	OUT
0	S 1	S 6	0 0
0	S 2	S 5	0 0
0	S 3	S 5	0 0
0	S 4	S 6	0 0
0	S 5	S 1	1 0
0	S 6	S 1	0 1
0	S 7	S 6	0 0
1	S 1	S 4	0 0
1	S 2	S 3	0 0
1	S 3	S 7	0 0
1	S 4	S 6	1 0
1	S 5	S 2	1 0
1	S 6	S 2	0 1
1	S 7	S 6	1 0

표 2. 표 1에 대한 상태할당
Table 2. State assignment of table 1.

IN	PCODE	NCODE	OUT
0	0 1 0	0 1 1	0 0
0	1 1 0	0 0 1	0 0
0	1 0 1	0 0 1	0 0
0	0 0 0	0 1 1	0 0
0	0 0 1	0 1 0	1 0
0	0 1 1	0 1 0	0 1
0	1 0 0	0 1 0	0 0
1	0 1 0	0 0 0	0 0
1	1 1 0	1 0 1	0 0
1	1 0 1	1 0 0	0 0
1	0 0 0	0 1 1	1 0
1	0 0 1	1 1 0	1 0
1	0 1 1	1 1 0	0 1
1	1 0 0	0 1 0	1 0

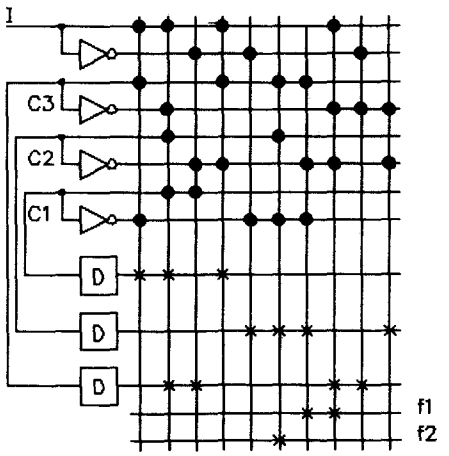

그림 4. 표 2의 상태에 대한 PLA 실현형태
Fig. 4. PLA pattern for table 2's state.

수 최소화 과정을 통해 이루어지며 이는 본 연구실에서 개발한 함수 최소화기인 PLAMIN(PLA minimizer)를 통해 실현된다.

또한 적항선은 상태에 따른 코드 할당에 따라서도 PLA의 크기가 변한다. 따라서 본 논문에서는 PLA의 면적을 최소화하기 위해 다음 상태 및 출력을 고려한 최적 상태할당법을 제안한다.

Ⅲ. 상태 코드 할당 알고리듬

1. 제반정의 및 성질

순서회로의 각종 상태를 Si라 표시하고, 다출력함수의 각 출력을 F,라 할때 함수최소화 기법을 염두에 두고 상태간 또는 상태와 출력과의 관계를 종합적으로 고려하여 최적 상태할당을 수행키 위해 다음과 같은 사항을 정의한다.

〔정의 1〕 현 상태에서 다음 상태로 천이하는 과정에서 입력조건에 의한 현재 및 다음 상태들간의 관계를 나타낸 것을 현상태-다음상태 행렬(SS 행렬)이라 정의한다.
이때 행은 다음 상태들을 나타내고, 열은 현 상태를 나타내며 행렬의 각 요소들을 천이하기 위한 입력조건들을 나타낸다. 입력조건들이 다수일 경우는 집합으로 나타내고 서로 관계가 없는 항들은 '-'로 표시한다.

예로 표 1에 대한 SS 행렬을 나타내면 표 3과 같다.

표 3. 표 1에 대한 SS 행렬
Table 3. SS matrix for table 1.

	S 1	S 2	S 3	S 4	S 5	S 6	S 7
S 1 :	-	-	-	1	-	0	-
S 2 :	-	-	1	-	0	-	-
S 3 :	-	-	-	-	0	-	1
S 4 :	-	-	-	-	-	{0, 1}	-
S 5 :	0	1	-	-	-	-	-
S 6 :	0	1	-	-	-	-	-
S 7 :	-	-	-	-	-	{0, 1}	-

〔정의 2〕 현 상태에서 출력이 1의 값이 되는 입력값들의 조합을 나타낸 것을 현상태-출력 행렬(SF 행렬)이라 정의한다.
이때 행은 출력값을 표시하고 열은 현상태를 나타내며 입력조건들이 다수일 경우는 집합으로 표시하며, 관계 없는 항들은 '-'로 표시한다.

예로 표 1에 대한 SF 행열을 나타내면 표 4와 같다.

〔정의 3〕 SS(SF)행렬 상에서 다음 상태(출력)에 대해 현상태의 입력조건중 동일한 상태의 수

표 4. 표 1에 대한 SF 행렬
Table 4. SF matrix for Table 1.

	f1	f2
S1 :	-	-
S2 :	-	-
S3 :	-	-
S4 :	1	-
S5 :	{0, 1}	-
S6 :	-	{0, 1}
S7 :	1	-

를 N이라 할때 C=⌈$\log_2$N⌉을 다음상태(출력)에 대한 큐브값이라 하고 이러한 큐브값에 해당하는 현상태의 집합을 다음상태(출력) 큐브집합이라 한다. 또 다음상태(출력) 큐브값중 최대치를 최대 큐브값이라 한다.

여기서 c의 큐브치를 갖는 상태 집합들은 2^c 개의 연속된 gray 코드값을 가질 수 있도록 그룹을 지을 수 있으며 이들은 우선순위, 인접도 등에 따라 순차적으로 gray 코드로 나타낼 수 있음을 표시한다.

위의 예에서 다음상태 큐브집합은 {(S4, S7), (S5, S6)}이고, 이의 최대 큐브값은 1이다. 또한 출력 큐브집합은 {(S4, S7)}이고 이에 대한 최대 큐브값은 1이다.

〔정의 4〕 SS(SF) 행렬에서 어느 다음상태(출력)에 대해 입력조건이 동일한 현상태들이 존재하면 이들 현상태들은 다음상태(출력) 에 대해 서로 연관되었다고 하며, 어느 다음상태(출력)에 대해서도 현상태들에 대한 입력조건들이 다르면 이들 현상태들은 다음상태(출력)에 대해 서로 독립이라 한다.

위의 예에서 다음 상태에 대해서 S2와 S3는 연관관계에 있고, S1과 S2는 독립관계에 있으며, 출력에 대해 S6와 S7은 서로 독립관계에 있다.

또한 주어진 상태의 총갯수를 P라 할 경우 이 상태에서 부여할 수 있는 최적인 코드의 수는 Cn=⌈$\log_2$P⌉이며, 입력의 총 갯수를 In, 출력의 총 갯수를 Fn이라 할 경우 상태 코드의 크기 Cn은 Cn>=$\log_2$(In+Fn)이다. 여기서 ⌈A⌉는 A보다 작지 않은 최소의 정수이다.

2. 상태 코드 할당 알고리듬

이상의 각종 성질을 이용하여 함수 최소화를 위해, 다음 상태 및 출력을 고려한 상태코드 할당 알고리듬은 다음과 같다.

〔단계 1〕 총 상태수 및 입, 출력을 고려하여 각 코드에 부여할 최적 코드길이 Cn을 구함.

〔단계 2〕 상태들 또는 상태와 출력간의 초기조건을 구성키 위해, 주어진 상태표에 대한 SS행렬 및 SF행렬을 구성하고, 각 행렬의 출력 또는 다음 상태에 대한 입력조건에 따른 큐브치 및 큐브집합을 구한다. 여러 개의 큐브 조건이 생길 경우는 현 상태와 큐브관계를 많이 갖는 다음상태 또는 출력순으로 해당 큐브 크기 만큼 순차적으로 선택하여 큐브 집합 형성. 현 상태들의 집합을 초기 주그룹으로 구성.

〔단계 3〕 현 주그룹의 상태수를 N이라 할때, SS 행렬 및 SF 행렬상에서 최대 큐브조건항, 서로 독립적인 큐브관계를 갖는 항, 독립항중에서 현상태에 대한 최대 항수를 갖는 상태순으로 2^{N-1}일때까지 상태수 그룹을 형성하여 이를 차기 주그룹이라 분류하고, 나머지 상태를 차기 부그룹으로 둠. 만일 SS 행렬과 SF 행렬의 큐브값이 서로 같을 경우는 해당 큐브들에 속한 다음상태 또는 출력의 갯수를 합하여 그 크기가 큰 순으로 주그룹 형성. 이 값도 같을 경우는 SS 행렬의 상태에 가중치 부여하여 선택.

〔단계 4〕 현 주그룹에 해당하는 코드의 최상위 bit를 0으로 하고 부그룹에 해당하는 코드의 최상위 bit를 1로 하여 그룹간을 분리하고 최상위 비트 할당. 현 주그룹내의 상태수가 2이거나 상태들간이 서로 연관관계가 있으면 단계 5로 그렇지 않으면 현 주그룹을 새 그룹으로 선정하여 Recursive 하게 단계 3 수행.

〔단계 5〕 현 주그룹내에서 SS 또는 SF 행렬상의 최대 큐브선택. 만일 최대 큐브가 다수이고 이들이 서로 독립이라면, 이들간에 새로운 그룹간 큐브조건을 형성하여 상위 비트에서 하위비트 순으로 그룹간 큐브 비트 형성. 나머지 최대 큐브 집합 내에서는 다음과 같은 우선순위에 의해 순차적으로 코드 할당. 최대 다음상태, 또는 출력을 갖는 상태 선정, 일정 코드 할당. 서로 연관된 항들을 이웃하도록 연결짓고 이들에 대해 한코드 차이 나도록 gray코드 할당. 만일 SS 또는 SF 행렬 모두에 대해, 서로 독

립적이면 임의의 상태 선정 나머지 gray코드 할당. 모든 주그룹내의 상태 코드할당이 끝날때까지 다음 상태 큐브 별로 Recursive하게 단계 5 반복.

〔단계 6〕 현 부그룹 내에서 주그룹과 연관관계가 없는 그룹들끼리 새로운 연관관계를 조사하여 주그룹내의 상태 코드할당과 마찬가지로 Recursive하게 상태코드 할당. 만일 SS 또는 SF 행렬과도 아무 연관관계도 갖지 않는 상태들에 대해서는 임의로 미할당된 gray코드를 할당. 부그룹내의 모든 상태들에 대한 상태코드 할당이 끝날때까지 Recursive하게 단계 6 반복수행. 모든 부그룹 할당이 끝나면 단계7.

〔단계 7〕 각 상태에 따른 코드를 할당한 후, 현상태와 입력을 새로운 입력조합으로, 다음상태와 출력을 새로운 출력 항으로 설정하고 상태할 당시 미할당된 코드들은 don't care 항으로 하여, 함수 최소화기의 입력으로 사용할 수 있도록 출력시키고 종료.

3. 적용 예

본 알고리듬의 유용성을 보이기 위해 표 1에 대한 본 알고리듬의 수행과정 및 결과는 다음과 같다.

우선 각 SS 행렬 및 SF 행렬상에서 각 상태에 대한 연관관계를 고려하여 주그룹 1으로 (S4, S7)(S5, S6)을 선택하고 부그룹 1으로 (S1, S2, S3)를 형성한다. 주그룹 1에 해당하는 부분의 최상위 비트에 0을 부그룹 1에 1을 할당(표 5-a). 주 그룹의 두 상태 중 (S4, S7)을 주그룹 2, (S5, S6)를 부그룹 2로 분할한 후, 차상위 비트에 각각 0, 1을 할당. (표5-b). (S4, S7)중 임의로 0과 1을 할당, (S5, S6) 중 S5는 S4와 출력에 대해 연관관계에 있으므로 0 할당. S6에 1할당. 따라서 주그룹 1할당 완료(표 5-c). 또한 부그룹 1중, 이미 할당된 주그룹과 연관있는 상태에 대한 코드 할당함. S1과 S4가 서로 다음 상태에 대해 연관되고, 부그룹에 대해 이미 할당된 비트가 존재하므로 S4와 나머지 비트는 동일하게 코드 할당. 또 나머지는 주그룹과는 상관없으나 S2와 S3는 서로 연관되므로 1비트 차이나게 이미 할당된 코드와는 다르게 gray 코드식으로 할당함으로서 전 상태에 대한 코드 할당 완료(표 5-d). 이에 대한 최종 코드를 나타내면 표 5와 같다.

여기서 don't care 항은 (110)이고, 이에 대한 각 상태 및 출력에 대한 함수 최소화 결과는 표 6과 같이 출력되며, 이를 PLA 자동 생성기의 입력으로 하여 마스크 패턴을 생성한 것이 그림 5와 같다. 이때 총 적항선 수는 9개이다.

여기서 본 알고리듬에 의한 상태 코드 할당을 한 결과 PLA의 적항선이 1개 감소하여 〔2〕의 알고리듬에 의한 결과에 비해 약 10%의 면적 감소효과를 가져왔다.

표 5. 본 알고리듬에 의한 표 1의 상태할당
Table 5. State assignment of table 1 by this algorithm.

	C1	C2	C3
S1 :	1	–	–
S2 :	1	–	–
S3 :	1	–	–
S4 :	0	–	–
S5 :	0	–	–
S6 :	0	–	–
S7 :	0	–	–

(a)

	C1	C2	C3
S1 :	1	–	–
S2 :	1	–	–
S3 :	1	–	–
S4 :	0	0	–
S5 :	0	1	–
S6 :	0	1	–
S7 :	0	0	–

(b)

	C1	C2	C3
S1 :	1	–	–
S2 :	1	–	–
S3 :	1	–	–
S4 :	0	0	0
S5 :	0	1	0
S6 :	0	1	1
S7 :	0	0	1

(c)

	C1	C2	C3
S1 :	1	0	0
S2 :	1	0	1
S3 :	1	1	1
S4 :	0	0	0
S5 :	0	1	0
S6 :	0	1	1
S7 :	0	0	1

(d)

표 6. 표 5의 코드에 대한 표 1의 함수 최소화
Table 6. Function minimization for table 1 with table 5's code.

4 9 5									
1 :	–	0	1	1	0	–	1	–	–
CC1 :	0	–	1	–	1	0	0	–	0
CC2 :	0	0	0	1	–	1	0	1	1
CC3 :	–	0	1	–	–	–	–	0	1
NC3 :	1	1	1	1	0	0	0	0	0
NC2 :	1	0	1	0	1	0	0	0	0
NC1 :	0	0	1	0	0	1	0	0	0
F1 :	0	0	0	0	0	0	1	1	0
F2 :	0	0	0	0	0	0	0	0	1

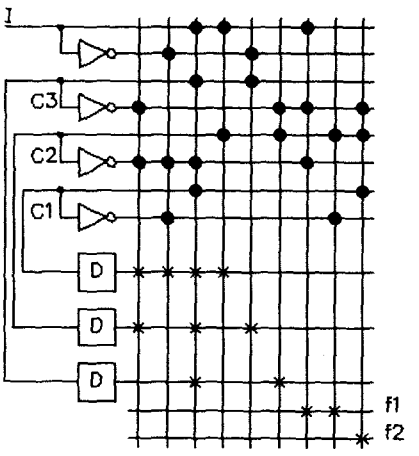

그림 5. 본 알고리듬에 의한 표 6의 PLA 형태
Fig. 5. PLA pattern of table 6 by this algorithm.

이는 다음상태 및 출력을 동시에 고려하여 현상태에 대한 이웃조건을 이용하여 상태에 대한 코드를 할당함으로서 설계효율을 향상시킬 수 있었으며 이를 [2]에서 제시한 여러 예에 적용하여 [2]의 방법에 의한 결과와 비교해 본 결과 평균 10%의 적항수가 줄어들었으며 이에 대한 비교표는 표 7과 같다.

표 7. 본 알고리듬의 수행 결과
Table 7. Execution result of this algorithm.

	Ins	Sts	Outs	Cn	prod. [1]	terms paper	Exec. T (sec)
Ex 1	4	5	1	3	10	9	0.2
Ex 2	1	7	2	3	11	9	0.7
Ex 3	4	8	3	3	18	16	2.0
Ex 4	2	15	3	4	17	16	5.0
Ex 5	4	19	5	5	20	19	11.0
Ex 6	6	23	7	5	23	20	15.4

Ⅳ. PLA 자동 생성

함수 최소화기를 거친 데이타들은 PLA 자동 생성기의 입력으로 사용되어 최소화된 함수에 대한 PLA 마스크 패턴을 생성한다. PLA는 전력소모, 면적 및 동작속도가 빠르며 전하 재분배 현상이나 잡음 여유의 감소 등의 장점을 가진 다이나믹 CMOS PLA의 일종인 Zipper CMOS PLA를 개발하여 사용하였다. 이 Zipper CMOS PLA의 일반 구성은 그림 6과 같이 AND(NAND) 평면의 적항선과 클럭킹 게이트 및 OR(NAND) 평면의 출력선과 클럭킹 게이트로 구성된다. AND(OR) 평면의 적항선(출력선)들은 외부입력(적항선)에 의해 제어되는 NMOS(PMOS) 트랜지스터들이 직렬(병렬)로 연결되어 있다. 또한 Zipper CMOS PLA의 적항은 적항선과 클럭킹 게이트로 구성되고 출력항은 출력선과 클럭킹 게이트로 구성된다. 시스템 클럭 SC가 인가되면 Zipper 드라이버의 출력에서 4개의 클럭신호 C.C', C.C'가 생성되며 이들은 각 클럭킹 게이트에 연결된다. 시스템 클럭이 1일때 각 적항선의 출력은 1이 되므로, OR 평면내의 PMOS들을 OFF시킨다. 다음으로 SC가 0으로 될때 회로 출력은 적항의 NMOS에 인가되는 외부 입력에 의해 0 또는 1이 된다. 따라서 이를 직접 순서 회로의 설계에 이용할 수가 있다. 여기서 그림 5에 대한 Zipper CMOS PLA의 마스크 패턴을 나타내면 그림 7과 같다.

Ⅴ. 시스템 구성도 및 프로그램구조

본 시스템은 순서회로를 PLA로 자동 설계하도록 구성되었으며, VLSI 제어부의 자동설계를 목표로 설계되었다.

전 구문구조는 자동 구문 분석기인 yacc 및 LEX를 사용하여 구성하였고, 각 경우에 따른 수행 함수들은 VAX 11/780 UNIX 4.3 BSD상에서 C언어로 프로그램하였으며 그래픽 터미날로서는 Tektronix 4111 및 4207을 이용하였고, 보조 입력장치로서 Mouse및 Tablet를 사용하였다.

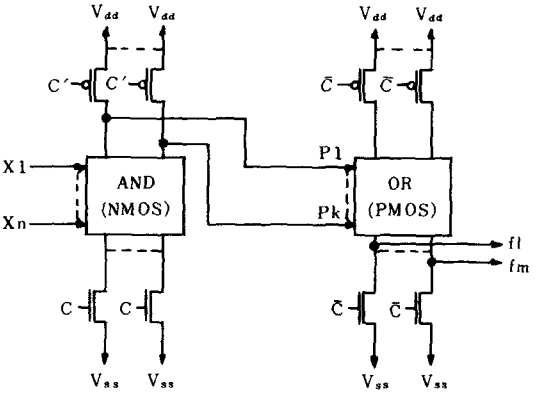

그림 6. Zipper CMOS PLA의 기본구조
Fig. 6. Basic structure of zipper CMOS PLA.

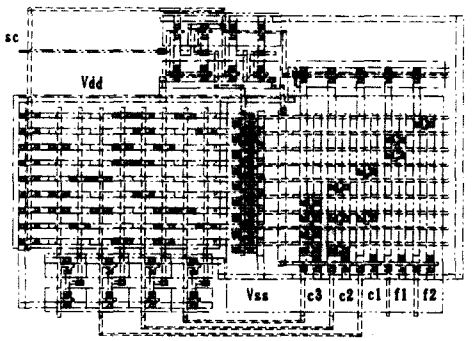

그림 7. 그림 5에 대한 Zipper CMOS PLA
Fig. 7. Zipper CMOS PLA for Fig. 5.

하드웨어 기술언어인 DASL은 구문 분석기를 포함하여 총 5000 라인으로 구성되며 이는 주어진 회로에 대한 하드웨어 기술을 한후 이를 컴파일하여, 상태별 코드를 할당한 후 PLA의 함수 최소화기에 적합한 입력형태로 출력시킨다. 함수최소화기인 PLA MIN은 총 4000 라인으로 구성되며 다입력, 다출력의 함수를 PLA를 구성하기에 적합한 형태로 출력시킨다. 이렇게 생성된 PLA의 입·출력 상태를 입력으로 하여 다이나믹 CMOS PLA의 일종인 Zipper CMOS PLA 자동 생성기에 의해 PLA의 마스크 패턴이 작성된다. 이 PLA 자동 생성기는 그래픽에 라이브러리를 포함하여 총 5000 라인으로 구성되며, PLA의 출력 형태는 그래픽 또는 각 회로 요소에 대한 CIF 형태나 Auto-CAD의 한 데이타 형태인 DXF 형으로 나타낼 수 있다.

일반적인 데이타 구조로는 Linked List 형태를 하고 있으며, 회로실현시 회로요소의 탐색 및 변경을 용이하게 하기 위하여 회로를 기능별로 계층구조를 이루도록 구성하였으며 각 회로들은 Quad-tree 형태로 데이타를 보관하게 하였다.

이에 대한 구성도를 나타내면 그림 8과 같다.

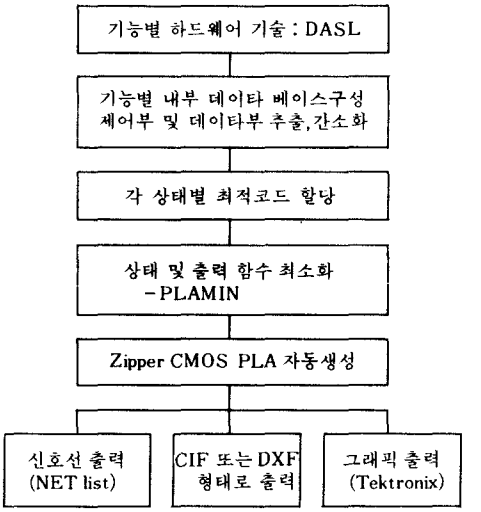

그림 8. 시스템 구성도
Fig. 8. System configuration.

Ⅵ. 결　　론

하드웨어 기술언어인 DASL을 이용하여 PLA로 순서회로를 자동설계하는 시스템을 구성하였고, 이 순서회로를 PLA로 실현할 경우 PLA 면적의 최소화를 위해, 함수 최소화 기법을 고려하고, 상태표의 다음상태 및 출력값을 종합적으로 고려한 최적 상태 코드 할당법을 제안하였다.

하드웨어 기술언어로 기술된 회로를 하드웨어 컴파일하여 간소화된 상태에 따라 코드의 최소 길이를 산출하고 이 코드길이 내에서, 이를 PLA로 실현할 경우 면적의 최소화를 기할 수 있도록 최적 상태 코드 할당법을 수행하였다.

종래의 제한 조건만을 고려하여 행렬 계산에 의한 휴리스틱한 상태할당에 비해 설계 효율이 증가하였으며, 기억용량 및 수행시간이 감소하였으며, gray code 할당 기법을 이용함으로써 코드할당의 간소화를 도모하였으며 이를 함수 간소화기 및 PLA 자동생성기의 입력으로 직접 사용하여 설계할 수 있게함으로써 순서회로의 자동설계 시스템을 구성하였다.

또한 본 시스템은 마이크로 인스트럭션 설계등 VLSI 회로의 제어부 설계에도 사용할 수 있어 실리콘 컴파일러의 일부로서도 사용될 수 있다.

본 상태 코드 할당 알고리듬을 FSM, 순서회로, 마이크로 인스트럭션 등의 코드할당에 적용한 결과 기존 방법에 비해 약 10% 정도의 면적 감소효과를 나타내어 일반 제어회로의 설계에도 확대 적용할 수 있음을 확인하였다.

앞으로의 연구과제로는 순서회로의 데이타부와 효율적으로 결합할 수 있는 하드웨어 기술언어의 확장 및 이를 자동적으로 실현키 위해 적절한 설계 기법을 채용하여 설계과정의 전과정을 자동화하려는 실리콘 컴파일러의 제작이며 현재 이 연구가 진행중이다.

參 考 文 献

[1] Carver Mead, Lynn Conway, "Introduction to VLSI system," Addison-Wesley Co. 1980.

[2] G. De Micheli, Robert K. Brayton, A Sangiovanni-Vincentelli, "Optimal state assignment for finite state machine," *IEEE Trans. on CAD* vol. CAD-4 no. 3 pp. 269-285, July 1985.

[3] S. Kang, W.M. Vancleemput, "Automatic PLA synthesis from a DDL-p description," 19th DA Conf. pp. 391-397.

[4] G. De Micheli, A Sangiovanni-Vincentelli, and T. Villa, "Computer aided synthesis of PLA based finite state machine," in Int. Conf. on Comp. Aid. Des. Santa Clara, CA pp. 154-157, Sept. 1983.

[5] M.J. Meyer, P. Agrawal, R.G. Pfister, "A VLSI FSM design system," 21th DA Conf. pp. 194-200.

[6] "YACC, LEX", UNIX Programmer's Supplementary Document. vol. 1. Ps1:15-1~ps1:16-13.

[7] 오창준, 이철동, 유영옥, "Microinstruction의 부호 할당에 관한 연구", 대한전자공학회논문지 vol. 25, no. 1, pp. 108~114, 1988.

[8] 김학림, 임재윤, 이기희, 홍인식, 임인칠, "하드웨어 기술언어 DASL의 설계와 VLSI FSM의 자동합성", 한국정보과학회 '88봄 학술발표논문집, pp. 295~298, 4. 1988. *

著 者 紹 介

林 載 允 (正會員) 第25卷 第8號 參照
현재 제주대학교 통신공학과 전임 강사

林 寅 七 (正會員) 第25卷 第8號 參照
현재 한양대학교 전자공학과 교수