EDIF Netlist를 이용한 PLD 설계용 툴 개발

김 희 석† · 변 상 준††

요 약

본 논문은 상용 툴인 OrCAD에서 생성한 디지털 회로의 EDIF 네트리스트를 이용하여 디지털 회로를 PLD로 구현하기 위한 PLD 설계 툴을 개발하였다.

EDIF 네트리스트를 이용하여 디지털 회로를 PLD로 구현하기 위해 각 셀(cell)간의 연결정보를 추출하는 연결정보 추출기(JIE)와 피드백(feedback)의 존재여부를 검색하는 피드백 노드 검출기(FND), 부울식을 생성하는 등의 알고리즘(BEG)들을 제안하였다.

또한 생성한 부울식을 최소화한 후, 최소화한 부울식의 입출력 변수 개수와 OR 텀의 수와 출력 특성을 고려하여 적합한 PLD 소자를 자동 선정하는 Auto select 기능과 상용 툴인 MyPLD에서 현재 제공하고 있는 PLD들 보다 용량이 큰 EPLD 타입의 GAL6001과 GAL6002의 JEDEC 파일 생성알고리즘도 제안하였다.

The Development of PLD Design Tool using the EDIF Netlist

Hi-Seok Kim† · Sang-Zoon Byun††

ABSTRACT

In this paper, the PLD design tool which realizes a digital circuit as PLD, by using EDIF netlist of the digital circuit designed at OrCAD have been developed.

This paper is proposed the following algorithms: JIE(Joined Information Extractor) which extracts the connecting information between both cells in order to realize the digital circuit as PLD using the EDIF netlist, FND(Feedback Node Detector) which look into whether feedback exists or not, BEG(Boolean Equation Generator) which generates a boolean equation, and so on.

Also, this paper is developed auto-select function which selects the PLD element with consideration of number of I/O variables of the minimized boolean equation, and algorithm generating JEDEC file of GAL6001 and GAL6002, having a forms of EPLD which is bigger than PLD.

1. 서 론

PLD(programmable logic devices)는 짧은 시간 내에 최적화된 회로의 구현이 가능하여 디지털 회로를 PLD 소자로 구현하는 PLD설계용 툴이 널리 보급되어 사용되고 있다.[1]-[9] 디지털회로를 PLD로 구현하는 방법에는 상태도를 작성하여 생성된 부울식을 PLD 소자로 구현하거나[9], 디지털회로의 동작특성을 HDL, VHDL과 같은 언어형태로 기술하여 PLD로 설계하는 방법[9] 등이 사용되고 있다. 상태도를 작성하여 부울식을 생성하는 MyPLD[2][3][4][5]의 경우, 사용자가 디지

※ 본 연구는 1996년도 교육부 반도체 분야 학술연구조성비(ISRC-96-E-2033)에 의하여 수행되었음.
† 정 회 원 : 청주대학교 전자공학과
†† 정 회 원 : 충남전문대학교 전자계산기과
논문접수 : 1997년 8월 6일, 심사완료 : 1998년 2월 5일

털 회로를 PLD로 구현하기 위해 부울식을 직접 추출하여 기술하여야 하며, 제공되는 소자(PAL16L8, PAL16R4, GAL16V8, GAL22V10)들도 설계용량이 작아서 부울식의 크기가 클 경우 PLD로 구현할 수 없고, 기술한 부울식의 입출력 변수의 수와 OR항의 수 등을 고려하여 회로에 적합한 소자를 자동으로 선택해 주는 기능이 없어 회로 구현시 많은 어려움이 있었다.[3][4][5][6]

따라서 본 논문에서는 상태도에서 부울식을 추출하지 않고 OrCAD 상에서 스키메틱도(schematic diagram) 형태로 작성된 디지털회로의 EDIF 네트리스트에서 부울식을 추출하여 MyPLD의 입력으로 제공하는 EDIF 판독기를 제안하였다. EDIF 판독기는 EDIF 네트리스트에서 각 게이트들의 연결 상태를 추출하는 연결 정보 추출기(Jointed Information Extractor)와 연결정보 추출기에서 추출된 연결 정보를 이용하여 피드백을 찾아내는 피드백 노드 검출기(Feedback Node Detector), 연결 정보 추출기와 피드백 노드추출기에서 검출된 정보를 PLD 구조에 적합한 2단 구조(Sum of Product)로 부울식을 생성할 수 있는 부울식 생성기(Boolean Equation Generator) 등으로 구성되어 있다.

또한, EDIF 판독기에서 생성된 부울식을 MyPLD에서 최소화 과정을 거쳐 입출력 변수의 수에 적합한 PLD를 자동으로 선정해주는 소자 자동 선택 기능과 MyPLD에서 제공되는 소자들보다 설계용량이 큰 EPLD형식의 GAL6001과 GAL6002의 JEDEC파일 생성 알고리즘 등을 제안하였다.[2][8]

2. EDIF 네트리스트를 이용한 PLD 설계용 툴의 구성

EDIF 네트리스트를 이용하여 디지털 회로를 PLD 소자로 구현하는 방법은 OrCAD상에서 기본 논리 게이트로 구성된 스키메틱도(schematic diagram)을 작성하여 EDIF 네트리스트를 생성한다. EDIF 네트리스트의 종류에는 여러 가지가 상용되고 있으나 본 논문에서는 OrCAD에서 생성되는 EDIF 2.0.0의 형식을 이용하였다.

EDIF 네트리스트는 헤더(header), 엔티티(entity), 회로 정보 부분으로 구성되어 헤더는 회로명과 EDIF 버전 등 회로도의 외형적인 정보들을 갖고 있으며, 엔티티는 회로에 사용된 셀(cell)들의 형명, 포트명 등의 정보를 가지고 있고 회로 정보 부분은 회로를 구성하는 게이트(gate) 또는 셀들간의 연결정보 등을 가지고 있다.

EDIF 네트리스트를 이용한 PLD 설계용 툴은 크게 EDIF 판독기와 MyPLD로 구성되어 EDIF 판독기는 EDIF 네트리스를 입력으로 받아서 연결정보추출기, 피드백 검출기, 부울식생성기를 통해 부울식을 생성하게 된다.

MyPLD는 생성된 부울식을 최소화하고 거기에 적합한 소자를 자동으로 선택하여 PLD 소자로 구현하게 된다. 그림 1에서는 EDIF 네트리스트를 이용한 PLD 설계용 툴의 전체 순서도를 나타내었다.

(그림 1) PLD 툴의 전체 순서도
(Fig. 1) Total flowchart of PLD design tool

2.1 EDIF 판독기의 구성

(1) 연결 정보 추출기(Joined Information Extraction)

연결 정보 추출기는 EDIF 네트리스트로부터 각 셀들의 정보와 연결 상태를 파악하여 셀 정보는 CIF(cell information file)에 연결정보는 JIF(Joined information file)에 각각 저장한다. 연결 정보 추출 알고리즘은 그림 2에 나타내었다.

```
Ext_Joined_Info(EDIF_FILE)
{
 Ext_Cell_Info(EDIF_FILE);
 // 셀 정보 추출
```

```
Add_Ref_Info();
// 참조 셀에 대한 정보
Ext_Joined_Info();
// 각 셀의 연결정보를 추출
}
```

(그림 2) 연결 정보 추출 알고리즘
(Fig. 2) Joined information extraction algorithm

(2) 피드백 노드 검출기(Feedback Node Detector)

피드백 노드 검출기는 회로상에 존재하는 피드백 정보를 검출하는 것이다. 회로에 존재하는 피드백을 검출하기 위해서는 연결 정보 추출기에서 추출한 연결정보를 이용하여 회로의 입출력노드와 각 게이트노드들을 그래프(graph)화한 후 출력단에서 입력단쪽으로 피드백이 있는 노드를 검색하여 피드백이 존재하는 노드를 검출하면 이 노드로 들어오는 피드백을 절단한 뒤, 새로운 출력노드를 생성하게 된다. 피드백이 존재하는 노드의 피드백을 절단하여 이 부분을 출력으로 선정하는 이유는 모든 피드백이 출력에만 존재하는 PLD 구조상의 특징 때문이다. 피드백 노드 검출 알고리즘은 BFS(Breadth First Search)검색방식이며 그림 3에 나타내었다.

```
Ext_Feedback_Node(G)
{
while(방문할 최종 출력 노드가 남아 있는 동안) {
OutNode = OutN(G); // 최종 출력 노드를 하나 입력
VisitN(OutNode); // OutNode를 방문
   }
}

VisitN(ParentNode)
{
 ID = 1;
 while(더 이상 방문할 노드가 없을 때까지) {
   while(tempNode = ChildN(ParentNode)) {
     // 자식 노드를 모두 방문할 때까지
     ID = ID + 1;
     if(tempNode가 이미 방문한 노드가 아니면)
{
tempNode.id = ID;
putN(tempNode); // queue에 put
}
else if(tempNode가 이미 방문한 노드라면) {
if(tempNode.id >= ParentNode.id)
FeedbackN = tempNode;
// 피드백 노드 집합에 tempNode를 추가
                    }
                  }
ParentNode = getN( ); // queue로부터 get
                }
```

(그림 3) 피드백 노드 검출 알고리즘
(Fig. 3) Feedback node detection algorithm

(3) 부울식 생성기(Boolean Equation Generator)

부울식 생성기는 피드백 노드 검출기에서 피드백이 존재하는 회로인 경우에는 새로 생성된 출력노드을 삽입하여 각 출력에 대해 그래프를 재구성한 다음, 입력변수노드부터 차례로 스택에 PUSH 하고 각 게이트노드(연산자)가 PUSH 되면 입력변수노드를 POP 한 뒤, 입력변수노드와 게이트노드를 서로 연산하여 출력변수에 해당되는 부울식을 생성하여 .EBU 파일에 저장한다. 부울식 생성 알고리즘은 그림 4에 나타내었다.

```
Ext_Bool_EQ(G)
{
while(방문할 최종 출력 노드가 남아 있는 동안) {
OutNode = OutN(G); // 최종 출력 노드를 하나 입력
pushN1(OutNode); // stack1에 push
VisitEQ(OutNode); // OutNode를 방문
ExtEQ(stack1Head);
        }
      }
VisitEQ(ParentNode)
{
while(더 이상 방문할 노드가 없을 때까지) {
while(tempNode = ChileN(ParentNode)) {
// 자식 노드를 모두 방문할 때까지
pushN1(tempNode); // stack1에 push
putN(tempNode); // queue에 put
        }
ParentNode = getN( ); // queue로부터 get
```

```
}
ExtEQ(stack1Head)
{
while(tempNode = popN1(stack1Head)) {
// stack에 노드가 남아 있을 동안
if(tempNode가 연산자라면)
tempNode = MakeEq(stack1Head, tempNode);
// 현재 연산자에 해당하는 변수들을
stack1에서 pop하여 부울식을 만든다.
pushN2(tempNode);
// stack2에 push
}
Write2File();
// 만들어진 식을 파일로 저장
}
```

(그림 4) 부울식 생성 알고리즘
(Fig. 4) Boolean equation generation algorithm

2.2 GAL6001, GAL6002 JEDEC 파일 생성 알고리즘

EDIF 판독기에서 생성된 부울식은 MyPLD상에서 JEDEC 파일을 생성하여 PLD 소자로 구현하게 되는데, 현재 보급되고 있는 MyPLD에서 제공되는 소자들 중 가장 큰 용량의 소자는 GAL22V10로 각 출력에서 사용할 수 있는 OR 텀의 수가 고정되어 있어 각 출력에 대한 OR텀의 수가 제한되어 있어서 MyPLD를 이용하여 디지털 회로를 구현하는데 많은 제한이 있었다.

따라서 본 논문에서는 OR 텀이 고정되어 있지 않아 하나의 출력이 최대 64개의 OR텀을 사용할 수 있는 EPLD 형식의 GAL6001, GAL6002소자의 JEDEC 파일 생성 알고리즘을 제안하였다.

GAL6001과 GAL6002 소자의 외형적인 핀의 구조는 10개의 입력 핀과 입출력으로 사용할 수 있는 10개의 양방향 I/O/Q로 구성되어 있고 내부구조는 입력을 제어하는 ILMC(Input Logic Macrocell)와 입출력을 제어한는 IOLMC(I/O Logic Macrocell)로 구성되어 있다.

GAL6001은 부울식의 전체 OR텀의 수를 최대 64개까지 유동적으로 사용할 수 있으며, 입력단과 입출력단의 핀을 각각 하나의 MUX를 이용하여 조합(combinational), 동기 레지스터(D-type register), 비동기 레지스터(D/E-type register)의 형태의 입력과 출력을 제어하도록 구성되어 있다. 또한 GAL6002 소자는 GAL6001소자와 외형적인 형태는 동일하지만 GAL6001소자가 MUX를 이용하여 입출력을 제어하는데 배해 GAL6002소자는 조합, 동기 레지스터, 비동기 레지스터의 입력과 출력의 형태를 각각 ILMC와 IOLMC를 이용하여 핀들을 제어할 수 있도록 구성되어 있다.

GAL6001, GAL6002의 JEDEC 파일은 88개의 열과 부울식을 퓨즈하는 64개의 행, 각 출력핀의 OE(Output enable)을 설정하는 10개의 행, 입력과 출력의 형태를 제어하는 3개의 행으로 구성되어 있다. GAL6001, GAL6002 JEDEC 파일 알고리즘은 우선 MyPLD에서 핀 할당한 정보와 부울식의 출력변수를 읽어들여 각 출력의 핀할당된 곳에 부울식을 퓨즈하고, 사용된 핀의 OE를 퓨즈한다음, 입력과 출력의 형태를 받아들여 마지막으로 퓨즈하게된다.

GAL6001, GAL6002의 JEDEC 파일 생성 알고리즘은 그림 5에 나타내었다.

```
GAL_logic[MAX_ROW][MAX_COL] // GAL6001, GAL6002의 fuse pattern

FUSE_MARKING( ) // fuse pattern의 초기화 및 marking
for i→23 downto i<13 // 각 출력핀에 할당된 부울식을 검사
j←l←k←0
while length_pin[i]>j
if pin[i][j] ← '*' // 연산자 * 이면 지정된 장소에 marking
then marking1(buf, k)
    l←0
else if pin[i][j]←'+'or pin[i][j]← EOF // 연산자 + 이면 라인 증가
then marking1(buf,k)
then marking2(i, k, buf1)
 l←0
 for l→0 to l<114
 if GAL_logic[l][k]≠ 'X'
 then GAL_logic[l][k]←'-'
 count ← k++
 else
 then buf[l++]←pin[i][j]
```

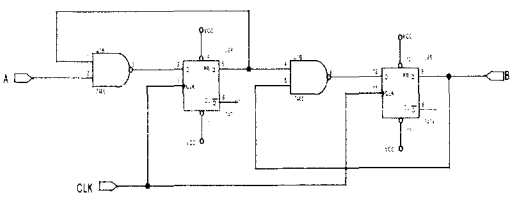
```
 j++

 // ILMC, IOLMC 퓨즈 생성
 LMC_FUSE( )
 if(GAL6001){
 GAL_logic[i][j]  ←  (IOPIN  =  REG ,
COM )?  X  : -
 }
 else if(GAL6002){
 for n←2 to n<24
    {
 GAL_logic[i][j]  ←  (IOPIN  =  REG ,
COM )?  X  : -
 }
 }
  // JEDEC 파일 생성
 JEDEC_GENERATOR( )
  for i ← 0 to i < MAX_ROW
    for j ← 0 to j < MAX_COL
 then
   GAL_logic[i][j]  ←  (GAL_logic[i][j]= X )?
0  :  1
```

(그림 5) GAL6001, GAL6002의 JEDEC 파일 생성 알고리듬
(Fig. 5) GAL6001, GAL6002 JEDEC file generation algorithm

3. 실험결과

디지털 회로를 OrCAD에서 작성하여 생성된 EDIF 네트리스트를 이용하여 PLD 소자로 구현하기 위해서는 본 논문에서 제안한 EDIF 판독기를 통해 부울식을 생성하고 이 부울식을 이용하여 MyPLD에서 PLD소자로 구현하면 된다. 예를 들어, 그림 6의 피드백이 있는 회로를 PLD 소자로 구현하는 설계과정은 다음과 같다.



(그림 6) 피드백이 있는 회로도
(Fig. 6) Feedback circuit

그림 6의 회로를 OrCAD 상에서 작성한 다음, 생성된 EDIF 네트리스트에서 연결 정보 추출기가 각 셀의 정보를 저장한 셀정보 파일과 각각의 셀들간의 연결 정보를 알 수 있는 연결 정보 파일을 생성한다. 연결 정보 추출기에서 생성된 셀 정보 파일은 그림 7에 연결 정보 파일은 그림 8에 나타내었다.

그림 7의 (a)는 EDIF 네트리스트에서 제공하는 7400과 7474에 대한 셀 정보를 본 논문에서 제안한 연결정보 추출기가 그림 7의 (b)와 같이 각 셀의 입력 개수와 출력개수, 입력명과 출력명으로 간략하게 변환한 것을 나타내었고, 그림 8의 (a)의 각 셀들의 연결 정보 부분을 기준 연결 정보로 변환하여 그림 8의 (b)에 나타내었다.

```
(cell &7400
 .....
(interface
 (port &I0 A (direction INPUT))
 (port &I0 B (direction INPUT))
 (port &I0_C (direction INPUT))
 (port &I0 D (direction INPUT))
 (port &I1 A (direction INPUT))
 (port &I1 B (direction INPUT))
 (port &I1 C (direction INPUT))
 (port &I1_D (direction INPUT))
 (port &O A (direction OUTPUT))
 (port &O B (direction OUTPUT))
 (port &O C (direction OUTPUT))
 (port &O D (direction OUTPUT))
 (port &VCC (direction INPUT))
 (port &GND (direction INPUT)))))
 .....
(cell &7474
 .....
 (interface
  (port &D A (direction INPUT))
  (port &CLK A (direction INPUT))
  (port &Q A (direction OUTPUT))
  (port &QBAR A (direction OUTPUT))
  (port &PR A (direction INPUT))
  (port &CL A (direction INPUT))
  (port &D B (direction INPUT))
  (port &CLK B (direction INPUT))
  (port &Q B (direction OUTPUT))
  (port &QBAR B (direction OUTPUT))
  (port &PR B (direction INPUT))
  (port &CL B (direction INPUT))
  (port &VCC (direction INPUT))
  (port &GND (direction INPUT)))))
```

EDIF

```
Cell &7400
NumberOfInput 2
NumberOfOutput 1
InputName
&IN0 &IN1
OutputName
&OUT
 .....

Cell &7474
NumberOfInput 4
NumberOfOutput 2
InputName
&D &CLK &PR &CLR
OutputName
&Q &QBAR
```

변환된 셀 정보

(a) (b)

(그림 7) 셀 정보 파일
(Fig. 7) Cell information file

```
(net A
 .....
  (portRef I1 A (instanceRef U1))
  (portRef A))
 .....
(net N00149
 .....
  (portRef Q A (intanceRef U2))
  (portRef I0 B (intanceRef U1))
  (portRef I0 A (intanceRef U1)))
 .....
```

EDIF

```
A       &G1.IN1
F1.Q  &G2.IN0  &G1.IN0
```

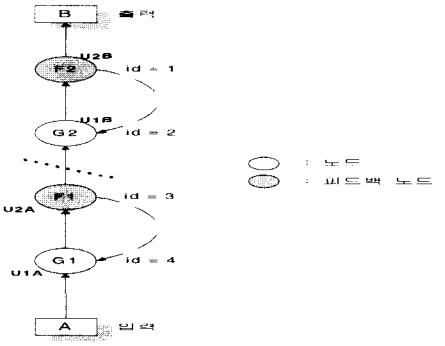
변환된 연결 정보

(a) (b)

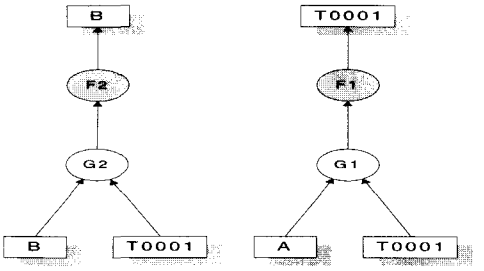
(그림 8) 연결 정보 파일
(Fig. 8) Joined information file

추출된 연결정보를 이용하여 피드백을 검출하는 과정은 회로의 입출력노드와 각 게이트노드를 하나의 그

래프로 구성하면 그림 9와 같다. 그림 9에서 BFS(Breadth First Search)방식으로 출력 B에서부터 입력 A 쪽으로 각 노드를 하나씩 방문하면서 방문한 곳에 단계값(id 값)을 설정한 뒤, F2 노드에 피드백이 존재하는지를 검출하기 위해 F2 노드와 G2 노드의 단계값과 비교하여 F2 노드의 단계값이 G2 노드의 단계값 보다 작거나 같으면 F2 노드를 피드백노드으로 인식하여 마킹하면 된다. F1 노드도 G1 노드의 단계값을 비교하여 F1 노드가 G1 노드의 단계값 보다 작거나 같기 때문에 피드백 노드로 결정한다. 피드백 노드을 검출한 다음 두 개의 피드백노드중 F1 노드와 G2 노드 사이를 절단하여 새로운 출력 T0001 선정하여 그래프로 재구성하면 그림 10과 같다.



(그림 9) 전체 회로의 그래프
(Fig. 9) Graph of circuit



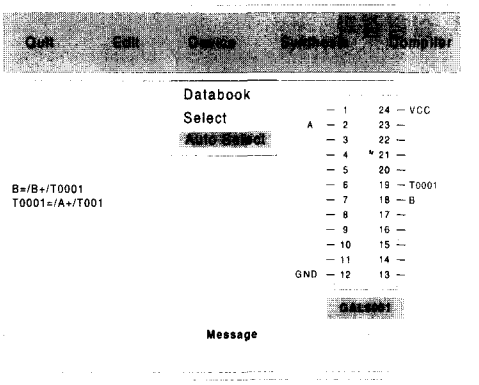
(그림 10) 피드백 검출 후 재구성된 그래프
(Fig. 10) Rearranged graph after detection of feedback

그림 10의 그래프는 피드백 검출기에서 피드백을 검출하여 새로 생성된 출력 T0001과 출력 B에 대해 재구성된 그래프로 사각형은 입출력을 의미하고 동그라미는 각 셀을 의미하며, 색칠된 것은 피드백을 갖는 셀을 의미한다. 그림 10을 이용하여 각 출력에 대해 부울생성기가 입력변수와 연산자의 연결 정보등을 종합하여 각 출력단에 대한 부울식을 생성하면 그림 11 과 같다.

B=/(B*T0001)
T0001=/(A*T0001)

(그림 11) 부울식 생성기에서 생성된 부울식
(Fig. 11) Boolean equation generated in Boolean equation generator

EDIF 판독기에서 생성된 부울식은 그림 12의 MyPLD의 핀 맵 에디터에서 부울식의 입력과 출력의 수를 고려하여 OR항의 수를 고려하여 자동으로 선택된 소자중에 GAL6001 소자를 선택하여 입력핀과 출력핀의 핀번호에 입력 변수와 출력 변수를 할당을 한 후 부울식을 입력하여 JEDEC 파일을 생성해야 한다. 그림 13은 MyPLD에서 생성된 JEDEC 파일을 나타내었다. JEDEC 파일에서 핀 맵이되지 않은 라인은 생략하였다.



(그림 12) MyPLD의 핀 맵 에디터
(Fig. 12) Pin map Editor of MyPLD

그림 13의 JEDEC파일을 PLD로 구현하여 실험한 결과와 PLD 설계용 상용툴인 MINC에서 생성된 JEDEC 파일을 PLD로 실험한 결과 정확한 동작 결과가 나왔으며, 본 논문에서는 몇 가지 예제들를 더 선정하여 MyPLD에서 생성된 JEDEC 파일과 상용화된 PLD 설계용 툴인 MINC에서 생성된 JEDEC 파일을 PLD로 구현하여 실험한 결과가 동일함을 알 수 있었

```
GAL6002
_SYNC.*
QP24*
QF8294*
F0*
L000011101111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111101111111111111111111111111*
L011411111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111101111111111111111111111111*
L022811111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111101111111111111111111111111*
L034211111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111101111111111111111111111111*
                                                    .
                                                    .
                                                    .
L729600000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000*
L737400000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000*
L745200000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000*
L753000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000*
L760811111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111*
L768611111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111*
L776400000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000*
L784200000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000*
L792000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000*
L799800000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000*
L807600000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000*
L815411111111111111111111111111111111111111111101110111111111111111111101010111111111111*
L823211111111111111111111111111111111111111111111111111111111111111111111111111111111111111*
L8310 11111111111111111*
E*
C5F34*
9AAD
```

(그림 13) 생성된 JEDEC 파일
(Fig. 13) JEDEC file generated

다. 표 1은 MyPLD와 MINC에서 생성된 JEDEC 파일들의 정확성을 검사하기 위한 용도로 사용되는 데이터의 합인 CHECKSUM을 비교한 표이다.

〈표 1〉 MyPLD와 MINC의 CHECKSUM 비교
〈Table 1〉 Comparison of MyPLD and MINC

예제명	사용 Device	MINC CHECKSUM	MyPLD CHECKSUM
4진카운터	GAL6001 GAL6002	CAE4B CB30A	C472B C4B36
4진 Gray 카운터	GAL6001 GAL6002	CE98A CEE49	CEA18 CEE23
BCD 7-Segment Decoder	GAL6001 GAL6002	CA64C CAB0B	C3DF9 C4204
4-BIT 가산기	GAL6001 GAL6002	CE064 CE523	CDDB8 CE1C3

4. 결 론

본 논문에서는 디지털 회로를 OrCAD상에서 작성하여 생성되는 EDIF 네트리스트를 이용하여 PLD로 구현할 수 있는 EDIF 네트리스트를 이용한 PLD 설계용 툴를 제안하였다. EDIF 네트리스트를 이용한 PLD 설계용 툴은 부울식을 생성하는 EDIF 판독기와 생성된 부울식을 PLD소자로 구현하는 MyPLD로 구성되어 있다. EDIF 판독기는 EDIF 네트리스트에서 회로도의 각 연결 상태를 추출할 수 있는 연결 정보 추출기와 피드백이 존재여부를 검출하는 피드백 노드 검출기, 피드백 노드 검출기에서 생성된 출력정보를 이용하여 부울식을 생성할 수 있는 부울식 생성기로 구성되어 있다.

또한, EDIF 판독기에서 생성된 부울식을 PLD 소자로 구현하기 위해 MyPLD 상에서 최소화 과정을 거쳐 부울식의 입출력의 수에 맞는 소자를 자동으로 선정해 주는 소자 자동 선택 기능과 대규모 디지털 회로의 구현이 가능한 EPLD형식의 GAL6001과 GAL6002의 JEDEC파일 생성 알고리즘을 제안하였다.

본 논문에서 제안한 EDIF 네트리스를 이용한 PLD 설계용 툴를 이용하여 몇 가지 예제들을 GAL6001,GAL6002 소자로 구현하여 실험한 결과와

상용화 툴인 MINC에서 생성된 JEDEC파일의 실험결과와 동일하였다.

향후 연구 과제로는 GAL6001, GAL6002 소자보다 게이트 수가 많은 대용량 PLD인 CPLD를 사용하여 회로를 구현할 수 있는 CPLD설계 시스템 개발에 대한 연구가 계속되어야 할 것이다.

참 고 문 헌

[1] Lattice, "Lattice Data Book"1994

[2] 김 희석, 이 근만, 임 인칠, "상태합성기(State Machine Synthesizer)설계를 위한 상태 CHDL 개발 및 Two-level minimizer 개발에 관한 연구" 대한전자공학회 논문집-A, 제 29권 4호, pp.83-89, 1992.

[3] 하 재희, 정 홍구, 변 상준, 김 희석, "PAL합성을 위한 JEDEC파일 알고리듬에 관한 연구", 대한전자공학회 전자계산, 반도체.재료 및 부품, CAD 및 VLSI설계 합동학술발표논문집, p.91-94, 1992.

[4] 하 재희, 정 홍구, 변 상준, 김 희석, "PLD 설계를위한 툴(PLD Designer)에 관한 연구", 대한전기,전자,통신학회 충북지부 학술논문집, pp.93-97, 1992.

[5] 김희석, 원충상, "PLD 설계용 툴 개발에 관한 연구", 한국정보처리응용학회 논문집, 제1권, 제3호, pp391-397(1994).

[6] 김재진, 김성무, 조남경, 변상준, 원충상, 김희석, "PLD partition을 고려한 PLD 설계용 툴 개발", 대한전자공학회 추계종합학술대회 논문집, pp.1415-1418, (1994).

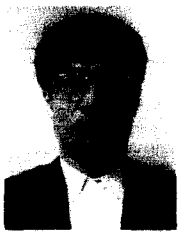
[7] 조남경, 조양기, 김재진, 변상준, 김희석, "EDIF 네트리스트를 이용한 PLD 구현", 대한전자공학회 CAD 및 VLSI 설계 연구회 학술발표회 논문집, pp.60-63, 1996

[8] 전종식, 조양기, 김재진, 변상준, 김희석, "EPLD 설계를 위한 JEDEC 파일 생성 알고리듬 개발", 대한전자공학회 하계종합학술대회 논문집, pp.587-590, 1996

[9] Parag K. Lala, "PLD Digital system design using programmable logic devices", Prentice Hall publication, 1990.

[10] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, "Introduction to algorithms", McGraw-Hill Book Company, 1992.

[11] Paul Stanford and Paul Mancuso, "EDIF : Electronic Design Interchange Format Version 200", Electronic Industries Association EDIF Steering Committee, 1989



김 희 석

1977년 한양대학교 전자공학과 졸업(공학사)

1980년 한양대학교 전자공학과 (CAD 전공) 졸업(공학석사)

1985년 한양대학교 전자공학과 (CAD 전공) 졸업 (공학박사)

1987년~1988년 미국 Univ. of Colorado at Boulder 객원 교수

1989년~1993년 청주대학교 전자공학과 부교수

1993년~현재 청주대학교 전자공학과 교수

관심분야 : CAD, 컴퓨터 아키텍춰, 컴퓨터 알고리즘

변 상 준

1988년 청주대학교 전자공학과 졸업(공학사)

1990년 청주대학교 전자공학과 (전자계산 및 계산기 전공) 졸업(공학석사)

1997년 청주대학교 전자공학과(전자계산 및 계산기 전공) 박사수료

1993년~1996년 충남전문대학 전자계산기과 전임강사

1997년~현재 충남전문대학 전자계산기과 조교수

관심분야 : CAD, 컴퓨터 아키텍춰, 컴퓨터 알고리즘