論文 91-28A-9-14

스키매틱 자동생성기의 개발

(The Development of Automatic Schematic Generator)

裵英煥* 白英錫* 朴星範* 李省奉* 張泳祚* 李玄燦**

(Young Hwan Bae, Young Seok Baek, Sung Bum Park, Seong Bong Lee, Young Jo Jang, and Hyun Chan Lee)

要 約

본 논문에서는 회로의 네트리스트로부터 스키매틱 다이아그램을 자동으로 생성하는 스키매틱 생성 알고리듬을 제안한다. 스키매틱 생성에 있어서 가장 중요한 문제는 결과로서 생성된 스키매틱 회로가 인간에게 얼마나 이해하기 쉽고 간결한가에 있다. 그러나 이러한 요구를 만족하는 스키매틱 생성은 매우 어려운 문제로서 본 논문에서는 전처리, 논리적 배치, 핀할당 및 배치 개선, 전체 배선, 상세 배선 등의 5단계의 세부문제로 나누어 각 단계에 있어서 이러한 목적 함수를 가능한 극대화 하도록 하였다.

본 스키매틱 생성기는 C 언어로 구현되었으며 논리합성의 결과를 그림으로 표현하여 설계자의 이해를 돕고, 설계자의 지식을 설계에 반영하기 쉽게 한다.

Abstract

In this paper, an algorithm for automatic schematic generation which creates schematic diagram from netlist are proposed. The important objectives on schematic generation are readability and clarity of resulting schematics. Each stage of generation should aim at enhancing these objectives. For this reason, schematic generation problem is divided into 5 subproblems; preprocessing, logical placement, pin assignment and improvement of placement, global routing, and detailed routing.

The algorithm is implemented in C language, and it generates schematics from the results of logic synthesis in order to make it easy for designers to understand the design and reflect their knowledge into design.

*正會員, 韓國電子通信研究所 自動設計研究室
(Design Automation Section, ETRI)

**正會員, 弘益大學校 産業工學科
(Dept. of Industrial Eng., Hongik Univ.)

接受日字 : 1991年 7月 1日

(※ 본 논문의 연구내용은 '90 과기처 특정연구과제인 "자동설계 환경구축에 관한 연구 III"에서 수행한 것임)

I. 서 론

최근 상위 수준 및 논리 합성에 대한 연구가 진행됨에 따라, 종래에 스키매틱 입력(schematic entry)시스템에서 설계자가 스키매틱 다이아그램을 입력했어야 했던 많은 작업들이 자동화되어 설계자가 알고리듬 수준 또는 레지스터 전송 수준에서 구현하고자 하는 시스템의 동작을 기술만하여 주면 자동적으로 원하는 구조(structure)의 합성이 얻어지는 단계에 이르렀다. 특히 논리 합성의 경우는 이미 상용화된

시스템의 출현에까지 이르게 되어 실제 칩의 설계에 이용되고 있고, 그 성능 또한 우수하다는 평가를 받고 있다.

일반적으로 컴퓨터에 의한 자동합성은 설계자의 숙련된 기술을 이용하면 더욱 양질의 설계를 얻을 수 있다. 그러나 현재까지 개발된 논리합성 시스템의 경우 설계자가 합성의 과정에 개입할 수 있는 방법은 입력 화일의 재작성 또는 제약조건의 수정 등으로 국한 되었다. 이러한 단점을 해결하기 위한 노력으로 U. C. Berkeley 대학에서는 MIS[1]라는 대화식 논리합성 시스템을 내놓았으나 입력 및 출력 방식이 문자 형태로 국한되어 있어 사용자의 이해가 매우 어려워 대화식 논리합성기라는 원래의 목적에 적합하지 못하다는 단점이 지적되고 있다.

인간은 단순한 문자 보다는 도형적 정보에 이해가 훨씬 빠르므로 설계 각 단계의 합성 중간 정보를 문자보다는 도형으로 전환하여 이를 설계자에게 제공하는 것이 보다 능률적인 설계 작업을 수행할 수 있다. 이러한 작업의 수행을 위해서는 스키매틱 다이어그램을 자동으로 생성하여주는 스키매틱 생성기가 필요하다.

스키매틱 생성기는 합성의 과정 또는 결과로 생성된 네트리스트를 입력으로 사용자의 이해가 쉽도록 도형적 정보를 합성하여 자동으로 스키매틱을 생성한다. 또한 현재 상용화되어 있는 스키매틱 입력 시스템들 간의 그래픽 데이타 형태가 서로 다르기 때문에 이들 시스템들간의 호환성 또한 문제가 되고 있다. 이들 시스템들 간에 네트리스트로 인터페이스를 이루고 스키매틱 생성기를 이용하여 각 시스템에 맞는 스키매틱을 자동으로 생성한다면 복잡한 스키매틱을 다시 그려야하는 시간과 노력이 절감될 것이다.

본 스키매틱 생성기는 ETRI의 논리 및 제어 디자이너[13]의 일부로서, 논리합성의 결과로 생성된 네트리스트로부터 스키매틱 다이어그램을 자동으로 생성하여 준다. 이 논리 및 제어합성기는 현재 ETRI 에서 개발중인 실리콘 컴파일러의 부분 시스템으로서 상위수준 합성(high level synthesis)의 데이타패스 합성의 결과로부터 제어정보를 상태천이표(state transition table)의 형태로 받아 제어부를 합성한다. 먼저 제어 합성 단계에서는 상태 최소화와 상태코드 할당이 수행되며 논리합성 단계에서는 2단 논리 합성, 다단 논리 합성 및 technology mapping을 거쳐 게이트 수준의 네트리스트를 생성한다. 스키매틱 생성기는 논리합성 중간단계 또는 결과로 생성된 네트리스트를 입력으로 스키매틱 회로도를 생성한다. 그림 1은 논리 및 제어 디자이너의 설계 단계를 보여주고 있다.

스키매틱 생성에 관한 연구는 1980년대에 들어 발표되기 시작하여, Brennan[2]은 스키매틱 상에서 심볼의 자동 배선에 관하여 연구 하였고, Kumar[3]는 배치와 배선 문제에 관한 연구를 발표하였다. 1983년에 발표된 HAL 시스템[4]은 스키매틱의 미적 속성의 극대화를 위하여 지식 기반 전문가 시스템을 발표하였다. 스키매틱의 생성을 위한 배치, 배선 단계의 지식들을 룰(rule)로 저장하여, 이 룰을 각 단계에서 적용하여 스키매틱을 생성하였다. 1986년에 발표된 AutoDraft 시스템[5]은 IC 레이아웃 알고리듬을 스키매틱 생성기에 적용하여 Adjacency Constructive 초기 배치와 배치 개선, 채널 배선기를 이용하여 최종 배선을 수행하였다. IDDD 시스템[6]은 논리 단계와 기하학적 단계로 나누어 논리 단계에서는 각 심볼들을 신호선의 흐름에 따라 논리적 그리드에 배치하고 그리드 상에서 논리적 배선을 수행한다. 기하학적 단계에서는 실제 절대좌표를 구하여 배치개선, 배선 및 컴팩션(compaction) 등을 수행한다. 1987년에 발표된 Vision 시스템[7]은 회로상의 피이드백 루프(feedback loop)로 인하여 일관성 있는 신호선 흐름결정에 장애가 되는 사이클 형성 아크의 제거를 위하여 incidence 행렬을 사용하였으나 심볼의 수가 증가함에 따라 행렬의 크기가 $O(n^2)$로 증가한다는 단점과 사이클 형성 아크를 제거할 때 전체 스키매틱의 열의 수를 고려하지 않은점 등이 지적되고 있다.

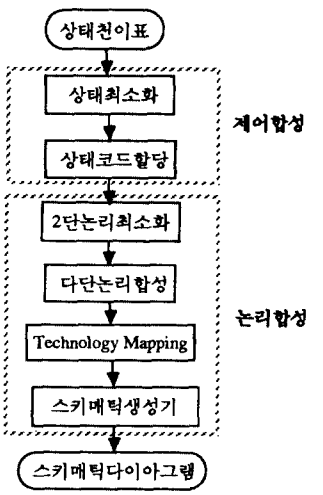


그림 1. 논리 및 제어 디자이너 흐름도.
Fig. 1. Design flow of logic and control designer.

본 논문에서는 스키매틱 생성시 고려하여야 하는 여러 미적 속성의 극대화를 위한 배치 및 배선 알고리듬을 제안한다. 순서회로내에 존재하는 피이드백 루프의 제거를 위하여 노드그룹을 구성하고 노드그룹의 특성에 따라 미적 속성을 고려하여 사이클을 제거한다. 또한 이미 많은 연구가 이루어진 IC 레이아웃 알고리듬들을 스키매틱 생성문제에 적합하도록 변형하여 설계자가 이해하기 쉽고 간결한 스키매틱을 생성하는 효율적인 알고리듬을 제안한다.

본 논문의 구성은 먼저 Ⅱ장에서 스키매틱 생성문제의 특성과 본 논문에서의 전체적인 접근방식에 관하여 설명하고, Ⅲ장에서 각 세부 단계에서의 알고리듬을 논한다. Ⅳ장에서는 생성기의 구현과 실험을 통한 스키매틱 생성 예제를 보인다. 끝으로 Ⅴ장에서는 결론과 앞으로의 연구 방향을 제시한다.

Ⅱ. 스키매틱 생성 문제

스키매틱 생성 문제와 일반적인 IC의 레이아웃 생성 문제는 많은 부분에서 서로 유사하지만 서로간의 목적함수에 있어서는 상이하다. IC의 레이아웃 생성 문제는 전체적인 칩 면적의 최소화 및 동작속도의 최소화를 목적으로하여, 일반적으로는 총 배선길이를 줄이거나 또는 특정 신호선이 주어진 제약 조건 내에서 배선되도록 배치와 배선을 수행한다. 그러나 스키매틱 생성의 경우 가장 중요한 문제는 가능한한 인간에게 이해가 쉽고 명확하도록 생성되어져야 한다. 그러나 이러한 목적 함수는 대체로 모호하기 때문에 숙련된 스키매틱 설계자의 지식을 바탕으로 다음과 같은 원칙에 따라 스키매틱을 생성한다.

○전체적인 신호선의 흐름은 일관성 있는 방향을 갖도록 배치하여 이해가 쉽게 한다. 즉, 입력단에서 출력단까지 왼쪽에서 오른쪽으로 신호선이 흐르도록 스키매틱을 생성한다.

○총 배선길이의 최소화가 최우선의 목적함수가 아니다. 그보다는 인간의 이해가 쉽도록 서로간에 연결도가 많은 심볼들 간에는 가능한 서로 가까이 배치하고, 그 목적이 허용하는 한 신호선의 길이도 최소화 한다.

○각 신호선은 꺾임 및 교차수가 최소가 되도록 심볼의 배치 및 신호선 배선을 수행한다. 이는 신호선에 꺾임과 교차수가 많아지면 스키매틱이 복잡해지고 이해하기가 어려워지기 때문이다.

○스키매틱의 크기를 최소화하는 것이 목적은 아니다. 그러나 필요 이상으로 스키매틱이 커지는 것은 바람직하지 않으므로 적당하게 면적에 제한을 가한다.

○버스(bus) 신호선의 경우는 다른 신호선과 분리하여 스키매틱의 상단 또는 하단에 모아서 배선하여 스키매틱을 간략하게 한다.

이러한 원칙에 맞는 스키매틱을 생성하기 위하여는 기존에 이미 많은 연구가 이루어진 IC 레이아웃의 알고리듬을 그대로 이용할 수 없다. 따라서 본 논문에서는 스키매틱 생성 문제를 잘 정의된 5가지 부문제(subproblem)로 나누고, 각 문제에 대하여 단계별 목적에 가장 적합한 알고리듬을 개발 또는 기존의 레이아웃 알고리듬을 응용하여 스키매틱 생성에 적합한 배치, 배선 알고리듬으로 제안하고 이를 구현한다. 즉 스키매틱 생성은 전처리, 논리적 배치, 핀할당 및 배치 개선, 전체 배선, 상세 배선의 5단계로 나뉘어진다.

먼저 전처리 단계에서는 논리합성의 결과로부터 생성된 네트리스트를 입력으로 이를 DAG(directed acyclic graph)로 변환한다. 이를 위해서는 입력회로를 DG(directed graph)로 구성하고 DG 상에서 사이클을 형성하는 아크(arc)를 제거한다. 논리적 배치 단계에서는 가상의 논리적 그리드상에 심볼을 배치하는 단계로서, DAG의 입력단으로부터 출력단까지 각 노드의 레벨을 계산하여 그리드상에서의 각 심볼의 열(column)의 좌표를 정한다. 각 심볼들의 행(row)의 좌표를 결정하기 위하여 각 심볼들간의 연결 관계를 고려하여 신호선들간의 교차수가 가능한 최소가 되도록 심볼들의 상대적인 행좌표를 결정한다.

논리적 배치 단계가 끝난 후 각 심볼들에 대하여 핀할당 및 배치 개선을 행한다. 핀 할당이란 한 심볼에 대하여 등가핀들에 연결되는 네트들을 교환하여 신호선들간의 교차수가 가능한 최소가 되게 한다. 배치 개선 단계에서는 각 열들 간의 심볼들을 연결한 신호선의 꺾임이 가능한한 최소가 되도록 심볼들의 위치를 조금씩 움직여 배치를 개선한다. 이단계 이 후에는 각 심볼들은 실제적인 y좌표를 갖게된다.

심볼의 배치가 끝난 후 전체 배선 단계에서는 열의 차가 2 이상 되는 심볼들간의 배선을 위하여 전체 배선을 수행한다. 이 단계에서는 열들간에 상세 배선이 이루어지지 않아 실제의 좌표를 구할 수 없기 때문에 각 열 상에서의 피이드스루(feedthrough)만을 설정하여 놓고 실제 배선은 상세 배선 단계에서 수행한다. 신호선 배선은 가능한 한 꺾임이 없이 배선이 되어야 스키매틱을 이해하기 쉬우므로 선분탐색(line-search) 알고리듬[12]을 이용하여 피이드스루를 할당한다.

전체 배선 단계가 수행된 후 각 열들 간에 배선을

수행할 채널을 정의하고 이 채널 상에서 채널 배선기[10]를 이용하여 상세 배선을 수행한다. 이 상세 배선 단계에서는 앞의 전체 배선 단계에서 피이드스루 할당으로 결정된 신호선의 연결도 수행된다. 각 열들 간의 채널 배선이 수행됨과 동시에 각 심볼 및 신호선의 절대 좌표가 결정된다. 스키매틱 생성이 끝난 후 결과는 그래픽 화면 상에 그려진다.

Ⅲ. 알고리듬

본 장에서는 스키매틱 생성 각 단계에서의 알고리듬을 설명한다. 문제의 복잡성으로 인하여 전체 문제를 전처리, 논리적 배치, 핀할당 및 배치 개선, 전체 배선, 상세 배선의 5단계로 나누어 수행한다.

1. 전처리 과정

전처리 단계는 실제 배치, 배선 단계에 들어가기 이전에 내부 데이타 구조를 형성하고, 배치단계에 제약을 주는 사이클 형성 아크를 제거한다. 논리 합성 시스템으로부터 생성된 게이트 수준의 네트리스트를 읽어들여 내부적으로 DG를 형성한다. DG는 그림 2(b)에서와 같이 노드들과 이들간의 방향성 에지들로 구성되며 DG의 각 노드는 입출력 핀 또는 게이트 심볼에 대응되고, 각 방향성 에지는 팬아웃(fanout) 게이트에서 시작하여 각각의 팬인(fanin)게이트를 지시한다. 그림 2(a)의 회로가 그림 2(b)의 DG로 재구성된 예를 보여준다.

심볼의 배치는 열의 좌표를 구하는 수평 배치 단계와 행의 좌표를 구하는 수직 배치 단계로 나누어 행하여 지는데, 수평적 배치는 신호의 흐름이 입력단에서 시작하여 출력단까지 좌에서 우로 진행하도록 배치해야 스키매틱의 이해가 쉽다. 이를 위해서는 DG상의 각 입력단으로부터 출력단까지 각 방향성 아크에 대하여 팬인 노드가 팬아웃 노드의 오른쪽에 위치하도록 배치하여 신호선의 전체적인 흐름이 왼쪽에서 오른쪽으로 흐르게 한다. 그러나 DG의 방향성 아크들 사이에 사이클이 존재하면 이러한 배치를 하는 것이 불가능하게 된다. 따라서 전처리 단계에서는 그래프상에서 사이클을 형성하는 아크들을 찾아내어 이를 제거한다.

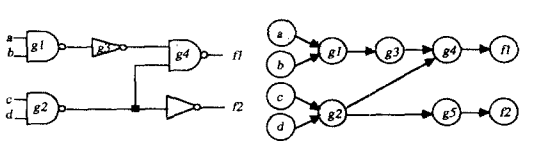


그림 2. Directed graph로의 변환
(a) 입력회로 (b) directed graph.

Fig. 2. Transformation to directed graph.
(a) intput cricuit, (b) directed graph.

DG상에서 사이클을 형성하는 방향성 에지들을 찾아내기 위하여 깊이 우선 탐색(depth first search) 알고리듬을 이용한다. 즉 그래프의 각 입력단에서 시작하여 방향성 에지가 지시하는 노드들을 깊이 우선 방식으로 탐색하다 이미 자신이 방문했던 노드를 다시 만나게 되면 DG내에 사이클 루프가 발견되게 된다. 이 경우 사이클을 형성하는 각 노드들과 방향성 에지를 하나의 노드 그룹 NG(node group)로 구성한다. 하나의 노드 그룹을 형성하는 노드의 수에 따라 다음과 같이 세 그룹으로 분류할 수 있다.

DG 상의 각 노드 그룹 NG에 대하여,

i) 단일 노드그룹, if ||NG||=1

ii) 2-노드 그룹, if ||NG||=2

iii) 다노드 그룹, if ||NG||>2

먼저 단일노드 그룹의 경우는 사이클을 형성하는 노드와 방향성 아크는 하나이므로 이 아크를 제거한다. 그림 3(a)의 경우는 단일노드 그룹을 보여주고 있다. 2-노드 그룹의 경우는 두 노드에 대응하는 게이트들이 래치(latch)를 형성하고 있으므로 이 두 게이트를 서로 분리하여 배치한다면 회로의 이해가 매우 어려울 것이다. 따라서 이 두 게이트를 하나의 래치 게이트로 묶어서 배치해야 한다. 또한 두 노드간의 사이클을 형성하는 신호선의 경우 그림 3(b)에서와 같이 45° 사선을 이용하여 배선하여야 하나 이를 나중의 상세 배선 단계에서 배선하고자 할 경우 채널 배선기가 45° 와이어(wire)를 배선해야 한다는 어려움이 있다.

이 문제의 해결을 위하여 본 시스템에서는 심볼을 자동으로 생성하여 주는 심볼 생성기(symbol generator)를 이용하여 그림4와 같이 두 심볼 게이트를 포함하면서 서로간에 연결된 피드백 아크를 45°와이어를 이용하여 연결한 새로운 래치 심볼을 자동으로 생성하고 사이클을 이루는 두 노드를 이 새로운 게이트에 대응하는 하나의 노드로 대치한다. 그림 3(b)는 이러한 2-노드 그룹의 예를 보여주고 있다.

다노드 그룹의 경우는 사이클을 형성하는 어떤 아크를 제거하는가에 따라 전체 열의 수가 늘어날 수 있기 때문에 그림 3(c)와 같이 사이클 아크의 제거로 생성된 DAG의 입력단으로부터 출력단까지의 전체 레벨 수가 최소가 되도록 사이클을 제거한다.

2. 논리적 배치

논리적 배치 과정은 전처리 과정에 의하여 생성된

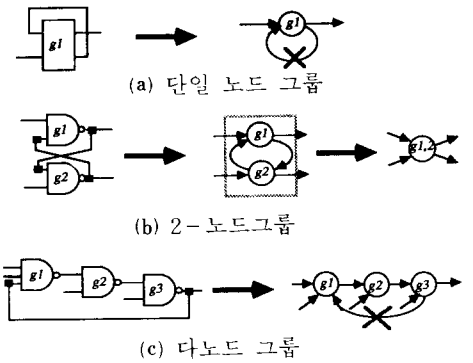


그림 3. 노드 그룹
(a) 단일 노드 그룹 (b) 2-노드 그룹
(c) 다노드 그룹

Fig. 3. Node group.
(a) Single node group, (b) Two-node group,
(c) Multi-node group.

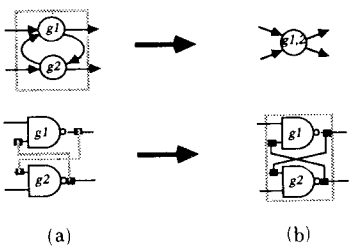


그림 4. 심볼 생성기
(a) 래치 회로 (b) 생성된 래치 심볼

Fig. 4. Symbol generator.
(a) Latch circuit,
(b) Generated latch symbol.

DAG의 각 노드들을 그림5와 같은 도면의 가상 그리드 상에 배치하기 위하여 상대적인 열과 행의 위치를 결정하는 단계로서 가로 배치와 세로 배치의 두 단계로 나누어 수행한다. 가로 배치는 각 노드들의 열의 좌표를 구하는 단계이고, 세로 배치는 각 열에 속하는 노드들의 행 좌표를 구하는 단계이다. 각 그리드 격자는 정수의 행, 열 좌표를 가지며, 그리드 격자에는 스키매틱의 심볼이 배치된다. 각 그리드 격자의 크기는 전체 심볼들의 크기를 고려하여 결정되며, 격자의 크기보다 큰 심볼은 하나 이상의 그리드 격자에 배치가 된다. 격자와 격자 사이에는 배선을 위한 수평 채널과 수직 채널이 존재하는데 수평 채널의 배선은 전체 배선 단계에서 신호선의 피이드스루가 할당되며, 수직 채널은 최종 단계인 상세 배

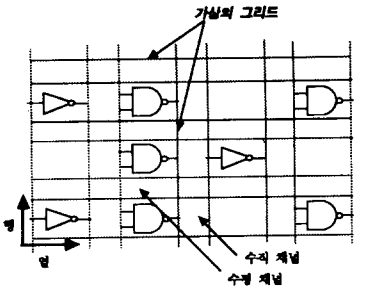


그림 5. 도면상의 가상의 그리드
Fig. 5. Vritual grid on schematic.

선 단계에서 채널 배선기에 의하여 상세 배선이 이루어 진다. 하나의 큰 심볼이 여러 그리드 격자를 차지하여 배치되는 경우에 이들 격자들간의 수평 채널은 심볼에 의하여 차단되므로 신호선 배선시 이 영역을 피하여 배선한다.

1) 가로배치

가로 배치 단계는 DAG의 각 노드들의 그리드 상에서의 열의 좌표를 구하는 단계이다. 전처리 과정에서 생성된 DAG의 각 노드 n에 대하여 다음 조건을 만족하도록 열의 좌표 COL(n)을 구한다.

DAG상의 모든 노드 n에 대하여, m은 노드 n의 팬인 노드일 때,

i) $COL(m) < COL(n)$,

ii) $\sum_{\text{for all node } n} \max_{m \in \text{fanin nodes of } n} COL(m)$이 최소.

즉 위의 조건 i)은 각 노드가 배치될 열의 좌표는 그 노드의 모든 팬인 노드의 열 좌표보다 큰 값을 갖도록하여, 신호선 흐름이 좌에서 우로 일관성을 유지하고자 하는 것이다. 그러나 각 노드의 열의 좌표가 무한정 큰 값을 갖게되면 생성될 스키매틱의 레벨의 수가 너무 커지므로 i)의 조건은 만족하되 전체적으로 가장 최소의 단 수를 유지하도록 ii)의 조건을 만족하여야 한다. 따라서 이러한 조건을 만족하도록 각 노드n의 열의 좌표 COL(n)을 구하면 아래 식과 같다.

$$COL(n) = \max_{m \in \text{fanin nodes of } n} COL(m) + 1$$

즉, 다시 말해서 각 노드의 열의 좌표는 입력단으로부터 그 노드까지의 최장경로가 되며, 각 노드까지의 최장 경로를 구하는 알고리듬은 그림6과 같이 입력단으로부터 출력단에 이르기까지 DAG상을 넓이 우선 탐색(breath first search)하여서 구할 수 있다.

```
procedure column_place(DAG)
DAG : Directed Acyclic Graph;
{
  각 노드들의 column 값을 0으로 초기화 한다;
  for(DAG의 각 입력노드 Ni에 대하여){
    노드 Ni를 큐에 넣는다;
    while(큐에 노드가 있는 동안 계속 수행){
      Ntmp<=큐로부터 하나의 노드를 꺼낸다;
      L=COL(Ntmp)+1;
      for(Ntmp의 각각의 fanout 노드 Nfanout에 대하여){
    if (COL(Nfanout)<L){
        COL(Nfanout)=L;
        Nfanout을 큐에 넣는다;
        }
      }
    }
  }
}
```

그림 6. 가로 배치 알고리듬
Fig. 6. Horizontal placemont algorithm.

2) 세로 배치

열의 좌표를 구하는 가로 배치 단계가 끝난 후 각 열안에서 노드들의 행 좌표를 구하는 세로 배치 단계가 수행된다. 세로 배치 단계에서는 신호선의 교차 및 꺾임의 수가 가능한 최소가 되도록 행의 좌표를 결정한다. 신호선들간의 교차수를 최소화 하도록 행의 배치를 수행하는 알고리듬은 May[8]에 의한 것이 좋은 결과를 얻을 수 있으나, 계산 시간이 매우 길다는 단점이 있다. Swinkels[9]에 의하여 신호선의 길이의 최소화를 목적함수로 하는 배치 알고리듬이 대개의 경우 신호선간의 교차수도 크게 줄일 수 있다는 제안을 통하여 좋은 배치 결과를 얻을 수 있었다. 즉 총 배선길이의 최소화가 세로 배치 단계의 목적함수는 아니지만 배선길이의 최소화가 대개의 경우 신호선간의 교차수도 적은 배치를 얻게한다.

본 논문에서는 IC 레이아웃의 배치 알고리듬의 하나인 FDR(force directed relaxation) 알고리듬[11]을 수정하여 일차원 행의 배치에 적용하였다.

먼저 배치 단계를 각 모듈에 작용하는 힘의 평형 위치를 찾는 물리적 문제와 연관시켜 신호선 연결관계가 있는 두 모듈간에 스프링 상수 k인 스프링으로 연결되어 있다고 가정할 때, 거리벡터 $\vec{S}$만큼 떨어진 두 모듈간에 작용하는 장력 $\vec{F}$는 Hook의 법칙에 의하여 $\vec{F}=k*\vec{S}$ 이다. 즉 이 두 모듈간에는 서로 가까워 지려는 $\vec{F}$만큼의 힘이 작용하여 다른 힘이 작용하지 않으면 두 모듈은 최대로 접근하여 거리벡터 $\vec{S}$ 및 $\vec{F}$가 0에 이르는 평형 상태에 이르게 될 것이다.

이 원리를 총 배선길이가 최소가 되는 배치를 구하는데 이용하면, 한 모듈 M에 연결관계가 있는 각 모듈 i와의 거리벡터를 $\vec{S}_{Mi}$, k는 연결된 신호선의 가중치(weight) 또는 연결 신호선수(connection number) C_{Mi}가 된다. 따라서 모듈 M에 작용하는 총 힘을 $\vec{F}_M$이라 할 때,

$$\vec{F}_M = \sum_{\text{all module i}} C_{Mi}\,\vec{S}_{Mi}$$

이다. 이 모듈이 놓여질 가장 안정된 위치는 이 모듈에 작용하는 각 힘들이 평형을 이루는 위치, 즉 힘의 벡터 합이 0이 되는 위치이다. 이때가 또한 이 모듈에 연결된 신호선 길이의 합이 최소가 되는 위치이다. 이 위치를 평형 위치 $\vec{T}$라고할 때 모듈 M의 평형위치 $\vec{T}_M$은

$$\vec{T}_M = \frac{\vec{F}_M}{\sum_{\text{all module i}} C_{Mi}} \text{ 가 된다.}$$

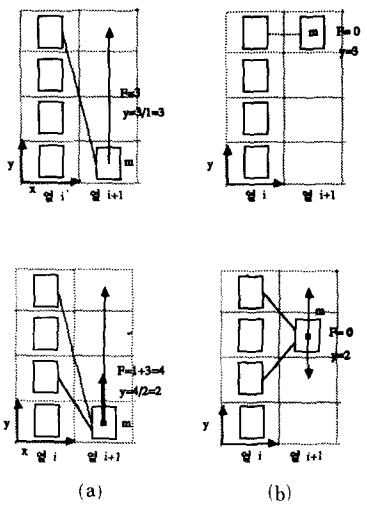
이러한 원리를 y축만의 일차원 모듈 배치 문제인 수직 배치에 적용하면, 모듈 M의 수직 방향에서의 힘의 평형위치 $\vec{Y}_M$은

$$\vec{Y}_M = \frac{\sum_{\text{all module i}} C_{Mi}\,\vec{Y}_i}{\sum_{\text{all module i}} C_{Mi}}$$

가 된다. 이 위치에서 모듈 M에 연결된 신호선 길이의 합이 최소가 된다.

만일 배치될 심볼의 높이가 한 그리드 격자의 높이보다 큰 경우에는 모든 핀들이 심볼의 중심에 있다고 가정하여 힘의 평행위치를 계산하고 그 평형위치를 중심으로 심볼의 크기에 따라 상하에 걸쳐 배치한다.

수직 배치의 경우 열의 좌표가 1인 열은 입력 노드들로서 각 입력 노드들의 상대적인 행의 좌표는 고정되어 있다고 가정하고 열 2부터 수행한다. 열i의 수직 배치의 경우 전열 i-1과의 연결관계가 있는 모듈들이 배치되어 있으므로 이미 배치가 끝난 전열 i-1과 열i의 각 모듈들간의 연결 관계만을 고려한다. 그림7은 열i의 수직 배치가 끝난 후 열i를 고정시키고 열 i+1에 배치될 모듈들의 평형 위치를 찾아 수직 배치를 수행하는 과정을 보여준다. 그림 7(a)에서 모듈 m에 작용하는 힘은 3이고 평형위치는 힘의 합을 연결된 모듈의 수로 나눈 3이 된다. 그림7(b)에서 모듈 m은 2개의 모듈에 연결되어 있고 힘의 합은 $|\vec{F}|=|1+3|=4$가 된다. 따라서 평형위치는 y=4/2



(a) (b)

그림 7. 수직 배치
(a) 연결도가 1인 경우
(b) 연결도가 2인 경우

Fig. 7. Vertical placement.
(a) case of connectivity 1,
(b) case of connectivity 2.

=2가 된다.

그러나 다수 모듈의 배치를 동시에 고려할 경우 서로간 평행위치가 중첩이 되는 경우가 발생하게 되므로 이러한 문제의 해결을 위하여 휴리스틱 알고리듬이 요구된다.

본 수직 배치 알고리듬에서는 배치된 각 모듈들에 대하여 아래와 같은 우선순위(priority)를 정의하고 둘이상의 모듈들의 평형 위치가 한 격자에 중첩되는 경우 우선순위가 가장 높은 모듈을 그 격자에 배치하고 나머지 모듈들은 빈 슬롯(slot) 또는 자기보다 우선순위가 낮은 모듈이 배치된 위치 중에서 힘의 합이 최소가 되는 위치를 찾아 배치한다.

본 논문에서 제안한 우선순위 함수의 정의는 다음과 같다.

[정 의] 수직 배치에서의 우선 순위 함수 정의

○집합S : 배치해야 할 전체 모듈의 집합

○집합S_i : $S_i \subset S$, 열 i에 배치해야할 모듈의 집합

○집합 G_m : $G_m \subset S_{i-1}$ and $m \in S_i$, 열 i-1에 속한 모듈들 중에서 열 i의 한 모듈 m과 연결관계가 있는 모듈들의 집합

○함수Y(m) : $m \in S$, 배치된 모듈 m의 행의 좌표

○함수BMAX(m) : $m \in S_i$, $BMAX(m) = \max_{i \in Gm} Y(i)$

○함수 BMIN(m) : $m \in S_i$, $BMIN(m) = \min_{i \in Gm} Y(i)$

위의 정의에 따라 모듈 m의 위치 y에 대한 우선순위 P(m, y)는,

$$P(m,y) = \begin{cases} \dfrac{1}{|BMAX(m) - BMIN(m)| + 1}, & \text{if } y > BMIN(m) \text{ and } y < BMAX(m) \\ 1, & \text{if } y = BMIN(m) \text{ or } y = BMAX(m) \\ 0, & \text{if } y < BMIN(m) \text{ or } y > BMAX(m) \end{cases}$$

이러한 우선순위는 0과 1사이의 값으로써 모듈m의 배치를 BMAX(m)과 BMIN(m)사이 구간으로 제한할 경우 그 영역내의 각 슬롯에서는 우선순위값의 역수만큼의 여유도를 갖게된다. 즉 여러 모듈간에 한 슬롯에 대한 배치요구가 발생할 경우 여유도가 가장 적은 모듈을 우선적으로 배치하여 가능한 구간내에 모두 배치할 수 있게하여 배선길이를 줄이는 것은 물론 이해하기 쉽도록 심볼을 배치한다. 경계선상인 BMAX(m)과 BMIN(m)에서 우선순위가 1이 되는 것은 모듈m의 배치가 BMAX(m)과 BMIN(m)사이 영역에서 다른 모듈들의 우선순위에 밀려 배치되지 못한경우 경계선상에서 최우선적으로 배치하여 영역 밖에 배치되는 경우를 방지한다.

이러한 우선순위를 고려한 수직배치 알고리듬은 다음과 같다.

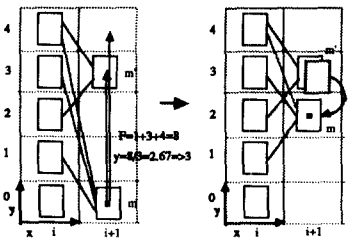
[단계 1] 열 i=2에서 마지막 열까지 단계2에서 단계 4까지를 수행

[단계 2] 열 i에 속하는 모듈 m, $m \in S_i$에 대하여 $Q(m) = \sum_{j \in Gm} C_{mj}$ 값을 계산한다.

[단계 3] 미 배치된 모듈, $m \in S_i$ 중에서 Q(m) 값이 가장 큰 (연결도가 많은) 모듈을 고른다. 배치할 모듈이 없으면 수행을 종료한다.

[단계 4] 선택된 모듈의 장력이 0인 평행 위치를 계산하여, 그 자리가 비었으면 배치하고 단계 3으로 간다. 만일 그 자리에 이미 다른 모듈이 배치되어 있는 경우에는, 이미 배치되어 있는 모듈과 현재 모듈간의 우선순위를 비교하여 현 모듈의 우선순위가 낮으면 주변의 위치중에서 비어 있거나 우선순위가 낮은 모듈이 있는 위치중에서 힘 F의 값이 가장 작은 위치를 선택하여 배치하고 단계 3으로 간다. 다른 모듈이 존재하는 위치를 빼앗아 배치한 경우 자리를 빼앗긴 모듈을 새로이 선택하여 선택되는 모듈이 있을 때까지 단계 4를 계속 수행한다.

그림8은 평형위치가 중복되는 두 모듈간에 우선순위에 따라 배치가 결정되는 예를 보여주고 있다. 열 i+1에 배치될 모듈 m은 열 i에 배치된 3개의 모듈과 연결되어 있다. 이때 모듈 m에 가해지는 힘의 합



(a) (b)

그림 8. 우선 순위에 따른 수직 배치
(a) 배치전 (b) 배치 후

Fig. 8. Vertical placement by priority.
(a) Before placement,
(b) After placement.

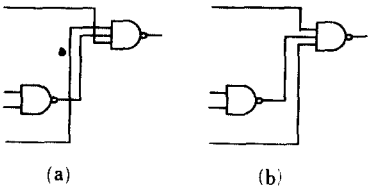
은 |F|=|1+3+4|=8이며 평행위치는 y=8/3= 2. 67=>3이 된다. 그러나 y=3의 위치에는 이미 다른 모듈 m*가 배치되어 있으므로 두 모듈 m, m*의 위치 y=3에 대한 우선순위를 비교한다. 모듈 m의 우선순위는 앞의 정의에 의하여 $P_m=1/(|4-1|+1)=0.25$이고, 모듈 m*의 우선순위는 $P_m=1/(|4-2|+1)=0.33$이므로 우선순위가 높은 모듈 m*가 그 자리에 배치되고 모듈 m은 그 주변의 다른 위치중에서 힘의 합 F의 크기가 가장 적은 위치를 찾는다. y=4에서의 힘의 합 |F|=|0+(-1)+(-3)|=4이고, y=2에서는 |F|=|1+2+(-1)|=2가 된다. 따라서 힘의 크기가 작은 y=2의 위치가 선택되고 그 위치는 비어 있으므로 모듈 m은 y=2에 배치된다.

3. 핀 할당 및 배치개선

배치가 끝난 후 각 심볼에 대하여 핀 할당 및 배치개선을 수행한다.

먼저 핀 할당 단계는 한 심볼의 여러 핀들 중에서 전기적으로 등가(논리적, 지연시간 등에서)인 핀들간의 네트(net)를 서로 교환하여 신호선들간의 교차수를 가능한 최소로하여 신호선 배선을 간결하게 하고자 하는 단계로서 그림 9(a)와 같이 연결된 신호선들간의 등가핀을 교환하여 그림 9(b)와 같은 결과를 얻는다.

핀 할당 방법은 각 핀들에 연결된 신호선에 따라 상측에 연결된 네트는 상측 핀을 할당하고 하측으로 연결된 네트는 가능한한 하측 핀을 할당한다. 즉 초기 배치가 끝난후 각 핀들에 대하여 그림10의 알고리듬과 같은 가중치(weight)를 할당한다. 가중치는 현재의 심볼보다 윗쪽에 연결 요구가 있으면 양의 값을, 아랫쪽에 연결요구가 있으면 음의 값을, 상하



(a) (b)

그림 9. 핀 할당
(a) 핀 할당 전 (b) 핀 할당 후

Fig. 9. Pin assignment.
(a) Before pin assignment,
(b) After pin assignment.

```
procedure assign_pin( )
{
  각 심볼의 모든 핀의 weight를 0으로 초기화;
  for(모든 심볼 Si에 대하여){
    Ysi=심볼 Si의 행의 위치좌표;
    for(심볼 Si의 각 핀 Pj에 대하여){
      /* wj는 핀 pj의 weight */
      for(핀pj와 신호선으로 연결된 모든 심볼 Sk에
      대하여){
        Ysk=심볼 Sk의 행의 위치좌표;
        if((Ysk-Ysi)*wj<0){
          wj=0; /* 상하측 모두에 연결 요구가 있는 핀*/
          break;
          }
          else if(|wj|<|Ysk-Ysi|)
            wj=Ysk-Ysi; /* 상측 또는 하측에만
            연결요구가 있는 핀 */
      }
    }
  }
  모든 심볼의 등가핀들 간에 할당된 weight에 따라서
  sorting;
}
```

그림10. 핀 할당 알고리듬

Fig. 10. Pin assignment algorithm.

모두 연결되었으면 0의 값을 할당한다. 가중치의 할당이 끝난후 각 등가핀들간에 가중치의 크기에 따라 상에서 하로 정렬(sorting)하면 연결도에 따라 상측에 연결된 핀의 순서대로 배열이 된다.

그림11(a)는 그림 9(a)의 회로상의 핀에 가중치를 할당한 예이고, 그림11(b)는 등가핀들간에 가중치의 크기에 따라 정렬하여 핀할당을 행하는 과정을 보여주고 있다. 현재 핀할당의 대상이 되는 심볼 m의 행좌표는 y=7이고 이것과 연결관계가 있는 열 i에 배치된 심볼들의 행좌표는 각각 10, 5, 4이다. 그림10의

핀할당 알고리듬에 의하여 심볼 m의 최상측핀의 가중치 $w_1=4-7=-3$이고, 나머지 핀들에 대하여도 같은 방식으로 각각 −2, 2가 할당된다. 세개의 핀은 논리적으로 등가이므로 각 핀의 가중치에 따라 정렬하면 그림11(b)와 같이 핀할당이 되어 신호선간의 교차수가 3에서 0으로 감소되었다.

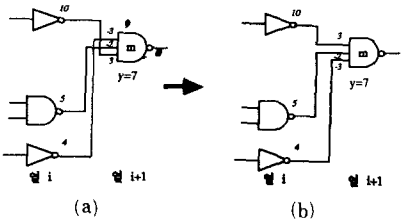


그림11. 핀의 가중치 할당의 예
(a) 핀 할당 전 (b) 핀 할당 후

Fig. 11. An example of pin weight assignment.
(a) Before pin assignment,
(b) After pin assignment.

논리적 배치와 핀 할당이 끝난 후, 각 심볼들은 그림12(a)와 같이 그리드상에 배치된다. 이 경우 인접한 두 열간에 연결하여야 할 신호선이 있는 경우 그림12(a)와 같이 연결하여 주면 신호선에 2개의 꺾임이 생기게 되므로 스키매틱 다이어그램이 복잡하게 된다. 이러한 불필요한 꺾어짐(Jog)을 없애기 위하여 각 열들 간에 연결 요구가 있는 심볼들의 위치를 조금씩 이동시켜 그림12(b)와 같이 연결된 두 핀간의 위치를 직선이 되도록 조정한다. 이러한 배치개선 단계는 각 열별로 순서적으로 수행한다.

그림12. 배치 개선
(a) 배치 개선 전
(b) 배치 개선 후

Fig. 12. Improvement of placement.
(a) Before Improvement,
(b) After improvement.

4. 전체배선

전체 배선 단계에서는 열 좌표의 차가 2이상인 심볼들 간의 배선을 위하여 피이드스루를 할당하는 단계이다. 피이드스루란 심볼들이 배치된 열 영역 사이를 신호선이 지나가는 위치로서 실제의 배선은 최종 단계인 상세 배선 단계에서 채널 배선기를 이용하여 이루어지며, 전체 배선 단계에서는 단지 각 신호선에게 피이드스루 만을 할당한다. IC 레이아웃의 경우 전체 배선(global routing)의 목적 함수는 전체 배선길이의 최소 또는 배선 밀도의 균등화이지만, 스키매틱 생성시에는 이와는 달리 신호선들의 흐름이 일관성있고 명확하도록 배선하여야 한다. 따라서 다음과 같은 원칙하에 전체 배선을 수행한다.

○각 열 영역에서의 피이드스루의 할당은 한 신호선에 대하여 오직 하나의 피이드스루만을 할당한다.

○꺾어짐이 가능한 최소인 직선이 되도록 피이드스루를 할당한다.

○각 피이드스루는 가능한한 연결될 핀들의 최소 경계사각형(minimum bounding box) 내에 존재하도록 할당한다. (버스 신호선은 제외)

○버스(bus) 신호선의 경우는 연결된 심볼들의 상대적인 위치를 고려하여 스키매틱의 상단 또는 하단에 모아서 직선이 되도록 할당한다.

IC 레이아웃의 경우 그림13(a)와 같이 배선하는 것이 경우에 따라서 총 배선길이의 최소화와 신호선 밀도를 줄일 수 있으므로 좋은 결과를 얻을 수도 있으나 스키매틱 생성기의 경우는 그림13(b)와 같이 배선하는 것이 총 배선길이는 길어질 수 있으나 신호선 흐름 파악에 있어서 좀 더 간결하므로 이 경우를 택한다. 또한 그림14(a)와 그림14(b)의 두 경우 총 배선길이는 동일하지만 그림14(b)의 경우가 꺾임수가 적어 이해하기 쉬우므로 가능한한 신호선이 직선이 되도록 피이드스루를 할당한다. 입력 단계에서 버스 신호선임을 사용자가 지정하거나 또는 다음과 같은 조건을 만족하는 신호선은 버스로 인식한다.

버스 신호선은,

○한 신호선 i에 연결된 심볼의 수 $N_i > C_{th}$ (C_{th}는

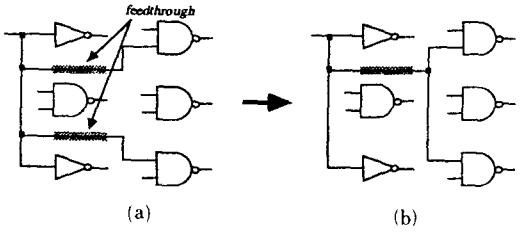


그림13. 피이드스루 할당 I
(a) 좋지 않은 경우 (b) 좋은 경우

Fig. 13. Feedthrough assignment I.
(a) not good case, (b) good case.

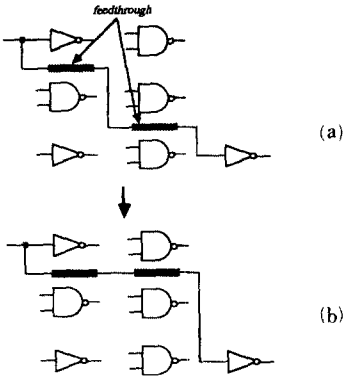


그림14. 피이드스루 할당 II
(a) 좋지 않은 경우 (b) 좋은 경우

Fig. 14. Feedthrough assignment II.
(a) not good case, (b) good case.

상수)이고,

○한 신호선 i에 연결된 심볼들의 최소경계사각형(minimum bounding box)의 열의 수 X_i가 $X_i \geq W_{th}$ ($W_{th} \geq 3$, W_{th}는 상수)인 신호선이다.

즉 한 신호선에 연결된 심볼의 수가 C_{th} 보다 크고, 그 심볼들이 W_{th} 열 이상에 분포되어 있는 경우에 일반적인 신호선과 같이 그리드상의 수평 채널에 피이드스루를 통하여 배선할 경우 스키매틱이 복잡해 지고 배선경로에 꺾임이 많아져 회로의 이해가 어려워진다. 따라서 이러한 경우 사용자가 원하면 그림15와 같이 사용자의 이해가 쉽도록 버스 신호선들을 스키매틱의 상단 또는 하단에 모아서 배선한다. 이 경우 버스 신호선에 연결된 전체 심볼의 상대적인 위치를 보아 스키매틱의 상단 또는 하단에 할당하고 이의 실제 배선은 상세 배선 단계에서 수행 한다.

버스 신호선 이외의 신호선의 전체 배선 단계에서의 피이드스루 할당을 위하여 본 논문에서는 선분 탐색(line-search) 알고리듬[12]을 이용하였다. 선분 탐색 알고리듬은 수행 속도가 빠르고 또한 배선경로가 가능한한 직선을 유지하여 꺾임이 적으므로 본 목적에 적합하다. 그림16은 선분 탐색 알고리듬을 이용하여 피이드스루를 할당하는 과정을 보여준다.

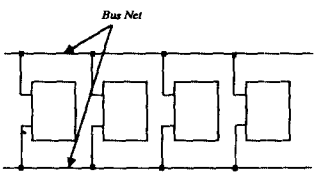


그림15. 버스신호선의 할당

Fig. 15. Assignment of bus signals.

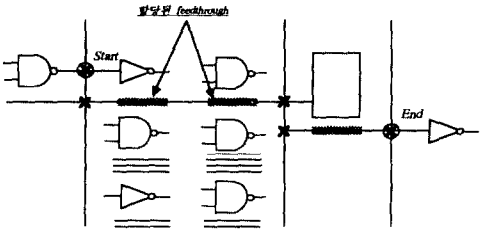


그림16. 선분 탐색 알고리듬을 이용한 전체 배선

Fig. 16. Global routing using line search algorithm.

한 신호선에 대하여 연결관계가 있는 심볼들의 경계사각형(boundind box)을 구하고 이 경계 사각형의 최소 열에서 시작하여 최대 열까지 선분 탐색 알고리듬을 이용하여 가능한 꺾임이 적은 하나의 경로를 구하여 그 경로가 통과하는 위치에 피이드스루를 할당하고 이 피이드스루간의 연결은 상세 배선 단계에서 수행한다.

5. 상세배선

상세 배선 단계에서는 인접한 두 열 사이에 채널을 정의하고, 각 열 간의 연결정보와 전체 배선의 수행 결과로 생긴 피이드스루 정보를 이용하여 채널 배선[10]을 수행한다. 본 스키매틱 생성기에서 사용된 채널 배선기는 원칙적으로 꺾음(dog-leg)을 허용하지 않으나 그림17(a)와 같이 신호선간에 사이클이 발생한 경우에만 꺾음을 사용하여 그림17(b)와 같이 배선한다.

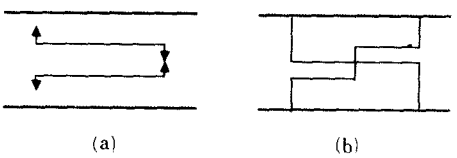


그림17. 채널 배선에서의 dog-leg
(a) 신호선 연결요구에 사이클이 발생한 경우
(b) dog-leg를 이용한 배선

Fig. 17. Channel routing using dog-leg.
(a) Cycle on vertical track assignment,
(b) Wiring using dog-leg.

또한 전체 배선단계에서 스키매틱의 상(하)단에 수평으로 할당된 버스 신호선들은 각 열들간의 수직 채널에서 채널 배선이 수행되어 스키매틱의 상(하)단의 버스 신호선으로 연결되는 수직 트랙이 결정된 후 이 정보를 이용하여 스키매틱의 상(하)단에서 채널 배선기를 이용하여 수평 트랙을 할당하여 배선한다. 그림18에서 심볼들이 배치된 영역들간의 모든 수직채널에서 채널배선기를 사용하여 신호선들을 배선하고, 그 결과 상측의 버스 신호선과 연결요구가 있는 신호선을 상측의 버스 신호선 채널에 할당한다. 최종적으로 채널 배선기를 이용하여 버스 신호선 채널을 배선한다.

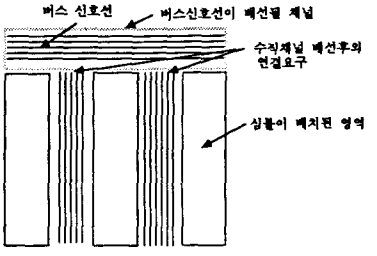


그림18. 채널 배선기를 이용한 버스 신호선의 배선
Fig. 18. Bus signal routing with channel router.

Ⅳ. 구현 및 실험

본 스키매틱 생성기는 Sun 워크스테이션 상에서 C 언어로 구현되었다. 실제 그래픽 도형을 화면상에 생성하기 위하여 Sunview와 Pixrect 그래픽 라이브러리를 이용하였다. 생성 후에는 화면상에서 윈도우 및 수정 기능등이 있다.

스키매틱 생성기는 ETRI의 논리및 제어디자이너의 합성 결과를 화면상에 스키매틱으로 그려주며, 생성된 스키매틱은 EDAS(electronic design automation system)의 공학용 디자이너[14]의 입력 화일로 변환되어 회로의 수정 및 검증이 가능하다.

그림19, 20, 21, 22는 논리 합성의 결과를 스키매틱 생성기를 이용하여 생성한 예이다.

그림19는 피이드백 루프가 있는 순서회로 합성의 예로서 입력 inst로 들어오는 입력패턴 (1|0)*100 을 인식하여 출력 accept를 1로 만들어 주는 FSM(finite state machine)을 레지스터 전송 수준의 회로로 합성하여 본 스키매틱 생성기로 출력하였다.

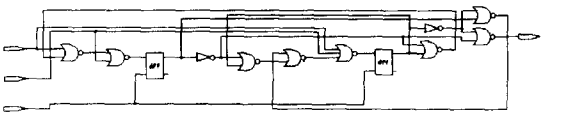


그림19. 입력(1|0)*100을 인식하는 순서회로 생성 예
Fig. 19. An synthesis example of a circuit reconizing input pattern (1|0)* 100.

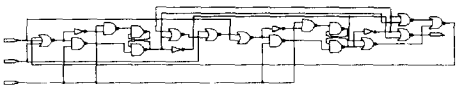


그림20. (그림19) 회로를 게이트 수준으로 flatten 하여 생성한 예
Fig. 20. An synthesis example of the flatten gate level circuit of fig 19.

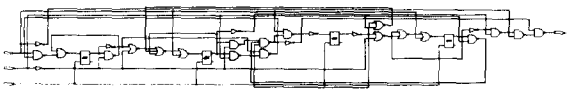


그림21. MCNC89 FSM 벤치마크 lion 회로 생성 예
Fig. 21. An synthesis example of lion, a MCNC89 FSM bench mark circuit.

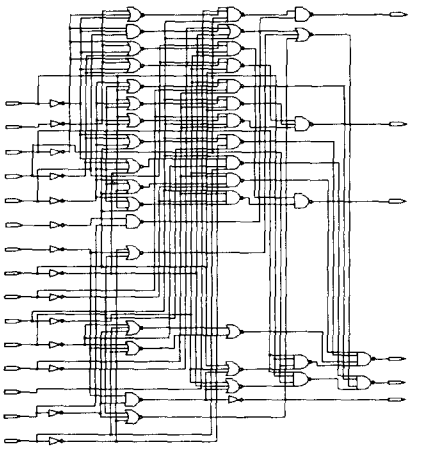


그림22. 조합회로 생성 예
Fig. 22. An synthesis example of a combinational circuit.

그림20은 그림19의 회로에서 D 플립플롭을 게이트 수준의 회로로 풀어서 스키매틱으로 생성한 예이다. D 플립플롭 회로에는 두개의 게이트가 래치를

형성하므로 전처리 단계에서 이를 2노드 그룹으로 인식하여 심볼 생성기를 이용 래치 심볼을 생성하고 이를 사용하여 스키매틱을 생성하였다.

그림21은 MCNC89 FSM 벤치마크(benchmark) 중에서 lion 회로를 논리합성하여 스키매틱으로 생성한 예이며, 그림22는 피이드백 루프가 없는 조합회로를 합성한 예이다.

본 스키매틱 생성기는 원래 대화식 논리합성기의 구현을 목표로 개발되었으나 아직까지는 단순히 합성의 중간결과 또는 최종 결과를 스키매틱으로 보여주는 기능만이 구현 되었다. 현재에는 논리 및 제어 디자이너와 함께 실제의 설계에 이용하여 그 성능을 평가중이며 설계자의 의견을 통해 기능을 보강하고 있다.

V. 결 론

본 스키매틱 생성기는 논리 및 제어디자이너[13]의 일부로서, 논리합성의 결과로 생성된 네트리스트로부터 사용자가 이해하기 쉬운 스키매틱 다이아그램으로 자동생성하여 준다. 스키매틱 생성 문제는 IC 레이아웃 생성문제와 많은 부분에서 유사하지만 목적함수에 있어서 면적 또는 동작시간 등의 최소화에 있는 것이 아니라 인간에게 좀 더 이해가 쉽고 간결한 스키매틱을 생성하는데 있다.

따라서 본 논문에서는 스키매틱 생성 과정을 전처리, 논리적 배치, 핀할당 및 배치 개선, 전체 배선, 상세 배선 등의 5단계로 나누어 각 단계에서의 알고리듬을 설명하였다.

스키매틱의 생성은 미적인 요소가 매우 중요하므로 미적인 요소의 증대를 위한 좀 더 많은 연구가 이루어져야 한다. 그 예로서 본 논문에서 고려하지 못한 상세 배선단계에서 신호선간의 교차수를 최소화 하도록 상세배선을 수행하는 알고리듬의 개발 등이 필요하다. 또한 단순한 스키매틱의 생성단계에서 발전하여 논리합성 또는 상위수준 합성기와의 밀접한 접속을 통하여 설계자의 지식을 합성과정에 반영하도록 하는 대화식 합성기 개발에 관한 연구도 이루어져야 한다. 이를 위해서는 스키매틱 생성의 속도도 빨라져야 하며 부분적 설계 변경을 점진적으로 반영하는 방안도 개발되어야 한다.

參 考 文 獻

[1] R. Brayton, R. Rudell, A. Sangiovanni-Vincentelli, and A. Wang, "MIS: Multiple-level interactive logic optimization system," *IEEE Trans. on CAD, vol.* Cad-6, no. 6, pp. 1062-1081, Nov. 1987.

[2] B.J. Brennan, "An Algorithm for Automatic Land Routing on Schematic Drawings," *IEEE Design and Test,* Feb. 1986.

[3] Ashok Kumar, et al., "Automatic Generation of Digital Systems," *IEEE Design and Test,* Feb. 1986.

[4] M.L. Ahlstrom, et al., "Hal: A Heuristic Approach to Schematic Generation," *Proc. of ICCAD 84,* pp. 436-442, 1984.

[5] Mira A. Majewski, et al., "AUTODRAFT: Automatic Synthesis of Circuit Schematics," *Proc. of ICCAD 86, pp.* 435-438, 1986.

[6] A. Arya, et al., "Automatic Generation of Digital System Schematic Diagrams," *22th Design Automation Conf. Proc.,* pp. 388 395, June 1985.

[7] Robert K. Chun, et al., "Vision: VHDL Induced Schematic Imaging On Netlists," *24th Design Automation Conf. Proc.* pp. 436-442, June 1987.

[8] M. May, et al., "Placement and Routing for Logic Schematics," *Computer Aided Design,* vol. 15, no. 3, pp. 89-101, May 1983.

[9] G. M. Swinkels and Lou Hafer, "Schematic Generation with an Expert System," *IEEE Trans. on CAD,* vol. 9, no. 12, pp. 1289-1306, Dec. 1990.

[10] T. Yoshimura and E.S. Kuh, "Efficient Algorithms for Channel Routing," *IEEE Trans. on CAD,* vol. CAD-1, pp 25-35, Jan. 1982.

[11] M.A. Breuer, *Design Automation of Digital Systems,* Prentice-Hall, vol. 1, pp. 254-257, 1972.

[12] D.W. Hightower, "A Solution to Line-Routing Problem on the Continuous Plane," *Proc. 6th Design Automation Workshop,* pp. 1-24, 1974.

[13] 한국전자통신연구소, "자동설계 환경구축에 관한 연구(Ⅱ) 최종연구보고서,"과학기술처, pp. 199-215, 1990. 7.

[14] 한국전자통신연구소, "전자자동설계시스템(EDAS) 운용기술서 제1권 : 공학용 디자이너," 과학기술처, 1988. 11.

[15] 배영환, 백영석, 박성범, 이성봉, 이현찬, "Sche Gen : 스키매틱 자동생성기," '91 씨에이디 전자계산 반도체 재료 및 부품 합동학술발표회 논문집, 1991. 5.

著 者 紹 介

裵 英 煥(正會員)

1962年 10月 29日生. 1985年 2月 한양대학교 전자공학과 졸업. 1987年 2月 한양대학교 대학원 전자공학과 졸업(공학석사). 1987年 2月～현재 한국전자통신연구소 자동설계연구실 근무. 주관심분야는 VLSI CAD, 컴퓨터 그래픽스 및 컴퓨터 아키텍춰 등임.

●

白 瑛 錫(正會員)

1963年 1月 14日生. 1985年 2月 한양대학교 전자공학과 졸업. 1987年 2月 한양대학교 대학원 전자공학과 졸업(공학석사). 1989年 8月～현재 한국전자통신연구소 자동설계연구실 근무. 주관심분야는 VLSI CAD, 그래픽, 컴파일러 등임.

●

朴 星 範(正會員)

1962年 3月 1日生. 1984月 2月 한양대학교 전자공학과 졸업. 1986年 2月 한양대학교 대학원 전자공학과 졸업(공학석사). 1986年 2月～현재 한국전자통신연구소 자동설계연구실 근무. 주관심분야는 VLSI CAD, 컴파일러 및 데이타 베이스 등임.

李 省 奉(正會員)

1962年 2月 1日生. 1984年 2月 한양대학교 전자공학과 졸업. 1986年 2月 한양대학교 대학원 전자공학과 졸업(공학석사). 1991年 2月 한양대학교 대학원 전자공학과 졸업(공학박사). 1991年 2月～현재 한국전자통신연구소 자동설계연구실 선임연구원. 주관심분야는 VLSI CAD 등임.

●

張 泳 祚(正會員)

1979年 2月 경북대학교 전자공학과 졸업. 1982年 2月 경북대학교 대학원 전자공학과 졸업(공학석사). 1981年 6月～현재 한국전자통신연구소 자동설계연구실 선임연구원.

●

李 玄 燦(正會員)

1978年 2月 서울대학교 산업공학과 졸업. 1980年 2月 한국과학기술원 산업공학과 졸업 (공학석사). 1988年 4月 The University of Michigan, Industrial and Operations Engineering 졸업(공학박사). 1980年 2月～1983年 6月 부산 금속파이프(주) 근무. 1988年 1月～1988年 8月 The University of Michigan, Lecturer. 1988年 9月～1991年 8月 한국전자통신연구소 자동설계연구실 실장. 1991年 9月 현재 홍익대학교 산업공학과 교수.