

게이트 어레이 레이아웃 형태에서의 기능 모듈 편집기의 구현

홍 성 현[†] · 정 영 숙^{††} · 임 종 석^{†††} · 손 진 우^{††}

요 약

본 논문에서는 게이트 어레이 레이아웃 형태에서 디지털 회로를 보다 발전적인 방식으로 설계하도록 하는 기능 모듈 구현 전용 편집기를 제안한다. 구현된 편집기에서의 기능 모듈 설계 작업은 전반적으로 설계자와의 대화 형식으로 이루어진다. 따라서, 설계자는 빠른 시간 내에 모듈을 설계할 수 있고 보다 만족스런 레이아웃 결과를 얻을 수 있다. 특히, 본 편집기에서는 IC원판을 구성하는 기초셀 형태에 대해 독립적이며, 배치 및 배선 작업 시에 수동과 반자동 방식을 함께 사용할 수 있다는 특징이 있다. 또한 기존의 툴에서는 제공되지 않는 다양한 기능들이 추가되어 있어 더욱 효율적으로 모듈 생성을 할 수 있다. 본 편집기는 X-윈도우 Motif 환경 하에서 C언어로 구현되었으며, 그 기능을 기존의 레이아웃 시스템인 Scadali를 사용했을 때와 비교하여 성능을 평가하였다. 구현한 편집기를 이용하면 설계 시간을 두 배정도 단축시킬 수 있다.

An Implementation of Functional Module Editor in the Gate-Array Layout Style

Sung Hyun Hong[†] · Young Sook Jeong^{††} · Jong Seok Rim^{†††} · Jin Woo Sohn^{††}

ABSTRACT

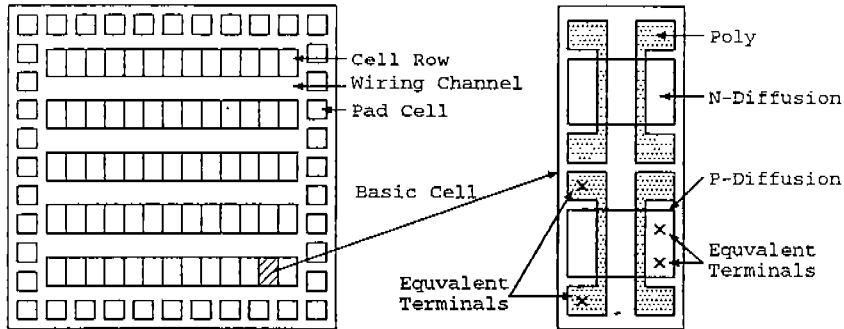
In this paper we propose a layout editor for the functional module generation in the Sea-of-Gates(SOG) layout style. The proposed layout editor provides interactive ways of designing a functional module to the designer so that the layout result is very satisfiable. Especially, the editor is independent on the shape of the basic cells in the gate array template, and provides semi-automatic layout methods as well as hand layout. It also has several special functions which are not able to find in other layout tools for the module generation, and hence the designer can generate modules very fast. The layout editor is implemented in C language with X-window Motif environment. When we compare our editor with the previous layout editor Scadali, the design time is reduced by a factor of two for several benchmark circuits.

1. 서 론

반도체 기술이 발전함에 따라 그 환경도 점점 주문

화되어 왔다. 특히, 게이트 어레이(Gate Array)방식이 소개되면서 고밀 도의 동작 속도가 빠른 회로를싼 가격에 빨리 제작할 수 있게 되어 ASIC시장이 점차 활기를 띠어 왔다[2, 3]. 기초셀을 포함하는 게이트 어레이 구조의 전형적인 예는 (그림 1)과 같은데 이 방식은 칩 내에 트랜지스터 형태를 미리 생성하는 공정을 거친 원판과 기초셀 모듈 라이브러리를 가지고 배치

† 정 회 원: 삼성전자 정보통신 시스템 본부, DSP 파트
†† 정 회 원: 한국전자통신연구소 교환기술연구단,
H/W 개발환경 연구실
††† 총신회원: 서강대학교 전자계산학과
논문접수: 1996년 3월 21일, 심사완료: 1996년 5월 15일



(그림 1) 기초셀을 포함하는 게이트 어레이 구조의 전형적인 예.

(Fig. 1) A typical example of architecture of Gate Array with Basic Cell.

와 배선 작업을 거쳐 디지털 회로를 구현하는 방식으로 설계 시간과 가공 시간이 짧은 것 등의 장점이 있다.

디지털 회로의 구현을 위해서 설계자는 기능 모듈 라이브러리를 이용하여 회로를 설계하고 사용된 모듈들을 원판에 배치한 후 그들 간을 배선하는 과정을 거쳐야 한다. 이에 대해 모듈들 사이의 배치와 배선에 대해서는 이미 많은 연구가 되어 왔지만[6, 9, 10], 셀 라이브러리 구축을 위한 기능 모듈 생성에 대해서는 아직 그 연구가 미미하다[1, 4, 7]. 뿐만 아니라, 기존에 연구되어 있는 기능 모듈 생성 방식으로는 설계 상 어려운 점이 많으며 전체 회로 구현에 많은 시간이 필요하고, 복잡한 회로의 경우 설계 면적이 매우 커질 우려가 있다. 또한, 기능 모듈 생성기는 기초셀 형태에 독립적이어야 하며, 논리 회로 구현시 배치와 배선 결과에 대해 설계자가 임의로 수정할 수 있어야 한다. 본 논문에서는 이러한 조건들을 만족시켜 효율적인 모듈 생성 결과를 얻을 수 있는 기능 모듈 생성을 위한 전용 편집기를 제안한다.

제안한 전용 편집기에는 게이트 어레이 레이아웃 형태에서 보다 효과적인 모듈 설계를 위한 새로운 기능이 포함되어 있다. 즉, 편집기 상에서 모듈을 설계할 때 배치 및 배선을 수동으로 할 수도 있어야 하지만, 모든 설계를 일일이 수작업 하기보다는 어느 정도 자동으로 처리하도록 하여 훨씬 더 적은 노력으로 짧은 시간 내에 모듈을 설계할 수 있도록 한다. 또한, 임의의 기초셀 형태에 대해서도 설계자가 이를 한 번

만 정의하여 주면 자동으로 이를 해석하여 설계자로 하여금 이를 기초로 한 기능 모듈을 효율적으로 편집할 수 있게 하여준다.

본 논문은 서론에 이어 제 2장에서 기존의 기능 모듈 생성 방법에 대하여 기술하고, 제 3장에 본론을 전개하는데 1절에서는 구현한 편집기의 전체 구성 및 기능에 관하여 기술하고, 2절에서는 여러 형태의 기초셀과 그로부터 게이트 어레이 원판을 구성하기까지에 대하여 설명한다. 이어 3절에서는 배치 및 배선에 대하여 논하고, 4절에서는 제공되는 환경 내에서 전개되는 설계의 전체 흐름에 관하여 설명한다. 제 4장에서는 편집기의 성능에 대한 결과를 보이고 그 응용 범위에 관해 기술한다. 그리고, 제 5장에 결론을 맺는다.

2. 기능 모듈 생성기

기초셀 모듈 라이브러리 구축을 위해 기능 모듈 생성기는 반드시 필요한 요소이다. 지금까지 기능 모듈 생성기로써의 기능을 해 온 생성 방법으로는 full-custom 방식에 의한 완전 수동 기법과 회로 구현에 필요한 모든 작업이 자동으로 처리되는 자동 방식에 의한 모듈 생성 기법으로 구분해 볼 수 있다[2].

완전 수동에 의하여 기능 모듈을 설계하기 위해서는 기존의 여러 상용 레이아웃 편집기를 이용할 수 있다. 그러나 이들을 이용하여 기능 모듈을 생성하고

자 할 때는 여러 가지 문제점이 있다. 생성하고자 하는 대상인 게이트 어레이의 기초셀 형태를 일일이 화면에 그려 주어야 하며, 주어진 논리 회로로부터 트랜지스터를 추출하여 기초셀 어레이 위로 할당하기 위해 몇 번의 시행 착오를 거쳐야 한다. 또한, 배선 과정에서 그 수정 횟수가 더욱 빈번해지므로 모듈 생성이 대단히 어려울 뿐 아니라 많은 시간이 소요된다. 또한, 레이아웃을 심볼릭 형태로 사용하지 않고 마스크 레이아웃을 그대로 사용하므로 설계시 많은 불편이 따른다.

이에 대해 기능 모듈들을 적은 노력과 시간으로 구현이 가능한 자동 방식이 필수적으로 발표되었는데, 이 방법에서는 배치와 배선의 각 단계에 맞는 알고리즘을 구현한 후[1, 4, 7], 그들을 통해 디지털 회로의 구현을 자동화하는 것이다. 이러한 자동 생성기는 보다 신속한 방법에 의한 기능 모듈 라이브러리 구축에 활용될 수 있다. 하지만 게이트 어레이 레이아웃의 기술에 따라 원판의 기초셀 모양 및 크기가 다를 수 있는데 이러한 모든 경우를 적용시키지 못한다는 점과 배치 및 배선의 자동 처리 결과에 대해 설계자가 더 나은 결과를 쉽게 제시할 수 있는 경우에도 그 의도대로 수정하기가 매우 힘들다는 등의 단점이 있다. 특히, 자동 배선에 실패할 경우 이의 배선을 완결하기가 대단히 어렵다.

가장 이상적인 방법의 하나로 제안될 수 있는 것은 기능 모듈을 생성하고 손쉽게 편집할 수 있는 전용 편집기의 개발이다. 배치 및 배선이 효율적으로 신속하게 이루어지는 자동에 의한 생성 방법을 우선 적용하여 편집기 상에서 결과를 얻은 후, 수동으로 배치나 배선 결과를 수정할 수 있도록 한다. 그러면 구현 결과에 대해 만족스럽지 못한 부분, 즉, 더 나은 결과가 쉽게 파악된다거나 배선이 이루어지지 않은 부분에 대해 기능 모듈 생성 전용 편집기를 사용하여 보다 나은 구현 결과를 빠른 시간 내에 얻을 수 있을 것이다.

이러한 목적을 위하여 본 논문에서 제안하는 기능 모듈 생성을 위한 전용 편집기의 특성은 완전 수동 방식의 기능 모듈 설계를 그 기본 바탕으로 하고, 자동 방식에 의한 결과를 이용할 수 있도록 하면서 기능 모듈 설계에 있어서의 특성을 고려한 반자동 기법을 갖추고 있다는 점등을 들 수 있다. 본 논문의 이어

지는 장에서는 이러한 편집기에 대해 기능 모듈 구현의 전반적인 과정과 함께 기술한다.

3. 기능 모듈 전용 편집기

3.1. 편집기의 구성 및 기능

기능 모듈을 설계하기 위한 기본적인 요소로는 논리 회로, 기초셀의 이미지와 그 기초셀로 구성된 기초셀 어레이 원판, 그리고 배치 및 배선기 등이 있다. 기초셀의 형태를 선택하는 것으로부터 시작해서 주어진 논리회로의 모듈을 게이트 어레이 상에 구현하기까지의 모든 작업을 편집기에서 한다. 그 구현 결과는 디지털 회로 설계 시스템으로 보내져 마스크 레이아웃 형태로써 회로 검증용 통해 기능 모듈 라이브러리로 첨가된다.

일반적으로 게이트 어레이의 기초셀은 이를 제공하는 회사에 따라 그 형태가 서로 다른데 이의 형태가 정해지면 편집기에서 추가로 제공하는 기초셀 편집기에 의하여 이를 입력하여 저장할 수 있다. 입력된 기초셀은 차후 이 편집기로 기능 모듈을 생성하거나 설계된 모듈을 수정할 때 배경 이미지로 나타나게 된다.

기능 모듈 설계를 위하여 편집기가 수행하는 첫 단계는 모듈 설계자가 선택한 기초셀로부터 폴리(poly) 또는 디퓨전(diffusion) 등의 서로 같은 물리적 특성을 갖는 영역들을 분리하고, 그들로 이루어지는 트랜지스터들을 모두 찾아낸다. 이 기초셀들은 게이트 어레이 원판 위에 수평, 수직으로 반복 배치되어 기초셀 어레이로 구성되고, 각 기초셀에서 추출한 각 트랜지스터의 터미널들에는 고유한 번호가 부여된다. 이러한 작업이 완료되면 모듈 설계자는 설계하고자 하는 논리 회로로부터 네트 리스트를 얻어 게이트 어레이 위에 배치하고 이들 간을 배선하는 과정을 수행한다.

배치는 설계자가 직접 할 수도 있고, 배치 알고리즘을 통해 그 결과를 얻을 수도 있는데[1, 4, 13], 설계자는 배치된 결과를 편집기 상에서 확인할 수 있다. 즉, 각 네트의 연결되어야 할 상태는 최소 스페닝 트리(MST)를 구하여 이를 에어-라인(air-line)으로 나타내 주고, 터미널간의 연결이 완료될 때마다 해당 에어-라인은 사라지게 된다.

배선 및 편집은 수동 및 반자동의 두 가지 방법을

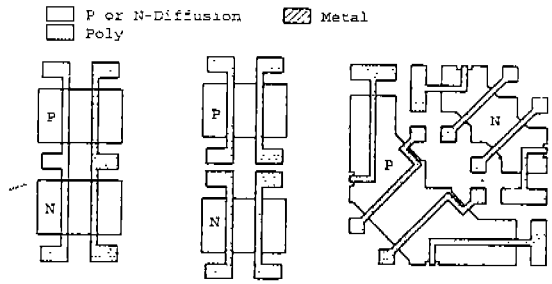
함께 사용할 수 있는데, 설계자가 에어-라인을 참고로 하여 배선 작업을 하나하나 수동으로 할 수도 있고, 설계자의 배선 의도를 파악해 어느 정도 자동으로 할 수도 있다. 하나의 네트에 대한 배선이 완료되면 배선의 완료를 알리는 메시지가 나오면서 그 자체는 심볼릭 레이아웃이 된다. 배치 및 배선의 결과에 대한 정보는 각각 텍스트 파일로 저장되어 필요할 때 언제든지 수정할 수 있다. 이러한 과정을 통해 생성된 기능 모듈은 마스크 레이아웃, 시뮬레이션을 통해 셀 라이브러리에 첨가시키기 위한 자료로 사용되게 된다.

3.2. 기초셀 어레이와 기본 트랜지스터의 추출

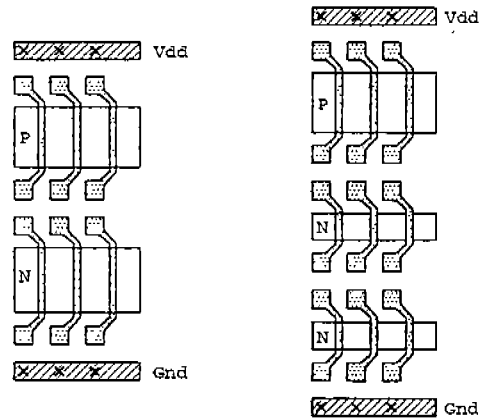
게이트 어레이의 구조는 oxide isolation 기법에 의한 "sea-of-gates"와 gate isolation 기법에 의한 "sea-of-transistor" 어레이 등의 두 가지 형태가 있다[1, 10]. 전자의 경우, 기초셀 모양은 다양하지만 한 셀을 다른 셀로부터 고립시키기 위하여 oxide를 통하여 디퓨전 레이어를 분리한다는 공통적인 특징을 갖는다. 이에 대해 후자의 경우는 셀의 양 끝 트랜지스터 게이트를 전원 선에 연결함으로써 다른 셀로부터 전기적으로 고립시킨다. 기초셀의 다양한 형태들을 (그림 2)의 (a)(b)에 각각 보인다.

기초셀의 유형은 여러 가지이므로 설계를 위해 주어진 기초셀을 구현된 편집기를 이용하여 일도록 한다. 사용자는 이미 정의되어 있는 기초셀 모양을 선택할 수 있을 뿐 아니라 현재 사용하여야 할 기초셀이 없는 경우에는 본 편집기에서 추가로 제공하는 기초셀 전용 편집기를 이용하여 설계 규칙을 충분히 고려하면서 주어진 형태에 맞게 입력할 수 있다. 이제, 기초셀이 주어졌을 때, 그로부터 기본 트랜지스터를 추출해 내는 방법 및 과정에 대해 설명하기로 한다.

우선, 주어진 기초셀에 대한 저장 파일로부터 기초셀을 구성하는 다각형 모양을 읽어 들여 그 내부의 점들을 찾고, 그 점들이 갖는 특성에 따라 각 점들의 집합을 구성하도록 한다. 여기서 각 점들은 주어진 배선 그리드의 한 점과 일치하는 다각형내부의 점들로 이들 위치에서 컨택(contact)을 생성하거나 배선을 수행한다. 다각형 내부의 점들을 같은 물리적 상태, 즉, 전기적으로 동일한 상태에 따라 서로 구분하는데, 이 과정을 우선 폴리로 주어지는 다각형에 대해 적용



(a) Oxide-isolation 구조[1.5, 11].



(b) Gate-isolation 구조[11].

(그림 2) Oxide-isolation 또는 Gate-isolation 구조의 기초셀 형태.

(Fig. 2) The Various types of Basic Cells.

하여 주어진 기초셀 내의 모든 폴리들을 같은 전기적 상태를 갖는 것들 끼리로 구분하도록 한다. 모든 폴리에 대해 이 작업이 끝나면 디퓨전인 다각형에 대해 같은 과정을 적용하는데, 디퓨전의 영역이 되는 다각형 내의 점들은 폴리에 의해 서로 다른 물리적 상태로 나누어진다. 이러한 작업은 flood fill 알고리즘[12]을 이용하여 영역 내의 한 점을 "seed"로 하고, 그것의 이웃이 되는 각 네 점에 대해 순환적으로 조사하는데 이때 폴리가 되는 점을 경계로 같은 영역 내에 있으면 동일한 상태의 점으로 간주한다.

이러한 과정을 통하여 주어진 기초셀에 포함된 모든 트랜지스터들을 찾아낼 수 있다. 즉, 하나의 폴리선과 그 폴리 선을 경계로 하는 디퓨전의 양쪽 영역

이 하나의 트랜지스터를 형성하는데 그 모양이나 이루어진 형태에 제한 없이 디퓨션 타입에 따라 P-, 혹은 N-트랜지스터로 구분하여 기초셀 내의 트랜지스터를 전기적으로 동일한 터미널들로 이루어진 집합으로 분리하여 추출해 낸다.

하나의 폴리 또는 디퓨션 블럭은 여러 개의 그리드 상의 점들을 포함하고 있는데 이들 중 하나만 선택하여 그 점위에 콘택을 생성하고 다른 블럭의 한 점과 연결하면 배선이 된다. 즉, 한 블럭 내에서 추출된 점들은 모두 전기적으로 동등하며 따라서 이러한 점들을 동등 터미널(equivalent terminal, (그림 1) 참조)이라고 한다. 구형한 편집기에서는 이러한 동등 터미널을 관리하여 이중 하나만 배선에 사용되면 이 블럭으로의 배선이 완료된 것으로 간주한다. 이러한 기능은 어떠한 다른 편집기에서도 가지고 있지 않은 제한한 편집기만의 고유한 기능이라고 할 수 있다.

3.3 배치 및 배선

기능 모듈을 생성하기 위해 먼저 논리회로 내의 트랜지스터들을 기초셀 어레이에 할당해야 한다. 지금까지 이에 대한 여러 알고리즘들이 개발되어 왔는데 [1, 4, 13], 배치 알고리즘이 원판의 특성에 적합하지 않거나 알고리즘의 결과가 설계자를 만족시키지 못하는 경우, 또는 원판에서의 설계상 알고리즘으로부터 배치 결과를 얻기 어려운 상황 등이 있을 수 있다. 이러한 점을 보완시키기 위해 수동으로 트랜지스터의 배치를 할 수 있는 기능이 요구된다. 따라서 제안한 편집기에서는 필요에 따라 기능 모듈 설계자가 고유의 문자와 번호를 이용하여 생성하고자 하는 기능 모듈에 대한 회로의 각 트랜지스터를 기초셀 어레이 위에 직접 할당할 수 있으며 또한 배치결과를 수정할 수도 있다.

배치가 끝나면 서로 연결하여야 할 트랜지스터들의 터미널간의 배선을 해결하여야 한다. 이러한 배선을 위해서는 기존 발표된 여러 자동 배선 방법을 이용할 수도 있지만 이 경우 생길 수 있는 몇 개의 미결선으로 인하여 기능 모듈의 생성 시간이 길어지기도 하고 최악의 경우 그 미결 선을 해결하지 못할 가능성도 있다. 또한, 그 배선 결과가 만족스럽지 못하여 이를 수정하고 싶은 경우도 발생할 수 있다. 이러한 상황을 효율적으로 해결하기 위하여 제안한 편집기

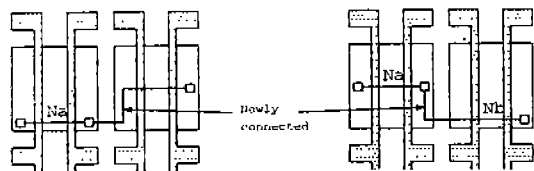
에서 제공하는 기능들은 완전 수동에 의한 방법과 반자동에 의한 방법 등이 있는데 본 절의 나머지 부분에서는 이에 관하여 설명한다.

3.3.1. 완전 수동에 의한 방법

논리 회로로부터 트랜지스터를 추출하여 원판 내에 배치하는 것으로부터 그들 사이를 배선하여 하나의 기능 모듈을 얻어내기까지를 완전 수동에 의한 방법으로 할 수 있다. 우선, 배치를 위해서 논리 회로의 트랜지스터들을 모두 추출해 낸다. 각 트랜지스터를 기초셀 어레이에 할당함으로써 배선 단계를 시작할 수 있는데, 설계자의 편의에 따라 편집기의 기능을 이용할 수도 있다. 예를 들면, 서로 연결하여야 할 터미널들에 하나의 동일한 번호를 대응시키고 게이트 어레이 원판 상에 있는 각 트랜지스터에 해당 번호를 표시하여 주면 된다. 이러한 배치가 완료되면 동일한 번호를 갖는 터미널들을 조건에 맞게 두개나 그 이상의 메탈 선을 사용하여 서로 연결해 주도록 한다.

완전 수동에 의한 방법에서는 설계자가 임의로 배선 작업을 수행할 수 있는데 이를 위해 편집기에서 내부적으로 고려된 사항들은 다음과 같다. 먼저, 메탈 레이어의 와이어간의 연결이나 혹은 분리에 관한 것으로, 만일 두 메탈 와이어가 서로 이어져 새로운 하나의 메탈 와이어가 될 때, 어느 한 와이어에 네트의 이름이 정해져 있으면 이를 새로운 와이어의 네트 이름으로 정하고(그림 3)(a) 양쪽 모두에 서로 다른 이름이 있으면 에러로 처리한다(그림 3)(b). 또한, 배선은 수평 또는 수직으로만 이루어지므로 대각선의 메탈 와이어도 오류로 처리된다. 메탈 와이어의 분리는 두 와이어가 서로 교차됨으로써 생기는 점에 의해 나뉘어지고, 이후 메탈 와이어의 수정이나 삭제는 그 점을 기준으로 이루어진다. 다음으로, 동등 터미널을 이용하여 하나의 네트로 연결하는 경우에 관한 것으로, 네트를 이루는 각각의 터미널은 자신 외에 동등한 터미널을 가지고 있을 수 있어 필요한 경우 그것을 이용하여 배선할 수도 있는데 이렇게 하면 들쭉 메탈 레이어의 사용을 줄일 수 있다.

완전 수동에 의한 배선에 있어 보다 빠른 시간 내에 보다 적은 수정 방법으로 배선을 완료하기 위해서는 본 편집기를 이용할 때 설계하는 과정에 다음 사항을 고려하도록 한다. 그것은 네트를 처리하는 순서



(a) 네트 이름의 복사. (b) 잘못된 배선의 예.

(그림 3) 배선에 따른 네트 이름의 처리.

(Fig. 3) Net name processing with net routing.

에 관한 것인데, 첫 번째로 전원선의 배선이다. 전원선의 위치는 미리 정해져 있거나, 그렇지 않더라도 일반적으로 그 위치를 고정시키기 때문에 모듈 생성에 필요한 Vdd나 Gnd선을 제일 먼저 고정시켜 두어야 한다. 그렇지 않으면 차후 배선에 많은 영향을 받게 되기 때문이다.

두 번째는 P-part나 N-part의 어느 한쪽에만 연결 조건을 갖는 네트에 대한 배선이다. 대체적으로 이러한 네트들은 전원 선에 가깝게 배선하는 것이 좋은데 이는 다른 네트들이 좀더 자유롭게 배선될 수 있도록 한다.

세 번째로 폴리 게이트로만 연결 조건을 갖는 네트에 대한 배선이다. 이들을 우선적으로 배선해 주지 않으면 보다 많은 배선에 대한 수정 작업이 일어나거나 될 수 있으면 억제되어야 할 들쭉 메탈 레이어의 사용이 늘어나게 된다. 남은 네트들에 대해서는 폴리 게이트를 연결해야 하는 수가 많은 네트들을 우선으로 하면서, 해당 터미널들을 모두 포함하는 bounding box의 크기가 작은 네트들부터 배선 과정을 완료해 간다.

구현한 편집기는 이러한 규칙을 내장하여 편집기 스스로 어떤 네트를 먼저 배선할 것인지를 설계자에게 윈도우 상에서 highlight를 통하여 알려준다. 따라서, 설계자는 완전 수동에 의한 배선을 수행할 경우에도 수정이 거의 필요 없이 보다 편리하게 배선을 수행할 수 있다.

3.3.2. 반자동에 의한 방법

수동에 의한 트랜지스터의 배치는 설계자가 목적에 따라 임의로 배치할 수 있고, 배치 상황을 마음대로 변경할 수 있다는 장점이 있지만 회로의 크기가 클 경우 좋은 배치 결과를 얻기란 쉬운 일이 아니다.

이러한 경우 자동 배치 프로그램[1, 4, 13]을 이용하여 게이트 어레이 원판에 맞는 배치 결과를 얻도록 한다. 그러면 구현된 편집기에서는 이 결과를 설계자가 직접 확인할 수 있도록 보여준다.

배치 결과는 네트에 부여된 고유 ID로 확인해 볼 수도 있으며, 각 네트에 해당하는 네트 이름으로 볼 수도 있다. 또한, 전체 배치 결과에 따른 연결 상황에 에어-라인으로 보여주는데, 서로 연결되어야 할 두 터미널을 하나의 라인으로 이어줌으로써 기능 모듈 설계자가 연결 상황을 쉽게 파악할 수 있도록 한다.

에어-라인을 위해서 각 네트의 터미널들을 정점으로 하고 네트 내의 두 터미널들 거리를 이어 그 거리를 비용으로 하는 에지를 만들어 무방향 연결 그래프를 구성한다. 거리를 구할 때 동등 터미널의 존재 여부에 따라 한 정점에 대한 터미널의 수가 유일하지 않으므로 그들중 가운데 터미널을 택하여 이를 정점으로 한다. 이렇게 구성한 그래프로부터 최소 스패닝 트리(MST)를 찾아 그 결과를 에어-라인으로 보여주도록 하면 되는데 이를 위해 PRIM의 MST 알고리즘[8]을 사용하였다.

이제, 배치된 결과를 보여주는 에어-라인을 따라 배선을 하도록 하는데, 앞 절의 완전 수동에 의한 방법에서 제시한 순서대로 배선을 해 나간다. 반자동에 의한 배선에서는 완전 수동 방식에서 처리되는 사항을 기본적으로 포함하면서 그보다 더 강력하게 사용자 지원하므로 이 방법을 우선적으로 사용하고 경우에 따라 필요하다면 수동 방식을 함께 취하도록 한다.

반자동에 의한 배선이란 설계자의 배선 의도를 어느 정도 파악해 뒤따르는 나머지 처리들을 자동으로 해주거나, 설계자의 의도와는 관계없지만 설계자가 차후에 연결해야 할 배선의 일부를 자동적인 방법으로 미리 처리해 주는 것을 의미한다. 즉, 본 편집기의 반자동 모드에서 설계자는 배선하고자 하는 네트의 터미널 분포를 관찰한 후 두 터미널을 선택하여 이들 간의 가장 바람직한 배선 형태를 결정하여 이를 마우스로 입력한다. 이때 설계자가 입력한 배선은 기존의 다른 네트의 배선 결과와 교차하거나 겹칠 수 있는데 편집기는 이들을 자동으로 다시 배선하여 네트간의 교차가 없는 올바른 배선으로 만들어 준다.

이러한 재배선에 대하여 구체적으로 논의하도록 한다. 먼저, 그 처리 과정에 대한 것으로 어떤 네트에

대한 배선을 시도할 때 그로부터 영향을 받는 네트들을 재배선 하기까지를 요약하면 다음과 같다.

- 단계 1. 새로운 네트(N)에 대한 배선 시도.
- 단계 2. 네트 N의 배선으로 인해 재배선 되어야 하는 네트들의 집합 $RN = \{N_1, N_2, \dots, N_n\}$ 을 구성한다.
- 단계 3. 각 네트 N_i ($1 \leq i \leq n$)에 대해,
 - 3.1. 재배선이 가능하면 RN에서 제거.
 - 3.2. 재배선이 불가능하면 그 원인이 되는 네트 중 하나를 제거하고 다시 한 번 재배선 시도, 제거된 네트는 재배선 되기 위해 RN의 원소가 된다.

다시 배선되어야 할 네트의 양 끝 두 점을 얻기 위해 네트 N의 배선과 접친 부분으로부터 시작하여 콘택이 있거나 다른 와이어 세그먼트와 교차하는 점을 만날 때까지 재 배선할 네트의 와이어 세그먼트들을 계속 조사하여 양 끝점을 결정한다. 양 끝점을 얻었으면 재배선을 시작하는데 재배선의 수가 많아질 수 있으므로 그 값에 한계를 정해 두도록 한다. 재배선이 이루어진 네트라 하더라도 더 나은 배선 방법이 있으면 설계자가 수정할 수 있다.

다음으로, 재배선 하는 방법을 설명한다. N_c 와 N_r 을 각각 새로 배선한 네트 그리고 다시 배선하여야 할 네트라고 가정한다. 또한, A와 B는 N_r 의 연결하여야 할 양 끝점인데, A는 N_c 의 와이어 세그먼트와 만나는 점이며 B는 그렇지 않은 점이라고 가정한다. 만약, 두 점이 N_c 의 와이어 세그먼트와 만나지 않으면 임의로 A, B를 결정한다. 이러한 정의 하에 N_c 의 배선으로 인해 다른 형태로 다시 배선되어야 하는 네트 N_r 에 대해 다음과 같은 단계로써 그 네트의 재배선을 시도한다.

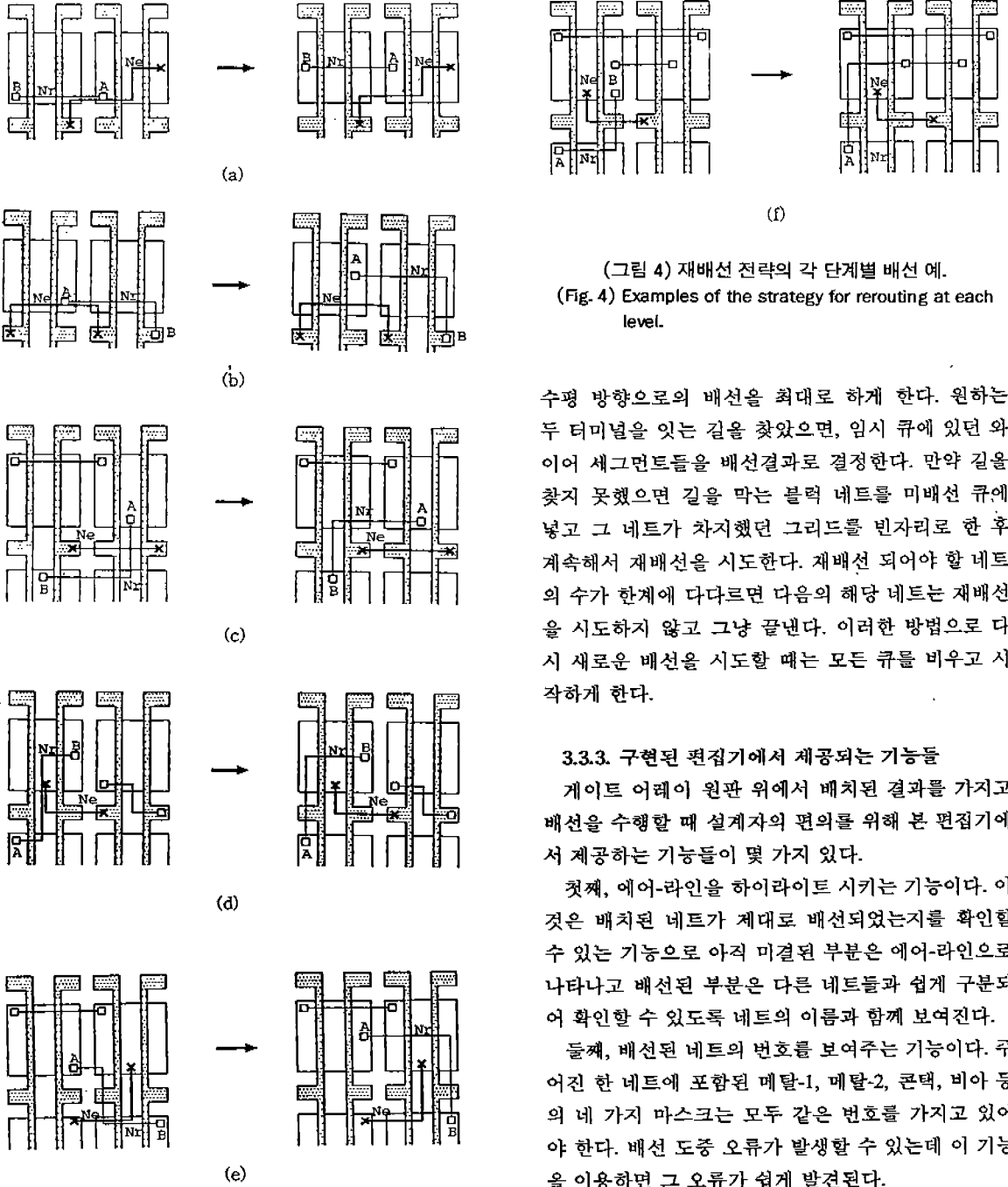
- 단계 1. N_c 가 N_r 의 콘택이 있는 터미널의 위치를 지날 때, 콘택이 있는 터미널의 가장 가까운 동등 터미널을 찾아 그 점으로 N_r 을 배선한다.
이 때, 재배선이 가능하면 단계 6으로 간다.
- 단계 2. 점 A와 점 B로 이루어지는 bounding box 내부에서 재배선을 시도한다. 재배선이 가능하면 단계 6으로 간다.

단계 3. 점 A와 점 B로 이루어지는 bounding box를 먼저 점 A에 대해 확장시키고 그 안에서 재 배선을 시도한다. 해결이 안되면 점 B에 대해서도 확장시켜 재시도 한다. 재배선이 가능하면 단계 6으로 간다.

- 단계 4. 다른 동등 터미널을 찾고 그들을 이용하여 재배선을 시도한다.
재배선이 가능하면 단계 6으로 간다.
- 단계 5. 재배선이 불가능인 채로 이 과정을 마친다.
- 단계 6. 재배선을 성공적으로 수행하였으므로 이 과정을 마친다.

콘택이 포함되어 있는 와이어 위로 배선을 시도하는 (그림 4)(a)(b)와 같은 경우는 모두 단계 1에 해당 되는데 콘택이 있었던 터미널 위에 또다시 콘택을 놓을 수 없으므로 다른 동등 터미널을 찾아서 배선한다. 이를 실패했거나 이의 조사 대상이 아닌 경우 제 2 단계로 bounding box안에서의 재배선을 시도한다 ((그림 4)(c)(d)). 배선이 가능하지 않다면 세 번째 단계로 bounding box를 확장시켜 재배선 해본다(그림 4)(e)). 여전히 배선이 안되는 네트에 대해서는 가장 가까운 동등 터미널 또는 같은 네트의 둘 이상의 와이어가 서로 만나는 점을 찾아 그 점으로 이루어지는 bounding box내에서 배선을 시도한다((그림 4)(f)). 이러한 재배선의 시도는 계속 일어날 수 있으므로 적절한 한계 값을 설정해 다른 네트의 배선에 지장을 주지 않도록 한다.

주어진 bounding box내에서의 배선 방법으로는 여러 가지가 가능하겠지만, 배선 단계를 고려해 불필요한 배선 시도는 줄이도록 하는 방법을 택하였다. 먼저, 재 배선되어야 할 네트에 대해 배선 가능성을 조사할 때, 첫째 메탈 레이어에서는 수직보다 수평으로 배선하는 것이 좋으므로 수평 방향으로 먼저 조사한다. Bounding box 내부에서 다른 네트가 이미 배선되어 있음이 발견되었으면 그 때까지 배선된 첫째 메탈 레이어의 와이어들을 네트 큐에 임시 저장한다. 그 다음 수직으로 방향을 바꾸고, 바로 인접한 그리드에 배선된 네트가 없을 때까지 수직으로 진행한다. 이것은 수직 메탈 레이어의 배선을 최소화하기 위한 의도이다. 인접한 그리드가 비어 있을 경우, 다시 그 때까지 배선된 네트를 임시 큐에 저장하고 수평으로 진행한다. 수평 방향으로는 갈 수 있을 때까지 진행하여



(그림 4) 재배선 전략의 각 단계별 배선 예.
 (Fig. 4) Examples of the strategy for rerouting at each level.

수평 방향으로의 배선을 최대한 하게 한다. 원하는 두 터미널을 잇는 길을 찾았으면, 임시 큐에 있던 와이어 세그먼트들을 배선결과로 결정한다. 만약 길을 찾지 못했으면 길을 막는 블록 네트를 미배선 큐에 넣고 그 네트가 차지했던 그리드를 빈자리로 한 후 계속해서 재배선을 시도한다. 재배선 되어야 할 네트의 수가 한계에 다다르면 다음의 해당 네트는 재배선을 시도하지 않고 그냥 끝낸다. 이러한 방법으로 다시 새로운 배선을 시도할 때는 모든 큐를 비우고 시작하게 한다.

3.3.3. 구현된 편집기에서 제공되는 기능들

게이트 어레이 원판 위에서 배치된 결과를 가지고 배선을 수행할 때 설계자의 편의를 위해 본 편집기에서 제공하는 기능들이 몇 가지 있다.

첫째, 에어-라인을 하이라이트 시키는 기능이다. 이것은 배치된 네트가 제대로 배선되었는지를 확인할 수 있는 기능으로 아직 미결된 부분은 에어-라인으로 나타나고 배선된 부분은 다른 네트들과 쉽게 구분되어 확인할 수 있도록 네트의 이름과 함께 보여진다.

둘째, 배선된 네트의 번호를 보여주는 기능이다. 주어진 한 네트에 포함된 메탈-1, 메탈-2, 콘택, 비아 등의 네 가지 마스크는 모두 같은 번호를 가지고 있어야 한다. 배선 도중 오류가 발생할 수 있는데 이 기능을 이용하면 그 오류가 쉽게 발견된다.

셋째, Undo의 기능으로 설계자가 어떤 네트를 배선하려다가 그것을 정정하고 싶을 때, 배선되지 않은 원래의 상태로 회복시키는 기능이다. 배선 의도가 잘못되었거나 배선을 다른 방향으로 바꾸고자 할 때 사

용하고, 네트에 이름을 지정할 때에도 마찬가지로 방법으로 사용할 수 있다.

넷째, 메시지를 전달하는 기능이다. 배선 도중 에러가 발생하였거나 배선이 완료되었음을 알려주는 기능으로, 처리되는 에러로는 설계자가 수직이나 수평이 아닌 대각선 방향으로 배선을 시도할 때, 서로 다른 이름이 있는 두 네트를 연결시키려고 할 때, 혹은 첫째 메탈 레이어의 와이어가 없는 상태에서 둘째 메탈 레이어로 배선을 할 때 등이 있다.

마지막으로 본 편집기는 심볼릭 레이아웃 편집기이므로 사용자는 단지 트랜지스터의 배치 및 배선만 해주면 된다. 나머지 실제 마스크 레이아웃으로 변환하는 작업은 자동으로 처리하여 준다. 이외에도 동등 터미널이나 트랜지스터 번호를 확인해 볼 수 있게 하여 배선을 보다 편리하고 신속하며 정확하게 할 수 있도록 하였다.

3.4. 설계의 전체 흐름도

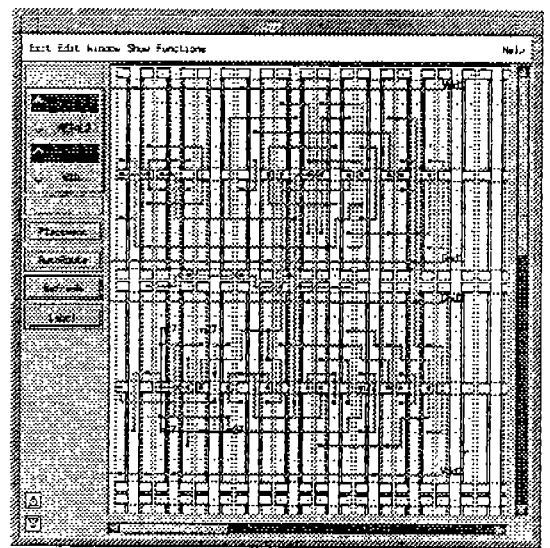
기능 모듈을 설계하기 위해 구현한 본 편집기는 X-윈도우 시스템을 바탕으로 하였다. X-윈도우 시스템은 Xlib, XtIntrinsics, widget set으로 구성되어 있는데 widget set으로는 OSF/MOTIF의 것을 사용하였다. X-윈도우 시스템을 바탕으로 하는 모든 응용 프로그램에서는 입출력 혹은 윈도우에 관한 작업들을 X 프로토콜에 맞춰 서버에게 요청하고 그 결과를 이벤트로써 통보 받는다. 본 편집기에서는 배치 및 배선 작업이 이벤트 주도로 처리되는데 그 전반적인 흐름은 다음과 같다.

우선, 다양한 형태의 기초셀 중 원하는 것을 하나 선택하고, 그로부터 기본 트랜지스터를 추출한다. 기초셀을 게이트 어레이 원판 위로 반복 배치하고 기초셀 어레이를 초기화한다. 이제 이벤트 주도하의 배치 및 배선을 처리하는데, 선택한 기초셀로 구성된 게이트 어레이 원판의 메인 윈도우에서 배치 결과 파일을 읽어 들이면 배선되어야 할 상황을 보여주는 에어라인이 네트의 이름과 함께 나타난다. 배선을 시작하기 전에 설계자는 이러한 각 네트에 대한 에어-라인을 하나씩 조사하고 배선의 난이도를 파악하여 네트의 배선순서를 정할 수 있다.

원하는 네트의 배선을 수행하고자 할 때, 마우스 버튼을 이용하는데 클릭된 두 점에 대해 반자동 배선

이 시작된다. 배선된 결과는 필요한 경우 이를 취소할 수도 있으며 이 경우 설계자가 약간의 배선을 수동으로 한 후 배선기가 자동으로 배선을 수행하도록 하여 원하는 배선 결과를 얻을 수 있다. 또한 신속한 배선을 위하여 배선과 동시에 콘택이나 비아도 함께 생성하도록 할 수도 있다. 그리고 배선도중 기존 배선된 결과의 일부도 자유롭게 수정할 수 있다. 마지막으로 하나의 네트에 대한 배선이 완료되면 자동으로 다음 배선되어야 할 네트를 에어-라인으로 표시해 주어 사용자가 쉽게 배선 단계를 진행할 수 있도록 한다. 배선 순서는 배선기가 정할 수도 있으나 설계자가 임의로 정하여 배선을 수행할 수 있다.

배선이 완료되면 심볼릭 레이아웃 그대로 하나의 기능 모듈이 되며, 다시 새로운 기능 모듈을 같은 원판에서 또는 다른 원판을 구성하여 설계를 시작할 수 있다. 그림 5에는 주어진 논리회로에 대해 위의 과정을 거친 배치 및 배선의 레이아웃 결과를 구현된 편집기와 함께 보인다.



(그림 5) 편집기에 구현된 레이아웃.
(Fig. 5) The layout implemented on Editor.

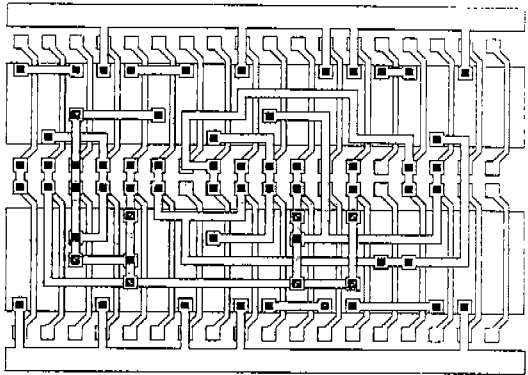
4. 구현 결과

게이트 어레이에서 기초셀 라이브러리 구축에 필

요한 기능 모듈을 설계할 때 설계자가 보다 편리하고 만족스런 설계 작업을 할 수 있도록 기능 모듈 생성 편집기를 구현하였는데, 이를 위해 X-윈도우 시스템을 그 기본 환경으로 하였으며 C언어를 사용하였고, 삼보 SDT-300 워크스테이션에서 그 기능을 시험, 수행하였다. 본 편집기로 기능 모듈을 설계한 결과중 하나를 (그림 6)과 (그림 7)에 보인다.

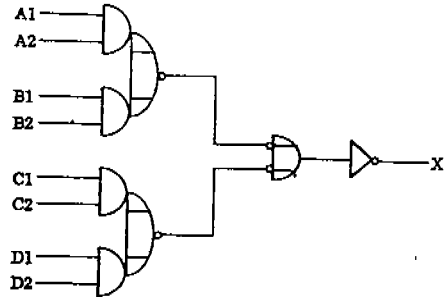
본 논문에서 제안한 편집기를 이용할 때 논리 회로의 구현이 얼마나 효과적으로 이루어지는가를 알아보기 위해 대화 방식의 설계가 가능한 시스템인 OCEAN[5]의 Seadali라는 편집기를 이용했을 때와 그 설계 결과를 비교하였다. 즉, 다섯 가지의 논리 회로에 대하여 임의로 선정된 세 사람이 주어진 각 논리 회로를 구현할 때, 구현에 필요한 설계시간 및 설계에 사용한 수평 트랙의 수, 그리고 사용한 메탈의 전체 길이를 조사하였다. 생성 시간은 주어진 논리 회로의 배치 결과로부터 배선의 결과를 확인하기까지 소요된 시간을 측정했으며, 메탈의 총 사용 길이는 그리드 하나를 그 기본 단위로 하여 조사하였다. 평균 설계시간은 세 사람이 소요한 시간의 결과를 평균한 것이다.

이러한 비교 결과를 <표 1>에 보인다. 표에서 보인바와 같이 기존의 시스템을 사용한 것 보다 약 2배 정도의 설계시간을 단축할 수 있었다. 그러나 Seadali 역시 게이트 어레이 설계 시스템으로 만들어져 기능 모듈 편집도 배경에 기초셀 이미지가 나타나고 따라서 설계자는 완전 수동으로 기초셀 이미지 상에서 배치 및 배선만을 수행하므로 어느 정도 게이트 어레이용 기능 모듈 생성을 위한 편집기라고 할 수 있다. 따라

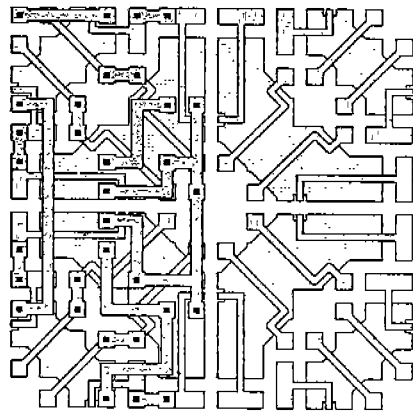


(b) 마스크 레이아웃.

(그림 6) NON-SCAN DFF with RESET.
(Fig. 6) NON-SCAN DFF with RESET.

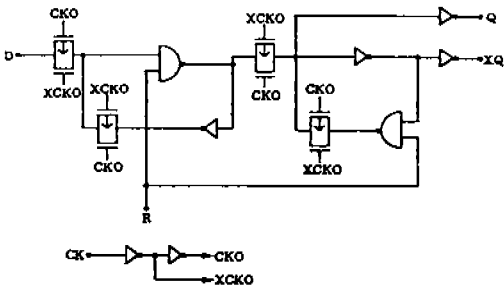


(a) 논리 회로.



(b) 마스크 레이아웃.

(그림 7) Power 2-And 4-wide Multiplexer.
(Fig. 7) Power 2-And 4-wide Multiplexer.



(a) 논리 회로.

〈표 1〉 주어진 논리 회로의 구현에 대한 비교 결과.
 〈Table 1〉 The result of comparison about the implementation of given logic circuit.

논리 회로	셀의 일 수	수평 트랙수	실제자 비교내용	기존의 물 (OCEAN)			에디터		
				J	P	S	J	P	S
Non-scan DFF with RESET (oxide-isolation)	1	15	생성시간 (배치및배선)	12	22	38	8	4	27
			사용한 수평트랙의수	12	12	12	13	13	10
			total wire length	175	156	169	177	174	154
			평균설계시간	24			13		
Non-scan DFF with RESET (gate-isolation)	1	15	생성시간 (배치및배선)	30	21	88	25	5	10
			사용한 수평트랙의수	13	13	15	13	15	13
			total wire length	176	146	156	158	172	163
			평균설계시간	46			23		
Scan 4-input DFF with clear & clock-inhibit	2	30	생성시간 (배치및배선)	59	52	53	32	12	61
			사용한 수평트랙의수	27	26	28	28	28	23
			total wire length	342	326	338	342	352	274
			평균설계시간	55			35		
Non-scan positive edge clocked power J-K flip flop with clear	3	45	생성시간 (배치및배선)	56	44	108	14	10	64
			사용한 수평트랙의수	31	32	28	34	33	28
			total wire length	282	252	283	342	290	288
			평균설계시간	69			29		
4-bit Data Latch with clear	4	60	생성시간 (배치및배선)	51	57	156	27	20	101
			사용한 수평트랙의수	48	47	38	42	50	37
			total wire length	340	342	340	341	371	317
			평균설계시간	88			49		

서 실제 상용 레이아웃 편집기로 기능 모듈을 생성할 때와 그 소요 시간을 비교하면 우리의 편집기를 사용하는 것이 훨씬 적게 필요할 것이다.

5. 결 론

기능 모듈의 설계를 자동화하는 것은 디지털 회로의 레이아웃 구현에 있어 필수적인 것이고, 그런 설계 자동화에 있어 범용성을 띠어야 한다는 것 또한 반드시 요구되는 조건이다. 이에, 본 논문에서는 이러한 요구들을 충족시킬 수 있도록 기능 모듈 설계를 위한 게이트 어레이 전용 편집기를 제안하였다. 본 편집기는 어떠한 기초셀 형태라도 그 위에서의 기능 모듈 설계가 가능하도록 하였으며 보다 효과적인 설계 작업이 가능한 환경을 구성하였다. 즉, 사용자와의 인터페이스가 잘 이루어질 수 있도록 X-윈도우 시스템을 사용하였으며 특히, 반자동 배선 기법을 사용하여 설계자가 모듈 설계를 위하여 수행하여야 할 많은 일들을 줄였다. 구현한 편집기를 평가하기 위하여 기존의 편집기를 사용하여 모듈을 생성할 때 소요되는 시간과 우리의 편집기로 생성하는데 소요되는 시간을 비교하였을 때 약 2 배 정도의 생성 시간을 단축할 수 있었다. 끝으로 구현한 편집기에 자동 생성기를 추가한다면 보다 더 효율적인 기능 모듈 생성기가 될 것이며 현재 이를 추진중이다.

참 고 문 헌

- [1] Glenn D. Adams and Carlo H. Sequin, "Template Style Consideration for Sea-Of-Gates Layout Generation," Proc. 26th Design Automation Conference, pp. 31-36, 1989.
- [2] J. Banker, A. Shanbhag, N. Sherwani, "Physical Design Tradeoffs for ASIC Technologies," Proc. 6th Annual IEEE International ASIC Conference and Exhibit, pp. 70-78, 1993.
- [3] M. A. Beunder, J. P. Kernhof and B. Hoefflinger, "The CMOS Gate Forest: An Efficient and Flexible High-Performance ASIC Design Environment," IEEE J. Solid-State, Vol. 23, No. 2, pp. 387-399. Apr. 1988.
- [4] Y. K. Choi, J. H. Lee and C. S. Rim, "Automatic Functional Cell Generation In The Sea-of-Gates Layout Styles," Proc. IEEE Int'l Symp. on Circuits and Systems, pp. 189-192, 1994.
- [5] P. Groeneveld, P. Stravers, OCEAN: the Sea-of-Gates Design System, Delft Univ. of Technology, Netherlands, Jul. 1993.
- [6] H. J. Kappen, F. M. J. deBont, "An Efficient Placement Method for Large Standard-Cell and Sea-of-Gates Designs," Proc. EDAC, pp. 312-316, 1990.
- [7] S. Noda, H. Yoshizwa, E. Fukuda, H. Kato, H. Kawanish and T. Fujii, "Automatic Layout Algorithms for functional Blocks of CMOS Gates Arrays," Proc. ICCAD-87, pp. 176-177, 1987.
- [8] C. H. Paradimitriou, K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity, Prentice-Hall, Inc.
- [9] E. Shragowitz, J. Lee and S. Sahni, "Algorithms for physical design of Sea-of-Gate Chips," Computer Aided Design, Vol. 20, No. 7, pp. 382-397, 1988.
- [10] R. S. Tsay, E. S. Kuh, and C. P. Hsu, "PROUD: A Fast Sea-of-Gates Placement Algorithm," Proc. 25th Design Automation Conf., pp. 318-323, 1988.
- [11] HARRY J. M. Veendrick, "An Efficient and Flexible Architecture for High-Density Gate Arrays," IEEE J. Solid-State Circuits, Vol. 25, No. 5, pp. 1153-1157, Oct. 1990.
- [12] F. S. Hill, JR., Computer Graphics, Maxwell Macmillan international ed., 1990.
- [13] 최연경, 임종석, "Sea-of-Gates 레이아웃 형태에서의 배치방법에 관한 연구," 한국 정보과학회, 학술발표 논문집, 제19권, 2호, pp. 503-506, 1992.



홍 성 현

1991년~1992년 서강대학교 수학과, 전자계산학과 졸업(학사).
1994년 서강대학교 대학원 전자계산학과(석사).
1994년 3월~현재 삼성전자 통신개발연구소 DSP 파트 연구원.

관심 분야: DSP control 소프트웨어, GUI 설계 등.



임 증 석

1981년 서강대학교 전자공학과 졸업(학사).
1983년 한국과학기술원 전기 및 전자공학과(석사).
1989년 Univ. of Maryland, College Park, 전기공학과 (박사).

1983년 3월~1990년 8월 한국 전자통신연구소 연구원, 선임연구원.
1990년 9월~현재 서강대학교 전자계산학과 부교수.
관심분야: VLSI CAD 알고리즘, DSP를 위한 VLSI 설계 등.



정 영 숙

1988년 이화여자대학교 전자계산학과 졸업(학사)
1989년~현재 한국전자통신연구소 H/W 개발환경 연구실 연구원
관심분야: Logic Emulation, 회로 다중분할, 시뮬레이션 알고리즘 등.



손 진 우

1973년 영남대학교 전자공학과 졸업(학사).
1994년 아주대학교 전자공학과 (박사).
1973년~1977년 공군 정보통신 장교.
1978년~1983년 삼성반도체통신(주) 설계기술과장.

1984년~현재 한국전자통신연구소 H/W 개발환경연구실장
관심분야: CAD/E, 데이터 통신망 등.