VLSI 레이아웃 이식 시스템에 관한 연구

곽성훈, 이기중, 김용배*, 이윤식**
㈜파이손테크, 호서대학교
e-mail : yslee@pishontech.co.kr, yslee@office.hoseo.co.kr

A Research for VLSI Layout Migration EDA System

Ki-Joong Lee, Sung-Hun Kwak, Yong-Bae Kim, Yun-Sik Lee
Pishon Technology Inc.
Dept. of Computer Engineering, Hoseo University

요 약

소형 고성능 가전기기를 실현하기 위한 다기능 고집적의 실리콘화에 대응하기 위하여 반도체 업계는 SoC(System On a Chip) 설계, 반도체 지적 재산권인 IP(Intellectual Property)에 관한 연구를 두개의 핵심 연구 항목으로 설정하여 진행되어 왔다. 반도체 레이아웃 이식 자동화 시스템은 설계 재활용(Design Reuse), IP 의 실용화와 확산을 위한 핵심 연구 과제 중의 하나로써, Time-To-Market 과 Time-To-Money 를 동시에 가능토록 하는 근간의 기술이 된다. 본 연구는 정확하고 고속의 IP 내의 반도체 소자 인식 알고리즘, 그래프를 이용한 제한 조건의 구현과 해석, 향상된 컴팩션(Compaction) 알고리즘의 연구로 말미암아 기존의 연구 결과 대비 평균 20 배의 속도 향상과 평균 41%의 메모리만을 사용함으로써 경쟁 기술 대비 월등한 우위를 보이고 있다. 이로써, 대형의 반도체 설계 도면의 처리를 가능하도록 하였으며, 반도체 IP 의 응용성(flexibility)을 부여 함으로써, IP 의 재활용의 기초 연구와 SoC 설계 확산에 지렛대 역할을 하는 연구가 되리라고 예측한다.

1. 서론

최근 급속한 공정 기술의 발전에 따라 수백만개의 트랜지스터로 구성된 복잡한 시스템을 단일 칩 상에 구현하는 것이 가능하게 되었다. 美 반도체 연구 조합인 SEMATECH 에 따르면 2001 년까지는 600MHz 로 동작하며 1,200 개의 논리소자로 구성된 반도체 칩이 등장할 것으로 전망된다[1,3,4]. 한명의 설계자가 하루에 100 개의 논리소자를 설계한다고 할 때 1,200 만개의 논리소자는 500 명이 1 년간 설계할 분량이며, 미화 75M$의 개발비용이 들게 된다. 지난 십 여년간 HDL (Hardware Description Language), 논리합성, HDL 시뮬레이터 등과 같은 기술들의 등장으로 괄목할 만한 생산성 향상에도 불구하고 설계자가 이와 같은 초대형 칩을 처음부터 설계한다는 것은 거의 비현실적인 일이다. 이러한 칩의 설계를 가능하게 하려면 표준화된 칩을 사용하거나 이미 설계 검증된 모듈들을 재활용하는 것만이 유일한 해결 방안이다. 또한 비용과 경쟁력 관점에서 볼 때, 가격과 성능의 요구조건을 만족하기 위하여 공정 기술은 점점 더 극 미세화 될 것이며, 동시에 새로운 공정 기술을 이용한 시스템 칩의 Time-to-Market 에 대한 시장의 압력은 점점 더 거세질 것으로 전망된다.

설계의 재활용(Design Reuse)은 기존의 설계를 재활용하는 것에 국한하지 않고, 이의 개념의 도입이 없이는 하나의 칩 당 개발비용이 98 년의 약 US 3M$에서 2007 년에는 US 193M$로 기하급수적으로 늘어나게 되리라는 예상이다[2]. 즉, SoC 설계 환경에 따른 새로운 기능들의 추가로 용량이 급속히 늘어나게 되고, 기존의 기능을 가급적 공유함으로써 새 기능의 추가 외에는 시간과 노력을 줄이고자 하는 것이 필수적인 상황이다. 역설적으로, 대형 칩의 설계와 검증이 점점 어려워져 감에 따라 설계의 효율이 점점 떨어지게 된다. 따라서 관련 자동화 소프트웨어에 대한 투자와 개발에 역점을 두어 설계 효율, 생산 효율을 증가 시키는 문제가 대두될 것이다. 최근 미국의 Motorola, National, ST Microelectronics 등과 같은 대형 반도체 업체들이 컨소시엄을 구성하여 SoC 를 효율적으로 개발하기 위한 발판을 마련하고 있다. 즉, 기존의 반도체 지적 재산권인 IP – 혹은 설계되어 생산 된 적이 있는 반도체 도면 -를 재사용하고, 재사용을 위한 IP 를 개발하는데 협력하고 있다는 사실만 보더라도 향후 설계의 재활

용을 위한 IP 의 중요성을 짐작할 수 있을 것이다.

IP 는 일반적으로 상위단계의 소프트 IP 와 레이아웃 단계의 하드 IP 로 분류된다. 소프트 IP 는 융통성을 제공하여 변경하기 용이한 반면, 레이아웃 단계에서의 반복적인 배치와 배선, 검증과정이 필요하다. 그러나, 하드 IP 는 기존의 셀 라이브러리와 블록들을 재사용하거나 변환하여 더 신속한 SoC 설계를 달성할 수 있다. 본 연구에서는 공정 기술 변화에 따른 마크로 라이브러리 및 ASIC 설계의 신속한 Time-to-Market 대응은 물론 재사용이 가능한 IP 또는 Virtual Components 라고 하는 기능 블록들의 생성, 제공 및 설계 응용 분야에 유용하게 사용될 수 있는 레이아웃 이식 또는 변환 (Layout Migration 또는 Technology Re-targeting)에 활용되는 기술을 연구하고 소프트웨어 시스템을 개발하는 것이다.

레이아웃 이식시스템은 그림 1 에서와 같이 4 가지 경우에 적용할 수 있다.

1. 기술 이식: 그림 1 의 첫번째 블록에서와 같이 0.6μm, 0.5μm, 0.35μm 등과 같은 서로 상이한 공정 기술(technology)의 설계규칙으로 설계된 레이아웃 도면을 조합하여 새로운 공정 기술(0.25 μm 이하)의 레이아웃 도면으로 변환한다.
2. 공정 이식: 그림 1 의 두 번째 블록에서와 같이 공정 A, 공정 B, 공정 C 등과 같은 서로 상이한 제조 공정용으로 설계된 레이아웃 도면을 조합하여 새로운 제조 공정 D 의 레이아웃 도면으로 변환한다.
3. 응용 이식: 그림 1 의 세번째에서와 같이 이미 검증된 표준 셀 라이브러리와 마크로 셀을 상이한 전압, 속도 및 특정한 응용에 따른 특성으로 변환한다.
4. 공정 선택: 그림 1 의 마지막 블록의 경우에서와 같이 예비 설계규칙으로 설계된 레이아웃 도면을 지정된 공정(A, B, 혹은 C 공정)으로 변환한다.

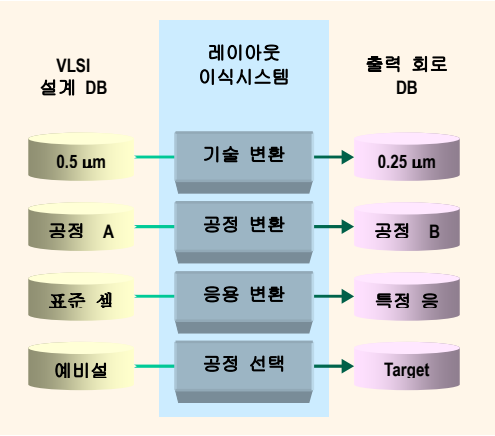

그림 1. 레이아웃 이식 시스템

그림 2, 3 은 그림 1 을 칩 크기의 변환 형태를 보여주고 있다. 즉, 그림 2 는 0.8μm ~ 0.35μm 기술로 설계된 레이아웃을 0.35μm 이하의 기술로 변환되었을 경우이며, 그림 3 은 동일한 공정 기술의 상이한 공정이나 특정한 응용으로 라이브러리를 변환하였을 때의 칩의 크기 형태를 나타낸다.

그림 2. 상이한 공정 기술간의 변환

그림 3. 동일한 공정 기술간의 변환

2. 레이아웃 이식 시스템의 구성과 고안된 알고리즘

레이아웃 이식 시스템 구성은 그림 4 에서 보는 바와 같이 크게 레이아웃 입력 형식인 GDSII (Graphic Data Stream II)로부터 (1)소자추출 부분, (2)소자 및 배선의 크기 조정 부분, (3) 컴팩션(Compaction) 부분으로 구성된다. 소자추출 부분은 이식하고자 하는 GDSII, CIF 형식의 설계 데이터를 입력으로, 이를 분석하여 소자를 추출하며, 크기 조정 부분은 목적하는 설계 규칙과 공정 기술을 만족하고 타이밍과 소비전력을 최적화할 수 있도록 소자 및 배선의 크기를 조정한다. 컴팩션 부분은 최적화 된 소자와 배선으로 구성된 레이아웃 데이터를 최소의 면적을 얻을 수 있도록 컴팩션 기능을 수행한다. 또한, 컴팩션 부분에서는 반도체 기술이 0.25μm 이하의 VDSM(Very Deep Sub-Micron)으로 심화됨에 따라 발생하는 문제점들 즉, 배선간의 간섭현상(Crosstalk), 안테나 효과(Antenna Effect), 광학적 근접효과 보정 등에 대한 분석기능을 제공한다.

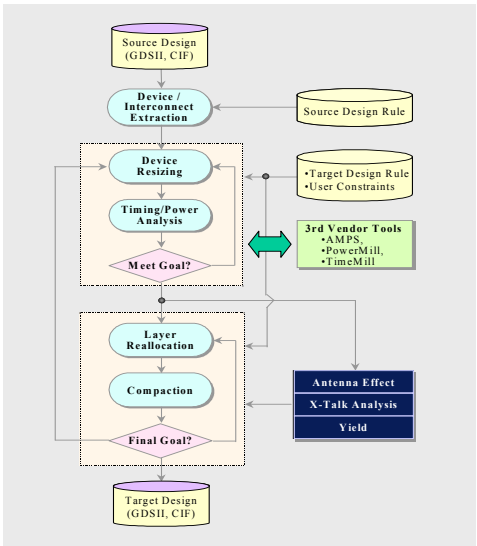

그림 3. 레이아웃 시스템의 구성

2.1. 전처리 기능과 알고리즘

전처리 기능은 레이아웃 설계 데이터(GDSII, CIF 등)를 처리하여 내부 DB 구조로 변환하는 입력데이터 파싱 부분, 내부 DB 를 분석하여 계층구조를 생성하는 부분, 이를 필요에 따라 평면구조(Flatten)로 변환하는 부분, 각 레이어(layer) 들을 구성하고 있는 도형(Polygon) 들을 병합하고 4 각형(Trapezoid)으로 변환하는 합병과 분할(Merge & Cut) 부분 등으로 구성되며, 이와 같은 일련의 작업은 소자추출을 위한 준비작업이라고 할 수 있다.

특히, 내부 데이터를 4 각형의 형태로 변경하는 부분은 생성된 자체 고유 데이터의 계층구조를 평탄화하고 스캔라인 (Scan-line) 알고리즘을 적용하기 위하여 순서적으로 배열 한다. 동일 레이어 상에 있는 폴리곤들 중 접촉 또는 겹쳐진 도형들을 동일한 도형으로 합병하고, 합병된 도형을 4 각형 형태로 분할한다. 이와 같이 분할된 4 각형들에 같은 번호를 부여하여 동일한 폴리곤의 일부임을 명시하도록 한다(polygon numbering 알고리즘).

이와 같이 생성된 데이터가 소자 추출에서 사용되는 패턴 연산의 입력 데이터로 사용된다.

2.2 소자 추출 및 크기 조정 기능과 알고리즘

소자 추출은 반도체 설계 방법 중 마스크 레이아웃 설계 및 검증에서 가장 핵심이 되는 영역중의 하나다. 최상위 단계인 동작 및 기능단계부터 회로 설계단계까지는 주로 다양한 시뮬레이션을 이용하여 검증이 이루어지며, 마스크 레이아웃 단계에서는 설계 규칙 검사(DRC), 레이아웃의 연결도(Connectivity)를 회로도면과 비교하는데 필요한 회로도 추출과 LVS(Layout Versus Schematic) 검증, 레이아웃으로부터 동작 및 타이밍 등을 검증하는데 필요한 LPE (Layout Parasitic Extraction)등이 있으며, 이들을 위해선 소자추출이 가장 기본이다. 마스크 레이아웃은 여러 개의 계층(Layer)으로 이루어지며 이들 각각은 도형들로 구성된다. 이들과 사용자가 지정한 레이어들간의 관계를 이용하여 소자를 추출하게 되는데, 일반적인 소자의 종류로는 트랜지스터 (MOS, Bipolar), 캐패시터, 저항, 다이오드 등이 있다. 특히 트랜지스터는 여러 개의 레이어로 구성되어 있으며, 이들 레이어들간의 패턴연산(AND, OR, NOT)을 수행하여 추출하게 된다. 본 연구에서는 이식 기술의 핵심인 컴팩션에서 이와 같이 추출된 소자를 활용하기 위하여 소자의 크기정보를 추출하여 적용할 공정 기술에 적합한 소자크기를 계산하고, 그 결과를 컴팩션에서 각 객체들간의 제약조건으로 사용될 수 있도록 한다.

2.2.1 패턴 연산 알고리즘

소자추출에서 사용되는 패턴연산은 마스크 레이아웃 데이터를 분석 및 검증하는데 가장 기본적이며 핵심기능이라고 할 수 있다. 레이아웃 데이터는 여러 개의 레이어들로 구성되어 있으며, 이 레이어들간의 조합으로 다양한 소자들을 구성한다. 따라서 이 레이어들로부터 소자를 추출하거나 레이어의 특정부분을 추출하기 위하여 그림 4 의 AND, OR, SUB, XOR 과 같은 기본의 부울 연산(Boolean Operation)과 이를 조합하여 사용하여 수행된다. 이것 이외에 인접한 레이어들 사이에서 특정조건을 만족하는 폴리곤을 선택하는 기능과 도형의 크기를 일정하게 확대하거나 축소하는 기능이 필요하다.

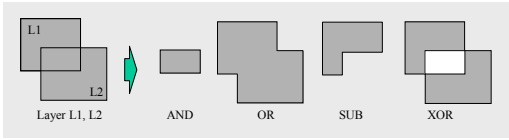

그림 4. 기본 폴리곤 연산

2.2.2 소자 추출 규칙

소자의 종류 및 형태는 능동 소자는 바이폴라, 모스트랜지스터이며, 수동 소자는 캐패시터, 저항, 다이오드로 구성된다. 실제 레이아웃 상에서는 많은 형태의 변형이 있을 수 있지만, 각 소자에 대한 기본 원칙은 변하지 않는다.

반도체 설계의 레이아웃 상에서 표현되는 다양한 형태의 소자들을 완벽하게 추출하기 위해서는 사용자의 역할이 중요하다. 사용자는 이미 정의된 형태의 패턴 연산자들을 조합하여 각각의 소자들을 추출할 수 있도록 포트 맵핑에 관한 규칙을 정의해야 하며, 이러한 입력 규칙들은 논리적인 오류가 없어야 한다. 예를 들어 MOS 트랜지스터의 경우, Substrate 를 제외하고 전기적 특성을 갖는 포트는 3 개 (게이트/소스/드레인)이며, 레이아웃 상에서 각 포트에 맵핑 될 수 있는 레이아웃은 그림 5 와 같이 표현 할 수 있다. 여기서, MOS 소자는 레이어 D 가 될 수 있으며, 게이트는 레이어 B, 소스와 드레인은 각각 레이어 C 로 맵핑 될 수 있다. 소자 추출 프로그램 상에서는 이렇게 만들어진 레이어 자료에 대해 맵핑 가능한 정보를 입력하면 조건에 맞는 데이터들을 추출하여, 정의된 형태의 소자로 출력하게 되는 것이다.

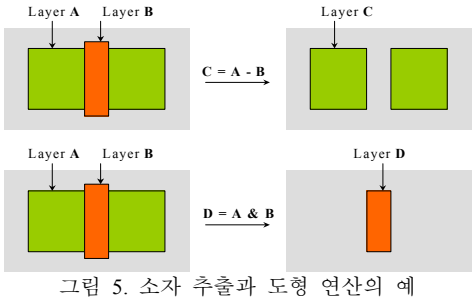

그림 5. 소자 추출과 도형 연산의 예

2.2.3 소자 크기 조정

소자에 대한 크기 조정은 앞 단계에서 추출된 소자와 연결 정보를 이용하여 적용 공정 기술과 설계규칙에 맞도록 소자의 크기를 계산하는 기능이다. 즉 트랜지스터로부터 폭과 길이 정보를 추출하여 얻어진 크기 정보에 대해 새로운 공정 기술을 적용시켜 새로운

값을 구하는 작업을 말한다.

3. 컴팩션 기능과 알고리즘

레이아웃 이식에서 컴팩션 부분은 전체 시스템의 성능을 좌우할 정도로 핵심이 되는 부분이다. 이에 대한 연구는 70 년대 중반부터 90 년대 초반까지 약 20 여년 간 활발히 수행되었으며, 일부 상용 툴들이 개발되었으나, 처리하는 용량과 성능의 한계로 인하여 만족스러운 결과를 얻지 못하고 있다. 초기 이 분야 연구의 주요 관점은 특정 공정 기술에 종속되지 않고 레이아웃 설계를 효율적으로 하는 것이었으며, 심벌릭 (symbolic) 레이아웃이 그 대상이었다. 그러나 최근 SoC 의 설계가 보편화되어, 하드 IP 의 중요성이 대두되면서 마스크 레이아웃이 컴팩션의 대상이 되고 있다. 이러한 도형(polygon) 형태의 레이아웃을 직접 컴팩션에 적용하려면 레이아웃을 심벌들로 구성된 레이아웃 형태로 변환하여 기존의 심벌릭 컴팩션 방법(symbolic compaction)을 적용하거나, 또는 도형의 각각 에지들을 직접 처리하는 에지 컴팩션 방법(edge-based compaction)을 사용한다. 심벌릭 컴팩션 방법은 처리하기가 비교적 간단하다는 장점이 있는 반면, 마스크 레이아웃으로부터 데이터의 손실 없이 모든 정보 (트랜지스터, 접점, 메탈, ...)등이 심벌 형태로 변환할 수 있어야 된다. 에지 컴팩션 방법은 도형을 구성하는 각각의 에지들을 객체로 처리해야 하므로 데이터가 기하 급수적으로 증가하여 대부분 처리 용량의 한계에 직면하게 된다.

컴팩션 방법은 대상이 되는 객체의 형태에 따라 심벌릭 컴팩션과 에지 컴팩션으로 분류되는 것 이외에도 처리하는 방식에 따라 다양하게 분류된다. 첫째로, 컴팩션하는 객체의 움직이는 방향에 따라 1 차원 컴팩션 방법과 2 차원 컴팩션 방법으로 분류할 수 있다. 1 차원 방법에서는 객체가 X 또는 Y 방향으로만 움직일 수 있으며, X, Y 방향을 반복적으로 수행한다. 2 차원 방법에서는 객체가 동시에 X, Y 방향으로 움직인다. 또한 1 차원 컴팩션 기능을 수행하면서 부분적으로 다른 방향으로 움직이게 하는 1.5 차원 방법이 있다. 둘째로는 컴팩션하는 객체 사이의 최소 거리를 계산하는 방법에 따라 제약조건 그래프방식과 가상 그리드(Virtual Grid) 방식으로 분류된다. 제약조건 그래프 방식에서는 객체들 사이에 적용되는 설계 규칙과 같은 제약 조건들을 선형 부등식(Linear Inequality)으로 기술하며, 이것을 방향성과 가중치가 있는 그래프(weighted directed graph)로 모델링하고 해석함으로써 컴팩션을 수행한다. 가상 그리드 방식은 모든 객체들이 가상적인 그리드 선상에 있다고 가정하고, 이 그리드 선상에 있는 객체들을 고려하여 인접한 그리드 선 사이의 최소거리를 계산하여 축소를 수행하는 방법이며, 제약조건 그래프방법보다 알고리즘이 비교적 간단하여 수행속도가 빠른 반면 최상의 압축된 결과를 얻지 못한다. 이외에 회로의 계층구조를 고려한 처리방식에 따라 평면구조(Flat Mode)와 계층구조(Hierarchy Mode)방법으로 분류하기도 한다. 일반적으로 이들 방법들 사이에는 축소된 면적과 사용된 메모리, 처리 소요시간 간의 상호 관계가 존재한다.

기존의 방식은 제약조건 그래프를 이용한 1 차원 에지 컴팩션 방법을 사용하고 있으며, 심벌릭 방법은 레이아웃 설계를 효율적으로 하기 위해 마스크 레이아웃을 생성하기 전 단계에서 편집용으로 일부 사용되고 있을 뿐, 이식 방법으로는 거의 사용되지 않는 실정이다.

3.1 설계규칙 (Design Rule)

컴팩션 과정에서 만족되어야 하는 제약조건으로는 객체들간의 최소거리를 지정하는 거리에 대한 제약조건과 연결 상태를 유지할 수 있는 연결 도에 대한 제약조건으로 대표할 수 있고, 이 제약조건을 기술하기 위한 설계규칙으로 Width, Distance, Overhang, Overlap Rule 과 같은 4 가지를 지원하며, 이에 대한 기술 형식은 다음과 같다.

(*RuleType LayerDefinition* [*LayerDefinition*] *MinValue*)

RuleType 은 위의 4 가지 규칙 형식을 지정하는 것이며, *LayerDefinition* 으로는 단순한 레이어 이름을 지정하거나, 부울 (AND, OR, NOT) 연산과 레이어 지정으로 표현할 수 있다.

(*LayerName* Op *LayerName*).(*LayerSelection*)

Width Rule 은 특정한 레이어에 최소 폭을 지정하는 설계 규칙이며, Distance Rule 은 동일한 레이어 또는 다른 레이어 사이에 대한 최소 거리를 지정하는 설계규칙이다. 또, Overhang Rule 은 하나의 레이어가 다른 레이어에 걸쳐있을 때 최소거리를 지정하는 설계규칙이며, Overlap Rule 은 두 레이어가 중첩되어 있을 때 최소 중첩거리를 지정하는 설계규칙이다.
위 4 가지 설계규칙은 부울 연산을 통하여 하나의 Width Rule 로 표현될 수 있으며, Width Rule 에 대한 제약조건 그래프 생성 기능만으로 다른 설계 규칙들을 처리할 수 있다. 또한, 다른 레이어 간의 설계규칙을 부울 연산을 통하여 한 레이어로 통합함으로써 처리하는 에지 간에는 서로 교차하지 않는 특성이 있으며 이는 제약조건 그래프 생성을 간단하게 한다.

3.2 제약조건 그래프 생성

컴팩션에서 일반적으로 사용되는 방법인 제약조건 그래프 방식은 그래프 생성이 가장 많은 시간을 소모하는 부분으로 많은 연구가 수행되었다. 그래프 생성 방법 중 간단한 방법은 설계규칙에 의한 제약조건이 있는 지를 모든 객체 쌍에 대하여 비교하는 것으로 비교 횟수는 $O(N^2)$가 된다. 이 비교 횟수를 줄이기 위한 방법으로 그림자 투사(shadow propagation)과 평면처리(Plane Sweep)이 있다. 그림자 투사 방식은 도형(polygon)의 그림자를 컴팩션 방향으로 전달 시키고 도형들간의 관계를 찾아내어 제약조건을 생성하는 방법이다[4,5,11]. 즉 하나의 도형에 가상의 빛을 비추

었을 때 그로 인하여 생기는 그림자를 차단하는 도형들과 제약조건을 생성하면 되며 비교 횟수는 $O(N^{1.5})$이다. 직각 평면 처리(Perpendicular-Plane-Sweep) 방법은 컴팩션하고자 하는 직각 방향으로 평면 처리를 적용하는 방법으로 도형이 스캔라인에 들어올 때와 나갈 때 제약조건을 검색하면 되며 비교횟수는 O(NlogN)이다. 그림자 투사와 평면 처리를 접목시키고, 처리 속도를 향상시킨 개량형 PPSS (Parallel Plane Sweep Shadowing) 방법은 축소하고자 하는 방향으로 평면 처리를 적용하는 방법으로 각 도형을 한번만 처리하면 되며, 벤치마킹에서 보인 것과 같은 월등한 처리 속도와 효율을 보이고 있다. 또한, 본 연구는 에지 컴팩션 방식과 개량형 PPSS 를 구현하고, 임의의 각도를 갖는 에지를 처리할 수 있도록 확장하였으나 현재는 45 도 에지까지만 지원한다.

```
(a)  Yqueue = Sort_edges_in Y_direction();
(b)  Sfront = 0 ;  Dfront = 0 ;
(c)  while ( Yqueue is not empty )
(d)     Nedges=get_scan_in_edge(Yo,NextYo,
                  Yqueue);
(e)       foreach ( E in Nedges )
(f)           update Sfront list using  E
(g)           and generate constraints
                among the interacting ones
(h)           if( E is not horizontal edge )
                push E into Dfront list;
(i)
(j)       if ( Dfront list is not empty )
(k)           do_oblique_shadowing ( Yo, NextYo,
                  Dfront, Sfront )
(l)
```

그림 6. 컴팩션 알고리즘

그림 6 의 구현 알고리즘은 기본적으로 PPSS[11]를 응용하였으며, Y 방향 컴팩션을 기본으로 작성되었고 (h,j,k)는 45 도 에지를 처리하기 위하여 추가된 부분이다. 알고리즘에서 Yqueue 는 에지의 정렬된 리스트를 저장하는 변수이다. Sfront 는 Shadow Front List 를 저장하기 위한 변수이며, Dfront 는 Shadow Front List 에 있는 45 도 에지 중 현재 스캔라인에 걸쳐있는 에지 리스트를 가지고 있는 변수이다. Yo 와 NextYo 는 현재와 다음 스캔라인의 Y 좌표를 가지고 있는 변수이다. (a)는 에지의 양 끝점 중 최소값의 Y 좌표(Ymin)를 기준으로 정렬을 하여 리스트 Yqueue 에 저장한다. (d)는 Yqueue 에 있는 에지를 참조하여 현재와 다음 스캔라인의 Y 값을 결정하고, Yo 에 걸쳐있는 에지들을 Nedges 에 저장한다. (f,g)는 설계규칙을 고려하여 제약조건 그래프에 제약조건을 등록한다. (h)는 현재의 스캔라인에 들어오는 45 도 에지를 Dfront 리스트에 등록하는 부분이다. (k)는 현재 스캔라인에 걸쳐있는 45 도 에지, 즉 Dfront 리스트에 등록되어 있는 에지에 대하여 투사을 수행하는 과정이며, 이때 다음 스캔라인 (NextYo)에서 나가게 되면 Dfront 리스트에서 삭제한다.

위 알고리즘을 그림 7 에 적용하여 설명하면, Scan-line 이 y0 에서 모든 과정을 마치고 나면 각 변수들은 다음과 같은 값을 가지게된다.

Yo = y0, NextYo = y1, Sfront = (E2, E3),
Yqueue = (E4, E5, E6)

다음 알고리즘의 (d) 연산에 의하여 변수 값들은 다음과 같이 변하게 된다.

Yo = y1, NextYo = y2, Sfront = (E2, E3),
Nedgs = (E4, E5) , Yqueue = (E6)

다음 알고리즘 (e-i) 연산에 의하여 Sfront =(E4, E5, E3)로 변하며, 제약조건 그래프에 설계규칙 Width 와 Distance 에 해당하는 Arc A2, A3 이 추가된다. 이와 같은 과정을 스캔라인을 증가 시키면서 반복하면 제약조건 그래프가 생성된다.

그림 6. (k)의 45 도 Edge 에 대한 투사는 현재 스캔라인 (Yo)과 다음 스캔라인 (NextYo) 사이에 있는 45 도 에지, 즉 Dfront 리스트에 대하여 투사를 수행하는 것이며, 이때 각 에지의 위상 순서(topology order)를 고려하여 순서에 맞게 처리해야 한다. 예를 들어, 그림 7 의 (a)를 보면 Edge B 가 Edge A 의 아래에 위치함으로 B 를 먼저 처리해야 하고, (b)(c)는 Edge 들 사이에 우선순위 관계가 없으며, (d)는 Edge A 를 먼저 처리해야 한다. (e)의 경우에는 Edge B, C, A 순으로 투사가 수행되어야 하며, 에지 B, C 는 다음 스캔라인과 교차하지 않으므로 Dfront 리스트에서 삭제된다. 이러한 우선 순위는 Dfront 리스트가 X 좌표에 의하여 정렬되어 있을 경우, 스택 연산에 의하여 간단히 구현될 수 있다.

45 도 에지에 대한 일반적인 처리 방법은 각 에지들을 위상순서에 맞게 정렬을 한 후 투사를 수행하는 것이나, 위에서 적용한 알고리즘은 간단히 에지의 Ymin 값을 기준으로 정렬을 하고, 스캔라인에 교차한 에지에 대해서 만 우선 순위를 고려하여 처리함으로 전체적인 정렬 시간을 줄일 수 있으며, 45 도 에지의 수가 작을 경우 특별한 부가 작업이 없이 처리할 수 있다. 또한, 에지의 교차를 허용하는 경우에는 위상 순서가 더욱 복잡하게 되며, 이러한 경우 위 알고리즘을 확장하면 적절히 활용할 수 있다.

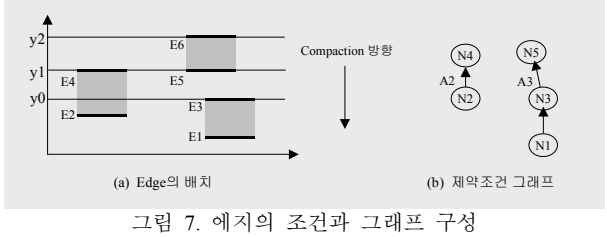

그림 7. 에지의 조건과 그래프 구성

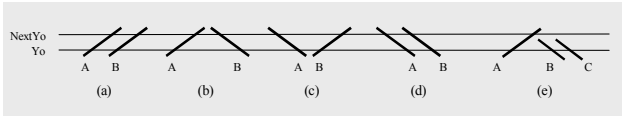

그림 8. 45 도 에지의 배치

3.3 제약조건 그래프 해석

제약조건 그래프 해석은 가장 긴 경로 탐색, 조그(Jog) 삽입, 배선길이 최소화 과정으로 분류된다[11].

가장 긴 경로 탐색은 제약조건을 위반하지 않는 각 도형들의 최소 위치를 결정하는 과정으로 가장 긴 경로를 임계경로라고 한다. 이때 사용자 제약조건 등으로 인하여 가장 긴 경로를 찾을 수 없는 경우가 발생하며, 이는 그래프에 사이클(positive cycle)이 존재하기 때문이고 이를 찾아내어 해결하는 것이 중요한 관건이 된다. 이에 대한 방법으로는 그래프에 사이클이 없을 때 적용 가능한 Bellmann & Ford 방법[6], 사이클이 있을 경우 Event-driven 기법을 사용하여 수렴할 때까지 반복적으로 수행하는 Liao & Wong 방법[6], 점증적인 방법을 이용하여 사이클이 발견되는 즉시 해법 과정을 수행하는 확장된 DBR 방법[6] 등이 있다

조그 삽입은 그래프 해석시 사이클의 해법 과정과 면적 최소화 과정에 이용될 수 있다. 조그를 이용한 면적 최소화는 임계경로 상에 있는 배선에 조그를 삽입하고 새로운 임계경로 해석을 수행하며 이 과정을 조그를 삽입할 공간이 없을 때까지 반복하는 것이다.

배선길이 최소화는 경로 탐색 과정에 의하여 컴팩션 방향으로 과도하게 이동한 도형들을 제약조건을 위반하지 않는 범위 내에서 배선길이가 최소화 되도록 컴팩션 반대 방향으로 이동시키는 과정이며, 이때 회로의 성능과 제조공정에서의 수율을 고려하여야 한다.

3. 실험 결과

본 연구에서 구현한 컴팩션 기능의 효율성을 검증하기 위하여 상용 S/W 와 비교하여 실험하였다. 벤치마킹에 사용된 시스템은 SUN SPARC 20, Solaris 2.5.1 이며, 트랜지스터의 수가 300 ~ 50000 여개의 1.0 ㎛ 공정 기술로 설계된 회로에 0.6 ㎛ 공정의 설계규칙을 적용하여 Y 방향, X 방향 컴팩션을 순차적으로 수행하였다. 표 1 은 컴팩션된 면적, 수행 시간, 사용된 메모리를 비교한 것이며, 면적 차이는 1% 미만으로 거의 동일한 결과를 얻을 수 있는 반면, 수행 속도에는 평균 27.4 배의 차이가 나며, 메모리 사용량은 경쟁 제품(표의 Other 에 해당)에 비해 평균 41%의 메모리를 사용하였다. 따라서, 회로 6 과 7 인 경우에는 메모리 overflow 로 인하여 수행이 불가능하여 'X'로 표시하였고, 이는 대형의 반도체 도면을 처리 할 수 없기 때문에, 본 연구의 경쟁 우월의 근거가 된다[9, 10]. 또, 경쟁 제품의 경우 전체 수행시간의 80% 이상을 제약조건 그래프 생성에서 소모하는 것으로 보아, 본 연구의 제약조건 그래프 생성 기능이 월등한 성능을 보임을 알 수 있다.

회로	Tr.의 수	면적(mm^2)		처리속도 (초)		
		결과	Other	결과	Other	比率
1	374	17,165	16,987	8	230	27.4
2	1,792	88,903	89,277	55	1,526	27.7
3	2,453	62,198	58,596	123	3,174	25.8
4	4,315	237,455	235,065	201	5,628	28.0
5	25,908	1,435,284	1,431,544	1,960	54,609	27.9
6	38,862	2,289,945	X	3,266	X	X
7	51,816	2,866,973	X	6,069	X	X
평균						27.4

4. 결론

지금까지 수행된 연구에 대해서 기술하였다. 앞에서도 설명하였듯이 레이아웃 이식은 마스크 레이아웃과 관련된 다양한 기술들로 구성된 복합기술이라고 할 수 있다. 즉, 반도체 설계 기술 중 가장 핵심부분 중의 하나인 검증기술의 기반기술이라고 할 수 있는 레이아웃 분석 및 처리 기술과 소자추출 기술, 그리고 컴팩션 기술 등이 이식 시스템의 근간을 이루고 있는 기술이다. 특히 컴팩션 기술은 그 중에서도 이식 시스템의 성능과 질을 좌우하는 핵심 부분이며, 실험 결과에서 보여 주듯이 현재 상용 소프트웨어와 벤치마크 테스트를 해본 결과 평균 20 배의 처리 속도와 1/3 이상의 메모리 사용량이 적은 결과를 얻었다. 동일한 조건에서 공정 기술의 최소 설계 규칙을 적용하였으므로 면적은 거의 유사하지만, 수행속도(성능)나 메모리 사용측면에서 월등한 결과를 얻었다. 최적화가 되지 않은 기본 프로토타입의 비교임에도 불구하고 이와 같은 결과를 얻었다는 사실만으로도 충분한 경쟁 가능성이 있다고 사료된다. 향후 마스크 레이아웃 이식에서 가장 문제시 되고있는 처리 용량의 한계를 극복하기 위하여 계층구조를 효율적으로 처리할 수 있는 연구개발이 중점적으로 진행될 예정이다. 이외에 VDSM 공정기술에서 발생할 수 있는 다양한 문제점들, 즉, 배선간의 상호 간섭현상(Crosstalk), 안테나 효과(Antenna Effect), 제조장비의 광학적 근접효과 보정(OPC: Optical Proximity Correction)의 요인들을 레이아웃 이식 시스템에 반영될 수 있도록 계속적인 연구가 될 것이다.

참고문헌

[1] SIA, "The National Technology Roadmap for Semiconductors", 1998

[2] Mike Keating, "Financial Model for Design Reuse" Synopsys Inc., 1997

[3] J. McLeod, "The State of the Art in IC Layout Migration," Integrated System Design Magazine, 1996.

[4] Y. E. Cho, "Subjective Review of Compaction," Proc. of the 22nd ACM/IEEE DAC, 1985.

[5] J. Fang, et. al. "A New Constraint Graph Generation Algorithm for VLSI Layout Compaction," Proc. ISCAS. pp.2858-2861, 1991.

[6] J. F. Lee and C.K.Wong, "A Performance-Aimed Cell Compactor with Automatic Jogs," IEEE Tr. on CAD, pp.1495-1506, 1992.

[7] D. Marple, "Transistor Size Optimization in the Tailor Layout System," Proc. of the 26th ACM/IEEE DAC, 1989.

[8] F. N. Najm, "A Survey of Power Estimation Techniques in VLSI Circuits," IEEE Tr. on CAD, 1994.

[9] DREAM User's Guide, V.2.6. 1998.

[10] LACE MEGASTAR User's Guide, Version 2.1, 1997.

[11] Ilhami H. Torunoglu, Murat Askar, "FAST CONSTRAIONT GENERATION ALGORITHMS FOR VLSI LAYOUT," pp. 577-580. DAC, 1994.