FRINGE를 이용한 자동배선에 관한 연구

(A Study on the Automatic Routing Using FRINGE)

朴 魯 京*, 全 興 雨*, 車 均 鉉*

(Nho Kyung Park, Heung Woo Jun and Kyun Hyon Tchah)

要 約

본 논문에서는 Fringe를 이용한 자동배선 알고리즘을 연구한다. 자동배선 알고리즘의 연구에 사용한 Fringe라는 개념은 매우 간단하지만, 단층배선을 행하는데 유용한 수단이다. 본 논문에서 연구한 자동배선 프로그램은 C언어로 작성하였으며, VAX 11/750과 IBM-PC AT로 실행하였다. 그리고 프로그램을 몇가지 예에 적용하여 배선 알고리즘의 유용성을 보였다.

Abstract

In this paper, an automatic routing algorithm using Fringe is described. The concept of the Fringe is very simple, but it provides a powerful and flexible tool for doing single layer routing. The automatic routing program is coded using C language, and implemented on a VAX 11/750 and IBM-PC AT computer. The usefulness of the routing algorithms is shown by the execution of the program on some examples.

I. 서 론

반도체회로의 집적도가 증가함에 따라서 칩안에 있는 셀들의 상호 배선문제는 회로의 설계시간과 칩면적의 증가를 가져왔다. 일반적으로, 전체 설계시간의 30%, 칩면적의 60%가 셀들의 상호배선에 소비된다.[1]

그러므로, 집적회로의 설계에 있어서 중요한 문제는 배선이 점유하는 면적을 최소로 하면서, 원하는 상호배선을 정확하게 실현하는 것이다. 그리고 집적도의 증가에 따른 회로의 설계시간을 단축하기 위한 설계자동화의 필요성에 따라서, 배선이 자동적으로 실현되도록 하는 자동배선 프로그램의 개발이 필요하다. 이와같이 연결할 점들 사이의 정확한 배선을 효율적으로 하기 위하여 현재 여러가지 배선알고리즘[2,3,4]이 연구개발되었으며, 배선을 자동화하는데 실제로 응용되고 있다. 본 논문에서는 배선간의 상호교차가 없고, 설계규칙에 어긋나지 않도록 하는 자동배선 프로그램을 개발하는데 있어서 Fringe라는 개념[5]을 사용하였다. Fringe는 배선을 하거나, 불규칙적인 셀의 경계를 기술하는데 있어서 효율적인 새로운 기법으로서 점들의 linked-list로 표시된다. 또한, 배선의 자동화에 있어서 배선의 순서를 결정하는 것은 매우 중요하다. 본 논문에서는 배선이 행해지는 순서를 결정하는데 있어서 quick-sort 알고리즘,[6] back-ward block-reverse 알고리즘[5]을 이용하였다. 그리고, 배선이 최대한 짧게 구성되도록 하고, 배선을 할 때 생길 수 있는 불완전 배선을 제거하도록 하는 자동배선 변환 처리기능을 부가하였다. 이와같은 방법들을 이용하여 개발된 자동배선 프로그램을 River routing, U-connection, Corner-connection에 대하여 확인하여 보았다.

*正會員, 高麗大學校 電子電算工學科
(Dept. of Elec. & Comp. Eng., Korea Univ.)

接受日字 : 1986年 9月 5日

II. Fringe의 정의

Fringe는 배치된 셀의 경계정보를 기술하는데 있어

서 효율적인 기법으로서 Fringe의 형태 및 그와 관련된 경계함수의 정의는 다음과 같다.

Fringe는 경계함수 f(x)를 표시하기 위하여 사용된 점들의 유한수열 (x1, y1) (x2, y2) … (xn, yn)로서 linked-list[7]로 표시된다. 여기서, 경계함수의 정의는 다음과 같다. 설계규칙에 의하여 셀들로부터 간격d 만큼 떨어진 영역내에서는 배선이 될 수 없도록 하면, 그 내부에 있는 점 (x, y)에 대하여 y<f(x)가 되도록 하는 최소함수 f(x)가 존재한다. 이때, 이러한 함수 f(x)를 경계함수라 정의한다. 그런데, 배치된각 셀들의 아랫변이 무한대까지 확장된 것으로 가정하면, 배선을 할 때 피해가야할 셀에 대한 정보는 셀에 대한 4변중 상측변의 좌표를 나타내는 x좌표의 하한값 xlow, 상한값 xhigh와 y좌표의 상한값 yhigh의 3개의 값 (xlow, xhigh, ylow)으로 표시할 수 있다. 이와같이 표시되는 3개의 값들을 경계구획(boundary segment)이라 한다.

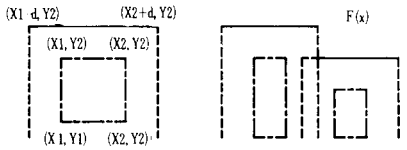


그림 1. 경계구획. 경계함수
Fig. 1. Boundary segment, function.

그러므로 xlow<x<xhigh이면 경계구획은 x를 포함하여 경계함수 f(x)는 x에 대한 모든 경계구획의 최대높이가 된다. 그림 1에서 실선은 경계구획 및 경계함수f(x)를 나타낸다. 그리고 list 및 Fringe의 형태는 다음같은 structure type를 갖는다.

```
typedef struct-list          typedef struct
  {point *car;                {byte orint;
   struct-list *car;}          list points;}
    *list;                     fringe;
```

이때, Fringe는 아래와 같은 특징을 갖고서 구성된다.

1) 수열은 x가 증가하는 순서로 구성된다.
   (xi<=xi+1)
2) 수열은 경계함수 f(x)가 값을 변경할 수 있는 위치를 표시한다.
3) xi<x<xi+1이면, f(x)=yi이다.
4) x<=x1이면, f(x)=—무한대 이다.
5) xi=xi+1이면, f(x)=min(yi, yi+1…yi+k)이다.

이와같은 특징을 갖도록 구성되는 Fringe 경계구획을 한번에 하나씩 부가하고, 중첩이 있는 곳에서는 상측의 경계구획이 하측의 경계구획을 포함하도록 함으로써 구성한다. 또한, 그 순서는 셀들이 배치되는 순서에 의하므로 Fringe의 구성방법은 모든 셀들이 개별적으로 분류되거나, 미리 분류될 필요가 없이 셀들이 배치되는 대로 Fringe에 부가 될 수 있으므로 그들이 부가된 순서는 중요하지 않다.

그리고, Fringe는 linked-list로 표시되므로 경계구획을 부가하는 것은 insertion sort[8]의 one insertion이다. 따라서, Fringe를 구성하는 것은 O(n**n) (n은 경계구획의 수) 시간이 걸린다. 그림 2에서 Fringe는 (a, b, c, d, e, f, g)이다. 이와같은 Fringe의 개념은 간단하면서도, 자동배선을 하는데 유용하게 응용될 수 있다.

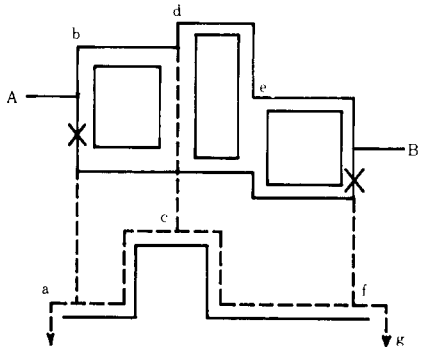


그림 2. 단일방향 회피
Fig. 2. One-sided avoidance.

Ⅲ. 배선 알고리즘

본 논문에서는 자동배선을 하거나, 불규칙 셀의 경계 정보를 기술하는데 있어서 Fringe의 개념을 이용하였다. 그러나 이 개념을 이용한 배선을 다음과 같은 2가지의 제약조건을 갖는다고 가정하였다.

1) 배선은 서로 교차하지 않는다.
2) 배선은 배치된 셀들의 상측으로만 지나감으로써 셀들을 피한다.

즉, 단일 layer(metal, poly, diffusion)로만 배선이 행해지고, 배치된 각 셀들의 아랫변이 무한대까지 확장된 것으로 가정함으로써 그림 2의 실선으로 배선이 형성된다.

이러한 조건하에서 Fringe를 이용하여 배선을 자동적으로 하기 위하여 여러가지 알고리즘이 필요하다. 이들의 기능은 Fringe의 변형, gap이나 U형태의 제거, 연결될 포트의 변형, 배선순서 결정등으로 분류될 수 있다.

1. 기본 Fringe의 설정

배선이 피해가야 할 셀의 지정은 각 layer가 배치될 수 있는 최저위치를 Fringe로 나타낸다. 즉 배치된 셀들이 기본 Fringe를 구성한다. 이때, Fringe의 구성은 셀들이 배치되는 순서에 관계없이 그 정보가 Fringe에 부가된다. 그리고, 배치된 셀들을 배선이 회피하는 방향은 배선의 연결상태에 따라서 적당히 선택하여야 한다. 즉 U-connection의 경우는 회피 방향이 +y, Corner connection의 경우는 회피방향이 +x, +y, −x, River routing의 경우는 회피방향이 +x, +y, −x, −y 이다. 회피방향에 따른 배선형태는 그림 3 와 같이 서로 다른 형태로 나타난다.

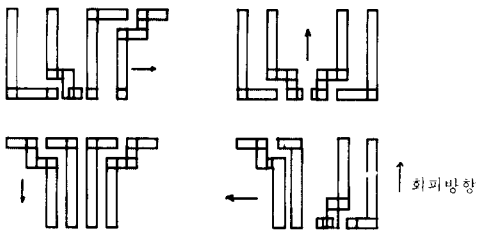


그림 3. 회피방향에 따른 River-routing
Fig. 3. River-routing considering avoidance direction.

2. Port의 지정 및 변환

Port는 points의 1 차원 배열로서 관련된 정보를 포괄하는 structure type를 갖는다. 이때 배선에 관련된 정보는 배선의 폭, layer의 명칭, 연결병행등을 포함한다. Port의 구조와 포트의 지정은 아래와 같은 형태로 실행된다.

```
TYPEDEF STRUCT-PORTS
        {   STRUCT-PORTS * NEXTPORT;
            POINT    *ARRAY;
            INT      WIDTH;
            SHORT    NUM;
            SHORT    LAYER;
            SHORT    PORTNAME;
            BYTE     DIR;          }
DefinePort  〈name〉 〈argument〉
  〈argument〉 : 배선방향(+x, −x, +y, −y)
                배선형식(Lwire, Cwire, Rwire)
                layer (Metal, Poly, Diffusion)
                Port번호 등
```

회피할 셀들에 대한 정보가 기본 Fringe로 구성된 후 연결할 포트들의 위치가 위와같이 지정되고, 그들은 연결될 배선의 변을 나타내는 점들의 쌍으로 변환된다. 이때 포트의 변환된 변이 기본 Fringe와 평행으로 배치되도록 하여 배선시 생길 수 있는 불완전한 사각형 형태의 hole을 제거하도록 한다. 이때, 불완전한 사각형 형태의 hole은 stub을 선정하여 부가하도록 한다. 즉 그림 4 에서 처럼 포트를 지정하고, 포트를 기준으로 하여 배선폭 만큼의 길이를 갖는 배선될 변을 나타내는 점들의 쌍으로 변환한다. 이때 변환된 변이 기본 Fringe의 방향과 수직이면 점선과 같은 hole이 생겨 불완전한 배선이 되므로 stub을 이용하여 이를 처리한다. 그리고, 포트의 방향을 배선될 방향으로 변환시킨다. 그림 4 에서 a, b는 연결될 포트를 나타내고, →은 포트의 연결방향이다. 이때 포트의 연결방향의 좌측으로 배선될 변이 확장된다.

불완전한 배선을 보강하는 알고리즘은 그림 5 와 같다.

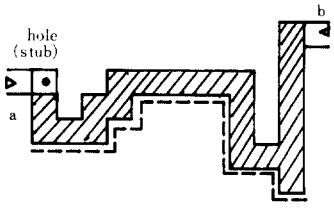


그림 4. 배선시 생길 수 있는 hole
Fig. 4. Hole occured when routing.

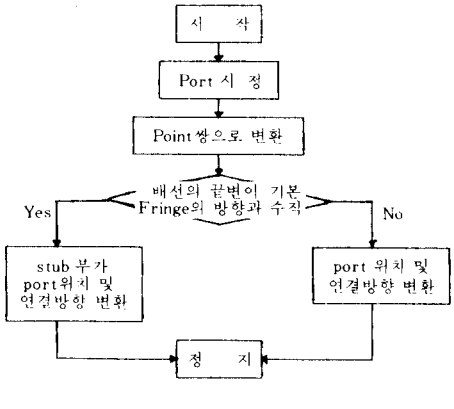


그림 5. Port 변환 알고리즘
Fig. 5. Algorithm for changing the port.

3. 배선순서의 결정

배선을 하는데 있어 그들이 서로 교차하지 않도록 하려면 그배선 순서의 결정이 매우 중요하다. 본 논문에서는 quick-sort와 reverse block 방법을 이용하여

배선순서를 결정하도록 하였으며, 배선순서결정 알고리즘은 그림 6 와 같다.

즉. U-connection과 Corner-connection에 대하여는 연결될 포트쌍의 좌측점들의 x값이 감소되는 순서로 분류하고, 동일한 x값을 갖는 경우에 있어서는 y값이 증가되는 순서로 분류하므로서 배선순서가 결정된다. River connection에 대하여서는 상기의 배선순서 알고리즘과 더불어, 한 배선의 좌측 끝점이 다른 배선의 끝점보다 높은 포트를 배선하는 그 순서를 역전시킴으로서 배선의 순서가 결정된다. 이때, 인접한 배선들이 x값에 대하여 충분히 떨어져 있으면 (xhigh(i) <xlow(i+1)) 본래의 순서를 따르도록 함으로써 배선순서가 결정된다.

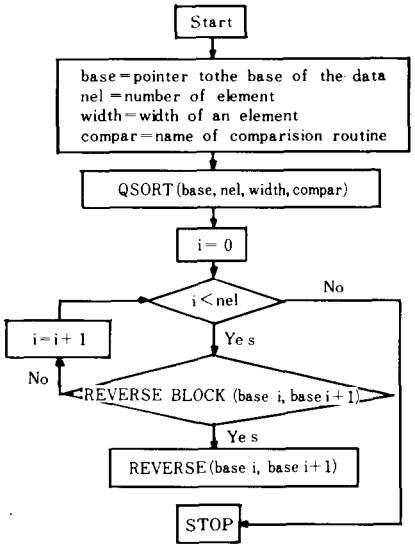


그림 6. 배선순서 결정 알고리즘
Fig. 6. Algorithm for sorting the routing order.

4. Fringe의 변경

Fringe가 배치된 셀들을 적어도 설계규칙에 의한 거리 d만큼 피하도록 Fringe를 변경하여야 한다. d를 위하여 사용되는 함수들은 Fringe를 지정된 셀로부터 d만큼 떨어지도록 변경하여 배선이 설계규칙에 맞도록 한다. 이와같이 변경된 Fringe를 이용하여 배선을 하기위하여 수평구획은(xi, yi)로 부터 (xi+1, yj+1) 까지 배선되도록 한다. 그런데, Fringe가 배선의 밑변을 지정하도록 하고, 배선폭이 연결포트의 좌변으로 확장하게 함으로써 배치된 셀과의 간격d를 항상 유지할 수 있어 배선폭의 지정은 간격d의 지정에 상관없게 할 수 있다. 그러므로 배선의 간격은 자동으로 처리된다. 그리고 특정포트들을 연결하기 위하여 그들이 부가되는데, 그들사이에 있지않는 모든 Fringe를없애는 기능이 부과된다.

5. U형태 배선 제거

Fringe의 정보를 따라서 배선을 할 경우에는 gap이나 U형태의 배선을 하므로, 가장 짧은 배선이 불가능하다. 그러므로 최소거리의 배선을 하기 위하여 이들을 제거하여야 한다. 이러한 것을 처리하기 위하여 본 논문에서는 stack, pop의 개념을 사용하였다. 이들은 gap 및 U형태를 추적하는데 사용되며, 아래와 같은 형태를 갖는다. 그리고, 이를 이용한 gap 및 U형태 제거 알고리즘은 그림 7 과 같다.

# define Push(x)
tmp=x→cdr, x→cdr=stack, stack=x, x=tmp

# define Pop(x)
tmp=x, x=x→cdr, Disc and (point, tmp→car), Discard(struct-list, tmp)

그림 7 의 알고리즘은 다음과 같은 동작을 한다.

1) Fringe의 다음 점이 stack의 top 보다 높고,stack의 top이 가장 높은 점이 아니면 stack의 top에 있는 점을 제거한다.
2) stack의 top이 Fringe의 다음점보다 더 높고, stack으로부터 적어도 하나의 점이 제거되었으면, Fringe의 다음 점을 제거된 마지막 점의 왼쪽으로 이동시킨다.
3) stack의 top에 있는 점이 Fringe의 다음점 앞에 있는 것들과 비교하여 가장 높으면, 다음 점이 stack에 부가된다.

이때, gap이나 U형태가 제거된 Fringe는 stack 에

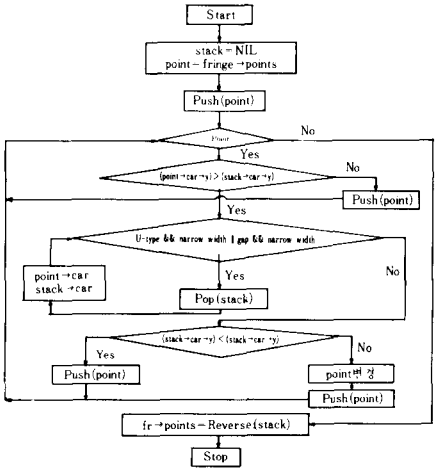


그림 7. U형태 제거 알고리즘
Fig. 7. Algorithm for U type removal.

있는 점들을 역전시킴으로서 구성된다. 이 알고리즘은 Fringe의 길이에 선형적인 시간이 걸린다. 그림 8은 각각 U형태를 그대로 둔것과 위의 방법으로 U형태를 제거한 것이다.

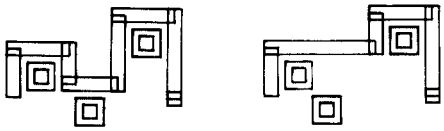


그림 8. U형태, U형태 제거
Fig. 8. U type, U type removal.

Ⅳ. 자동배선 프로그램의 구성 및 실험

본 논문에서 연구한 자동배선 프로그램에 의한 자동배선은 일반적으로 다음의 5 단계로 이루어진다.

1) 회피할 셀을 지정한다.
2) 연결될 포트들을 지정하고 각 포트를 배선의 끝면을 표시하는 점의 쌍으로 변환한다.
3) 배선순서를 결정한다.
4) Fringe 및 포트를 분석 처리한다.
5) 결정된 배선 순서대로 Fringe에 상응하는 배선을 한다.

이러한 동작은 Ⅲ장에서 설명한 알고리즘들의 조합에 의하여 행해지며, 자동배선 프로그램의 흐름도는 그림 9 와 같다.

본 자동배선 프로그램은 C언어를 이용하여 실현하였으며, VAX750과 IBM-PC AT에서 실행하였다. 그리고, U-connection, corner-Connection River-routing 등에 대하여 프로그램을 적용 실험하여 보았다. 이들은 폭 4μm를 갖고, 배선간 간격 4μm을 유지하면서 최소거리로 배선되도록 하였다.

그림 3 와 그림10는 이들에 대한 자동배선 결과로서 배선들의 회피방향에 따라 배선이 서로 다른 것을 알 수 있다. 그러나 설계규칙에 어긋나지 않고 최소거리로 자동배선이 이루어졌다. 그림11은 회피 방향을 +y로 하였을 때 연결된 배선 결과로서 9 개의 셀들이 배치될 때 14쌍의 포트를 연결한 결과로서 2.68초의 run time이 걸렸다. 그림12은 배선규모가 큰 실질적인 예로서 회피방향을 +y로 하였다. 그림12a는 10개의 셀이 배치되었을 때 50쌍의 포트를 연결한 예로서 2.87초의 run time이 걸렸고, 그림12b는 배치된 셀이 그림12a와 같고 100쌍의 포트를 연결한 예에서 3.76초의 run time이 걸렸다.

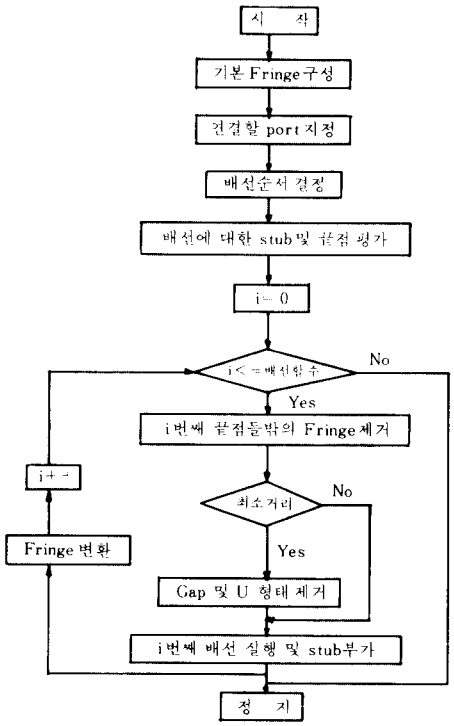


그림 9. 자동 배선 프로그램의 흐름도
Fig. 9. Flowchart of automatic routing program.

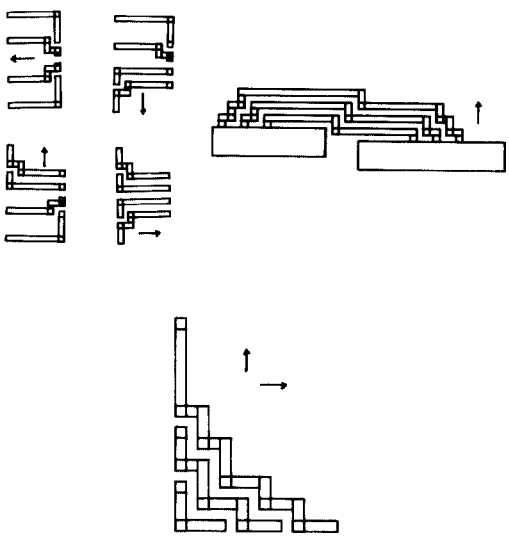


그림10. River-routing, coner-connection, U connection결과
Fig. 10. River-routing, coner-connection, U connection result.

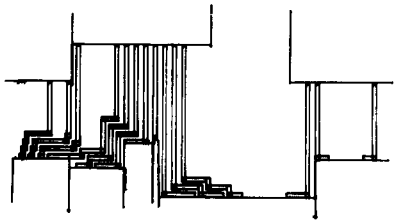
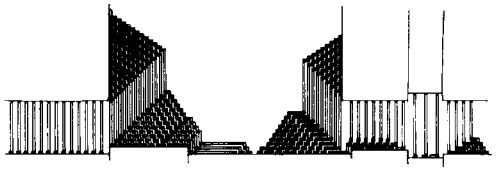
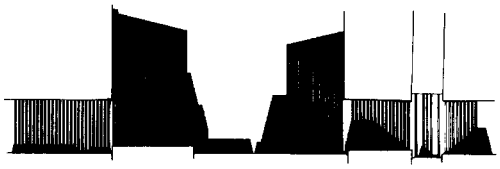


그림11. 간단한 배선 예
Fig. 11. A simple routing example.



(a)



(b)

그림12. A배선 예
b배선 예
Fig. 12. A routing example.
B routing example.

V. 결 론

본 논문에서는 자동배선을 하는데 있어서 Fringe라는 개념을 도입하여 자동배선 프로그램을 개발하였으며, 몇가지 배선을 실험하여 보았다. 개발된 자동배선 프로그램은 배선간에 교차가 없고, 배선간의 간격 및 회피할 셀들과 배선간의 간격이 설계규칙에 어긋나지 않도록 일정한 거리를 유지하면서 연결된 포트들을 자동적으로 배선됨을 보였다. 그리고, 그 배선의 길이는 최소의 길이가 되도록 처리된다. 본 논문에서는 회피할 셀이 12개이고, 배선수가 100쌍인 경우까지 River-connection, U-connection이 결합된 실질적인 배선을 실행하는데 3.76초의 시간이 걸렸다. 또한 River connection Corner-connection, U-connection과 그들의 조합으로 이루어진 배선도 처리할 수 있었다. 그리고 Fringe라는 개념은 배치된 셀의 경계정보를 기술하는데 있어서 효율적인 기법으로서, 자동배선 프로그램 개발에 상당히 유용하게 이용될 수 있음을 보였다. Gate Array 같은 것은 다층 채널배선을 사용하므로 Fringe를 이용한 자동배선은 그러한 설계형태에는 거의 영향을 주지 않을 것이다. 본 배선 프로그램 수행시 CIF가 출력되도록 하였으며, IBM-PC AT상에서 layout을 할 수 있는 YACC(yet another compiler compiler)를 이용한 DXF(draw exchange file) 변환 프로그램을 부가하였다. 따라서 다른 기종에 맞는 레이아웃 정보로의 변환이 매우 용이할 것이다.

參 考 文 獻

[1] Amar Mukherjee, *Introduction to nMOS & CMOS VLST systems Design*, Prentice Hall, pp. 306-308, 1986.

[2] Jeffery D.U11man, Computational aspects of VLSI, *computer science prass*, pp. 452-458, 1984.

[3] Ron Y.Pinter, River Routing, *Methodolgy and Analysis*, Bell Lab., 1983.

[4] J.soukup & J.C.Royle, On Hierarchical Routing, *computer scinence press*, 1981.

[5] Kevin Karplus, *CHISEL*, Stanford Unversity. pp.39-59. 1983.

[6] UNIX Programmer's Manual, *4.2 Berkery software Distrubution virtual VAX-11 version*, pp.579,1979.

[7] Ellis Horowitz, Sartaj sahnc, Fundamentals of computer Algorithms, *computer science press*, pp. 113-121. 1978.

[8] Patrick Henry, winston, Berthold Klaus paul Horn. *LISP*, Addison Wesley, pp. 131-150, 1984.