

論文97-34C-10-2

이차원 트랙 할당에 의한 FPGA 상세 배선

(A Detailed FPGA Routing by 2-D Track Assignment)

李征周 * , 林鍾錫 **

(Jung Joo Lee and Chong Suck Rim)

요약

FPGA에서는 기존의 다른 레이아웃 형태에서의 배선과는 달리 주어진 배선구조의 특징을 배선에 활용할 수 있다. 특히, Xilinx의 XC4000 계열의 FPGA에서는 그 배선구조의 형태를 고려할 때 배선문제가 이차원 트랙할당문제와 동일하다. 본 논문에서는 이러한 사실을 이용하여 이차원 트랙할당을 위한 휴리스틱 알고리즘을 개발하고 이를 사용한 새로운 FPGA 배선방법을 제안한다. 제안한 배선방법은 임의의 개략배선 결과를 입력받아 이를 토대로 각 연결 블록에서 필요로 하는 와이어 세그먼트의 갯수가 최소가 되도록 상세배선을 수행한다. 제안한 배선방법을 여러 시험회로에 대하여 시험한 결과 개략배선후 상세배선 하는 기준의 다른 배선방법들에 비하여 보다 더 작은 수의 트랙만으로 배선을 완료하였으며 직접 상세배선을 수행하는 배선방법과도 비교할만한 결과를 얻는다.

Abstract

In FPGAs, we may use the property of the routing architecture for their routing compared to the routing in the conventional layout style. Especially, the Xilinx XC4000 series FPGAs have very special routing architecture in which the routing problem is equivalent to the two dimensional track assignment problem. In this paper, we propose a new FPGA detailed routing method by developing a two dimensional track assignment heuristic algorithm. The proposed routing method accept a global routing result as an input and obtain a detailed routing such that the number of necessary wire segments in each connection block is minimized. For all benchmark circuits tested, our routing method complete routing with much smaller tracks than those by previously reported methods which use global routing results. The number of used tracks are also similar to the results by the direct routing methods.

* 正會員, 三星電子, 半導體 System LSI 本部, LSI2 事

業部 MICOM2 팀

(MICOM Team 2, LSI Division 2, Semiconductor
System LSI Business, Samsung Electronics)

** 正會員, 西江大學校 電子計算學科

(Dept. of Computer Science, Sogang University)

※ 본 연구는 서울대학교 반도체공동연구소의 교육부
반도체분야 학술연구조성비(과제번호 : ISRC 95-E-
2103)에 의해 수행되었습니다.

接受日字: 1996年8月7日, 수정완료일: 1997年9月22日

I. 서론

FPGA(Field-Programmable Gate Arrays)는 사용자가 구현하고자 하는 집적 회로를 칩 설계 및 제조 공정을 거치지 않고 직접 구현할 수 있는 기능을 가진 소자이다. 기존의 ASIC(Application Specific Integrated Circuits) 설계 방법인 표준셀 등을 시제품(prototypes)을 만드는데 많은 시간과 비용이 필요한데 비해 FPGA는 설계 후 시제품을 신속히 적은 비용으로 만들 수 있는 이점이 있을 뿐만 아니라 빠른 제작시가 때문에 설계사의 우류 박격과 이에 따른 재

설계 과정에 필요한 시간을 줄일 수 있어 ASIC 개발 기간을 단축하는데 기여할 수 있다^[3,13].

FPGA는 로직 블록(Logic Block(LB)), 입/출력 블록(Input/Output Block(IOB)) 그리고 배선자원 등으로 구성된다. LB와 IOB는 각각 필요한 논리 함수를 구현하기 위한 프로그래밍이 가능한 기본적인 회로로 구성되어 있으며, 배선 자원은 이러한 블록간의 연결을 위하여 이미 공정이 완료된 와이어 세그먼트(wire segment)들과 스위치(switch)들로 구성되어 있다(편의상 앞으로 “와이어 세그먼트”를 단순히 “세그먼트”라고 부른다). 그리고 각 블록은 미리 정해진 수의 입/출력 핀(pin)이 있으며 이들은 스위치를 통하여 세그먼트와 연결된다. 주어진 회로를 FPGA에 구현하기 위해서는 기술 매핑(Technology mapping), 배치(placement), 배선(routing) 등의 과정을 거친다^[3], 이중 본 논문에서는 배선 문제를 다룬다.

FPGA는 LB의 배열 방법과 배선 영역의 형태에 따라 여러 종류로 나뉘어지는데 본 논문에서는 LB가 이차원 배열 형태로 배치된 FPGA를 다룬다^[3,16,17]. 그림 1에 이러한 형태의 FPGA 구조를 보인다. 그림에서 보인바와 같이 FPGA의 내부에는 LB외에도 스위치 블록(SB)과 커넥션 블록(CB)이라는 배선자원이 있다. SB는 인접한 CB 사이에 존재하는데 이의 스위치를 통하여 서로 다른 CB 내의 두 세그먼트를 연결할 수 있다. 그리고 CB는 두 인접한 LB 사이에 존재하는데, LB의 각 핀은 스위치를 통하여 CB 내의 세그먼트와 연결된다. 실제 상용 FPGA에는 long line, double line 등 그 길이가 큰 세그먼트들이 있으나 본 논문에서는 이들을 고려하지 않는다.

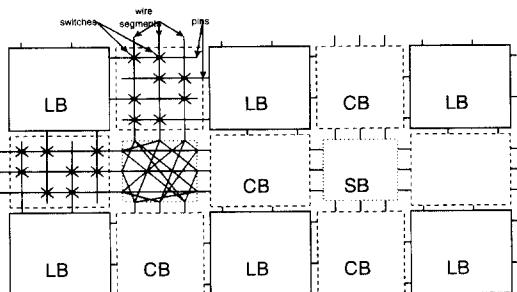


그림 1. 이차원 배열 형태의 FPGA 구조

Fig. 1. Two dimensional array type of FPGAs.

Brown 등은 이러한 FPGA 구조에서 SB와 CB의

특성을 규정하기 위하여 F_s 와 F_c 그리고 W 라는 세 개의 매개변수를 도입하였다^[3,4,5,6]. F_s 는 CB내의 각 세그먼트가 하나의 SB내의 스위치를 통하여 연결할 수 있는 다른 CB의 세그먼트들의 개수이고 F_c 는 LB의 한 핀에서 연결할 수 있는 CB내의 세그먼트의 개수이며 W 는 각 CB에 존재하는 세그먼트들의 수이다. 이러한 매개변수들은 FPGA의 배선에 많은 영향을 주는데 예를 들어 본 논문의 주된 목표인 Xilinx의 XC4000 계열의 FPGA는 $F_s = 3$ 이며 $F_c = W$ 이다. 따라서 이러한 경우에는 CB의 각 세그먼트는 인접한 다른 CB의 세그먼트 중 단 하나의 세그먼트와만 연결이 가능하며 LB의 각 핀은 해당 CB의 모든 세그먼트와 연결이 가능하다(그림 2).

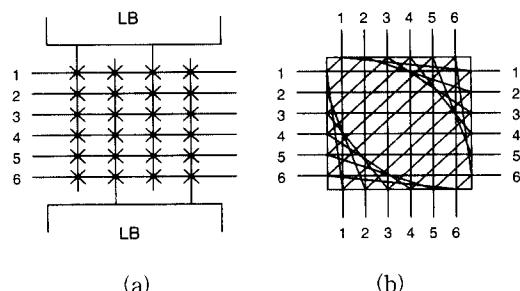


그림 2. XC4000 계열 FPGA의 연결 블록과 스위치 블록의 구조 (W 값은 편의상 임의로 설정하였다)

Fig. 2. The structure of the switch and connection blocks in XC4000 series FPGAs (W is arbitrary selected for convenience).

일반적으로 FPGA의 배선을 위하여 두 가지 접근 방법이 있다. 그 하나는 주어진 네트리스트의 각 네트를 FPGA의 배선구조에 직접 배선을 수행하는 방법이고 (“직접 상세배선”)^[1,10,15,18], 다른 하나는 각 네트에 대한 개략배선을 수행한 후 이를 토대로 배선을 수행하는 방법이다 (“개략배선후 상세배선”)^[5,11]. 전자의 경우에는 주로 미로 배선 기법^[9]을 적용하여 배선을 수행하며 후자의 경우에는 개략배선 결과를 FPGA의 세그먼트에 매핑하는 개념의 기법을 사용하고 있다.

한편, FPGA의 배선구조의 특징과 배선과의 관계를 연구한 결과도 있다. 즉, Wu 등은 $F_s = 3$, $F_c = W$ 인 경우 배선문제가 이차원 트랙할당문제와 동등함을 보이고 나아가 이 문제가 NP-hard임을 보

였다^[14].

본 논문에서는 이러한 이차원 트랙할당문제에 대한 휴리스틱 알고리즘과 이를 통한 FPGA 배선에 관하여 기술한다. 제안한 알고리즘에서는 이차원 트랙할당이라는 문제의 특성을 최대한 파악하여 “커넥션 그룹”이라는 세그먼트의 모임을 정의하고 이에 여러 우선순위를 도입하여 각 네트를 효율적으로 트랙에 할당한다. 제안한 방법을 기존의 여러 벤치마크 회로에 적용한 결과 개략배선 결과를 수정하지 않고 배선할 경우 대부분 이의 최대밀도 만큼의 트랙을 사용하여 배선을 완료한다.

지금까지 발표된 배선 결과를 살펴보면 개략배선후 상세배선 방법은 개략배선 결과에 의존하므로 직접 상세배선 하는 방법에 비하여 일반적으로 그 결과가 좋지 않다^[5,7,10,11,15]. 따라서 본 논문에서는 제안한 방법을 적용할 때 필요한 경우 개략배선 결과를 배선 중 수정하여 배선하는 방법을 아울러 제안한다. 이는 기존의 들어내기 및 재배선(rip-up and rerouting) 기법^[12]과 유사한데 이차원 트랙할당 중 최소 트랙에 할당을 완료하는데 방해가 되는 네트를 재배선 하여 필요한 트랙 수를 줄일 수 있도록 한다. 이러한 방법을 제안한 이차원 트랙할당 알고리즘과 조합하여 적용하였을 때 기존의 직접 상세배선 하는 방법과 비교하여 유사한 배선 결과를 얻는다.

본 서론에 이어 다음 2 장에서는 $F_s = 3$ 이고 $F_c = W$ 일 경우의 FPGA 배선문제를 이차원 할당 문제로 변환될 수 있음을 본 논문에서 제안하는 알고리즘의 설명을 위한 관점에서 간략히 기술한다. 그리고 3 장에서는 이차원 트랙할당 문제를 해결하기 위한 휴리스틱 알고리즘을 기술하고 4 장에서 이러한 알고리즘 적용시 수행할 수 있는 개략배선 결과의 재배선 방법을 설명한다. 제안한 배선 방법에 대한 실험결과를 5 장에서 보이고 마지막으로 6 장에서 결론을 맺는다.

II. 배선구조 형태에 따른 FPGA 배선

본 장에서는 FPGA의 배선 구조가 $F_s = 3$ 이고 $F_c = W$ 이며 CB와 SB가 그림 2에 보인바와 같은 구조를 갖는 경우에서의 배선 문제를 기술한다. 이와 같은 형태를 갖는 FPGA로는 Xilinx의 XC4000 계열

의 FPGA 등을 들 수 있는데^[16,17], 이러한 경우의 배선에 대한 자세한 문제 정의, 복잡도 분석 등은 참고문헌 [14]에 기술되어 있으므로 본 장에서는 제안한 알고리즘의 설명을 위한 수준 정도로 간략히 설명한다.

그림 2에 보인 배선 구조의 가장 큰 특징은 SB를 중심으로 인접한 세그먼트간의 연결 제약이다. 즉, 그림에서 보인바와 같이 CB의 각 세그먼트에 수평 CB의 경우 위에서 아래로 그리고 수직 CB의 경우 좌에서 우로 인덱스를 붙이면 CB의 각 세그먼트는 인접한 CB의 같은 인덱스를 갖는 세그먼트와만 SB의 스위치를 통하여 연결이 가능하다.

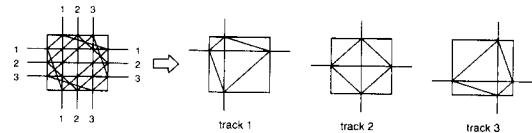


그림 3. $W = 3$ 일 경우 배선 구조의 특성

Fig. 3. The property of the routing structure when $W = 3$.

그림 3에 $W = 3$ 일 경우의 이러한 예를 보인다. 이러한 사실은 전체 배선 영역에 대해서도 마찬가지이다. 즉, 인덱스 k 를 갖는 하나의 세그먼트는 다른 CB의 같은 인덱스를 갖는 세그먼트와만 연결이 가능하다. 따라서 2-핀 네트의 경우 이를 배선하기 위하여 핀 하나에서 이 핀을 인덱스 k 를 갖는 세그먼트와 연결하였다면 나머지 배선에 인덱스 k 를 갖는 세그먼트만을 사용하여 다른 핀과 연결하여야 한다. 이는 FPGA의 전체 배선 자원을 동일한 인덱스를 갖는 세그먼트들과 이들에 관련된 스위치들로 구성된 W 개의 배선 자원의 그룹들로 분할하여 두 핀간의 연결을 이들 그룹중 하나를 선택하여 이에 속한 배선자원만을 사용하여 배선하는 것과 동일하다.

그림 4에 $W = 3$ 일 경우 분할된 배선자원그룹들을 보인다. 여기서 본 논문에서는 분할된 각 배선자원그룹을 ‘트랙’이라고 하고 인덱스가 k 인 세그먼트들과 관련 스위치들로 구성된 배선자원그룹을 ‘트랙 k ’라고 정의한다. 그림 4에서 보인 바와 같이 본 논문에서 정의한 트랙은 기존의 채널 배선 문제^[8,12,13]에서와는 달리 이차원 격자구조를 갖는다.

네트리스트 N 이 주어지고 이의 각 네트에 대한 개략배선 결과가 주어졌다고 하자. 여기서 편의상 N 의

각 네트는 두 개의 핀으로 구성되어 있다고 가정한다. 본 논문에서는 이러한 2-핀 네트를 ‘커넥션’이라고 부르며 문맥상 필요한대로 네트 또는 커넥션을 혼용하여 사용한다. 네트의 개략배선 결과는 어떤 CB를 사용하여 그 네트를 배선할 것인지를 알려준다^[3,5]. 따라서 이차원 격자구조를 갖는 트랙에서 보면 한 커넥션에 대한 개략배선 결과는 rectilinear line 형태를 갖는다.

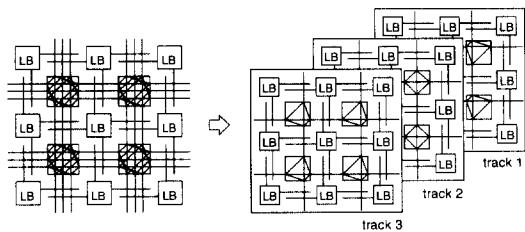


그림 4. FPGA 배선자원의 분할

Fig. 4. Partition of FPGA routing resources.

N 의 한 네트 $n_i = (s_i, t_i)$ 의 배선을 보자. 여기서 s_i 와 t_i 는 각각 n_i 의 두 핀을 의미한다. 앞에서 설명한바와 같이 s_i 를 이 핀이 속한 CB의 인덱스가 k 인 세그먼트에 연결한다면 n_i 의 배선을 위한 나머지 세그먼트들의 인덱스도 모두 k 이어야 하며 이는 배선을 위하여 단 하나의 트랙을 사용하여야 한다는 것을 의미한다. 따라서 만일 어떤 네트 n_i 의 개략배선 결과가 n_i 의 개략배선 결과와 일부 같은 CB를 사용한다면 두 네트의 배선은 이들의 개략배선 결과를 수정하지 않는 한 서로 다른 트랙에서 수행되어야 한다. 이러한 사실로부터 N 의 모든 네트에 대한 개략배선 결과가 주어지고 이러한 결과에 따라 배선(상세배선)을 수행하기를 원할 경우 우리는 이 문제를 다음과 같이 생각할 수 있다.

N 의 네트 n_i 를 트랙 k 에 배선한다는 것은 이의 개략배선 결과에 포함된 CB들의 인덱스가 k 인 세그먼트들을 사용하여 배선한다는 것이다. 따라서 우리는 이를 “네트 n_i 를 트랙 k 에 할당한다”라고 말한다. 이러한 정의 하에 우리의 배선 문제는 두 개의 서로 다른 네트가 같은 세그먼트를 공유하지 않도록 N 의 각 네트를 트랙에 할당하되 최소의 트랙을 사용하여 (즉, 최소의 W) 배선을 완료하는 것이며 우리는 이를 “이차원 트랙할당문제”라고 한다.

이차원 트랙할당문제는 $NP-hard$ 문제로 알려져 있다^[14]. 따라서 다음 3 절에서는 이를 위한 휴리스

틱 알고리즘을 제안하고 이를 통한 FPGA 배선방법을 설명한다.

III. 이차원 트랙할당을 통한 FPGA 배선

본 절에서는 먼저 앞의 2 장에서 소개한 이차원 트랙할당문제를 위한 휴리스틱 알고리즘을 제안하고 이를 활용한 FPGA 배선방법을 기술한다.

1. 이차원 트랙할당 알고리즘

N 을 주어진 네트리스트라고 하고 N 의 각 네트에 대한 개략배선 결과가 주어졌다고 하자. 앞에서 설명한 바와 같이 이러한 개략배선 결과는 각 네트가 어떤 CB를 사용하여 배선할 것인지를 나타낸다. 따라서 하나의 CB에는 이 CB의 세그먼트를 사용하여 배선될 네트들이 이미 결정되어 있는데 이러한 네트의 갯수를 그 CB의 ‘밀도(density)’라고 한다. 그리고 모든 CB의 밀도중 최대 값을 ‘최대 밀도’라고 하고 이를 d_{max} 로 표시한다. 참고로 d_{max} 는 개략배선 결과를 그대로 상세배선 결과로 바꿀 경우 필요한 트랙수의 하한임을 쉽게 알 수 있다.

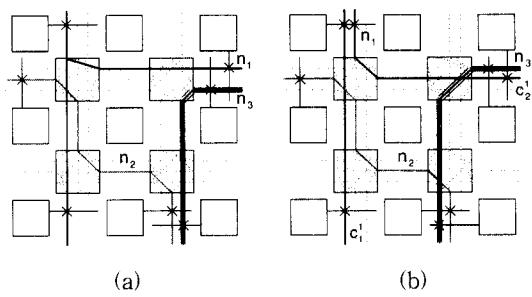


그림 5. 2-핀 네트로의 분할 (a) 분할 전 (b) 분할 후
Fig. 5. Partition to 2-pin nets. (a) Before parti-
tion. (b) After partition.

본 논문에서 제안한 알고리즘은 개략배선 결과를 상세배선 결과로 바꾸는 것이니 다른 배선구조의 특징을 반영하여 먼저 각 네트를 2-핀 네트 즉, 커넥션으로 나누어 배선을 수행한다. 각 커넥션에 대한 개략배선 결과는 원래 주어진 개략배선 결과로부터 두 핀간의 연결을 위한 배선 결과만을 취하여 얻는다. 네트를 2-핀 네트로 나누어 배선을 수행하는 이유를 그림 5를 통하여 설명한다. 그림 5(a)의 예에서는 세 개의 네트를 배선하는데 세 개의 트랙이 필요하다. 그러나 네트를 2-핀 네트로 분할하여 각각 배선하면 그림

5(b)에서 보인바와 같이 두 개의 트랙만으로 배선을 완료할 수 있다. 이 예에서 보이는 바와 같이 본 논문에서 다루는 배선구조는 네트를 분할하지 않고 그대로 배선할 경우 그 네트의 배선을 하나의 트랙에서 완료하여야 하기 때문에 다른 네트를 같은 트랙에 할당하기 어려워 필요한 트랙 수를 증가시킬 수 있다. 물론, 다른 네트의 배선에 방해가 되지 않는 범위에서 가능한 많은 부분의 네트가 하나의 트랙에 배선되어야 필요한 트랙 수를 감소시킬 수 있는데 우리는 본 절의 마지막 부분에서 설명하는 “커넥션 그룹”을 사용하여 이러한 효과를 얻도록 한다.

설명을 쉽게 하기 위하여 편의상 N 을 2-핀 네트로만 구성된 네트리스트라고 하고 각 네트에 대한 개략배선 결과가 주어졌다고 하자. 본 논문에서 제안하는 배선방법은 하나의 트랙에 가능한 많은 네트를 할당하고 이때 할당되지 못한 네트들은 새로운 트랙을 추가하여 이들의 할당을 계속하되 모든 네트가 트랙에 할당될 때까지 반복하는 것이다. 따라서 하나의 트랙에 네트들을 할당할 때 어떤 네트를 먼저 할당하여 추가로 필요한 전체 트랙 수를 줄이는가가 중요하다.

```

Calculate  $d_{\max}$  from the global routing results for  $N$ ;
 $k = 1$ ;
 $N_{tmp} = N$ ;
 $d^k_{\max} = d_{\max}$ ;
while ( $N_{tmp} \neq \emptyset$ ) {
    for ( $d^k = d^k_{\max}$ ;  $d^k \neq 0$ ;  $d^k = d^k - 1$ ) {
        Cover as many CBs of density  $d^k$  as possible by
        allocating nets in  $N_{tmp}$  to track  $t_k$ ;
        Remove the allocated nets from  $N_{tmp}$ ;
    }
     $k = k + 1$ ;
    Update  $d^k_{\max}$  from the global routing results for  $N_{tmp}$ ;
}

```

그림 6. 이차원 트랙할당 알고리즘의 개요

Fig. 6. The outline of the two dimensional track assignment algorithm.

그림 6에 제안한 알고리즘의 윤곽을 보인다. 그림의 알고리즘은 주어진 네트리스트 N 이 2-핀 네트로만으로 구성되어 있다는 가정 하에 작성한 것인데 임의의 네트리스트 또는 본 절의 끝 부분에서 설명하는 커넥션 그룹을 기본 단위로 하여도 알고리즘의 윤곽은 동일하다. 네트리스트 N 과 최대밀도가 d_{\max} 인 이들의 개략배선 결과에 대해서 현재 $k-1$ 개의 트랙에 네트를 할당하였다고 하고 아직 할당되지 않은 네트들

의 집합을 N_{tmp} 라고 하자. 그리고 N_{tmp} 의 네트들의 개략배선 결과만으로 구한 최대밀도를 d^k_{\max} 라고 하고 현재 할당이 진행중인 k 번째 트랙을 t_k 라고 하자. 이 경우 지금까지 $k-1$ 개의 트랙을 사용하였고 N_{tmp} 의 최대밀도가 d^k_{\max} 이므로 모든 네트의 배선을 위하여 최소 $k+d^k_{\max}-1$ 개의 트랙이 필요하다는 것을 쉽게 알 수 있으며 우리의 알고리즘에서는 이를 달성하기 위하여 노력한다.

이를 위하여 제안한 알고리즘에서는 먼저 밀도가 d^k_{\max} 인 CB들이 가능한 모두 배선에 사용되도록 트랙 t_k 에 네트를 할당한다. 만일 이를 달성하였다면 다음 트랙 t_{k+1} 에 네트를 할당할 때 그 밀도 d^{k+1}_{\max} 는 $d^k_{\max} - 1$ 이 되어 최소 트랙에 모든 네트를 할당할 가능성이 높아진다. 여기서 어떤 네트 n_i 를 트랙 t_k 에 할당하여 배선할 때 n_i 의 개략배선 결과에 속한 CB의 인덱스가 k 인 세그먼트가 n_i 의 배선에 사용되게 되는데 이때 우리는 “ n_i 가 CB를 트랙 k 에서 커버(cover)한다” 또는 단순히 “CB를 커버한다”라고 말한다.

트랙 t_k 에서 밀도가 $d^k = d^k_{\max}$ 인 CB들을 N_{tmp} 의 네트로 최대한 많이 커버하였다고 하자. 그런데 이러한 작업을 마친 후에도 트랙 t_k 에는 커버되지 않은 CB들이 존재할 수 있는데 이러한 CB들이 추가로 커버될 수 있도록 네트를 추가로 할당한다. 이때 가능한 많은 CB들이 커버될 수 있도록 네트를 할당할 수도 있으나 이보다는 다음 이어지는 트랙에서의 네트할당에 도움을 주는 방향으로 네트를 할당하는 것이 보다 유리하다. 이를 위하여 제안한 알고리즘에서는 밀도가 $d^k = d^k_{\max} - 1$ 인 CB들을 최대한 커버한 후 다음에는 밀도가 $d^k = d^k_{\max} - 2$ 인 CB를 가능한 많이 커버하도록 한다. 마찬가지로 만일 이러한 과정 후에도 커버가 안된 CB가 존재할 경우 밀도가 $d^k = d^k_{\max} - 3$ 인 CB들을 가능한 많이 커버하도록 네트들을 할당하며 이러한 과정을 반복적으로 더 이상 할당할 네트가 없을 때까지 계속한다. 여기서 편의상 d^k 를 ‘타겟 밀도’라고 부르고 밀도가 d^k 인 CB를 ‘타겟밀도 CB’라고 부른다.

지금까지 2-핀 네트로만 네트리스트가 구성되어 있을 경우의 배선에 대하여 설명하였다. 만일 주어진 네

트리스트가 다중-핀 네트로 구성되어 있다면 앞에서 언급한 바와 같이 각 네트를 2-핀 네트로 분할하여 배선을 수행한다. 그러나 이 알고리즘을 보다 효율적으로 적용하기 위해서는 커넥션 단위로 트랙에 할당하기 보다는 필요한 경우 이들을 묶어서 할당하는 것이 보다 효율적일 수도 있다.

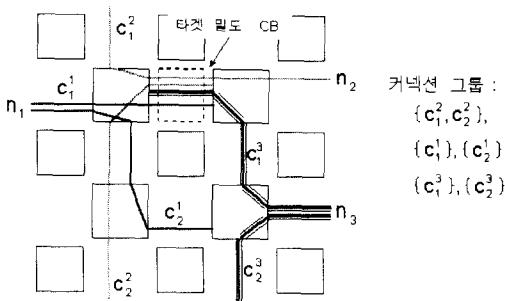


그림 7. 커넥션 그룹의 생성 예

Fig. 7. An example of forming connection groups.

그림 7의 예는 세 개의 네트를 각각 커넥션으로 나눈 결과를 보인 것이다. 이 예에서 타겟 밀도(이 경우 3)를 갖는 CB를 커버하기 위하여 커넥션 c_1^2 를 현재의 트랙에 할당하였다면 커넥션 c_1^2 역시 같은 트랙에 할당하여야 이 CB를 완전히 커버할 수 있다. 이와 같이 타겟밀도를 갖는 CB를 커버하여 그 밀도를 하나 줄이는 과정에서 동시에 할당하여야 할 커넥션들의 묶음을 ‘커넥션 그룹(CG)’이라고 한다. 제안한 알고리즘에서는 각 트랙에서 이러한 CG를 구성하여 이를 단위로 트랙에 할당하는데 이를 통한 배선과정은 그림 6에 보인 과정에 CG를 구하는 과정을 추가하고 할당의 기본 단위를 네트 대신 CG로 대치하면 충분하다. 참고로 그림 7의 예에서 커넥션 c_1^1 과 c_2^1 은 c_1^1 하나만 할당하였어도 현재의 타겟밀도 CB를 커버할 수 있기 때문에 CG로 묶이지 않으며 이는 커넥션 c_1^3 과 c_2^3 에 대해서도 마찬가지이다.

마지막으로 그림 6에 보인 배선 과정에서 필요한 사항은 어떤 네트 또는 CG를 먼저 선택하여 배선할 것인지를 판단하기 위한 기준이다. 이를 위하여 CG간의 우선순위를 정의하여 이에 의한 선택을 하는데 이어지는 2절에서는 이러한 우선순위에 관하여 기술한다.

2. 커넥션 그룹의 우선순위

기존의 1-차원 트랙 할당을 위해서 각 네트에 적용

하는 우선순위는 일반적으로 밀도가 높은 곳에 위치하는 네트를 우선적으로 할당하거나 왼쪽 에지 우선(left edge first) 순위를 적용하여 네트를 할당한다^[12,13]. 그러나 본 논문에서 다루는 문제는 2-차원 트랙 할당 이므로 보다 다른 형태의 방법으로 네트를 선택하여야 한다. 실제로 우리의 알고리즘에서는 네트 대신 CG를 기본 단위로 하여 이들을 선택하여 트랙에 할당하므로 이에 대한 선택 방법이 필요하다. 본 절에서는 이러한 CG를 선택하는 방법에 대하여 설명한다.

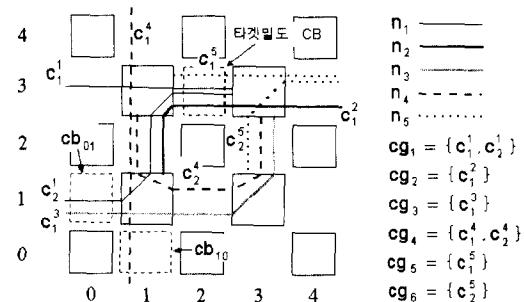


그림 8. CG의 우선 순위 설정을 설명하기 위한 예

Fig. 8. A illustration for setting CG priorities.

그림 8에 이를 위한 하나의 예를 보인다. 그림의 예는 5 개의 네트로 구성된 경우인데, 현재 트랙 1에 CG를 할당중이라고 가정한다(타겟밀도가 3). 이 경우 CG들은 그림에 보인바와 같다. 설명을 위하여 편의상 가장 오른쪽 아래에 존재하는 LB를 기준으로 그림에서 보인바와 같이 가로 및 세로 좌표에 의하여 각 CB를 명시한다. 예를 들어 그림의 가장 왼쪽 아래에 존재하는 수직 CB는 cb_{10} 으로 표시하며 수평 CB는 cb_{01} 로 표시한다. 따라서 그림에서 현재 타겟밀도 CB는 cb_{12} , cb_{23} , cb_{32} 이다.

그림에서 현재 cg_5 (즉, c_1^5)의 할당이 이미 이루어졌다고 가정하자. 이러한 상태에서 현재의 트랙에 할당할 수 있는 CG들은 cg_3 , cg_4 , cg_6 이다. 그런데 만약 cg_3 를 현재의 트랙에 할당한다면 더 이상 아무런 추가의 CG를 할당할 수 없게 되어 타겟밀도 CB인 cb_{12} 를 현 트랙에서 커버할 수 없다. 즉, 이 경우 배선 완료를 위하여 최대밀도보다 적어도 하나 더 많은 트랙이 필요하다. 이와 같이 CG 하나의 할당으로 인하여 커버할 수 없는 CB가 생길 경우 “그 CG가 CB를 가로막는다”라고 말하고 이 CG의 할당으로 인하여 현 트랙에 할당할 수 없는 CG들을 “가로막힌 CG”라

고 한다. 하나의 CG가 가로막는 CB의 수가 많으면 그 CG의 할당으로 인하여 현재의 트랙에서 커버될 수 있는 타겟밀도 CB의 수가 적어지기 때문에 타겟밀도를 갖는 모든 CB를 커버할 가능성이 적어진다. 따라서 이러한 CG의 할당 우선순위를 작게 한다. 이를 위하여 각 CG마다 *blocked_cbs*라는 변수를 주는데 이는 그 CG가 가로막는 타겟밀도 CB의 수를 나타낸다. *blocked_cbs*가 1 이상인 CG를 배선하면 가로막히는 CB가 생기게 되어 사용해야 할 트랙의 수가 늘어나게 된다. 따라서 *blocked_cbs*가 0인 CG들을 우선적으로 배선해야 한다. 그러나 모든 CG들의 *blocked_cbs*가 0 이상이면 그 값이 작은 CG일 수록 높은 우선순위를 갖는다. 이것은 가로막히게 되는 CB의 수를 줄여서 후의 개략배선 결과의 수정을 가능한 피하기 위함이고 또한, 모든 타겟 밀도 CB가 커버되지 못하면 다음 트랙 할당 단계에서 커버하여야 하는데 이때 타겟 밀도 CB의 수를 줄여 쉽게 커버할 수 있도록 하기 위함이다. 참고로 그림 8의 예에서 cg_3, cg_4, cg_6 에 대한 *blocked_cbs* 값은 각각 1, 0, 1이다. 따라서 이 경우 cg_4 가 선택되어 트랙 1에 할당된다. 이렇게 함으로써 이 예는 3 개의 트랙으로 배선을 완료할 수 있다.

만일 어떤 CG의 *blocked_cbs*가 0 이라면 이 CG를 ‘자유 CG’라고 한다. 자유 CG는 타겟밀도 CB를 가로막지 않으므로 가능한 많이 존재하는 것이 좋다. 각 CG에 주어진 또 하나의 변수는 *blocked_free_cg*s이다. 이는 하나의 CG를 할당하였을 때 이로 인하여 가로막힌 CG 중 자유 CG의 개수이다. 그런데 이 값이 클수록 현재 할당할 수 있는 자유 CG의 수가 적어지므로 가능한 이러한 CG를 할당하지 말아야 한다. 예를 들어 그림 8의 예에서 cg_3 을 할당함으로써 가로막히는 CG들은 cg_4, cg_6 인데 이중 cg_4 가 자유 CG이므로 cg_3 의 *blocked_free_cg*s 값은 1이다.

하나의 타겟밀도 CB를 지나며 현재 할당 가능한 CG의 개수를 그 CB의 ‘자유도’라고 한다. 변수 *min_routable_cg*s는 하나의 CG를 할당하였다고 가정하고 계산한 각 타겟밀도 CB의 자유도의 최소 값이다. 이 값이 크다는 의미는 각 타겟밀도 CB를 커버할 수 있는 CG들이 아직도 최소한 *min_routable_cg*s 만큼 있다는 것을 의미하므로 가능한 이 값이 큰 CG를 할당하도록 한다. 그런데 만일 현재 이미 가로막힌

타겟밀도 CB가 존재한다면 *min_routable_cg*s는 0이므로 이 변수의 효력은 사라진다.

변수 *covered_cbs*는 CG를 할당하였을 때 이 CG가 커버하는 타겟밀도 CB의 수이고 *length*는 이 CG가 커버하는 전체 CB의 수이다.

트랙할당을 위하여 각 CG에 대한 이러한 변수들의 값을 유지하며 아래와 같은 우선순위로 heap을 구성한다:

1. *blocked_cbs*가 작은 CG.
2. *blocked_free_cg*s가 작은 CG.
3. *min_routable_cg*s가 큰 CG.
4. *covered_cbs*가 큰 CG.
5. *length*가 작은 CG.

제안한 알고리즘에서 CG를 선택하여 트랙에 할당하는 과정은 이렇게 구성한 heap의 루트에 있는 CG를 heap에서 제거한 후 현재의 트랙에 할당하고, 할당한 CG에 의하여 가로막힌 CG들을 heap에서 제거한 후, heap의 나머지 CG들의 변수 값의 이 할당에 따른 갱신을 수행함과 동시에 heap을 재조정한다.

IV. 배선향상 기법

지금까지 주어진 개략배선 결과를 수정하지 않고 이를 그대로 상세배선 형태로 바꾸는 이차원 트랙할당 방법을 설명하였다. 이 방법은 만일 개략배선 결과가 지연시간을 고려하여 이를 최소화 하도록 배선한 결과인 경우^[27]에 활용할 수 있다. 그러나 만일 배선의 목적이 가능한 적은 트랙을 사용하여 배선하는 것이라면 기존의 직접 상세배선 하는 방법에 비하여 그 배선율이 매우 떨어진다. 따라서 본 절에서는 개략배선 결과가 주어졌을 경우 이차원 트랙할당시 필요한 경우 개략배선 결과를 수정하여 배선율을 높이는 두 가지의 배선향상 기법에 대하여 설명한다.

첫 번째 방법은 다음과 같다.

앞의 3.1 절에서 기술한 이차원 채널할당 알고리즘을 적용할 때 타겟밀도 d^k 인 CB 중 일부가 커버되지 않는다고 하자. 만일 $d^k = d_{\max}^k$ 이라면 이렇게 커버되지 못하는 CB로 인하여 배선을 완료하기 위한 트랙수가 필연적으로 d_{\max} 보다 커지게 된다. 또한 $d^k < d_{\max}^k$ 일 경우에도 커버되지 못하는 CB들은 그 밀도가 줄지 않아 다음 트랙에서의 할당시 부담으로

남게된다. 이러한 부담을 덜기 위하여 현재 가로막힌 CB를 지나는 네트중 하나를 선택하여 그 개략배선 결과를 수정하고 가로막힌 CB를 지나지 않도록 함으로써 가로막히는 CB의 수를 가능한 작게 할 수 있다.

개략배선 결과를 수정하는데는 미로배선을 사용한다. 그런데 미로배선을 단지 네트의 배선이 어떤 CB를 통하여 배선되는가를 결정하는 것이기 때문에 이에 필요한 계산 시간은 많지 않다. 수정할 네트를 구성하는 커넥션중 현재 가로막힌 CB를 지나는 커넥션을 제거하면 네트의 개략배선 결과는 단락 되고 이에 미로배선을 통하여 다시 연결을 시도한다. 이때 만일 이러한 배선 결과가 커버되지 않은 타겟밀도 CB의 수를 증가시킨다면 이 네트에 대한 수정을 포기하고 다른 네트에 대한 재배선을 시도한다.

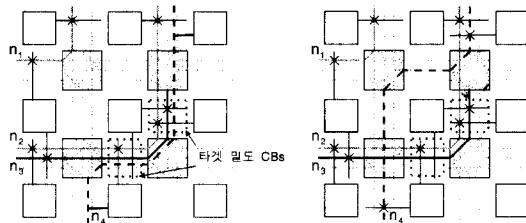


그림 9. 가로막힌 CB를 우회한 배선 (a) 가로막힌 CB가 있는 예 (b) 우회

Fig. 9. Detouring blocked CBs. (a) An example of blocked CBs. (b) Detouring.

이러한 재배선 방법을 이미 커버된 CB의 할당되지 않은 네트에 적용할 경우 d_{max} 보다 작은 수의 트랙으로도 배선을 완료할 수도 있다. 그림 9에 이러한 예를 보인다. 이 예에서는 각 네트가 2-편 네트로 구성된 것이므로 앞으로 CG 대신 네트를 할당한다고 설명하겠다. 그림에서 네트 n_1 , n_2 를 이미 트랙 1에 할당하였고 현재 타겟밀도가 2이며 트랙 2에 네트를 할당중이라고 하자. 그림 9(a)에서 보인바와 같이 네트 n_3 의 배선을 통하여 타겟밀도 CB들을 커버하였다. 그런데 이러한 타겟밀도 CB의 아직 할당되지 않은 네트 n_4 를 재배선하면 이를 그림 9(b)와 같이 이를 트랙 2에 할당할 수 있어 $d_{max}(=3)$ 보다 작은 2 개의 트랙만으로 배선을 완료할 수 있다.

트랙 수를 줄이기 위하여 사용한 또 다른 방법은 전통적인 들어내기 및 재배선 방법을 사용한 것이다. 이 방법은 이미 기존의 레이아웃 형태에서 잘 알려진

방법이므로 여기서는 이 방법을 본 논문에서 다루는 문제에 어떻게 적용하였는지를 그림 10에 보인 예를 통하여 간략히 설명한다.

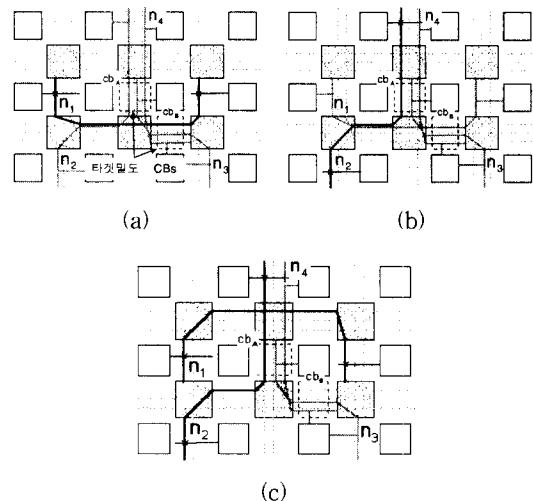


그림 10. 드러내기 및 재배선을 통한 배선율 개선
(a) 가로막힌 cb_A (b) n_2 를 할당, n_1 의 할당 취소 (c) n_1 의 재배선을 통한 트랙할당

Fig. 10. An routability improvement by rip-up and rerouting. (a) Blocked cb_A . (b) Allocate n_2 and deallocate n_1 . (c) Track allocation by the rerouting of n_1 .

그림 10(a)에서 현재 트랙 1에 네트를 할당중이라고 하자(타겟밀도 = 3). 만일 네트 n_1 을 할당하였다면 더 이상 할당할 수 있는 네트가 없어 CB cb_A 가 가로막히게 되고 따라서 배선을 완료하는데 4 개의 트랙이 필요하게 된다. 이를 해결하기 위하여 cb_A 를 지나는 네트중 하나를 골라 일단 트랙 1에 할당한다(본 예에서는 네트 n_2). 다음, 이러한 할당과 충돌하는 이미 할당된 네트(즉, n_1)를 할당되지 않은 것으로 한다. 이때 만일 가로막힌 CB의 수가 줄면 이 과정을 끝낸다. 그러나 이러한 수가 같거나 늘면 이전 할당되었던 네트를 재배선 하여 다시 할당하여 본다. 본 예에서는 그림 10(b)에 보인바와 같이 cb_B 가 새로운 가로막힌 CB가 되었다. 따라서, 네트 n_1 을 재배선 하여 다시 할당하면 그림 10(c)에 보인바와 같이 모든 타겟밀도 CB가 커버된 결과를 얻는다. 그런데 만일 재배선 후 가로막힌 CB의 수가 이전과 같거나 증가하면 이러한 들어내기 및 재배선 결과를 포기하고 다른 네트를 선

택하여 더 이상 시도할 네트가 없을 때까지 반복한다. 마지막으로 이러한 들어내기 및 재배선 방법도 이미 커버가 된 CB의 미 할당된 네트에 대해서 적용하여 필요한 트랙 수를 d_{\max} 보다 작게 할 수도 있다.

V. 실험 결과

본 장에서는 제안한 배선 방법을 여러 시험 회로에 적용하여 실험한 결과를 보인다. 제안한 배선방법을 C 언어를 사용하여 구현하여 SUN Sparc-20호환 기종인 삼보 워크스테이션 SDT-820에서 실행하였다. 시험 회로로는 기존의 개략배선후 상세 배선기 SEGA^[11]의 평가에 사용된 9 개의 시험 회로를 사용하였다. 표 1에 이러한 시험 회로의 특성을 보인다*.

표 1. 실험에 사용한 시험 회로

Table 1. Benchmark circuits.

회로 이름	FPGA array의 크기	네트의 수	커넥션의 수
alu4	19 × 17	256	850
aplex7	12 × 10	126	300
term1	10 × 9	88	202
example2	14 × 12	205	444
too_large	14 × 14	186	519
k2	22 × 20	404	1256
vda	17 × 16	225	722
9symml	11 × 10	79	259
alu2	15 × 13	153	510

각 회로들은 SEGA의 입력으로 사용되었던 회로이기 때문에 개략배선 결과를 포함하고 있다. 이러한 개략배선 결과를 입력받아 먼저 본 논문에서 제안한 이차원 트랙할당 알고리즘의 성능을 비교하기 위하여 이를 수정하지 않고(즉, 4 장에서 제시한 배선 향상 기법을 사용하지 않고) 그대로 배선을 수행하여 보았는데, 이러한 배선 결과를 개략배선 결과를 그대로 배선하는 배선기인 SEGA의 배선 결과와 비교하였다.

* 본 논문에서 사용한 데이터는 SEGA 패키지에 포함되어 있는 데이터로 이중 alu4와 apex7 등은 Robins 등이 사용한 회로와는 약간의 차이가 있다.

표 2. SEGA와의 비교

Table 2. Comparative results by SEGA.

회로 이름	d_{\max}	제안한 방법	수행시간(초)	SEGA
alu4	13	13	65	15
aplex7	13	13	2	13
term1	9	9	2	10
example2	17	17	15	17
too_large	11	11	12	12
k2	15	16	210	17
vda	13	14	44	13
9symml	10	10	2	10
alu2	10	10	14	11
Total	111	113		118

표 2에 이러한 두 비교 결과를 보인다. 표의 결과는 주어진 개략배선 결과를 수정하지 않고 배선을 수행할 경우 배선완료에 필요한 최소 트랙의 수이다. 표에서 보인바와 같이 전체적으로 볼 때 제안한 이차원 트랙 할당 방법이 기존의 방법에 비하여 보다 적은 트랙에 네트의 배선을 완료함을 볼 수 있으며, 제안한 방법은 대부분의 회로에 대하여 d_{\max} 만큼의 트랙만을 사용하여 배선을 완료한다.

표 3. 기존의 배선 방법들과의 비교

Table 3. Comparative results with other routing method.

회로 이름	제안한 방법	수행시간(초)	SEGA	Tracer_fpga_PR	Robins	CMRF
alu4	10	70	15	11	11	11
aplex7	9	3	13	8	10	9
term1	8	2	10	7	8	8
example2	11	18	17	10	11	11
too_large	9	14	12	9	10	10
k2	14	232	17	14	15	15
vda	11	49	13	11	12	11
9symml	7	3	10	6	8	7
alu2	8	16	11	9	9	9

다음으로 실험을 수행한 사항은 4 장의 배선 향상 기법을 이차원 트랙할당 알고리즘에 추가하여 배선을 수행한 것이다. 이 경우 주어진 개략배선 결과가 변형되므로 보다 작은 수의 트랙을 사용하는 배선 결과를 기대할 수 있다. 이러한 실험결과를 기존의 직접 상세 배선하는 방법의 결과와 비교하였다. 비교에 사용한

배선기는 Wu의 TRACER_fpga_PR^[10], Robins의 방법^[1] 그리고 최진영의 CMRF^[18] 등이다.

표 3에 이러한 실험결과를 보인다. 그런데 표에서 SEGA에 의한 결과는 표 2에 보인 결과와 동일한 것이며 여기서는 단지 직접 상세 배선한 결과와의 성능의 차를 보이기 위하여 포함하였다. 참고로 개략배선 결과를 수정하는데 소요되는 시간은 표 3에서 보인바와 같이 표 2의 수행시간에 비하여 그리 크지 않다. 이는 개략배선의 수정에 사용하는 미로배선을 배선자원 전체가 아닌 하나의 트랙을 대상으로 수행하기 때문이다.

표에서 보인바와 같이 Robins의 방법과 CMRF에 의한 결과 보다는 모든 시험 회로에 대해서 사용한 트랙의 수가 같거나 작았다. 그러나 TRACER_fpga_PR의 결과와 비교하면 몇몇 시험 회로(alu4, alu2)에 대해서는 적은 트랙을 사용하였지만, 더 많은 수의 트랙을 사용하는 시험 회로(apex7, term1, example2, 9symml)가 많았다. 이는 제안한 배선 방법이 개략 배선 결과를 입력으로 받아 이를 토대로 배선하므로 최종 배선 결과가 주어진 개략 배선 결과의 영향을 받기 때문이다. 즉, 4 장의 배선 향상 기법을 사용하여도 개략 배선 결과에 대한 의존도를 줄인 것이지 완전히 독립적이지는 않기 때문이다. 따라서 보다 우수한 개략배선 결과를 사용한다면 더 좋은 배선 결과를 낼 수 있을 것이다. 그림 11에 alu2에 대한 최종 배선 결과를 보인다.

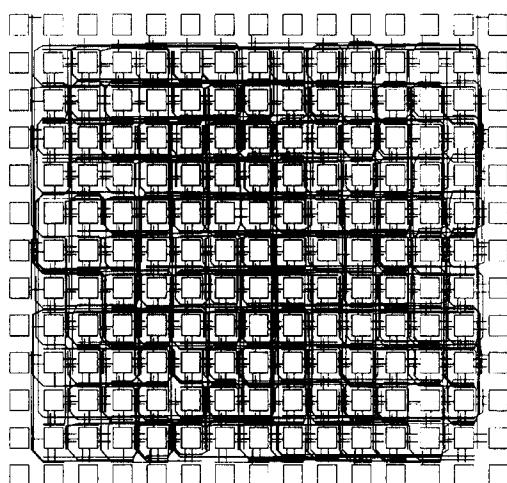


그림 11. 회로 alu2에 대한 배선결과

Fig. 11. The routing result for 'alu2'.

VI. 결 론

본 논문에서는 FPGA의 배선구조의 특성을 이용한 새로운 배선 방법을 제안하였다. FPGA의 연결 블록과 스위치 블록이 앞의 그림 2에 보인바와 같은 형태를 가질 경우에는 배선문제는 이차원 트랙할당 문제로 변환될 수 있다. 따라서 본 논문에서는 이러한 형태의 배선을 위하여 이차원 트랙할당을 위한 새로운 휴리스틱 알고리즘을 제안하였다. 제안한 방법에서는 주어진 네트의 개략배선 결과를 커넥션 그룹이라는 단위로 나누어 이 각각에 우선순위를 주고 이에 의하여 각 커넥션 그룹을 트랙에 할당한다. 제안한 배선방법을 다수의 시험회로에 적용한 결과 기존의 같은 부류의 배선기보다 우수한 결과를 얻는다.

본 논문에서 제안한 배선 방법은 입력으로 주어진 개략배선 결과를 토태로 배선을 수행한다. 따라서 개략배선 결과를 수정하지 않고 그대로 상세배선으로 바꾸면 기존의 직접 상세배선하는 방법에 비하여 그 결과가 좋지 않을 수도 있다. 따라서 본 논문에서는 이차원 트랙할당중 필요시 개략배선 결과를 수정하여 보다 좋은 결과를 내기위한 방법을 아울러 제안하였다. 이러한 방법을 사용할 경우 기존의 직접 상세배선하는 방법과 유사한 결과를 얻는다.

앞으로 이러한 접근 방법에 배선을 뿐만 아니라 지역시간도 아울러 고려할 수 있는 방안이 필요하다.

참 고 문 헌

- [1] M. J. Alexander and G. Robins, "New Performance-Driven FPGA Routing Algorithms", Proc. of 32nd DAC, 1995, pp. 562-567.
- [2] K. D. Boese, A. B. Kahng and G. Robins, "High-Performance Routing Trees With Identified Critical Sinks", Proc. of 30th DAC, 1993, pp. 182-187.
- [3] S. Brown, R. J. Francis, J. Rose, and Z. G. Vranesic, *Field-Programmable Gate Arrays*, Kluwer Academic Publishers, 1992.
- [4] S. Brown and J. Rose, "Flexibility of Interconnection Structures for Field-Programmable Gate Arrays", *IEEE J. of Solid-State Circuits*, vol. 26, no. 3, pp.

- 277-282, Mar. 1991.
- [5] S. Brown, J. Rose and Z. G. Vranesic, "A Detailed Router for Field-Programmable Gate Arrays", *IEEE Trans. on CAD*, vol. 11, no. 5, pp. 620-628, May 1992.
- [6] S. Brown, J. Rose and Z. G. Vranesic, "A Stochastic Model to Predict the Routability of Field-Programmable Gate Arrays", *IEEE Trans. on CAD*, vol. 12, no. 12, pp. 1827-1838, Dec. 1993.
- [7] J. Cong, A. B. Kahng, G. Robins, M. Sarrafzadeh and C. K. Wong, "Provably Good Performance-Driven Global Routing", *IEEE Trans. on CAD*, vol. 11, no. 6, pp. 739-752, June 1992.
- [8] A. Hashimoto and J. Stevens, "Wire Routing by Optimization Channel Assignment Within Large Apertures", *Proc. of 8th DAC*, 1971, pp. 155-163.
- [9] C. Y. Lee, "An Algorithm for Path Connection and its Application", *IRE Trans. on Electronic Computers*, EC-10, pp. 346-365, Sep. 1961.
- [10] Y.-S. Lee and A. C.-H. Wu, "A Performance and Routability Driven Router for FPGAs Considering Path Delays", *Proc. of 32nd DAC*, 1995, pp. 557-561.
- [11] G. G. Lemieux and S. Brown, "A Detailed Routing Algorithm for Allocating Wire Segment in Field-Programmable Gate Array," *Proc. of PDW*, Apr. 1993, pp. 215-226.
- [12] T. Ohtsuki, *Layout Design and Verification*, North-Holland, 1986.
- [13] N. A. Sherwani, *Algorithms for VLSI Physical Design Automation*, Kluwer Academic Publishers, 1993.
- [14] Y.-L. Wu and M. Marek-Sadowska, "Graph Based Analysis of FPGA Routing," *Proc. of EDAC*, 1993, pp. 104-109.
- [15] Y.-L. Wu and M. Marek-Sadowska, "An Efficient Router for 2-D Field Programmable Gate Arrays," *Proc. of EDAC*, 1994, pp. 412-416.
- [16] Xilinx, *The Programmable Gate Array Data Book*, Xilinx, Inc., 1992.
- [17] 최기영외 3인, "FPGA 구조 및 개발 시스템에 관한 연구," 2 차년도 연구보고서(ISRC 94-E-2100), 서울대학교 반도체공동연구소, 1995
- [18] 최진영, 임종석, "동시 미로 배선 방법에 의한 새로운 FPGA 배선 방법," *전자공학회논문지*, 제 31 권, A 편, 제 10 호, pp. 119-131, 1994

저자 소개



李征周(正會員)

1994年 2月 서강대학교 전자계산학과 학사. 1996年 2月 서강대학교 전자계산학과 석사. 1996年 2月 ~ 현재 삼성전자 반도체 System LSI 본부

林鍾錫(正會員)

1981年 서강대학교 전자공학과 학사. 1983年 한국 과학기술원 전기 및 전자공학과 석사. 1989年 Univ. of Maryland, College Park, 전기공학과 박사. 1983年 3月 ~ 1990年 8月 한국 전자통신연구소 연구원. 1990年 9月 ~ 현재 서강대학교 전자계산학과 부교수