論文2000-37SD-2-9

재구성 가능한 회로 보드를 위한 새로운 Quadratic Boolean Programming 수식에 의한 분할

(Circuit Partitioning Using A New Quadratic Boolean Programming Formulation for Reconfigurable Circuit Boards)

崔 然 景 *, 林 鐘 錫 **

(Yeun Kyung Choi and Chong Suk Rim)

요 약

본 논문에서는 IC(Integrated Circuits) 칩들간의 배선 위상(topology)이 정해진 재구성 가능한 (reconfigurable) FPGA(Field Programmable Gate Array) 기반 보드로의 회로 분할 문제로써 새로운 quadratic boolean programming 수식(formulation)을 제안한다. 본 수식의 목적은 회로 분할 시 사용하는 핀수와 네트들의 배선 길이의 합을 최소화하는 것이며 기존의 분할 방법에서 고려하는 제약조건 외에 서로 인접하지 않은 IC 칩들을 연결하기 위하여 다른 IC 칩을 통과(pass through)하는 네트들에 의해 사용되는 핀수도 고려한다. 또한 본 논문에서는 제안한 분할 문제를 효율적으로 해결하기 위하여 모듈 할당 방법으로 구성되어 있는 휴리스틱(heuristic) 분할 방법을 제안한다. 입력된 회로에 대하여 다른 분할 방법과 비교하여 실험한 결과 분할 문제의 주어진 제한들을 모두 만족하였다. 대부분의 배선된 회로에 대하여 핀 사용률이 적게 나타났으며 네트들의 사용한 배선 길이의 합은 최대 34.7% 적게 나타났다.

Abstract

We propose a new formulation by quadratic boolean programming to partition circuits for FPGA based reconfigurable circuit boards, in which the routing topology among IC chips are predetermined. The formulation is to minimize the sum of the wire length by considering the nets passing through IC chips for the interconnections between chips which are not adjacent, in addition to the constraints considered by the previous partition methods. We also describe a heuristic method, which consist of module assignment method to efficiently solve the problem. Experimental results show that our method generates the partitions in which the given constraints are all satisfied for all the benchmark circuits tested. The pin utilization are reduced for the most of the circuits and the total wire length of the routed nets are improved up to 34.7% compared to the previous method.

* 正會員, 慶敏大學 電子計算科
(Dept. of Computer Science, Kyungmin College)
** 正會員, 西江大學校 컴퓨터學科
(Dept. of Computer Science and Engineering, Sogang University)
※ 본 연구는 1998년도 정보통신부 연구비 지원에 의하여 수행되었습니다.
接受日字 : 1999年1月29日, 수정완료일 : 1999年12月27日

Ⅰ. 서 론

최근 FPGA(Field Programmable Gate Array) 기술은 그 용량 및 속도면에서 눈부신 발전을 계속하고 있으며 이에 따른 응용 분야 역시 점점 확산되고 있다. 재프로그래밍(reprogramming)이 가능한 FPGA들은 개발된 회로 성능 시험 보드(circuit emulation board)에 놓여 대용량 VLSI 시스템의 프로토타이핑(prototyping)

개발을 위하여 흔히 사용되고 있다. 또 다른 대표적인 응용 분야로는 하나의 동일 시스템으로 여러 가지 서로 다른 일을 할 수 있도록 하드웨어 시스템을 재구성할 수 있는 재구성 가능한 시스템(reconfigurable system)을 들 수 있다.

FPGA들을 기반으로 하는 시스템들은 FPGA IC 칩과, ROM, RAM, 마이크로 프로세서 등으로 구성되어 있다. FPGA들은 회로 구현(이를 앞으로 '회로 구현용 FPGA'라고 함) 및 외부 입출력 신호와 연결(이를 앞으로 '입출력 구현용 FPGA'라고 함)하기 위하여 사용되며, 또한 칩들간의 연결을 위해서도 사용된다. 어떤 시스템에서는 FPGA 칩들간을 연결을 재프로그래밍 가능하도록 하는 연결 전용 칩을 따로 두고 있다. 이의 한 예로 AXB-AP4 보드에서 사용하는 FPICTM(Field Programmable Interconnect Component)[21]가 있다. 이들 칩들은 보드 상에서 놓을 수 있는 위치와 서로간을 연결할 수 있는 연결선들이 고정되어 있어서 각 구성원들(components)간에는 특정한 위상(topology)이 형성되어 있다.

FPGA 기반 시스템들을 사용하기 위해서는 입력 회로를 주어진 보드의 칩들에 놓기 위한 회로 분할 방법이 필요하며 분할 후 칩간을 주어진 연결선들을 사용하여 배선(routing)하여야 한다. 그런데 FPGA 기반 시스템에서의 분할 문제는 일반적인 레이아웃 형태에서의 분할 문제와는 달리 분할 후에 네트의 배선을 완료하기 위해서 네트들이 다른 칩들간의 연결을 위하여 통과하게 되는 칩에서의 핀수도 함께 고려하여야 한다. 그림 1(a)에서처럼 일반적인 레이아웃 형태에서는 배선을 위하여 칩들 사이에 존재하는 채널의 빈 공간을 사용하여 자유롭게 배선할 수 있으나, FPGA 기반 시스템에서는 그림 1(b)와 같이 칩들간에 주어진 연결선들을 이용하여 배선하여야 한다. 그림 1(b)의 경우 네트의 배선을 위하여 네트의 배선 경로(routing path)상에 놓여 있는 다른 칩들을 통과(pass through)하여야 한다. 서로 다른 칩간의 네트를 연결하기 위하여 중간에 놓인 다른 칩을 통과하여야 하는 네트를 '통과 네트'(bypass net)라고 한다. 그림 1(b)의 경우에서는 통과 네트들이 최단 경로(shortest path)를 사용하여 연결되었으나 다섯 개의 핀을 추가로 사용하였다(그림 1(b)에서 작은 회색 원으로 표시함).

입출력 전용 FPGA 칩에서는 다른 FPGA 칩들에서와는 달리 보드 외부와의 연결을 위한 연결선들이 존재하여 외부 신호와 연결한다. 그림 2에 이러한 연결선의 사용 예를 보인다. 그림의 회로 a는 입출력 전용 FPGA에 놓여 있으며 보드 외부와 한 개의 신호(회로 a의 오른쪽으로 연결된 선)를, 다른 FPGA 칩과 두 개의 신호(회로 a의 왼쪽으로 연결된 선)를 연결하여야 한다. 이를 위하여 회로 a는 세 개의 연결선(세 개의 핀)을 사용하였으며 다른 회로와는 달리 외부와의 연결을 위하여 입출력 칩의 한 개의 연결선(한 개의 핀)을 추가적으로 사용하였다.

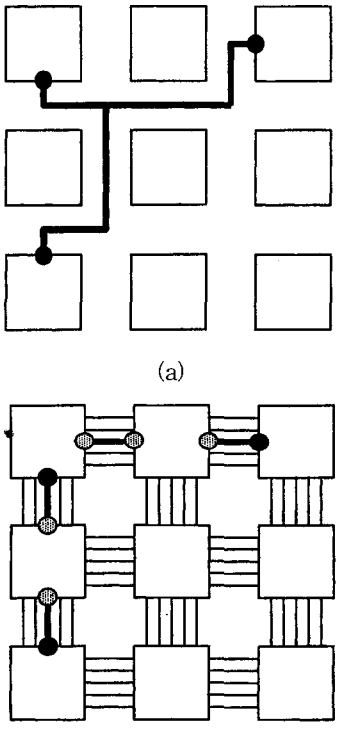


그림 1. 위상기반 회로분할문제의 설명을 위한 예
(a) 기존 레이아웃 형태에서의 배선
(b) Reconfigurable system에서의 배선

Fig. 1. An Example for describing the thoplogy based partition.
(a) Routing for the general layout style.
(b) Routing for the reconfigurable circuit boards.

기존의 일반적인 레이아웃 형태에서의 분할 문제가 회로가 놓이는 기하학적(geometric) 위치를 기반으로 하는 분할 문제임에 비하여 본 논문에서 고려하는 FPGA 기반 회로 분할 문제는 칩들간의 위상을 고려하

는 분할 문제이므로 이를 '위상 기반 회로 분할 문제'(topology based circuit partition problem)라고 한다. 주어진 회로를 FPGA 기반 시스템으로 분할할 경우에는 다음에 제시한 제약 사항들을 모두 고려하여야 한다.

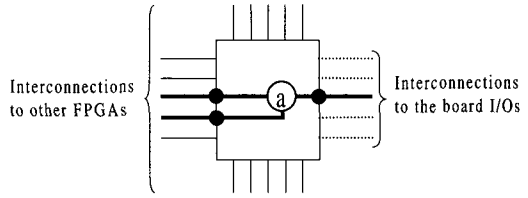


그림 2. 보드 입출력과의 연결
Fig. 2. Interconnctions to the board I/Os.

1) 칩의 용량,
2) 서로 다른 회로 구현용 칩들간을 연결하는 연결선들이 사용하는 핀수,
3) 통과 네트들이 통과하는 칩에서 사용하는 핀수,
4) 입출력 구현용 칩들과 다른 구성원들간의 연결선들이 사용하는 핀수.

기존의 많은 분할 방법들이 있으나[1,4-9,10,14,16-19] 대부분[1,4-7,9,10,17-19]이 오직 제약 사항 1)과 2)를 고려하였으며 다른 방법들에서도 제약 사항 3) 또는 4)항 둘 중에 한가지만을 고려하였거나[5,8,16], 특정한 보드 형태에서 3)과 4)를 고려하였다[14]. Fiduccia and Mattheyses[7]가 회로 이동(movement)에 의한 분할 방법을 제안한 이후 많은 이동에 의한 분할 방법들이 제안되었다[6,8,15]. 이동에 의한 분할 방법은 단순하여 구현하기 쉬우나 초기 분할에 많이 의존하여 지역적 최소해(local minimum solution)로 빠질 우려가 있다.

회로의 크기가 증대됨에 따라서 큰 회로를 작은 크기의 부회로들(sub-circuit)로 묶는 클러스터링(clustering)에 의한 분할 방법들이 많이 사용되었다. Chou[6]은 빠른 시간에 분할 해를 얻기 위하여 비율 컷(ratio cut) 방법을 개선한 지역적 비율 컷(local ratio cut) 방법을 제안하였다. 최근에는 디자이너 회로 설계 시 정의한 회로들의 계층구조를 이용하거나, 회로들을 반복적으로 클러스터링하여 계층구조를 형성한 후에 이를 이용하여 분할을 하는 방법들이 제안되었다[4,10].

수학적인 기법을 사용한 분할 방법들은 네트 리스트(net list)나 비용에 관한 전역적(global)인 고려를 할 수 있는 특징 때문에 흔히 사용되었다[1,5,9,17-19]. 이들 방법을 사용할 경우에는 좋은 분할 해를 얻을 수 있으나 많은 메모리와 연산에 의하여 수행시간에 대한 부담으로 사용하기가 어려웠다. 그러나 하드웨어의 발전과 연산의 속도를 단축할 수 있도록 개발된 알고리즘 및 효율적인 자료구조의 사용으로 이를 많이 해결할 수 있다.

분할 및 배치 문제를 정의하기 위한 방법으로 quadratic programming을 이용한 방법들도 많이 존재한다[9,17,18]. Gordian[9]의 경우에는 quadratic programming 수식으로 배치 문제를 정의하고 얻은 해를 배치 방법의 초기해로 사용하였으며 이를 이용하여 다른 배치 방법을 다시 적용하여 초기해를 개선하였다. Shih와 Kuh[17]의 경우에는 quadratic boolean programming에 의한 분할 방법을 제안하였으며 후에 이를 변형하여 용량과 I/O 핀수를 고려하는 분할 방법[18]을 구현하였다. 그러나 이 방법들도 제약 사항 3)과 4)를 고려하지 않았다.

보드 구조를 고려하는 분할 방법으로 Chan 등[5]은 각 FPGA로 분할된 회로가 각 칩안에서 배선 복잡도를 고려하는 분할방법을 구현하였다. 이 방법에서는 반복적으로 칩에서의 배선 결과를 이용하여 다음 분할 방법에 적용하였으나 칩들간의 배선에 대해서는 고려하지 않았다. Riess 등[14]과 Sankarasubramanian 등[16]의 경우에는 특정한 보드 형태에서의 분할 방법을 고려하였으며, Kim 등[8]의 경우에는 다양한 형태의 에뮬레이션 보드에서 칩간의 배선 형태를 고정하고 이를 분할 과정에서 사용하는 이동에 의한 분할 방법을 제안하였으나, 이 방법에서도 역시 제약 사항 4)를 고려하지 않았다.

본 논문에서는 앞에서 제시한 모든 제약 사항들을 고려하는 위상 기반 회로 분할 문제를 위한 quadratic boolean programming 수식을 제안한다. 본 문제는 보드의 주어진 제약 사항들을 고려하면서 사용하는 핀수와 네트의 배선 길이의 합을 최소화하는 문제이다. 이 때 핀수는 분할된 회로의 배선 결과에 의존하므로 분할 시 배선이 고려되어야 계산할 수 있다. 그러나 분할 도중에 다양한 배선 구조를 갖는 보드에서 배선을 예측하는 것이 어려우므로 본 논문에서는 각 IC 칩을 위하여 그 칩이 통과 네트에 의하여 사용될 확률을 정의한다. 확률은 본문의 사용하는 핀수를 예측하기 위한 함수에 포함되어 사용되며, 다양한 보드 구조에 대해서도 적용할 수 있다.

주어진 문제는 선형(linear)의 문제로 변환된다. 변환된 문제는 전단계의 분할 해를 이용하여 문제를 변형하며 계속적으로 해를 구하는 3 장에서 기술하게 될

기존의 휴리스틱 분할 방법의 전체적인 흐름을 이용하면서 본 논문의 네 가지 제약조건들을 만족할 수 있는 분할 방법을 제안한다. 그런데 휴리스틱 분할 방법은 3.1 절에서 기술할 선형의 문제를 해결하는 방법을 반복적으로 사용하여 해를 구한다. 본 논문에서는 주어진 제한을 모두 고려할 수 있으면서도 빠르게 선형 문제의 해를 얻을 수 있는 모듈 할당 방법을 제안한다. 주어진 회로를 사용하여 실험한 결과 이전의 분할 방법과 비교하여 대부분의 회로에 대하여 핀 사용률이 적었으며 네트들이 사용한 배선 길이의 합이 최대 34.7% 감소하였다. 먼저 2 장에서는 quadratic boolean programming에 의한 위상 기반 회로 분할 문제를 정의한다. 3 장에서는 분할 방법을 기술하고 이의 실험결과를 4 장에 보인 후 결론을 낸다.

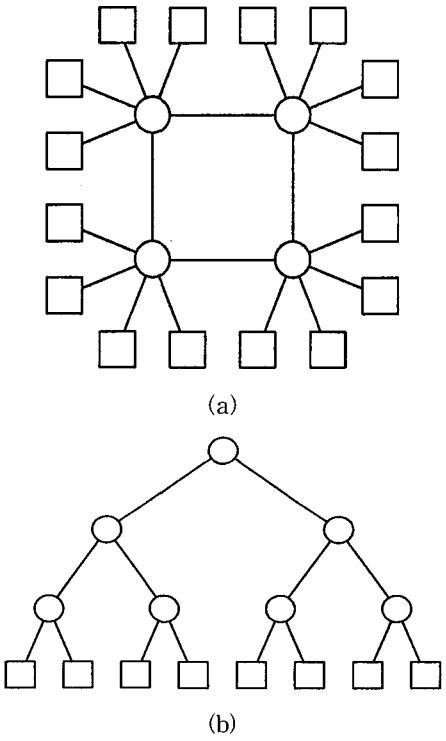
Ⅱ. 분할 문제

본 장에서는 FPGA 기반 시스템을 위한 위상 기반 회로 분할 문제를 정의한다. 분할 문제는 앞에서 제시한 시스템의 모든 제약 사항들을 고려한다. 특히 제약사항 3)을 계산하기 위하여 각 칩에 대하여 임의의 통과 네트가 그 칩을 통과할 확률을 정의한다.

FPGA 기반 시스템들을 다음과 같은 위상그래프 $G=(I,E)$로 나타낼 수 있다. 여기서 집합 I 는 보드에 놓이는 FPGA 칩이나 연결 전용 칩 등의 IC 칩에 해당하는 노드(node)들의 집합이며, 집합 E 는 집합 I 의 IC 칩간에 연결선들을 나타내는 에지(edge)의 집합이다. 그림 1 (b)에서는 메쉬(mesh) 구조의 위상그래프를 보였으며 또 다른 두 가지 위상그래프를 그림 3에 보인다. 그림에서 원은 연결 전용 IC 칩들, 사각형은 FPGA 칩과 그 이외의 IC 칩들 등의 I 의 집합에 속한 노드들을 나타낸다. 칩들간의 실선은 IC 칩들간의 연결선들인 E 의 에지들을 나타낸다.

집합 I 의 크기를 K 라고 하자. 각 칩 $i \in I$ 는 크기 c_i 와 사용 가능한 핀수 t_i 가 정해져 있다. 입출력 구현용 FPGA 칩들은 보드 내부에 존재하는 IC 칩들간의 연결을 위한 핀수 이외에 외부와의 연결을 위하여 사용할 수 있는 입출력용 핀수가 정해져 있으며 이를 b_i 라고 한다. 칩들간의 거리(distance)는 $K \times K$ 행렬 $D=[d_{i_1 i_2}]$로 표현한다. 여기서 $d_{i_1 i_2}$는 두 칩 i_1과 칩 i_2간의 거리이며 이는 두 칩간에 최단 경로 상에 존재하는 에지의 수로 한다.

분할의 대상이 되는 입력 회로는 로직 회로, 입출력 버퍼, 메모리, 클러스터, LUT 또는 IOB[20] 등의 형태가 모두 가능하며 편의상 모두 모듈이라고 한다. 집합 J 를 입력되는 모듈의 집합으로, N 을 전체 모듈의 수로 하자. 각 모듈 $j \in J$ 는 크기 s_j 를 갖는다. 모듈들간의 연결선의 수는 $N \times N$ 행렬 $A=[a_{j_1 j_2}]$로 표현하며, 이때 $a_{j_1 j_2}$는 두 모듈 j_1와 j_2간의 연결선의 수를 나타낸다.



(a)

(b)

그림 3. 위상그래프
(a) AXB-AP4 위상[21]
(b) TM-2 위상[12]

Fig. 3. Topology graph.
(a) AXB-AP4 topology.
(b) TM-2 topology.

만일 모듈에 연결된 네트 중에서 보드 밖으로 연결되어야 할 외부 시그널(external signal)이 존재할 경우에 이 네트는 입출력 구현용 FPGA 칩을 사용하여 외부와 연결하여야 한다. 이와 같은 모듈을 입출력 구현용 FPGA 칩에 놓을 경우 사용하는 핀수는 다른 모듈과의 연결을 위해 사용하는 핀수와 외부와의 연결을 위해 사용하는 핀수의 합으로 계산하여야 한다. 그러므로 외부 시그널이 있는 모듈 j 는 외부와의 연결에 필요한 연결선의 수를 다른 연결선들과 구분하여 g_j 로

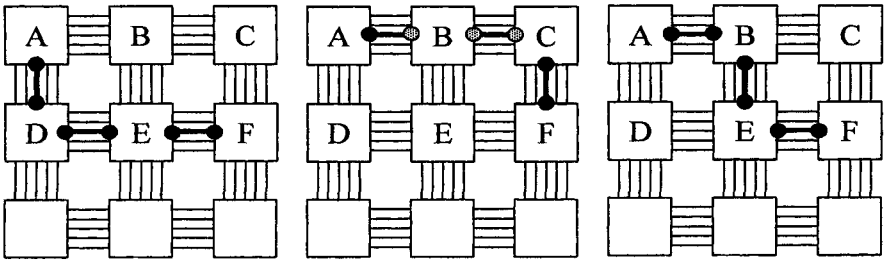


그림 4. 세 최단 경로들
Fig. 4. Three shortest paths.

나타낸다.

분할 시 각 IC 칩에서 통과 네트에 의해 사용되는 핀수를 고려하기 위하여 다음을 정의한다. 통과 네트가 임의의 IC 칩을 통과할 확률을 '통과 확률'(bypass probability)이라고 한다. 각 IC 칩 i_1의 통과 확률은 행렬 $P^{i_1}=[p^{i_1}_{i_2 i_3}]$로 나타낸다. 여기서 $p^{i_1}_{i_2 i_3}$는 IC 칩 i_2에서 IC 칩 i_3 사이에 연결해야 할 네트가 존재할 경우에 그 네트가 IC 칩 i_1을 통과할 확률이다. 이는 IC 칩 i_2와 i_3 사이에 존재하는 모든 최단 경로 중에서 IC 칩 i_1이 경로에 포함되는 최단 경로의 수로 하며 이를 수식으로 표현하면 다음과 같다.

$$p^{i_1}_{i_2 i_3}=\frac{\text{the number of shortest paths, each of which crosses chip } i_1}{\text{the number of all shortest path from chip } i_2 \text{ to chip } i_3}$$

그림 4에 한 예를 보인다. 만일 IC 칩 A와 F가 그림과 같이 놓인 경우에 칩 A와 F간을 연결하는 네트의 최단 경로의 수는 그림 4에 나타낸 바와 같이 세 개이다. 이때 네트가 IC 칩 B와 E를 통과하는 최단 경로의 수가 두 개 존재하므로 p^B_{AF}와 p^E_{AF}는 각각 2/3이다. 동일한 방법으로 p^C_{AF}와 p^D_{AF}는 각각 1/3이다. 이와 같은 방법으로 계산된 통과 확률은 행렬로 구성되어 각 IC 칩을 위하여 정해진다. 이는 앞으로 제안할 분할 문제 수식에서 핀수 계산을 위하여 사용된다. 통과 행렬은 주어진 보드 구조에 따라서 다르게 나타나지만 보드가 정해지면 분할 과정 중에는 고정된다.

칩 i에 모듈 j가 놓이는 것을 '할당(assign)된다'고 하며 이를 이진 변수 x_{ij}로 나타낸다. 만일 모듈 j가 칩 i에 할당되면 $x_{ij}=1$이고 그렇지 않을 경우에는 0이다.

주어진 입력 회로의 모듈들을 FPGA 기반 시스템의 IC 칩으로 분할하는 것은 모듈의 집합 J들을 위상그래프 G의 각 칩으로 할당하는 문제이다. 이 문제를 quadratic boolean programming 형식으로 수식화하면 다음과 같으며

$$\min \sum_{i_1=1}^{K}\sum_{j_1=1}^{N}\sum_{i_2=1}^{K}\sum_{j_2=1}^{N} a_{j_1 j_2} d_{i_1 i_2} x_{i_1 j_1} x_{i_2 j_2} \qquad (1)$$

수식 (1)은 다음의 네 가지 제약조건들을 모두 만족하여야 한다.

크기제약조건 C_1 :

$$\sum_{j=1}^{N} s_j x_{ij} \le c_i, \quad \forall i \in I .$$

핀제약조건 C_2 :

$$\sum_{j_1=1}^{N}\sum_{j_2=1}^{N} a_{j_1 j_2}(x_{i j_1}(1-x_{i j_2})+x_{i j_2}(1-x_{i j_1}))$$
$$+2\sum_{i_2=1}^{K}\sum_{j_1=1}^{N}\sum_{i_3=1}^{K}\sum_{j_2=1}^{N} p^{i_1}_{i_2 i_3} a_{j_1 j_2} x_{i j_1} x_{i j_2} \le t_{i_1}, \forall i_1 \in I .$$

외부핀제약조건 C_3 :

$$\sum_{j=1}^{N} g_j x_{ij} \le b_i, \quad \forall i \in I .$$

할당제약조건 C_4 :

$$\sum_{i=1}^{K} x_{ij}=1, \quad \forall j \in J .$$

수식 (1)은 본 분할 문제의 목적을 나타내는 것으로써 주어진 제약조건들을 만족하면서 서로 다른 칩에 할당된 모듈들간의 네트의 배선 길이의 합을 최소화하는 분할 해 x_{ij}들의 행렬을 구하는 것이다. 크기제약조건 C_1은 각 칩에 할당되는 모듈들의 크기가 각 칩의

용량을 넘지 않도록 하여 분할 후 각 칩에서 회로들이 구현될 수 있도록 하는 제약조건이다.

핀제약조건 C_2는 각 칩에서 사용하는 핀수가 그 칩의 사용 가능한 핀수보다 작도록 하는 제약조건으로 이는 분할 후 칩간의 네트들을 모두 연결할 수 있도록 하는 중요한 조건이다. 제약조건 C_2의 식은 각 칩에서 사용하는 핀수의 합을 나타내며 이는 두 개의 항목의 합으로 나타낸다. 첫 번째 항목은 각 칩에서 그 칩에 할당된 모듈들이 다른 칩에 속한 모듈들간에 연결을 위하여 필요한 핀수를 나타낸다. 두 번째 항목은 각 칩에서 통과 네트들에 의하여 사용되는 핀수를 나타내며, 이는 그 칩을 통과하는 통과 네트의 연결선의 수와 통과 확률과의 곱으로 계산한다.

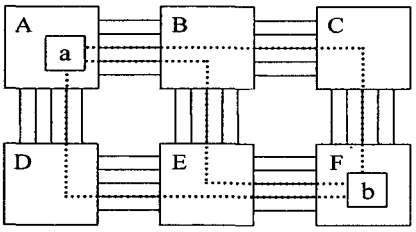


그림 5. 핀제약조건 설명을 위한 예
Fig. 5. An example for describing the pin constraint.

표 1. 핀수의 계산
Table 1. Pin calculation.

칩 명	핀수계산항	핀 수
A	1	1
B	2	2/3
C	2	1/3
D	2	1/3
E	2	2/3
F	1	1

그림 5에 표 1에 두 모듈 a와 b가 놓여 있는 경우에 핀제약조건의 두 항목 식의 계산 방법을 보인다. 그림에서 칩 A와 F의 두 모듈간에는 점선으로 표시한 세 개의 최단 경로가 존재하며 경로상에 놓인 각 칩에서의 통과 확률은 그림 4의 예에서와 동일하다. 만일 모듈 a와 b간에 연결해야 선의 수가 1이라면 각 칩에서 사용되는 핀수는 표 1과 같다. 표에서 '핀수계산항'이란 핀수 계산을 위하여 사용되는 핀 제약조건의 두 항목을 나타내며 핀수계산항의 숫자들은 각각 첫 번째 항목을 '1', 두 번째 항목을 '2'로 표시한 것이다. 그러므로 그림 5의 두 모듈간의 네트에 대해서는 표에 보인 '핀수'의 수만큼이 해당하는 칩에 핀수로 계산된다. 동일한 방법으로 모든 네트에 대하여 계산이 이루어지면 각 칩에서 사용하는 핀수의 합이 된다.

핀제약조건의 의미를 살펴보면 첫 번째 항은 분할 시 필요한 핀수의 최소값으로 기존의 분할 방법에서 많이 고려한 제한이다. 두 번째 항은 배선을 위하여 추가적으로 사용할 수 있는 핀수에 대한 '여유분'이다. 여유분을 두는 것은 핀제약조건이 두 항의 합으로 표현되어 분할 시 배선을 위한 핀들을 고려하기 위한 것으로 분할 후 배선 시 모듈간의 연결선들이 특정한 칩 주위에 치우쳐 배선되지 않으면서도 가능한 짧게 배선될 수 있도록 모듈을 분할하기 위함이다. 그림 4와 5에서 설명한 바와 같이 통과 네트의 최단 경로에 대해서만 고려하고 이를 통과 확률로서 표현하는 이유는 실제 배선을 예측하기 어려우므로 최단 경로에 놓인 칩들에 대하여 얼마만큼씩의 핀수를 여유로 남겨 두기 위함이다. 이때 여유로 남겨둘 핀수는 배선 시 어떤 칩이 보다 사용될 가능성이 많으면 높은 확률을, 적으면 낮은 확률을 사용하는 방식으로 정하는 것이 바람직하며 이를 기준으로 각 칩에 대한 통과 확률을 정의한 것이다.

외부핀제약조건 C_3는 외부 시그널이 연결된 모듈에 의하여 사용되는 핀수가 입출력 구현용 FPGA 칩에서 외부 입출력을 위하여 사용 가능한 핀수보다 작도록 하는 제약조건이다. 제약조건 C_3는 부등식에서 b_i는 입출력 구현용 FPGA 칩인 경우에만 $b_i > 0$이므로 외부 시그널과 연결이 필요한 모듈들은 입출력 구현용 FPGA 칩에만 놓이도록 제한한다. 이는 외부 시그널과 연결이 필요한 모듈들이 다른 칩에 놓일 경우 분할 후에 보드 밖으로의 연결을 위하여 입출력 구현용 FPGA 칩과 연결이 추가적으로 필요하기 때문이다. 제약조건 C_4는 일반적인 할당제약조건으로 하나의 모듈은 하나의 칩에 할당되도록 한다.

Ⅲ. 위상 기반 회로 분할 알고리즘

본 장에서는 수식 (1)의 해를 구하는 방법을 기술한다. 수식 (1)은 다음의 3.1 절에서 기술하게 되는 선형

1. $k=1$, $h_r^{(0)}:=0(1\le r\le KN)$을 초기화한다. 상한계 $\theta_r(1\le r\le KN)$를 계산하고 초기해 $y^{(1)}$를 최적해 $y^*=y^{(1)}$, $z^*=y^{(1)T}Qy^{(1)}$로 초기화한다.
2. 모든 $1\le r_2\le KN$ 에 대하여 다음을 계산한다 .

$$\alpha_{r_2}^{(k)}=\sum_{r_1=1}^{KN} w_{r_1r_2}\, y_{r_1}^{(k)},$$

$$\phi^{(k)}=\sum_{r_1=1}^{KN}\theta_{r_1}\, y_{r_1}^{(k)}.$$

3. 다음식을 구한다.

$$z=\min_{y\in S}\sum_{r=1}^{KN}\alpha_r^{(k)}\, y_r$$

4. 단계 3의 결과를 이용하여 $1\le r\le KN$ 에 대하여 다음을 계산한다.

$$h_r^{(k)}=h_r^{(k)}+\frac{\alpha_r^{(k)}}{\max(1,|z-\phi^{(k)}|)}.$$

5. 다음 식을 구한다.

$$\min\sum_{r=1}^{KN} h_r^{(k)} y_r^{(k+1)}$$

이때 구한 해를 $y^{(k+1)}$라고 한다.

6. 만일 $z^{(k+1)}=(y^{(k+1)})^T Qy^{(k+1)}$에 대하여 $z^{(k+1)}<z^*$ 이면 $y^*=y^{(k+1)}$, $z^*=z^{(k+1)}$로 해를 갱신한다.
7. 만일 $k\ge M$ 중단하고, 그렇지 않으면 $k=k+1$로 갱신하고, 단계 2로 간다.

그림 6. 보완된 Buckard와 Bonniger의 휴리스틱 알고리즘
Fig. 6. Enhanced heuristic algorithm of Buckard and Bonniger.

문제로 변환되며 그의 해는 3.2 절에 제안한 모듈 할당 방법을 사용하여 구한다.

1. 분할 문제의 변환과 분할 방법

수식 (1)은 일차원 벡터 $y=\{y_1,y_2,\ldots,y_{NK}\}$를 사용하여 다음의 식으로 변형할 수 있다.

$$\min\ y^T Wy,\quad \text{subject to } C_1,C_2,C_3,C_4. \tag{2}$$

여기서 벡터 y의 각 원소인 y_r은 이차원 행렬의 원소인 x_{ij}를 일차원으로 배열한 것으로 인덱스(index) i, j 와 r 를 수식 $r=i+(j-1)\times K$ 에 의하여 대응한 것이다. 행렬 $W=[w_{r_1r_2}]$는 $w_{r_1r_2}=a_{j_1j_2}b_{i_1i_2}$ 로써, 이때 $r_1=i_1+(j_1-1)\times K$, $r_2=i_2+(j_2-1)\times K$ 이다. 그러므로 수식 (1)의 해를 구하는 것은 주어진 제약조건들을 만족하면서 수식 (2)를 최소화하는 해 y 를 구하는 것과 동일하다.

수식 (2)는 앞으로 정의하는 벡터들과 상수들에 의하여 선형 문제로 변형이 가능하다. 이를 위하여 주어진 모든 제약조건들을 만족하는 분할 해의 집합 S 를 다음과 같이 정의하자.

$$S=\{y\mid y \text{ is a vector satisfying } C_1,C_2,C_3,C_4\}$$

S 를 이용하여 다음의 식을 만족하는 벡터 $\theta=\{\theta_1,\theta_2,\ldots,\theta_{KN}\}$를 구할 수 있다.

$$\theta_{r_1}\ge\sum_{r_2=1}^{KN} w_{r_1r_2}y_{r_2},\quad \forall y\in S,\quad 1\le r_1\le KN$$

또한, 임의의 k 번째 해인 $y^{(k)}\in S$ 를 위한 벡터 $\alpha^{(k)}=\{\alpha_1^{(k)},\alpha_2^{(k)},\ldots,\alpha_{KN}^{(k)}\}$와 상수 $\phi^{(k)}$를 다음과 같이

$$\alpha_{r_2}^{(k)} = \sum_{r_1=1}^{KN} w_{r_1 r_2}\, y_{r_1}^{(k)} + \theta_{r_2}\, y_{r_2}^{(k)}, \quad 1 \le r_2 \le KN$$

$$\phi^{(k)} = \sum_{r=1}^{KN} \theta_r\, y_r^{(k)}$$

정의하면, 수식 (2)에서 최적해 y^*를 구하는 것은 주어진 제약조건을 만족하면서 벡터 $\alpha^{(k)}$와 상수 $\phi^{(k)}$를 이용하는 다음의 수식 (3)의 최적해 y^*를 구하는 것과 동일하다[2].

$$\min_{y \in S} \; z, \text{ such that } z \ge \sum_{r=1}^{KN} \alpha_r^{(k)} y_r - \phi^{(k)} \qquad (3)$$

그러므로 본 논문의 분할 문제의 해를 구하는 것은 주어진 제약조건을 만족하면서 수식 (3)의 해를 구하는 것과 동일하다. 그런데 Buckard 등[3]이 수식 (3)과 같은 quadratic boolean programming 형식의 문제를 해결하는 휴리스틱 방법을 제안하였으며 그림 6에 그의 방법을 수식 (3)을 해결하는데 적합하도록 본 논문의 수식을 사용하여 표현한 알고리즘을 보인다.

그림 6의 알고리즘은 그림에 나타난 것처럼 반복적으로 해를 구하고 구한 해를 이용하여 단계 3과 5의 수식을 변형해가면서 다시 해를 구한다. 단계 7에서 M은 사용자가 정의한 반복회수이며 단계 6의 두 개의 출력 y^*와 z^*는 각각 수식 (3)의 해와 그때의 최소값이다. 그런데 실제로 분할 해를 구하기 위하여 해결하여야 하는 것은 단계 3과 5의 a와 h를 비용함수로 하는 문제이다. 두 식은 일반화된 할당 문제(generalized assignment problem)[13]이며 이를 해결하는 방법이 이 휴리스틱 방법의 가장 중요한 부분이다. 이 휴리스틱의 문제 해결 방법을 본 논문의 문제를 위한 분할 방법으로 사용하기 위해서는 본 논문에서 제시한 제약조건들을 추가하여야 하며 본 논문에 적합한 일반화된 할당 문제를 해결하는 방법이 필요하다. 이를 위하여 본 논문에서는 전체적인 Buckard의 휴리스틱 방법의 흐름을 이용하면서 단계 3과 5의 문제에 제약조건 C_2와 C_3을 추가하고 주어진 모든 제약조건들을 만족하면서 일반화된 할당 문제의 해를 구하는 모듈 할당 방법을 제안하였으며 이를 다음 절에 기술한다.

2. 분할 방법을 위한 모듈 할당 방법

본 절에서는 FPGA 기반 회로 분할 문제를 위한 제약조건들을 만족하는 그림 6의 단계 3과 5를 위한 모듈 할당 방법을 기술한다. 모듈 할당 방법은 일반화된 할당 문제를 해결하는 Martello등의 방법[13]에서처럼 유사하게 주어진 문제가 최소화되는 할당하는 방법을 통하여 해결할 수 있다. 그러나 본 분할 방법에서는 주어진 제한들이 많기 때문에 기존의 방법을 그대로 사용하기 곤란하므로 주어진 제약조건들을 빠른 시간에 고려할 수 있는 할당 방법을 구현하였다.

본 논문의 분할 방법에서는 앞에서 설명한 바와 같이 제약조건을 만족하면서 앞 절의 단계 3과 5의 문제에서 최소값을 갖는 해를 구해야 한다. 두 문제는 비용함수만 바꾸면 동일한 방법으로 해결할 수 있으므로 단계 3의 문제에 대한 할당 방법을 기술한다. 임의의 해 $y \in S$에서 임의의 성분 $y_r = 1$이면 인덱스 r은 식 $r = i + (j-1) \times K$에 의하여 인덱스 j와 i가 대응되므로 모듈 j가 칩 i에 할당됨을 알 수 있다. 그러므로 할당 방법에서 단계 3의 문제를 최소화하는 해를 구하는 것은 주어진 식을 최소화하는 모듈과 칩의 쌍들을 구하는 것과 동일하다. 그러므로 할당 방법은 인덱스 j와 i에 의하여 모듈과 칩으로 기술한다.

모듈 할당 방법에서는 모듈들을 순차적으로 칩에 할당한다. 모듈들은 할당할 칩과 함께 쌍을 구성하며 할당 우선 순위값인 할당순서가 정해진다. 각 모듈에 대하여 우선 순위값이 정해지면 우선 순위값에 의하여 순서대로 리스트를 형성하고 그 순서에 의하여 모듈은 쌍을 이루는 칩에 할당한다. 그림 7에 할당의 예를 보인다. 그림 7 (a)와 (b)는 각각 모듈 b의 할당 전과 후에 사용하는 핀수를 보인다. 그림에서 큰 타원안에 보인 작은 원들은 아직 할당되지 않은 모듈들을 나타낸다. 모듈 b의 할당으로 칩 C에는 한 개의 핀이 추가되었으며 칩 B는 모듈 a와 모듈 b간의 통과 네트와 통과 확률 $p_{AC}^{B} = 1$에 의하여 두 개의 핀이 추가되었다.

모듈과 모듈을 할당할 칩의 쌍을 구성하기 위해서는 먼저 모듈을 쌍을 이루는 칩에 주어진 제한을 모두 만족하면서 할당할 수 있어야 한다. 임의의 모듈 j가 칩 i로 할당될 수 있는지의 여부는 주어진 제약조건을 만족하는 지에 의하여 결정되며 이를 기호로 $feas(j, i)$로 표기한다. 만일 모듈 j가 칩 i로 할당될 때 모든 제약조건을 만족하면 $feas(j, i)$=TRUE이며 '모듈 j가 칩 i로 할당 가능하다'라고 한다. 반대로 제약조건이 만족하지 않을 경우에는 $feas(j, i)$=FALSE이며 '모듈 j가 칩 i로 할당 가능하지 않다'라고 한다. 외부 시그널을 갖

는 모듈들은 외부핀제약조건에 의하여 입출력 구현용 FPGA 칩으로만 할당 가능하다. 모든 모듈들은 크기제약조건에 의하여 배선 전용 칩으로는 할당 가능하지 않다.

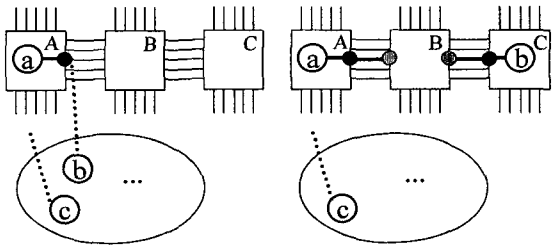


그림 7. 할당의 예
(a) 할당 전 (b) 할당 후

Fig. 7. An assignment example.
(a) Before the assignment.
(b) After the assignment.

주어진 제한을 만족하는 모듈과 칩의 쌍 중에서 우선적으로 할당할 쌍을 구하기 위하여 할당 우선 순위값을 정의하며 이를 f_{ij}로 표현한다. 할당 우선 순위값은 다음과 같이 계산한다. 먼저, 임의의 모듈 j를 위하여 할당이 가능한 칩의 집합을 만들고, 이 집합에서 모듈이 칩으로 할당될 때의 가장 적은 비용과 두 번째로 적은 비용을 갖는 칩 두 개를 선택한다. 여기서 비용이란 그림 6의 단계 3의 비용함수에서 가중치를 의미한다. 이때 가장 적은 비용을 갖게 하는 칩을 i라고 하면 그때의 두 비용의 차를 할당 우선 순위값 f_{ij}라고 한다. 이와 같은 방식으로 우선 순위값을 정하는 것은 두 비용의 차가 크다는 것이 모듈을 칩 i에 넣지 않고 다른 칩에 넣을 때에는 비용이 크게 증가할 수 있음을 의미하므로 가능한 비용의 차가 큰 것을 먼저 선택하여 할당하고자 하는 것이다. 모든 모듈에 대하여 같은 방법으로 할당 우선 순위값을 계산한다. 모든 할당 우선 순위값이 계산된 후에는 우선 순위값 중에서 가장 큰 값을 갖는 쌍부터 순차적으로 선택하여 해당하는 모듈을 칩에 할당하면 된다. 할당되지 않은 모듈과 칩에 대하여 가장 큰 우선 순위값을 $f_{i'j'}$로 표현하며 이에 해당하는 모듈 j'를 칩 i'에 먼저 할당한다.

모듈을 할당한 후에는 할당된 칩과 통과 네트에 의하여 사용 가능한 칩의 용량이나 핀수 등을 갱신하여야 하면 할당되지 않은 모듈들이 선택한 칩으로 할당 가능한지를 확인하여 필요한 경우 우선 순위값을 갱신하여야 한다. 그런데 모듈을 할당할 때마다 모든 칩의 제약조건이 만족되는지를 고려하는 것은 제약조건 C_2의 통과 네트에 대한 고려 때문에 많은 시간을 필요로 한다. 그런데 모듈을 할당하는 초기에는 칩들에 할당된 모듈이 적고 핀수제약조건을 모두 만족하므로 모듈을 할당할 때마다 제약조건을 확인하고 모든 모듈에 대하여 f_{ij}를 다시 계산할 필요가 없다. 그러므로 모듈 할당을 시작하기 전에 f_{ij}들을 구한 후 그들을 감소 순(decreasing order)으로 정렬한 리스트 F를 유지한다. 리스트 F의 순서에 의하여 가장 큰 값을 갖는 $f_{i'j'}$를 선택하고 모듈 j'를 칩 i'에 할당한다.

모듈 할당 후 칩의 사용 가능한 핀과 용량에 대한 갱신 방법은 다음과 같다. 모듈이 할당된 칩에 대해서는 칩의 용량 $c_{i'}$에서 모듈의 크기 $s_{j'}$를, 사용 가능한 핀수 $t_{i'}$에서 새로 추가되는 네트의 연결선의 수를 빼주면 된다. 여기서 칩 i'에 새로 추가되는 연결선의 수는 새롭게 추가되는 모듈에 연결된 전체 연결선의 수에서 이미 칩 i'안에 할당된 모듈들과의 연결선의 수를 뺀 값($a(j', i')$로 나타냄)이다. 할당된 모듈이 외부 시그널을 갖는 경우에는 입출력 구현용 FPGA 칩의 $b_{i'}$에서 외부로 나가는 시그널의 수 $g_{j'}$를 빼준다.

모듈 j'를 칩 i'에 할당할 때 통과 네트들이 존재하면 통과 네트가 통과하는 모든 칩들에 대해서도 사용 가능한 핀수 t_{i_1}를 수정하여야 한다. 통과 네트들은 이들의 연결을 위하여 그 네트들의 연결선의 수의 두 배 만큼의 핀수가 필요하므로 통과 네트의 연결선의 수가 $a_{j_1 j'}$인 경우에 t_{i_1}에서 $2\times p^{i_1}_{i_2 i'} \times a_{j_1 j'}$ 만큼의 핀수를 빼준다.

그림 8에 제안한 모듈 할당 방법의 알고리즘을 보인다. 할당 방법은 모든 모듈이 할당될 때까지 반복된다. 만일 리스트 F의 순서에 의하여 모듈을 반복적으로 할당할 경우에 기존의 할당된 모듈에 의하여 다음으로 할당할 모듈이 제약조건에 의하여 쌍을 이루는 칩으로 들어갈 수 없는 경우가 발생할 수 있으므로 모듈을 할당하기 전에 먼저 칩으로 할당이 가능한지를 확인한다(단계 5). 이와 같이 제약조건이 만족되지 않을 때에는 할당이 가능한 칩을 구한다(단계 5). 할당이 가능한 칩을 구할 수 없는 경우에는 다른 할당되지 않은 모듈들에 대하여 다시 할당 우선 순위값을 구하고 리스트를 재정렬한다(단계 3). 그러나 제약조건이 위배되지 않을

1. 반복변수를 $k=1$로 초기화한다.
2. 모든 할당되지 않은 모듈들에 대하여 feas(j, i) = TRUE이며 최소의 비용을 갖게하는 칩 i와 할당 우선 순위값 f_{ij}를 구하고 이들의 리스트 F를 구성한다.
3. F를 감소순으로 정렬한다.
4. F 중에서 가장 큰 값을 갖는 $f_{i'j'}$를 선택하고, $F = F - \{f_{i'j'}\}$한다.
5. 만일 (feas(j', i') = TRUE)이면 단계 6으로 간다.
   그렇지 않으면 모듈 j'를 위한 다른 할당 가능한 칩 i'를 구한다.
6. 모듈 j'를 칩 i'에 할당한다.
7. 칩 i'를 갱신한다; $c_{i'} = c_{i'} - s_{j'}$, $t_{i'} = t_{i'} - a(j', i')$, $b_{i'} = b_{i'} - g_{j'}$.
8. 모듈 j'와 네트로 연결된 할당된 모듈들 j_1에 대하여 통과 확률이 $p_{i_2 i'}^{i_1} > 0$가 되는 칩 i_1들에 대하여 핀수를 갱신한다; $t_{i_1} = t_{i_1} - 2 \times p_{i_2 i'}^{i_1} \times a_{j_1 j'}$.
9. 반복변수를 $k = k+1$로 증가시킨다.
   만일 ($k = N$)이면 중단한다.
   만일 (mod(k ,V) = 0)이면 단계 2로 간다. 그렇지 않으면 단계 3으로 간다.

그림 8. 할당 알고리즘
Fig. 8. Assignment alogorithm.

경우에도 할당되는 모듈에 의하여 할당되지 않은 다른 모듈들의 할당에 영향을 받게 될 수 있으므로 V만큼의 모듈이 할당될 때마다 다시 제약조건을 확인하고 리스트를 정렬한다(단계 9).

모듈 할당 알고리즘(그림 8)의 복잡도(complexity)는 단계 2의 제약조건 확인 과정과 정렬과정에 의하여 $O(K^2N + N\log N)$이다. 지금까지 기술한 전체 분할 방법의 복잡도는 그림 6의 행렬 연산에 의하여 $O(K^2N^2)$이다. 그러나 행렬들의 값들이 대부분 0이고 이들에 대해서는 연산을 하지 않으므로 실제의 복잡도는 표기한 복잡도보다 적은 시간에 분할이 수행된다.

Ⅳ. 실험결과

논문에서 제안한 분할 방법은 C언어로 구현하였으며 Sun Ultra-2에서 실험하였다. 표 2에 입력으로 사용한 MCNC Partition93의 회로들을 보인다. 이들은 Xilinx XC-3000 시리즈의 FPGA 칩[20]으로 기술사상(technology mapping)된 회로를 사용하였다. 입력으로 사용한 회로들은 클리크(clique) 모델로 변환하였으며 변환된 네트의 연결선의 수는 변환 전 네트에 속한 핀수를 'p'라고 할 때 1/p(p-1)로 하였다.

회로는 본문의 그림 1과 3에서 보인 세 가지의 위상 그래프에 대하여 실험하였다. 크기가 적은 회로 s5378과 s9234를 할당하기 위해서는 네 개의 FPGA 칩을, 나머지 큰 회로에 대해서는 열 여섯 개의 FPGA 칩을 사용하여 실험하였다. 각 회로를 주어진 FPGA 칩으로 균등하게 할당하기 위하여 FPGA 칩의 크기 용량을 실험을 통하여 결정하였다. IC 칩들간의 연결 가능한 연결선의 수와 핀은 실험을 통하여 회로의 크기에 비례한 값으로 정하였다.

입력에서 IOB들을 주입출력(primary I/O)을 갖고 보드 밖으로 연결되어야 할 시그널을 갖는 모듈로 하여 입출력 구현용 FPGA 칩에 할당될 수 있도록 하였다. 그림 3의 구조에서 입출력 구현용 FPGA 칩은 각 보드 구조에서 그들이 놓이는 위치에 존재한다고 하였으며[12,21], Mesh 구조에서는 보드의 가장자리에 놓인 FPGA 칩에 보드 외부로의 연결선들이 존재한다고 가정하였다.

표 3-5에 본 논문에서 구현한 분할 방법의 실험결과를 기존의 Shih 등의 분할방법[18]을 이용하여 구현한 결과와 비교하여 사용하였다. Shih 등의 방법에서는 외부 시그널에 대한 처리가 없으므로 IOB들을 본 분할 방법에서와 동일하게 처리할 수 있도록 그들의 방법을 수정하였다. 두 분할 방법에서 그림 6의 분할 방법의 반복회수 M을 100으로 하였다. 분할 후 분할 해에서

표 2. 입력회로
Table 2. Test circuits.

Circuits	*CLBs*	*IOBs*	*Nets*
s5378	381	84	626
s9234	454	41	714
s13207	915	152	1375
s15850	842	99	1262
s35932	2317	192	3360

표 3. Mesh 위상에서의 분할결과
Table 3. Partition results for Mesh topology.

Circuits	*Chips*	SK 분할			Our 분할		
		$Pins_{max}$(%)	$Conn_{total}$	*Time*	$Pins_{max}$(%)	$Conn_{total}$	*Time*
s5378	(2x2)	111.0	299	172	98.5	201	368
s9234	(2x2)	76.0	226	256	85.5	220	349
s13207	(4x4)	88.3	2215	1221	84.5	2013	1477
s15850	(4x4)	88.3	1975	1268	93.7	1942	1780
s35932	(4x4)	90.4	5346	3091	85.8	5168	4089
s38417	(4x4)	73.4	4638	9210	87.7	4358	11466

표 4. AXB-AP4 위상에서의 분할결과
Table 4. Partition results for AXB-AP4 topology.

Circuits	SK 분할			Our 분할		
	$Pins_{max}$(%)	$Conn_{total}$	*Time*	$Pins_{max}$(%)	$Conn_{total}$	*Time*
s5378	77.8	1221	159	97.2	897	233
s9234	61.9	1003	167	48.9	951	187
s13207	68.9	2014	1850	81.1	1810	2862
s15850	72.7	1803	1594	63.7	1529	2509
s35932	60.9	4508	3215	56.1	3322	6662
s38417	53.3	4338	11292	49.9	2538	16945

표 5. TM-2 위상에서의 분할결과
Table 5. Partition results for TM-2 topology.

Circuits	SK 분할			Our 분할		
	$Pins_{max}$(%)	$Conn_{total}$	*Time*	$Pins_{max}$(%)	$Conn_{total}$	*Time*
s5378	104.0	554	179	90.9	551	222
s9234	74.4	633	172	71.6	598	270
s13207	59.4	4798	3580	48.6	3558	5684
s15850	55.3	4590	1519	41.1	2962	2408
s35932	81.4	8323	5050	74.0	8269	5399
s38417	56.3	6437	11772	48.6	5112	17435

사용한 핀수와 연결선들의 수를 확인하기 위하여 미로 배선[11]을 사용하여 칩들간의 네트에 대하여 배선을 수행하였으며 이의 실험결과를 각 표에 나타낸다.

표 3-5에서 'SK 분할'과 'Our 분할'은 각각 Shih 등과 본 논문의 분할 방법의 실험결과를 나타낸다. 'Pinsmax'는 배선 후에 최대 핀수를 갖는 칩의 핀수 사용률을 나타낸다. 'Conntotal'은 네트들이 배선 시 사용한 연결선 수의 합으로 배선길이의 합을 나타낸다. 'Time'은 수행시간을 초 단위로 표시한 것이다. 표 2에서 'Chips'는 Mesh 구조에서 보드 상에 놓인 FPGA 칩들의 수를 (행, 열)로 나타낸 것이다.

표 3과 5에서 Shih 분할 방법의 경우에는 핀 사용률이 100%가 넘어서 배선이 완료될 수 없는 경우도 있으나 본 논문에서 구현한 분할 방법에서는 주어진 모든 위상그래프로 배선이 완료되면서 분할되었음을 알 수 있다. 본 논문의 방법에서 몇 개의 회로에 대하여 Pinsmax가 많게 나타난 것은 분할 시 최단 경로를 사용하여 배선하기 위하여 노력하였기 때문이며 전체적으로 네트들이 사용한 연결선의 수가 적으며 최대 34.7%의 향상을 보인다. 결과적으로 실험결과에서 볼 수 있듯이 통과 확률을 포함한 제약조건에 의하여 분할 시 배선이 고려된 분할 결과를 얻을 수 있음을 볼 수 있다.

V. 결 론

본 논문에서는 배선 위상이 이미 결정되어 있는 FPGA를 기반으로 하는 재프로그래밍이 가능한 시스템을 위한 분할 문제인 위상 기반 회로 분할 문제를 위한 새로운 quadratic boolean programming 수식을 제안하였다. 본 수식은 FPGA 기반 시스템의 제약조건들을 고려하면서 배선 길이의 합을 최소화하는 문제이다. 수식에서는 서로 인접하게 놓여 있지 않은 IC 칩간을 연결하기 위하여 다른 IC 칩을 통과하는 통과 네트를 추가적으로 고려하였다. 제안한 분할 문제는 선형의 할당 문제로 변환되었으며 할당 문제를 해결하는 효율적인 휴리스틱 방법을 기술하였다. 실험결과에 보인 것과 같이 실험에 사용한 모든 회로가 주어진 보드로 분할이 되었으며 다른 방법과 비교하여 분할 후 대부분의 회로에 대하여 핀 사용률이 적었으며 네트들이 사용한 배선 길이의 합이 최대 34.7% 적게 나타났음을 보였다. 앞으로 보다 빠른 분할을 위하여 회로를 클러스터링하는 방법이나 회로 지연을 고려하는 분할 방법에 관한 연구가 요구된다.

참 고 문 헌

[1] C.J. Alpert, J. Huang and A.B. Kahng, "Multilevel Circuit Partitioning", 34th DAC, pp. 530- 533, 1997.

[2] E. Balas and J.B. Mazzola, "Quadratic 0-1 Programming by a New Linearization", Presented at the TIMS/ORSA Meeting, 1980.

[3] R.E. Buckard and T. Bonniger, "A Heuristic for Quadratic Boolean Programs with Applications to Quadratic Assignment Problems", European Journal of Operational Research, pp. 374-386, 1983.

[4] V.C. Chan and D. Lewis, "Hierarchical Partitioning for Field-Programmable Systems", ICCAD, pp. 428-435, 1997.

[5] P.K. Chan, M.D.F. Schlag, J.Y. Zien, "Spectral K-Way Ratio-Cut Partitioning and Clustering", 30th DAC, pp. 749-754, 1993.

[6] N.C. Chou, C.K. Cheng. W.J. Dai and R. Lindelof, "Circuit Partitioning for Huge Logic Emulation Systems", 31th DAC, pp. 244-249, 1994.

[7] C.M. Fiduccia and R.M. Mattheyses. "A Linear-time Heuristic for Improving Network Partitions", 19th DAC, pp. 175-181, 1982.

[8] C. Kim and H. Shin, "A Performance-Driven Logic Emulation System: FPGA Network Design and Performance-Driven Partitioning", IEEE Trans. on CAD of ICs and Systems, Vol. 15, No. 5, May, 1996.

[9] Kleinhans, G. Sigle, F.M. Johannes and K.L. Antreich, "Gordian : VLSI Placement by Quadratic Programming and Slicing Optimization", IEEE Trans. on CAD, Vol. 10, No. 3, March, 1991.

[10] H. Krupnova, A. Abbara and G. Saucier, "A Hierarchy-Driven FPGA Partitioning

Method", 34th DAC, pp. 522-525, 1997.

[11] C. Lee, "An Algorithm for Path Connections and Its Applications", IEEE Trans. on Electric Computer, VEC-10, pp. 346-365, Sept. 1961.

[12] D.M. Lewis, D.R. Galloway, M. Ierssel, J. Rose and P. Chow, "The Transmogrifier-2 : A 1 Million Gate Rapid-Prototyping System", IEEE Trans. on VLSI Systems, Vol. 6, No. 2, June, 1998.

[13] S. Martello and P. Toth, Knapsack Problems, Chap. 7, pp. 189-220, 1990.

[14] B.M. Riess, H.A. Gieselbrecht, and B. Wurth, "A New K-Way Partitioning Approach for Multiple Types of FPGAs", Proc. of the Asia and Sourth Pacific DAC, pp. 313-318, 1995.

[15] L.A. Sanchis, "Multiple-Way Network Partitioning", IEEE Trans. on Computers. Vol 38. No 1. Jan, 1989.

[16] V. Sankarasubramanian and D. Bhatia, "Multiway Partitioner for High Performance FPGA Based Board Architectures", ICCAD, pp. 579-585, 1996.

[17] M. Shih and E.S. Kuh, "Quadratic Boolean Programming for Performance-Driven System Partitioning", 30th DAC, pp. 761-765, 1993.

[18] M. Shih and E.S. Kuh, "Circuit Partitioning by Capacity and I/O constraints", IEEE CICC, pp. 659-662, 1994.

[19] J.Y. Zien, P.K. Chan and M. Schlag, "Hybrid Spectral/Iterative Partitioning", ICCAD, pp. 436-440, 1997.

[20] Xilinx, Inc., The Programmable Gate Array Data Book, Xilinx, San Jose, 1992.

[21] Aptix, Inc., System Data Book, Aptix, 1993.

저 자 소 개

林 鐘 錫(正會員)第 36卷 C編 第 1號 參照
1981년 서강대학교 전자공학과 학사. 1983년 한국과학기술원 전기전자 및 전자공학과 석사. 1989년 Univ. of Maryland 전기공학과 박사. 1983년 3월 - 1990년 8월 한국전자통신연구소 연구원. 1990년 9월~현재 서강대학교 컴퓨터학과 정교수. 1999년~현재 서강대학교 정보통신대학원 부원장

崔 然 景(正會員)第 36卷 C編 第 1號 參照
1991년 서강대학교 전자계산학과 학사. 1993년 서강대학교 전자계산학과 석사. 현재 서강대학교 컴퓨터학과 박사과정. 1996년 3월~2000년 2월 경민대학 전자계산과 조교수. 2000년 3월부터 경민대학 컴퓨터정보계열 멀티미디어과 조교수