한국시뮬레이션학회 '99춘계학술대회 논문집 1999.4.17 동의대학교

PCB 회로의 분할 및 착색 알고리즘에 관한 연구

°김현호*, 이용희**, 장중식**, 이천희**
충북도립옥천대학*, 청주대학교**
(우:360-764) 청주시 상당구 내덕동 36 Tel : 0431-229-8448, Fax : 0431-213-6392
E-mail : yicheon@alpha94.chongju.ac.kr

A study on the Partition and Coloring Algorithm of the PCB Circuits

°Kim, Hyeon-ho*, Lee, Yong-hui**, Jang, Jung-sik**, Yi, Cheon-hee**
Okchon Provincial College*, Cheongju Univ.**

요약

시스템 레벨 PCB(Printed Circuit Board) 디자인은 최종적인 시스템 특성에 정확한 정보를 갖지 못하는 디자인 결정을 하기 위해 여러 가지 정보가 필요하다. 또한 분할 할 때 분할 시간과 방법은 매우 중요하고 합성 결과의 특성은 교환(tradeoffs)과 디자인 결정에 매우 민감하다. 그러므로 만일 디자인이 합성되고 단일 보드로 디자인된다 할지라도 후에 다중 보드로 분할 될 수 있다.

따라서 본 논문에서는 PCB 회로 디자인의 제약구동 방법 중 off-critical-path 분할기법을 사용한 휴리스틱(heuristic) 방법을 제안했고 교환 그래프 착색 알고리즘을 제안했다.

I. 서론

PCB(Printed Circuit Board) 회로 디자인 [1]연구에서 많은 교환(tradeoff)과 물리적인 디자인 특성 사이의 상호 연관성은 PCB 회로 디자인 결정에 많은 추정을 할 수 있다. 또한 분할할 때 분할 시간과 방법은 매우 중요한데 합성결과의 특성은 교환과 디자인 결정에 매우 민감하다. 만일 디자인 합성이 수행되고 단일 보드로 디자인된다 할지라도 뒤에 다중 보드로 분할 될 수 있다. 하위 디자인 합성을 하기 위한 변화는 초기 디자인 결정이 분할 경계에 대한 정보 없이 디자인되기 때문이다. 시간 스텝 스케줄링은 클럭 싸이클에 대한 최소, 최대 시간 제약을 포함한다. 최대 클럭 싸이클 시간은 처리량(throughput)과 지연시간 제약 그리고 시간 계획 때문에 존재한다. 만일 합성 후 분할을 수행한다면 최대 클럭 싸이클 제약을 위반하는 통신 지연시간의 삽입 때문에 분할 실행이 불가능하다. 합성 후 통신 지연시간의 삽입은 처리량 또는 지연시간이 제약을 위반하는 클럭 싸이클 시간을 이용하기 위해 결정된다. 이러한 지연시간은 멀티플렉싱, pad, 그리고 배선 지연시간에 연관된 보드의 제한 때문에 발생한다.

짧은 시간에 해결할 수 있는 문제는 모듈 선택, 계획, 할당과 분할, 그리고 상호 배선

에 주안점을 두는 데이터 패스(data-path) 디자인 합성에 있다.

따라서 본 논문에서는 PCB 회로의 디자인 결정을 하기 위한 제약 구동 방법 중 off-critical-path 분할 기법을 사용한 휴리스틱(heuristic) 방법을 제안했고 교환 그래프 착색 알고리즘을 제안했다.

II. 연관된 이론

그래프 이론의 분할 기술은 Gupta와 De Micheli[2]에 의해서 정립된다. 디자이너는 안전하게 분할 시작점을 수작업으로 찾아야 한다. 그리고 디자인에 대한 초기 계획이 진행된 다음 Kernighan-Lin 알고리즘 또는 Simulated Annealing 기법은 분할을 마무리 하기 위해 사용된다.

Vahid와 Gajski는 분할기술[3,4,5]을 사용하여 핀수, 면적, 그리고 성능 분할 과정에 대한 특정분할 방법을 제한했다. 면적과 성능 그리고 핀 제약에 대한 추정은 분할을 유도하기 위해 사용된다. 상위-노드의 면적과 지연시간 추정은 합성과 추정 기술의 조합에 의해서 그리고 시뮬레이션에 의해서 얻을 수 있다.

III. 디자인 원리

1. 분할 휴리스틱

본 논문에서 제안된 방법에는 탐색 공간 크기를 감소시키기 위한 소거 방법 두 가지가 있다. 하나는 처리량, 지연시간, 그리고 면적 제약을 종속시키기 위해 각각 독립적으로
분할을 수행하는 것이고 다른 하나는 분할의 특성을 산정 하는 것인데 이것은 다음과 같은 간단한 정리에 의해서 언급된다.

정리 1 : 보드에서 분할되는 i를 고려하자. 가능한 전체 디자인에 사용되는 분할 k의 실행 면적 A_k는 다음과 같은 경계 면을 갖는다.

$$A_k \leq \text{보드 면적 제약} - \sum_{i \neq k} A_{i,min}$$

여기서 $A_{i,min}$은 분할 i의 최소 실행 면적이다.

정리 2 : 분할 i의 k^{th} 수행 지연시간 $D_{j,k}$를 고려하자. 식 $D_{i,min} = \min_{k} (D_{i,k})$이 주어진 분할 i의 최소 수행 지연시간 $D_{i,min}$을 고려하자. 분할 i에 대한 $D_{i,k}$를 제외한 모든 분할에 대해서 $D_{i,min}$을 사용한 전체 디자인의 임계-패스 지연시간 $CP_{i,k}$를 고려하자. 만일 $CP_{i,k}$ ≥ 전체 지연시간 제약, 이면 분할 i의 k^{th} 수행은 가능한 전체 디자인에 사용할 수 없다.

정리 1에서 어떠한 분할 수행 면적은 분할이 할당된 보드에 점유할 수 있는 최대 면적에 의해서 경계된다. 최대 면적은 같은 보드에서 다른 분할이 가능한 가장 작은 면적으로 수행될 때 성취된다. 분할 수행이 선택됐으면 정리 2는 성취할 수 있는 최소 시스템 딜레이를 정의한다. 만일 성취된 시스템 딜레이가 선택된 것보다 만족하지 못하면 분할 결과에 영향을 미치지 않는 범위에서 제거할 수 있다.

연속적인 수행 선택은 연속적인 경우에 시스템 시간지연은 최소화된다. 이러한 선택은 off-critical -path 분할(시간을 고려한 지연시간 실행에 관계되는 off-critical)의 연속성을 나타낸다. 이 알고리즘은 그림 1에서 언급한다.

<처리 가능한 분할 실행>

W_i는 추정될 분할 i의 실행이라고 가정.
L_i는 W_i의 초기 구간이라고 가정.
S는 연속적인 후보 분할이라고 가정.
오름차순으로 각각의 분할 i에 대해서 W_i를 정렬(초기 구간과 시스템 시간지연에 대해서), 최종 면적에 대한 W_i를 정렬.

```
for(각각의 가능한 초기 구간 ℓ)
 for(각각의 분할 i)
    분할 i의 정렬된 추정에서 첫 번째 추
      정 실행에 대한 Wi를 초기화.
    Li ≥ ℓ 일 때까지 각각의 Wi를 실행 시
      키고 Wi는 Li ≤ ℓ 일 때 non-pipeline
      실행임.

   while (TRUE)
     IF ∃ 어떤 Wi가 초기구간 ℓ을 갖는
        실행이 아니면
          break
     ℓ과 WiS를 사용한 통합 시스템을 추정.
      IF 추정 결과가 가능하다면
        모든 추정 결과를 기록.
        모든 추정 list에 집합 S를 찾음.
      else
        면적 제약을 갖는 칩의 분할에서
        집합 S는 마지막 시스템 통합 추정
          에 의해서 위배됨.
        하위 특성을 갖거나 ℓ을 갖고 실행
          할 수 없는 추정 포인트를 skip
          하기 위해 S에 순방향으로 분할
          Wi 를 이동시킴.
      for(S에 대한 각각의 분할)
        S에 대한 각각의 분할 list에 순방향
          으로 Wi를 이동시킴.
        딜레이 스케쥴링을 사용하기 위해
          예상되는 시스템 딜레이를 찾음.
        Wi의 값을 재 저장.
        최소 시스템 딜레이에 결정되는 분
          할을 기록.
      IF ∃ 기록된 분할에 대해서 분할을
          순서적으로 나열함.
      lese
          순서 없이 분할을 나열함.
 End
```

그림 1. 분할 평가에 대한 반복적인 휴리스틱

2. 교환 그래프 착색 알고리즘

그림 2는 6개의 노드에 대한 그래프를 나타낸다. 이 그래프는 4 3 6 1 5 2 교환에 대한 반전 그래프이다. 방법을 알아보기 위해 4번을 고려해보자. 원래의 순서 1 2 3 4 5 6에서 4번은 1 2 3 다음에 나타난다. 4 3 6 1 5 2 교환에서 4번은 1 2 3 다음에 나타난다. 이러한 교환에서 4번은 1 2 3에 관련해서 옳은 위치에 나타나지 않는다. 그러므로 다음 3개의 절선 41, 42, 43은 반점 그래프에 대한 교환도(permutation diagram)를 나타낸다. 이것은 그림의 상단에 1 2 3 4 5 6 하단에 4 3 6 1 5 2를 기록한 후 같은 숫자를 가지는 점들을 연결함으로써 얻어진다. 교환도, 교환 그래프 및 교환자체는 모두 동일한 정보에 대한 대체표현들이다. 이와 같은 표현들 중 하나를 고려해 보면 다른 것들도 유도될 수 있다.

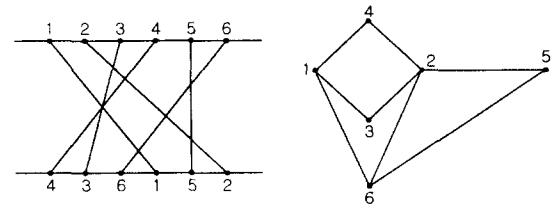

그림 2. 교환 그래프와 해당 교환

교환 그래프를 착색하기 위한 순차 알고리즘은 Galumbic에 의해 기술되었다[6,7]. N-노드 교환 그래프에 대한 최적 착색을 얻기 위해서는 O(NlogN)시간이 요구된다. 알고리즘에 대한 입력은 그래프 G에 대한 교환표현 Π이다. 동일한 칼라로 주어질 수 있는 모든 노드들은 동일 큐로 삽입된다. λ큐들은 알고리즘의 마지막에 생성된다. i번째 객체를 삽입할 때 알고리즘은 바쁘게 시도하고 기존 큐가 사용될 수 있는지를 조사한다. Πi는 큐내 최근의 요소는 Πi보다 적으므로 이와 같은 큐들 중 하나로 들어갈 수 있다. 그 절차는 이와 같은 특성을 만족하는 첫 번째 큐로 Πi가 새로운 큐에서 첫 번째 객체로 만들어진다.

2진 검색(binary search)은 Πi가 삽입될 큐를 위치시키기 위해 이용될 수 있다. 그러므로 i≤N에 대해 요소를 삽입하기 위해서는 복잡도는 O(NlogN)이다. 그림 2는 이러한 기술을 설명해 준다. 예제에 대한 교환 표현은 3 6 1 5 2이다. 노드 4는 우선적으로 검사되어 큐1로 삽입된다. 다음 차례로 노드 3이 된다. 3은 4보다 적기 때문에 노드 3은 그래프에서 노드 4와 절선을 공유한다. 따라서 두 번째 큐가 생성되어 3은 큐 2로 삽입된다. 노드 6은 4보다 크기 때문에 큐 1로 삽입될 수 있다. 순차에서 다음 차례인 노드 1은 큐1이나 큐2로 들어갈 수 있다. 마지막으로 노드 2는 큐3으로 삽입될 수 있다. 그 그래프에 대한 채색 수는 3이고 최적 착색은 다음과 같다. 노드 4, 6에 대해서 칼라 1, 노드 3, 5에 대해 칼라 2 그리고 노드 1과 2에 대해서는 칼라 3이다.

병렬 알고리즘은 교차 그래프들을 착색하기 위해 설계될 수 있다. 알고리즘은 N개의 프로세서들에 대해 선형 배열을 사용한다. 각각의 프로세서는 N+1개의 레지스터를 구성하는 지역 메모리를 가진다. 이들은 각각은 $\log_2 N$ 비트의 길이를 가지며 큐를 형성한다. 그것은 만약 프로세서가 $\log_2 N$ 비트 수를 비교할 능력을 가지며 큐 삽입 알고리즘용 내장(Built-in)논리를 가진다면 충분하다. 프로세서들은 파이프라인 형식으로 동작한다. 교환 표현은 프로세서 1에 대한 입력으로 순차 $\Pi_1 \cdots \Pi_N$으로 제공된다.

이 알고리즘이 실행되기 전에 각각의 프로세서내의 레지스터 R_0는 0으로 초기화된다. 각각의 프로세서는 그림 3과 같이 동일한 코드를 실행한다. 알고리즘의 끝 부분에서 프로세서 i의 현지 큐 내에 모든 요소들을 칼라 i를 가진다. 그림 4는 이와 같은 예를 보여준다.

1,2 …, N 각각의 프로세서에 대해서 N 번 병렬로 수행

begin
 왼쪽 프로세서에 i번째 요소를 받음.
  만일 i>R_0
   begin
     local 큐에 i를 삽입.
     오른쪽 프로세서에 0을 보냄.
   그밖에
     오른쪽 프로세서에 i을 보냄.
 end

그림 3. 교환 착색을 위한 파이프라인 알고리즘

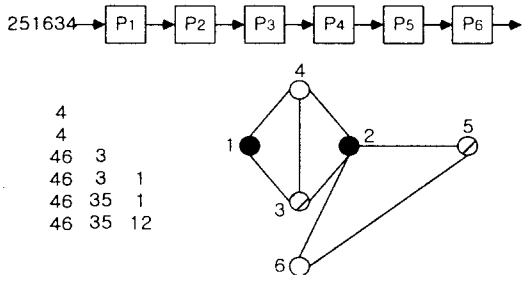

그림 4. 교환 그래프 착색에 대한 예

O(NlogN)인 순차 알고리즘에 비해 얻어진 속도는 logN이다. 병렬 알고리즘에 대한 프로세서-시간 적은 $O(N^2)$이다. 그러므로 병렬 알고리즘은 최적이 아니다. 이것은 순차 알고리즘이 Πi 요소를 삽입하기 위한 큐를 찾기 위해 2진 검색을 사용하기 때문이다. 그러나 병렬 알고리즘은 Πi요소를 삽입하기 전에 가능한 i-1개의 모든 큐들을 통해 조사한다. 이것은 $O(N^2)$작업과 O(N) 병렬시간을 요구한다. 최적병렬 알고리즘은 교환 그래프들을 착색하기 위해 설계되어 질 수 있을까? 그와 같은 알고리즘은 N개의 프로세서를 사용하며 O(logN) 실행시간이 요구된다.

IV. 결론

본 논문에서는 PCB 회로 디자인의 제약 구동 방법 중 off-critical-path 분할 기법을

사용한 휴리스틱(heuristic) 방법을 제안했고 교환 그래프 착색 알고리즘을 제안했다. 이러한 기술은 개개의 분할 면적, 처리량(throughput) 그리고 시스템의 지연시간과 같은 제약을 만족하고 다중 보드 분할 과정을 쉽게 한다.

정확한 상호 연결 면적과 시간지연 추정 기술, 분할 기술, 디자인 방법, 비용 등을 고려한 방법들은 계속 연구해야할 것이며 자동 배선과정, 저소비전력, 그리고 신뢰성 등을 고려한 가장 적은 비용을 들인 보드-레벨 PCB 회로 디자인을 제작하기 위해 사용될 것이다.

참고 문헌

[1] Charles J. Alpert and Andrew B.Kahng. "Multiway partioning via geometric embeddings, orderings, and dynamic programming". Transsation on Computer-Aided Design of Integrated Circuits and Systems, 14:1342-1358, 1995.

[2] R. Gupta and G. De Micheli. "Partitioning of Functional Models of Synchronous Digital Systems". In Proc. Int'l Conf. on Computer-Aided Design, p.p 216-219. IEEE, November 1990.

[3] Lung-Tien Liu, Ming-Ter Kuo, Chung-Kuan Cheng, and T.C. Hu. "A replication cut for two-way partitioning", Transaction on Computer-Aided Design of Integrated Circuits and Systems, 14:623-630, 1995.

[4] Frank M. Johannes, "Partition of VLSI Circuits and Systems", In Proc. 33rd Conf. on Design Automation, pp83-87, June 1996.

[5] B.M.Riess, K.Doll and F.M.Johannes, "Partioning very large circuits using analytical placement techniques, Proc. ACM/IEEE Design Automation Conf., pp646-651, 1994.

[6] M.C. Galumbic, "Algorithm graph theory and perfect graphs", Academic Press, NY, 1988.

[7] Lung-Tien Liu, Ming-Ter Kuo, Chung-Kuan Cheng, and T.C. Hu. "Performace-Driven Graph Approach". In Proc. 32nd Conf. on Design Automation, pp206-210, June 1995.