퍼지이론을 이용한 FPGA 회로의 효율적인 테크놀로지 매핑

이 준 용† · 박 도 순††

요 약

테크놀로지 매핑은 VLSI 설계자동화(CAD) 시스템의 한 단계로서, 설계된 회로를 논리적 단계에서 물리적 단계로 매핑해 준다. 테크놀로지 매핑의 효율성은 매핑된 회로의 지연시간과 회로의 면적에 의해서 평가되어진다. 특히 순차회로에서는 레지스터 사이의 조합회로의 최대지연시간에 의해서 전체회로의 지연시간이 결정된다. 본 논문에서는 순차회로에 대한, 건설적인(constructive) 단계와 반복적인(iterative) 단계의 리타이밍 기술과 퍼지 논리에 의해 향상된 FPGA 매핑 알고리즘을 소개한다. 주어진 초기회로는 건설적인 방법에 의하여 FPGA회로로 초기매핑 되어진 후 반복적인 리타이밍에 의하여 매핑회로의 효율을 높이게된다. 초기회로에 주어진 여러가지 기준들은 결정 함수(Decision Making Function)에 대한 퍼지 이론 규칙의 계층적인 구조로 구성된다. 제안된 매퍼는 MCNC 벤치마커의 실험을 통해 지연시간과 면적에서 기존 매핑시스템의 성능을 능가함을 보여준다.

Efficient Technology Mapping of FPGA Circuits Using Fuzzy Logic Technique

Jun-Yong Lee† · Do-Soon Park††

ABSTRACT

Technology mapping is a part of VLSI CAD system, where circuits in logical level are mapped into circuits in physical level. The performance of technology mapping system is evaluated by the delay and area of the resulting circuits. In the sequential circuits, the delay of the circuit is decided by the maximal delay between registers. In this work, we introduce an FPGA mapping algorithm improved by retiming technique used in constructive level and iterative level, and by fuzzy logic technique. Initial circuit is mapped into an FPGA circuit by constructive manner and improved by iterative retiming. Criteria given to the initial circuit are structured hierarchically by decision-making functions of fuzzy logic. The proposed system shows better results than previous systems by the experiments with MCNC benchmarkers.

1. 서 론

테크놀로지 매핑은 VLSI 설계자동화(CAD) 시스템의 한 단계로서, 설계된 회로를 논리적 단계에서 물리적 단계로 매핑해 준다. 테크놀로지 매핑의 효율성은 매핑된 회로의 지연시간과 회로의 면적에 의해서 평가되어진다. 특히 순차회로에서는 레지스터 사이의 조합회로의 최대지연시간에 의해서 전체회로의 지연시간이 결정된다.

대다수의 FPGA에 관한 매핑 방법 기술은 각 CLB가 5개의 입력까지의 부울 방정식을 구현할 수 있는 SRAM LUT의 2^5 bits를 가진 XILINX XC3000 FPGA

※ 이 논문은 1997년 한국학술진흥재단의 공모과제 연구비에 의하여 연구되었음.
† 정 회 원 : 홍익대학교 컴퓨터공학과 교수
†† 종신회원 : 홍익대학교 컴퓨터공학과 교수
논문접수 : 2000년 3월 29일, 심사완료 : 2000년 8월 10일

회로 구조에 대하여 부울 네트워크를 매핑하기 위해 고안되었다. FPGA가 더욱 대중화됨에 따라서 FPGA 구조가 발전되어 왔다. XILINX XC4000[6] 이나 AT&T의 ORCA[7]와 같은 새로운 SRAM-based FPGA 구조에서는 단일 CLB의 구성(configuration)이 더욱 복잡하게 되어 있다. 본 연구에서는, 최근 사용되는 FPGA 중에서 가장 보편적인 타입중 하나인 Xilinx XC4000 시리즈를 매핑을 위한 타겟으로 선택하였다. Xilinx XC4000 FPGA의 경우, 한 개의 CLB는 3개의 LUT들로 구성되는데, 최대 9개의 입력이 허용된다.

이 연구에서는 이와같은 복합적인 CLB구조를 가진 FPGA를 위한 FMS(Fuzzy Logic Mapper for Sequential Circuits)를 소개한다. FMS 매핑 알고리즘 기술은 CLB에 게이트들의 매핑을 위해 퍼지 논리를 적용한다. 퍼지 논리 접근 방식은 일찍 몇몇 레이아웃(layout) 애플리케이션[8, 9]를 위한 CAD 시스템에서, 그리고 우리의 이전 연구[10]에서 논리 회로를 FPGA로의 매핑하는데 성공적으로 사용되었다.

2. 문제 정의

테크놀로지 매핑 시스템은 DAG형태로 입력된 회로를 FPGA회로로 구현한다. 이 시스템은 주어진 DAG의 노드를 타겟(target) FPGA의 CLB로 할당한다. 이러한 매핑의 결과인 FPGA회로는 타겟(target) FPGA 구조의 모든 제한조건을 만족하여야 한다. 본 연구에서는 현재 FPGA에서 가장 많이 사용되고 있는 Xilinx XC4000 제품군을 매핑에 있어서의 타겟으로 선택하였다.

테크놀로지 매핑에는 여러 목표가 주어질 수 있다. 그 중에서 가장 중요한 것은 사용되는 CLB의 갯수의 최소화, 시간지연(timing delay)의 최소화, 그리고 결과 회로의 연결성(routability)의 실현성 등이다.

이와 같은 목적 중에서 가장 중요한 것은 CLB의 개수를 늘이지 않으면서 지연시간을 개선하는 것이다. 지연시간의 경우, 룩업테이블(look-up-table) 방식의 FPGA에 있어서, 각각의 CLB에서 상당한 시간의 일정한 지연이 발생하기 때문에 전체 지연시간을 줄이기가 쉽지 않다. 그렇지만 임계경로(critical path)상의 연결된 CLB의 갯수를 줄일 수 있다면, 지연시간은 개선될 수 있다.

3. 리타이밍을 사용한 순차회로의 최적화

리타이밍이란, 회로의 기능은 그대로 유지하면서 클럭 피리어드를 최소화하기 위해 레지스터를 재배치하는 회로의 최적화 기법이다. 이 기법은, Leiserson et. al [2]에서 처음 소개되었는데, 이 문제에 대한 여러가지 요소들에 대하여 연구하고 있다. 그들은 클럭 피리어드를 최소화하는 리타이밍을 결정하는 선형시간(polynomial time) 알고리즘과 전체 레지스터의 갯수를 최소화하는 알고리즘을 제시하였다.

이 논문에서는 다중그래프 $G=<V, E, v_h, d, w>$를 사용하여 결과를 유도하였는데, 이 그래프 G는 각 노드(vertex)와 간선(edge)에 가중치가 주어져 있고 방향성이 있으며, 루트(root)가 주어지는 유한 그래프로 가정되었다. 여기서 V는 회로의 기능적 요소인 노드들의 집합을 나타내고, 전파지연(propagation delay) $d(v)$가 각 노드에 주어진다. 간선(edge)은 레지스터의 갯수 $w(e)$로 가중치(weight)를 주었다.

그래프 G에서의 경로는 일련의 노드들과 간선(edge)으로 정의된다. 레지스터 카운드 함수는 경로에 대해서도 정의된다. 경로가중치(path weight) $w(p)$는 경로 p상에서의 간선(edge)의 가중치의 합으로 정의된다.

$$w(p) = \sum_{i=0}^{k-1} w(ei).$$

그리고 경로지연 $d(p)$는 경로 p상에서의 각 노드들의 지연의 합으로 정의된다.

$$d(p) = \sum_{i=0}^{k} d(Vi).$$

추가적인 조건으로는 각각의 노드에 대한 전파지연이 음수가 아니어야 하며, 각 간선(edge)에 대한 레지스터 갯수도 음수가 아니어야 한다는 것이 있다. 또한, 그래프 G에는 가중치가 0가 아니며, 싸이클이 없다고 가정한다. 이 모델에서 클럭 피리어드는 선형시간 알고리즘으로 정의된다. 리타이밍 변환은 레지스터의 재배치를 통하여 클럭 피리어드를 줄인다. $r(v)$는 각 노드 v의 회로에서의 출력간선(out-edge)로부터 입력간선(in-edge)으로 이동되는 레지스터의 갯수를 나타낸다. 회로 G에서 r의 리타이밍은 리타이밍된(retimed) 그래프 G_r이 다음 조건들을 만족할때 적합하다.

1) 레지스터 갯수 $w(e)$가 모든 간선(edge)에 대하여

음수가 아니다.

2) G의 임의의 방향있는 싸이클에 있어서, 양수의 레지스터 갯수를 갖는 간선(edge)이 있다.

변환된 그래프에서, 각 간선(edge)에 대한 가중치 w_r은 다음과 같이 정의된다.

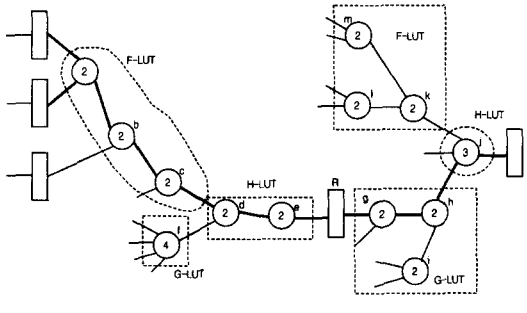
$$wr(e) = w(e) + r(v) - r(u)$$

이때, v와 u는 간선(edge)의 시작점과 끝점이다.

리타이밍된 회로의 최소 클럭 피리어드 $P(G)$는 임계경로(critical path) 상의 최대 전파지연에 의해서 결정된다. 각 노드와 연관되는 차이 $r(v)$의 값의 형태에 있어서 최적 리타이밍은 제한 그래프 상의 단일 최단 경로 알고리즘에 의해서 결정된다.

Malik et.al [3]은 리타이밍을 이용한 순차회로의 다른 최적화 방법을 제안했다. [2]에서는 조합회로 요소의 수정에 대한 언급이 없었던 반면에, [3]에서는 조합회로를 최적 순차회로로 재합성하고 있다. 각 FPGA 생산자가 레지스터의 숫자와 같은 LUT와 CLB에 허용되는 입력과 출력과 같은 CLB 구조를 자신들만의 모델을 가지고 있다. 그들은 서로가 모두 다르다. 일반적인 리타이밍 알고리즘은 이러한 FPGA 구조에 대한 제한사항을 고려하지 않았으며, FPGA의 매핑에도 효율적이지 않다.

(그림 1)은 일반적인 타이밍 최적화 알고리즘이 FPGA의 매핑에 있어서 부적합한 결과를 보여주는 경우에 대한 예이다. 전파지연은 조합회로의 노드에서 발생한다. 각 노드에 대한 지연은 입력의 숫자와 비례관계에 있다. (그림 1)에서 모든 노드에 대한 입력은 레지스터의 출력이거나, PI라고 가정한다.



(그림 1) 순서회로의 매핑

경로지연은 노드의 지연의 합으로 가정한다. 처음과 두 번째 단계에 대한 최대 경로지연은 각각 10과 7이다. 일반적인 리타이밍 알고리즘은 레지스터 R을 노드 e 이전으로 옮기려고 할 것이다. 이것은 최대 경로지연을 10에서 9로 줄이게 된다.

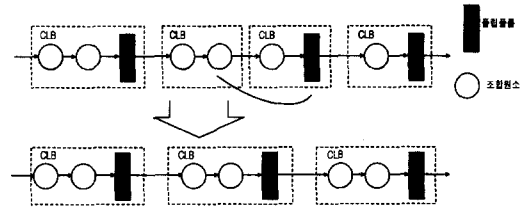
그러나 이와같은 일반적인 리타이밍 연산은 FPGA 활용에 부정적인 영향을 주게된다. 주어진 그래프에서 각 단계의 조합회로는 단일 CLB에 매핑시킬 수 있다. 각 단계에 대한 최대지연은 각각 CLB 레벨이 1로 결정된다.

그러나 레지스터 R을 노드 e 이전으로 옮기는 리타이밍 연산은 두 번째 단계의 조합회로가 하나의 CLB로 매핑되지 못하게하는 결과를 야기하게 된다. 이것은 두 번째 단계에서 CLB 레벨이 1에서 2로 증가하는 결과를 가져온다.

4. 해결 방법의 개요

제안된 알고리즘은 두 모듈로 구성된다. 테크놀로지 매핑과 리타이밍으로 주어진 회로의 타이밍을 향상시키기 위해 반복된다. 조합회로의 매핑을 위한 이 테크놀로지 매핑 알고리즘은 순차회로의 매핑으로 확장된다. 기본 차이점은 레지스터도 또한 CLB 속으로 매핑되고, 타이밍 지연은 PI와 PO 사이보다는 플립플롭 사이로 계산된다.

리타이밍 모듈은 플립플롭을 재배치하여 수정된 프로시져는 연속된 플립플롭 사이의 최대 타이밍 지연을 적게 한다. 앞 절에서 언급한대로 존재하는 지연 최적화 알고리즘들은 FPGA에 대해 적용되지 않는다. FPGA를 위해, 회로는 룩업 테이블로 구현된다. 이때 룩업 테이블의 지연은 룩업 테이블로 매핑되는 게이트의 숫자와 무관한 상수이다. 여기서는 매핑 프로시져에서 타이밍을 향상시키는 지역 회로 변환을 정의한다.



(그림 2) FPGA 매핑에서의 매핑

제안된 알고리즘에서, 플립플롭은 그래프 상에서 노드로 표현된다. 이 노드들은 앞으로 혹은 뒤로 타이밍

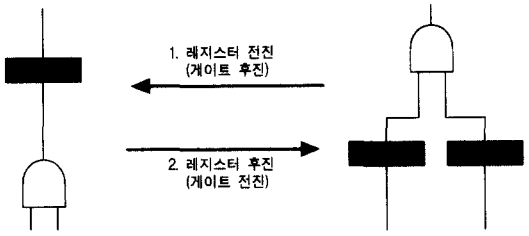
지연을 줄이기 위해 간선(edge)을 따라서 움직인다. (그림 2)는 이 리타이밍 알고리즘의 기본 아이디어를 보여준다.

4.1 기본 연산

다음과 같은 네가지 기본 리타이밍 연산을 조합 회로나 출력점(fanout point) 사이의 플립플롭을 재배치하는 지역 변환을 위해 정의한다.

4.1.1 레지스터 후진(Move-backward)(게이트 전진(Move-forward))

이 연산은 레지스터를 (그림 3)에서 묘사한 것처럼 게이트의 출력으로부터 게이트의 입력으로 움직인다.



(그림 3) 리타이밍의 기본 연산

이전에 정의된 기호를 사용하면, 게이트 g에 대한 r의 값은

$$r(g) = 1$$

이 된다. 그리고, 외부간선 e_{out}과 내부간선 e_{in}에 있어서 간선(edge)의 무게는 다음과 같다.

$$w_r(e_{out}) = w(e_{out}) - 1$$
$$w_r(e_{in}) = w(e_{in}) + 1$$

외부간선에서 레지스터의 존재는 처음에 $w_r(e_{out}) \geq 1$과 같이 보여진다. 레지스터가 뒤로 움직인다음, 합법적인 리타이밍인 $w_r(e_{out}) \geq 0$ 조건을 여전히 만족한다.

이 연산의 결과로 레지스터의 개수는 (k-1)만큼 증가한다. 이때, k는 게이트의 입력의 숫자이다. 이 연산은 게이트를 레지스터의 앞으로 이동으로 볼 수 있다.

4.1.2 레지스터 전진(게이트 후진)

이 연산은 레지스터의 후진 연산과 반대이다. 입력쪽의 모든 레지스터는 게이트와 정렬된 독립 레지스터를 따라서 전진한다. 게이트 g의 리타이밍은

$$r(g) = -1$$

이고 그 결과는 다음과 같다.

$$w_r(e_{out}) = w(e_{out}) + 1$$
$$w_r(e_{in}) = w(e_{in}) - 1$$

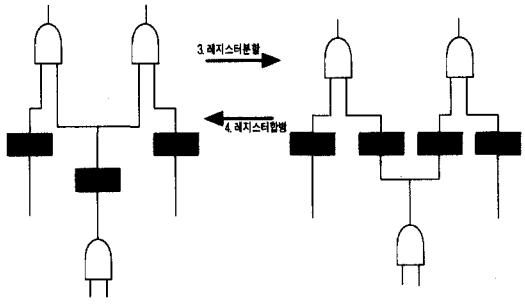
이 연산을 적용하기 위하여 게이트의 모든 입력은 레지스터의 출력에서부터 온다.

이 경우, 합법적인 리타이밍의 조건을 만족하는 게이트 g의 모든 입력간선에 대하여 $w_r(e_{out}) > 0$가 성립한다.

이 연산은 레지스터의 개수를 (k-1)만큼 감소시킨다. 이는 또한 입력 레지스터를 하나로 병합하는데 있어서 게이트를 뒤로 움직이는 것을 설명한다.

4.1.3 레지스터 분할

회로가 레지스터의 출력으로부터 출력선을 포함할 때, 레지스터는 (그림 4)에서 보는 것처럼 분기 출력선(fanout branch)으로의 분할이 가능하다. 이 연산은 리타이밍 연산이 아니지만, "레지스터 전진" 연산에 있어서 만족해야할 선결 조건으로 가끔 필요하다. 이 연산은 출력선의 개수에 활용되는 레지스터의 개수를 증가시킨다. 이 연산은 $w_r(e_{out}) \geq 0$의 조건을 만족한다.



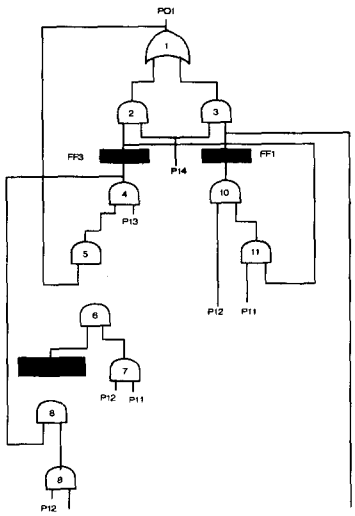
(그림 4) 리타이밍의 기본연산 (계속)

4.1.4 레지스터 병합

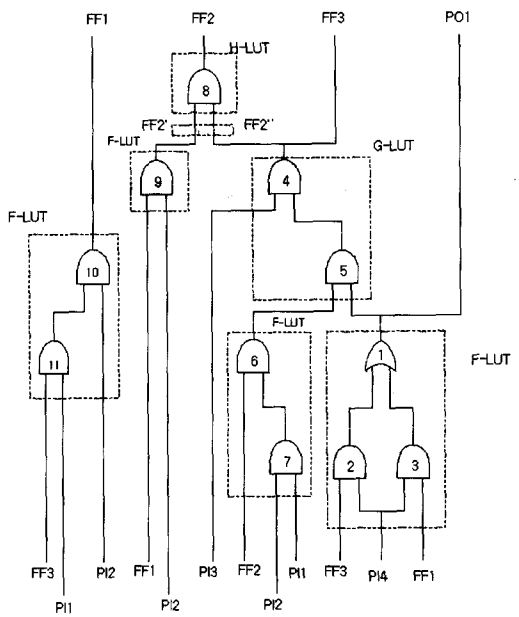
이것은 "레지스터 분할" 연산의 반대이다. 이 연산은 과다한 레지스터의 병합을 하거나 레지스터와 게이트 사이의 출력선 포인트(fanout point)를 제거하여 "후진(move-backward)"가 가능하게 한다. 이 변환은 한 게이트의 출력에서 다른 게이트로의 입력으로 가는 경로에서 레지스터의 개수가 동일하도록 유지시켜준다.

4.2 리타이밍의 예

(그림 5)에 주어진 회로는 이 알고리즘을 증명하기 위해서 선택하였다. 그 회로는 (그림 6)에서 PO와 레지스터의 입력이 위로, PI와 레지스터의 출력이 아래로 간 형태로 다시 나와있다. (그림 6)과 같이 회로를 재정형하여 임계경로를 찾는 것이 더욱 쉬워졌다.



(그림 5) 순서회로에서 플립플롭의 초기 위치

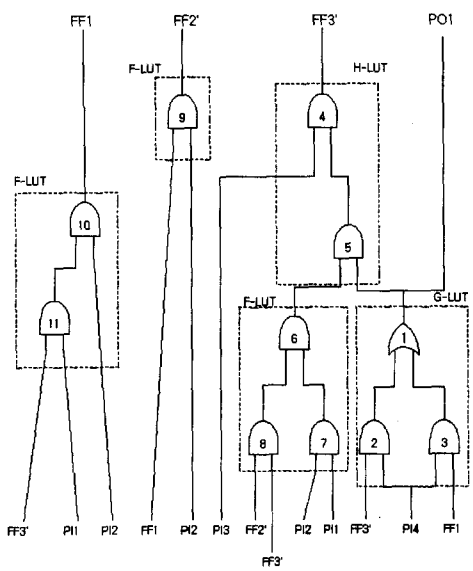


(그림 6) 순서회로의 초기 FPGA 매핑

FPGA로의 초기 매핑은 루트노드에서 시작되는 건설적(constructive) 알고리즘으로 행하여졌다. 초기 매핑이후에, 회로는 CLB 레벨의 숫자와 연관되어 임계경로 정의하고, 임계 경로에 포함되어있는 노드를 찾는 것으로 평가된다. 주어진 예제 회로에서, 임계경로는 게이트 8-4-5-6-7, 8-4-5-1-2, 혹은 8-4-5-1-3에 의해서 만들어진다. 이 세 경로들은 매핑 후에 CLB레벨 두 개를 사용한다.

클럭 피리어드를 최소화하기 위하여, 게이트 8은 임계경로로부터 제거될 수 있다. 몇몇 기본 리타이밍 연산은 이를 얻을 수 있다. 첫째, "게이트의 전진" 연산은 게이트 8에 (그림 6)에서 보여지는 것처럼 입력측에서 두 플립플롭 FF2'와 FF2"를 생성하는데 사용되었다. 그리고, "레지스터 병합" 연산은 FF2"와 FF3에 FF2'를 생성하기 위해 수행되었다.

이러한 리타이밍 연산 이후에 매핑이 다시 수행된다. 결과적으로, 새로운 임계경로는 게이트 4-5-6-8, 4-5-6-7, 4-5-1-2, 4-5-1-3으로 만들어진다. (그림 7)에서 게이트 4, 5는 H-LUT에 매핑될 수 있고, 게이트 6, 7과 게이트 1, 2, 3은 같은 CLB의 F-LUT와 G-LUT에 매핑될 수 있다. 결과적으로, (그림 7)에서와 보여지는 것처럼 CLB 레벨이 2에서 1로 줄어듦에 따라서 클럭 피리어드는 감소될 수 있다.



(그림 7) 리타이밍 후 순서회로의 최종 매핑

5. 알고리즘의 서술

제안된 테크놀로지 매핑 알고리즘은 주어진 DAG를 FPGA 포맷에 영역과 시간적 지연을 만족하면서 매핑한다. 입력회로로 CLB로의 조합 부분 매핑이 우선 수행된다. 매핑의 과정에 있어서, 선택된 기준은 요구조건에 따라 임의로 강조될 수 있다.

초기 매핑 이후에, 시간적 지연을 줄이기 위해서 플립플롭이나 노드를 재배치하는 리타이밍 프로시져가 수행된다. CLB-매핑 프로시져와 리타이밍 프로시져는 리타이밍이 시간적 지연에 있어서 더 나은 결과를 얻지 못할때까지 반복된다. (그림 8)은 순차회로에 대한 매핑 알고리즘의 의사코드(pseudo-code)이다.

5.1 CLB-매핑 프로시져

CLB 네트웍으로의 DAG 매핑은 조합회로 매핑에 사용되는 것과 유사한 구조적인 휴리스틱(constructive heuristic) 알고리즘에 의해 수행된다.

이전 매핑 프로시져와의 차이점은 순서회로 매퍼(mapper)가 입력 DAG의 플립플롭을 CLB로 매핑하는 것이다.

```
Initialize membership functions for Fuzzy Rules and
Preference Rules
ending_condition = 0
Call CLB_map();
  /* procedure CLB_map maps a DAG to a network of CLBs */

Anayse the result and compare it with stated goals
If ( the result satisfy goals )
  then ending_condition = 1;

While ( ending_condition = 0 ) {
  Call Retiming();
  Call CLB_map();
  Analyse the result and compare it with stated goals
  If ( the result satisfy goals or no more improvement is
       expected )
    then ending_condition = 1;

}
```

(그림 8) 순차회로의 리타이밍에 대한 매핑 알고리즘의 의사코드

퍼지로직(Fuzzy logic) 기술은 조합회로에서 CLB로의 매핑에 있어서 노드를 선택한다. 순서회로 매핑에 사용되는 퍼지로직 규칙과 표준은 조합회로 매핑에 사용되는 것과 동일하다.

5.2 리타이밍 프로시져

리타이밍 프로시져는 DAG를 FPGA로 매핑한 이후에 타이밍을 향상시켜준다. 매핑의 결과로부터, CLB의 최대 레벨의 수는 계산되어진다. 프로시져의 목표는 DAG를 수정하여 최대 레벨의 수를 다음 매핑의 단계에서 줄이는 것이다.

최대 CLB 레벨이 계산되면, 최대 CLB 레벨을 가지고 경로상에 있는 CLB를 찾게된다. 그러면 최대 레벨을 가지고서 CLB에 매핑된 노드셋을 찾게된다. 이 노드셋은 조합회로의 서브그래프(subgraph)를 이루게되고, 이 서브그래프의 출력은 레지스터, 혹은 PO의 입력과 연결되며, 서브그래프의 입력은 레지스터, 혹은 PI로부터 나온다.

이 서브그래프로부터 리타이밍 위한 후보노드(candidate node)는 퍼지회로를 이용한 각 노드의 시간 임계성(timning criticality)에 의해서 결정된다. 이 노드는 서브그래프의 크기를 줄이기 위해서 재배치된다. 제안된 시스템에서 앞 절에서 언급한 것과 동일한 연산으로, 플립플롭의 재배치보다는 노드의 재배치를 고려한다.

노드의 재배치를 위해서 그 노드는 주어진 위치에서 앞으로 혹은 뒤로 움직이게 된다. 만약 노드가 서브그래프에서의 루트(root)라면, 그 노드의 출력은 레지스터 혹은 PO로 연결된다. 만약 노드가 루트가 아니라면, 노드는 서브그래프의 크기를 줄이기 위해서 register보다 앞으로 이동될 수 있다.

만약 후보노드가 서브그래프의 최외곽노드(leaf node)라면, 그 노드의 입력은 레지스터 혹은 PI와 연결된다. 만약 노드의 모든 입력이 레지스터와 연결되었다면, 그 노드는 서브그래프의 크기를 줄이기 위해서 register보다 뒤로 이동된다. 그러나 후보노드의 이동이 언제나 가능하지는 않다.

```
V ← all nodes included in the CLBs which belong to the path
      with the largest CLB levels
Q ← candidate nodes from CLB levels

While ( Q ≠ Ø ) {
 Choose a node n from Q which maximizes timing criticality
 D(x)
  /* D(x) is a multi-criteria decision function and x is an
  element in the solution space X */

 If ( the output of node n is connected to input(s) of register )
 {
   If ( number of registers connected to n > 1)
```

```
    Call Merge_register to merge all registers into one
    Call Move_forward(n);
    /* procedure Move_forward moves the node n forward
    over the regisger */
  }
else if ( all the inputs of the node n is connected to the outputs
of registers )
  {
   for each register whose output is connected to the node
n
    If ( number of fanouts from the register > 1 )
     Call Split_register to duplicate the register for its
     fanouts
    Call Move_backward(n);
    /* procedure Move_backward moves the node n
    backward over the register */
  }
Remove the node n from Q
}
```

(그림 9) 순서회로의 매핑 알고리즘의 의사코드

노드의 "전진" 연산을 수행하려면, 그 노드의 출력은 레지스터와 바로 연결되어야 한다. 노드의 출력이 하나이상의 레지스터와 연결되어 있다면, 그 레지스터들은 "레지스터 병합" 연산에 의해서 하나의 레지스터로 병합된다.

노드의 "후진" 연산을 수행하려면, 그 노드의 입력은 레지스터로부터 출력점(fanout)이 없이 연결되어야 한다. 레즈스터가 몇 노드로의 출력점을 가지고 있다면, 그 레지스터는 "레지스터 분할" 연산에 의해서 각 출력점당 하나의 레지스터가 되도록 분할되어야 한다. (그림 9)는 리타이밍 알고리즘의 의사코드이다.

6. 실험 결과 및 결론

순서회로를 위한 제안된 테크놀로지 매퍼(mapper)는 약 9500줄의 C 코드로 구현되었다. 매퍼의 효율성을 나타내기 위하여, 16개의 벤치마크 회로를 순서회로를 위한 MCNC 시험케이스(test case)에서 무작위로 선택하였다.

실험의 결과는 <표 1>에 나와있다. 테크놀로지 매핑의 결과는 Xilinx의 배치(placement), 경로지정(routing)으로 완성되었다.

첫 세개의 열은 사용된 CLB의 개수를 나타내고 있으며, 초기 매핑이후의 임계경로의 지연과 최대 CLB 레벨은 시험케이스로 수행되었다. 두 번째 세 개의 열은 매핑과 리타이밍의 반복 이후의 최종 매핑의 결과를 보여준다. 마지막 세 개의 열은 같은 시험케이스에 대해서 Xilinx 시스템(매핑+물리적 디자인(physical design)에 의해서 얻어진 결과를 나타내고 있다.

〈표 1〉 실험 결과

회로명	제안된 FMS						Xilinx의 매핑시스템		
	초기 매핑			리타이밍 후					
	CLB의 갯수	최대 레벨	지연 (ns)	CLB의 갯수	최대 레벨	지연 (ns)	CLB의 갯수	최대 레벨	지연 (ns)
s1488	157	5	62.2	155	4	51.3	168	6	82.5
s1494	153	5	58.3	157	4	52.5	167	6	80.6
s208	17	4	46.1	17	4	46.7	19	4	46.1
s298	23	4	48.0	26	4	42.9	20	4	51.2
s344	24	5	52.6	24	5	55.1	27	6	61.3
s349	26	6	65.2	24	5	56.4	28	6	56.9
s382	27	4	51.4	28	4	51.1	29	4	50.9
s386	38	4	52.1	38	3	43.5	39	6	55.3
s400	27	4	51.8	28	4	50.6	30	4	56.6
s420	24	5	57.3	33	5	57.1	46	6	69.5
s444	33	6	62.8	32	6	64.3	36	6	64.6
s510	53	4	54.2	55	4	53.7	64	7	78.8
s526	48	4	51.7	50	4	50.2	46	4	57.2
s820	85	4	56.2	88	3	45.8	90	4	58.5
s832	90	4	55.5	92	3	44.9	90	5	61.3
s838	78	9	90.4	76	8	82.9	98	10	100.7
비율	0.924	0.891	0.912	0.938	0.816	0.844	1	1	1

최종 결과는 제안된 시스템이 Xilinx의 테크놀로지 매핑보다 영역과 시간적인 면 모두에서 더 나음을 보여준다. 제안된 FMS 시스템에서는 배치 이후에 실제 지연의 15.6%의 감소를 보이면서 최대 CLB 레벨의 개수가 18.4% 줄어들었다. 이 시스템은 또한 사용 CLB의 개수도 6.2% 줄였다. 이 실험은 제안된 매퍼가 Xilinx 매핑의 결과보다 리타이밍과정이 없이도 시간적인 면에서 약 10%정도 나은 결과를 생산한다는 것을 보여준다. 리타이밍 단계는 초기 매핑 결과를 10%정도 추가적으로 향상시킨다.

참 고 문 헌

[1] R. K. Brayton, R. Rudell, A. Sangiovanni-Vincentelli, and A. R. Wang, "MIS, a multiple-level logic optimization system," *IEEE Transactions on Computer-Aided Design*, Vol.CAD-6, pp.1062- 1081, Nov. 1987.

[2] C. E. Leiserson and J. B. Saxe, "Retiming Synchronous Circuitry," *Algorithmica*, Vol.6, No.1, pp.5-35, 1991.

[3] S. Malik et. al., "Retiming and Resynthesis : Optimizing Sequential Networks with Combinational Techniques," *IEEE Transaction on CAD*, Vol.10, No.1, January 1991.

[4] T. Miyazaki et. al., "Performance Improvement Technique for Synchronous Circuits Realized as LUT-Based FPGAs," *IEEE Transaction on VLSI Systems*, Vol.3, No.3, September 1991.

[5] N.-S. Woo, "ATOM : Technology Mapping of Sequential Circuits for Lookup Table-based FPGAs," Technical Report, AT&T Bell Laboratories, November 1991.

[6] *Xilinx Programmable Logic Data Book*, Xilinx Inc., San Jose 1998.

[7] N.-S. Woo, "A Study on the structure of the Intermediate Network in FPGA Technology Mapping," Oxford 1991 International Workshop on FPGAs, pp.170-178, 1991.

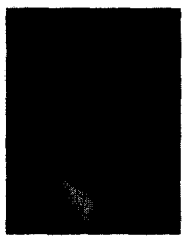
[8] R. Lin and E. Shragowitz, "Fuzzy Logic approach to placement problem," Proc. of the 29th Design Automation Conference, pp.153-158, 1992.

[9] J.-Y. Lee and E. Shragowitz, "Performance Driven technology mapper for FPGAs with Complex Logic Block Structures," Proceedings of IEEE International Symposium on Circuit and Systems (ISCAS), Vol.2, pp.1219-1222, Seattle, WA, 1995.

[10] J.-Y. Lee and E. Shragowitz, "Technology Mapping for FPGAs with Complex Logic Block Structures by Fuzzy Logic Technique," Proceedings of the Asia South-Pacific Design Automation Conference (ASP-DAC), pp.295-300, Makuhari, Japan, 1995.

[11] J. Cong and Y. Ding "On Area/Depth Trade-off in LUT-based FPGA Technology Mapping," Proc. of the ACM/IEEE 30th Design Automation Conference, pp.213-218, Jun. 1993.

[12] J. Cong and Y. Ding "An Optimal Technology Mapping Algorithm of Delay Optimization in LUT-based FPGA Designs," Proc. of International Conference on Computer-Aided Design, pp.48-53, Nov. 1992.



이 준 용

e-mail : jlee@cs.hongik.ac.kr

1986년 서울대학교 공과대학 컴퓨터공학과 졸업(학사)

1988년 미국 University of Minnesota 대학원, Dept. of Computer Sci. & Eng. (석사)

1996년 미국 University of Minnesota 대학원, Dept. of Computer Sci. & Eng.(공학박사)

1996년~1997년 미국 IBM Staff 연구원

1997년~현재 홍익대학교 컴퓨터공학과 교수

관심분야 : VLSI Design, CAD, 컴퓨터구조

박 도 순

e-mail : dspark@cs.hongik.ac.kr

1978년 서울대학교 공과대학 전자공학과 졸업(공학사)

1980년 한국과학기술원 전산학과 졸업(이학석사)

1988년 고려대학교 수학과 (이학박사)

1980년~1983년 국방과학연구소

1983년~현재 홍익대학교 컴퓨터공학과 교수

관심분야 : 컴퓨터구조, 설계자동화