순차회로를 위한 효율적인 FPGA 매핑

이준용
홍익대학교 컴퓨터공학과

Performance Driven FPGA Mapping of Sequential Circuits

Jun-Yong Lee
Dept. of Computer Engineering, Hongik University

요약

네크놀로지 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 방법에의하여 FPGA회로로 초기매핑되어진후 반복적인 리타이밍에 의하여 매핑회로의 효율을 높이게된다. 초기회로에 주어진 여러가지 기준들은 결정 함수(Decision Making)에 대한 퍼지 이론 법칙의 계층적인 구조에 의해 연결되어져 있다. 제안된 매퍼는 MCNC 벤치마커의 실험을 통해 지연시간과 면적에서 기존 매핑시스템의 성능을 능가함을 보여준다

1. 소개[1)]

최근까지 조합 회로를 위한 많은 매핑 알고리즘 기술이 제안되어 있지만, 순차 회로에 관해서는 몇과 적은 수의 알고리즘만이 소개되어 있다 더욱이 순차 회로 매핑에 직접 응용할 수 있는 매핑 알고리즘도 많지않다.

그 중에는 Xilinx의 XNFMAP과 PPR, 그리고 AT&T의 ATOM 있다. Xilinx의 XNFMAP과 PPR은 조합과 순차 논리 모두를 위한 XC3000 과 XC4000 FPGA를 위한 매핑을 수행한다 또한 [12]에 따르면 ATOM 이라고 불리는 AT&T의 시스템은 XC3000과 AT&T의 FPGA를 위한 순차 회로를 처리할 수 있다 이러한 매핑 알고리즘 기술은 사용하는 CLB의 개수를 줄이는 것이 주된 초점이다 ATOM 시스템은 플립플롭을 노드 타입으로 고려하고 조합회로에서 응용하는 유사한 방법으로 순차회로를 매핑한다

Miyazaki et. al [10] 는 순차 회로 매핑을 위한 일반적 성능 개선 기술을 제안했다 불락 스피드와 데이터 처리량을 개선하기 위해, 매핑된 그래프의 구조를 바꾸지 않고 주어진 순차 회로 안에 레지스터를 삽입하였다 두 가지 제안, "loop shrunking" 과 "register insertion"이 수행된다 "loop shrunking"은 매핑을 위한 전처리(예비처리)로서 주어진 입력 그래프 G에 응용된다. 이 단계에서 레지스터 삽입 지점 graph theoretical 문제를 해결함으로써 결정된다

그 다음에, 프로시저 "register insertion"은 불락 사이클(Clock cycle)을 줄이기 위해 가능한 많은 레지스터를 모든 경로(path)에 삽입한다 이 기술은 회로의 처리량을 개선할 수 있다 그러나 이는 불락 사이클이나 레이턴시(latency)의 수를 증가시킨다 이 기술은 FPGA 구조의 부여된 제약을 처리할 수 없는 이유로 인하여 종종 그래프로 레지스터를 삽입하는 것을 하지 못한다

대다수의 FPGA에 관한 매핑 방법 기술은 각 CLB가 5개의 입력까지의 부울 방정식을 구현할 수 있는 SRAM LUT의 2^5 bits를 가진 XILINX XC3000 FPGA 회로 구조에 부울 네트워크를 매핑하기 위해 디자인된다 FPGA가 더욱 대중화됨에 따라서 FPGA 구조는 발전되어 왔다 XILINX XC4000[13] 이나 AT&T의 ORCA[3]와 같은 새로운 SRAM-based FPGA 구조에서는 단일 CLB의 구성(configuration)은 더욱 복잡하게 되어 있다 오늘날 FPGA 중에서 가장 보편적인 타입중 하나인 Xilinx XC4000 시리즈는 매핑을 위한 타겟으로 선택되어진다 Xilinx XC4000(CLB는 3개의 LUT들로 구성)는 9개의 입력에 대해 사용되어진다.

이 논문에서는 복합 CLB를 가진 FPGA를 위한 FMS(Fuzzy Logic Mapper for Sequential Circuits)를 소개한다. 매핑 프로시저 FMS 기술의 입력은 부울 방정식과 레지스터의 DAG이다.

매핑 알고리즘 FMS 기술은 CLB에 게이트들의 매핑을 위해 퍼지 논리를 적용한다 퍼지 논리 접근 방식은 일찍 몇몇 레이아웃(layout) 애플리케이션[7],[4]를 위한 CAD에서 그리고 우리의 이른 연구[6]에서 논리 회로를 FPGA로의 매핑에서 성공적으로 사용되었다

2. 리타이밍 과정의 개요

리타이밍 모듈(retiming module)은 수정된 회로가 연속적인 플립플롭 사이의 최대 지연 시간(retiming delay)을 줄이기 위해 플립플롭을 재배치한다. 전에 언급한 것에 따라 지연 시간을 최적화하기 위한 현존의 알고리즘은 FPGA(look-up 테이블로 매핑되어진 게이트들(gates)의 개수에도 불구하고 look-up 테이블의 지연 시간이 일정한)를 기반으로 하는 look-up 테이블에 적용되어질 수 없다 여기서 우리는 매핑 프로시져에서 타이밍을 향상시킬 수도 있는 논리식으로 동등한 지역 회로 변환들(local circuit transformations)을 정의한다

1) 이 논문은 1997년 한국학술진흥재단의 공모과제 연구비에 의하여 연구되었음

이 모델에서 플립플롭들은 노드들의 특정 타입(type)으로써 DAG에 추가되어진다. 이러한 노드들은 지연 시간을 줄이기 위해 에지(edge)를 따라 앞뒤로 이동되어진다. 그림 2는 우리의 리타이밍 알고리즘의 기본 직인 생각을 보여주고 있다.

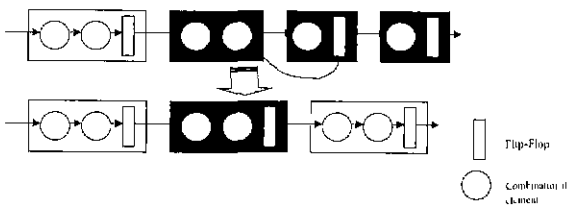

그림 2 . FPGA 매핑에서의 리타이밍

2.1. 기본 연산(Basic Operations)

4개의 기본 리타이밍 연산들(retiming operations)은 조합 게이트들 또는 팬아웃 지점들(fanout points)을 가로질러 플립플롭들을 재배치한 지역 변환에 대해서 정의되어진다. LUT들과 CLB들에 대한 모든 제약 조건들은 이 연산이 수행될 때 만족되어진다고 가정되어 있다

1. Move-backward a register(or Move-forward a gate)

이 연산은 그림 3에서 보여주는 것과 같이 레지스터(register)를 게이트의 출력으로부터 게이트의 입력으로 이동한다. [8]에서의 보편적인 리타이밍 기법을 사용한 게이트 g에 대한 r의 값은

$$r(g) = 1$$

이다

그러므로, out-edge e_{out}과 in-edge e_{in}에 대해 에지 가중치(edge weight)는 다음과 같이 된다

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

out-edge 위에 레지스터가 있는 것은 초기적으로 $w_r(e_{out}) \geq 1$ 이라는 것을 나타낸다. 레지스터가 뒤로 이동된 뒤에는 합법적인(legal) 리타이밍에 대해 $w(e_{out}) > 0$ 이라는 조건이 여전히 만족되어진다. 이 연산은 레지스터의 앞으로 게이트를 이동시키는 것으로 보여질 수 있다.

2. Move-forward a register (or Move-backward a gate)

이 연산은 "Move-backward a register" 연산의 역이다. 입력 쪽에 있는 모든 레지스터들을 게이트를 가로질러 앞으로 이동되어지고 하나의 레지스터로 통합된다. 게이트 g에 대한 리타이밍은

$$r(g) = -1$$

이고 결론적으로

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

이다

이 연산을 적용하기 위해서 게이트의 모든 입력들은 레지스터의 출력들로부터 와야 한다. 이 경우에 게이트 g의 모든 입력 에지들(edges)에 대해 $w(e_{in}) > 0$ 은 합법적인 리타이밍의 조건을 만족한다. 이것 또한 게이트를 뒤로 이동하고 반면에 입력 레지스터는 하나로 통합된 것으로 해석되어질 수 있다

3 레지스터 분할(Split a register)

회로가 레지스터의 출력으로부터 팬아웃(fanout)을 가지고 있을 때 그 레지스터는 그림 6에서처럼 팬아웃 갈래(fanout branches)로 분할될 수있다. 이 연산 자체가 리타이밍 연산은 아니다. 그러나"move-forward register" 연산을 위한 예비 조건으로 필요하다. 이 연산은 팬아웃(fanout)의 수에 사용되는 레지스터의 수를 증가시킨다. 이 연

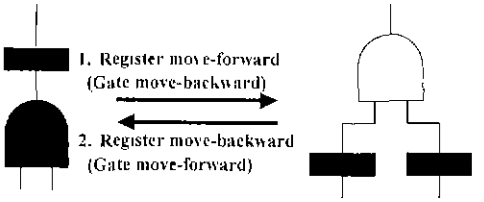

그림 3 리타이밍에 대한 기본 연산

산은 $w_r(e_{out}) \geq 0$ 이라는 조건을 만족한다

4. 레지스터 합병(Merge registers)

이 연산은 레지스터 분할 연산의 역이라 할 수 있다. 이 연산은 불필요한 레지스터들을 합병하거나 레지스터와 게이트 사이의 팬아웃 지점(fanout point)을 감소시키는 "move-backward" 연산을 가능하게 하는데 사용된다. 이 연산은 한 게이트의 출력으로부터 또 다른 게이트의 입력까지의 경로에 있는 레지스터의 개수를 그대로 유지한다

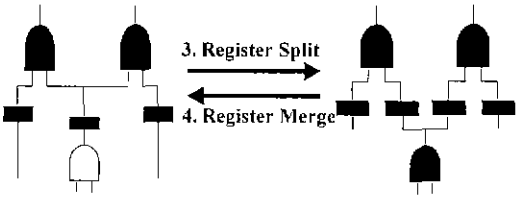

그림 4 리타이밍에 대한 기본 연산(계속)

3. 매핑 알고리즘

입력 회로에 대해서 CLB로의 조합과 순서 부분의 매핑이 먼저 수행된다. 매핑의 과정에서 선택된 기준은 preference 법칙(rule)에 의해 강조될 수 있다

초기 매핑 후에 시간 지연을 줄이기 위한 플립플롭(flipflop)과 노드들을 재배치하는 리타이밍 프로시저(procedure)가 적용된다. CLB-map-procedure와 리타이밍 프로시저는 시간 지연에 대해서 더 이상의 개선이 나타나지 않을 때까지 반복된다

리타이밍 프로시저는 DAG를 FPGA 형태로 매핑한 후에 시간의 향상을 위하여 DAG를 수정한다. 매핑의 결과로부터 CLB의 최대 레벨(level)의 수가 계산된다. 이것은 DAG를 수정해서 최대 레벨(level)의 수가 다음 번 매핑에서 줄어들도록 하기 위함이다

일단 CLB 레벨(level)의 최대수가 초기 매핑 후에 계산되면 최대 레벨(level)과 관련된 경로에 있는 CLB들이 발견된다. 그리고 나면 가장 긴 경로에 있는 CLB들에 배정되어 있는 노드들이 확인된다.

이 노드들의 집합은 레지스터나 PO의 입력에 연결된 게이트들의 출력과 함께 조합 게이트들의 서브그래프를 형성한다. 서브그래프에 있는 입력들은 레지스터나 PI들로부터 나온다

이 서브그래프로부터 리타이밍을 위한 후보 노드들이 각 노드들의 시간 임계도에 따른 퍼지 논리 룰에 의해 선택된다. 이 노드들은 서브그래프

의 크기를 줄이기 위해 재배치된다. 제안된 시스템에서 우리는 전 section에서 보여진 것처럼 플립플롭들을 재배치하는 것보다 노드들을 재배치하는 것을 고려하였다.

노드들을 재배치하기 위해서는 각 노드들이 기존의 위치로부터 앞으로 또는 뒤로 이동되어야만 한다. 만약에 노드가 서브그래프의 루트 노드라면 그 노드의 출력은 레지스터나 PO에 연결되어야 한다. 또 레지스터에 직접적으로 연결된 노드는 서브그래프의 크기를 줄이기 위해 레지스터 앞쪽으로 이동되어야 한다.

어떤 노드에 "move-forward" 연산을 수행하기 위해서는 그 노드의 출력을 레지스터에 직접 연결해야 한다. 그 노드의 출력이 하나 이상의 레지스터에 연결되어 있을 때 그 레지스터는 "merge register" 연산에 의해 하나의 레지스터로 합병되어야 한다.

어떤 노드에 "move-backward" 연산을 수행하기 위해서는 그 노드의 각 입력이 팬아웃(fanout)이 없는 레지스터로부터 나와야 하고 그 레지스터가 여러 개의 노드에 팬아웃(fanout)을 가지고 있을 때 "split-a register" 연산에 의해 각 팬아웃(fanout)에 대해 하나씩 레지스터가 연결되도록 여러 개의 레지스터로 분할되어야 한다. 만약에 선택된 후보 노드에 대한 리디이밍 프로시져가 실패하면 다음 후보 노드가 조사된다. 새로운 시간(timing)이 전보다 나쁘지 않으면 그 결과가 받아들여진다.

4. 실험결과

제안된 순차회로매핑 시스템에 대한 입력 회로들은 BLIF(Berkeley Logic Interchange Format)으로 기술된 매퍼(mapper)의 성능을 알아보기 위해 16개의 벤치마크(benchmark) 회로들을 순차 회로에 대한 MCNC 실험 회로들로부터 임의적으로 선택하였다. 이러한 실험 회로들을 사용하여 기술 매핑(mapping)과 리디이밍(retiming)을 수행한다. 최종 기술 매핑(mapping)의 결과는 Xilinx의 XNF(Xilinx's Netlist File) 형식으로 표현된다. 이 결과들은 Xilinx의 placement와 routing 시스템(system)에 의해 FPGA 구조로 완성된다.

먼저 Xilinx의 매핑 결과를 얻기 위해, 선택된 BLIF 형식의 MCNC 실험 회로들은 각각 Xilinx의 XNF 형식으로 변환되고 Xilinx의 기술 매핑(mapping), placement, routing 시스템에 사용된다.

표 1은 실험의 결과들을 보여주고 있다. 첫 번째 세 개의 열은 매핑(mapping)과 리타이밍(retiming)의 반복후의 최종 매핑(mapping)의 결과들을 나타낸다. 마지막 세 개의 열은 똑같 실험 예들에 대한 Xilinx 시스템에 의해 수행된 결과들을 보여주고 있다.

긴빈적인 결과들은 제안된 시스템이 Xilinx의 기술적인 매핑 시스템보다 공간(area)과 시간(timing)면에서 성능이 더 우수함을 보여준다. 제안된 순차회로 매핑 시스템은 최대 CLB 수준(level)의 개수를 18.4% 줄였다. 이것이 레이아웃(layout)후에는 실제적인 지연 시간을 13.6% 줄이는 결과를 만들어냈다. 우리의 시스템은 또한 사용된 CLB의 개수를 6.2% 줄였다.

Circuit Name	FMS			Xilinx's PPR		
	No. of CLBs	Max. Levels	Delay (ns)	No. of CLBs	Max Levels	Delay (ns)
s1488	155	4	51.3	168	6	82.5
s1494	157	4	52.5	167	6	80.6
s208	17	4	46.7	19	4	46.1
s298	26	4	42.9	20	4	51.2
s344	24	5	55.1	27	6	61.3
s349	24	5	56.4	28	6	56.9
s382	28	4	51.1	29	4	50.9
s386	38	3	43.5	39	6	55.3
s400	28	4	50.6	30	4	56.6
s420	33	5	57.1	46	6	69.5
s444	32	6	64.3	36	6	64.6
s510	55	4	53.7	64	7	78.8
s526	50	4	50.2	46	4	57.2
s820	88	3	45.8	90	4	58.5
s832	92	3	44.9	90	5	61.3
s838	76	8	82.9	98	10	100.7
ratio	0.938	0.816	0.844	1	1	1

표1 실험 결과

5. 참조 문헌

[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] R J Francis, J. Rose, and Z Vranesic "Chortle-crf Fast Technology Mapping for Lookup Table-Based FPGAs," *Proc 28th ACM/IEEE Design Automation Conference*, pp 227-233, 1991

[3] D Hill, B Britton, B Oswald, N-S Woo, S Singh, T Poon, B Krambeck, "A new Architecture for High Performance FPGAs," *Int Workshop on FPGA*, Sep 1992

[4] E Kang, R. Lin and E Shragowitz, "Fuzzy Logic Approach to VLSI Placement," *IEEE Transactions on VLSI Systems*, vol 2, pp 489-501, Dec 1994

[5] J-Y Lee and E Shragowitz, "Performance Driven Technology Mapper for FPGAs with Complex Logic Block Structures," *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS)*, vol 2, pp 1219-1222, Seattle, WA, April 1995

[6] J-Y Lee and E Shragowitz, "Technology Mappings for FPGAs with Complex Block Architectures by Fuzzy Logic Technique," *Proceedings of the Asia South-Pasific Design Automation Conference (ASPDAC)*, pp. 295-300, Makuhari, Japan, August 1995

[7] R Lin and E Shragowitz, "Fuzzy logic approach to placement problem," *Proc of the ACM/IEEE 29th Design Automation Conference*, pp 153-158, 1992

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

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

[10] 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 1995

[11] E Shragowitz, J-Y Lee and E. Kang, "Application of Fuzzy Logic to Computer-Aided Design of Electronic Systems" (Invited Paper) *Proceedings SPIE's 1996 International Symposium on Aerospace Defence Sensing and Control*, Orlando, FL, April, 1996.

[12] N-S Woo, "A1OM: Technology Mapping of Sequential Circuits for Lookup Table-Based FPGAs" Technical Report, AT&T Bell Laboratories, November 1991