Journal of The Institute of Electronics and Information Engineers Vol.52, NO.9, September 2015 http://dx.doi.org/10.5573/ieie.2015.52.9.028
ISSN 2287-5026(Print) / ISSN 2288-159X(Online)

논문 2015-52-9-5

클럭 게이팅 구동신호 기반 상위수준 전력모델의 전력 상태 수 감소

(Reduction of the Number of Power States for High-level Power Models based on Clock Gating Enable Signals)

최 호 석*, 이 준 환**

(Hosuk Choi and Joonhwan Yi©)

요 약

본 논문은 클럭 게이팅 구동신호를 이용한 전력 모델링 방법에서 회로에서 나타나지 않는 잉여 전력 상태를 확인함으로써 전력 상태 수를 줄이는 방법을 제안한다. 회로에 나타나지 않는 전력 상태를 확인하기 위해 함수적 종속성과 구조적 종속성을 확인한다. 본 논문에서는 2개의 클럭 게이팅 구동신호 간에 나타나는 함수적 종속성 중 동치 관계, 역관계, 포함 관계만을 다룬다. 구조적 종속성은 클럭 게이팅 셀의 위치적 특성에 의한 종속성을 의미한다. 두 종속성으로 발견한 관계를 이용해 전력 상태의 수를 줄였으며, 감소 후 남은 전력 상태수를 세기위해 이진결정다이어그램을 사용하였다. 함수적 종속성과 구조적 종속성을 이용해 전력 상태 수를 알고리즘 적용 전 대비 평균 59%까지 감소시켰다.

Abstract

In this paper, we propose to identify redundant power states of high-level power model based on clock gating enable signals(CGENs) using dependencies of Boolean functions and structural dependencies of clock gating cells. Three functional dependencies between two CGENs, namely equvalence, inversion, and inclusion, are used. Functions of CGENs in a circuit are represented by binary decision diagrams (BDDs) and the functional relations are used to reduce the number of power states. The structural dependency appears when a clock gating cell drives another clock gating cells in a circuit. Automatic dependency checking algorithm has been proposed. The experimental results show the average number of power state is reduced by 59%.

Keywords : Boolean dependancy, high-level power analysis, binary decision diagram, clock gating, levelization

* 학생회원, ** 평생회원, 광운대학교 컴퓨터공학과
(Dept. of Computer Engineering, Kwangwoon University)
© Corresponding Author(E-mail: joonhwan.yi@kw.ac.kr)
※ 이 논문은 2011년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (No. NRF-2011-0025385)
※ 이 논문은 2012년도 광운대학교 교내학술연구비 지원에 의해 연구되었음
※ 이 논문은 2012년도 지식경제부 글로벌전문기술개발사업으로 지원된 연구결과입니다. [10047664, 저전력 설계를 위한 고속 (게이트수준 대비 300배) 정확한 (게이트수준 대비 80%) 전력모델 자동생성 소프트웨어 개발]
Received ; February 2, 2015 Revised ; July 2, 2015
Accepted ; September 7, 2015

Ⅰ. 서 론

기술이 발전함에 따라 칩의 복잡도와 클럭의 주파수가 증가하고 있다. 이러한 증가로 칩에서 소모하는 전력량이 증가하고 있으며, 전력 소모량은 칩을 설계함에 있어 주요 제약사항 중 하나로 고려되고 있다. 따라서 칩을 제작하기 전에 칩이 소모할 전력 값을 분석하는 것은 필수적이다.

전력 분석 기술은 전력을 분석하는 시간과 전력 값에 대한 정확도가 중요하다[1]. 그림 1은 각 설계 수준에서 측정되는 전력 값에 대한 정확도와 전력 분석 속도를 나타내고 있다. 게이트 수준이하의 하위 수준에서 전력

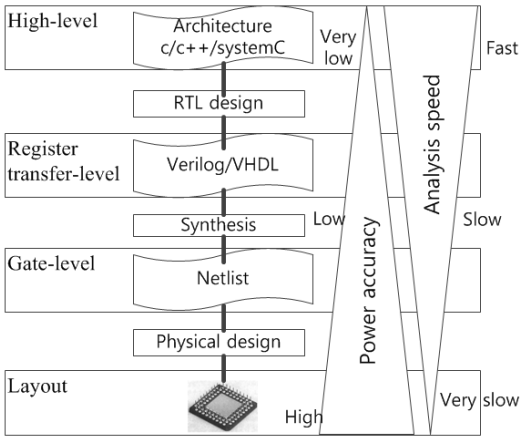

그림 1. 전력 분석의 계층별 특징
Fig. 1. Hierarchical features in power analysis.

을 분석할 경우 정확도는 높지만, 전력을 분석하는데 많은 시간을 필요로 한다. 따라서 하위 수준의 전력 분석은 전체 회로가 아닌 일부 회로의 전력을 분석한다. 전체 회로의 전력 분석은 회로의 일부 회로에서 분석된 전력 값들을 합산하는 방법을 취한다. 하지만 칩 전체가 한 순간에 동작하지는 않아 정확한 전력 분석이 되지 않는다.

상위 수준에서의 전력 분석 방법은 하위 수준에서 측정된 전력 값을 이용한다. 이 방법은 크게 하향식 접근법과 상향식 접근법으로 나뉜다[2~3]. 하향식 접근법은 RTL(register transfer level) 이하의 코드 없이 디자인 명세(design specification) 또는 상위 수준 언어를 이용하여 전력을 분석함으로 속도가 빠르다는 장점을 갖는다. 하지만 전력 값을 계산함에 있어 단위 기능 블록에 포함되지 않는 디지털 회로의 전력 소모량과 클럭 게이팅과 같이 하위 수준 저 전력 설계기술로 변경되는 전력 소비량 등이 고려되지 않아 정확도가 크게 떨어진다는 단점이 있다.

상향식 접근법은 각 전력 상태에 RTL 이하의 코드로부터 측정된 평균 전력 값을 할당해주는 전력 모델링을 수행하여 소비 전력을 계산하는 방법이다. 많은 상향식 접근법은 전력 소모량이 크게 다른 회로의 상태인 전력 상태를 정의한다. 하위 수준에서 측정된 평균 전력량을 정의된 각 전력 상태에 할당하는 전력 모델링을 수행함으로써 전력을 계산한다. 본 방법은 하위 수준 전력 분석 방법에 비해 빠른 전력 분석이 가능하며, 하향식 접근법에 비하여 정확도가 높다. 하지만 상향식 접근법은 전력 모델링을 하는 방법에 따라 정확도가 달라지기 때문에 정확도를 높일 수 있는 전력 모델링이 필요하다.

기존 상향식 접근법들 중에는 임의의 소프트웨어를 구동할 때 수행되는 명령어(instruction)에 따라 전력 값을 할당하는 전력 모델을 기반으로 한 전력 분석 방법[1, 4], 칩에 존재하는 블록별로 다른 전력 모델링을 적용하여 전력을 분석하는 방법[5]등이 존재한다. 이러한 기존의 전력 분석 방법들[1, 4~5]은 전력 상태 수는 적지만 회로에 대한 내부지식을 이용하여 수동으로 전력 모델을 생성한다. 따라서 전력모델의 품질을 일관성 있게 유지하기 힘든 문제가 있다.

두 가지 전력모델 자동생성기법[3, 6~7]이 현재까지 제안되었다. [6]에서는 SystemC 회로의 입출력신호, 내부변수, 회로의 상태 및 이들의 시퀀스로 전력 상태를 정의한다. 앞서 언급한 변수들의 수 n에 따라 총 2^{2n}개의 시퀀스가 전력 상태로 정의되어, 그 수가 비현실적으로 커지는 문제가 있다. 따라서 n을 줄이기 위해 전체변수의 부분집합을 수동으로 선택한 후, 일부 전력 상태를 제거했을 때 정확도의 변화를 반복적으로 분석하여 전력 상태수를 줄이는 통계적인 방법을 제안한다. 클럭 게이팅 구동 신호 기반 상위수준 전력 모델[3,7]은 클럭 게이팅 구동신호들의 값의 조합으로 전력 상태를 정의한다. 따라서 칩에 대한 내부 지식 없이 전력 모델링이 가능하며, 높은 정확도를 갖는다[7]. 하지만, 클럭 게이팅 구동신호의 수 n에 따라 전력 상태 수가 2^n개로 크게 증가하는 문제가 있다. 이 기술의 전력 상태 수를 줄이는 기술은 현재까지 제안된 바가 없다.

본 논문에서는 클럭 게이팅 구동신호들의 조합들로 생성될 수 있는 필요 전력 상태들을 클럭 게이팅 구동신호들 간의 기능적, 구조적 종속성을 판단하여 전력 상태의 수를 감소시키는 방법을 제안한다. 이는 클럭 게이팅 구동신호들의 부울 함수와 이들의 연결 관계를 이용하는 것으로 자동화가 가능하다.

본 논문은 다음과 같이 구성된다. Ⅱ장에서는 논문에서 소개하고자 하는 기술을 설명하는데 필요한 배경지식인 클럭 게이팅 기술에 대하여 서술한다. Ⅲ장에서는 클럭 게이팅 구동신호들 간에 함수적, 공간적 종속성을 확인하여 전력 상태 수를 줄이는 방법과 이진결정다이어그램을 이용하여 줄어든 전력 상태 수를 세는 방법에 대해서 서술한다. Ⅳ장에서는 실험 결과를 서술하였으며, 마지막 Ⅴ장에서 결론으로 마무리한다.

Ⅱ. 클럭 게이팅

클럭 게이팅[8]은 입력 값이 변하지 않는 레지스터에 클럭을 차단하여 동적 전력 소비량을 줄이는 기술이다. 클럭 게이팅 셀은 그림 2와 같이 클럭 게이팅 구동신호 en_i(clock gating enable signal; CGEN)가 1인 경우 회로의 클럭을 clk로 입력 받아 출력 gclk로 통과시킨다. en_i가 0인 경우 회로의 클럭을 차단하여 gclk를 0으로 출력한다. gclk와 연결된 네트는 레지스터로 입력되어, 레지스터가 구동하지 않을 때 클럭을 차단함으로써 동적 전력 소비량을 줄인다. 임의의 회로 C에 n개의 en_i가 있을 때, $en_i(1 \le i \le n)$는 k_i개의 클럭 게이팅 셀 $cg_j(1 \le j \le k_i)$들을 제어한다. 하지만 fan-out limitation으로 인해 k_i는 제한된다.

클럭 게이팅 구동신호들은 회로의 일부 입력들과 플립플롭의 출력들에 따라 정해진다. 클럭 게이팅 구동신호 en_i들에 대한 부울 함수의 부울 변수는 C의 일부 입력들과 플립플롭의 출력들이기 때문이다. 본 논문에서는 임의의 클럭 게이팅 구동신호의 값에 영향을 주는 C의 일부 입력들과 플롭플롭의 출력들, 네트들, 그리고 게이트들의 집합을 클럭 게이팅 입력 로직 콘 $cgilc$ (clock gating input logic cone)라 부른다. 예를 들어 그림 3과 같은 회로를 살펴보자. 회로 내에는 5개의 게이트 $G=\{g_i;1 \le j \le 5\}$, 9개의 네트 $N=\{a,b,clk,n_1,n_2,n_3,n_4,en_1,en_2\}$, 그리고 1개의 플립플롭 f_1이 존재한다. 이 때 en_1의 클럭 게이팅 입력 로직 콘은 $\{en_1,g_2,g_3,n_2,n_3,b\}$가 된다.

그림 2. 클럭 게이팅 셀
Fig. 2. Clock gating cell.

Ⅲ. 본 문

본문에서는 클럭 게이팅 기술이 적용된 임의의 회로 C 에 대하여 함수적 종속성과 구조적 종속성을 고려하여 필요한 전력 상태를 계산하는 방법과 필요 전력 상태의 수를 세는 방법을 소개한다. Ⅲ-1절에서는 함수적 종속성을 확인하기 위해 클럭 게이팅 구동신호들의 부울 함수를 저장하는 방법에 대해 설명한다. Ⅲ-2절 함수적 종속성 부분에서는 각 클럭 게이팅 구동신호의 부울 함수간의 동치 관계, 역 관계, 그리고 포함 관계를 이용해 전력 상태 수를 줄이는 방법을 소개한다. Ⅲ-3절 구조적 종속성에서는 클럭 게이팅 셀의 위치에 따른 구조적 종속성에 대하여 소개한다. Ⅲ-4에서는 이진 결정 다이어그램을 이용하여 위의 두 종속성을 고려했을 때, 클럭 게이팅 구동신호로 정의할 수 있는 필요 전력 상태의 수를 세는 방법에 대하여 소개한다.

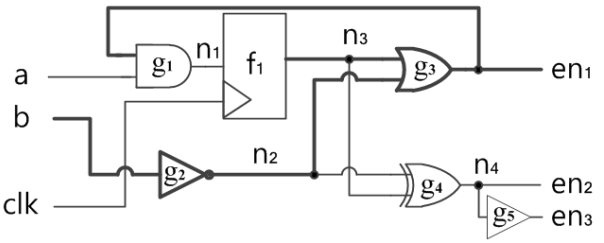

그림 3. 클럭 게이팅 입력 로직 콘
Fig. 3. Clock gating input logic cone.

1. 이진 결정 다이어그램 생성

클럭 게이팅 구동신호들의 함수적 종속성을 확인하기 위해 각 클럭 게이팅 구동신호들에 대한 부울 함수를 추출한다. 이 때 부울 함수간의 함수적 종속성을 확인하기 위해서는 부울 함수의 결과 값을 확인할 수 있어야 한다. 또한 부울 함수는 구동신호의 수가 많아질 경우를 대비하여 적은 메모리를 사용하는 형태로 저장되어야 한다. 따라서 본 논문에서는 위의 설명한 특징을 가진 자료 구조 중 하나인 이진결정다이어그램으로 부울 함수를 저장하였다.

이진결정다이어그램으로 회로에 존재하는 각 클럭 게이팅 구동신호 $en_i(1 \le i \le n)$의 부울 함수를 저장하기 위해서는 레벨화[3, 12]가 필요하다. 레벨화는 각 셀에 레벨 값을 할당해 주는 알고리즘이다. 만약 레벨화를 수행하지 않고 이진결정다이어그램을 생성하면, 필요 없는 연산을 수행함으로써 이진결정다이어그램을 생성하는데 많은 시간을 필요로 하게 된다[3]. 레벨화를 수행한 후 셀에 할당된 레벨의 순서대로 이진결정다이어그램을 생성하면 en_i에 대한 이진결정다이어그램 bdd_i가 생성된다. 생성된 bdd_i를 이용하여 다음 절에서 함

수적 종속성을 판단한다.

2. 함수적 종속성

n개의 클럭 게이팅 구동신호간의 함수적 종속성에 의한 관계를 클럭 게이팅 구동신호를 $m(2 \le m \le n)$개씩 묶어 확인할 수 있다. 예를 들어 m이 2일 때 종속성을 표현한 관계는 표 1과 같다. 2개의 클럭 게이팅 구동신호 값의 조합 {(0,0), (0,1), (1,0), (1,1)}의 수는 2^2개이다. 이 중, 일부 조합이 종속성으로 인해 회로 동작 중 발생할 수 없다면, 이 조합이 들어간 전력 상태는 잉여전력 상태가 된다. 모든 조합이 가능한 관계의 수는 ${}_4C_0$개, 불가능한 조합의 수가 1개일 때 관계의 수는 ${}_4C_1$개, 2개일 때는 ${}_4C_2$개, 3개일 때는 ${}_4C_3$개, 4개일 때는 ${}_4C_4$개로, 표 1과 같이 총 16개의 관계가 나타난다. 따라서 잉여 전력 상태가 $i(0 \le i \le m)$개일 때 나타나는 전력 상태의 수는 식 1과 같다.

$$\sum_{i=0}^{m} {}_{2^m}C_i = 2^{2^m} \tag{1}$$

따라서 총 관계의 수는 ${}_nC_m \times 2^2$으로 m값에 증가에 따라 계산량이 지수적으로 증가한다. 하지만 m이 작을 때 나타나는 관계들로 m이 클 때의 관계들 중 50%이상이 확인된다. 예를 들어 m이 2일 때 나타나는 관계들로 m이 3일 때 나타나는 관계들의 64.84%에 해당하는 관계가 확인된다[3]. 본 논문에서는 계산량을 고려하여 m이 2인 경우의 관계만을 확인한다.

두 개의 클럭 게이팅 구동신호 en_i와 en_j의 부울 함수를 나타내는 이진결정다이어그램 bdd_i와 bdd_j가 존재할 때 각 관계는 다음과 같이 정의한다.

정의 1. 동치 관계 : 두 신호 en_i와 en_j의 부울 함수가 en_i=en_j일 때, en_i와 en_j는 서로 동치 관계이다.

이 때 식2를 만족하면 en_i와 en_j는 서로 동치 관계임을 확인할 수 있다.

$$bdd_i \hat{} bdd_j = 0 \tag{2}$$

정의 2. 역 관계 : 두 신호 en_i와 en_j의 부울 함수가 en_i= $!en_j$일 때, en_i와 en_j는 서로 역 관계이다.

이 때 식 3을 만족하면 en_i와 en_j는 서로 역 관계임을 확인할 수 있다.

$$bdd_i \hat{} bdd_j = 1 \tag{3}$$

정의 3. 포함 관계 : 두 신호 en_i와 en_j의 부울 함수가 $en_i \mid en_j = 1$일 때, en_i와 en_j는 서로 포함 관계이며, 이는 $en_i \rightarrow en_j$와 같이 표기한다.

마지막으로 식 4는 $en_i \rightarrow en_j$일 때 만족되는 식이다.

$$bdd_i \& bdd_j = bdd_j \tag{4}$$

표 2는 $en_1 = en_2$, $en_3 = !en_4$, 그리고 $en_3 \rightarrow en_1$인 경

표 1. 2개의 클럭 게이팅 구동신호로 나타나는 모든 관계
Table 1. All relation between two CGENs.

$\{en_i, en_j\}$ Relation	{0,0}	{0,1}	{1,0}	{1,1}
R_2-1	O	O	O	O
R_2-2	X	O	O	O
R_2-3	O	X	O	O
R_2-4	O	O	X	O
R_2-5	O	O	O	X
R_2-6	X	X	O	O
R_2-7	X	O	X	O
R_2-8	X	O	O	X
R_2-9	O	X	X	O
R_2-10	O	X	O	X
R_2-11	O	O	X	X
R_2-12	X	X	X	O
R_2-13	X	X	O	X
R_2-14	X	O	X	X
R_2-15	O	X	X	X
R_2-16	X	X	X	X

표 2. 각 경우에 대한 전력 상태의 존재 여부
Table 2. Irredundant power states presence in each cases. case 1: $en_1 = en_2$; case 2: $en_3 = !en_4$; case 3: $en_3 \rightarrow en_1$

Power state	en_1	en_2	en_3	en_4	Irredundant power state		
					case 1	case 2	case 3
ps_1	0	0	0	0	O	X	O
ps_2	0	0	0	1	O	O	O
ps_3	0	0	1	0	O	O	X
ps_4	0	0	1	1	O	X	X
ps_5	0	1	0	0	X	X	O
ps_6	0	1	0	1	X	O	O
ps_7	0	1	1	0	X	O	X
ps_8	0	1	1	1	X	X	X
ps_9	1	0	0	0	X	X	O
ps_{10}	1	0	0	1	X	O	O
ps_{11}	1	0	1	0	X	O	O
ps_{12}	1	0	1	1	X	X	O
ps_{13}	1	1	0	0	O	X	O
ps_{14}	1	1	0	1	O	O	O
ps_{15}	1	1	1	0	O	O	O
ps_{16}	1	1	1	1	O	X	O

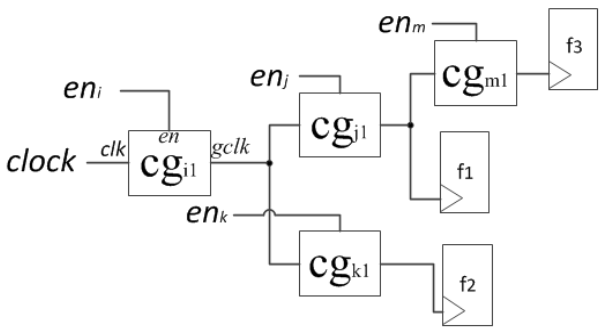

그림 4. 클럭 게이팅 셀의 종류를 나타낸 회로
Fig. 4. A circuit expressed kinds of CGENs.

우 각 경우에 대하여 필요 전력 상태가 나오는 경우는 'O', 그렇지 않은 경우는 'X'로 표기한 표이다. case 1의 경우 총 16가지 전력 상태 중 식 2를 만족하는 전력 상태의 수는 8개이다. case 2의 경우 식 3을 만족하는 전력 상태의 수는 8개이며, case 3의 경우 식 4를 만족하는 전력 상태의 수는 12개이다. 위의 3가지 경우를 동시에 만족하는 전력 상태는 ps_2, ps_{14}, 그리고 ps_{15}로 총 3개임을 알 수 있다.

3. 구조적 종속성

다음으로 확인할 부분은 구조적 종속성이다. 구조적 종속성은 클럭 게이팅 셀의 위치적 특성으로 나타나는 종속성이다. 본 종속성의 설명을 위해 세 종류의 클럭 게이팅 셀인 leaf cell과 stem cell, 그리고 branch cell에 대하여 설명한다.

Leaf cell을 클럭 게이팅 셀 cg_{ij}중 레지스터만을 구동시키는 셀을 말한다. 그림 4에는 4개의 클럭 게이팅 셀 cg_{i1}, cg_{j1}, cg_{k1}, cg_{m1}과 3개의 플립플롭 f_1, f_2, f_3가 존재한다. 이 때 cg_{k1}은 f_2만을 구동시키므로 leaf cell이라 할 수 있으며, cg_{m1}도 f_3만을 구동시키므로 leaf cell이라 할 수 있다. Stem cell은 임의의 클럭 게이팅 셀 cg_{ij}가 다른 클럭 게이팅 셀들만을 구동시키는 셀을 말한다. 예를 들어 그림 4에서 플립플롭은 구동시키지 않으며, cg_{i1}은 cg_{j1}과 cg_{k1}만을 구동시키는 것을 확인할 수 있다. 따라서 cg_{i1}은 steam cell이라 할 수 있다. 마지막으로 임의의 클럭 게이팅 셀 cg_{ij}가 다른 클럭 게이팅 셀들과 레지스터들을 동시에 구동시킬 때 cg_{ij}를 branch cell이라 한다.

구조적 종속성은 위의 세 종류의 클럭 게이팅 셀 중 steam cell과 branch cell에서 발생한다. 만약 stem cell 인 cg_{ij}의 클럭 게이팅 구동신호 en_i가 0인 경우를 가정하자. cg_{ij}에서 출력되는 클럭을 입력으로 받는 다른 클럭 게이팅 셀들의 클럭 게이팅 구동신호가 1이 되어도 구동되지 않는다. 따라서 cg_{ij}에서 출력되는 클럭을 입력으로 받는 다른 클럭 게이팅 셀들의 출력은 0이 된다. 예를 들어 그림 4에서 en_i가 0인 경우 en_j와 en_k는 1이 되어도 cg_{j1}과 cg_{k1}의 출력은 0이다. 따라서 en_i가 0이고 en_j또는 en_k가 1인 전력 상태는 잉여 전력 상태가 된다.

본 논문에서는 함수적 종속성과 구조적 종속성을 고려하여 전력 상태 수를 줄였다. 하지만 클럭 게이팅 구동신호가 n개일 때, 전력 상태 수가 2^n개로, 값에 따라 전력 상태수가 지수적으로 증가한다. 따라서 전력 상태를 세는 것도 하나의 문제였다. 이러한 문제는 다음 절에서 다루도록 하겠다.

4. 필요 전력 상태 수 카운트

전력 상태의 수를 감소시키기 위해 함수적 종속성과 구조적 종속성을 고려하였을 때, 발생할 수 있는 전력 상태 수를 확인할 수 있어야 성능확인이 가능하다. 따라서 함수적 종속성과 구조적 종속성을 고려했을 때, 전력 상태 수를 계산하는 방법을 다음과 같이 구현하였다.

먼저 함수적 종속성과 구조적 종속성에서 나타나는 클럭 게이팅 구동신호들 간의 관계를 추출한다. 추출된 정보는 각 관계 별로 한 쌍의 클럭 게이팅 구동신호의 인덱스 값을 가지고 있다. 이러한 정보를 이용하여 각 관계를 만족할 때, 1이 되는 부울 함수를 생성한다. 이러한 부울 함수를 만들어 주는 이유는 각 관계에 대한 필요 전력 상태의 수는 부울 함수를 1로 만들어 주는 입력 조합의 개수와 같기 때문이다. 각 관계를 만족할 때 결과가 1이 되는 부울 함수는 다음과 같다.

두 클럭 게이팅 구동신호 en_i와 en_j가 동치 관계를 가질 때, 식 5의 B_{eq}는 항상 1이 된다.

$$B_{eq} = !(en_i \hat{} en_j) \tag{5}$$

다음으로 en_i와 en_j가 역 관계일 때, 항상 1이 되는 부울 함수 B_{inv}는 식 6과 같다.

$$B_{inv} = en_i \hat{} en_j \tag{6}$$

마지막으로 en_i와 en_j가 포함 관계 또는 구조적 종속

성을 가질 때, 항상 1이 되는 부울 함수 $B_{include}$는 식 7과 같다.

$$B_{include} = en_i \mid !en_j \tag{7}$$

예를 들어 표 2의 3가지 경우를 확인해 보자. Case 1은 $en_1 = en_2$인 경우이다. 이 때 en_1과 en_2에 대하여 식 5를 만족하는 클럭 게이팅 구동신호의 조합 개수는 8개이다. Case 2는 $en_3 = !en_4$인 경우이다. 이 때 en_3와 en_4에 대하여 식 6을 만족하는 클럭 게이팅 구동신호의 조합 개수는 8개이다. 마지막으로 case 3은 $en_3 \rightarrow en_1$인 경우이다. 이 때 en_1과 en_3에 대하여 식 7을 만족하는 클럭 게이팅 구동신호의 조합 개수는 12개이다.

이 때 동치 관계의 경우 en_i와 en_j로 식 5를 만족시키는 부울 함수를 생성하지 않고, 이진 결정 다이어그램을 생성 시 en_j를 삭제한다. en_j를 삭제하는 이유는 만약 여러 클럭 게이팅 구동신호의 부울 함수가 동일한 경우에도 필요 전력 상태 수는 2개이기 때문이다. 예를 들어 en_1=en_2=en_3라고 가정하자. 이 때 필요 전력 상태는 3개의 클럭 게이팅 구동신호가 모두 0일 때와 1일 때 두 가지이다. 이를 이진 결정다이어그램으로 생성 시 하나의 클럭 게이팅 구동신호를 제외한 나머지 구동신호는 삭제해도 필요 전력 상태의 수는 동일하며, 이진 결정다이어그램의 크기는 작아진다. 따라서 동치 관계의 경우 하나의 클럭 게이팅 구동신호만으로 이진 결정 다이어그램을 생성한다.

각 관계로 생성된 부울 함수를 모두 and연산하여 부울 함수를 생성한다. And연산으로 생성된 부울 함수의 결과가 1인 입력의 조합의 개수는 필요 전력 상태의 개수와 같기 때문이다. 마지막으로 이를 이진 결정 다이어그램으로 생성한다. 이 때 sifting 알고리즘[13]을 적용하여 이진 결정 다이어그램의 크기를 줄였다. 마지막으로 생성된 이진 결정 다이어그램의 터미널 노드가 1인 경로의 개수를 확인한다[3]. 터미널 노드가 1인 경로의 개수는 필요 전력 상태의 개수와 같기 때문이다.

Ⅳ. 실험 결과

본 논문에서는 opencores.org에 있는 3개의 IP인 FPU[16], UART[17], 그리고 Nova[18]로 실험을 진행하였다. 표 3는 각 IP의 정보를 보여준다. FPU(floating point unit)는 64bit 소수에 대하여 사칙연산을 수행시켜주는 연산기이다. UART(universal asynshronous receiver transmitter)는 범용 비동기화 송수신기이다. Nova는 모바일 어플리케이션을 타겟으로 한 H.264/AVC baseline decoder이다. 추가적으로 많은 클럭 게이팅 구동 신호가 존재할 때에 대한 실험을 위해 Nova에 Design Complier[14]를 이용하여, 하나의 클럭 게이팅 셀이 구동할 수 있는 최소의 플립플롭의 수(minimum bandwidth)에 따라 클럭 게이팅 셀을 삽입하였다. 삽입된 클럭 게이팅 셀의 minimum bandwidth가 8인 회로는 Nova8이며, 16인 회로는 Nova16, 32인 회로는 Nova32이다. NovaN는 Design complier로 클럭 게이팅 셀을 넣지 않고, Nova에 있는 클럭 게이팅 셀만으로 실험을 진행하였다.

종속성 확인 및 전력 상태 수를 세기위해 이진결정다이어그램이 필요하다. 본 논문에서는 이진결정다이어그램 설계를 위해 Colorado university decision diagram (CUDD) package[15]를 사용하였다. 각 IP정보를 입력받아 각 클럭 게이팅 구동신호에 대하여 이진결정다이어그램을 생성하였다.

각 IP의 함수적 종속성과 구조적 종속성을 확인하였다. 각 IP별 발견된 관계의 수는 표 4와 같다. 또한 함수적 종속성과 구조적 종속성으로 판별하여 클럭 게이팅 구동신호로 생성될 수 있는 전력 상태 수는 표 5와 같다. FPU의 경우 3쌍의 동치 관계와 2쌍의 포함 관계

표 3. 실험 IP들의 정보
Table 3. Information about each IPs.

IP	Area (gate count)	The number of CGEN
FPU	60K	14
UART	35K	16
NovaN	304K	37
Nova32	262K	796
Nova16	251K	854
Nova8	258K	1,209

표 4. IP들에 존재하는 클럭 게이팅 구동신호 간 관계수
Table 4. The number of functional & structural dependancy in each IPs.

IP	The number of relations			
	Equivalence	Inverse	Inclusion	Structural
FPU	3	0	2	0
UART	0	0	8	0
NovaN	0	0	4	0
Nova32	120	2	14,030	551
Nova16	122	2	15,347	590
Nova8	124	2	22,075	867

표 5. 클럭 게이팅 구동신호로 생성될 수 있는 전력 상태 수
Table 5. The number of irredundant power states according to dependencies on CGENs.

IP	The number of CGEN	The number of power states	The number of irredundant power state (the number of power states / the number of irredundant power states %)	
			Functional dependencies	Functional & structural dependencies
FPU	14	16,384	2,304 (14%)	2,304 (14%)
UART	16	65,536	51,712 (79%)	51,712 (79%)
NovaN	37	1.37×10^{12}	40,802,189,312 (30%)	40,802,189,312 (30%)
Nova32	796	4.16×10^{24}	$8.47 \times 10^{22}(2.03 \times 10^{-17}\%)$	$5.64 \times 10^{21}(1.35 \times 10^{-19}\%)$
Nova16	854	1.20×10^{25}	$8.50 \times 10^{22}(7.08 \times 10^{-18}\%)$	$4.16 \times 10^{22}(3.46 \times 10^{-20}\%)$
Nova8	1,209	8.81×10^{36}	$4.33 \times 10^{33}(4.91 \times 10^{-20}\%)$	$6.59 \times 10^{33}(7.48 \times 10^{-21}\%)$

가 발견되었다. 발견된 관계를 적용함으로써 총 16,384개의 전력 상태 중 14%에 해당하는 2,304개가 필요 전력 상태로 발견 되었다. UART의 경우 8쌍의 포함 관계가 발견되었다. 발견된 관계를 적용함으로써 65,536개의 전력 상태 중 79%에 해당하는 51,712개만이 필요 전력 상태로 발견 되었다. NovaN의 경우 총 4쌍의 포함 관계가 발견되었다. 이를 바탕으로 1.37×10^{12}개의 전력 상태 중 30%인 40,802,189,312개만이 필요 전력 상태로 발견되었다. Nova32에서는 120쌍의 동치 관계, 14,030개의 포함 관계, 2쌍의 역 관계, 그리고 551쌍의 구조적 종속성이 발견되었다. 총 4.16×10^{24}개의 전력 상태 중 함수적 종속성만을 판단하였을 때 2.03×10^{-17}%에 해당하는 8.47×10^{22}개가 필요 전력 상태로 발견되었다. 이에 구조적 종속성을 추가적으로 판단했을 때, 1.35×10^{-19}%에 해당하는 5.64×10^{21}개의 필요 전력 상태가 확인되었다. Nova16에서는 122쌍의 동치 관계, 15,347개의 포함 관계, 2쌍의 역 관계, 그리고 590쌍의 구조적 종속성이 확인되었다. 총 1.20×10^{25}개의 전력 상태 중 함수적 종속성만을 판단하였을 때는 7.08×10^{-18}%에 해당하는 8.50×10^{22}개의 필요 전력 상태가 발견되었으며, 구조적 종속성까지 판단하였을 때 3.46×10^{-20}%에 해당하는 4.16×10^{22}개의 필요 전력 상태가 발견되었다. 마지막으로 nova8에서는 124쌍의 동치 관계, 22,075개의 포함 관계, 2쌍의 역 관계, 그리고 867쌍의 구조적 종속성이 확인되었다. 총 8.81×10^{36}개의 전력 상태 중 함수적 종속성만을 판단하였을 때 4.91×10^{-20}%에 해당하는 4.33×10^{33}개의 필요 전력 상태가 발견 되었으며, 구조적 종속성까지 판단하였을 때는 7.48×10^{-21}%에 해당하는 6.59×10^{33}개의 필요 전력 상태가 발견되었다. 실험결과 총 4개의 IP에 대하여 함수적 종속성과 구조적 종속성을 이용해 전력 상태 수를 최대 7.48×10^{-21}%까지 감소시켰다.

V. 결 론

본 논문에서는 클럭 게이팅 구동신호간의 함수적 종속성과 구조적 종속성을 확인함으로써, 전력 상태 수를 줄이는 방법을 제안했다. 함수적 종속성을 확인하기 위하여 클럭 게이팅 구동신호의 부울 함수를 이진결정다이어그램으로 저장하였다. 저장된 이진 결정다이어그램으로 함수적 종속성 관계인 동치 관계와 역 관계, 그리고 포함 관계를 각각 분석하였다. 구조적 종속성의 경우 stem cell과 branch cell에서 나타나는 셀 간의 종속성을 이용하여 분석하였다. 분석된 결과를 이진결정다이어그램을 이용하여 전력 상태의 수를 확인하였다.

함수적 종속성과 구조적 종속성을 모두 확인한 결과 opencores.org에 있는 3개의 IP에서 최소 21%에서 최대 86%까지 전력 상태 수를 감소시켰다. 추가적으로 NovaN에 클럭 게이팅 구동 신호의 수를 증가시킨 경우에는 99.9%이상의 전력 상태 수를 감소시켰지만, 감소 후에도 많은 양의 전력 상태가 존재하였다.

본 방법은 클럭 게이팅 구동 신호 간 종속성이 발견되지 않으면 전력 상태 수를 감소시키지 못한다. 하지만 하나의 종속성만 발견이 되어도 최대 50%의 전력 상태 수를 감소시킨다. 따라서 본 방법은 전력 상태 수가 많고, 클럭 게이팅 구동 신호 간 종속성이 많이 발견될수록 많은 전력 상태 수를 감소시킬 수 있다.

REFERENCES

[1] C.M. Lee, C.K. Chen, and R.S. Tsay. “A Basic-block Power Annotation Approach for

Fast and Accurate Embedded Software Power Estimation," the International Conference on VLSI, pp. 118~123, Oct. 2013.
[2] G. Vijin, Oklobdzija, "The Computer Engineering Handbook," CRC Press,Dec. 2001
[3] H. Choi, "Reduction of the Number of Power States for High-level Power Models based on Clock Gating Signals," Kwangwoon university, Feb. 2015.
[4] N.F. Ghohroud. Z. Navabi, "Back-annoation of Gate-level Power Properties into System Level Descriptions" ICCC New Circuits and Systems Conference 12th, pp.237-240, Jun. 2014.
[5] I. Lee, H. Kim, S. Yoo, E. Y. Chung, K. M. Choi, J. K. Kong and S. K.. Eo. "Powervip: Soc power estimation framework at transaction level" In Proc. of South Pacific Design Automation Conference, pp.551 - 558, 2006.
[6] N. Bansal, K. Lahiri, A. Raghunathan, "Automatic Power Modeling of Infratructure IP for SystemonChip Power Analysis," 20th International Conference on VLSI Design, pp.513-520, 2007.
[7] J. Kim, J. Yi, "Case study on the High-Level Power Modeling Based on Clock Gating," 2013 IEIE summer Conference, pp. 1945~1948, Vol. 2013, No. 7, Jul. 2013.
[8] S. Wimer, I. Koren, I. Cederbaum "Design Flow for Flip-Flop Grouping in Data Driven Clock Gating" IEEE Transactions on VLSI Systems, Vol. 22, pp.771~778, May. 2013.
[9] R. Fraer, G. Kamhi, and M. K. Mhameed. "A new paradigm for synthesis and propagation of clock gating conditions" In Proc. of Design Automation Conference, pages 658~663, Jun. 2008.
[10] S.B. Aker,. "Binary decision diagrams," IEEE Transactions on Computers, Vol. C-27, no.6, pp. 509~516, Jun. 1978.
[11] P.K. Sharma, N.K. Singh, "Improved BDD Compression by Combination of Variable Ordering Techniques" the International Conference on Communications and Signal Processing, pp.3-5, Apr. 2014.
[12] E. Albert, E. Ruehli, A.L. Sangiovanni-vincentelli, G. Rabbat, "Time Analysis of Large-Scale Circuits Containing One-Way Macromodels," IEEE Transactions on Circuits and Systems, Vol. 29, pp. 185-190, Mar. 1982.
[13] M.T. Kuo, T. Wang, "BDD-based Logic Partitioning for Sequential Circuits" Design Automation Conference, pp.607-612, Jan. 1997.
[14] Design Complier (http://www.synopsys.com)
[15] Colorado university decision diagram (CUDD) package (http://www.cs.uleth.ca/~rice/cudd.html)
[16] Floating Point Unit (opencore.ore/project,fpu)
[17] UART to Bus (opencore.ore/project,uart)
[18] H.264/AVC BaselineDecoder (opencore.ore/project,nova)

저 자 소 개

최 호 석(학생회원)
2013년 광운대학교 컴퓨터공학과 학사 졸업.
2015년 광운대학교 컴퓨터공학과 석사 졸업.

<주관심분야 : 디지털회로 설계, 설계 자동화>

이 준 환(평생회원)
1991년 연세대학교 전자공학과 학사 졸업.
1998년 University of Michigan, Ann Arbor, 전기컴퓨터공학 석사
2002년 University of Michigan, Ann Arbor, 전기컴퓨터공학 박사

<주관심분야 : 저전력 SoC 설계, 설계자동화>