論文2000-37SD-11-9

부분 스캔을 고려한 최적화된 상태할당 기술 개발
(Development of Optimized State Assignment Technique for Partial Scan Designs)

曺 尙 煜*, 梁 世 陽**, 朴 成 柱*
(Sangwook Cho, Saeyang Yang, and Sungju Park)

요 약

유한상태기의 상태할당은 이로부터 구현되는 순차회로의 속도, 면적, 테스트가능도에 큰 영향을 미친다. 본 논문에서는 상태변수 그룹들 사이에 상호 의존성(dependency)을 최소화하여 스캔선택이 필요한 플립플롭 수를 최소화하기 위한 m-블록 분할을 이용한 새로운 상태할당 기술을 소개한다. 제안되는 방법을 통하여 우선 상태할당을 수행하고 논리 합성을 거친 후에 부분 스캔 설계가 이루어진다. 벤치마크 회로에 대한 실험 결과 면적과 속도 면에서 최적을 유지한 채로 테스트가능도가 현저히 개선되었음을 확인하였다.

Abstract

The state assignment for a finite state machine greatly affects the delay, area, and testabilities of the sequential circuits. In order to minimize the dependencies among groups of state variables, therefore possibly to reduce the length and number of feedback cycles, a new state assignment technique based on m-block partition is introduced in this paper. After the completion of proposed state assignment and logic synthesis, partial scan design is performed to choose minimal number of scan flip-flops. Experiment shows drastic improvement in testabilities while preserving low area and delay overhead.

I. 서 론

칩의 집적도의 증가에 따라 설계점검 및 칩제작 후의 기능점검 등은 더욱 더 어려운 문제로 부각되고 있다. 이러한 테스트 문제를 해결하기 위하여 체계적인 테스트설계 기술이 널리 개발되고 있다. 주문형반도체를 테스트하는 방법으로서 많이 사용되는 스캔 방식은 회로내의 플립플롭이나 래치를 제어, 관찰하는 것을용이하도록 회로 구조를 설계함으로서 테스트를 효과적으로 수행할 수 있게 한다[1]. 그러나 회로내의 모든 메모리 소자들을 스캔 가능하게 하는 완전스캔(full scan) 방식은 이를 채용한 회로가 스캔 방식을 채용하지 않은 회로와 비교하여 성능 저하 및 칩 크기 증가와 같은 문제점을 내포하고 있다. 이를 해결하는 효율적인 방법으로서 회로내의 모든 메모리 소자들 대신 이들의 최소 부분만을 스캔 가능하게 하는 부분 스캔(partial scan) 방식이 널리 사용되고 있다[2]. 본 논문에서는 회로의 면적 최소화와 함께 테스트가능도를 최적화 할 수 있는 상태할당 기술을 연구하였다. 상태할당 방법에는 One-hot, Random, Jedi 와같은 방법이 개발되었다[3,4]. One-hot은 n개의 상태들로 구성되어진 유한상태기를 정확하게 n개의 플립플롭을 사용해서 할당하는 기술이다. 이는 상태할당 방법들 중에서 가

* 正會員, 漢陽大學校 電子計算學科
(Dept. of Computer Science & Engineering, Hanyang Univ.)
** 正會員, 釜山大學校 컴퓨터工學科
(Dept. of Computer Engineering, Pusan Univ.)
※ 본 논문은 한국과학재단 특정기초연구(과제번호: 95-0100-21-01-3, 95-0100-21-02-3)과제로부터 지원을 받아서 진행되었습니다.
接受日字:1999年 11月12日, 수정완료일:2000年 9月29日

장 많은 수의 플립플롭을 필요로 하지만 속도를 줄이는 면에 있어서 효율적이다. Random은 「logN」 개의 플립플롭을 사용하여 임의의 이진코드 패턴으로 상태할당을 수행한다. 그리고 Jedi는 역시 「logN」개의 플립플롭을 사용하여 면적을 예측하는 값함수를 이용한 고유의 알고리듬을 이용하여 상태할당을 하며 멀티레벨 구현을 위한 일반적인 symbol encoding에 목표를 두고 있는데, 이용되는 값함수는 심벌의 쌍들간에 무게를 부여하는 휴리스틱 방법을 적용하고 있다. 이는 문자 절약 면에서 효율적이고 궁극적으로 면적을 최소화하는 상태할당 방법이다. 이러한 상태할당 기술은 구현되는 동기 순차회로의 면적에 큰 영향을 미칠 뿐만 아니라[5-9], 회로의 테스트가능도에도 지대한 영향을 미칠 수 있다는 연구가 발표되고 있다[6,10]. [11]에서는 2-블록 분할(2-block partition)을 이용하여 순차 테스트를 용이하게 하면서 면적최소화도 가능하게 하는 상태할당 방법을 제안하였다. 본 논문에서는 기존의 2-블록 분할을 이용한 방법보다, 더욱 일반화되어진 m-블록 분할을 이용한 상태할당을 통하여 테스트가능도 및 면적을 최소화 할 수 있는 기술을 개발하였다.

본 논문은 다음과 같이 구성되어 있다. Ⅱ절에서는 상태할당을 통한 순차회로 구조에서 상태변수들간의 의존성을 비교하고, Ⅲ절에서는 부분 스캔을 고려한 최적의 상태할당 기술을 모델링한다. Ⅳ절에서는 상태변수들의 의존성을 줄이는 방법을 소개하며, Ⅴ절에서는 벤치마크 회로에 대한 실험결과를 보여주고, 마지막 장에서는 결론 및 향후 계획을 기술한다.

Ⅱ. 순차회로에 대한 상태할당 방법

순차회로에 대한 논리 합성에 있어서 주된 관심사 중의 하나는 유한상태기를 이용하여 추출한 상태천이표에서 각 상태들에 이진 코드 값을 할당하는 것이다. 이 경우, 그 상태들과 출력 변수의 기능적인 의존도 뿐 만 아니라 보다 적은 수의 메모리 소자들만으로 회로가 이루어 질 수 있도록 할당을 하여야 한다. 그러나 실제로 상태할당이 상이하게 되어지는 모든 경우의 수는 상태 수와 비교하여 지수 함수적으로 많아지는데, 예를 들어 n개의 상태들로 구성된 유한상태기는 n!개의 다른 상태할당을 가진다. 따라서 그 중에서 가장 최적으로 할당된 상태들을 구하는 것은 NP-complete 문제이다.

PS/ X1X2	NS			
	00	01	11	10
a	e	c	d	e
b	g	a	b	g
c	a	c	h	e
d	c	a	f	g
e	e	c	d	e
f	g	a	b	g
g	a	c	h	e
h	c	a	f	g

(a)

α	β
y1y2y3	y1y2y3
a : 0 0 0	a : 0 0 0
b : 1 0 0	b : 1 1 0
c : 0 1 0	c : 0 1 0
d : 0 1 1	d : 1 0 0
e : 1 1 1	e : 0 0 1
f : 1 0 1	f : 1 1 1
g : 1 1 0	g : 0 1 1
h : 0 0 1	h : 1 0 1

(b)

그림 1. 상태천이표 및 상태할당
Fig. 1. State Transition Table and Assignment.

다음에는 본 논문에서 연구된 부분스캔을 고려한 최적화된 상태할당의 연구 동기를 예를 통하여 설명한다. 우선 그림 1의 (a)와 같은 상태천이표로 표시되는 유한상태기를 임의의 상태할당을 거쳐 구현하는 경우 상태 변수들간의 의존성을 비교해본다. 그림 1의 (a)에 있는 상태천이표로부터 (b)의 상태할당에 의하여 회로를 구현하면 유한상태기내의 3개의 플립플롭 Y1(y1), Y2(y2), Y3(y3) (Y는 다음상태, y는 현재상태) 들간에는 "상호간" 완전한 의존성이 생기게 된다(그림 2). 또한 [12]에서는 스캔-그래프에서 서로 다른 단순 사이클들이 얼마나 존재하는 가를 알 수 있다.

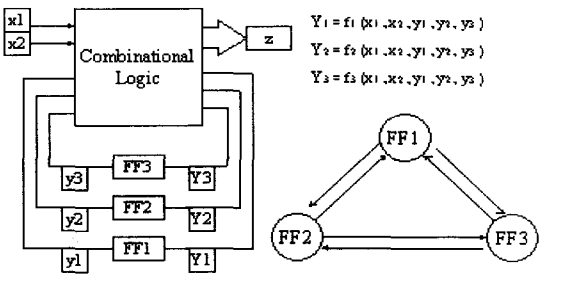


그림 2. 상태할당 α에 의한 회로구조 및 스캔 그래프
Fig. 2. Circuit structure and Scan Graph by State Assignment α.

그림 2에 의해서 구현되는 회로에서 이 스캔-그래프를 구성하면 2개의 단순 사이클이 존재함을 알 수 있다. 이와 같은 경우에는 3개의 플립플롭 가운데 최소한 하나의 플립플롭을 스캔 가능하게 하여야 부분 스캔에 의한 테스트가 이루어 질 수 있다. 반면, 그림 3과 같이 회로가 구현된 경우에는 유한상태기내의 3개의 플립플롭 Y1(y1), Y2(y2), Y3(y3) 들간에는"한

방향으로의" 의존성(dependency)만이 생기게 된다. 이는 부분 스캔의 입장에서는 매우 바람직한 구조이다. 왜냐하면단순 사이클이 존재하지 않음을 알 수 있으며 이는 회로내의 플립플롭을 전혀 스캔 가능하게 하지 않아도 부분 스캔에 의한 테스트가 가능할 수 있기 때문이다.

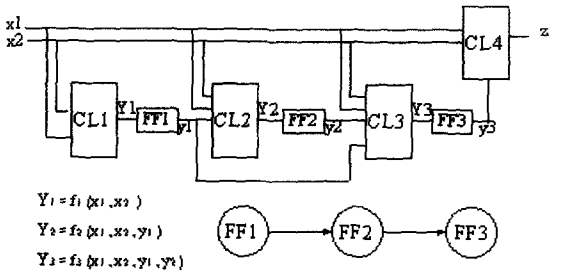


그림 3. 상태할당β에 의한 회로구조 및 스캔 그래프
Fig. 3. Circuit structure and Scan Graph by State Assignment β.

또한, 표 1과 같이 모든 현재상태에서 다음상태로 천이 되는 비트별 변화의 차이에서도 상태할당β가 6 비트 더 적게 나타남을 알 수 있다. 결론적으로 상태할당 β가 상태할당 α보다 회로 면적 증가도 최소화하므로 부분 스캔 테스트에 효율적이라 할 수 있다.

표 1. 모든 State 천이에 대한 비트 변화의 수
Table. 1. The number of bit change for all state transitions.

	a-e	a-c	a-d	·····	h-a	h-f	h-g	합계
상태할당α	3	1	2	·····	1	1	3	52
상태할당β	1	1	1	·····	2	1	2	48

Ⅲ. 부분스캔을 고려한 최적화된 상태할당 기술 소개

먼저, 최적화된 상태할당 기술을 위한 용어를 정의한다

정의 1] S : FSM 상태천이도의 심벌상태
정의 2] 분할(partition) : 교집합이 S인 연결되지 않은 부분집합
정의 3] 블록(block) : 각 부분집합
정의 4] m-블록분할 : 부분집합의 수가 m 인 경우
정의 5] i&s : 상태 s에서 입력 i에 의하여 천이 되는 다음상태
정의 6] 분할 쌍(partition pair) : (p1,p2)는 각각의 입력 i에 대하여 p1의 같은 블록에 상태 s1과 s2가 같이 있으면 i&s1, i&s2가 p2에서 같은 블록에 있는 분할의 순서쌍(ordered pair)
정의 7] 의존성 : 다음상태 변수들의 값은 남아 있는 변수의 값에 의존하는 경우

논문 [13]의 예제를 인용하여 용어를 설명한다. 그림 5 M의 상태천이표로부터 다음과 같은 두 개의 분할 p' = (a, d : b : c ,e : f)와 p" = (a, e : b, d, c, f)를 보자. p'은 4-블록 분할이고 p"는 2-블록 분할이다. 또한 순서쌍 (p',p")은 분할쌍이다. 그러면 분할과 상태할당과의 관계는 어떠한 지를 그림 4의 예제를 통해 알아본다.

PS/	NS				y1y2y3	
X1X2	00	01	11	10		
a	a	c	d	f	a	0 0 0
b	c	b	f	e	b	0 1 1
c	a	b	f	d	c	1 0 1
d	e	f	b	c	d	0 0 1
e	e	d	c	b	e	1 0 0
f	d	f	b	a	f	1 1 1

그림 4. 상태천이표 M 및 상태할당
Fig. 4. State Transition Table M and State Assignment.

상태할당으로부터 각각의 상태변수 하나는 해당 2-블록 분할(y1은 p1 = (a, b, d : c, e, f), y2는 p2 = (a, c, d, e : b, f) 그리고 y3은 p3 = (a, e : b, d, c, f))에 대응시킬 수 있음을 알 수 있다. 뿐만 아니라 두 개 이상의 상태변수들로부터는 해당 m-블록 분할(y1과 y2로부터 p12=(a, d : b : c, e : f)의 4-블록 분할)을 대응시킬 수 있다. 즉, 상태할당과 블록 분할은 구별 없이 같이 생각할 수 있는 것인데, 블록 분할로부터순차회로 내의 메모리 소자들간의 상호 의존도를 유추해 낼 수 있다. 이를 그림 5에서 살펴보면 다음상태 변수 Y3은 현재상태 변수 y1과 y2 및 입력변수에 의존적임을 알 수 있다. 그리고 메모리소자 FF3과 FF2, FF3과 FF1간에는 이러한 상호궤환이 없다. 이와 같이 상태할당(분할)을 통하여 구현된 모든

순차회로의 테스트를 용이하게 하는 요인은 분할쌍의 결과이다. 이러한 분할을 이용한 상태 할당은 회로의 면적최소화면에서도 바람직한 결과를 낼 수 있다는 것이 알려져 있다[10,12]. 본 논문에서는 임의의 상태 할당(분할)을 가능한 한 분할쌍을 연속적으로 이용한다는 면에서는 [11]과 같으나 분할이 반드시 2-블록 분할이 되도록 하지 않고 m-블록 분할까지 모두 허용한다는 것이 [11]과 다르다.

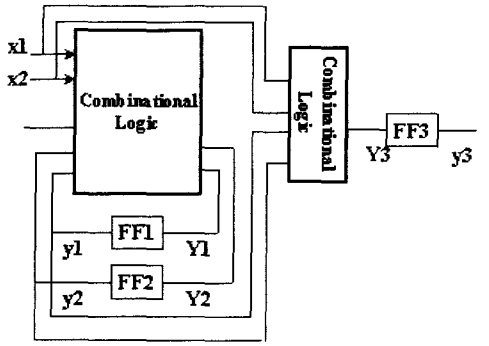


그림 5. 분할쌍에 의한 회로 구조
Fig. 5. Circuit Structure by Partition Pair.

이와 같은 차이점으로부터 부분 스캔을 위한 상태할당에서 본 논문에서의 방법이 [11]에서의 방법보다 더욱 일반화되어진 방법이라고 할 수 있다. 즉, [11]에서의 방법은 상태 변수 하나들간의 의존성만을 고려하여 상태할당을 수행하는 반면에, 본 방법은 상태 변수 그룹(그룹에는 상태변수가 1개 이상 있을 수 있음)들간의 의존성을 고려하여 상태할당을 수행함으로 더욱 일반화되어지고, 따라서 부분 스캔의 측면에서 상태 변수들간의 의존성을 더욱 전역적(global)으로 고려하여 최소화할 수 있다. 이를 좀 더 구체적인 상황으로서 설명하면 다음과 같다. 임의의 유한상태기 M을 2-블록 분할에 의하여 부분 스캔을 고려한 분할쌍을 구하고 상태변수를 구한 경우를 Y1, Y2, Y3, Y4, Y5, Y6라고 했을 때 이들 Yi(i = 1, 2, .. 또는 6)와 Yj (i = 1, 2, .. 또는 6)간에는 상호 의존성이 존재하게 되어 부분 스캔의 측면에서는 바람직하지 않게 되는 반면에, 4-블록 분할에 의하여 부분 스캔을 고려한 분할쌍을 구하고 상태변수를 구한 경우를 Ya, Yb, Yc라고 했을 때 이들 간에는 일방 의존성만이 존재하게 됨으로 부분 스캔의 측면에서 바람직하게 될 수 있다. 여기에서 바람직하다는 의미는 첫째로는 적은 수의 플립플롭을 스캔 가능하게 만듦으로서 오버헤드를 최소화할 수 있다는 것과, 둘째로는 좀 더 적은 수의 스캔 가능 플립플롭만을 사용하는 것이 가능하지는 않더라도 이와 같은 상태할당을 통한 상태변수들간의 의존성 정보를 이용하여 효과적으로(예로 좀 더 빠른 시간 안에) 스캔이 필요한 플립플롭들을 선택할 수 있다는 것이다.

Ⅳ. 상태변수들의 의존성을 줄이는 방법

본 장에서는 상태 변수들의 의존성을 줄이는 방법을 설명하는데 필요한 정의는 다음과 같다.

정의 8] 닫혀진 분할(Closed partition) : P 분할되어진 m -블록 내에 있는 상태들에 대해 다음 상태들이 분할된 m-블록 내에 존재한 경우

정의 9] $\pi(0)$: 닫힌 분할 중 적(product)을 통해 하나의 블록 내에 상태 하나만이 존재하는 경우

정의 10] y1,y2, … , yk : 상태들에 할당될 변수들

정의 11] a,b,c, … : 상태(state)들

정의 12] predecessor partition (γ(ab)) : 한 블록(ab), 나머지 상태들은 분리된 블록을 포함한 분할

정의 13] m-partition = m(γ(ab)) : r(ab)에 의해 다음상태들간의 적을 통해 유도되어진 다음 상태들을 포함하는 가장 작은 분할

정의 14] M-partition = M(γ) : m(ab)에 의해 분할된 블록들에 포함되어지는 γ들의 합(sum)을 결정하는 가장 큰 분할

정의 15] Mm pairs : (M(γ1),m(γ1)) 는 각각의 입력에 대하여 M(γ1)의 같은 블록에 상태 s1과 s2가 같이 있으면 i&s1, i&s2가 m(γ1)에서 같은 블록에 있는 분할의 순서쌍(ordered pair)

●방법 1 : 우선 임의의 상태들 a, b에서 predecessor partition(γab)으로 m-partition{m(γab)}계산한다. 같은 방식으로 m(γab), m(γac) ,…, m(γcf)를 구한다. 다음은 이들 m(γab)의 모든 가능한 sum을 구하는 M-partition을 구한다. 나머지 역시 같은 방식으로 남아있는 M-partition을 구한다. 이렇게 하면 여러 개의 Mm pairs를 구할 수 있다.

상기 방법은 메모리 소자들간의 상호 의존성을 줄일 뿐만 아니라 상태할당을 위한 선택 범위 또한 줄여준다. 그러나 구한 Mm 분할쌍 모두를 적용하는 것 또한 비효율적인 단점이 있다. 이러한 단점을 다음과 같은 방법으로 보완한다.

● **방법 2** : 구한 Mm 분할쌍들 중에서 closed partition이면서 $\pi(0)$인 것을 산출해낸다. 위의 조건을 만족한 분할쌍에 상태할당을 통하여 메모리 소자들간에 의존성이 감소되는 것을 확인 할 수 있다.

다음은 그림 4의 상태천이표 M을 이용하여 본 논문에서 적용한 방법으로 구한 예를 보여주고 있다.

```
                p1={a,b,c ; d,e,f}
                P2={a,e ; b,f  ; c, d}
   y1y2y3       p1·p2 = { a ; b ; c ; d ; e ; f}=Π(0)
a : 0 0 0
b : 0 1 1
c : 0 1 0
d : 1 1 0       logical equation of assignment
e : 1 0 0       Y1=f1(x1,y1)
f : 1 1 1       Y2=f2(x1,x2,y3)
                Y3=f3(x1, x2, y2)
```

그림 6. Mm pairs에서 $\pi(0)$를 만족하는 예
Fig. 6. Example satisfying with $\pi(0)$ in Mm pairs.

그림 6에서는 상태변수 Y1 과 Y2는 상호 궤한루프가 줄어들어 부분 스캔에 의해 테스트가 용이한 반면에 그림 7에서는 상대적으로 메모리 소자들간의 상호 궤한루프가 형성되므로 회로를 부분스캔이나 논 스캔에 의하여 테스트하는 것이 용이하지 못할 것이다.

```
                p1={a,c,f  : b  : d,e }
                P2={a,b,c,d,f  : e}
   y1y2y3       p1·p2 = { a,c,f ; b  ; d ; e } ≠ Π(0)
a : 0 0 0
b : 1 1 0
c : 0 0 0
d : 1 0 0       logical equation of assignment
e : 1 0 1       Y1=f1(x1,y1)
f : 0 0 0       Y2=f2(x1,x2,y1,y2,y3)
                Y3=f3(x1, x2,y1,y2,y3)
```

그림 7. Mm pairs에서 $\pi(0)$를 만족하지 못하는 예
Fig. 7. Example not satisfying with $\pi(0)$ in Mm pairs.

또한 그림 8은 메모리 소자들간의 상호 의존성을 최소화하는 상태할당 알고리듬을 보여주고 있다.

```
1. encoding 할 n state를 위한 k= ⌈ log n ⌉ state
   변수 사용
2. while ( states >2) {
     상태천이표에서 M, m구함
     IF (M, m = ( (분할쌍 AND Closed partition) )
       Mm pairs
       Mm pairs 중에서 π(0)를 구함
       M 분할내에서 상태변수 yr이 선택 (1≤r<k )
       state 변수에 state value assign
       m 분할 내에서 상태 변수yk-r 이 선택
       state 변수에 state value assign
   }
3. Block partition 시 partition 간에partition pair가
   아닌 경우 2의 과정을반복
```

그림 8. 상태할당 알고리듬
Fig. 8. State Assignment Algorithm.

V. 실험결과

부분 스캔을 위한 유한상태기 합성 알고리듬의 성능은 표 2에서 보여주고 있다. 실험 방식은 One-hot, Random, Jedi, 2-block 상태할당 알고리듬을 수행하고 논리합성 단계를 거친 합성 결과와 본 논문에서의 상태할당 알고리듬을 수행하고 논리합성 단계를 거친 합성 결과에 대하여 순차적 ATPG를 수행시켜 고장 점검도를 비교하였다. 실험결과를 얻기 위해 사용한 Tool은Berkeley 대학에서 개발한 논리 합성기인 SIS, 논리 합성 후 나온 blif 파일을 bench 파일로 바꾸는 스크립트 bliftobench, Illinois 대학에서 만든 순차회로용 테스트패턴 생성기인 HITEC을 실험하는 Tool로 사용하였다. 표 2에서 보는 바와 같이 본 연구에서 제안하는 알고리듬을 상태할당에 이용하는 경우 기존의 상태할당 방법(One-hot, Random, Jedi)에 비해 테스트가능도가 현저히 향상되는 것을 알 수 있다. scf회로의 one-hot 상태할당 방법은 SIS에서 상태할당이 되지 않은 경우이다. 특히 don't care set을 많이 포함하는 s1과 styr 회로에서 지극히 낮은 fault coverage를 가져오는 결과는 논리 합성기인 SIS tool 내에서 각 노드를 간소화하는 기능 등 논리 합성에 필요한 기능을 제대로 수행하지 못하여 효율적으로 fault를 검출하지 못하는 구조로 합성되기 때문이다. tbk회로는 다른 상태할당 방법에 비해 고장점검도가 높아졌으며, keyb 회로는 one-hot에 비해 0.74% 낮지만 jedi, random, 2-block 방식에 비해 각각 5.38%, 1.84%, 3.22% 증진되었다. 모든 회로에 대해서

m-block 방식이 2-block 방식보다 항상 더 높은 고장점검도가 나타내지는 않았지만 대체적으로 m-block 방식이 상태 의존성을 최소화하여 테스팅 관점에서 효과적임을 보여준다.

표 2. 상태할당을 통한 고장점검도
Table.2. Fault Coverage using State Assignment.

Circuit	Ns /Nb	고장점검도(단위:%)				
		Jedi	Random	Mm pairs	2-block	One-hot
Mark1	16/4	98.10	94.47	98.85	98.85	97.12
bbsse	16/4	98.24	90.66	98.85	98.12	97.80
keyb	19/5	91.50	95.04	96.88	93.66	97.62
s1	20/5	1.97	0.88	1.06	0.84	51.69
s832	25/5	97.56	98.88	97.92	45.61	86.43
styr	30/5	1.36	29.32	1.28	0.41	87.15
tbk	32/5	96.97	98.59	98.98	98.98	97.38
s1494	48/6	96.81	96.34	94.87	98.26	56.97
scf	121/7	95.76	44.59	96.72	97.66	수행불능

표 3. 상태할당을 통한 Area/Delay비교표( )
: gate수
Table.3. Comparison of Area/Delay upon Different State Assignments.

Circuit	Ns/ Nb	상태 할당을 통한 면적 및 속도				
		Jedi	Random	Mm	2-block	One-hot
Mark1	16/4	(44)	(51)	(41)	(41)	(68)
		18.72	14.97	16.22	16.22	12.26
bbsse	16/4	(64)	(65)	(62)	(64)	(84)
		19.65	24.55	17.84	24.58	18.43
keyb	19/5	(99)	(162)	(90)	(97)	(188)
		30.22	36.28	27.27	30.44	19.36
s1	20/5	(81)	(108)	(93)	(90)	(191)
		26.77	41.38	23.75	26.50	15.60
s832	25/5	(149)	(169)	(146)	(412)	(201)
		25.28	20.70	22.13	27.96	14.84
styr	30/5	(211)	(274)	(200)	(171)	(315)
		73.73	68.29	42.94	59.35	19.66
tbk	32/5	(86)	(140)	(89)	(89)	(131)
		28.32	46.82	34.79	34.79	12.88
s1494	48/6	(264)	(284)	(261)	(252)	(366)
		58.52	73.99	55.02	49.77	18.21
scf	121/7	(364)	(465)	(358)	(355)	-
		51.97	45.62	45.87	45.83	-

표 3은 area 및 delay를 관측한 결과이며 영역 면에서 최적화 된 jedi 방법에 근접 또는 좀더 나은 결과를 나타내었고 속도 면에서는 효율적인 속도를 나타내는 one hot방법에 대해 기대되는 만큼 높은 결과를 나타내지는 않지만 jedi 보다는 나은 결과를 나타낸다. 결국 모든 기존 방법과 본 방안이 유사한 결과를 나타내지는 않았지만 본 논문의 목적은 고장점검도 증진에 있었으며 실험을 통하여 속도 면에서 one hot과는 차이가 있었지만 다른 방법에 비해서 jedi 등 기존 최적의 상태 할당과 큰 차이가 없었음을 확인하였습니다.

VI. 결론 및 향후 연구 계획

본 논문에서는 m-블록 분할을 이용한 상태할당(또는 분할)을 통하여 테스트 가능도 증가 및 면적최소화를 효과적으로 동시에 달성 할 수 있는 방법을 제안하였다. 고장시뮬레이션을 수행하여 다른 상태할당 기법에 비해 높거나 유사한 고장점검도를 나타냄을 알 수 있었고, 영역과 속도 관점에서 볼 때도 효과적임을 알 수 있었다. 향후 연구 계획으로는2-block 과 m-block을 병행하는 최적의 상태 분할 방법에 대한 연구를 진행할 예정이다.

참 고 문 헌

[1] M. Abramovici et al., Digital Systems Testing and Testable Design, Computer Science Press, 1994.

[2] K. T. Cheng and V. D. Agrawal, "A Partial Scan Method for Sequential Circuits with Feedback," IEEE Trans. on Computers, Vol. 39, No. 4, pp. 544-548, April 1990.

[3] Randy H. Katz, Contemporary Logic Design, University of California Benjamin Cummings/Addison Wesley Publishing Company, 1993.

[4] Xuejun Du, Gary Hachtel, Bill Lin, and A. Richard Newton, "MUSE: A MUltilevel Symbolic Encoding Algorithm for State Assignment," IEEE Trans on CAD., Vol. 10, NO. 1, pp. 28-38,January 1991.

[5] E. Goldberg et al., "Theory and Algorithms for Hypercube Embedding," IEEE Trans on CAD., Vol. 17, pp. 472-488, June 1998.

[6] Saeyang Yang and Maciej J. Ciesielski, "Optimum and Suboptimum Algorithms for Input Encoding and Its Relationship to Logic Minimization," IEEE Trans. on CAD., Vol 10. No. 1. pp. 4-12, Jan. 1991.

[7] D. B. Armstrong, "A Programmed Algorithm for Assigning Internal Codes to Sequential Machines," IRE Trans. on Computers, Vol. EC-11, pp. 466-472, Aug. 1962.

[8] G. De Micheli, "Symbolic Design of Combinational Sequential Logic Circuits Implemented by Two-level Logic Macros," IEEE TCAD, Vol. CAD-5, pp. 597-616, Oct. 1986.

[9] S. Devadas et al., "MUSTANG: State assignment of finite state machines targeting multi-level logic implementations," IEEE TCAD. Vol. 7, pp. 1290-1300, Dec. 1988.

[10] T. Villa et al., Synthesis of FSMs : Logic Optimization. New York: Kluwer Academic, 1997.

[11] K. T. Cheng, and V. D. Agrawal, "Design of Sequential Machines for Efficient Test Generation," in Proc. of ICCAD, pp. 358-361, 1989.

[12] R. K. Brayton, G. D. Hatchel, C. T. McMullen, and A. L. Sangiovanni-Vincentelli, Logic Minimization Algorithms for VLSI Synthesis, Norwell, MA: Kluwer Academic, 1984.

[13] Z. Kohavi, Switching and Finite Automata Theory, McGraw-Hill, 1978.

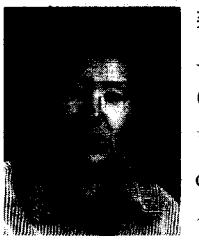
저 자 소 개

曺 尙 煜(學生會員)

한양대학교 전자계산학과 학사(1997), 한양대학교 전자계산학과 석사(1999), 한양대학교 전자계산학과 박사과정(2000-현재), 관심분야는 테스트 합성, Scan Design, VLSI 시스템 & 테스팅, ASIC 설계 등

朴 成 柱(正會員)

한양대학교 전자공학과 학사(1983), 금성사 소프트웨어개발(1983-1986). Univ. of Massachusetts 전기 및 컴퓨터공학과 박사(1992), IBM Microelectronics 연구스텝(1992-1994), 한양대학교 전자계산학과 부교수(1995-현재). 관심분야는 테스트 합성, Built-In Self Test, Scan Design, ATPG, ASIC 설계, 고속 신호처리 시스템 설계, 그래프이론 등



梁 世 陽(正會員)

고려대학교 전자공학과 공학사(1981), 고려대학교 전자공학과 공학석사(1985), Univ. of Massachusetts 전기컴퓨터공학과 공학박사(1990), Microelectronics Center of North Carolina 선임연구원(1990-1991), 부산대학교 컴퓨터공학과 부교수(1991-현재), 관심분야는 VLSI-CAD 및 테스팅, System/Logic Verification