論文 90-27-7-18

Domino CMOS 논리회로의 테스트 생성에 관한 연구

(A Study on Test Generation for Domino CMOS Logic Circuits)

李 在 旼* 李 俊 模* 鄭 峻 模**

(Jae Min Lee, Joon Mo Lee, and Joon Mo Jung)

要 約

본 논문에서는 Domino CMOS 논리회로의 고장 검출을 위한 새로운 테스트 생성방법을 제안한다. Domino CMOS회로내 발생가능한 고장들을 모델화할 때 종래의 stuck-at형의 고장모델만으로는 충분히 모델화할 수 없으므로 회로내 트랜지스터들을 스위치 레벨에서 stuck-open형 및 stuck-on형의 고장과 트랜지스터 리이드 간의 bridging 고장으로 모델화한다. Domino CMOS회로의 테스트생성 문제는 함수블럭의 테스트 생성문제로 귀착됨을 보이고 테스트 집합을 별도의 알고리듬을 사용하지 않고 회로설계 과정에서 거치게 되는 논리최소화를 이용하여 최소화된 함수의 적항들로 부터 생성하는 방법을 제안하고 테스트 집합 간소화를 위한 알고리듬도 제시한다. 제안한 방법은 다양한 형태의 회로에 적용하기 쉬우며 설계시스템 내 자동 테스트 생성기의 구축을 용이하게 한다. 제안한 방법의 알고리듬을 실현하고 예제에 적용해 봄으로써 그 유효성을 확인한다.

Abstract

In this paper a new test generation method for Domino CMOS logic circuits is proposed.

Because the stuck-at type fault is not adequate for Domino CMOS circuits the stuck-open fault, stuck-on fault and bridging fault are considered as fault models. It is shown that the test generation problem of Domino CMOS circuits results in functional block test generation problem. Test set is generated by using the logic minimizer which is a part of logic design system. An algorithm for reduction of test set is described. The proposed test method can be easily applied to various figures of circuits and make it easy to construct automatic test generator in design system. The proposed algorithms are programmed and their efficiency is confirmed by examples.

*正會員, 関東大學校 電子工學科
(Dept. of Elec. Eng., Kwandong Univ.)
**正會員 漢陽大學校 大學院 電子工學科
(Dept. of Elec. Eng., Graduate School of Hanyang Univ.)
接受日字 : 1990年 4月 14日
(※본 논문은 한국과학재단 연구비 지원에 의해 이루어진 결과임.)

I. 서 론

반도체 기술의 급속한 발전으로 회로의 복잡도가 크게 증가함에 따라 대규모 회로의 테스트가 점차 어려워지고 이에 따른 설계비용의 증가도 큰 문제가 되고 있다. 최근 기존 CMOS의 장점을 가지면서 타임스큐(time skew)의 문제점을 해결하고 CMOS 보다 칩 면적이 작으며, 속도가 빠르고 NMOS 기술을

그대로 이용할 수 있는등 많은 장점을 갖는 Domino CMOS에 대한 연구가 활발히 이루어지고 있으며 앞으로 VLSI설계에 많이 사용될 전망이어서 이에 대한 테스트 방법의 연구가 더욱 필요시 되고 있다. 기존의 VLSI 테스트를 위한 테스트 생성 알고리듬은 대부분 게이트 레벨에서 발생 가능한 고장을 모델링하여 테스트 집합을 생성하였으나 MOS계열의 회로를 테스트하기 위하여 고장을 모델링할 때 종래의 stuck-at형의 고장 모델만으로는 높은 고장 검출률(fault coverage)을 얻을 수 없다는 연구결과가 최근 많이 보고되고 있다.[5,7] 즉 종래의 stuck-at 고장으로는 MOS 트랜지스터의 양방향 특성을 모델링할 수 없으며 트랜지스터의 stuck-open고장이나 stuck-on고장을 검출하기 어려운 문제점을 갖고 있다. 따라서 이러한 문제점을 해결하기 위해서 MOS회로를 스위치 레벨에서 고찰하여 트랜지스터들을 stuck-open과 stuck-on 고장으로 모델링하고 이에 대한 테스트 집합을 생성하는 새로운 방법이 최근 많이 연구되고 있다.[2-7,9-12]. Domino CMOS 회로는 클럭킹 게이트와 인버어터 및 NMOS함수 블럭으로 구성되어 있어 CMOS부분인 클럭킹 게이트와 인버어터의 테스트 방법과 함께 NMOS함수 블럭의 효율적인 테스트 방식이 필요하다. 기존의 일반 Domino CMOS 회로의 테스트에 관한 논문[10]에서는 클럭킹 게이트와 인버어터의 stuck-open 고장검출 방법에 대해 주로 논하였으나 NMOS함수블럭의 테스트 알고리듬에 대해서는 구체적으로 제시하지 않고 있다. 그런데 Domino CMOS내 함수블럭의 테스트는 일반 NMOS 회로의 테스트 방식을 이용할 수 있는데 NMOS회로의 테스트를 위해 S. Y. H. Su등은[12] 행렬연산을 이용한 방법을 제시하였고 R. I. Damper등은[7] 경로대수(path algebra)를 이용하여 게이트로 변환이 불가능한 조합논리회로를 블럭화하고 여기에 D-알고리듬을 적용한 방법을 제안한바 있다. 그런데 S. Y. H. Su 등이 제안한 행 변환 알고리듬은 MOS트랜지스터의 양방향 특성을 모델화할 수 없는 단점을 갖고 있으며 R. I. Damper등이 제안한 경로대수를 이용하는 테스트 생성 방법에는 특이커버(singular cover)생성과 D-큐브 생성과정이 복잡한 단점이 있다.

본 논문에서는 Domino CMOS회로의 고장검출을 위해 테스트집합을 생성할 때 별도의 알고리듬과 연산을 필요로 하지 않고 회로 설계 과정에서 거치게 되는 논리함수 최소화[8,13,14]를 이용하여 Domino CMOS 회로의 클럭킹 게이트와 인버어터 및 NMOS 함수블럭내 트랜지스터들의 stuck-open 고장과 stuck-on 고장 및 트랜지스터 리이드 간의 bridging 고장을 검출할 수 있는 새로운 테스트 생성방법을 제안한다. 제안하는 방법은 종래의 방법에 비해 테스트 생성이 쉽고 D-알고리듬과 같이 테스트 생성을 위한 논리의 반복 수행이 적어 효율적인 테스트 생성이 가능한 장점을 갖고 있으며 회로의 형태에 따라 테스트 생성 알고리듬이 크게 변하지 않는 특징을 갖고 있다. 또한 본 논문에서는 종래의 논문들에서 구체적으로 언급하지 않은 테스트 집합의 간소화방법도 제시한다.

Ⅱ. 고장 모델화

Domino CMOS회로의 고장을 모델화할 때 종래의 게이트 레벨에서 사용하던 stuck-at형의 고장만으로는 회로내 발생 가능한 고장을 충분히 모델화할 수 없는데 Domino CMOS회로를 게이트 레벨에서 stuck-at 형으로만 모델화할 경우 다음과 같은 단점을 갖게 된다.

(1) Domino CMOS회로에서는 stuck-at 형의 고장뿐 아니라 트랜지스터의 stuck-open 및 stuck-on 형의 고장이 발생한다.

(2) Domino CMOS회로내 각 트랜지스터들은 양방향성인데 게이트레벨의 등가회로에서는 모델화되지 않는다.

(3) MOS트랜지스터는 고장 발생시 용량성 부하로서 작용하여 래치(latch)처럼 동작한다.

(4) stuck-open 고장이나 트랜지스터 리이드간 bridging 고장 발생시 논리함수의 내용이 바뀐다.

따라서 최근에는 보다 실질적으로 회로 내 트랜지스터를 하나의 스위치로 보는 스위치 레벨에서의 고장모델이 사용되고 있다.

그림1은 Domino CMOS 회로내 발생 가능한 고장 모델을 나타낸 것이다.

그림1의 회로에서 트랜지스터 tr1, tr2, tr3, tr4에 stuck-on 고장이 발생했을 경우 이 고장이 회로에 미치는 영향은 p형 트랜지스터 tr1, tr3의 저항 Rp와 n형 트랜지스터(tr1, tr4, 함수블럭)의 저항 Rn의 비율에 의하여 결정된다. 만약 $Rp \ll Rn$이면 p형 트랜지스터가 "dominant"라 하는데 이는 p측 및 n측이 모두 on될 경우 출력의 값이 p-dominant 이면 1이 되고 n-dominant이면 0이 되는 것을 의미한다. p-dominant인 경우 tr1, tr3 트랜지스터에 stuck-on고장이 발생하면 출력은 stuck-at 고장형태 (tr1 고장시 stuck-at-0, tr3 고장시 stuck-at-1)로 되는데 이 고장은 검출이 가능하다. 그러나 n-dominant한 경우 stuck-on 고장인 p형 트랜지스터는 NMOS 기술에서의 디플리션(depletion)트랜지스터와 같은 기능을 갖

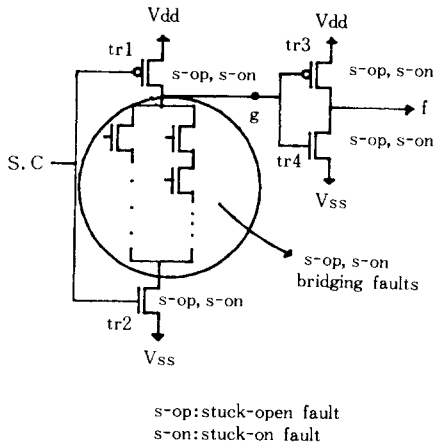


그림 1. Domino CMOS회로의 고장 모델
Fig. 1. Fault models for domino CMOS circuits.

는 부하로서 동작한다. 즉 회로가 NMOS와 같이 동작한다. 이러한 고장으로 회로의 동작시간 특성이 크게 나빠지지 않고 고장이 전류계에 의하여 감지되지 않을 정도라면 이러한 고장은 검출이 거의 불가능하다. 즉 stuck-on 고장이 존재함에도 불구하고 회로가 정상적으로 동작하게 된다. tr4의 stuck-on 고장이 발생할 경우 n-dominant이면 출력은 stuck-at 0의 고장형태가 되며 이러한 고장은 검출 가능하다. 그러나 p-dominant이면 위에서 기술한 것과 같은 이유로 정상적인 동작을 하게 되어 검출이 불가능해 진다. tr2의 stuck-on 고장이 발생할 경우 클럭이 1일 때 정상 상태와 같이 동작하고 클럭이 0일 때도 p-dominant인 경우 f=0이면 출력은 0이 되지만 f=1이면 함수블럭의 저항 때문에 dominant 관계를 계산할 수 없어 경우에 따라 불규칙한 출력값을 갖게되므로 고장 검출이 곤란하다. 본 논문에서는 테스트를 용이하게 하기 위하여 회로 설계 시 p-dominant 하게 설계한다고 가정한다. 이 가정하에서 트랜지스터 tr1, tr3의 stuck-on고장은 출력의 stuck-at 고장으로 검출되며 트랜지스터 tr2, tr4의 stuck-on 고장은 정상회로와 같이 동작하므로 논리적인 방법으로는 검출이 어렵다. 또 Domino CMOS회로 내 트랜지스터가 영구적으로 단선되는 고장 즉 stuck-open 고장이 발생하면 출력이 고 임피던스 상태가 되고 이 때 출력값은 CMOS회로의 전하 저장기능으로 인하여 이전 상태의 논리값을 그대로 갖게되어 조합 논리회로를 순서 논리회로처럼 동작하도록 만든다. 이러한 stuck-open 고장을 검출하기 위해서는 일반적으로 두개의 테스트 패턴 즉 테스트 시이퀀스가 필요하게 된다.

그림 1과 같은 Domino CMOS회로 내 발생 가능한 stuck-open 고장은 다음 4가지 형태로 나눌 수있다.

[형태 1] tr3의 단선으로 인한 stuck-open고장:이 고장은 인버어터의 출력과 Vdd와의 연결을 끊는 것과 같은 효과로 pull-down 회로가 off될 때 pull-up회로가 on되지 못한다. 따라서 출력의 논리값은 정상상태일 때 1 이어야 하나 stuck-open 고장으로 고 임피던스가 되어 전 상태값을 가지게 된다. 여기서 전 상태값의 지속시간은 출력단에서의 누설전류에 의하여 결정된다.

[형태 2] tr1의 단선으로 일어나는 stuck-open고장: 이 고장은 g(f')와 Vdd의 연결을 끊음으로써 클럭이 0일 때 g가 1값을 가질 수 없게 한다. 이 때 g는 고 임피던스가 되어 이전 상태값을 유지하게 된다.

[형태 3] tr4의 단선으로 일어나는 고장:이 고장은 인버어터의 출력과 Vss와의 연결을 끊는 것과 같은 효과이며 pull-up회로가 off될 때 pull-down회로가 on되지 못하여 출력은 고 임피던스 상태가 된다. 정상상태일 때 출력은 0이어야 하나 이 고장 발생시 이전 상태값을 가지게 된다.

[형태 4] tr2의 단선으로 일어나는 stuck-open고장: g와 Vss와의 연결을, 끊음으로써 클럭이 1일 때 함수블럭의 on/off에 관계없이 g를 고 임피던스로 만든다. 이 때 g의 전상태는 클럭이 0일 때 1이므로 g는 1을 유지하게 된다.

stuck-open 고장을 검출하기 위해서는 2개의 연속된 테스트 패턴이 필요한데 앞의 테스트 패턴을 초기화 패턴이라 하며 뒤의 테스트 패턴을 검사테스트 패턴이라 한다. 연속된 두 테스트 패턴은 이를 정상회로에 인가하였을 때 출력값이 보수로 얻어지도록 구성된다. 만약 회로에 고장이 발생하면 두 테스트 패턴에 의한 출력값이 모두 초기화 테스트패턴에 의한 논리값과 같게 되므로 고장 유무를 확인할 수 있다.

형태1에서 형태4까지의 stuck-open 고장을 검출할 수 있는 테스트 패턴을 구하면 표 1과 같다.

표 1에서 알 수 있듯이 형태 1및 형태 4의 고장은 클럭펄스가 인가될 때 SC가 0에서 1로 변화하는 과정이 존재하므로 함수블럭내 트랜지스터의 stuck-open고장을 검출하기 위한 테스트패턴에 의해 검출

표 1. 그림 1의 회로내 stuck-open 고장검출을 위한 테스트 패턴

Table 1. Test patterns for stuck-open faults in fig. 1 circuit.

s-open 고장	SC	함수블럭경로	g	f
형태 1	0	on	1(1)	0(0)
	1	on	0(0)	1(0)
형태 2	0	on	1(X)	0(X)
	1	on	0(0)	1(1)
	0	off	1(0)	0(1)
	1	off	1(0)	0(1)
형태 3	0	on	1(1)	0(X)
	1	on	0(0)	1(1)
	0	off	1(1)	0(1)
	1	off	1(1)	0(1)
형태 4	0	on	1(1)	0(0)
	1	on	0(1)	1(0)

여기서 ( )은 고장시의 논리값을 표시하며 X는 미지의 논리값을 가르킨다.

할 수 있다. 또 형태 2 및 형태 3의 고장은 함수블럭내 트랜지스터의 stuck-open고장에 대한 테스트와 stuck-on 고장에 대한 테스트를 연속해서 인가함으로써 검출 가능하다. 따라서 Domino CMOS 회로의 함수블럭 이외의 부분에 대한 stuck-open 고장 및 stuck-on 고장은 함수블럭내 트랜지스터의 stuck-open 고장 및 stuck-on 고장을 검출하기 위한 테스트 패턴들에 의해 검출할 수 있다.

결국 함수블럭내 트랜지스터들의 stuck-open 고장 및 stuck-on고장 검출을 위한 테스트 생성이 Domino CMOS회로의 테스트를 위한 가장 중요한 부분이 된다.

본 논문에서 구체적으로 고려하고 있는 고장은 다음과 같다.

(1) 클럭킹 게이트의 stuck-open 및 stuck-on 고장
(2) CMOS 인버어터의 stuck-open 및 stuck-on 고장
(3) 함수블럭 내 트랜지스터의 stuck-open 및 stuck-on 고장
(4) 함수블럭 내 트랜지스터 리이드 간의 bridging 고장

[정의 1] (1), (2), (3), (4)의 고장 집합을 FS (fault set)라 한다.

Ⅲ. 테스트 생성의 원리

그림 2는 Domino CMOS 회로와 회로내 NMOS함수블럭을 그래프로 나타낸 것이다. 여기서 절점(vertex)은 회로소자의 연결점을, 가지(edge)는 트랜지스터를 나타낸다. NMOS 함수블럭내 트랜지스터들에 대한 stuck-open 및 stuck-on고장 검출을 위한 테스트 집합의 생성 문제는 그래프에서 모든 path-set 및 cut-set을 찾은 후 이로부터 최소의 테스트 집합을 구하는 문제로서 이는 일반 조합논리회로의 테스트 생성문제와 마찬가지로 NP-Complete 하다.

[정의 2] NMOS 함수블럭 내 트랜지스터들의 stuck open 고장을 검출할 수 있는 테스트 집합을 SPTS (stuck-open test set)이라 한다.

[정의 3] NMOS 함수블럭 내 트랜지스터들의 stuck-on 고장을 검출할 수 있는 테스트 집합을 SNTS (stuck-on test set)라 한다.

[SPTS의 생성]

그림 2의 회로는 회로 설계 과정에서 논리최소화[13,14]를 거쳐 적항 사이에 리던던시(redundancy)가 없는 논리함수로 부터 설계된 것이라 하자. 논리함수는 $f(a, b, c, d, e) = ad + be + ace + bcd$으로 표현되며 회로의 함수블럭 부분에 대함 함수식을 fd로 표현하면 $fd = f = ad + be + ace + bcd$가 된다.

[정리 1] 회로의 함수블럭 부분에 대한 함수식 fd의 적항들은 회로내 트랜지스터들의 stuck-open 고장을 검출할 수 있는 테스트 패턴 SPTS가 된다.

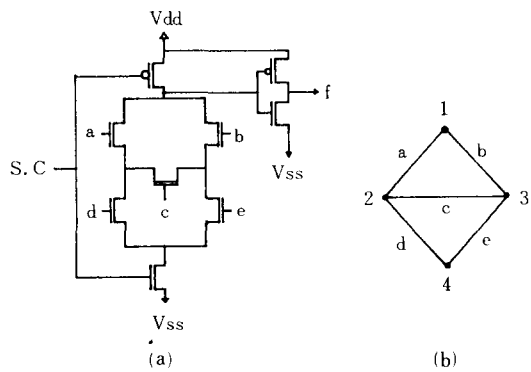


그림 2. Domino CMOS회로와 함수블럭의 그래프 표현
(a) Domino CMOS회로
(b) 함수블럭의 그래프 표현

Fig. 2. Domino CMOS and graph representation for functional block.
(a) Domino CMOS,
(b) Graph representation for functional block.

[증 명] fd를 구성하는 각 적항들은 Vdd로부터Vss에 이르는 모든 경로들을 나타내므로 표 2와 같이 적항을 구성하는 각 변수의 논리값을 1로하고 나머지 변수의 논리값을 0으로 하면 적항으로 이루어지는 경로만이 활성화 되어 경로상의 단일 및 다중 stuck-open 고장이 발생할 때 활성화된 경로가 차단되므로 출력에서 고장을 검출할 수 있다. (증명 끝)

표 2. 그림 2의 회로에 대한 SPTS
Table 2. SPTS for the circuit in fig. 2.

	a	b	c	d	e
ad	1	0	0	1	0
be	0	1	0	0	1
ace	1	0	1	0	1
bcd	0	1	1	1	0

[SNTS의 생성]

stuck-on고장 테스트 생성과정은 회로 그래프에서 모든 cut-set을 구하는 문제이므로 stuck-on 고장 테스트 집합은 stuck-open 고장 테스트 집합의 각 패턴을 구성하는 변수를 하나씩 포함하는 패턴들로 이루어진다.

[정리 2] 최소화된 함수식 fd′를 구성하는 적항들은 회로내 트랜지스터들의 stuck-on 고장을 검출할 수 있는 테스트 패턴 SNTS가 된다.

[증 명] stuck-on 고장의 검출을 위해서는 회로 그래프에서 모든 cut-set을 구해야 하므로 Vdd로부터 Vss에 이르는 모든 경로상의 변수를 하나씩 포함하는 테스트 패턴을 구해야 한다. 또한 stuck-on 고장을 검출할때는 cut-set을 구성하는 각변수에 논리값0을 인가해야 하므로 stuck-open 테스트 생성을 위한 테스트 내 변수들을 보수를 취하고 각 항의 변수들을 조합하여 테스트 집합을 생성할 수 있는데 이것은 Boole 대수의 성질에 의해 바로 fd′를 구성하는 적항들이 된다. (증명끝)

표 3은 그림 2의 회로에 대한 SNTS을 나타낸 것이다.

stuck-open 고장의 테스트시와 마찬가지로 cut-set에 의해 생성된 stuck-on 고장 검출용 테스트 집합

표 3. 그림 2의 회로에 대한 SNTS
Table 3. SNTS for the circuit in fig. 2.

	a	b	c	d	e
a′b′	0	0	1	1	1
d′e′	1	1	1	0	0
a′c′e′	0	1	0	1	0
b′c′d′	1	0	0	0	1

도 각 테스트 패턴에 의해 제어되는 트랜지스터들의 단일 및 다중고장 모두를 검출한다.

[bridging 고장의 테스트 생성]

Domino CMOS회로내 발생가능한 고장은 트랜지스터의 stuck-open 및 stuck-on 고장 이외에도 그림 3과 같이 NMOS 함수블럭내 트랜지스터의 리이드간 bridging 고장이 발생할 수 있다. 이와 같은 bridging 고장은 단락되는 부분에 가상의 트랜지스터를 부가하여 새로운 함수를 구하므로써 얻어지는 SPTS 중에서 가상한 트랜지스터의 입력을 포함하고 있는 적항들로 부터 고장 집합을 생성한다. 그림3에서 마디2와 3사이에 bridging 고장이 발생하였을 경우 입력 변수 s를 갖는 가상 트랜지스터를 삽입하면 이에 대한 함수식은 fd=ac+bd+asd+bsc가 되며 가상 트랜지스터의 입력변수 s를 갖는 두 항 asd, bsc에서 각각 s를 제거하면 테스트 패턴은 {ad, bc}가 된다. 이 가운데 하나를 테스트 패턴으로 사용한다.

[정의 4] Domino CMOS회로의 함수블럭 내 트랜지스터 리이드간 bridging 고장을 검출하는 테스트 집합을 BFTS(bridging fault test set)라 한다.

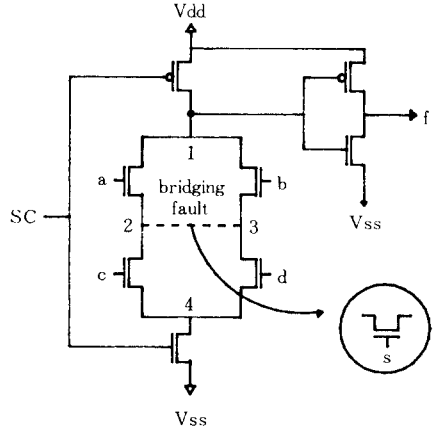


그림 3. 마디 사이의 단락 고장을 갖는 회로
Fig. 3. Circuit with bridging fault.

Ⅳ. 회로의 종류에 따른 테스트 생성

기존의 테스트 생성방법[7] 에서는 직병렬회로의 경우 기본회로와는 달리 행렬표현시 직병렬관계를 나타내기 위해 행렬요소의 표현식이 복잡해지고 이에 따른 연산 복잡도가 증가되는 등의 단점이 있으나 제안한 본 방법에서는 직병렬회로의 테스트 집합을 기본회로의 테스트 방법과 동일하게 논리함수에 대한 연산만으로 테스트 집합을 구할 수 있다. 또한 내부분기(internal fanout)가 있는 회로에 대해서도 함수블럭을 구성하는 적항들이 서로 리던던트하지 않으면 기본회로와 동일한 방법을 적용하여 테스트 집합을 생성할 수 있다.

그림 4는 입력b가 내부 분기되는 회로이다.

$fd = ad + bc + ab + cd + be$

$fd' = a'b'c' + a'c'e' + b'd' + b'd'e'$ 이므로 테스트 집합을 구하면

$SPTS = \{ad, bc, ab, cd, be\}$

$SNTS = \{a'b'c', a'c'e', b'd', b'd'e'\}$ 가 된다.

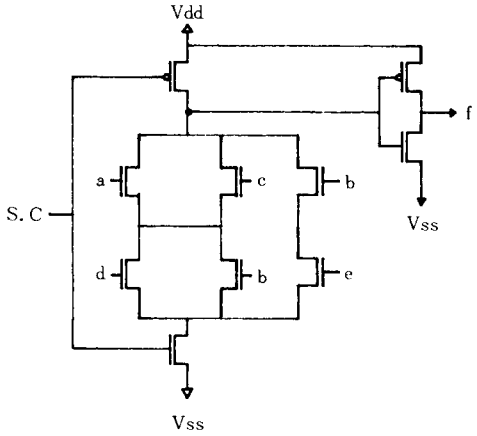


그림 4. 내부 분기입력이 있는 회로.
Fig. 4. Circuit with internal fanout.

Ⅴ. 테스트 집합의 간소화

제안한 방법에 의해 테스트 집합을 생성할 때 그림 2와 같이 한 트랜지스터의 입력변수가 최소화된 함수의 서로 다른 적항들에 포함되어 있을 때 생성한 SPTS나 SNTS내에는 다른 테스트 패턴에 의해 검출되는 고장을 검출하기 위한 테스트 패턴이 존재한다. 즉 테스트 패턴에 리던던시가 존재하게 되는데 이러한 리던던시를 제거하므로써 테스트 길이를 줄이고 테스트 인가(test application)나 테스트 평가(test response evaluation)를 위한 비용을 줄일 수 있다.

그림 2의 회로에 대한 SPTS으로서 {ad, be, ace, bcd}를 얻을 수 있었다. 그런데 이 테스트 패턴 중 실제 stuck-open 고장 검출을 위해서 필요한 패턴은 {ace, bcd}뿐이다. 테스트 집합의 간소화는 논리간소화의 주항선택 과정과 유사한데 일차 생성된 적항군에서 될 수 있는대로 긴 경로 및 cut-set을 감지 가능상태(sensitive state)로 만들면서, 테스트 패턴을 구성하는 변수 모두를 포함하도록 선택하는 과정이다.

[정의 5] 한 변수가 오직 하나의 테스트 패턴에만 포함되어 있을때 이와 같은 테스트 패턴을 ETP(essential test pattern)이라 한다.

[정의 6] 각 테스트 패턴을 구성하는 변수의 갯수를 테스트 패턴의 폭(width)이라 한다.

[정의 7] 논리함수로 부터 생성된 테스트 패턴들에 임의의 변수가 포함되는 횟수를 그 변수의 포함도라 한다.

표 4는 그림 2의 회로내 stuck-open 고장 검출을 위한 테스트 패턴을 간소화하기 위한 표이다.

표 4. 테스트 패턴 간소화를 위한 표
Table 4. Table for test pattern simplification.

	a	b	c	d	e	테스트 패턴의 폭
ad	*			*		2 → 1 → 0
be		*			*	2 → 1 → 0
ace	*		*		*	3 → 0
bcd		*	*	*		3 → 2 → 0
포함도	2	2	2	2	2	

먼저 포함도가 1인 테스트 패턴이 존재하지 않으므로 ETP는 생성되지 않는다. 다시 테스트 패턴의 폭이 최대인 패턴을 고르면 ace, bcd 두개의 테스트 패턴이 선택된다. 이 경우와 같이 테스트 패턴의 폭이 같은 것이 두개 이상 존재하면 이 가운데 포함도의 합이 작은 것을 선택하므로써 해의 결과를 향상시킬 수 있다. 그 이유는 포함도의 합이 작을수록 현재 선택된 테스트 패턴을 구성하는 변수가 아직 선택되지 않은 적항에 적게 포함되므로 이것은 다른 적항이 아직 커버 되지 못한 변수를 많이 갖게되는 조건이 되어 해의 최적도를 높일 수 있다. 또한 포함도가 같은 테스트 패턴들이 다수개 있을때는 최초의 테스트 패턴의 폭이 최소인 것을 선택한다. 이는 활

성화되는 트랜지스터의 갯수를 적게 하므로써 테스트시 전력소모를 줄이고 테스트 시간을 줄일 수 있다. ace와 bcd의 경우 포함도가 동일하므로 여기서는 임의로 ace를 선택한다. ace가 선택된 후 테스트 패턴의 폭이 갱신되고 다시 bcd가 다음 테스트 패턴으로 선택된다. bcd의 선택으로 테스트 패턴의 폭이 모두 0으로 갱신되고 이는 모든 변수 a, b, c, d, e가 ace, bcd에 포함됨을 의미하며 이로써 테스트 패턴 간소화 과정이 끝난다.

이상의 방법으로 SPTS와 SNTS를 구하면

SPTS={ace, bcd}

SNTS={a′c′e′, b′c′d′}가 된다.

[테스트 간소화 알고리듬]

[단계 1] 회로에 대한 함수 fd 및 fd′의 각 적항으로부터 SPTS와 SNTS를 구한다.

[단계 2] SPTS 및 SNTS내 ETP를 생성하고 간소화표에서 제거한다.

[단계 3] SPTS 및 SNTS내 테스트 패턴의 폭이 최대인 것을 선택한다. 동일한 조건을 만족하는 테스트 패턴이 다수개일 때는 포함도의 합이 최소인 것을 선택한다. 포함도의 합에 관한 조건을 만족하는 것이 다수개일 때는 임의로 선택한다. 선택된 테스트 패턴을 간소화표에서 제거한후 테스트 패턴의 폭 및 포함도를 갱신한다.

[단계 4] 간소화표의 나머지 테스트 패턴들에 대해서 단계 3의 과정을 모든 테스트 패턴의 폭이 0이될 때까지 반복한 후 알고리듬을 종료한다.

Ⅵ. 테스트생성 알고리듬

Ⅱ절에서 기술한바와 같이 Domino CMOS내 클럭킹 게이트와 인버어터를 구성하는 트랜지스터들의 stuck-open 고장은 함수블럭내 트랜지스터들의 stuck-open 및 stuck-on고장의 검출을 위한 테스트 시이퀀스에 의해 모두 검출 가능하다.

테스트 패턴 생성 알고리듬 기술을 위해 다음과 같이 정의한다.

[정의 8] T0:클럭 SC=0일 때 NMOS 함수블럭을 활성화시키는 테스트 패턴

[정의 9] T1:클럭 SC=1이고 NMOS 함수블럭을 활성화시키는 테스트 패턴

[정의10] T2:클럭 SC=0일 때 NMOS 함수블럭을 차단시키는 테스트 패턴

[정의11] T3:클럭 SC=1이고 NMOS 함수블럭을 차단시키는 테스트 패턴

[정의12] T4:클럭 C=1이고 bridging 고장 검출을 위한, 가상의 트랜지스터를 포함하는 경로를 활성화하는 테스트 패턴

다음은 Domino CMOS 회로의 테스트 생성 알고리듬이다.

[테스트 생성 알고리듬]

[단계 1] fd와 fd′의 최소화 결과로부터 테스트 집합 SPTS 및 SNTS를 생성한다.

[단계 2] (T0, T1, T2, T3)의 시이퀀스를 생성한다. 이 때 T0, T1에는 SPTS내 패턴들을, T2, T3에는 SNTS내 패턴들을 차례대로 조합한다.

[단계 3] 단계 2의 과정을 SPTS 및 SNTS내 테스트 패턴들이 모두 결합될 때까지 반복한다.

[단계 4] (T0, T4)를 BFTS내 모든 테스트 패턴이 결합될 때까지 반복 조합한다.

[정리 3] 제안한 알고리듬에 의해 생성된 테스트 집합으로 그림 2와 같은 기본 Domino CMOS 회로내 발생가능한 고장 집합 FS에 속한 어떠한 고장도 검출 가능하다.

[증 명] 클럭킹 게이트 및 인버어터의 stuck-open 및 stuck-on 고장은 NMOS 함수블럭내 트랜지스터의 stuck-open 및 stuck-on 고장을 위한 테스트 패턴으로 검출 가능하며, (T0, T1, T2, T3)들로 구성된 테스트 시이퀀스는 함수블럭내 모든 트랜지스터의 stuck-open과 stuck-on 고장을 검출하기 위한 테스트 패턴들이고 (T0, T4)들로 구성된 테스트 시이퀀스는 회로 내 bridging 고장을 검출하기 위한 테스트 패턴들이므로 이를 사용하면 FS내 모든 고장들을 검출할 수 있다. (증명 끝)

Ⅶ. 프로그램 개발 및 검토

제안한 테스트 생성 알고리듬을 APOLLO 워크스테이션(UNIX BSD 4.3)상에서 C언어로 실현하였다. 그림 5는 테스트 생성기(test generator)의 구성도이다.

테스트 생성기에 입력 가능한 화일 형태는 설계 시스템에서 하드웨어 기술 언어로 작성된 회로를 컴파일하여 얻어지는 표준형(standard form)의 논리함수와 논리최소기[13,14]가 허용하는 canonical형 및 2진수형의 논리함수이며 테스트패턴 간소화와 테스트 패턴 조합과정을 통해 최종 테스트 집합을 출력한다. 그림 6은 테스트 생성기의 입력 및 출력의 예를 나타낸 것이다.

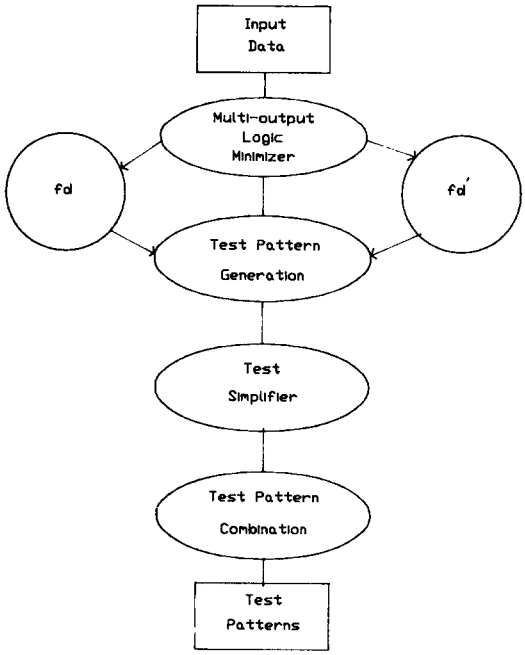


그림 5. 테스트 생성기의 구성도
Fig. 5. Schemetic diagram of test generator.

(a)는 논리함수 $f(a, b, c, d) = \sum(5, 7, 11, 12, 13, 14, 15)$을 각각 표준형(standard form), 십진수형(decimal form) 및 2진 형(binary form)으로 나타낸 것이며 데이타의 첫 번째 줄은 데이타형, 입력수, 출력수 및 최소항수 표준형의 경우는 적항수)를 가리킨다. 또한 (b)는 (a)데이타에 대한 출력양식을 나타낸다.

그림 7의 (a)와 (b)는 그림2 및 그림3의 회로에 대한 최종 테스트 집합을 프로그램을 수행하여 얻은 결과이다.

표5는 개발한 프로그램에 여러가지 회로를 적용하여 얻은 결과이다.

표 5. 적용 예
Table 5. Experimental results.

회로명	입력수	f의 적항수	f'의 적항수	테스트 시이퀀스 길이
C41	4	4	3	9
C51	5	7	5	19
C61	6	16	10	58
C71	7	29	19	173
C81	8	48	42	618

각 회로에 대해 생성된 테스트 시이퀀스로 가정한 고장(FS)들이 100% 검출됨을 확인할 수 있었다.

```
s 4 1 4          d 4 1 7          b 4 1 8
0 1 _ 1          5 1              0101 1
1 1 0 _          7 1              0111 1
1 _ 1 1          11 1             1011 1
1 1 1 _          12 1             1100 1
                 13 1             1101 1
                 14 1             1110 1
                 15 1             1111 1

<standard form>  <decimal form>   <binary form>
```

(a)

```
*** Test Sequences for Domino CMOS ***

SC  a  b  c  d
---------------
0   1  0  1  1
1   1  0  1  1
0   0  1  0  1
1   0  1  0  1
0   1  1  0  0
1   1  1  0  0
0   1  0  1  1
1   1  0  0  1
---------------
```

(b)

그림 6. 테스트 생성기의 입출력 화일 형태
Fig. 6. Input-output file type of test generator.

```
*** Test Sequences for Domino CMOS ***

SC  a  b  c  d  e
------------------
0   1  0  1  0  1
1   1  0  1  0  1
0   0  1  0  1  0
1   0  1  0  1  0
0   0  1  1  1  0
1   0  1  1  1  0
0   1  0  0  0  1
1   1  0  0  0  1
------------------
```

(a)

```
*** Test Sequences for Domino CMOS ***

SC  a  b  c  d
--------------
0   1  0  1  0
1   1  0  1  0
0   0  0  1  1
1   0  0  1  1
0   0  1  0  1
1   0  1  0  1
0   1  1  0  0
1   1  1  0  0
0   1  0  1  0
1   1  0  0  1
--------------
```

(b)

그림 7. 그림2 및 그림3의 회로에 대한 테스트 생성 프로그램 수행 결과
Fig. 7. Program execution results for the circuits in fig. 2 and fig. 3.

Ⅷ. 결 론

VLSI설계시 어떤 논리소자를 선택할 것인가는 설계 및 테스트의 용이성과 비용의 절감이 가장 큰 조건이 된다. Domino CMOS는 이러한 요구를 만족시키기에 충분한 장점을 갖고 있으므로 최근 VLSI 설계에 그 사용이 점차 확대되고 있다. 본 논문에서는 Domino CMOS회로의 고장 검출을 위한 새로운 테스트 생성방법을 제안하였다. 제안한 방법은 논리회로 또는 디지탈 시스템 설계시 거의 필수적으로 거치게 되는 논리최소화의 결과로서 얻어지는 함수식의 적항군으로 부터 stuck-open 및 stuck-on고장 검출을 위한 테스트 집합을 간단히 생성할 수 있고 게이트모델에 비해 테스트 생성을 위한 연산의 수가 적어 효율적이며 테스트 패턴의 수가 간소화된 논리함수의 적항의 수 이하로 결정되어 테스트 길이가 비교적 짧아 ASIC 등을 위한 설계 자동화에 사용되어질 때 설계비용의 절감을 가능하게 한다. 또한 스위치모델에 대한 테스트 집합의 특성으로 인하여 다중고장의 검출이 용이한 장점도 갖는다. 제안한 방법은 Domino CMOS 회로뿐만 아니라 일반 스태틱 CMOS에도 응용 가능하다. CMOS회로의 경우 NMOS 부분의 stuck-open 고장은 PMOS 부분의 stuck-on 고장을 위한 테스트 패턴으로 동시에 검출 가능하며 이것의 역도 역시 성립한다. 따라서 제안한 방법에 의해 CMOS 회로의 고장검출을 위한 테스트 집합을 생성하기 위해서는 NMOS부분에 대한 테스트 집합을 생성하는 것으로 충분하다. 또한 Domino CMOS PLA에도 적용 가능한데 이 경우 다출력 함수의 논리최소화를 통해 논리함수를 구성하는 각 적항간에 리던던시가 없도록 하면 최소화 결과로부터 테스트 집합을 간단히 얻을 수 있다. 특히 NMOS 함수블럭내 트랜지스터들의 연결형태는 단순직렬만 존재하므로 fd′는 논리최소기를 거치지 않고 fd의 결과로부터 각 변수를 하나씩 보수를 취하므로써 구할 수 있다. 제안한 방법은 Domino CMOS를 비롯한 MOS 계열의 VLSI테스팅 방식으로 유용하게 사용될 수 있을 것으로 생각된다.

參 考 文 獻

[1] B.T. Murphy, et. al., "Twin tubs, Domino logic, CAD speed up 32-bit processor," *Electronics*, pp. 106-111, Oct. 1981.

[2] D.S. Ha and S.M. Reddy, "On the design of testable Domino PLAs," *IEEE Int. Test Conf.* pp. 567-573, 1985.

[3] J. Galiay, Y. Crouzet and M. Vergniault, "Physical versus logical fault models for MOS LSI circuit: Impact on testability," *IEEE Trans. Comput.* vol. C-29, pp. 527-531, 1980.

[4] P. Banerjee and J.A. Abraham, "Generating tests for physical failures in MOS logic circuits," *IEEE Test Conf.*, pp. 554-559, 1983.

[5] P.L. Flake, P.R. Moorby and G. Musgrave, "Logic simulation of tri-state gates," *Int. Conf. Circuit and Computer* pp. 593-600, 1980.

[6] R. Chandramouli, "On testing stuck-open faults," *The 1983 Int. Symp. on FTC.*, pp. 258-265, June, 1983.

[7] R.I. Damper and N. Burgess, "MOS test pattern generation using path algebras." *IEEE Trans. Comput.*, vol. C-36, pp. 1123-1128, 1987.

[8] R.K. Brayton et al., Logic minimization algorithms for VLSI synthesis, Kluwer Academic Publishers, 1986.

[9] S.K. Jain and V.D. Agrawal, "Modeling and test generation algorithms for MOS circuits," *IEEE Trans. Comput.*, vol. C-34, no. 5, pp. 426-533, May 1985.

[10] V.G. Oklobdzija and P.G. Kovijanic, "On testability of CMOS Domino logic," *IEEE the 14th Int. Conf. on FTC* pp. 50-55, 1984.

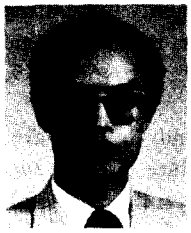
[11] W. Maly, "Realistic fault modeling for VLSI testing," *24ACM/IEEE Design Autom. Conf.* pp. 173-180, 1987.

[12] Y.M. Elziq and S.Y. H. Su, "Fault diagnosis of MOS combinational network," *IEEE Trans. Comput.* vol. C-31 pp. 129-139, 1982.

[13] 이재민, 임인칠, "PLA 논리최소화를 위한 휴리스틱 알고리듬-PLA 논리최소화 프로그램 PLAMIN-," 대한전자공학회 논문지 pp. 63-68 1986년 5월.

[14] 정성무, 선선구, 이재민, 임인칠, "PLA의 고밀도 설계를 위한 논리최소화 알고리듬-PLA 논리최소화 프로그램 PLAMIN II-," 대한전자공학회 반도체, 재료, 씨에이디 및 부품연구회 합동학술 발표회 논문집 pp. 191-195, 1988년 5월

著 者 紹 介



李 在 旼(正會員)

1956年 1月 6日生. 1979年 한양대 전자공학과 졸업. 1981年 한양대 대학원 전자과 졸업. 1987年 2月 한양대 대학원 전자과 박사과정 졸업(공학박사). 1986年 ~ 현재 관동대학교 전자과 조교수 1990年 8月~1991年 8月 University of Illinois at Urbana-Champaign Visiting Professor(예정). 주관심분야는 VLSI CAD(테스팅, PLA Synthesis, Fault Simulation), Multi-valued Logic 등임.

●

李 俊 模(正會員)

1949年 3月 28日生. 1979年 명지대 대학원 전자과 졸업(공학석사). 1988年 3月~현재 단국대학교 대학원 전자과 박사과정. 1977年 3月~1980年 2月 안양공업전문대학 전자과 교수. 1980年 3月~현재 관동대학교 전자과 부교수. 주관심분야는 멀티프로세서 디자인, 컴퓨터 설계 및 제어 등임.

鄭 峻 模(正會員)

1962年 6 月 28日生. 1985年 2月 한양대학교 전자공학과 졸업. 1987年 2月 한양대학교 대학원 전자공학과 석사학위 취득. 1987年 3月~현재 한양대학교 대학원 전자공학과 박사과정 재학중. 주관심분야는 VLSI Logic Testing, Testable Design 및 Simulation.