전류 테스팅을 위한 객체 기반의 무해고장 검출 기법

정회원 배 성 환*, 김 관 웅**, 전 병 실***

An Object-Oriented Redundant Fault Detection Scheme for Efficient Current Testing

Sung-Hwan Bae*, Kwan-Woong Kim**, Byoung-sil Chon*** *Regular Members*

요 약

전류 테스팅은 전류 테스팅은 CMOS 회로의 합선고장을 효과적으로 검출할 수 있는 기법이다. 그러나 합선고장의 복잡도가 $O(n^2)$이고, 또한 전류 테스트 방식이 전압 테스트 방식에 비해서 상대적으로 긴 테스트 시간이 필요하기 때문에 두 합선된 노드가 항상 같은 값을 가지는 노드를 찾아내어 제거하는 효율적인 무해고장 검출기법이 필요하다. 이러한 무해고장은 보다 정확한 고장 검출율을 위해서 ATPG 툴을 이용하여 검출될 수 있어야 한다. 본 논문에서는 효율적인 전류 테스트를 위한 객체 기반의 무해고장 검출기법을 제안한다. ISCAS 벤치마크 회로에 대한 실험을 통해서 제안된 기법이 기존의 다른 방식보다 더 효과적임을 보여주었다.

ABSTRACT

Current testing(Iddq testing) on monitoring the quiescent power supply current is an efficient and effective method for CMOS bridging faults. The applicability of this technique, however, requires careful examination. Since cardinality of bridging fault is $O(n^2)$ and current testing requires much longer testing time than voltage testing, it is important to note that a bridging fault is untestable if the two bridged nodes have the same logic values at all times. Such faults should be identified by a good ATPG tool; otherwise, the fault coverage can become skewed. In this paper, we present an object-oriented redundant fault detection scheme for efficient current testing. Experimental results for ISCAS benchmark circuits show that the improved method is more effective than the previous ones.

I. 서 론

VLSI의 집적도가 증가함에 따라 고장을 회로의 한 라인에 0 또는 1의 논리 값이 고정된 것으로 가정하여 테스트하는 전압 측정에 기반을 둔 논리 테스트 또는 전압 테스팅 방법으로는 검출하기 어려운 많은 고장들이 발생하고 있다[1]. 최근 회로내의 고장 유무에 따라 크게 변하는 평형상태의 전류 값(Iddq : quiescent current)을 비교하여 CMOS VLSI 회로에서 발생 가능한 다양한 형태의 고장을 용이하게 검출할 수 있는 테스트 방식으로 전류 테스팅 방식이 소개되었다[2-3].

전류 테스팅은 정적상태(steady state)에서 합선결함, 게이트 옥사이드 단락, 기생 트랜지스터 누설, 누설 PN 결합, 개방 결합 등과 같은 물리적 결함이나 고장으로 인하여 CMOS 회로의 VDD와 GND 사이에 형성된 전류 경로에 흐르게 되는 큰 정적상태 전류 값을 통해서 고장을 용이하게 검출 할 수 있다. 특히 CMOS VLSI 회로에서 발생 빈도가 높은 합선고장(bridging fault)을 효과적으로 검출할 수 있다[2-4].

테스트 대상회로에 합선고장이 발생한 경우 전류

* 한려대학교 멀티미디어정보통신공학과 (shbae@hlu.hanlyo.ac.kr),
** 전북대학교 전자공학과 (kwkim@cslab.chonbuk.ac.kr),
*** 전북대학교 전자정보공학부 (bschon@moak.chonbuk.ac.kr)
논문번호 : K01159-0718, 접수일자 : 2001년 7월 18일

테스팅을 이용하여 고장을 검출하기 위해서는 합선이 발생한 두 노드가 "1"과 "0" 혹은 "0"과 "1"의 논리 값을 가지게 하여 VDD와 GND 사이의 전류 경로를 형성시켜 고장 검출을 할 수 있다.

그러나 전류 테스팅 기법은 정적상태에서 테스트 전류의 측정이 가능하기 때문에 전압 테스트 방식에 비해서 상대적으로 긴 테스트 시간을 요구한다. 또한 합선고장은 테스트 대상 회로의 노드수가 n인 경우, 고려해야 할 합선고장의 수는 $O(n^2)$의 복잡도를 가지게 되어 회로의 집적도가 증가함에 따라 고려할 고장의 수가 매우 크게 된다. 따라서 전류 테스트 방식을 이용하여 합선고장을 효율적으로 검출하기 위해서는 높은 고장 검출률을 유지하면서 가능한 적은 수의 테스트 패턴을 얻어야 하고, 동시에 두 합선된 노드가 항상 같은 논리 값을 갖는 등가 노드를 찾아내어 제거하는 무해고장 검출 기법이 필요하다.

합선고장 수를 줄이기 위해서 Jee와 Ferguson은 IFA(Inductive Fault Analysis) 방식을 사용하여 CMOS 회로의 결함을 해석하였다[5]. IFA 방식은 회로의 고장을 레이아웃 레벨에서 회로 레벨로 변환시키는 추출과정이 필요하므로 테스트 회로의 레이아웃에 관한 자세한 정보가 요구된다. 그러나 회로의 레이아웃 정보는 항상 유용하지 않으며, 정보가 유용해도 집적도가 큰 회로의 경우에는 레이아웃 정보를 해석하여 고장을 추출하는 과정은 많은 시간과 메모리의 사용이 필요하다[6-8]. 따라서 레이아웃에 관한 정보에 의존하지 않고, 일반적으로 적용 가능한 합선고장 모델링 기법이 필요하다.

본 논문에서는 테스트 대상 회로의 레이아웃 정보에 의존하지 않고 일반적으로 적용 가능한 합선고장 모델[7-12]에서 발생 가능한 단락을 고려하여 두 노드가 항상 같은 값을 가지는 노드를 찾아내어 제거하는 전류 테스트를 위한 객체 기반의 무해고장 검출 기법을 제안한다. 본 논문은 2장에서 고장 모델과 무해고장 검출 기법에 관해서 논의하고, 3장에서는 객체 기반의 무해고장 검출 기법을 제안한다. 또한 4장에서는 제안된 무해고장 검출 기법을 ISCAS 벤치마크 회로에 대한 모의 실험결과를 검토하고, 마지막으로 5장에서 결론을 맺는다.

Ⅱ. 고장 모델과 무해고장 검출 기법

1. 고장 모델

합선은 CMOS VLSI 회로에서 가장 발생 빈도가 높은 고장이며, 회로의 두 노드 사이에 하나의 단락이 발생할 경우에 존재한다. 이때 단락은 제조공정 중에 먼지나 혹은 추가적인 물질이 놓인 경우에 흔히 발생하게 된다. 또한 불완전한 마스크나 에칭과정 등에서도 자주 발생한다[7]. 단락은 결함이 발생되는 위치에 따라 다음과 같은 모델링이 가능하다.

1) 외부 합선고장 : 논리 게이트간에 발생 가능한 단락. 즉 입력과 출력노드 사이의 단락[8, 10-11]
2) 전체 합선고장 : 논리 게이트들의 내부 및 외부의 모든 경우에서 발생 가능한 단락[7, 9, 12]

테스트 회로에 합선고장이 발생한 경우 전류 테스팅을 이용하여 고장을 검출하기 위해서는 합선이 발생한 두 노드가 "1"과 "0"의 논리 값을 갖도록 (혹은 그 반대로) 해 준다면 VDD와 GND 사이의 전류 경로가 형성되어 높은 IDD가 흐르게 됨으로서 가정된 모델의 고장을 검출 할 수 있다. 그림 1에는 외부노드인 2-입력 NOR 게이트 출력단 J와 2-입력 NAND 게이트의 내부노드 D 사이에 발생한 합선고장의 예를 보인다. 그림 1의 회로에 테스트 벡터 T(A, B, C) = "101"을 입력 단에 인가하면, 테스트 대상 회로의 단락이 발생한 두 노드 {D, J}에서 T(D) = "1", T(J) = "0"의 값을 가지게 되어 VDD와 GND 사이에 전류 경로가 형성되어 고장을 검출할 수 있다.

특히, 전체 합선고장 모델은 테스트 회로의 내부와 외부 노드의 모든 경우에서 발생 가능한 단락을 동시에 고려해야 하기에 테스트 대상 회로를 게이트 레벨이 아닌 스위치 레벨로 해석해야 한다. 발생 가능한 합선고장의 수가 외부노드 사이에서 발생하는 합선고장 모델에 비하여 매우 크기 때문에, 모든 입력 테스트 패턴에 항상 같은 값을 가지는 등가 노드를 효과적으로 찾아내어 제거하는 무해고장 검출 기법이 필요하다.

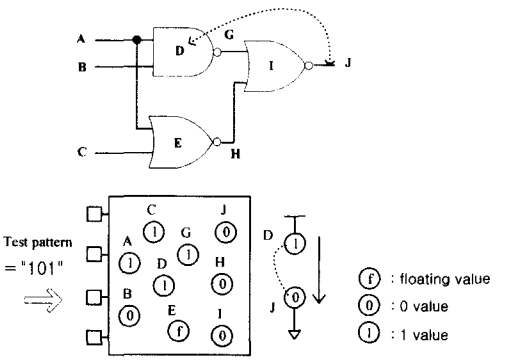


그림 1. 내부와 외부노드 사이에 발생한 합선고장의 예

2. 무해고장의 검출 기법

두 개의 합선된 노드가 임의의 입력 테스트 패턴에 대해서 항상 같은 논리 값을 갖는 경우, 단락된 두 노드에 "1"과 "0" 값의 적용이 불가능하게 된다. 이러한 합선고장은 전류 테스팅에서 모든 테스트 패턴으로도 검출이 불가능하여 무해고장으로 부르며, 무해고장을 효과적으로 검출하는 기법은 전류 테스팅으로 검출하지 못하는 노드를 삭제함으로서 합선고장 모델에서 테스트 패턴 생성에 요구되는 시간과 생성된 테스트 패턴의 수를 줄여준다. 기존의 논문에서는 무해고장을 검출하기 위해서 그림 2와 같은 경우를 고려하였다[7-8, 10-12].

만약 1), 2), 3)의 경우에서 노드 a와 b사이에 단락이 발생하면 합선고장의 검출이 불가능하게 된다. 예를 들어 그림 3(a)은 경우 1과 2의 예로서 회로에는 총 14개의 노드가 존재하여 노드간에 $_{14}C_2$ = 91개의 합선고장이 존재한다. DFS(Depth First Search)나 BFS(Breadth First Search) 기술을 이용하여 최대의 연결구조를 찾으면 모든 테스트 입력 패턴에 항상 같은 값을 가지게 되는 두 개의 등가 그룹 Group1 = {G2, G5, G8, G9}와 Group2 = {G3, G6}로 분해되어 고장이 가능한 노드는 10개로 감소하게 되어 총 $_{10}C_2$ = 45개의 합선고장이 존재하게 된다. 또한 그림 3(b)는 경우 3의 예로서 회로에는 총 9개의 노드가 존재하여 노드간에 $_9C_2$ = 36개의 합선고장이 존재한다. NAND 게이트 G2와 G3는 같은 입력 노드 G0와 G1을 가지게 되어 두 개의 등가그룹 Group1 = {G2, G3}와 Group2 = {a, b}로 분해되어 고장이 가능한 노드는 2개가 감소하게 된다. 가능한 합선고장의 수는 $_7C_2$ = 21이 된다.

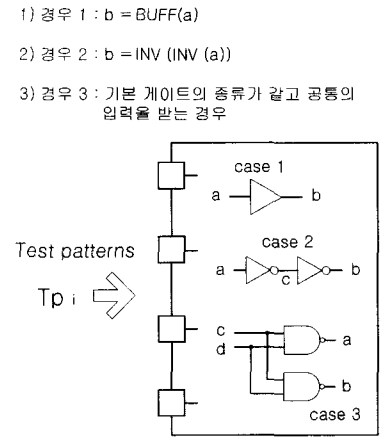


그림 2. 무해 고장의 검출 기법

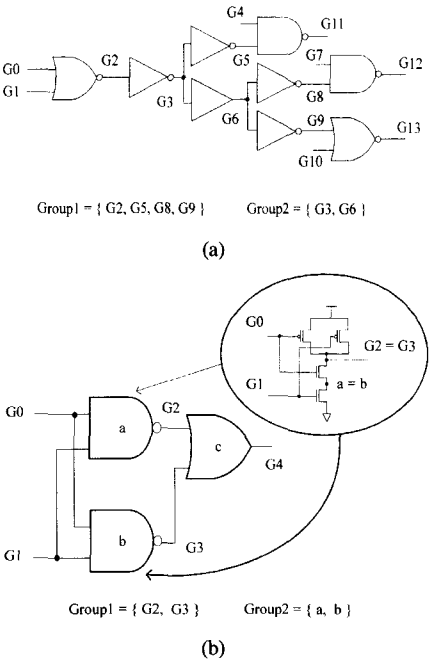


그림 3. 등가 노드의 검출의 예

Ⅲ. 제안된 무해고장 검출 기법

제안한 객체 기반의 무해고장 검출 알고리즘은 논리 게이트간에 발생 가능한 단락, 즉 입력과 출력노드 사이의 단락을 고려하는 외부 합선고장 모델이나, 논리 게이트들의 내부 및 외부의 모든 경우에서 발생 가능한 단락을 모두 찾는 전체 합선고장 모델에 적용할 수 있으며, 기존의 무해고장 검출 방법[7]을 확장하여 기존 알고리즘이 검출할 수 없는 상수 값을 가지는 등가 노드와 게이트의 치환과 분해 과정을 통해서 더 많은 등가 노드를 검출할 수 있다.

전체적인 알고리즘의 흐름은 그림 4에 보인 바와 같이 4종류의 collapsing-프로세스로 이루어져 있으며, 구현된 프로세서는 Window NT상에서 C++ 언어를 이용하여 구현 하였다. CollapsingBufferAndInverter와 CollapsingWithSameType 프로세스는 각각 기존 알고리즘의 경우 1, 2와 경우 3에 관련된 등가 노드를 검출하는 프로세싱 과정이다. 또한 나머지 두 가지 중 CollapsingByConvertedGate 프로세스는 경우 3을 확대 적용하여 등가 노드를 찾아내는 과정이며, 마지막 SelfCollapsing 프로세스는 게이트의 입력단자로 collapsing을 수행하여 상수 노드의 검출과 게이트의 치환과정을 수행한다.

특히, 전체 합선고장 모델의 경우에는 테스트 대

상 회로의 모든 노드(내부, 외부노드)를 고려해야 하기 때문에 스위치 레벨의 분석이 필요하므로 본 논문에서는 전처리 과정으로서 버퍼, AND, OR 게이트의 경우는 NAND, NOR, 인버터 등의 게이트로 분해하는 방식을 사용한다. 실제로 버퍼, AND, OR 게이트는 분해가 불가능하지만, 그림 5의 예와 같이 AND 게이트는 NAND와 인버터의 구조로 분석 할 수 있다. 기존의 전체 합선고장 모델의 무해고장 검출 알고리즘에서는 AND 게이트 G1과 인버터 G2는 서로 값이 달라 등가노드를 검출할 수 없지만, AND 게이트를 NAND 게이트와 인버터로 분해했을 경우에는 AND 게이트의 내부노드 G0과 인버터 G2는 등가노드가 되어 검출이 가능해진다.

1. CollapsingBufferAndInverter 프로세스

버퍼와 인버터의 경우에는 입력이 하나이기 때문에 경로를 탐색하여 항상 같은 값을 가지게 되는 노드를 collapsing한다. Collapsing을 수행하기 전 버퍼와 인버터의 경우, 게이트의 출력 값에 이전 게이트의 ID가 저장된다. 버퍼는 이전 게이트의 ID를 그대로 저장하나, 인버터의 경우는 ID에 음수를 곱해 저장한다

Collapsing 수행 시 먼저 경로 탐색 알고리즘을 적용하여 최대 연결구조를 찾은 후에 경로를 따라 같은 값을 가지는 게이트들끼리 collapsing을 수행한다. 예를 들어 그림 6과 같은 연결 구조에서 G2, G3, G5, G6은 (-G1)의 값을 가지고 G4는 (G1)의 값을 가지게 되어 G1부터 시작하여 출력단 쪽으로 경로를 찾아가면서 (-G1)값을 가지는 Group1과 (G1) 값을 가지는 Group2로 collapsing이 이루어진다.

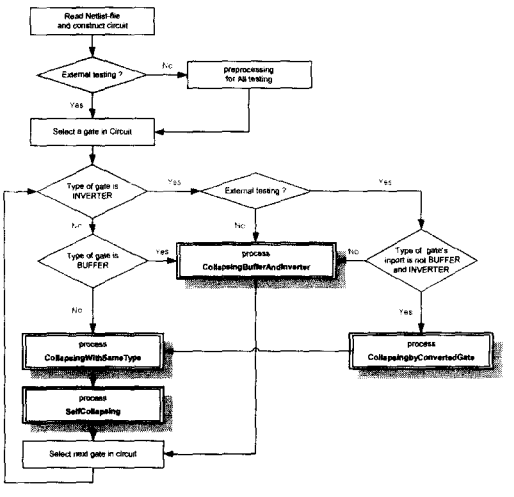


그림 4. 제안된 무해고장 검출 알고리즘의 흐름도

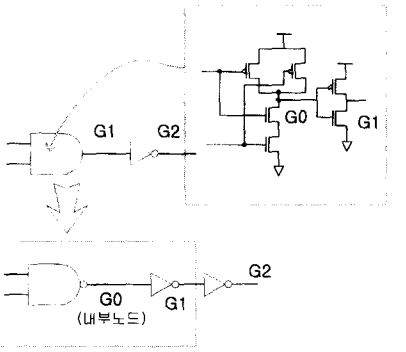


그림 5. 전체 합선고장 모델의 전처리 예

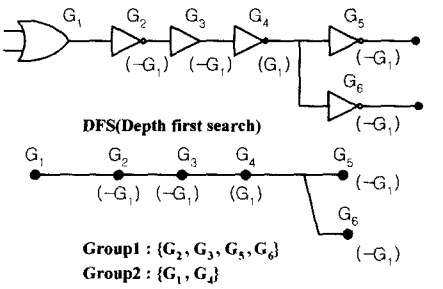


그림 6. CollapsingBufferAndInverter 프로세스에 관련한 예

2. CollapsingWithSameType 프로세스

테스트 대상 회로에서 버퍼와 인버터를 제외한 기본 게이트에 대하여 게이트 종류는 같고 공통의 입력을 받게 될 경우에 collapsing을 수행한다. 또한 여기에 게이트들의 입력이 같은 그룹일 경우에도 collapsing을 수행하도록 확대하여 적용하였다.

3. CollapsingByConvertedGate 프로세스

CollapsingByConvertedGate 프로세스는 게이트가 인버터일 때, 인버터의 입력 게이트가 인버터나 버퍼가 아닐 경우, 인버터의 입력 게이트와 인버터를 입력게이트와 반대 종류인 게이트로 치환하여 collapsing을 수행한다. 인버터의 입력단자의 게이트가 AND이면 인버터와 입력단자의 AND 게이트를 AND의 반대 종류인 NAND로 치환하고, OR이면 NOR, XOR이면 XNOR로 치환하고 CollapsingWithSameType 프로세스를 수행하면, 다른 종류의 게이트들과도 collapsing이 이루어질 수 있다. 예를 들어 그림 7의 경우에는 NOR 게이트 노드 5021과 인버터 노드 5046은 CollapsingByConvertedGate 프로세스 과정 후 CollapsingWithSameType 프로세스 과정을 수행하면 노드 5018과 노드 5046은 입력단자가 같으므로 등가 게이트로 collasping이 수행된다.

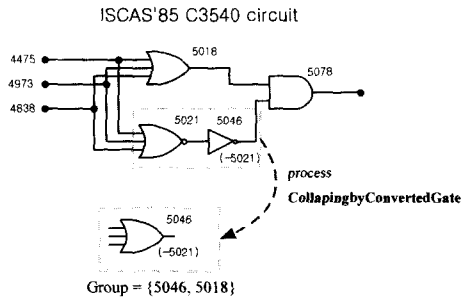


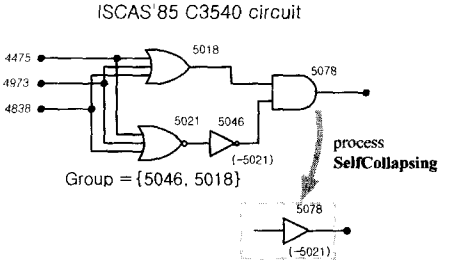
그림 7. CollapsingByConvertedGate 프로세스에 관련한 예

이러한 처리과정에 의사코드는 다음과 같다.

```
process CollapsingByConvertedGate(gate g)
{
  gate con_g = contrary gate of g's input gate
  con_g.ID = g.ID
// con_g의 게이트번호와 g의 게이트 번호를 일치시킨다.
  copy g's input to con_g;
// g의 입력게이트의 입력을 con_g로 복사한다.
  call CollapsingWithSameType(con_g);
// process CollaspingWithSameType
  copy con_g to g;     // con_g를 g로 복사
}
```

4. SelfCollapsing 프로세스

SelfCollapsing 프로세스는 게이트의 입력단자를 검색하여 collapsing을 수행하여, 상수 노드의 검출과 경우에 따라서 게이트를 다른 형태의 게이트로 치환하여 collapsing을 수행한다. 그림 8(a)은 게이트의 형태를 치환하여 collapsing을 하는 예이다. AND게이트 노드 5078의 경우 CollapsingBy-ConvertedGate와 CollapsingWithSameType 프로세스에 의해 두 입력이 등가 관계가 되고 SelfCollapsing 프로세스에 의해서 노드 5078은 노드 5018, 5046과 등가노드가 된다.



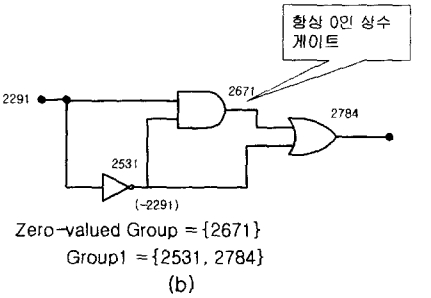


그림 8. SelfCollapsing 프로세스의 예

그림 8(b)은 SelfCollapsing 프로세스에 의한 상수 게이트를 검출하는 예이다. AND 게이트 노드 2671은 입력이 2291과 인버터 노드 2531이다. 인버터 2531의 출력 값이 (-2291)이므로 두 입력이 서로 반대 값을 가진다. 따라서 노드 2671은 항상 0의 값을 가지는 상수 노드가 된다. 다음 단계로 OR 게이트 노드 2784는 노드 2671이 항상 0이고 입력단자가 2개이므로 버퍼로 치환되어 노드 2531과 등가노드가 된다. SelfCollapsing 프로세스에 관련된 의사코드는 아래와 같다. SelfCollapsing 프로세스는 게이트의 입력단자를 검색하여 4가지 경우에 따라 collapsing을 수행하여 게이트 노드의 치환과 상수 노드의 검출이 이루어진다.

```
process SelfCollapsing (gate g)
{
        // 입력단자의 ID가 다른 입력단자의 val와 같고 부호가 다를때
 if (g.input[0].ID == -1 * g.input[1].val){
    switch (type of g)
        case AND, NOR, XNOR: grouping to Zero-valued gates
        case OR, NAND, XOR : grouping to One-valued gates
 }
        // 입력단자의 ID가 다른 입력단자의 val와 같을 때
 else if (g.input[0].ID == g.input[1].val){
    switch (type of g)
        case AND, OR: g convert to BUFFER
            call CollaspingBufferAndInverter(g)
        case NAND, NOR : g convert to INVERTER
            call CollaspingBufferAndInverter(g)
```

```
            case XOR : grouping to Zero-valued
            gates
            case XNOR : grouping to One-valued
            gates
    }
// g의 입력단자중 0을 가지는 상수게이트가 있으면
    else if (Zero-valued gate exist in g's input){
        switch (type of g)
            case AND, NOR : grouping to
            Zero-value gates
            case OR : g convert to BUFFER
                call CollaspingBufferAndInverter(g)
            case NOR : g convert to INVERTER
                call CollaspingBufferAndInverter(g)
    }
// g의 입력단자중 항상 1을 갖는 상수게이트가 있
으면
    else if (One-valued gate exist in g's input){
        switch (type of g)
            case OR, NAND : grouping to
            One-valued gates
            case AND : g convert to BUFFER
                call CollaspingBufferAndInverter(g)
            case NOR : g convert to INVERTER
                call CollaspingBufferAndInverter(g)
            }
}
```

Ⅳ. 모의 실험 결과

제안된 객체기반의 무해고장 검출 알고리즘의 효율성을 검증하기 위해서 ISCAS'85 벤치마크 회로에 적용한 모의실험 결과를 표 1에 보였다.

표 1은 외부 합선고장과 전체 합선고장 모델의 경우에서 각각의 제안된 무해고장 검출 알고리즘을 이용하여 무해고장을 검출하고 남은 합선고장의 수를 나타내며, 표 1의 결과를 통해서 제안된 알고리즘이 더 많은 수의 등가노드를 검출함을 보인다. 더욱이 테스트 대상 회로의 크기가 커질수록 제안된 검출 기법이 더 효과적임을 알 수 있다.

그림 9와 10에는 가능한 총 합선고장 수와 무해고장을 제거하고 남은 합선고장 수와의 비율(%)을 외부 합선고장과 전체 합선고장의 경우에 따라 각각 도시하였다. 표 1과 그림 9, 10의 결과를 통해서 제안된 검출 기법이 전류 테스팅을 이용하는 합선고장 모델에 더 적합함을 확인할 수 있다.

표 1. ISCAS '85 벤치마크 회로에 적용한 무해고장의 결과

무해고장 검출후 외부 합선고장 수					
	(1)	(2)	(3)	(4)	(5)
c432	19,110	18,145	18,145	18,145	18,145
c499	29,403	22,155	22,155	22,155	22,155
c880	97,903	78,210	75,466	75,466	75,466
c1355	171,991	136,503	136,503	136,503	136,503
c1908	416,328	270,480	191,271	270,480	115,440
c2670	1,016,025	746,031	627,760	746,031	440,391
c3540	1,476,621	1,010,331	668,746	823,686	517,653
c5315	3,086,370	2,316,628	1,867,278	2,316,628	1,540,890
c6288	2,995,128	2,956,096	2,956,096	2,956,096	2,842,920
c7552	6,913,621	5,022,865	3,924,201	5,019,696	3,081,403

무해고장 검출후 전체 합선고장 수				
	(6)	(7)	(8)	(9)
c432	108,345	106,030	74,305	75,305
c499	526,851	494,515	271,953	167,331
c880	436,645	-	323,610	374,545
c1355	675,703	603,351	593,505	639,015
c1908	1,540,890	832,695	634,501	459,361
c2670	3,904,615	2,907,666	2,168,403	1,842,240
c3540	6,402,831	4,235,505	3,777,126	3,081,403
c5315	15,100,260	11,681,361	10,490,490	8,994,289
c6288	12,941,328	12,860,056	12,622,800	12,477,510
c7552	27,162,135	19,615,716	17,508,403	13,873,278

(1) 합선고장 수(외부) (2) Reddy[10]
(3) Thadikaran[11] (4) Shinogi[8] (5) 제안한 방법
(6) 합선고장 수(전체) (7) Thadikaran[7]
(8) TerryLee[12] (9) 제안한 방법

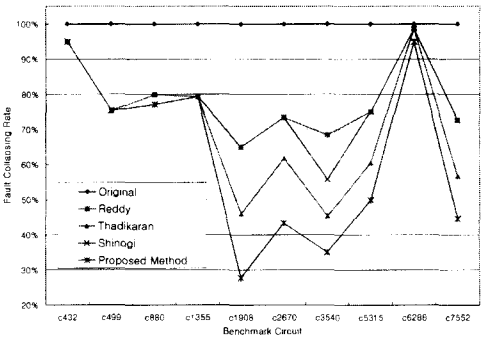


그림 9. 무해고장 검출률의 비교(외부)

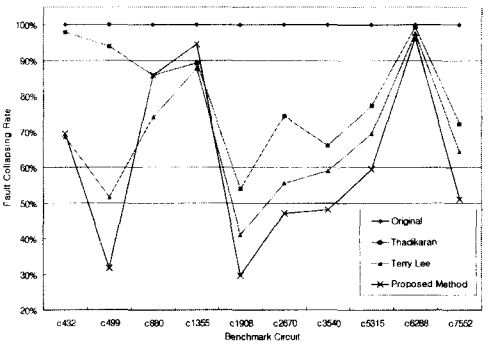


그림 10. 무해고장 검출률의 비교(전체)

V. 결 론

전류 테스팅 방식을 이용하여 CMOS VLSI 회로에서 발생 빈도가 가장 높은 합선고장을 검출할 경우에 테스트 대상 회로의 노드수가 n인 경우, 고려해야 할 합선고장의 수는 $O(n^2)$의 복잡도를 가지게 되어 회로의 집적도가 증가함에 따라 고려할 고장의 수가 매우 크게 된다. 또한 전류 테스트 방식이 전압 테스트 방식에 비해서 상대적으로 긴 테스트 시간이 필요하기 때문에 본 논문에서는 두 합선된 노드가 항상 같은 값을 가지는 노드를 찾아내어 제거하는 효율적인 객체 기반의 무해고장 검출 알고리즘을 제안하였다.

제안된 알고리즘은 테스트 대상 회로의 레이아웃 정보에 의존하지 않고 일반적인 외부 합선고장과 전체 합선고장 모델에 적용 가능한 기법으로 ISCAS 벤치마크 회로를 통한 모의실험을 수행하여 기존의 알고리즘에 비해서 효과적임을 확인하였다 (표 1과 그림 9, 10 참조). 따라서 제안된 알고리즘을 전류 테스팅을 이용하는 합선고장 모델에 적용할 경우 테스트 패턴 생성에 요구되는 시간과 생성되는 테스트 패턴의 수를 효과적으로 줄여준다. 이러한 무해고장 검출 기법은 보다 정확한 고장 검출율을 위해서 ATPG 툴에 이용될 수 있다.

앞으로 제안된 알고리즘을 이용하여 전류 테스트를 위한 패턴 생성 기법에 관련된 연구를 진행시킬 경우, 상대적으로 느린 전류 테스팅의 단점의 보완과 전압 테스트 방식에 비해서 신뢰성 있는 테스트가 가능하리라 생각된다.

참 고 문 헌

[1] J. A. Abraham, "Challenges in fault detection," *International Symposium on Fault-Tolerant Computing,* pp. 96-114, 1995.

[2] R. Rajsuman, *IDDQ Testing for CMOS VLSI,* Artech House, 1994.

[3] 전병실 외, "기능테스트와 IDDQ 테스트를 위한 자체 점검 BIST 회로의 설계," 서울대학교 반도체공동연구소 연구보고서, 1998.

[4] 전병실 외, "합선고장을 위한 IDDQ 테스트 패턴 발생기의 구현," 한국통신학회논문지, Vol. 24, No. 12-A, pp. 2008-2014, 1999.

[5] A. Jee and F. J. Ferguson, "Carafe: An inductive fault analysis tool for CMOS VLSI circuits," *Proc. IEEE Int'l Conf.*, pp. 73-82, 1993.

[6] S. Chakravarty and P. J. Thadikaran, "Simulation and generation of IDDQ tests for bridging faults in combinational circuits," *IEEE Trans. Computers, Vol.* 45, No. 10, pp. 1131-1140, Oct. 1996.

[7] P. J. Thadikaran, "Evaluation, selection and generation of IDDQ tests," PHD. Thesis, Department of Computer Science, State University of New York, 1996.

[8] T. Shinogi and T. Hayashi, "An iterative improvement method for generating compact tests for IDDQ testing of bridging faults," *IEICE Trans. INF & SYST.,* Vol. E81-D. No. 7, July 1998.

[9] 전병실 외, "CMOS VLSI의 효율적인 IDDQ 테스트 생성을 위한 패턴 생성기의 구현" 대한전자공학회논문지, Vol. 38-SD, No. 4, pp. 292-301, 2001.

[10] R. S. Reddy, I. Pomerantz, S. M. Reddy, S. Kajihara, "Compact test generation for bridging faults under IDDQ testing," *IEEE VLSI Test Symposium,* pp. 310-315, 1995.

[11] P. J. Thadikaran and S. Chakravarty, "Fast Algorithm for Computing I_{DDQ} tests for Combinational Circuits," *IEEE International Conference on VLSI Design,* pp. 103-106, 1996.

[12] T. Lee, I. N. Hajj, E. M. Rudnick, J. H. Patel, "Genetic-algorithm based test generation for current testing of bridging faults in CMOS VLSI circuits," *IEEE VLSI Test Symposium*, pp. 456-462, 1996.

배 성 환(Sung-hwan Bae) 정회원
제25권 제12호 참조
현재 한려대학교 멀티미디어정보통신공학과

김 관 웅(Kwan-woong Kim) 정회원
제26권 제9호 참조
현재 전북대학교 대학원 전자공학과 박사과정

전 병 실(Byoung-sil Chon) 정회원
제26권 제9호 참조
현재 전북대학교 공과대학 전자정보공학부 교수