

論文99-36C-10-4

# 임계-쌍 경로를 이용한 시험 불가능 결함의 확인

## (Untestable Faults Identification Using Critical-Pair Path)

徐 聖 煥 \* , 安 光 善 \*\*

(Seong-Hwan Seo and Gwang-Seon Ahn)

## 요 약

본 논문은 조합 논리회로에서의 시험 불가능한 결함(untestable faults)을 확인하는 새로운 알고리즘 RICIP(Redundancy Identification using Critical-pair Paths)를 제시한다. 조합 논리회로에서의 시험 불가능 결함은 회로의 과잉(redundancy)에 의해서 발생한다. 회로의 과잉은 팬 아웃 스템(fanout stem)과 재결집 게이트(reconvergent gate)의 영역을 분석함으로써 찾을 수 있다. 시험 불가능한 결함들은 임계 경로의 확장된 개념인 임계-쌍 경로를 이용하여 스템 영역을 분석함으로써 확인되어진다. RICIP 알고리즘이 FIRE(Fault Independent REDundancy identification) 알고리즘보다 효율적이라는 것을 보여준다. ISCAS85 벤치마크 테스트 회로에 대한 두 알고리즘의 실험 결과를 비교하였다.

## Abstract

This paper presents a new algorithm RICIP(Redundancy Identification using Critical-pair Paths) to identify untestable faults in combinational logic circuits. In a combinational logic circuit, untestable faults occurred by redundancy of circuits. The redundancy of a circuit can be detected by analyzing areas of fanout stem and reconvergent gates. The untestable faults are identified by analyzing stem area using Critical-Pair path which is an extended concept of critical path. It is showed that RICIP is better than FIRE(Fault Independent REDundancy identification) algorithm in efficiency. The performance of both algorithms was compared using ISCAS85 bench mark testing circuits.

Keyword : untestable faults, critical path, stem region, redundancy.

## I. 서 론

논리회로의 규모가 커지면서 설계자들이 범하게 되는 실수 중의 하나가 논리 과잉(logical redundancies)의 추가이다. 이것은 결국 VLSI의 설계 및 테스트 생성 과

정에서의 비용을 증대시킨다. 일반적으로 VLSI의 테스트 생성은 NP-complete problem<sup>[1]</sup>로 알려져 있고, 결함 검출을 위한 테스트 패턴 생성에서 대부분의 시간을 테스트 패턴이 존재하지 않는 시험 불가능 결함(untestable or undetectable fault)의 확인에 소모한다. 순서논리회로에서는 플립플롭이 존재하기 때문에 초기 값의 영향으로 논리 과잉이 아니더라도 시험 불가능한 결함이 있을 수 있지만, 조합논리 회로에서의 시험 불가능 결함은 항상 논리 과잉으로 야기되는 잉여 결함(redundant faults)이다.<sup>[2]</sup> 논리 과잉 회로는 다른 결함의 검출 성능에도 영향을 주게되므로 잉여 결함의 확인과 제거는 논리 합성이나 자동 테스트 생성(Automatic Test Generation : 이하 ATG)에 매우 유

\* 正會員, 東洋大學校 컴퓨터工學部  
(School of Computer Engineering, Dongyang University)

\*\* 正會員, 慶北大學校 컴퓨터工學科  
(Dept. of Computer Engineering, Kyoungpook National University)

接受日字:1999年7月5日, 수정완료일:1999年9月30日

용한 작업이다.

잉여 결함을 확인하는 기술에는 간접적인 기술(indirect technique)과 직접적인 기술(direct technique)이 있는데, 전자는 ATG를 사용하여 테스트 생성하는 과정에서 부산물로 얻게되는 방법이고 후자는 테스트 생성 과정에 포함시키지 않고 별도의 과정을 통해서 직접 논리 과잉을 확인하는 방법이다. 직접적인 기술은 다시 정적인 기술(static technique)과 동적인 기술(dynamic technique)로 나눌 수 있다. 정적인 기술은 ATG의 전처리 과정으로서 논리 과잉을 확인하는 방법이고, 동적인 기술은 ATG를 수행하는 중에 확인하는 방법이다.<sup>[3]</sup>

테스트 생성이나 결함 시뮬레이션을 하는 과정에서 시험 불가능 결함인 줄 모르고, 그 결함에 대한 테스트 패턴을 발견하기 위해서 많은 시간을 소모한다는 것은 시간 낭비이다. 그러므로 일반적으로 회로의 테스트 시에 시험 불가능 결함을 확인해서 대상 결함에서 제외시키는 것이 효율적이다. 일반적으로 테스트 생성의 전처리 과정으로 시험 불가능 결함의 확인 작업을 수행한다. 그래서 시험 불가능 결함의 확인은 대부분 직접적인 기술의 형태를 취한다.

조합 논리회로에서 주 출력 단으로부터 어떤 위치까지의 임계-쌍 경로가 존재한다는 것은 그 경로 상에서는 시험 불가능 결함이 없다는 것을 의미한다<sup>[4]</sup>. 그러므로 본 논문에서는 시험 불가능 영역에 대하여 임계-쌍 경로의 여부를 확인함으로써 시험 불가능 결함을 발견하는 새로운 방법을 제시한다.

본 논문의 구성은 다음과 같다. 제 2 절에서는 기존의 시험 불가능 결함의 확인 방법들을 소개하고, 제 3 절에서는 임계-쌍 경로를 이용한 시험 불가능 결함의 확인 방법을 소개하며, 제 4 절에서는 ISCAS85의 회로를 이용한 실험을 통해서 기존 방법과의 성능 비교를 한 후, 제 5 절에서 결론과 추후 연구과제를 기술한다.

## II. 기존의 시험 불가능 결함의 확인 방법들에 대한 소개 및 평가

직접적으로 시험 불가능 결함을 확인하는 방법으로 다양한 기술들이 연구되어 왔다. 고장 재결집 팬 아웃 구조의 분석에 기초를 둔 기법<sup>[5, 6]</sup>, 제어-관측성(controllability / observability)의 분석에 기초를 둔 기

법<sup>[7, 8]</sup>, 회로의 구조 변형에 기초를 둔 기법<sup>[9]</sup> 등이 정적인 기술에 속한다고 할 수 있다. 동적인 기술로는 테스트 생성기가 대상 결함을 위한 테스트 생성에 실패한 후에 부가적으로 잉여 결함을 확인하게 하는 방법이나 테스트 생성과 결함 시뮬레이션 중에 미발견 결함을 발견하면 그 잉여 부분을 제거함으로써 한 번(one-pass)만에 잉여 결함을 발견하고 제거하는 방법<sup>[3]</sup> 등이 있다.

최근에는 직접 영향 분석(direct implication) 방법으로 알 수 없는 필요 할당(necessary assignment)을 발견하기 위한 방법으로 간접 영향 분석(indirect implication)의 방법을 사용한 재귀적 학습(recursive learning) 방법을 사용하기도 한다.<sup>[10, 11]</sup>

시험 불가능 결함의 확인 방법 중에서 최근 가장 활발히 연구되고 있는 방법 중의 하나인 FIRE(Fault Independent REdundancy identification) 알고리즘<sup>[2, 12]</sup>을 소개하고 본 논문에서 제안하는 방법과 비교하고자 한다.

이 알고리즘은 제어 불가능성(uncontrollability)과 관측 불가능성(unobservability)의 분석에 기초를 둔 방법이다. 제어 불가능성은 순방향(forward)으로 전파시키고, 관측 불가능성은 역방향(backward)으로 전파시킴으로서 시험 불가능 결함을 확인한다. 회로선 상의 값 0(1)에 대한 제어 불가능 상태를  $\bar{0}$ ( $\bar{1}$ )로 나타낸다. 제어 불가능 상태의 전파에 대한 예를 그림 1에 나타내었다.

회로 상에서 순방향으로 제어 불가능성을 전파시킬 수 있다면 그 회로 선은 관측이 불가능하게 된다. 만일 어떤 게이트의 입력이 제어불가능 상태 값을 설정할 수가 없다면 다른 모든 입력들은 관측 불가능 상태가 된다(그림 1에서 "\*"로 나타내었다).

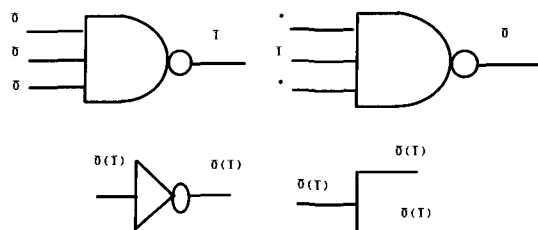


그림 1. 제어 불가능성의 전파  
Fig. 1. Propagation of uncontrollability.

제어불가능 상태  $\bar{0}(\bar{1})$ 인 회로 선에는 s-a-1(s-a-0)을 발생시킬 수가 없고 전파시킬 수도 없다. 비정상적인 결합에 의한 잉여 결합을 발견하기 위한 일반적인 절차는 다음과 같다.

만일  $l_1 = v_1, l_2 = v_2, \dots$ 가 비정상적인 결합의 회로 선과 그 값이라면

1. 모든  $l_i = v_i$ 에 대하여 제어 불가능한 회로 선과 관측 불가능한 회로 선을 모두 결정하기 위해  $l_i = \bar{v}_i$ 를 인가한다. 그 값에 대응하는 결합의 집합을  $S_i$ 라고 한다.
2. 주어진 비정상적인 결합에 의해 야기되는 잉여 결합은  $S_i$ 에 대한 공통 결합( $\cap S_i$ )으로 구한다.

이 FIRE 알고리즘은 팬 아웃과 그 팬 아웃의 FOB(FanOut Branches)들이 재결집 되는 게이트들에 의해서 결정되는 영역을 잉여 지역(Redundant Region) 정한다. 그리고 잉여 지역의 팬 아웃과 재결집 게이트의 입력 선들에 대하여 위의 과정을 적용함으로써 시험 불가능한 결합을 찾는다<sup>[12]</sup>.

FIRE에 대한 개략적인 알고리즘은 그림 2에 나타나 있다.

```

FIRE() {
    Form a list L of all stems and reconvergent inputs of reconvergent gates in the circuit.
    For every line l in L {
        Imply  $l = \bar{0}$  to determine all lines becoming uncontrollable or unobservable.
        Let  $S_0$  be the set of the corresponding faults.
        Imply  $l = \bar{1}$  to determine all lines becoming uncontrollable or unobservable.
        Let  $S_l$  be the set of the corresponding faults.
        The redundant fault are in the set  $S = S_0 \cap S_l$ .
    }
}
    
```

그림 2. FIRE 알고리즘의 개략  
Fig. 2. Outline of FIRE algorithm.

### III. 임계-쌍 경로를 이용한 시험 불가능 결합의 확인 방법

FIRE 알고리즘의 단점은 팬 아웃 스템에서의 제어 불

가능성이 게이트 특성 때문에 재결집 게이트까지 전파 되지 않는 경우가 많다는 사실이다. 그러므로 재결집에 의한 제어 불가능성을 확인하기 위하여 모든 재결집 게이트의 모든 입력에 대하여서도 제어 불가능성을 검토해야 한다. 이런 단점을 극복하기 위하여 임계-쌍 경로를 이용한 알고리즘을 제안한다.

어떤 회로 선으로부터 다른 회로 선 까지 임계-쌍 경로가 존재한다면 그 두 회로 선 사이의 경로 상의 모든 결합은 검출 가능하다<sup>[4]</sup>. 그러므로 임계-쌍 경로를 잉여 지역에 적용하면 팬 아웃 스템으로부터 재결집 게이트 사이에 결합이 존재하는 지를 확인 할 수 있다.

먼저 본 논문에서 제안하는 알고리즘의 설명을 위해서 사용하고 있는 용어들에 대한 개념을 설명한다. 먼저 어떤 회로의 주출력단(Primary Output : 이하 PO)에 임계-쌍 값  $\nu$ 를 설정하기 위한 어떤 경로선의 한 쌍의 값(value-pair)의 유형은 다음과 같다.

- 1)  $\omega : 1/0$ . 즉, PO의 값을 1로 만드는 경로선의 값이 1이고, PO의 값을 0으로 만드는 경로선의 값이 0인 경우의 한 쌍의 값.
- 2)  $\omega' : 0/1$ . 즉, PO의 값을 1로 만드는 경로선의 값이 0이고, PO의 값을 0으로 만드는 경로선의 값이 1인 경우의 한 쌍의 값.  $\omega$ 의 보수값.
- 3)  $1(\nu) : 1/1$ . 즉, PO의 값을 1로 만드는 경로선의 값과 PO의 값을 0으로 만드는 경로선의 값이 모두 1인 경우의 한 쌍의 값.
- 4)  $0(\nu') : 0/0$ . 즉, PO의 값을 1로 만드는 경로선의 값과 PO의 값을 0으로 만드는 경로선의 값이 모두 0인 경우의 한 쌍의 값.  $\nu$ 의 보수값.

게이트를 노드로 표현함으로써 논리회로를 그래프 형태로 변형시킨 후 정의한 임계-쌍 경로의 정의를 살펴보면 다음과 같다<sup>[4]</sup>.

정의 1 : 어떤 한 쌍의 테스트  $\tau(l \text{ and } l')$ 를 적용한 그래프에서 만일 결합 쌍(1/0 and 1/1)을 발견할 수 있는 값  $\nu(\omega[1/0]$  혹은  $\omega'[0/1])$ 을 갖는 경로선 l을 테스트 쌍  $\tau$ 에서의 임계-쌍이라고 한다. 이때 경로선 l은 임계-쌍 값  $\nu$ 를 갖는다고 말하고, 이때의 테스트 패턴의 쌍  $\tau$ 을 임계-쌍 테스트 패턴(critical-pair test patterns)라고 한다.

정의 1로부터 유도된 다음의 정리는 명백하다.

정리 1. 어떤 한 쌍의 테스트  $\tau(l \text{ and } l')$ 를 적용한 그래프 상에 있는 두 지점 l1과 l2사이 임계-쌍 경로가

존재한다면 임계-쌍 경로 상의 모든 결함은 발견할 수가 있다. 단,  $I_1$ 은  $I_2$ 의 선행자(predecessor)이다.

증명) 정의 1로부터 명백히 알 수가 있는데, 만일 두 지점  $I_1$   $I_2$  사이 발견할 수 없는 결함이 존재한다면 이 두 지점 사이에는 임계-쌍 경로가 존재하지 않게 된다. 그러므로  $I_1$ 과  $I_2$  사이의 모든 결함은 발견할 수가 있다.

임계-쌍 경로를 이용하여 생성된 테스트  $\tau$ 는 PO(Primary Output)에서의  $w(0/1)$ 를 설정한 경우나  $w'(0/1)$ 를 설정한 경우가 동일한 결과를 얻게 된다. 그리고 임계-쌍 경로가 존재하면 임계-쌍 경로의 주 출력(PO) 쪽에서 역방향으로 추적하든지 주 입력(PI) 쪽에서 순방향으로 추적하든지 동일한 임계-쌍 경로와 테스트 패턴을 얻을 수 있다.

예를 들어 그림 3의 회로 (1)에서 회로 선 A에서부터 임계-쌍 경로를 설정하면  $A = w(1/0)$ 이면  $E = w(1/0)$ 이 되고, A가 감지선이 되기 위해서는 다른 입력(B)은  $v(1/1)$ 로 될 수밖에 없다. 동일한 방법으로  $Y = w(1/0)$ ,  $F = v(1/1)$ 이 된다. C와 D 값은 역추적 방법과 동일한 방법으로 모두  $v(1/1)$ 이 된다는 것을 알 수가 있다.

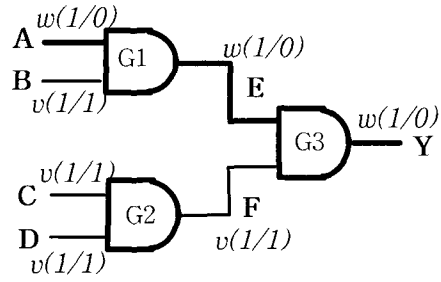


그림 3. 회로(1)  
Fig. 3. Circuit (1).

그러므로 다음의 정리가 가능하다.

정리 2. 어떤 회로에서 어떤 PI(Primary Input)인  $I_1$ 과 어떤 PO인  $I_2$ 사이 임계-쌍 경로가 존재한다면, PO인  $I_2$ 에서 역추적 의한 정당화 방법으로 임계-쌍 경로를 구할 수 있을 뿐만 아니라 PI인  $I_1$ 에서의 감지 경로 설정으로 동일한 임계-쌍의 테스트를 구할 수 있다.

증명) PI에  $w(1/0)$  혹은  $w'(0/1)$ 를 설정한 다음 게이트를 만날 때마다 임계-쌍 경로 선이 감지 선(sensitive line)이 되도록 게이트의 다른 입력을 설정한다. XOR(XNOR) 게이트를 제외한 모든 게이트는 임계

표 1. AND, OR, NAND, NOR 및 XOR 게이트에 대한 임계-쌍 값의 전파  
Table 1. Propagation of critical-pair values for AND, OR, NAND, NOR and XOR gates.

AND(NAND)				NOTE	OR(NOR)				NOTE
INPUT		OUTPUT	Z		INPUT		OUTPUT		
SL	SI				SL	SI			
<b>*w*</b>	*v-	<b>*w*(*w'*)</b>	· SI의 값은 항상 v이고, 모든 입력 값이 1인 경우만 critical	<b>*w*</b>	-v'	<b>*w*(*w'*)</b>	· SI의 값은 항상 v'이고, 모든 입력 값이 0인 경우만 critical		
<b>*v*</b>	*v*	<b>*v*(*v'*)</b>		<b>*v*</b>	-v'-	<b>*v*(*v'*)</b>			
<b>*w'*</b>	-v*	<b>*w'*(w*)</b>		<b>*w'*</b>	*v'-	<b>*w'*(w*)</b>			
<b>*v'*</b>	-v-	<b>*v'*(v*)</b>		<b>*v'*</b>	*v'*	<b>*v'*(v*)</b>			

XOR(inputs 수 : odd)				NOTE	XOR(inputs 수 : even)				NOTE
INPUT			OUTPUT		INPUT			OUTPUT	
SL	SL_1	SL_X	Z	SL	SL_1	SL_X	Z		
<b>*w*</b>	<b>*w*</b>	<b>*w*</b>	<b>*w*</b>	· 다양한 설정 방법이 존재하지만 이 규칙을 적용함.	<b>*w*</b>	<b>*v'*</b>	<b>*w*</b>	<b>*w*</b>	· 다양한 설정 방법이 존재하지만 이 규칙을 적용함.
<b>*v*</b>	<b>*v*</b>	<b>*w*</b>	<b>*w*</b>		<b>*v*</b>	<b>*w*</b>	<b>*w*</b>	<b>*w'*</b>	
<b>*w'*</b>	<b>*w'*</b>	<b>*w'*</b>	<b>*w'*</b>		<b>*w'*</b>	<b>*v'*</b>	<b>*w'*</b>	<b>*w'*</b>	
<b>*v'*</b>	<b>*v'*</b>	<b>*w'*</b>	<b>*w'*</b>		<b>*v'*</b>	<b>*w*</b>	<b>*w*</b>	<b>*w*</b>	

\* SL=Stem Lines, SI=Stem Inputs, SL\_1= first SI, SL\_X=other SI,

\* critical-pair value : bold type

-쌍 경로에 속하는 입력이 하나 밖에 없으므로 표 1에 나타난 것처럼 경로 정당화의 역 방법으로 설정할 수가 있다. XOR(XNOR)의 경우는 경로 정당화 방법과 동일한 방법을 사용하면 된다. 역추적으로 경로 선의 값을 설정해야할 경우에는 PO로부터 역추적 하는 방법을 따른다. 그리고 팬 아웃이 존재하면 임계-쌍 경로가 생성되기 힘들지만 가능한 경우에는 만나면 모든 FOB에 동일한 값을 설정한다.

논문 [4]에서 정의한 것처럼 일반적으로 사용하는 '임계'를 '단일 임계'라고 하며 단일 임계의 종류와 '이중 임계'를 다음과 같이 정의한다.

정의 2 : 단일 임계는 '1-단일 임계'와 '0-단일 임계'로 구분한다. 1-단일 임계는 스템 입력 단의 값이 1일 때의 임계를, 0-단일 임계는 스템 입력 단의 값이 0일 때의 임계를 의미한다. 그리고 어떤 경로선의 값이 임계-쌍은 아니면서 1 및 0-단일 임계일 때를 '이중 임계(dual critical)'이라고 한다.

임계 여부의 표시는 \*(임계) 혹은 -(비임계)으로 표시하는데 1-단일 임계는 경로선의 값의 왼쪽에 표기하고, 0-단일 임계는 경로선의 값의 오른쪽에 표기한다. 만일 재결집 게이트가 있는 스템 지역에서 스템인 팬 아웃(fanout)과 재결집 게이트 사이에 임계-쌍 경로가 존재한다면, 그 재결집 게이트는 정상적인 결합 회로일 것이고 그렇지 않다면 임계-쌍 경로가 존재하지 않을 것이다. 그러므로 스템으로부터 재결집 게이트 사이에 임계-쌍 경로가 존재한다는 가정 하에 임계 경로를 설정하게 되면 비정상적인 결합을 한 게이트에서는 임계

성(criticality)이 전파되지 못하게 된다. 임계성이 전파되지 못하는 것은 제어 및 관측이 불가능해진다는 것이므로 시험 불가능한 결합을 확인할 수가 있다.

재결집 스템 지역에서의 경로선들을 다음의 형태로 분류하고 정의한다.

정의 2. 스템 지역의 회로 선이 팬 아웃의 후속자(successor)인 선을 스템선(SL; stem line)이라고 하고, 그 외의 회로 선들은 스템 입력(SI; stem input)이라고 한다. 그러므로 SI는 팬 아웃의 후속자가 아니다.

정의 3. 스템 지역에 속하는 어떤 게이트의 입력 선 중에 2 개 이상의 스템 선이 있으면 재결집 게이트(RG; reconvergent gate)라고 하고, 한 개 이하의 스템 선이 있으면 비결집 게이트(NRG; non-reconvergent gate)라고 한다. 스템의 출력에 해당하는 재결집 게이트들의 출력은 특별히 스템 출력(SO; stem output)이라고 한다.

정의 2와 3을 이용하여 임계-쌍 경로를 이용한 시험 불가능 결합의 확인 방법을 설명하면 다음과 같다.

첫째, 스템 선에 임계-쌍 값인  $w(1/0)$ 를 설정한 다음 출력 SO 쪽으로 감지 성을 이용하여 임계-쌍 값을 전파시킨다.

둘째, 팬 아웃으로부터 SO들 방향으로 임계-쌍 값을 전파시키는 중에 NRG를 만나면 표 1을 사용하여 임계-쌍 값의 전파 조건을 생각하여 게이트 출력 값 및 SI의 값을 설정한다. 그리고 만일 RG를 만나게 되면 입력 값에 의한 출력선의 값을 설정한 다음 임계성의 전파를 검토한다. 스템의 출력선인 SO는 항상 임계성을

표 2. AND(NAND), OR(NOR) 및 XOR 게이트의 경로 정당화  
Table 2. Line justification for AND(NAND), OR(NOR) and XOR gates.

AND(NAND)					NOTE	OR(NOR)					NOTE
INPUT			OUTPUT			INPUT			OUTPUT		
A	B	C	AND	NAND	A	B	C	OR	NOR		
<b>*w*</b>	<b>*v-</b>	<b>*v-</b>	<b>*w*</b>	<b>*w'*</b>	*w*:한 개 *v-:그 외	<b>*w*</b>	<b>-v'*</b>	<b>-v'*</b>	<b>*w*</b>	<b>*w'*</b>	*w*:한 개 -v'*:그 외
<b>*v-</b>	<b>*w*</b>	<b>*v-</b>	<b>*w*</b>	<b>*w'*</b>		<b>-v'*</b>	<b>*w*</b>	<b>-v'*</b>	<b>*w*</b>	<b>*w'*</b>	
<b>*v-</b>	<b>*v-</b>	<b>*w*</b>	<b>*w*</b>	<b>*w'*</b>		<b>-v'*</b>	<b>-v'*</b>	<b>*w*</b>	<b>*w*</b>	<b>*w'*</b>	
XOR (inputs 수 : odd)					XOR (inputs 수 : even)						
INPUT			OUTPUT	NOTE	INPUT		OUTPUT	NOTE			
A	B	C	Z		A	B	Z				
<b>*w*</b>	<b>*w*</b>	<b>*w*</b>	<b>*w*</b>	*w*:홀수 개의 *w* (짝수 개의 *w'*)	<b>*v'*</b>	<b>*w*</b>	<b>*w*</b>	*v*(or *v'*) : 한 개 *w*(or *w'*) :그 외			
<b>*w*</b>	<b>*w'*</b>	<b>*w'*</b>	<b>*w*</b>		<b>*w*</b>	<b>*v'*</b>	<b>*w*</b>				
<b>*w'*</b>	<b>*w*</b>	<b>*w'*</b>	<b>*w*</b>		<b>*w'*</b>	<b>*v*</b>	<b>*w*</b>				
<b>*w'*</b>	<b>*w'*</b>	<b>*w*</b>	<b>*w*</b>		<b>*v*</b>	<b>*w'*</b>	<b>*w*</b>				

By critical-pair cubes(critical-pair value : bold type)

값기<sup>[13]</sup> 때문에 RG에서의 출력선의 값에 대해서는 항상 임계성이 있는 것으로 간주하고 역방향으로 임계성을 정당화한다[표 2 참조].

이렇게 임계-쌍 경로를 이용한 임계성의 전파에 의해서 검출 가능한 결함을 확인 할 때의 문제점은 SO 게이트에 도달한 입력 값들이 임계성을 가진 값들이 여러 개가 있음으로서 오히려 임계성을 상실한다는 것이다.

예를 들어보면 그림 4의 예제 회로(2)에서 SO 게이트인 G3의 입력인 D와 E의 0-임계값이 둘 다 0이므로 임계성의 정당화에 의해 D와 E의 선행자들(A, B1, C, B2)에 대한 임계성이 제거되었다. 그러나 만일 이 값들이 한 쪽에서만 전파되었다면 임계성을 가질 수가 있다. 이와 같은 현상을 등가 현상(EVP; Equivalent Values Phenomenon)이라고 하고 다음과 같이 정의한다.

정의 4. 임계성을 가지고 전파되던 SO 게이트의 입력선 들의 동시적인 임계성이 SO의 임계성 정당화에 의해서 임계성이 제거되는 현상을 등가현상(EVP)에 의한 임계성의 제거라고 한다.

등가 현상에 의한 임계성의 제거가 발생하면 스템에 의해서 영향을 받지 않는 SI의 임계성을 확인해 보아야 한다. 그림 4의 회로(2)의 경우 이미 스템 B의 값에 의한 전파는 동시적인 임계값으로 판명되었기 때문에 스템에 의해 영향을 받지 않는 SI에 의한 경우만 남겨 된다. 그러므로 D=0, E=1일 때나 D=1, E=0일 때 SI의 임계 가능성을 확인해 보아야 한다. 결국 이 과정은 각 SO의 입력선 들에 대한 1 혹은 0의 SI 임계성을 확인 하는 것과 동일하다. 그러므로 SO의 각 입력선 들로부터 역방향으로 스템에 이르기까지의 경로 상에서 SI를 위한 임계-쌍 경로의 설정으로 확인할 수가 있다.

그림 4의 회로(2)에 적용을 하면 D= $w^*$ 에 의해서 A= $w^*$ , B1= $v^*$ 가 되고, E= $w^*$ 에 의해서 C= $w^*$ , B2= $v^*$ 가 된다.

그러므로 시험 불가능한 결함은 {B11, B21}이 된다.

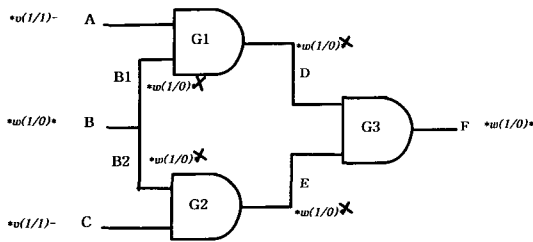


그림 4. 회로(2)  
Fig. 4. Circuit (2).

셋째로는 만일 SO의 입력 값이 등가 현상에 의한 임계성 제거가 발생하면, SO의 각 입력선 들로부터 역방향으로 스템에 이르기까지의 경로 상에서 SI를 위한 임계-쌍 경로의 설정으로 확인한다.

제안 알고리즘을 RICP(Redundancy Identification using Critical-pair Path) 알고리즘이라고 부르고 그 개략적인 절차를 그림 5에 나타내었다.

```

RICP() {
  /* 회로를 stem region으로 분할 한 후에 다음의 작업을 한다 */
  각각의 stem region에 대하여 { /* 검출 가능한 결함을 확인 과정 */
    stem region의 모든 stems에 대하여 {
      임계-쌍 값( $w^*$ )을 모든 SO로 전파시켜서 검출 가능한 결함을 확인한다.
      만일 SO의 입력에서 등가현상에 의한 임계성 제거가 발생하면
      모든 SO의 입력에 대해서 {
        임계-쌍 값( $w^*$ )을 역방향으로 스템에 이르기까지 SI에 의한 임계성의 확인을 통해 검출가능한 결함을 확인한다.
      }
    }
  }
  전체 결함에서 검출 가능한 결함이 아닌 것을 검출 불가능 결함으로 확정한다.
}
    
```

그림 5. RICP 알고리즘의 개략  
Fig. 5. Outline of RICP algorithm.

IV. 실험 및 고찰

FIRE 알고리즘은 그림 2에서 보는 바와 같이 모든 스템과 재결집 게이트의 모든 입력들에 대해, 또한 각 경로선의 값이 0과 1일 때에 대하여 각각 시험 불가능 결함을 찾기 위한 작업을 하기 때문에 매우 많은 경로선에 대하여 알고리즘을 적용해야 한다. 그러나 제안 알고리즘인 RICP는 모든 스템에 대해서만 알고리즘을 적용하며, 각 스템에 대하여서도 하나의 임계-쌍 값( $w$ )을 적용하기 때문에 알고리즘의 적용이 훨씬 줄어

든다.

FIRE 알고리즘을 설명하기 위한 논문 [12]에서 예들 든 그림 6의 회로(3)(경로선의 명칭 뒤의 괄호 안에 있는 숫자는 프로그램의 상에서 사용된 경로선의 번호임)를 이용하여 FIRE 알고리즘과 제안 알고리즘의 성능을 비교 평가해 본다.

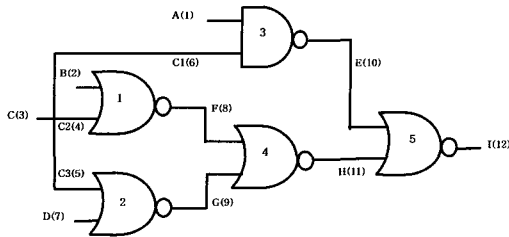


그림 6. 회로(3)  
Fig. 6. Circuit (3).

그림 2에 나타난 FIRE 알고리즘을 그림 6의 회로(3)에 적용시키면 다음과 같다. 스템들과 재결집 게이트의 입력들인 L의 집합은 {C, F, G, E, H}이다. 먼저 C에 대하여 FIRE 알고리즘을 적용시키면 다음과 같다. 다음에서 회로선 1 상의 s-a-v 결함을 lv로 나타낸다.

$C = \bar{0}$ 에 의한 제어 불가능성의 전파에 의해서  $C1=C2=C3=\bar{0}$ ,  $F=G=\bar{1}$ ,  $H=\bar{0}$  및  $I=\bar{1}$ 가 된다. 그리고 관측 불가능한 회로 선이 B, D, E, A, C1이 된다. 그러므로  $S0=\{C11, C21, C31, F0, G0, H1, I0, B0, B1, D0, D1, E0, E1, A0, A1, C10\}$ 이다.

$C = \bar{1}$ 에 의한 제어 불가능성의 전파에 의해서  $C1=C2=C3=\bar{1}$ ,  $E=\bar{0}$  및  $I=\bar{1}$ 가 된다. 그리고 관측 불가능한 회로 선이 H, F, G, B, C2, C3, D가 된다. 그러므로  $S1=\{C10, C20, C30, E1, I0, H0, H1, F0, F1, G0, G1, B0, B1, D0, D1, C20, C21, C30, C31, D0, D1, A0, A1\}$ 이다. 그러므로  $S0 \cap S1 = \{A0, A1, B0, B1, C10, C21, C31, D0, D1, E1, F0, G0, H1, I0\}$ 가 된다.

여기서 시간 복잡도(time complexity) 분석을 위해서 경로 탐색 회수를 측정해보면 그림 5의 결과처럼 제어 불가능성의 전파를 위한 순방향 탐색 회수가 36 회이고, 관측 불가능성을 찾기 위하여 역방향으로 탐색하는 회수가 62 회이다.

그 외의 테스트 대상 회로 선인 {F, G, E, H}에 대한 결과는 그림 7에 나타내었다.

한편, 임계-쌍 경로법을 이용한 경우에는 테스트할 대상 회로 선이 스템들뿐이다. 그러므로 그림 4의 회로

에서는 스템인 C만 대상이 된다. C에 대한 임계-쌍 경로법의 적용은 다음과 같다.

$C=C1=C2=C3=*w*$ 이고, 이 임계값을 각 NRG인 게이트 1, 2, 3의 출력 선에 전파하면  $E=*w'$ ,  $F=G=*w*$ 가 된다. 그리고 SI인 A, B, D에 대한 값은 표 1에 의해서  $*v-$ ,  $-v'$ ,  $-v'$ 이 된다. 게이트 4는 RG이므로 값을 게이트 형태에 따라 전파시키고 임계성을 평가하면  $H=*w*$ 가 된다. 그러므로 SO인 I의 값은  $v'$ 이 된다. 그리고 SO의 임계성 평가에 따라 경로선 E는 1-단일 임계가 제거되기 때문에 역방향의 A, C1의 1-단일 임계도 제거시킨다. 그리고 경로선 H에서는 0-단일 임계가 제거되므로 역방향으로 F, B, C2 및 G, C3, D의 0-단일 임계도 제거시킨다. 그래서 모든 경로선의 임계성은 다음과 같다.

$$C=*w*, C1=-w*, C2=C3=*w-, A=-v-, B=D=-v-, F=G=*w', E=-w', H=*w-, I=*v'$$

SO의 입력인 E와 H가 증가현상에 의한 임계성 제거가 발생되지 않는다. 그러므로 역방향으로의 임계-쌍 값에 의한 검출 가능 결함을 추적할 필요가 없다. 그래서 검출 가능 결함은 {C0, C1, C11, C20, C30, E0, F1, G1, H0, I1}이다

그러므로 시험 불가능한 결함들은 임계값을 설정할 수 없는 경로 값들인 {A0, A1, B0, B1, C10, C21, C31, D0, D1, E1, F0, G0, H1, I0}이다. 제안 알고리즘에 의한 경로선 탐색 회수는 순방향 8회, 역방향 15회로 FIRE에 비해서 훨씬 적다.

FIRE 알고리즘과 제안 알고리즘의 실험 결과를 그림 7과 그림 8(btest1은 그림 6의 회로에 대한 파일명)에 나타내었다. 이 알고리즘들은 visual C++로 구현하였고, ISCAS85 벤치마크 회로[14]를 사용해 Pentium Pro PC에서 검증했다. 실험을 위한 스템 영역은 회로 상에 나타나는 모든 팬 아웃에 대하여 재 결집 게이트가 있는 잉여 영역이다. FIRE에서는 모든 재결집 게이트의 입력에 대하여 2 번씩의 확인 작업을 추가적으로 해야하기 때문에 훨씬 많은 시간이 소요된다. 그러므로 표 3에 나타난 실험 결과에서 보듯이 테스트해야하는 대상(test lists)이 FIRE 알고리즘의 5.5%(C17 회로를 제외하면 2.3%) 밖에 되지 않는다. 제안 알고리즘에 의한 시험 불가능 결함의 확인 시간이 FIRE를 사용한 시간의 2.6%(C17 회로를 제외하면 2.8%) 밖에 걸리지 않았다. 경로 탐색 회수도 FIRE 방법의 18.8%(C17 회로를 제외하면 15.2%) 정도만 탐색하면 된다. 그러므로

RICP 알고리즘이 FIRE 알고리즘보다는 훨씬 효율적인 것으로 나타났다.

```

* Identification Times of Untestable Faults using FIRE for btest1
=====
STEM(FOS) : Start : Types : Srch no : Bak no. : RCN_cnt / PO list : UTF_cnt / Untestable Faults lists
total faults : TF_cnt / Testable Faults
=====
0(3) : 3 : Stems : 13 : 13 : 2 : (11:2)(12:1) : 14 : 1/1, 1/0, 2/1, 2/0, 4/1, 5/1, 6/0, 7/1, 7/0, 8/0, 9/0, 10/1, 11/1, 12/0.
0.000 sec : 24 : 10 : 3/0, 3/1, 4/1, 5/1, 6/0, 8/0, 9/0, 10/1, 11/1, 12/0.
-----
0(3) : 8 : Input : 6 : 13 : 2 : (11:2)(12:1) : 1 : 8/0.
0.000 sec : 24 : 23 : 1/0, 1/1, 2/0, 2/1, 3/0, 3/1, 4/0, 4/1, 5/0, 5/1, 6/0, 6/1, 7/0, 7/1, 8/0, 9/0, 9/1, 10/0, 10/1, 11/0, 11/1, 12/0, 12/1.
-----
0(3) : 9 : Input : 5 : 12 : 2 : (11:2)(12:1) : 1 : 9/0.
0.000 sec : 24 : 23 : 1/0, 1/1, 2/0, 2/1, 3/0, 3/1, 4/0, 4/1, 5/0, 5/1, 6/0, 6/1, 7/0, 7/1, 8/0, 8/1, 9/0, 10/0, 10/1, 11/0, 11/1, 12/0, 12/1.
-----
0(3) : 10 : Input : 9 : 17 : 2 : (11:2)(12:1) : 11 : 2/1, 2/0, 4/1, 5/1, 7/1, 7/0, 8/0, 9/0, 10/1, 11/1, 12/0.
0.000 sec : 24 : 13 : 1/0, 1/1, 3/0, 3/1, 4/1, 5/1, 6/0, 6/1, 8/0, 9/0, 10/1, 11/1, 12/0.
-----
0(3) : 11 : Input : 3 : 7 : 2 : (11:2)(12:1) : 0 :
0.000 sec : 24 : 24 : 1/0, 1/1, 2/0, 2/1, 7/0, 7/1, 3/0, 3/1, 4/0, 4/1, 5/0, 5/1, 6/0, 6/1, 8/0, 8/1, 9/0, 9/1, 10/0, 10/1, 11/0, 11/1, 12/0, 12/1.
-----
Total evaluation : Time = 0.000 sec, FIRE Search numbers = 36, Backward Tracing = 62
Total : Faults = 24, Untestable Faults = 14, Testable Faults = 10
=====
    
```

그림 7. 회로(3)에 대한 FIRE 알고리즘의 결과  
 Fig. 7. Result using FIRE algorithm for circuit (3).

```

* Identification Times of Untestable Faults using CPP for btest1
=====
STEM(FOS) : RCN_cnt : RCN list : UTF_cnt : Untestable Faults lists
total faults : TF_cnt : Testable Faults
-----
0(3) : 2 : (11) (2) : 14 : 1/1, 1/0, 2/1, 2/0, 4/1, 5/1, 6/0, 7/1, 7/0, 8/0, 9/0, 10/1, 11/1, 12/0.
0.000 sec : 24 : 10 : 3/0, 3/1, 4/0, 5/0, 6/1, 8/1, 9/1, 10/0, 11/0, 12/1.
=====
Total evaluation : Time = 0.000 sec, CPP Search numbers = 8, Backward Tracing = 15
Total : Faults = 24, Untestable Faults = 14, Testable Faults = 10
=====
    
```

그림 8. 회로(3)에 대한 RICP 알고리즘의 결과  
 Fig. 8. Result using RICP algorithm for circuit (3).



표 3. ISCAS85 벤치마크 회로에서의 시험 불가능 결함의 확인 시간 비교(FIRE와 제안 알고리즘)  
 Table 3. Identification Time Comparing for Untestable Faults on ISCAS85 Benchmark Circuits  
 (FIRE and RICP algorithm).

Circuits	Steps	total lines	FIRE algorithm					RICP algorithm					RICP / FIRE					
			$N_{uf}$	$L_t$	$L_{fs}$	$L_{bs}$	$T_i$	$N_{uf}$	$L_t$	$L_{fs}$	$L_{bs}$	$T_i$	$L_t$	$L_{fs}$	$L_{bs}$	$L_s$	$T_i$	
c1355	259	1,355	15	6,565	366,070	237,664	12,390	15	229	13,300	25,728	0.450	0.034	0.036	0.108	0.064	0.036	
c17	3	17	0	8	53	59	0.050	0	3	20	41	0.000	0.375	0.377	0.694	0.544	0.000	
c1908	385	1,908	4	22,320	586,726	881,171	43,410	4	382	45,929	87,379	0.690	0.017	0.078	0.099	0.090	0.015	
c2670	454	2,670	29	7,666	901,018	177,998	14,970	28	391	17,587	29,968	0.350	0.051	0.019	0.168	0.044	0.023	
c3540	579	3,540	93	56,883	712,670	1,247,155	110,010	93	569	126,723	248,463	2.070	0.010	0.177	0.199	0.191	0.018	
c432	89	432	16	6,513	103,470	188,366	12,780	3	83	11,483	27,553	0.160	0.012	0.110	0.146	0.133	0.012	
c499	59	449	18	3,192	115,308	130,336	5,940	7	40	7,188	12,076	0.110	0.012	0.062	0.092	0.078	0.018	
c5315	806	5,315	20	16,132	204,793	290,616	28,720	20	636	47,468	69,893	1.370	0.039	0.231	0.240	0.236	0.047	
c6288	1456	6,288	33	769,268	9,624,257	7,690,971	1,477,910	34	1,456	1,247,473	2,293,274	73.460	0.001	0.129	0.298	0.204	0.049	
c7552	1300	7,552	30	50,415	653,599	1,000,070	100,690	30	1,275	128,659	249,939	2.720	0.025	0.196	0.249	0.228	0.027	
c880	125	880	40	2,140	28,051	37,713	3,850	41	71	5,575	11,271	0.160	0.033	0.198	0.298	0.256	0.041	
Avg.													전체 평균	0.055	0.146	0.235	0.188	0.026
													c17 제외 평균	0.023	0.123	0.189	0.152	0.028

\*  $N_{uf}$  = numbers of untestable faults,  $L_t$  = test lists,  $L_{fs}$  =forward searching lines,  
 $L_{bs}$  =backward searching lines,  $L_s$  =total searching lines,  $T_i$  =identification times

### V. 결론

본 연구에서 우리는 재결집 영역에 존재하는 시험 불가능한 결함의 효율적인 검출 방안으로서 회로의 스템으로부터 임계-쌍 경로를 설정하여서 임계 설정이 불가능한 경로선의 값을 구하는 방법을 제안하였다. 기존의 FIRE 방법이 모든 스템들과 재결집 게이트의 입력들에 대해서 제어 및 관측의 불가능성을 확인하는데 비해서 제안 알고리즘에서 시험 대상이 되는 경로선은 단지 스템들뿐이기 때문에 시험 불가능 결함의 확인 속도나 과정에서의 경로 탐색 회수에 있어서 훨씬 효과적이었다.

일반적으로 테스트를 생성하는 과정에서 시험 불가능 여부를 모르게 되면 테스트를 생성할 수 있을 때까지 반복적인 시행 착오를 겪게 되어서 많은 시간을 시험 불가능한 경로선의 결함 발견에 소모하게 된다. 그러므로 회로의 시험 불가능한 결함을 발견하기 위한 직접적인 기술로서든지, 간접적인 기술로 사용하여 회로의 테스트를 생성하는 전처리 과정에서든지 간에 제

안 알고리즘을 사용하면 전체적인 테스트 과정의 시간을 줄이는데 효율적인 것으로 사료된다.

만일 ATG를 임계-쌍 경로법으로 행한다면 테스트를 생성하는 과정에 동적으로 이 알고리즘을 적용시킬 수가 있어서 더욱 더 효율적인 것이다. 그러므로 다음의 연구과제는 임계-쌍 경로법을 사용하여 테스트 생성이나 결함 시뮬레이션을 하는 방법에 관한 연구이다.

### 참고 문헌

- [1] O. H. Ibarra and S. K. Sahni, "Polynomially Complete Fault Detection Problems," IEEE Trans. on Computers, Vol. C-24, pp.242-249, March 1975.
- [2] Mahesh A. Iyer and Miron Abramovici, "Sequentially Untestable Faults Identified Without Search," Proc. 1994 International Test Conference, pp.259-266, 1994.
- [3] Miron Abramovici and Mahesh A. Iyer, "One-Pass Redundancy Identification and Removal," Proc. 1992 International Test

- Conference, pp.807-815, 1992.
- [4] 서성환, 안광선, "FFR에서의 임계-쌍 경로를 이용한 효율적인 테스트 생성", 전자공학회 논문지 제 36 권 C 편 제 4 호, pp.215~230, 1999년
- [5] M. Harihara and P. R. Menon, "Identification of Undetectable Faults in Combinational Circuits," Proc. International Conference on Computer Design, pp.290-293, Oct. 1989.
- [6] M. Abramovici, D. T. Miller and R. K. Roy, "Dynamic Redundancy Identification in Automatic Test Generation," IEEE Trans. on CAD, pp.404-407, Mar. 1992.
- [7] I. M. Ratiu, A. Sangiovanni-Vincentelli and D. O. Pederson, "VICTOR: a fast VLSI testability analysis program," Proc. 1982 International Test Conference, pp.397-401, 1982.
- [8] M. Abramovici, J. J. Kulikowski, and R. K. Roy, "The Best Flip-Flop to Scan," Proc. International Test Conference, pp.166-173, 1991.
- [9] M. Abramovici, J. J. Kulikowski, P. R. Menon and D. T. Miller, "SMART and FAST: Test Generation for VLSI scan-design circuits," IEEE Design and Test of Computers, pp.43-54, Aug. 1986.
- [10] Wolfgang Kunz and Dhiraj K. Pradhan, "Recursive Learning: A New Implication Technique for Efficient Solutions to CAD Problems-Test, Verification, and Optimization", IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, Vol. 13, No. 9, Sept. 1994.
- [11] Wanlin Cao and Dhiraj K. Pradhan, "Sequential Redundancy Identification Using Recursive Learning," IEEE/ACM International Conference on Computer-Aided Design, pp.56-62, 1996.
- [12] Mahesh A. Iyer and Miron Abramovici, "Low-Cost Redundancy Identification for Combinational Circuits," Proc. 1994 International Conference on VLSI Design, pp.315-318, 1994.
- [13] M. Abramovici, M. A. Breuer, and A. D. Friedman, Digital Systems Testing and Testable Design, Computer Science Press, 1990.
- [14] F. Brglez and H. Fujiwara, "A Neutral Netlist of 10 Combinational Benchmark Design and special Translator in Fortran," International Symposium on Circuits and Systems, June 1985.

---

 저 자 소 개
 

---

徐 聖 煥(正會員) 第 36卷 C編 第 4號 參照

安 光 善(正會員) 第 36卷 C編 第 4號 參照