

## New Scan Design for Delay Fault Testing of Sequential Circuits

許 瓊 會\* · 康 容 碩\*\* · 姜 成 昊\*\*\*

(Kyung-Hoi Huh · Yong-Seok Kang · Sungho Kang)

**Abstract** - Delay testing has become highlighted in the field of digital circuits as the speed and the density of the circuits improve greatly. However, delay faults in sequential circuits cannot be detected easily due to the existence of state registers. To overcome this difficulty a new scan flip-flop is devised which can be used for both stuck-at testing and delay testing. In addition, the new scan flip-flop can be applied to both the existing functional justification method and the newly-developed reverse functional justification method which uses scan flip-flops as storing the second test patterns rather than the first test patterns. Experimental results on ISCAS 89 benchmark circuits show that the number of testable paths can be increased by about 10% on the average.

**Key Words** : delay fault, delay testing, scan flip-flop, functional justification

### 1. 서 론

디지털 회로가 좀 더 빨라지고 집적되면서, 회로를 더 효과적으로 테스트하는 것이 중요하게 되었다. 게다가, 고성능의 칩 개발에 대한 요구가 증가하면서 회로의 기능뿐만 아니라, 신호 천이 지연을 검사하는 것이 필요하게 되었다. 따라서, 제조 공정에서 생긴 결함을 검출하기 위한 고착 고장파 지연 고장의 테스트가 많이 사용되는데, 지연 고장 테스트의 목적[1]은 생산된 회로가 요구되는 클럭 주파수 내에서 제대로 동작하는 지를 확인하는 것이다. 이를 위해, 두 개의 연속적인 벡터가 필요한데, 이것은 주입력단의 신호 천이가 회로를 통해 주어진 시간 내에 주출력까지 전파되는 지를 확인해야 하기 때문이다.

시스템의 시간에 관련된 결함은 집중된 지연 고장으로 인해 생기거나 경로에 흩어진 분산된 지연 고장들이 합쳐져서 생기게 된다. 두 종류의 지연 고장 모델이 개발되어 널리 쓰이고 있는데 이는 게이트 지연 고장 모델(gate delay fault model)[2, 3]과 경로 지연 고장 모델(path delay fault model)[4, 5]이다. 게이트 지연 고장 모델은 지연이 회로 내의 게이트 입력이나 출력에 집중되었다고 가정하는 것이므로 회로내의 각 게이트에 널리 퍼져 있는 작은 지연으로 생긴 결함은 검출 불가능하다. 반면에, 경로 지연 고장 모델은 시험 중인 경로를 따라 있는 작은 지연들이 모여서 회로가 오동작하도록 하는 것을 말한다. 따라서, 이 모델에 기초를 둔 테스트는 회로의 경로에 따른 집중 또는 분산된 결함을 모두 검출할 수 있다. 또, 회로 내의 긴 경로들의 고장에 대해 클럭 속도에 따른 테스트 성공 여부에 관한 정확한 모습을 제공하므로 속도에 따라 칩을 분류하는 데 쓰일 수 있다[6].

경로 지연 고장에 대한 테스트에는 크게 무해저드 경성(hazard free robust) 테스트, 경성(robust) 테스트, 연성(non-robust) 테스트의 세 종류로 나눌 수 있다[7]. 무해저드 경성 테스트는 경로에의 신호들에 동적 해저드(dynamic hazard)가 없고 회로의 다른 부분에서의 지연과 무관하게 경로상의 지연 고장을 검출할 수 있는 두 패턴의 테스트이다. 경성 테스트는 회로의 다른 부분에서의 지연과 무관하게 경로상의 지연 고장을 검출할 수 있는 두 패턴의 테스트로 경로에 특정 천이를 유발시켜야 하고 경로상의 모든 신호들은 특정 천이가 도달하기 전까지 최종 값을 알 수 없어야 한다. 경성 테스트는 시험중인 경로가 초과 지연을 가지면 실패하지만, 회로 내의 다른 경로가 초과 지연을 갖는 경우에도 실패한다. 따라서 회로에 대한 테스트의 목적이 진단이 아니고 단지 고장 검출 만이라면 경성 테스트를 구하는 것이 무해저드 경성 테스트를 구하는 것보다 유리하다. 경성 테스트의 요건을 갖추지 못한 테스트를 연성 테스트라고 부른다. 이는 두 개로 다시 구분될 수 있다. 먼저 경성 테스트에 가까운 연성 테스트를 강연성(strong non-robust) 테스트라 부른다. 강연성 테스트는 모든 요건이 경성 테스트와 같지만 정적 해저드(static hazard)가 있어도 무방한 두 패턴의 테스트이다. Schulz[8]는 두 번째 사이클의 경우에 경성 테스트와 요구되어지는 값이 같고 첫 번째 사이클의 경우 요구되어지는 것이 경로의 시작 부분의 초기값으로만 결정되는 방법으로 연성 테스트를 구하였는데 이를 약연성(weak non-robust) 테스트라고 부른다. 약연성 테스트를 이용할 경우는 경로를 제외한 회로의 나머지 부분에서는 지연 고장이 없다

트는 회로의 경로에 따른 집중 또는 분산된 결함을 모두 검출할 수 있다. 또, 회로 내의 긴 경로들의 고장에 대해 클럭 속도에 따른 테스트 성공 여부에 관한 정확한 모습을 제공하므로 속도에 따라 칩을 분류하는 데 쓰일 수 있다[6].

경로 지연 고장에 대한 테스트에는 크게 무해저드 경성(hazard free robust) 테스트, 경성(robust) 테스트, 연성(non-robust) 테스트의 세 종류로 나눌 수 있다[7]. 무해저드 경성 테스트는 경로에의 신호들에 동적 해저드(dynamic hazard)가 없고 회로의 다른 부분에서의 지연과 무관하게 경로상의 지연 고장을 검출할 수 있는 두 패턴의 테스트이다. 경성 테스트는 회로의 다른 부분에서의 지연과 무관하게 경로상의 지연 고장을 검출할 수 있는 두 패턴의 테스트로 경로에 특정 천이를 유발시켜야 하고 경로상의 모든 신호들은 특정 천이가 도달하기 전까지 최종 값을 알 수 없어야 한다. 경성 테스트는 시험중인 경로가 초과 지연을 가지면 실패하지만, 회로 내의 다른 경로가 초과 지연을 갖는 경우에도 실패한다. 따라서 회로에 대한 테스트의 목적이 진단이 아니고 단지 고장 검출 만이라면 경성 테스트를 구하는 것이 무해저드 경성 테스트를 구하는 것보다 유리하다. 경성 테스트의 요건을 갖추지 못한 테스트를 연성 테스트라고 부른다. 이는 두 개로 다시 구분될 수 있다. 먼저 경성 테스트에 가까운 연성 테스트를 강연성(strong non-robust) 테스트라 부른다. 강연성 테스트는 모든 요건이 경성 테스트와 같지만 정적 해저드(static hazard)가 있어도 무방한 두 패턴의 테스트이다. Schulz[8]는 두 번째 사이클의 경우에 경성 테스트와 요구되어지는 값이 같고 첫 번째 사이클의 경우 요구되어지는 것이 경로의 시작 부분의 초기값으로만 결정되는 방법으로 연성 테스트를 구하였는데 이를 약연성(weak non-robust) 테스트라고 부른다. 약연성 테스트를 이용할 경우는 경로를 제외한 회로의 나머지 부분에서는 지연 고장이 없다

\* 準 會 員 : 延世大 大學院 電氣工學科 碩士課程  
 \*\* 正 會 員 : 延世大 大學院 電氣工學科 博士課程  
 \*\*\* 正 會 員 : 延世大 工大 電氣工學科 助教授 · 工博  
 接受日字 : 1999年 7月 23日  
 最終完了 : 1999年 8月 4日

는 보장이 있어야 경로상의 지연 고장을 검출할 수 있다는 단점이 있다.

이 네 가지 종류의 테스트의 정의에 따른 요건들을 비교하면 약연성, 강연성, 경성, 무해저드 경성 순으로 테스트를 구하기 위한 요건이 많아진다. 요건이 많아질수록 테스트하기 힘들고, 따라서 위의 네 가지 종류에서 무해저드 경성 테스트가 가장 양질의 테스트이다.

본 논문에서는 동기 순차 회로의 지연 고장 검출을 위한 효과적인 방법에 대한 연구이므로 위의 네 가지 종류의 테스트 중 경성 테스트를 우선 고려하였고, 차선의 방법으로 강연성 경로 지연 고장에 대한 테스트 생성에 대해서 실험을 하였다.

다음 장에서는 기존의 테스트 방법에 대해 알아보도록 하고, 3장에서는 역기능적 지정을 이용한 테스트 방법과 그에 따른 ATPG 도구의 적용 방법을 설명한다. 4장에서는 3장에서 설명한 테스트 방법을 실제 적용하기 위한 새로운 스캔 구조를 제안한다. 새로운 테스트 알고리즘에 따른 테스트 순서는 5장에서 다루어지며, 6장에서는 새로운 테스트 알고리즘을 적용한 결과를 보이고, 이에 대한 의미를 설명한다. 마지막 7장은 결론으로써, 본 연구의 내용을 요약하고, 새로운 과제를 제시한다.

## 2. 기존의 테스트 방법

조합 회로의 경우에는 앞서 설명한 것처럼, 지연 고장 검출을 위해 두 개의 테스트 벡터를 가하면 된다. 그러나 이와는 달리 순차 회로의 지연 고장을 위한 테스트 패턴을 생성하는 것은 힘들다. 이는 순차 회로의 주출력이 회로의 주입력뿐만 아니라 초기 상태에 의해서도 영향을 받기 때문이다. 순차 회로의 고착 고장을 검출하기 위해서, 플립-플롭을 변형한 스캔 플립-플롭을 사용하는데, 스캔 플립-플롭은 초기 상태를 임의로 조정 가능하므로, 고착 고장을 위한 테스트 패턴을 회로에 쉽게 가할 수 있다. 그러나, 지연 고장 테스트를 위해서는 두 개의 테스트 패턴이 연속적으로 인가되어야 하므로, 스캔 플립-플롭을 사용하더라도 지연 고장 검출률을 충분히 높일 수 없다.

일반적으로, 표준 스캔(standard scan)을 사용하는 지연 고장 테스트 방법으로는 스캔 이동(scan shifting) 방법과 기능적 지정(functional justification) 방법이 있다[9]. 스캔 이동 방법의 경우에는 테스트 가능한 많은 경로에 대한 테스트를 생성하지 못할 수가 있는데, 그림 1은 이러한 예를 보여준다. 만일 스캔 사슬이 A-B로 연결되어 있고 A의 하강 천이를 고려할 경우 이의 테스트를 위해 B는 해저드 없는 0을 가져야 한다. 하지만 스캔 이동에 의해 B에서는 두 번째 시간대에 1을 가질 수밖에 없다. 따라서 스캔을 이용하여 이 고장을 검출하는 테스트 패턴을 연속적으로 인가하는 것은 불가능하다.

이러한 스캔 이동의 문제점을 해결하고 정확한 지연 고장 검사를 위해 기능적 지정 방법이 사용된다. 이 방법에서 첫 벡터는 스캔 플립-플롭으로 임의적 들어가고, 두 번째 벡터는 회로의 기능에 따라 정해진다. 그러나, 이 역시 두 벡터가 두 시간대 동안 테스트에 필요한 논리 값들을 만족시켜야 한다는 문제점을 가지고 있다. 또한, 두 번째 시간대에 필요한

테스트 패턴들의 경로의 입력들은 표 1에서 보는 바와 같이 'X(don't care)'가 아닌 '0'이나 '1'을 갖는다. 즉, 테스트하려는 경로의 경로의 입력들 때문에 두 번째 시간대에 필요한 테스트 패턴들의 논리 값들은 'X' 보다 '0'이나 '1'을 더 많이 갖는다.

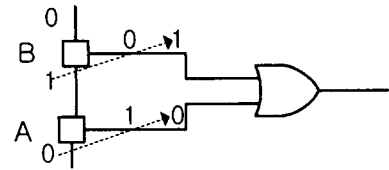


그림 1 스캔 이동으로 테스트 생성이 불가능한 예  
Fig. 1 Example of undetectable fault with scan-shifting

표 1 경로 지연 고장 테스트에서의 경로의 입력  
Table 1 Off-path input constraints for path delay fault test

소자 유형	경로 입력 천이	경로의 입력 조건	
		경성	연성
AND/NAND	상승 천이	X1	X1
	하강 천이	11(무해저드)	X1
OR/NOR	상승 천이	00(무해저드)	X0
	하강 천이	X0	X0

결국, 순차 회로의 지연 고장을 완벽히 검출하기 위해서는 확장 스캔 플립-플롭(enhanced scan flip-flop)을 사용하여야 한다. 확장 스캔 플립-플롭은 두 개의 패턴을 저장하기 위해 두 개의 플립-플롭을 사용하므로, 순차 회로를 조합 회로화할 수 있다. 그러나, 추가되는 플립-플롭만큼의 면적 오버헤드가 증가하므로 실질적인 순차 회로에 거의 사용되지 못하고 있다.

## 3. 역기능적 지정을 이용한 테스트 방법

### 3.1 테스트 방법

테스트 방법을 좀 더 명확히 설명하기 위해, 우선 몇 가지 용어들을 정리하면 다음과 같다.  $P1$ ,  $P2$ 를 각각 첫 번째 테스트 패턴과 두 번째 테스트 패턴이라고 생각한다. 좀 더 세부적으로  $P1$ 은  $Pp1$ 과  $Ps1$ 로 구성되는데,  $Pp1$ 은 첫 번째 테스트 패턴 중 주입력에 가해지는 패턴을 나타내고,  $Ps1$ 은 플립-플롭에서 가해지는 패턴을 의미한다.  $P1$ 과 마찬가지로  $P2$ 도  $Pp2$ 와  $Ps2$ 로 구성되며, 의미하는 바는  $Pp1$ ,  $Ps1$ 과 동일하다. 그리고 마지막으로 플립-플롭의 값에 상관없이  $Ps1$ 을 생성할 수 있는 주입력 패턴을  $Pp0$ 으로 표시한다. 출력의 경우에는,  $R1$ 과  $R2$ 를  $P1$ 과  $P2$ 의 응답 패턴을 의미하는 것으로 각각 나타낸다.

앞서 언급했듯이, 기능적 지정 방법은 조절용이도가 높은 스캔 플립-플롭을 'X' 값이 많은 첫 번째 테스트 패턴을 저장하는 데 사용한다. 이러한 낭비를 없애기 위해, 두 번째 테스트 패턴을 스캔 플립-플롭에 저장하는 방법을 제안한다. 반대로, 첫 번째 테스트 패턴은 회로의 기능에 의해 지정되도록 할 수 있고, 따라서 이러한 방법을 역기능적 지정(reverse

functional justification) 방법이라고 정의한다. 이 방법을 적용하기 위해서는  $PsI$ 을 회로의 기능을 통해 효과적으로 생성하는 방법이 필요하다. 본 논문에서는 이를 위해, 회로의 주 입력만을 조정하는 방법을 사용한다.

### 3.2 ATPG를 이용하여 $PsI$ 을 얻는 방법

실제적인 순차 회로에 역기능적 지정 방법을 적용하기 위해서, 본 논문에서는 ATPG를 이용하여  $PsI$ 을 생성하는 방법을 제시한다. ATPG 도구는 주어진 고착 고장에 대한 테스트 패턴을 생성하는 프로그램이다. 순차 회로에서 플립-플롭을 모두 제거한 조합 회로 부분을 고려했을 때,  $PsI$  패턴 중 '0' 값을 필요로 하는 출력단에 모두 인버터를 삽입하면, 조합 회로 부분의 출력은 '1' 값이나 'X' 값을 가지게 될 것이다. 이제, 주출력 부분을 제외한, 나머지 플립-플롭과 연결된 출력들 중에 '1' 값을 갖는 것들을 모두 하나의 AND 게이트의 입력으로 묶고, 이 AND 게이트의 출력 부분에 0-고착 고장을 삽입한다. ATPG는 이 고장에 대한 테스트 패턴을 생성할 것이고, 생성된 패턴들이 바로  $PsI$ 을 만들어 낼 수 있는 입력 패턴이다. 여기서  $Pp0$ 을 얻기 위해서는 ATPG 도구를 실행할 때, 플립-플롭과 연결된 입력들에 'X' 값으로 제한을 두어,  $PsI$ 을 생성할 수 있는 순수한 주입력 패턴을 얻을 수 있다.

그림 2는 S27 순차 벤치마크 회로의 경우에 수정된 조합 회로 부분을 나타낸다. 예를 들어 만약, 필요한 첫 번째 테스트 패턴이 '10X(G11, G10, G13)'라고 하면, G11 게이트의 출력은 AND 게이트의 입력과 직접 연결시킨다. '0' 논리 값을 갖는 G10 게이트와 AND 게이트 사이에는 인버터를 삽입하고, 'X' 논리 값을 갖는 G13은 수정을 할 필요가 없다. 그림 2에서 첫 번째 테스트 패턴에 따라 추가된 게이트들을 회색으로 칠하여 표시하였다. AND 게이트의 출력부에 0-고착 고장을 삽입한 후, 플립-플롭과 연결된 입력인 G5, G6, G7에 'X' 논리 값으로 제약 조건을 주고 ATPG 도구를 시행한다. ATPG 도구는 결과로서  $PsI$ 을 생성할 수 있는  $Pp0$ 을 제공해 준다.

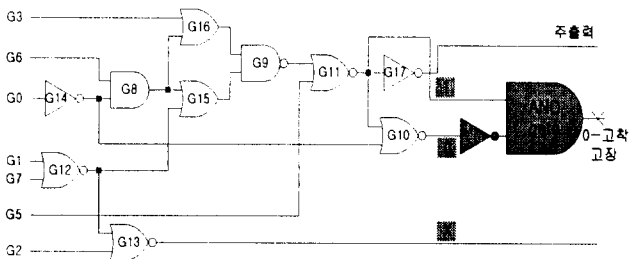


그림 2 S27 순차 벤치마크 회로의 수정된 조합 회로 부분  
Fig. 2 Modified combinational part of S27 benchmark circuit

ATPG의 고장 검출률은 역기능적 지정 방법의 성능을 직접적으로 표시하는 중요한 의미가 있는 값이다. 왜냐하면, 만약 ATPG 도구가  $Pp0$ 을 생성하지 못한다면 그에 해당하는 경로에 대한 지연 고장은 테스트할 수 없기 때문이다.

### 4. 새로운 스캔 플립-플롭 설계

역기능적 지정 방법을 적용할 때의 요구 조건을 만족시키기 위해 새로운 스캔 플립-플롭을 고안해 내었다. 이 스캔 플립-플롭은  $Ps2$ 를 저장한 채로  $PsI$ 을 통과시킬 수 있어야 하기 때문에, 면적 오버헤드를 감수해야만 하지만, 기능적 지정 방법과 역기능적 지정 방법을 혼용할 수 있으므로 지연 고장 검출률을 많이 높일 수 있다.

그림 3은 새로운 스캔 플립-플롭 구조를 CMOS 수준으로 설계한 것이다. 이 스캔 플립-플롭은 세 개의 래치(L1, L2, L3)와 두 개의 멀티플렉서로 구성되어 있는데, 회색으로 칠한 소자들은 표준 스캔 플립-플롭과 비교해 추가된 것들이다. 실제 회로에서는  $Data\ in$ 과  $Scan\ in$ , 그리고 스캔의 출력에 인버터가 붙는데, 각각의 새로운 스캔 플립-플롭마다의 면적 오버헤드는 이를 포함하여 46 퍼센트 정도이다. 이러한 오버헤드는 문제가 될 수 있지만, 지연 고장 검출률을 1.5배 이상 향상시킬 수 있다면 이 정도의 오버헤드는 충분히 받아들일 수 있다. 게다가 확장 스캔 플립-플롭의 오버헤드가 약 80 퍼센트에 이른다는 것을 고려한다면 더욱 합당하다고 할 수 있다.  $test\_opt$ 은 외부 핀으로 추가되어 어떤 테스트 방법으로 테스트를 할 것인지를 선택할 수 있게 한다.

정상 동작 모드( $test\_mode$ 와  $test\_opt$  신호가 모두 '0' 값을 가질 때)에서, 이 스캔 플립-플롭은 보통의 플립-플롭과 똑같이 동작한다. 스캔 이동 모드로는 다음과 같은 두 가지가 있는데, 첫 번째는  $test\_mode$ 가 '1' 값을,  $test\_opt$ 이 '0' 값을 가질 때로서, 두 번째 래치(L2)를 슬레이브로 사용하여 스캔 이동을 수행하는 모드이다. 반면에,  $test\_mode$ 와  $test\_opt$  모두 '1' 값을 가졌을 때는 세 번째 래치(L3)를 슬레이브로 사용하여 스캔 이동을 수행한다.

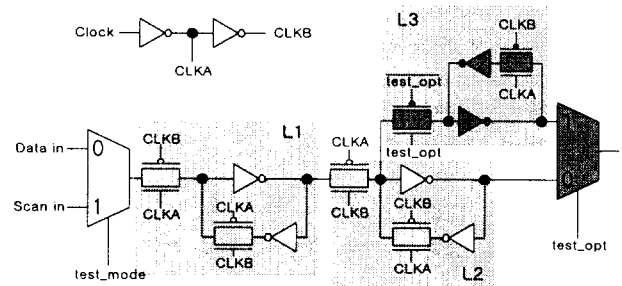


그림 3 새로운 스캔 플립-플롭  
Fig. 3 New scan flip-flop

기능적 지정 방법 역시 이 스캔 플립-플롭을 사용하여 적용할 수 있다. 만약,  $test\_opt$  신호가 항상 논리 '0'으로 조정된다면, 표준 스캔 플립-플롭과 마지막에 추가된 멀티플렉서에서의 지연 시간 외에는 다를 것이 전혀 없다. 마지막으로 특별한 모드가 추가되었는데, 이를 클럭 모드라고 부르기로 한다. 즉,  $test\_mode$ 는 논리 '0'의 값을 그리고  $test\_opt$ 은 '1' 값을 가질 때인데, 이 때는  $test\_opt$  신호가 보통의 클럭 신호와 같은 역할을 한다. 이에 관해서는 4장에서 좀 더 자세히 다루기로 한다. 표 2는 이와 같은 모드들을 제어 신호인  $test\_mode$ 와  $test\_opt$ 에 따라 구분지어 보여준다.

표 2 동작 모드의 종류

Table 2 Types of operational modes

test_mode	test_opt	동작 모드
0	0	정상 동작 모드
0	1	클럭 모드
1	0	L2-스캔 이동 모드
1	1	L3-스캔 이동 모드

5. 테스트 알고리즘

순차 회로에서 지연 고장의 테스트를 행하는 방법은 기능적 지정 방법과 역기능적 지정 방법에 따라 차이가 난다. 기능적 지정 방법을 사용할 경우에는, test\_opt 신호가 항상 '0' 값으로 고정된 상태에서 우선 L2에 첫 번째 테스트 패턴을 가져다 놓는다. 이제, 입력 값이 주출력까지 뻗어나갈 수 있을 만큼의 충분한 시간을 줄 수 있는 느린 클럭을 가하고, 연속적으로 보통 클럭을 한 주기만큼 가한다. 결과적으로, 이에 대한 응답 패턴이 스캔 플립-플롭에 저장되고, 이들은 스캔 출력을 통하여 모두 관찰 가능하다.

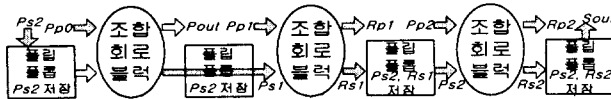


그림 4 역기능적 지정 방법

Fig. 4 Reverse functional justification method

역기능적 지정 방법은 그림 4에 나타나 있다. 그림에서 회색으로 칠해진 화살표는 필요한 데이터의 흐름을 나타낸다. 처음에, L3-스캔 이동 모드에서 Ps2가 L3에 저장된다. 첫 번째 테스트 패턴을 생성하기 위해 Pp0이 주입력단에 준비되고, 정상 동작 모드에서 느린 클럭이 가해지면, Ps1이 L1에 도달하게 된다. 느린 클럭이 한 주기 더 가해지는 동시에 주입력단에는 Pp1이 활성화되면 회로는 초기화 되게 되고, 이제 두 번째 테스트 패턴을 가하기만 하면 된다. Ps2가 L3에 저장되어 있기 때문에, test\_opt 신호는 '1' 값을 가져야 한다. 그러나, 만약 test\_opt 신호와 클럭 신호가 동시에 '1' 값을 갖게 된다면 L3은 마지막 테스트 과정 동안 Ps2를 유지할 수 없게 된다. 따라서 본래의 클럭 신호는 '0' 값으로 유지되고, 대신에 test\_opt이 보통 클럭과 같은 역할을 하게 된다. 이를 위해, test\_opt 신호가 '1' 값을 유지하는 시간은 보통 클럭 주기와 정확히 같아야 한다. 게다가, L1은 test\_opt 신호가 다시 '0' 값으로 돌아오는 동시에 포착 모드로 가야하는데, 이것은 원하지 않는 출력값에 의해 L1에 저장된 Rs2 값이 영향을 받지 않게 하기 위해서이다. 이러한 이유 때문에, test\_opt이 논리 '0'으로 돌아올 때, 클럭은 '1' 값을 갖도록 되어야 한다. 결과적으로 이러한 테스트 과정 후에 Ps2는 스캔 플립-플롭에 저장되어 있으므로 스캔 출력을 확인하여 신호 천이가 일어났는지를 확인할 수 있다. 시간에 따른 전체적인 테스트 순서는 그림 5에 나타나 있다.

효과적인 테스트를 수행하기 위해 위의 두 방법, 즉 기능적 지정 방법과 역기능적 지정 방법을 같이 사용함으로써 새로운 테스트 알고리즘을 개발할 수 있다. 두 가지 방법을 동

시에 이용하므로 새로운 테스트 알고리즘을 이용하면 지연 고장을 위한 테스트 패턴을 보다 많은 경로에 대해 생성해 낼 수 있다.

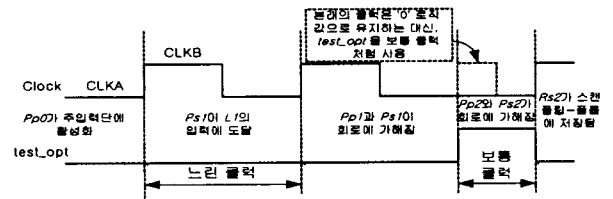


그림 5 테스트 순서

Fig. 5 Timing diagram of testing

6. 결과

표 3은 순차 벤치마크 회로에 대해 새로운 테스트 알고리즘을 적용하여 얻은 경성 경로 지연 고장에 대한 결과이다. 괄호 사이의 두 숫자는 전체 경로 수에 대해 지연 고장 검출 가능한 경로 수를 의미한다. 전체 경로 수가 5000개 이하인 회로들에 대해서는 전체 경로에 대해 실험하였고, 5000개를 초과하는 회로들에 대해서는 5000개의 임의의 경로에 대해 분석하였다. 경성 경로 지연 고장 테스트는 서론에서 살펴본 것과 같이 다른 경로에 독립적으로 고장 검출을 할 수 있는 장점을 가지고 있다. 결과를 보면 알 수 있듯이 새로운 테스트 알고리즘을 사용했을 때, 지연 고장에 대한 검출률이 약 10 퍼센트 정도 향상된 것을 알 수 있다. 비록 여전히 전체 경로의 반 정도가 고장 검출 불가능하지만, 기능적 지정 방법에서 검출하지 못한 경로들을 검출하기 위해 역기능적 지정 방법에서 검출 가능한 모든 경로들에 대해 테스트를 수행할 필요는 없다. 오히려, 역기능적 지정 방법만으로 검출 가능한 경로들을 알 수 있으므로, 이들에 해당하는 테스트만 수행하여 시간적 낭비를 줄일 수 있다.

표 3 기능적 지정 방법과 새로운 알고리즘을 이용한 결과(경성)

Table 3 The result of testable paths using functional justification and new algorithm (robust)

벤치마크 회로	기능적 지정 방법 [%] (테스트 가능 경로 / 전체 경로)	새로운 알고리즘 [%] (테스트 가능 경로 / 전체 경로)
S208	27.59 (80 / 290)	46.90 (136 / 290)
S382	19.25 (154 / 800)	30.88 (247 / 800)
S420	28.18 (208 / 738)	44.72 (330 / 738)
S713	61.32 (3066 / 5000)	69.2 (3460 / 5000)
S838	31.47 (635 / 2018)	44.20 (892 / 2018)
S1488	42.06 (2103 / 5000)	49.84 (2492 / 5000)
S1494	41.12 (2056 / 5000)	49.44 (2472 / 5000)
S9234	47.08 (2354 / 5000)	49.58 (2479 / 5000)