Signal Integrity 를 고려한 SoC 연결선 테스트용 다중 천이 패턴 생성기

A New MT(Multiple Transition) Pattern Generator for Signal Integrity Test on SoC Interconnects

김용준, 양명훈, 박영규, 이대열, 윤현준, 강성호
연세대학교 전기전자공학과
{yjkim, mhyang, hipyk, eto10, hjyoon}@soc.yonsei.ac.kr, shkang@yonsei.ac.kr

Abstract

In this paper, we propose a novel test pattern generation method and hardware architecture for testing signal integrity on SoC interconnects. Using this method, effective test patterns are generated with short test time and low hardware overhead.

I. 서론

SoC와 같은 초 고집적 반도체의 경우 기존의 회로에서는 고려하지 않던 다양한 종류의 고장이 발생한다. 특히 연결선 테스트의 경우, 보드 수준에서는 IEEE 1149.1 std.를 이용하여 별도 테스트 가능하지만, SoC의 경우 개별 코어를 IP의 형태로 입수하므로 연결선 역시 최종 SoC 수준에서 테스트되어야 하므로 그 중요성이 더욱 크다. SoC의 연결선 테스트는 IEEE 1500 std.를 기반으로 하여 수행이 가능하지만, 나노 공정과 같은 기술하에서 제작된 SoC의 경우 보드 수준에서 고려하지 않던 cross coupling 현상에 의해 noise와 delay 고장이 더욱 빈번하게 발생하므로 이를 테스트하기 위한 새로운 방안이 필요하다. 다중천이(MT: Multiple Transition) 테스트 패턴은 signal integrity를 테스트하기 위한 효과적인 방법을 제안하고 있으나, 이를 위해서는 여전히 긴 테스트 적용 시간 및 큰 하드웨어 오버헤드를 감수해야 한다는 단점이 있다. 본 논문에서는 내장형 자체 테스트(BIST: Built-In Self Test)가 가능한 다중천이 패턴 생성기를 제안한다. 이는 짧은 테스트 시간 및 적은 하드웨어 오버헤드를 통한 효과적인 signal integrity 테스트를 가능하게 한다.

II. 본론

MT 테스트 패턴은 연결선 테스트를 수행하기 위한 최대한의 조건을 인가함으로써, 테스트 성능을 극대화한 알고리즘이다. 이때 최대한의 조건이라 함은 다수의 연결선 중 단일하게 선택된 victim line에 대해 다수개의 aggressor line이 가할 수 있는 다양한 입력을 모두 고려한 것을 의미한다. 이를 기반으로 하여 효과적인 테스트가 가능하지만, [1]에서 제안된 방안에 의한 MT 패턴 생성기는 비효율적인 패턴을 생성한다. 즉 victim line은 한 테스트 주기에서 단일하게 선택되지만, 실제로는 모든 연결선을 victim line으로 한번씩 선택하여야 하므로 테스트 패턴이 그림 1과 같이 중복되는 현상이 발생한다.

		0XX→0XX				0XX→1XX				1XX→1XX				1XX→0XX			
1	Vict. line	0→0	0→0	0→0	0→0	0→1	0→1	0→1	0→1	1→1	1→1	1→1	1→1	1→0	1→0	1→0	1→0
	Aggr. line	0→1	0→1	1→0	1→0	0→1	0→1	1→0	1→0	0→1	0→1	1→0	1→0	0→1	0→1	1→0	1→0
	Aggr. line	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0
	Line set	X0X→X0X				X0X→X1X				X1X→X1X				X1X→X0X			
2	Aggr. line	0→1	0→1	1→0	1→0	0→1	0→1	1→0	1→0	0→1	0→1	1→0	1→0	0→1	0→1	1→0	1→0
	Vict. line	0→0	0→0	0→0	0→0	0→1	0→1	0→1	0→1	1→1	1→1	1→1	1→1	1→0	1→0	1→0	1→0
	Aggr. line	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0
	Line set	XX0→XX0				XX0→XX1				XX1→XX1				XX1→XX0			
3	Aggr. line	0→1	0→1	1→0	1→0	0→1	0→1	1→0	1→0	0→1	0→1	1→0	1→0	0→1	0→1	1→0	1→0
	Aggr. line	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0
	Vict. line	0→0	0→0	0→0	0→0	0→1	0→1	0→1	0→1	1→1	1→1	1→1	1→1	1→0	1→0	1→0	1→0

그림 1. 모든 victim line을 고려한 MT 테스트 패턴

본 논문에서는 이를 보완하기 위해 그림 1에서 중복된 부분을 제거한 MT 테스트 패턴을 제안한다. MT 테스트 패턴은 초기값에서 특정 비트만 천이를 발생시키는 천이값으로의 변화가 생기면서 생성되므로, 이를 이용하여 그림 2와 같은 새로운 MT 패턴을 생성하는 것이 가능하다.

	0→1	0→1	0→1	0→1	1→0	1→0	1→0	1→0
	0→1	0→1	1→0	1→0	0→1	0→1	1→0	1→0
	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0
283	1→1	1→1	1→1	1→1	0→0	0→0	0→0	0→0
	1→0	1→0	0→1	0→1	1→0	1→0	0→1	0→1
	1→0	0→1	1→0	0→1	1→0	0→1	1→0	0→1
183	1→0	1→0	1→0	1→0	0→1	0→1	0→1	0→1
	0→0	0→0	1→1	1→1	0→0	0→0	1→1	1→1
	0→1	1→0	0→1	1→0	0→1	1→0	0→1	1→0
182	0→1	0→1	0→1	0→1	1→0	1→0	1→0	1→0
	0→1	0→1	1→0	1→0	0→1	0→1	1→0	1→0
	1→1	0→0	1→1	0→0	1→1	0→0	1→1	0→0

000	0→1→1→0→1	100	1→0→0→1→0
	0→1→0→0→1		0→1→0→0→1
	0→1→0→1→1		0→1→0→1→1
001	0→1→1→0→1	101	1→0→0→1→0
	0→1→0→0→1		0→1→0→0→1
	1→0→1→0→0		1→0→1→0→0
010	0→1→1→0→1	110	1→0→0→1→0
	1→0→1→1→0		1→0→1→1→0
	0→1→0→1→1		0→1→0→1→1
011	0→1→1→0→1	111	1→0→0→1→0
	1→0→1→1→0		1→0→1→1→0
	1→0→1→0→0		1→0→1→0→0

그림 2. 제안하는 MT 테스트 패턴

그림 2의 왼쪽 그림은 그림 1에서 표시된 중복 패턴을 제거한 후, 모든 패턴에 대해 새로운 순서로 재구성한 것이다. 또한 오른쪽 그림은 왼쪽 그림의 세로열에 대해 한 패턴의 천이값이 다음 패턴의 초기값으로 설정됨에 의해 추가적인 패턴의 감소를 가능하게 한 결과다. 따라서 본 논문에서 제안하는 최종적인 MT 테스트 패턴은 그림 2의 오른쪽과 같다.

본 논문은 MT 테스트 패턴을 자체 생성하는 패턴 생성기를 내장한 내장형 자체 테스트를 목적으로 한다. 그림 2의 MT 테스트 패턴을 생성하기 위한 하드웨어

구조는 초기상태 카운터와 쉬프트 레지스터를 통해 비교적 간단하게 구현이 가능하다. k 비트 초기상태 카운터는 제안하는 MT 테스트 패턴의 초기값을 생성하는 부분이며, 각 테스트 패턴은 그림 2와 같이 천이의 개수 및 순서가 달라지므로, 이를 제어하기 위한 k+2 비트 쉬프트 레지스터가 필요하다. 각 비트의 천이는 플립플롭의 출력을 feedback하여 사용하며, 예정된 모든 천이가 발생하고 나면 b_{k+1} 비트 값이 갱신되어 k 비트 초기값 카운터 역시 갱신되면서 새로운 초기값을 생성한다.

표 1은 k+2 비트 쉬프트 레지스터의 각 상태별 패턴 생성기 기능을 나타낸다. 이때 쉬프트 레지스터의 초기값은 0111…111로 설정한다. 결국 쉬프트 레지스터의 기능은 크게 세 가지인데 이는 초기값 생성을 위한 신호 생성, 모든 비트에 대한 천이값 생성, 그리고 특정 비트를 제외한 나머지 비트에 대한 천이값 생성이다.

표 1. Shift register의 상태별 기능

쉬프트 레지스터의 비트별 상태			기능
b_{k+1}	b_{k+0}	b_{k+1}~ b_0	
0	0	111...111	k 비트 카운터의 초기값 갱신
1	0	111...111	모든 비트의 천이생성
1	1	011...111~111...110	0으로 설정된 비트를 제외한 모든 비트의 천이생성

이는 쉬프트 레지스터의 최상위 두 비트를 이용하여 제어가 가능하며, 이후 하위 k 비트에 의해 생성할 패턴의 천이를 조절한다. 따라서 제안하는 MT 테스트 패턴은 제안된 하드웨어를 이용하여 효과적으로 생성된다.

III. 실험결과

제안된 MT 테스트 패턴 생성기의 성능을 검증하기 위해 하드웨어 오버헤드 및 패턴 생성 시간을 비교한다. 표 2는 하드웨어 오버헤드를 비교한 것이다. Boundary scan은 별도의 패턴 생성기를 사용하지 않고 IEEE 1149.1 구조를 이용한 경우이며, [1]은 기존의 MT 패턴 생성 방안을 의미한다. 표에서 n은 전체 연결선의 개수이며, k는 이중 동일하게 그룹핑된 연결선의 개수이다. 기존의 방법들의 경우, 모두 경계주사 구조를 기본으로 하여 모든 경계주사 셀에 하드웨어가 추가되므로 이에 대한 오버헤드는 상당하다. 표에서는 제한된 개수의 연결선 만을 가정했지만, 실제로 연결선의 수는 엄청나게 증가할 것이므로 그 차이는 표에 표시된 것보다 훨씬 클 것으로 예상된다.

표 2. 하드웨어 오버헤드 비교

연결선 수		하드웨어 오버헤드(NAND)	
n	k	[1]	제안된 방안
기본 셀		483	–
8	2	3864	1261
	3	3864	1808
	4	3864	2361
	5	3864	2931
16	2	7728	1261
	3	7728	1808
	4	7728	2361
	5	7728	2931
32	2	15456	1261
	3	15456	1808
	4	15456	2361
	5	15456	2931

또한 표 3의 경우 테스트 패턴이 생성되는 시간은 기존의 방안은 경계주사 구조를 기반으로 하므로, n 및 k의 값에 따라 지수적으로 증가한다. 그러나 제안하는 방안은 n의 값에 상관없이 테스트 패턴을 생성하므로, 패턴 생성 시간 역시 큰 차이를 나타냄을 확인할 수 있다.

표 3. MT 패턴 생성 시간 비교

연결선 수		MT 패턴 생성 시간(cycle)	
n	k	[1]	제안된 방안
		$(2k+1)(2n+8k)2^{2k}$	$(k+2)2^k$
8	2	2560	16
	3	17920	40
	4	110592	96
	5	630784	224
16	2	3840	16
	3	25088	40
	4	147456	96
	5	811008	224
32	2	6400	16
	3	39424	40
	4	221184	96
	5	1171456	224

IV. 결론

초고집적 반도체의 테스팅 환경에서 연결선상의 signal integrity를 적합한 수준으로 테스트하는 것은 SoC 시대에서 반드시 충족해야 할 사항이다. 이를 위해 본 논문에서 제안한 MT 테스트 패턴 및 하드웨어 구조는 기존의 방식에 비해 훨씬 적은 하드웨어 오버헤드를 통해 더욱 짧은 시간에 테스트 패턴을 생성하므로 매우 효과적인 연결선 테스트 방안을 제공할 것으로 기대된다.

참고문헌

[1] M. H. Tehranipour, et al., “Multiple Transition Model and Enhanced Boundary Scan Architecture to Test Interconnects for Signal Integrity,” in Proc. *ICCD ’03*, pp. 554-559, 2003.

[2] Y. Kim, et al, “An Effective Test pattern Generation for Signal Integrity,” in Proc. *ATS ’06*, pp. 279-284, 2006.