韓國 OA學會 論文誌
第3卷 第2號, 1998. 6.

98-3-2-2-5

고장 진단 생성 시스템 설계에 관한 연구

김 철 운*

A Study on the Generation System Design for Fault Detect

Cheol-Woon Kim*

요 약

본 논문에서는 다단 논리회로의 고장을 완벽하게 검출할 수 있는 테스트 패턴 생성기를 설계하였다. 이 테스트 기법은 테스트 패턴 생성 논리회로를 사용하여 생성하였다. 생성된 테스트 패턴은 기존의 전체 테스트 방법에 비해 패턴을 크게 감소시켰다. 이 테스트 패턴 생성기는 다단 논리회로에서의 모든 고장을 검출할 것으로 본다. 여러 가지 I.C 테스트 방법 중에서 어떤 방법을 선택할 것인지는 고장검출 속도에 영향을 준다. 가장 중요한 것은 생산단가이며 설계된 테스트 패턴 생성기는 저가형이다.

Abstract

In this paper I designed test pattern generater which will be completely detected the faults of multi-stage Logic Circuit. I generated this pattern using the test pattern generation Logic Circuit. The generated test patterns compared with the exhausted testing was decreased pattern. This test pattern generater will detect the all single stuck-at faults in the multi-stage Logic Circuit. The choice of which of the many I.C testing methods to use can have a effect on the success or failure of the fault detected. One of the most important considerations is cost and designed test pattern generater is very low cost type.

---

* 전남과학대학 사무자동화과 조교수
논문접수 : 98.4.20 심사완료 : 98.5.22

1. 서 론

디지탈 회로에서 데이터를 처리할 때 고장으로 인하여 데이터를 처리하지 못한다면 심각한 문제가 된다. 그러므로 컴퓨터의 정상적인 시스템을 운영하기 위하여는 회로의 효과적인 고장진단방법은 대단히 중요하다. 최근 VLSI 시스템 설계기술 및 컴퓨터의 성능은 빠르게 발전하고 있다.

이와 같은 논리장치를 설계할때 중요한 문제는 제작된 회로가 설계된대로 제 기능을 수행하는가를 테스트 확인하는 일이다.[1]

테스트 패턴은 순서회로 및 조합회로에 다같이 적용할 수 있으면서 적당한 검사 패턴을 생성할 수가 있어야 하며 필요한 검사패턴의 수와 적용을 용이하게 하기 위해서는 회로검사에 따른 확실한 신뢰성을 얻도록 재설계가 요구되기도 한다.[2] 전반적인 반도체 시장에 영향을 미치는 많은 요소들은 증가된 시스템 집적도와 전력소비의 감소 및 신뢰성을 포함한다. 특히 이러한 집적화 추세는 1990년대에 가속화되었고 집적도가 높아질 수록 검사 방법이 중점적으로 연구되고 있다. 왜냐하면 회로의 테스트 비용은 테스트시간과 관계가 있고 테스트 시간은 칩의 단가와 직접적으로 관련되므로 회로 테스트는 신속하게 이루어져야 단가를 낮출 수 있기 때문이다.[3][4] 본 논문에서는 테스트 대상회로를 이용하여 고장을 빠르고 완전하게 검출할 수 있는 테스트 패턴 생성기를 개발하고자 한다.

2. 각종 테스트 방법

회로의 테스트는 그림 1과 같이 입력에 테스트 패턴을 인가하고 출력에서 그 결과를 관측함을 의미한다. 조합 논리 회로에 존재하는 고장 검출의 방법은 두 가지가 있다. 첫째는 회로의 고장에 의한 영향이 출력에 나타나도록 입력에 인가할 테스트 패턴을 결정하는 방법이고 둘째는 회로의 기능을 함수로 표현하고 함수의 처리에 의해 입력의 테스트 패턴을 결정하는 방법이다.[5] 이를 위하여 회로의 고장 테스트 패턴 생성은 진리표를 기본으로 한 경로 활성화 방법, 회로의 분할 방법등이 있다. 이들 방법 중에서 범용적이고 대표적인 방법으로는 부회로의 분할, D-알고리즘등이 있다.[5]

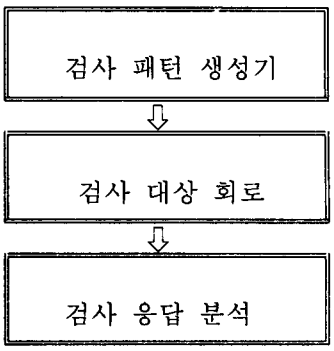

그림 1 회로의 테스트 과정

2.1 테스트 대상 회로의 분할

n개의 입력과 m개의 다출력을 갖는 테스트 대상 회로는 2n개의 입력 조합으로 생성되는 전체 테스트 패턴으로 m개의 다출력단을 각각 정확하게 테스트해야 한다. 이 때 n개의 입력을 갖는 회로의 전체 테스트 비용은 식 (1)과 같이 테스트 패턴의 수로 정해진 tl(test length)이다.[6]

$$COST(exhaustivetl) = 2^n \quad (1)$$

입력수 n이 커지면 전체 테스트의 길이는 지수적으로 증가하여 테스트 시간이 많이 걸리게 되므로 테스트 대상 회로를 분할하여 각각의 분할된 부 회로를 전체 테스트해야 한다. 테스트 대상 회로를 의사-전체 테스트하기 위해서는 회로 분할 방법을 사용하여 출력수 m개의 부 회로로 분할되는 각 부 회로 Seg.i를 전체 테스트할 수 있다. 테스트 대상 회로의 각 출력으로부터 모든 입력으로 역추적하여 TODC(Tracing Output Dependency Cones)를 형성하면 각 분할의 Seg.i의 입력 수는

$$n(Seg.i) = Ii + Ci \quad (2)$$

이다. 여기서, Ii : Seg.i 회로의 입력 수, Ci : 이웃하는 Seg.i로부터 한계선(cutline)을 통하여 들어오는 회로의 입력수이다.

이 때 각 분할의 한계선 상에서 전달되는 정확한 Ci의 입력 수를 알아야 하며 테스트 비용은 식 (3)과 같이 테스트 패턴의 전체 수로 정해진 tl이다.

COST(pseudo-exhaustive)

$$\sum_{allSeg.i}^{m} 2^{(n(Seg.i)} = \sum_{i=1}^{m} 2^{(Ii+Ci)} \quad (3)$$

테스트 대상 회로 분할의 최적 조건은 결국 테스트 비용 절감으로 식 (4)와 같다.

COST(pseudo-exhaustive*te*)

$$\min_{(Ii+Ci)} \sum_{i=1}^{m} 2^{(Ii+Ci)} \quad (4)$$

2.2 D-알고리즘

D-알고리즘은 1차원 활성화 방식의 결함을 극복하고 복수의 경로를 동시에 활성화하기 위해서 Roth가 착안한 방식으로서 대단위 조합 논리 회로의 정의된 s-a-고장을 테스트하는데 있어 비교적 효율성이 있었다.[7]

이 방식은 IBM에서 채택하여 표준 테스트 패턴 생성으로 이용하였으며 기본 개념은 고장이 존해하는 위치부터 출력단까지의 경로를 선택한다. 테스트 대상 회로에 존재하는 s-a-고장의 오차 신호를 회로의 출력단자에 전달시킬 수 있는 경로를 활성화 경로라 한다. 소자의 고장 진단에서 오차 신호는 입, 출력의 라인이 논리 상태 1 또는 0으로 고착되어버린 것으로 s-a-1, s-a-0로 표현한다. s-a-1은 정상 오차 0에서 오차 신호 1을 발생한 D 고장이고 s-a-0은 정상 신호 1에서 오차 신호 0을 발생한 D 고장이다. 그리고 각 논리 게이트이 입력 할당은 진리표를 간소화한 특이 커버를 적용하여 게이트의 고장 단자에서 출력 단자에 이르는 경로를 활성화한다. 다음에는 고장이 존재하는 위치부터 출력단까지의 경로를 통하여 진행하면서 라인 상에 논리값을 할당한다. 이는 고장이 있는 라인상의 신호값을 출력단까지 전달하려는 것으로서 이 과정을 순방향 구동(forward dirve : FD)이라한다. 그리고 PI의 결정은 모든 입력의 테스트 패턴 생성이 되며 이 과정을 역방향 구동(backward drive : BD)이라 한다. D-알고리즘의 특성은 임의의 고장을 검출할 수 있는 입력의 테스트 패턴이 있으면 전체 입력의 테스트 패턴을 생성할 수가 있으므로 한 개고장을 검출하는 입력 패턴으로 다른 고장도 검출이 된다. 그리고 회로 내부에 EXOR 게이트가 있으면 알고리즘의 실행이 복잡해진다. 사용 논리값은 0, 1, X, 1/0(D), 0/1(D)의 5개이고 D-알고리즘의 실행단계는 다음과 같다.

단계 1. 논리 소자의 고장 진단에서 입력에 D, D'의 오차신호를 할당하고, 나머지 입력은 X를 할당하는 기본 D-큐브 (primitive D-cube of failure : pdcf)를 실행한다.

단계 2. 오차 신호를 회로의 출력 단자에 전달시키는 경로를 작성하는 D-드라이브 절차를 실행한다.

단계 3. 오차 신호를 전달시키는 조건을 나타내는 전달 D-큐브 (propagation D-cube : pdc)를 실행한다.

단계 4. 단계 1의 기본 D-큐브와 STEP 3의 전달 D-큐브를 교집합을 하는 D-교차를 실행한다.

단계 5. 단계 4의 D-교차에서 가 나오면 오차 신호를 전달할 수 없는 조건이 되어 테스트 패턴을 생성하지 못한다.

단계 6. 경로 활성화가 이루어진 게이트에 일반 동작을 적용한다.

단계 7. D와 D'의 신호 전달을 위한 라인 정당화를 실행한다.

단계 8. 각 논리 게이트에 대한 큐브를 구하고 고장이 발생한 게이트의 pdcf를 구하여 전체 입력의 테스트 패턴을 생성한다.

2.3 CAMELOT 알고리즘

Bennetts는 논리회로의 testability (TY)를 분석하기 위하여 CAMELOT (Computr Aided Measure for Logic Testability)을 발표하여 TY가 낮은 회로에 대하여 설계 초기에 수정을 하는 방법을 착안하였다.[8]

테스트 회로의 입력값을 setting된 논리값으로 할당해야 한다. 이러한 실행을 할 수 있는 용역성을

이 노드의 controllability (CY)라 하며 CY의 실행으로 테스트 대상 회로의 응답은 출력단까지 유지되기 위하여 활성화 경로를 필요로 하게된다. 또한 할당되지 아니한 입력단을 이용하여 다른 노드상에 setting된 논리값을 출력단에서 관측하기 위하여 논리값을 할당해야 한다. 각 노드에 대한 이러한 과정의 완성도를 observability (OY)라 한다. 그러므로 각 노드의 TY는 CY와 OY에 대한 함수임을 알 수 있다. CY는 PI에 가까운 노드일수록 높고, PI에서 먼 노드일수록 낮아지며 PI에서 최대치 "1"을 갖고 출력단에서 최소치 "0"을 갖는다. OY는 반대로 출력단에 가까운 노드일수록 높고, 출력단에서 먼 노드일수록 낮아니지며 출력단 에서 최대치 "1"을 갖고 PI에서 최소치 "0"을 갖는다. 그러므로 CY와 OY는 반비례 관계를 가지며 CY가 높은 노드는 낮은 OY를 가지며, OY가 높은 노드는 낮은 CY를 갖는다.

2.4 의사 전체 테스트

전체 테스트는 고장모델과 고장 시뮬레이션이 제거된 테스트 방식이다. 회로기능의 다양화 및 복잡도의 상승에 따라 입력수가 증가하게 된다. 전체 테스트는 입력수가 n일때 테스트 패턴의 수는 지수적으로 증가하여 2n으로 생성되므로 2진 계수기를 테스트 패턴 생성기로 사용할 수 있지만 다입력의 회로에서는 비실용적인 테스트 방법이 된다.

조합적 고장모델의 표현을 위한 회로의 경우에 전체 테스트가 필요하나 테스트 패턴의 수와 테스트 시간이 크므로 제한적 고장모델을 채택할 필요성이 있게 된다. 이 테스트방식의 장점은 조합회로를 순서회로로 바꾸지 않고 검출 가능한 모든 조합적 고장을 검출할 수 있는 고장적용 범위가 크다는 점이다. 만약에 테스트 패턴의 생성순서가 중요하지 않다면 모든 상태를 자동적으로 순회하는 LFSR (linear feedback shift register)의 사용이 보다 효과적이 된다. 회로의 주출력들이 모든 주입력에 의존하지 않는 회로는 회로 전체를 테스트할 필요가 없다. 그러므로 하나의 주 출력이 의존하는 주 입력으로 구성된 회로의 부분만을 테스트하는 패턴만을 생성하므로써 테스트 패턴을 감소시킬수 있다. 이 테스트 방법은 고장모델에 근거를 두고 주 출력이 의존하는 주입력의 조합이 고장에 의해 증가하지 않게 되므로 회로 테스트는 주 출력의 기능에 따른 부 회로를 중심으로 실행이 된다. 그리고 이 테스트 방법은 회로의 자세한 분석없이도 단일 고장의 적용 범위를 보장할 수 있다. 입력수 n이 커지면 전체 테스트의 패턴수는 지수적으로 증가하여 테스트 시간이 많이 걸리게 되므로 회로를 분할하여 의사-전체 테스트를 한다.

3. 고장 검출 알고리즘

3.1 테스트 대상 회로의 입력

테스트 대상 회로의 데이터입력으로 0과 1을 사용할때 어드레스비트의 setting에 따라 입력값을 변화없이 출력으로 내보내므로 데이터 입력값과 출력값에 따라 연결된 경로상의 모든 고장들은 표 1의 패턴이 된다. 그림 2는 사용된 테스트 대상 회로이다.

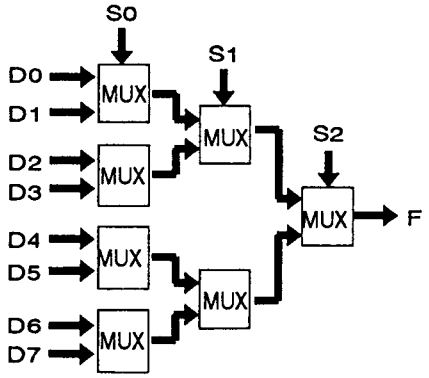

그림 2. 테스트 대상 회로

표 1. 테스트 대상 회로의 입력 테스트 패턴.

S_2	S_1	S_0	D_1	D_2	D_3	D_4	D_4	D_5	D_6	D_7	F
0	0	0	1,0	X	X	X	X	X	X	X	1,0
0	0	1	X	1,0	X	X	X	X	X	X	1,0
0	1	0	X	X	1,0	X	X	X	X	X	1,0
0	1	1	X	X	X	1,0	X	X	X	X	1,0
1	0	0	X	X	X	X	1,0	X	X	X	1,0
1	0	1	X	X	X	X	X	1,0	X	X	1,0
1	1	0	X	X	X	X	X	X	1,0	X	1,0
1	1	1	X	X	X	X	X	X	X	1,0	1,0

3.2 선택 라인의 입력

다단 회로에서는 게이트에서의 어드레스 비트들이 회로의 다른 레벨들이기 때문에 동등하게 취급될 수는 없다. 회로의 1단 고장 테스트에는 D_0 부터 D_7 까지 0, 1, X, X, X, X, X, X의 패턴을 적용하면 된다. 1단에도 어드레스 라인이 있으므로 패턴의 d 비트들은 1로 setting된다. 2단에 대한 s-a-1 테스트를 하기 위해서는 라인을 0으로 setting해야 하고 3단 라인을 1로 setting해야 한다. 이것은 2단을 통하는 경로를 활성화 하기 위함이다. 즉, D_0=0, D_2=1, S_0=0로 setting하거나 D_1=0, D_3=1, S_0=1로 setting한다. 같은 방법으로 2단에서의 모든 어드레스 입력들은 두 가지 가능한 s-a-1 테스트 패턴들을 가진다. 1단에서의 d비트와 대조적으로 단지 d/2와 같은 라인이 있기 때문에 d/2비트가 바꾸어 진다. 이 비트들 각각에 대해 두가지 가능한 선택이 있으며 3단에서는 4개의 선택이 있다. 그리고 d/4비트들이 setting된다.

임의의 i단에서는 각 고장들에 대해 2^i개의 선택이 있고 d/2i비트가 1로 setting되어야 한다. 1단에서는 S0가 s-a-0고장이 발생하면 S_0=1일 때 D_0, D_2, D_4, D_6가 다음 단으로 활성되고 D_1, D_3, D_5, D_7은 선택되지 않게되며, S_1=1일때 D_1, D_5가 다음 단으로 활성되고 D_3, D_7은 선택되지 않는다. S_2=1일 때 D_3가 출력으로 얻어지고 D_7은 선택되지 않는다. 게이트의 어드레스 입력의 s-a-1 고장 테스트를 유도해 내기 위한 4가지 기본 단계가 있는데 1단계는 테스트 할 라인을 0으로 놓고 어드레스 초기 입력 비트를 setting하고 2단계는 현재의 단과 다음 단의 어드레스 비트를 그 라인의 1차 출력으로 하면 된다.

표 2. 두 개의 가능한 패턴

S_2	S_1	S_0 = 0		S_0 = 1	
pattern		0	1	0	1
0	0	D_0	D_2	D_1	D_3
0	1	D_2	D_0	D_3	D_1
1	0	D_4	D_6	D_5	D_7
0	1	D_6	D_4	D_7	D_5

표 3. 선택 라인의 입력고장 패턴

Input			s-a-1			D	s-a-0			D
S_2	S_1	S_0	S_2	S_1	S_0		S_2	S_1	S_0	
0	0	0	0	0	1	0				
0	0	1					0	0	0	1
0	1	0	0	1	1	2				
0	1	1					0	1	0	3
1	0	0	1	0	1	4				
1	0	1					1	0	0	5
1	1	0	1	1	1	6				
1	1	1					1	1	0	7
0	0	X	0	1	1	0				
0	1	X					0	0	1	2
1	0	X	1	1	1	4				
1	1	X					1	0	1	6
0	X	X	1	0	0	0				
1	X	X					0	0	0	4

고장 테스트에서 게이트의 입력으로 1을 얻기 위해 1대신에 0으로 setting시켜야 할 필요가 있을 경우에 테스트 패턴은 d개의 데이터 입력일 때 2d개의 패턴으로 고장 테스트를 할 수가 있으며, 테스트 집합은 2단계로서 어드레스의 가능한 모든 조정에 의해 선택된 데이터 비트는 1을 갖는 패턴이어야만 하고 표 2처럼 두 개의 선택들로 된다. 어드레스 입력의 고장 가능한 경우에 따라 1로 setting될 값을 나타내면 표 3과 같다. 이와 같은 고장을 테스트할 수 있는 패턴생성기는 그림 3과 같이 설계하였으며 고장을 검출할 수 있는 방법은 그림 4와 같다.

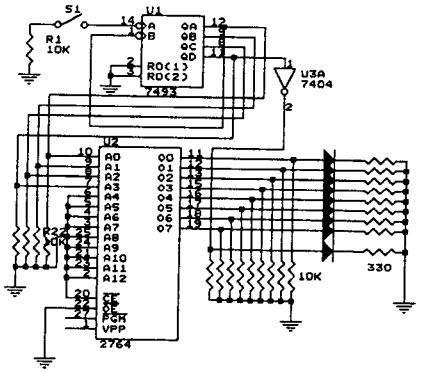

그림 3. 테스트 패턴 생성기의 설계

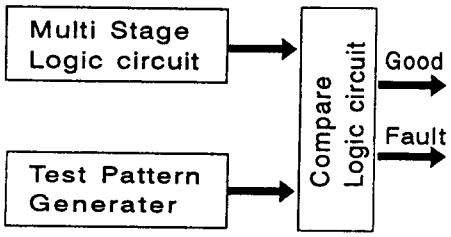

그림 4. 고장 검출 방법

4. 결 론

본 논문에서는 다단 논리회로의 고장을 검출하기 위한 테스트 패턴을 제안하고 패턴을 생성할 수 있는 회로를 구현하였다.

이 테스트패턴 생성기를 비교회로에 적용하면 다단회로에서 발생할 수 있는 모든 고장은 완전하게 검출할 수 있으며, 시뮬레이션 결과 테스트할 수 있는 패턴을 정확하게 생성하였다.

본 논문에서 사용된 회로는 입력의 총수가 데이터 라인 8개와 선택 라인 3개인 회로인데 제안된 테스트 패턴을 수행하는데에는 LED 9개만으로 충분한 회로를 설계하였다. 기존의 알고리즘적 방법에 비하여 하드웨어를 사용하므로서 테스트 시간이 매우 단축되므로 기술적인 면에서 적극 응용되어야 하겠다.

References

[1] Chun-Yeh Liu and Kewal K. Saluja, "Built-In Self Test Design of Large PLA," 24th ACM/IEEE Design Automation Conference, 1987, pp. 385-391.

[2] D.M.Miller, J.C.Muzio,"Spectral Fault Signatures for Single Stuck-at-fault in Combinational Network," IEEE, Trans. on Computers, vol.c-33, no.8, Aug. 1984, pp. 766-769.

[3] D.T.Tang and L.S.Woo, "Exhaustive test pattern generation with constant weight vectors," IEEE Trans. on Computers, vol.C-32, Dec. 1983, pp,1145-1150.

[4] I.Shperling, E.J.McCluskey, "Circuit Segmentation for Pseudo-exhaustive Testing," CRC Tech. Report, no. 87-2, Feb. 1987.

[5] J.G. Udell, Jr., "Test Set Generation for Pseudo-exhaustive BIST," in Dig. Papers, IEEE 1996 Int,l Conf. Computer-Aided Design (Santa, Clara, CA).Nov. 11-13 1996, pp. 52-55.

[6] McCluskey, E.J., "Exhaustive and Pseudo-exhaustive Test," Built-in Test Concepts and Techniques, Tutorial, ITC 83.

[7] McCluskey, E.J.,"Verification testing A Pseudo-exhaustive test technique," IEEE Trans. on Computers, vol.c-33.no.6, June. 1994, pp. 125-129.

[8] Paul H. Bardell, Jr. Willam H. Mcanney, "Built-in Test for RAMS," IEEE Design&Test Of Computers, Aug. 1988, pp. 29-37.

● 저자소개

김 철 운

1977년 : 전남대학교 공과대학 공업교육학과 (전자공학 전공)졸업.공학사
1982년 : 전남대학교 교육대학원 공업교육학과졸업. 교육학석사
1992년 : 조선대학교 대학원 전자공학과 졸업. 공학석사
1998년 : 전남대학교 대학원 전자공학과 졸업. 공학박사
1992년 ~ 현재 : 전남과학대학 사무자동화과 조교수.
관심분야 : 디지탈 회로 설계, 전자 재료.