기술해설

경로지연고장에 대한 테스트 생성과 고장 시뮬레이션

강 성 호
(연세대학교 공과대학 전기공학과 조교수)

1. 서 론

집적도 증가에 따라 칩에 대한 테스트의 중요성이 강조되고 있다. 고질의 칩을 얻기 위하여 기존의 stuck-at 고장에 대한 테스트 외에도 지연 테스트(delay test)이 수행되고 있다. 시스템을 오동작하게 하는 제조 결함 중에서 지연 고장(delay fault)는 논리가 정상보다 늦게 반응하는 것이다. stuck-at 고장과는 달리 지연고장은 시스템의 정상 상태에서의 논리동작에는 영향을 미치지 않지만 전체 시스템의 성능을 낮게 한다. 회로내의 모든 경로(path)가 신호를 주어진 시간 명세(time specification)보다 일찍 전달한다면 지연고장이 없게 된다. 지연 시험(delay testing)[1,2,3,4,5]의 목적은 주어진 클럭 레이트(clock rate)에서 정상적으로 회로가 동작하는 것을 확인하는 것이다. 지연테스트는 신호의 변화가 회로를 통하여 주어진 시간내에 통과하느냐를 결정하기 때문에 적어도 2개의 벡터를 필요로 한다.

여기서는 최근 활발하게 연구되고 있는 지연시험에 대해서 지연고장모델, 지연테스트의 종류, 여러 자동 테스트 생성 기법, 고장시뮬레이션 방법들에 대해서 살펴보고자 한다.

2. 지연고장모델

2종류의 지연고장 모델이 개발되어 널리 쓰이고 있는데 이는 게이트 지연고장모델[6]과 경로지연고장모델[7]이다. 게이트 지연고장 모델에서는 회로내의 게이트의 입력이나 출력에 집중된 경우이다. 이 모델의 장점은 stuck-at 고장을 위한 테스트를 이용해서 지연고장을 위한 테스트를 만들 수 있다는 것이다. 즉 이를 위해서는 2벡터중의 첫째 벡터는 stuck-at 고장을 위한 테스트에 시험중인 게이트가 반대되는 값을 갖도록 설정하면 된다

그러나 이 모델은 회로내의 각 게이트에 널리 퍼져있는 작은 지연으로 구성된 결함은 고려하지 못한다.

경로 지연고장 모델은 시험중인 경로를 따라 있는 작은 지연들이 모여서 회로가 오동작하도록 되는 것을 말한다. 이 모델에 기초를 둔 테스트는 회로의 경로에 따른 집중 또는 분산된 결함을 모두 검출할 수 있다. 그러나 회로에는 상당히 많은 경로가 있어서 모든 경로를 다 시험하는 것은 거의 불가능하다. 왜냐하면 회로의 깊이가 증가함에 따라 경로의 수는 지수함수적으로 증가한다. 따라서 경로지연시험은 모든 경로중의 부분집합에 촛점을 맞추게 되는데 이의 선택이 중요하게 된다. 가장 많이 쓰이는 방법은 가장 긴 경로들을 고려하는 것이다.

회로를 설계하는데 있어서 클럭레이트를 결정하는 데는 2종류의 설계방법[8]이 있다. 최악의 경우를 생각하는 설계방법은 경로에서의 지연은 각 게이트의 최악의 지연과 각 게이트 사이의 최악의 지연의 합으로 주어지고 모든 경로가 주어진 시간보다는 짧은 최악의 지연을 가져야 한다. 하지만 어떤 경로위의 모든 게이트가 최악의 지연을 갖는 것은 매우 드물다. 따라서 이는 매우 보수적인 클럭레이터를 만들게 된다.

둘째방법은 통계적인 것으로 각 게이트가 여러 변수에 의해 지연이 어떤 분포를 갖는 것이다. 따라서 이 게이트 분포의 합은 경로의 지연분포가 된다. 경로위의 게이트들의 전형적인 지연과 이의 변이가 시스템의 클럭레이터를 결정하는데 쓰인다. 따라서 각 게이트지연은 주어진 시간내에 동작을 하지만 어떤 경로는 시스템의 시간명세를 맞추지 못할 수 있다. 어떤 방법을 사용하든지 지연시험은 제조공정에서 발생한 경로의 초과지연을 검출해야 한다.

위에 서술한 2가지 지연고장 모델 중에서 경로지연모델을 사용하면 여러가지 유리한 점이 있다. 이는 회로의 시간경계 근처에 있는 경로에 해로운 영향을 미치는 변이들을 관측할 수 있게 한다. 또 회로내의 긴 경로들의 시간고장에 대한 테스트는 실패가 생길수 있는 클럭속도

에 대한 정확한 모습을 제공하므로 속도에 따라 칩을 분류하는데 쓰일 수 있다[9]. 따라서 최근에는 게이트 고장 모델보다 경로지연 모델을 이용한 연구가 많이 수행되고 있으며 여기서도 경로지연 모델에 집중하여 살펴보고자 한다.

3. 경로지연 테스트의 종류

경로지연 고장에 대한 테스트에는 여러 종류가 있다. 첫째로 무해저드경성 (hazard free robust) 테스트(HFR)[10]는 2개의 벡터로 이루어진 테스트로서 경로에의 신호들이 dynamic hazard가 없고 지연고장 검출이 회로의 다른 부분의 지연과 무관한 것이다. 따라서 이는 지연고장 진단에 적합하고 시험중인 경로가 초과의 지연을 가지면 테스트가 실패하고 또 테스트가 실패하면 시험중인 경로가 초과의 지연을 갖는다.

2번째 종류의 테스트는 경성(robust)테스트(ROB)인데 이는 지연고장 검출이 회로의 다른 부분의 지연과 무관한 것이다. 하지만 이는 경로위의 신호들이 동적 해저드를 안 갖는 것을 보증하지는 않는다. 이 종류의 테스트는 시험중인 경로가 초과지연을 가지면 실패한다 하지만 이는 또한 다른 이유로도 실패할 수 있다. 즉 이는 회로내의 다른 경로가 초과지연을 갖는 경우이다. 따라서 회로에 대해서 테스트를 하는 경우에 진단이 필요한 경우가 아니고 단지 고장검출만이 목적이라면 경성테스트를 구하는 것이 무해저드경성테스트를 구하는 것보다 유리하다. 이는 경성테스트를 구하는 것이 더 쉽기 때문이다.

경성테스트의 요건을 갖추지 못한 테스트를 연성 (non-robust)테스트[11]라고 부른다. 이는 2개로 구분이 될 수 있는데 경성테스트를 생성하는 접근법으로부터 나온 연성테스트를 강연성(strong non robust)테스트(SNR)이라 부른다.

이는 또 근사경성(approximate robust)테스트라고도 불리는데 모든 요건이 경성테스트와 같고 단지 요건이 2개의 사이클 동안 같은 값을 가지며 해저드가 없게하는 요건이 여기서는 2개의 사이클 동안 같은 값만을 가지며 해저드가 있어도 무방한 경우이다.

Schulz 등은[12] 두번째 사이클의 경우에 경성테스트와 요구되어지는 값이 같고 첫번째 사이클의 경우 요구되어지는 것이 경로의 시작부분의 초기값으로만 결정되는 방법으로 연성테스트를 구하였는데 이를 약연성(weak non-robust)테스트(WNR)이라고 부른다[9].

또 경로토글(path toggle)테스트(TOG)[13]는 경로위의 모든 소자에서 원하는 신호변이를 갖게 하는 테스트이다.

이 5가지 종류의 테스트의 정의에 따른 요건들을 비교하면 다음의 그림 1과 같다.

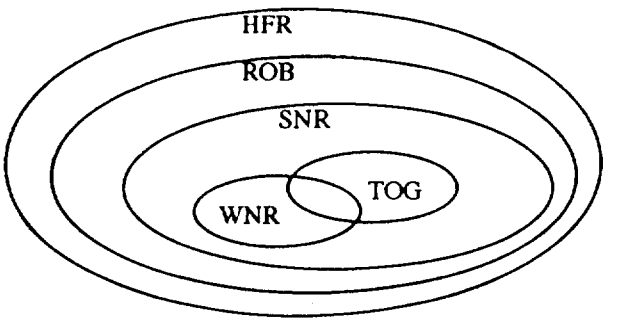

그림 1. 경로지연 테스트들의 요건집합

위의 그림에서 보면 경성테스트의 요건집합이 강연성테스트의 요건집합을 포함한다. 이는 경성테스트를 구하기 위한 요건이 더 많다는 것이며 따라서 더 구하기가 어렵다. 따라서 경성테스트는 강연성테스트보다는 고질의 테스트이며 어떤 한 경로에 따라 경성테스트가 있으면 강연성테스트도 존재한다. 그러나 강연성테스트가 있다고 해서 반드시 경성테스트가 존재하는 것은 아니다.

4. 경로지연에 대한 테스트 생성

경로지연에 대한 테스트 생성은 테스트의 종류에 따라 경로를 따라가며 off path 입력에 제한값을 주게 되며 이 제한값을 2사이클 동안 만족시키는 입력을 구하면 이것이 바로 원하는 테스트가 된다.

Lin[11]은 combinational 회로에 대한 테스트 생성을 연구하였다. 여기서는 해저드가 없는 1(S1), 해저드가 없는 0(S0), 2사이클 동안 x(xx), 2번째 사이클에만 1(U1), 2번째 사이클에만 0(U0) 등 5개의 논리값을 사용하여서 테스트를 생성하였다. 그림 2에 나타난 off path 입력을 이용해 경성테스트를 구하였다.

Transition	AND/NAND	OR/NOR
Rising	U1	S0
Falling	S1	U0

그림 2. Off path 입력

이 연구에서는 기본적인 게이트들을 대상으로 하였고 PODEM[14]을 기초로 하여 제한값들을 만족시키는 입력값을 구하였다.

Schulz 등은[12] 10개의 논리값을 사용하였다. 테스트 생성하는 기준은 역의 각 off path 입력에 제한값을 주고 이를 만족시키는 입력을 구하는 것이다. 이는 SOC-

RATES[15]에 쓰인 여러 기법들을 근거로하고 있고 이를 위해 10개의 근거값을 사용하였다. 이도 역시 combinational 회로만을 대상으로 한다. 이는 static learning, multiple backtracing 등의 기법을 이용하여 효율적으로 테스트를 생성하였다. 그러나 이들 combinational 회로에 쓰이는 생성기법은 실제로는 별 의미가 없다. 기존의 실제회로를 사용하기 위해서는 sequential회로를 다루어야 한다.

실제로 stuck-at 고장을 위한 테스트에서는 테스트를 돕기 위해 scan을 이용하여 sequential 회로를 combinational 회로로 바꾼다. 지연테스트에서는 scan을 쓰는 회로에 대해서도 2개의 cycle을 가진 sequential 회로로 여겨진다. 이를 기존의 combinational 회로에 대한 기법을 사용하려면 2개의 벡터를 동시에 저장할 수 있는 enhanced scan을 사용하여야 한다. 그러나 이의 사용시 너무나 많은 면적을 차지하기 때문에 이를 거의 사용하지 않는다. 따라서 표준 scan을 이용한 회로에 대해서 연구가 활발해지고 있다. 표준 scan회로의 경우에는 scan 이동 (scan shifting) 기법을 이용하거나 functional justification을 이용하는 방법이 있다.

scan shifting 기법을 사용하는 경우에는 테스트 가능한 많은 경로에 대한 테스트를 생성하지 못할 수 있다. 간단한 예는 그림 3과 같다.

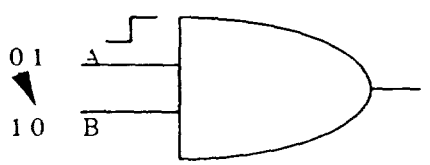

그림 3. scan shifting으로 테스트 불가능한 예

만일 scan chain이 A-B로 되어 있고 A의 상승전이를 고려할 경우 이의 테스트를 위해 B는 해저드가 없는 0을 가져야 한다. 하지만 scan shifting 에 의해 B에는 2번째 cycle에 0을 가지게 된다. 따라서 이 고장을 검출하는 테스트를 구할 수 없다. 따라서 정확한 지연 검사를 위해서는 첫 벡터를 읽어서 이를 사용하고 2번째 벡터는 회로를 통해 회로의 기능에 따라 정해지며 2 cycle 동안 테스트에 필요한 요건들을 만족시켜야 한다. 이를 functional justification 이라 부르는데 이를 이용해야 한다.

Cheng등[16,17]은 scan을 사용한 회로를 대상으로 테스트 생성기를 연구하였다. 여기서는 functional justification 기법을 이용하였고 또한 각 scan flop에 대해서 onset과 offset을 미리 구하여서 효율적으로 테스트를 생성하였다. 하지만 onset 등을 미리 구하기 때문에 상당히 많은 기억소자를 사용하고 또한 이들이 사용한 방법은 경성테스트를 만들어내지 못하게 된다.

Underwood 등[9]에서는 비슷한 방법이지만 큰 회로를 다루기 위해서 기억소자 사용을 절약하였지만 시간을 역으로 다루어 플립플롭을 다른 소자와 똑같이 생각하는 방법으로 이 역시 2개의 사이클에 대해서 논리값을 분리해서 생각하여 경성테스트를 만들어내지 못하고 생성된 근사경성테스트를 고장시뮬레이션을 통해 경성테스트와 강연성테스트를 생성하였다. 따라서 주어진 경로 고장에 대해서 직접 경성테스트를 구하는 방법이 필요하게 되었다.

Underwood 등[18]에서는 29개의 논리값을 사용하여 scan회로에 대해서 경성테스트를 구했다. 이 연구에서는 다른 연구와는 달리 기본적인 게이트 외에도 tri-state 소자, MUX, AOI 등의 소자들도 다루었고 또한 트랜지스터로 구성되어 논리기능만 알 뿐 실제 구성을 모르는 블럭에 대해서도 다루었다. 많은 논리값을 갖게 된 또하나의 이유는 제한값을 만족시키는 입력을 찾는 과정에서 생기는 모순을 가능하면 빨리 찾도록 하는 근사값을 포함하였기 때문이며 이때문에 많은 시간을 절약하게 된다. 여기서도 기억소자의 절약을 위해 reverse time processing을 이용하였다. 기본적인 알고리즘은 그림 4와 같다. 또 여기서는 가능한한 많은 테스트를 구하기 위하여 먼저 경성테스트를 구하고 이 경성테스트를 구할 수 없는 경로에 대해서는 강연성테스트를 구한다. 또 강연성테스트도 구할 수 없는 경로에 대해서는 약연성이나 경로토글테스트를 구한다.

```
테스트의 종류에 따라 제한값을 설정
해저드가 없는 값에 대해서 justification 수행
2번째 사이클의 값에 대해서 justification 수행
1번째 사이클의 값에 대해서 justification 수행
```

그림 4. 테스트 생성 알고리즘

5. 경로지연 고장 시뮬레이션

최근 경로지연 고장에 대한 벡터 자동생성에 대한 많은 연구가 있었지만 고장 시뮬레이션에 대해서는 그 중요성에 비해 많지 않았다. 고장 시뮬레이션은 설계의 기능을 검증할 때 쓰였던 벡터나, 또는 회로의 설계자가 특정한 임계경로(critical path)를 확인하기 위해 만들어진 벡터를 이용할 수 있고 또는 임의로 만들어진 벡터들을 이용하여 경로지연 고장을 검출하는데 쓰인다. 또한 자동생성기로 생성된 벡터를 확인하는데도 쓰인다.

Smith [7] 는 combinational 회로에 대해 고장 시뮬레이션을 수행하였다. 여기서는 2사이클동안 해저드가 없는 0, 2사이클동안 해저드가 없는 1, 하강전이, 상승전이, 2번째 사이클 값이 0, 2번째 사이클 값이 1인 6개의

논리값을 사용하였다. 각 소자에 대해서는 6개의 논리회로를 이용한 연산표를 이용하여 시뮬레이션을 수행하였다.

Schulz 등[19]은 PPSFP [20]기법에 기초를 두고 경로지연 고장에 대한 고장 시뮬레이션을 개발하였다. 이 역시 6개의 논리값을 사용하였고 3개의 bit으로 encoding하였다. 3개의 bit는 각각 최종값이 0인가 1인가, 정상상태인가, 그리고 전이가 전달되는가를 나타낸다. 이 encoding 이용해 각 소자를 연산할 때 bitwise 이진연산을 이용하여 시뮬레이션을 수행한다. 이 역시 combin- ational 회로에만 사용가능하다.

Kang 등[21]은 표준 scan 회로에서의 경로지연 고장에 대한 고장 시뮬레이션을 개발하였다. 이 역시 PPS-FP에 기초를 두고 있다. 이는 기존과는 다른 논리값을 사용하였다. 기존의 0 1 X에 tri-state 소자를 다루기 위해 high impedance값 z을 포함하였다. 2bit를 이용하여 기존 0 1 X Z 를 encoding하고 정상상태를 나타내기위해 1bit를 더 사용하였다. encoding은 이진연산을 주소화하도록 선택되었다. 각 테스트의 요건에 따라 각 소자에 계산을 위한 연산식을 구해서 계산시 이를 사용한다. 이는 scan을 이용한 회로가 2사이클의 sequential회로인 것을 생각하여 각 소자마다 2사이클동안의 값을 보관하도록 하여 효율적으로 시뮬레이션을 수행한다. 각 소자계산에 대한 식뿐만 아니라 정상상태를 나타내는 것, 또 고광을 검출하는 것 모두 이진연산을 이용하여 병렬적으로 수행되어진다. 고장의 검출은 시험중인 경로를 따라가면서 테스트의 종류에 따라 on path나 off path의 값을 확인함으로써 결정을 하게 된다.

의 경고지연고장에 대한 시뮬레이션을 개발하였다. 이는 주어진 벡터에 대해서 (1)하나의 빠른 클럭을 이용한 클럭구성이 병렬로 시뮬레이션되고 (2)시뮬레이션 동안 (1)에서와 같은 fault coverage를 갖는 최소의 클럭구성을 구할 수 있다. 또 주어진 클럭구성에 따라 여러개의 빠른 클럭을 이용하여 시뮬레이션을 가능하게 했다.

6. 결 론

최근에 활발한 연구가 진행되고 있는 지연시험에 대하여 살펴보았다. 보다 고질의 칩을 개발하기 위하여 지연시험의 수행은 필연적이라 할 수 있다. 또 여기서는 다루지 않았지만 모든 경로에 경성테스트가 생성될 수 있도록 회로를 재합성하는 방법[23], DFT기법을 이용하는 방법[24]등 여러 분야에 걸쳐서 연구가 활발하게 진행되고 있다. DFT기법을 이용한 회로나 아니면 기존의 회로에 대하여 보다 효율적인 방안이 개발된다면 지연테스트 생성기나 고장 시뮬레이터의 활용도가 크게 증가될 것이다.

참 고 문 헌

[1] E. Hsieh et al., "Relay Test Generation", Proc. of DAC, 1977

[2] J. Lesser and J Shedletsky, "An Exprimental delay test generator for LSI logic", Trans. Computers, March 1980

[3] J. Savir and W. Mcanney, "Random pattern trtability of delay faults", Proc. of ITC, 1986

[4] V. Iyengar et al., "Delay Test Generation-Concepts and coverge Metrics", Proc. of ITC, 1988

[5] J. Carter et al., "Efficient Test coverage Determination for Delay Faults", Proc. of ITC, 1987

[6] J. Waiwkauski et al., "Transition fault simulation by parallel pattern single fault propagation", Proc. of ITC, 1986

[7] G. Smith, "Model for delay faults based upon paths", Proc. of ITC, 1985

[8] E. Park and M. Mercer, "Robust and Non-Robust Tests for path delay faults in a combinational circuit", Proc. of ITC, 1987

[9] B. Underwood et al., "A Path-Delay Test Generator for Large VLSI Circuits", Proc. of ICVC, 1993

[10] A. Pramanick and S. Reddy, "On Multiple Path Propagating Tests for Path Delay faults", Proc. of ITC, 1991

[11] C. Lin and S. Reddy, "On Delay Fault Testing in Logic Circuits", Trans. on Computer, September 1987

[12] M. Schulz et al., "Advanced Automatic Test Pattern Generation Techniques for Path Delay Faults", Proc. of FTCS, 1989

[13] C. Glover and M. Mercer, "A Method of Delay Fault Test Generation", Proc. of DAC, 1988

[14] P. Goel, "An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits", Trans. on computers, March 1981

[15] M. Schulz et al., "SOCRATES : A Highly Efficient Automatic Test Pattern Generation System",Proc. of ITC, 1987

[16] K. Cheng et al., "A Partial Enhanced-Scan Approach to Robust Delay Fault Test Generation

for Sequential Circuits", Proc. of ITC, 1991

[17] K. Cheng et al., "Robust Delay Fault Test Generation for Sequential Circuits", Proc. of ITC, 1991

[18] B. Underwood et al., "Fastpath : Robust Path Delay Test Generator for Standard Scan Designs" Proc. of ITC, 1994

[19] M. Schulz et al., "Parallel Pattern Fault Simulation of Path Delay Faults", Proc. of DAC, 1989

[20] W. Daehn and M. Geilbert, "Fast Fault Simulation for Combinational Circuits by Compiler Driven Single Fault Propagation", Proc. of ITC, 1987

[21] S. Kang et al., "Path - Delay Fault Simulation for a Standard Scan Design Methodology", Proc. of ICCD, 1994

[22] I. Pomeranz and S. Reddy, "SPADES-ACE:A Simulation for Path Delay Faults in Sequential Circuits with Extensions to Arbitrary Clocking Schemes", Trans. on CAD, 1994

[23] X. Xie et al., "Design of Robust - Path - Delay -Fault-Testable Combinational Circuits by Boolean Space Expansion", Proc. of ICCAD, 1992

[24] T. Chakraborty et al., "Design for Testability for Path Delay Faults in Sequential Circuits", Proc. of DAC, 1993

저 자 소 개

강성호(姜成昊)

1963년 4월 13일생. 1986년 2월 서울대 공대 제어계측공학과 졸업. 1988년 5월 The University of Texas at Austin 전기 및 컴퓨터공학과 졸업 (석사). 1992년 5월 The University of Texas at Austin 전기및 컴퓨공학과 졸업(공박). 美國 Schlumberger연구원. Motorola 선임연구원. 현재 연세대학교 공과대학 전기공학과 조교수.