특 집

VLSI 테스팅의 기술 동향

강 성 호, 강 용 석
연세대학교 전기공학과

I. 테스팅의 필요성

테스팅이란 주어진 시스템이 정상적으로 동작하는 가를 분석하여 불량품을 가려내는 것으로 제품의 신뢰성을 높이고 더 나아가 고장 위치 혹은 원인의 진단을 위한 과정을 수행할 수 있도록 한다. 시스템의 비정상적인 동작은 설계 오류(design error)에 그 원인이 있을 수도 있는데 이는 시뮬레이션을 통한 설계 검증 과정을 통해 분석한다. 테스팅은 설계 검증을 거친 시스템에 대해 제조 후나 사용 중에 발생하는 비정상적인 동작을 검출하는 것으로 설계 검증과 구별된다. 비정상적인 동작을 유발하는 원인을 결함(defect)이라 부르고 이러한 결함에 의해 발생하는 장애(failure)를 모델링한 것을 고장(fault)이라 한다. 따라서 테스팅은 고장을 모델링하고 이러한 고장이 시스템상에 존재하는가를 분석하는 모든 과정을 말한다.

현재 반도체 기술이 급속도로 발달함에 따라 시스템의 복잡도가 급격히 상승하고 있으며 이러한 급격한 기술 향상과 그에 따른 발전된 칩의 개발은 전체적인 테스팅 기술 및 테스팅의 경제성 향상을 담보로 한다. The international roadmap for semiconductor 1997[1]에 의하면 집적 회로의 개발과 제조 과정에서 테스트 프로그램 개발, 디버깅 및 ATE(Automatic test equipment)을 위한 비용은 6백만 달러에 이르고 이 비용중 35%가 NRE (nonrecurring engineering) 비용이다. 제조 공정이나 설계 방법의 변화는 더욱 경제적인 한계를 넘어서는 비용을 초래하게 될 것이다. 따라서 현재와 같은 반도체 기술 및 산업의 발전을 위해서는 테스팅에 관한 연구 및 관심은 필수적이다.

II. 테스팅 분류

테스팅의 여러 연구분야에 대한 기술에 앞서 기본적인 테스팅 방법 및 용어를 설명하고자 한다. 테스팅되고 있는 회로의 출력을 주출력(primary output)이라 하고 이들은 쉽게 관찰할 수 있다고 가정한다. 마찬가지로 회로의 입력은 주입력(primary input)이라 하고 이들은 쉽게 조절할 수 있다고 가정한다. 회로의 조절용이도(controllability)란 회로의 주출력들에 적당한 값을 설정하여 회로내의 각각의 노드에 특정한 신호값을 발생시킬 수 있는 정도를 나타낸다. 관측용이도(observability)란 주입력을 조절하는 동안 주출력을 관측함으로써 회로내의 어떤 노드에 있는 신호의 값을 확인할 수 있는 정도를 말한다. 테스트용이도(testability)는 회로에 테스팅을 수행하는 과정의 수월함의 정도를 반영한다. 일반적으로 높은 테스트용이도를 갖는 회로는 낮은 테스트용이도를 갖는 회로에 비해 관측용이도와 조절용이도가 높은 값을 갖는다.

고장 검출(fault detection)은 주어진 회로가 하나 이상의 고장을 가지고 있는가를 확인하는 과정이다. 이것은 테스트 패턴(test vectors or patterns)이라 부르는 일련의 입력값들을 회로에 가하고 주출력에서 결과값들을 관측하여 수행된다. 만약 출력값들이 예상된 값들과 다르면 회로 내에 고장이 존재하는 것이다. 테스팅에서 테스트 패턴 집합은 가능한 한 많은 고장을 검출하기 위해 가해지게

〈표 1〉 테스팅 방법에 따른 분류

분류 방식	테스팅 방법	테스팅 명칭
설계 유무	기능만 아는 경우	흑상자 테스팅(black box testing)
	설계를 아는 경우	백상자 테스팅(white box testing)
테스팅 수행 시기	일반동작과 동시	동시 테스팅(on-line testing or Concurrent testing)
	일반동작과 테스팅 분리	비동시 테스팅(off-line testing)
테스트 패턴 공급원	시스템 자체내	자체 테스팅(self-testing)
	외부 테스터를 사용	외부 테스팅(external testing)
테스팅 목적	설계 오류	설계 검증(design verification)
	제조상 불량품	가부 테스팅(acceptance testing)
	제조 공정 불안정	번인 테스팅(burn-in)
	공정 안정화기 물리 결함	품질보장 테스팅(quality-assurance testing)
	물리적 결함	현장 테스팅(field testing) 유지 테스팅(maintenance testing)
테스팅 대상	집적 회로	부품 수준 테스팅
	회로 보드	보드 수준 테스팅
	시스템	시스템 수준 테스팅
테스팅 속도	일반 동작보다 저속	직류 테스팅(DC testing)
	일반동작 속도	교류 테스팅(AC testing) At-speed 테스팅
테스팅시 접근 가능한 노드	입출력 핀	입출력 핀 테스팅(Edge-pin testing)
	입출력 핀 및 내부 노드	탐침 테스팅(Guided-probe testing) 침상 테스팅(Bed-of-nails testing) 전자선 테스팅(electron-beam testing) 회로내 테스팅(in-circuit testing)

되는데 테스트 패턴의 효과는 고장 검출률(fault coverage)로 측정하게 된다. 이것은 회로내의 모든 검출 가능한 고장수에 대한 테스트 패턴에 의해 잠재적으로 검출될 수 있는 고장의 수의 비로 나타낸다. 태스트 패턴의 길이는 전체 테스트 패턴의 개수이고 테스트 시간은 테스트 패턴을 가하고 출력을 관측하기 위해 필요한 시간을 말한다. 주어진 테스팅 방법의 품질을 평가하기 위한 주된 척도는 테스트 패턴을 생성하기 위한 시간, 고장 검출률, 그리고 테스트 시간을 포함한다. 물론 테스팅을 용이하게 하기 위해 설계시에 관측용이도와 조절용이도를 높이기 위한 회로나 입출력 핀이 추가된 경우에는 증가된 설계 면적, 회로 성능 저하, 및 증가된 입출력 핀들의 개수가 어떤 테스팅 방법을 선택할 것인가에 많은 영향을 미치게 된다. 때때로 서로 다른 테스팅 방법이 이러한 여러 조건들 사이에 다양한 선택 조건을 갖는다.

테스팅은 방법에 따라 여러 종류로 세분화할 수 있으며 이를 〈표 1〉에 나타내었다.

III. 테스팅의 연구 분야

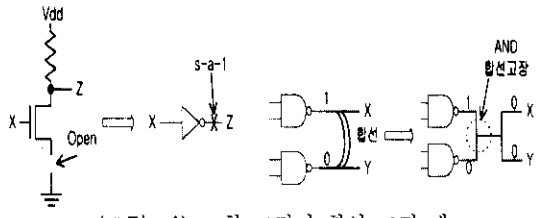
1. 고장 모델링

고장 모델링[2]이란 물리적 고장을 논리 고장으로 전환하는 작업을 말하며 이러한 과정을 통해 고장 분석 문제가 제조 기술이나 공정에 무관한 논리상의 문제로 바뀌게 된다. 즉 논리 고장은 모델링된 시스템의 동작에서 물리적 고장의 효과를

나타낸다. 소자를 모델링하는 과정에서 논리 기능(function)과 타이밍을 구별하기 때문에 대체로 논리 기능에 영향을 주는 고장과 동작 속도에 영향을 주는 지연 고장(delay fault)을 구별하여 사용한다. 일반적으로 물리적 고장과 논리 고장이 일대일로 대응되지는 않지만 대개의 경우 논리 고장을 찾기 위한 테스트 패턴값은 대체로 불량칩을 검출할 수 있으므로 논리 고장을 검출하기 위한 테스트 패턴값을 구하여 이를 집적회로를 테스팅하는 데이터로 많이 사용한다. 따라서 고장 모델이 더욱 정확할수록 보다 많은 물리적 고장들을 검출할 수 있다. 그러나 보다 높은 정확성을 갖는 고장 모델은 테스트 패턴을 생성하기 위한 알고리듬 복잡도의 증가, 테스트 생성 시간의 증가 및 긴 테스트 시간을 비용으로 지불해야 얻을 수 있다.

고장 모델들은 다양한 추상 수준에서 고장을 표현할 수 있다. 가장 일반적인 추상 수준은 게이트 수준이다. 물론 보다 높은 수준뿐 아니라 트랜지스터 수준의 고장을 표현하는 고장 모델들이 존재한다. 회로내의 모든 고장들의 가능한 조합수는 너무 많기 때문에 이를 위한 테스트 패턴을 생성하는 것이 불가능하므로 회로는 한개의 고장을 가지고 있다고 가정한다.

게이트 수준의 고장에서는 고착(stuck-at) 고장과 합선(bridge) 고장으로 모델링된다. 고착 고장 모델은 게이트 수준에서 가장 많이 사용되는 모델로 물리적 결함이 논리 게이트 입출력선의 값을 영구적으로 논리 0(s-a-0으로 표기), 혹은 논리 1(s-a-1로 표기)로 고정된다고 가정한다. 고착 고장 모델은 회로의 물리적인 결함을 대체로 잘 반영하고 고장 개수가 작아(단일 고장인 경우 n개의 신호선에 대해 2n개의 고착 고장이 존재한다.) 테스트 패턴 생성에 용이하게 사용할 수 있어 실제 테스트 분야에서 가장 널리 사용된다. 합선 고장은 인접한 두개 이상의 신호선이 합선되는 고장을 말한다. 회로의 성질에 따라 합선 부분이 AND 게이트 혹은 OR 게이트와 등가가 된다. 합선 고장의 경우 간단한 회로의 경우에도 합선된 신호선의 개수와 위치에 따라 그 조합수가 너무나 많기 때문에 일반적인 회로에서 모든 조합의 합선 고장을



〈그림 1〉 고착 고장과 합선 고장 예

고려하는 것은 불가능하다. 또한 회로의 레이아웃에 대한 정보가 결여되어 있는 경우에는 어떤 신호선이 인접해 있는가를 알 수 없기 때문에 정확하게 합선 고장을 모델링하는 것은 불가능하다. 〈그림 1〉에 고착 고장과 합선 고장의 예를 나타내었다.

고착 고장이나 합선 고장은 물리적 결함을 잘 기술할 수 있고 이들 고장을 검출하기 위한 테스트 패턴값으로 많은 결함을 검출할 수 있지만 이 고장 모델로 검출할 수 없는 결함들이 많이 존재하고 트랜지스터 수준에서 고려한 고장이 반드시 게이트 수준의 고장과 대응되는 것은 아니어서 결함에 대한 표현이 어렵다. 따라서 보다 정확하게 물리적 결함을 검출하기 위해 트랜지스터 수준의 고장 모델을 사용할 필요성이 대두된다. 이러한 트랜지스터 고장 모델을 사용하는 경우에는 복잡도가 크게 증가하게 된다. 또한 게이트 수준과 트랜지스터 수준에서 사용되는 일반적인 고장 모델 외에 회로 소자의 특성에 따라 새로운 고장 모델을 사용할 수 있다. 대표적인 것이 메모리와 PLA 소자이다. 이들 소자는 소자의 구조상 고착 고장과 합선 고장 외에 부가적인 고장 모델을 사용하여 테스팅한다. 회로가 거대한 경우 게이트 수준이나 트랜지스터 수준의 고장 모델을 사용하면 테스팅을 위한 고장의 수가 너무나 많기 때문에 보다 높은 수준인 기능적(behavioral) 수준 고장 모델을 사용하기도 한다. 그리고 지금까지 언급한 고장 모델들은 물리적 결함이 영구적으로 비정상적인 동작을 유발하는 것을 고려하였다. 이와 함께 최근에는 전자기나 알파 입자의 방사 및 전원 공급기의 변동에 의해 발생하는 과도 고장(transient fault)과 몇몇 변수의 변화로 잡음 여유가 감소하여 고장 신호가 주기적으로 회로의 동일한 부분 혹은

기능에서 발생하는 간헐 고장(intermittent fault)에 대한 연구도 활발히 진행되고 있다.

2. 고장 시뮬레이션

고장 시뮬레이션[3]은 테스팅 기술에서 가장 기본적인 것으로 회로 내에 가상적으로 고장을 주입하고 고장에 의한 회로의 반응을 확인하여 주어진 테스트 패턴으로 검출할 수 있는지 여부를 판별하는 실험이다. 고장 시뮬레이션의 응용 분야는 크게 다음과 같이 나눌 수 있다.

• 고장 검출률 평가

주어진 테스트 패턴으로 어느 정도의 고장을 검출할 수 있는가를 판별한다.

• 테스트 패턴 생성

어떤 목표 고장을 테스팅하기 위해 생성된 테스트 패턴으로 그 외의 검출 가능한 고장을 가려내고 이에 관한 정보를 테스트 패턴 생성을 위해 사용한다.

• 고장 사전(fault dictionary) 작성

각 고장과 테스트 패턴 사이의 검출 관계를 기록한 파일로 고장 진단을 위한 데이터로 사용한다. 일반적으로 고장 검출이 가능한 모든 테스트 패턴을 기록한 후 각 기록된 패턴으로 검출할 수 있는 고장들을 교집합 연산을 수행하여 고장 위치를 판단하기 위해 사용된다.

• 회로 분석

높은 신뢰도를 갖는 시스템을 위해 회로 분석 도구로 사용된다. 회로에 고장이 있는 경우 레이스(race)나 해저드(hazard)를 발생시킬 수 있고 또한 오실레이션이 생기거나 고정 상태(deadlock state)에 진입할 수가 있다. 그리고 순차회로의 경우 적절한 초기화를 벗어날 수도 있으며 조합회로에서 순차회로로 혹은 동기회로에서 비동기회로로 변환될 수 있다. 이러한 문제들을 고장 시뮬레이션을 통해 사전에 분석하고 이에 대한 대체 방안을 마련할 수 있다.

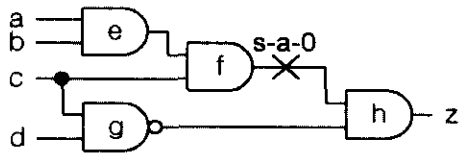
일반적으로 고장 시뮬레이션은 고장이 없는 회로(good circuit)와 고장이 주입된 회로(faulty circuit)에 동일한 테스트 패턴을 가한 후 결과를 비교하여 가정된 고장이 검출되는 가를 판단한다. 한번에 하나의 고장을 회로에 주입하는 직렬 고장 시뮬레이션은 논리 시뮬레이션 방식을 거의 수정없이 사용할 수 있으나 단일 고착 고장의 경우 고장의 수가 회로의 크기에 선형적으로 비례하여 증가하므로 전체 고장 시뮬레이션의 속도가 느려 실용적이지 못하다. 이러한 단점을 극복하기 위해 여러 개의 고장 회로를 고장이 없는 회로와 비교하는 병렬 고장 시뮬레이션 방법이 있다. 만약 2진 논리값을 시뮬레이션에 사용한다면 워드가 W 비트로 구성되어 있는 경우 한 개 비트를 고장이 없는 회로에 배정하고 나머지 W-1개의 비트는 고장이 있는 회로에 배정하여 한 번의 연산으로 W-1개의 고장을 시뮬레이션할 수 있다. 병렬 고장 시뮬레이션은 직렬 고장 시뮬레이션에 비해 속도가 빠르지만 시뮬레이션 수행시 고장이 없는 회로와 고장 회로의 회로내 노드들의 값이 대부분 같기 때문에 필요 없는 병렬 논리 연산을 수행하게 된다. 이러한 단점을 없앤 것이 고장이 없는 회로와 논리값의 차이가 있는 고장 회로의 노드에 대해서만 시뮬레이션을 수행하는 동시(concurrent) 고장 시뮬레이션이다. 이 외에 연역 고장 시뮬레이션은 주어진 테스트 패턴으로 고장이 없는 회로에 시뮬레이션을 수행한 후 회로내의 각 신호선까지 고장의 영향이 전파되는 고장을 연역적으로 추론하는 방식이다. 따라서, 주입력에서 시작하여 각 신호선까지 전파되는 고장의 집합을 구하여 최종적으로 주출력에서 검출 가능한 고장의 집합을 구한다. 이 외에 고속 고장 시뮬레이션을 위해 워드의 각 비트에 하나의 테스트 패턴을 배정하여 고장이 없는 회로에 대하여 시뮬레이션을 한 뒤 한번에 하나의 고장만을 주출력으로 전파시키는 방법인 병렬 패턴 단일 고장 전파 시뮬레이션(PPSFP: Parallel Pattern Single Fault Simulation)이 있다. 이 방법은 조합 회로에서 기존의 병렬, 연역 및 동시 고장 시뮬레이션에 비해 수십배 이상 빠른 것으로 알려져 있다.

3. 테스트 패턴 생성

테스팅의 주된 문제는 결국 테스트 패턴을 어떻게 효과적으로 생성하는 가이다. 자동 테스트 패턴

생성(ATPG : automatic test pattern generation)은 주어진 회로에 대해 가정된 고장 모델을 검출하기 위한 테스트 패턴 생성을 자동화시키는 것이다. 테스트 생성 방법은 크게 임의 테스트 패턴 생성과 결정적 테스트 패턴 방식으로 나눌 수 있다. 임의 테스트 패턴 생성은 고장 모델이나 회로 구조에 비교적 무관하게 임의의 테스트 패턴을 생성하는 방법으로 회로 내에 하드웨어로 패턴 생성기를 구현하는 경우에 용이하나 일관성 있는 높은 고장 검출률을 보장하지 못한다. 고장 검출률을 높이기 위해 조절용이도, 관측용이도와 같은 테스트 용이도를 측정하거나 회로의 구조를 분석하여 임의 패턴에 가중치를 부과하여 사용하기도 한다. 이에 반해 결정적 테스트 패턴은 회로와 소자의 동작 특성에 따라 알고리듬을 사용하여 주어진 고장 모델을 위한 패턴을 생성하는 것으로 높은 고장 검출률을 갖는다. 이 글에서는 임의의 회로의 고착 고장을 위한 결정적 테스트 패턴 생성에 관한 내용을 주로 설명한다.

ATPG는 검출 가능한 모든 고착 고장을 주출력에서 관측할 수 있도록 하는 주입력에 가하는 테스트 패턴을 생성해야 하는데 주출력에 고장 효과가 나타나게 하기 위해서는 두 부분의 과정이 필요하다. 하나는 고장 유발(fault excitation)이라 부르는 것으로 고장 위치에 고장이 있을 경우와 없는 경우의 차이를 발생시키는 것이다. 그 후 이 차이를 하나 이상의 주출력에 전파시켜는 고장 전파 과정이 필요하다. 〈그림 2〉의 f에서 s-a-0 고장을 유발하기 위해서는 f를 논리값 1로 해주어야 한다. 따라서 주입력 a, b, c는 모두 1이 되어야하고 유발된 고장을 전파하기 위해 주입력 d는 0이 되어 결국 테스트 패턴은 (a, b, c, d)=(1, 1, 1, 0)가 된다. ATPG는 이러한 과정을 자동으로 수행하는 것을 말한다.



〈그림 2〉 고장 유발 및 고장 전파를 위한 예

대표적인 ATPG 알고리듬은 D-알고리듬, PODEM 알고리듬, 그리고 FAN 알고리듬이 있다[2,3]. 이들 알고리듬의 기본적인 차이는 탐색 공간의 차이, 탐색 공간 생성을 위해 사용되는 전략, 그리고 탐색에 도움을 주기 위해 사용되는 방법의 차이에 있다. D-알고리듬의 탐색 공간은 회로내의 모든 노드로 0, 1, X의 논리값에 D(고장시:0, 정상시:1)와 $\overline{D}$**(고장시:1, 정상시:0)를 추가하여 5개의 논리값을 사용한다. 테스트 패턴을 생성하기 위해 D-알고리듬은 모든 노드에 대해 탐색을 수행하여 주입된 고장 위치에 s-a-0이면 D를 s-a-1이면 $\overline{D}$를 만들기 위한 입력값들을 찾아 고장을 유발시키고 적어도 한개의 주출력에 D나 $\overline{D}$를 전파시키기 위한 입력값을 전체 노드를 탐색 공간으로 하여 찾는다. 반면에 PODEM(path oriented decision making) 알고리듬은 탐색 공간을 주입력만으로 축소한다. 조합 회로의 모든 노드의 값은 주입력값들의 조합에 의해 만들어지는 것이므로 주입력만을 고려하여도 충분하기 때문이다. 〈그림 2〉의 고장을 위한 테스트 패턴을 PODEM을 사용하여 생성하면 고장 위치 f에서 주입력 쪽으로 경로를 선택하여 해당경로의 시작이 되는 주입력인 c에 1을 설정하게 된다. 주입력의 새로운 논리값이 설정된 효과를 판단하기 위해 영향 분석(implication)을 수행하여 유발된 고장이 주출력에 나타나는 가를 확인하고 주출력에 고장이 나타나지 않으면 다른 주입력을 선택하여 논리값을 설정한 후 영향 분석을 수행하여 충돌이 있는가 아니면 테스트 패턴이 생성됐는가를 확인하게 된다. 어떤 하나의 주입력의 값을 설정한 후 영향 분석을 하여 논리값의 충돌이 발생하면 간단히 설정된 주입력 값의 보수를 다시 입력으로 취한 후 시뮬레이션을 수행한다. 만약 보수값으로도 충돌이 발생하게 되면 해당 고장을 테스트하기 위한 테스트 패턴을 생성할 수 없음을 쉽게 알 수 있다. 〈그림 2〉에서 만약 h가 s-a-0인 경우에 h-g-c 경로를 선택하여 c를 0으로 설정하여 영향 분석을 수행하면 h가 0이 되어 이미 설정된 D 값과 충돌을 일으키게 된다. 따라서 c를 이전에 설정된 값의 보수인 1을 취하여 영향 분석을 수행한다. 충돌이 없으면 원하는 값을 설정

하기 위한 노드에서 경로를 선택하여 해당 경로의 주입력에 논리값을 가한 후 테스트 패턴이 생성됐는가 혹은 충돌이 발생했는가의 과정을 반복한다. FAN 알고리듬은 PODEM과 비슷한 과정을 통해 테스트 패턴을 생성하는데 탐색공간이 헤드라인(headline)이라고 불리는 노드들로 구성된다. 헤드라인이란 〈그림 2〉의 e, c, d와 같이 그 위치에서 주입력까지의 후방 내부점에 분기점(fanout)이 없는 일종의 트리 구조의 루트를 말한다. 따라서 〈그림 2〉의 테스트 패턴을 생성하는 경우 PODEM의 탐색 공간이던 a, b를 제외하고 e를 추가하게 된다. 결국 PODEM에 비해 탐색 공간이 줄어들게 되어 테스트 생성을 위한 시간을 줄일 수 있다.

일반적으로 테스트 패턴 생성 문제는 고장 검출을 위한 테스트 패턴이 존재하지 않는 무해 고장(redundant fault)의 확인이나 난이 고장(difficult fault)에 대한 테스트 패턴 생성에 대부분의 시간을 소모한다. 따라서 현재의 ATPG tool들은 충돌을 미리 알기 위해 논리값을 추가하여 사용하거나 테스트 패턴 생성을 위한 탐색 과정에서 반드시 만족시켜야 할 강제 조건들을 미리 저장하여 전체 탐색 공간의 크기를 감소시키는 방법을 개발하여 사용하고 있으며 조합 회로의 고착 고장에 대하여 100%의 고장 검출률을 얻을 수 있다. 하지만 순차 회로에 대한 ATPG는 아직까지 만족할 만한 수준에 도달하지 못했다. 그 이유는 회로 내부에 상태 소자가 존재하기 때문이다. 순차 회로의 테스트 생성에 가장 많이 사용되는 것은 반복 어레이 모델(iterative array model)로 상태 소자의 되먹임(feedback)이 있는 경우 되먹임 루프를 자르고 시간적으로 전개하여 조합회로가 반복되어 있는 것과 같이 가정하는 것이다. 조합 회로의 경우는 주출력으로 고장을 전파하면 되나 순차 회로의 경우에는 상태 소자의 입력으로 고장을 전파되면 다음 시간 프레임(frame)에서 다시 고장 전파를 시도한다. 고장 유발을 위한 입력도 주입력뿐 아니라 상태 소자의 출력값도 필요하기 때문에 시간에 대해 역방향으로 상태 소자에 설정된 값을 지정(justification)하는 과정이 필요하다. 따라서 회로 내부의 관측용이도와 조절용이도가 조합 회로에 비해 현저히 낮고 테스트 패턴의 길이가 긴 경우가 많으며 탐색 공간의 크기도 커지게 된다. 또한 주어진 고장이 단일 고장이라 해도 반복 어레이로 전개한 조합 회로 상에서는 이것을 다중 고장으로 처리하여야 한다. 이러한 여러 가지 이유들로 인해 순차 회로의 테스트 패턴 생성은 많은 어려움을 갖고 있으며 수행된 연구들도 아직까지 만족할 만한 결과를 내놓고 있지 못하다. 순차 회로의 테스트 패턴의 생성의 어려움을 극복하기 위해서는 테스트용이화 설계(DFT:design for testability) 방법이 병행되어야 한다.

4. 테스트용이화 설계 및 내장된 자체 테스트

테스팅을 고려하지 않은 회로는 관측용이도와 조절용이도와 같이 테스트용이도가 낮은 회로나 신호 노드를 갖는 경우가 많다. 이러한 회로를 테스팅하기 위해서는 많은 시간과 비용이 필요하고 원하는 고장 검출률을 얻기 힘들다. 따라서 테스팅을 쉽게 하기 위해 주어진 회로를 변형시키거나 여분의 하드웨어를 첨가시키는 방식이 사용되는 이를 테스트용이화 설계라고 말한다. 그러나 테스트용이화 설계 방법을 적용할 경우 부수적으로 회로의 크기, 입출력단자(I/O ports)의 증가 및 성능저하가 생기게 된다. 그러나 테스트 비용 및 테스트 시간 절감 등, 테스팅 설계방법에 따른 이득이 매우 크고, 또한 최근에 개발되는 시스템들에 있어서 테스팅 설계방법을 적용하지 않을 시에는 테스팅 그 자체가 거의 불가능하기 때문에 거의 대부분의 반도체 회사에서는 이를 적극적으로 적용하고 있다. 단, 사용하고자 하는 테스팅 설계방법의 적용에 따른 이득과 부가적으로 요구되는 오버헤드의 증가에 따른 손실사이에 있어서 최적의 선택을 할 수 있도록 주의를 기울여야만 한다.

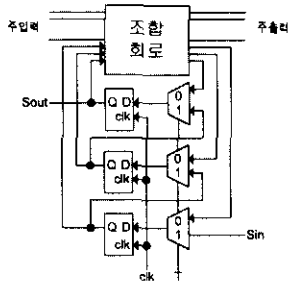
기본적인 테스트용이화 설계 방법으로 조절용이도 및 관측용이도가 낮은 노드에 조절 접점(control point)이나 관측 접점(observation point)을 삽입하여 각각 주입력이나 주출력으로 만들어 테스트용이도를 향상시키는 접점 삽입 기법과 모든 조합에 대한 테스팅을 수행하려면 많은 클럭을 소모하는 계수기나 쉬프트레지스터를 분할하여 필

요한 클럭수를 줄이는 분할기법이 있다.

테스트용이화 설계 기법 중 가장 널리 사용되는 방법이 주사(scan)기법[3]이다. 자동 테스트 패턴 생성을 위한 여러 가지 기법들의 등장으로 조합회로에 대한 테스트 패턴생성은 비교적 쉽게 할 수 있게 되었지만, 앞서 언급한 바와 같이 순차회로에 대해서는 많은 문제점을 가지고 있다. 주사기법은 순차회로의 기본을 이루는 상태소자(memory element)들을 테스트 모드에서 외부로부터 쉽게 제어할 수 있도록 한 기법이다. 〈그림 3〉과 같이 상태소자에 MUX를 추가하여 테스트모드(T=1)에서는 상태소자들이 직렬로 구성되게 하여 입력(Sin) 통해 원하는 값을 상태소자에 설정할 수 있도록 하고 출력(Sout)을 통해 상태소자의 값을 읽을 있도록 하여 관측용이도와 조절용이도를 대폭 향상시키고 순차회로의 테스트 패턴 생성 문제를 조합회로에 대한 것으로 만들었다. 주사기법은 모든 상태소자를 주사소자로 바꾸는 완전 주사 기법과 일부를 대체하는 부분 주사 기법이 있다. 그리고 완전 주사 기법중에 두개의 래치(latch)를 사용한 주사소자로 주사 경로를 만든 LSSD(level sensitive scan) 기법은 가하진 테스트 패턴사이의 레이스가 없음을 보장할 수 있어 지연 고장의 테스팅에 유리하다.

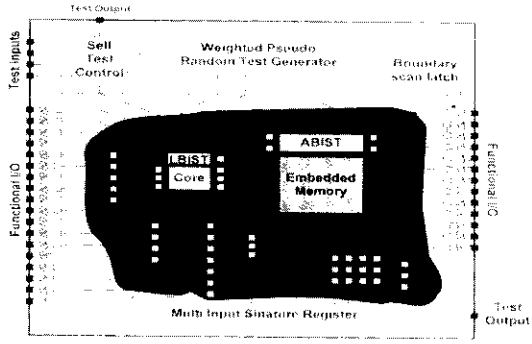
칩 수준의 테스트용이화 기법인 주사 기법에 대응하는 인쇄회로 기판(PCB:printed circuit board) 수준의 테스트용이화 기법이 경계 주사 기법이다. 이를 사용하여 기판에 실장된 칩들에 대한 테스팅과 칩들 사이의 신호선(interconnection)에 대한 고장유무를 확인한다.

기존의 단순한 구조를 갖는 기판의 경우에는 직접 테스트 단자를 기판에 인가하여 신호를 관측하는 방법(bed of nail) 등의 사용이 가능하였으나, 다층 구조를 갖는 기판이 등장하고 표면 실장 기술의 사용에 따라 기판 수준의 테스팅은 매우 어려운 문제가 되고 있으나 경계주사기법은 이러한 기판 수준 테스팅의 많은 문제점을 해결할 수 있다. 이것은 1990년도에 IEEE에서 제정된 기판 수준 테스팅을 위한 IEEE 1149.1라는 표준안[4]이 되었다.



〈그림 3〉 주사를 이용한 순차 회로

이러한 여러 테스팅에 관한 연구에도 불구하고 칩 외부에서 테스트 패턴을 인가하고 이에 따른 테스트 응답을 저장된 정확한 값과 1:1 비교하는 방식의 전통적인 테스트 기법은 내장된 코어나 메모리를 사용하여 칩의 복잡도가 급격히 증가하여 점차 그 적용이 어려워지고 있다. 이러한 문제점을 효과적으로 해결하기 위한 방편으로 내장된 자체 테스트(BIST:built-in self test) 기법[5]이 등장하게 되었다. 이 기법은 테스트 패턴을 칩 내부에서 생성하고 그 테스트 응답 또한 칩 내부에서 압축하여 단지 고장의 유무만을 외부에 알려주는 매우 효율적인 테스트 기법이다. 테스트 패턴은 하드웨어로 구성하기 용이한 임의 테스트 패턴을 사용하고 고장 검출률을 높이기 위해 가중치를 부가하기도 한다. 또한 모든 입력에 대한 출력값을 비교하는 대신 이를 압축한 압축치(signature)와 응답을 비교한다. BIST를 이용하면 계층적인 테스팅이 가능해져 시스템, 기판, 칩, 내장 코어 혹은 메모리에 대해 BIST를 구현하게 되면 상위 계층에서 하위 계층의 테스팅을 위해 BIST 회로를 구동하기만 하면 되어 복잡한 회로의 테스팅 문제를 많이 해결한다. 또한 회로 내부에서 테스팅이 가능하게 되어 시스템의 안정성을 판단하기 위한 도구로 이용될 수 있으며 동작 속도로 테스팅을 수행할 수 있는 장점이 있다. LBIST(logic BIST)는 임의 논리 회로나 내장된 코어를 위해 사용하고 내장된 메모리의 테스팅을 위한 MBIST(memory BIST)는 실제 많이 응용되고 있다. 물론 다른 테스트용이화 기법과 마찬가지로 테스팅을 위한 회로로 인한 성능 감소, 면적 증가, 입출력핀 증가 등의 문제들과 테스팅 비용 및 시간 사이에서 최적의 선택을 취



〈그림 4〉 테스트용이화 기법의 적용 개념도

해야 한다. 〈그림 4〉에 다양한 테스트용이화 기법을 사용한 회로의 개념도를 나타내었다.

5. 기타 주요 테스팅 분야

앞에서 언급한 것 외에 테스팅에 관한 연구 분야가 많이 있다. 고착 고장 이외의 다른 고장 모델을 테스팅하기 위한 전류 테스팅, 지연 고장 테스팅, 합선 고장 테스팅 등이 있으며 소자 구조의 특성에 의해 기존의 방법으로 테스팅하기 힘든 메모리 테스팅, PLA 테스팅 등이 있다. 또한 새로운 설계 방식과 회로 구성을 위한 비동기 회로 테스팅, 아날로그 회로 테스팅 등이 많이 연구되고 있다.

1) IDDQ 테스팅

고착 고장 이외의 고장에 대한 다양한 테스팅이 연구되고 있다. 전압대신 CMOS 회로의 전류를 측정하는 IDDQ(direct drain quiescent current) 테스팅[6]이 대표적인 예이다. IDDQ 테스팅은 칩의 입력값을 변화시킨 후 정적 논리 상태에서 전원의 전류를 측정하여 고장의 유무를 판별한다. 고장이 있는 회로의 경우 없는 회로에 비해 정적 전류의 크기가 크다. IDDQ 테스팅은 방법이 간단하고 고착 고장 모델로 검출하기 힘든 누설전류, 게이트 산화막 단락, 고저항 합선, 부동 게이트 등의 결함들을 검출할 수 있으며 결함이나 장애의 초기 원인이 IDDQ의 증가에 있기 때문에 칩의 신뢰도를 측정하는 방법으로도 사용할 수 있다. 정확한 IDDQ를 측정하기 위한 테스트 패턴의 생성, 전류 감지기 설계 등의 연구가 행해지고 있다.

2) 지연 고장 테스팅

고착 고장과는 달리 지연 고장은 시스템의 정상 상태에서의 논리 동작에는 영향을 미치지 않지만 전체 시스템의 성능을 저하시킨다. 따라서 칩이 고속화되면서 지연 고장을 위한 테스팅이 중요한 문제로 부각되어 이에 대한 연구가 활발히 진행되고 있다. 지연 고장 테스팅은 신호의 변화가 회로를 통하여 주어진 시간내에 통과하느냐를 결정하는 것이기 때문에 동적 논리 회로(dynamic logic)를 제외하고는 적어도 2개의 테스트 패턴을 사용해야 한다[7]. 지연 고장 테스팅에서는 회로의 지연이 게이트의 입력이나 출력에 집중되어 있다고 모델링한 게이트 지연 고장 모델과 경로를 따라 있는 작은 지연들이 합쳐져서 회로의 지연을 일으킨다고 모델링한 경로 지연 고장 모델을 주로 사용한다. 게이트 지연 고장의 경우 회로내 고려해야할 고장의 수가 그리 크지 않고 기존의 고착 고장을 위한 테스트 패턴을 이용하여 테스트 패턴을 생성할 수 있지만 회로내에 널리 분포된 작은 지연들로 인한 지연 결함을 검출할 수 없다.

반면에 경로 지연 고장을 이용한 테스팅은 회로의 해당 경로에 집중 또는 분포된 지연 결함을 모두 검출할 수 있다. 또한 회로의 시간 경계 근처에 있는 경로에 해로운 영향을 미치는 변이들을 관측할 수 있으며 긴 경로들의 클럭 속도에 대한 정확한 정보를 제공하므로 속도에 따라 칩을 분류하는데 이용할 수 있다. 하지만 회로내의 모든 경로를 고려하는 것은 불가능하므로 일반적으로 정적 시간 분석(static timing analysis)을 이용하여 가장 긴 경로들을 테스팅하게 된다. 최근에는 고성능의 칩의 개발을 위해서는 지연 고장 테스팅이 필수적이 되어 지연 고장 테스팅을 위한 테스트 패턴의 생성뿐 아니라 테스트용이화 기법에 대한 연구도 진행되고 있다.

3) 메모리 테스팅

집적도 가장 높은 반도체 소자인 메모리는 시장 규모가 크고 빠른 시장 진출 시기와 가격 경쟁력이 사업의 성패를 좌우한다. 따라서 작은 테스팅 비용과 효과적인 테스팅 방법의 개발이 중요하다. 집적도의 향상은 메모리 셀간의 거리를 가깝게 만

들어 이웃셀의 영향을 많이 받게 하고 저전력화와 셀 크기의 감소에 따라 이웃셀의 접근이나 쓰기 동작에 의한 오동작의 가능성이 커졌다. 따라서 효과적으로 메모리를 테스팅하기 위해서는 이웃셀의 영향을 모델링할 수 있는 고장 모델이 필요하며 DRAM과 SRAM, 그리고 ROM의 셀 설계 방식이 차이가 나므로 각각에 대해 적절한 고장 모델을 선택하거나 개발하여 테스팅을 수행해야 한다[8]. 가장 많이 사용되는 고장 모델로는 셀의 논리값이 0이나 1로 고정된 고착 고장, 상승이나 하강 천이가 발생하지 못하는 천이 고장이 있으며 k개의 셀 중 k-2개의 셀이 어느 특정한 상태에 있고 한 셀의 내용이 반전되면 다른 한 셀의 값에 영향을 미치는 k-결합 고장 있다. 관련된 k개의 셀의 위치에 대한 제약 조건이 없다면 검출하기 매우 어렵다.

위치에 제약을 두는 k-결합 고장으로 특정 셀과 주위의 4개 셀을 고려하거나 주위의 8개의 셀을 고려하는 이웃 패턴 감응 고장이 있다. 결합 고장과 이웃 패턴 감응 고장은 셀에 미치는 영향의 종류에 따라 다시 여러 종류로 세분화된다. 주로 SRAM에서는 2개의 셀을 고려하는 결합 고장을 사용하고 DRAM의 경우에는 커패시터 누설 전류를 고려하기 위해 주위의 셀들을 고려하는 이웃 패턴 감응 고장을 사용한다. 이과 같이 시간과 관련이 없는 고장 모델을 정적 고장 모델이라 하는데 이외에도 시간에 따라 오동작이 발생하는 원인을 모델링한 동적 고장 모델도 사용한다. 이와 같이 많은 고장 모델을 사용하기 때문에 많은 테스트 패턴을 필요로 하게 되어 테스트 시간이 길어진다.

또한 메모리는 현재 논리 회로내에 내장된 형태로 많이 사용되고 있어 이러한 메모리들을 테스팅하기 위해 DFT 기법이나 BIST 기법에 대한 연구가 활발하다. 이들 연구는 테스팅 시간을 줄이기 위해 병렬로 테스팅하기 위한 방법의 개발이 주를 이루고 있으며 최근에는 사용 중에 테스팅을 수행한 후에 원래 메모리에 저장된 내용을 복구할 수 있는 투명(transparent) BIST에 대한 연구도 진행되었다.

IV. 향후 연구 과제

테스팅과 관련된 향후 8~10년 사이의 반도체 관련 문제점은 다음과 같이 예상할 수 있다.

1) 잠재적 수율 저하: 수율 저하와 비용 상승을 일으키는 주된 기술적 문제는 ATE의 느린 속도 증가와 빠른 집적 회로의 속도 증가에 있다. 테스터의 정확도는 1년에 12%가 상승하는데 비해 집적회로의 속도는 1년에 30%씩 증가하고 있다[1]. 일반적으로 테스터의 속도는 소자에 비해 5배 정도 빨라야 하는데 이러한 추세가 계속 된다면 2010년쯤에 테스터와 집적회로 속도가 비슷해질 것이다. 그러나 2001년쯤부터 테스터의 부정확성에 의한 수율 저하가 받아들이기 힘들게 될 것이다.

2) ATE 비용: 고성능 시스템을 위한 핀당 ATE 비용은 20여년동안 $10k/핀 근처로 평행한 진행을 해 왔다. 고속, 고정확도, 많은 테스트 집합, 그리고 증가하는 테스트 패턴으로 인하 메모리에 대한 요구는 ATE 기술 향상에 따른 비용 감소의 폭을 감소시키고 있다. 보다 많은 BIST와 DFT 기술을 접목한 설계상의 변화가 없는 한 이러한 추세는 $20M까지 진행될 것이다. 2012년쯤에는 트랜지스터를 제조하기 위한 비용에 비해 이를 테스팅하기 위한 비용이 더 크게 될 것이다.

3) 테스팅 복잡도를 관리하기 위한 새로운 devide-and-conquer 기술의 개발이 필요하다. 테스팅 문제를 자동화와 표준화를 사용하여 관리 가능한 문제들로 세분화하지 않으면 테스팅은 받아들일 수 없을 정도로 느려질 것이다.

4) 코어의 재사용은 테스팅의 재캡슐화와 재사용 또한 필요로 한다. 코어 테스팅을 위한 인터페이스와 접근 방법이 필요하다. 그리고, 아날로그 코어를 포함한 여러 개의 코어로 구성된 하나의 칩을 테스팅하기 위한 각각의 코어 테스트 방법의 조합 방법에 관한 연구가 필요하다. 코어들 사이의 연결선에 대한 테스트 방법의 개발이 필요하고 코어가 내장된 경우 거대한 칩 안에서 전원, 접지, 그리고 연결선의 잡음에서도 예상된 기능을 수행하는가를 보장하기 위한 테스팅 방법이 필요하다.

더구나 내장된 코어의 테스트를 위해서는 오늘 가장 널리 사용되고 있는 방법과 달리 보다 테스터 독립적인 테스팅 방법의 개발이 필요하다.

5) BIST는 초심 설계자에 의해 짧은 설계 사이클 환경에서 사용될 수 있도록 만들어져야 한다. 논리 회로 BIST 방법은 반드시 난이(hard-to-target) 고장과 필요한 모든 종류의 고장을 테스트할 수 있는 고장 검출률을 보장할 수 있도록 해야 한다.

6) 신호 통합과 전자기 현상은 집적 회로와 테스트 장비가 복잡해짐에 따라 점점 중요한 테스팅 문제로 부각될 것이다. 전자기장의 영향을 포함하는 새로운 고장 모델이 개발되어야 한다. 다양한 설계 부문에서 설계 조건, 제조성, 그리고 테스트 용이도 사이의 관계성이 개발돼야하고 테스트 패턴 생성기는 신호 통합 문제들을 고려해야 한다.

7) 타이밍 테스팅 문제는 연결선의 지연, 저속 합성 시스템 드라이버, 높아가는 주파수, 클럭 skew에 의해 특히 중요하게 부각된다. ATPG는 임계 경로에 근접한 많은 수의 경로들을 수용할 수 있어야 하며 BIST 시스템은 타이밍 고장에 대한 높은 고장 검출률을 보장해야 할 것이다.

8) HW/SW co-design은 테스팅을 위해서나 테스팅 자체 그대로 사용되기 위한 시스템 소프트웨어를 위한 새로운 기회를 제공할 것이다. 모든 높

〈표 3〉 테스팅의 당면 기술적 과제

테스팅 분야	연구 과제 요약
BIST와 DFT	DFT와 BIST 기술의 발전 없이는 테스트 장비의 비용은 $20M가 되고 수율은 0으로 떨어질 것이다. 주사기법과 BIST는 20년 동안은 주된 DFT 기법으로 남을 것이지만 조절과 관측을 위한 새로운 DFT가 필요하다.
ATE	높은 주파수와 많은 핀을 테스트하기 위한 탐침과 테스트 소켓 기술 개발이 필요하다.
혼합 신호	보다 큰 밴드폭, 샘플링 주파수, 낮은 잡음이 필요한 RF와 아날로그 회로가 많은 수의 디지털 회로를 포함한 경우의 테스팅 기술 개발이 필요하고 혼합 신호 테스팅 개발 시간을 줄여야 한다. 이를 위해 아날로그 DFT 및 BIST에 대한 연구가 필요하다.
IDDQ 테스팅	백만개 이상의 트랜지스터를 포함한 경우에는 이용하기 힘들게 된다. 따라서 장애 분석을 위해 확장되거나 대체 기술을 개발하거나 회로분할 및 내장된 전류감지기의 개발이 필요하다.
고장 모델	진보된 다층 금속 배선(multi metal layer)의 사용으로 새로운 고장 모델이 필요하다. 기존의 고착 고장은 효과적이지 못하게 될 것이다. 순수한 타이밍 및 신호 통합 문제가 규격히 상승하여 새로운 형태의 칩 장애가 발생하고 있으므로 새로운 종류의 고장이 반드시 신속하게 정의돼야 하고, 그들을 위한 테스팅 방법이 개발돼야 한다.
테스팅 소프트웨어	자동으로 테스팅 프로그램과 DFT의 정확성을 판별할 수 있는 도구 및 규격의 개발이 필요하다. 회로 합성 시스템은 많이 사용되고 있는 고장 모델에 대한 높은 고장 검출률을 갖는 ATPG 기능을 포함해야 한다. 메모리, 데이터 경로, 그리고 그 밖의 회로 종류를 위한 회로 합성 시스템은 완전 테스팅 자동화를 제공해야 할 것이다
테스팅 표준화	테스팅 도구의 공통된 정의가 필요하다. 최소의 노력으로 테스팅을 다른 도메인으로 이동할 수 있는 명명법을 개발해야 한다. 고장 모델과 이들 모델에 대한 고장 검출률의 표준화가 필요하다. 따라서 고장 검출률을 표현하기 위해 공통의 언어를 사용해야 한다.
장애 분석 및 진단	다층 금속 배선 공정에서 결함을 제거하기 위한 3차원 CAD 및 공장자동화 시스템 개발이 필요하다. 장애 분석과 수율 향상 사이클 시간을 줄이려면 진단을 위한 설계와 합성이 포함되어야 한다.

은 수준의 설계 방법론들은 특별한 테스팅 문제와 알 수 없는 고장 정보를 가진 미지의 라이브러리와 공정을 항상 고려해야 한다.

9) 아날로그/RF 시스템은 계속해서 테스팅에 주된 문제점이 될 것이다. 왜냐하면 아날로그 신호는 다양한 인코딩을 사용하여 데이터를 표현하기 때문이다. 또한, 의미 있는 고장 모델을 만들기 위한 단순화에 내재된 어려움이 있어 응용 종류는 각각 유일한 것으로 취급해야 한다.

10) 위에서 언급한 외에 테스트용이화를 위한 회로의 추가에 의한 시스템 성능 저하, 테스팅 중에 발생하는 전력 소비에 대한 고려가 필요하다. 또한 테스팅 프로그램의 동작이 soft error를 발생시킬 수 있는 과도한 잡음을 유발해서는 안된다.

이러한 문제를 해결하기 위한 기술적 과제는 〈표 2〉와 같다.

V. 결 론

지금까지 살펴 본 바와 같이 반도체 산업의 발전은 테스팅에 대한 연구와 기술 개발이 필수적인 것을 알 수 있다. 지금까지는 주로 테스팅 문제들이 비용과 성능에 관련된 것이지만 앞으로는 테스팅이 반도체 산업에 새로운 기능을 수행하게 될 것이다. 현재도 제조 공정상의 결함을 분석하기 위한 가장 좋은 도구는 테스팅 과정에서 사용되는 장비와 소프트웨이다. time-to-yield, time-to-money, time-to-quality, 그리고 time-to-market은 모두 테스팅에 의해 많은 부분이 결정되고 있다. 테스팅을 수행하는 과정에서 필요한 피드백 과정은 오늘날 공정상의 많은 수의 결함을 분석하고 분리하기 위한 유일한 방법이다. 따라서 테스팅 기술은 제조된 칩중 불량품을 제거하는 기능에 추가하여 비용에 효율적인 공정 평가와 결함 분리를 지원하여 제조된 칩과 제조 공정의 경쟁력을 갖추기 위한 중용한 기술로서 그 역할을 수행할 것이다.

참 고 문 헌

[1] The Nationale Technology Roadmap for Semiconductors-Technology Needs 1997 Edition, Semiconductor Industry Association, 1997.

[2] 홍성제, 박은세, 강성호, 최호용, 장 훈, 테스팅 및 테스터을 고려한 설계, 홍릉출판사, 1998.

[3] M. Abramovici, M. A. Breuer and A. D. Friedman, Digital Systems Testing and Testable Design, Computer Science Press, NY, 1990.

[4] Standard Test Access Port and Boundary-Scan Architecture, Sponsored by Test Technology technical Committee of the IEEE Computer Society, Document P1149.1 /D5(Draft), June 20, 1989.

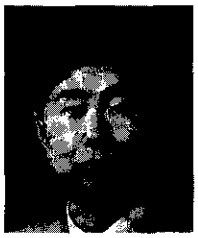
[5] V. D. Agrawal, C. R. Kime and K. K. Saluja, "A Tutuorial on Built-In Self-Test," IEEE Design & Test of Computers, March, 1993.

[6] W. Mao, R. Gulati, D. Goel and M. Ciletti, "QUIETEST : A quiescent current testing methodology for detecting leakage faults," Proc. ICCAD, 1990.

[7] E. Hsieh et al., "Delay Test Generation," Proc. DAC, 1977.

[8] A. J. Van de Goor, Testing Semiconductor Memories, John Wiley & Sons, 1996.

저 자 소 개



姜 成 昊

1963년 4월 13일생, 1986년 2월 서울대학교 제어계측공학과 학사, 1988년 5월 The University of Texas at Austin 전기 및 컴퓨터 공학과 석사, 1992년 5월 The University of Texas at Austin 전기 및 컴퓨터 공학과 박사, 1989년 11월~1992년 8월 Schlumberger Inc. Research Scientist, 1992년 8월~1994년 6월 Motorola Inc. Senior Staff Engineer, 1994년 9월~현재 연세대학교 기계전자공학부 조교수 〈주관심 분야: Testing and Testable Design, VLSI CAD, Design Verification, VLSI & System Design〉

姜 容 碩

1972년 7월 10일생, 1995년 2월 연세대학교 전기공학과 학사, 1997년 8월 연세대학교 전기공학과 석사, 1997년~현재 연세대학교 전기공학과 박사 과정, 〈주관심 분야: Testing and Testable Design, VLSI & CAD〉