특 집

SOC 테스트 기술 동향

강 성 호, 송 동 섭
연세대학교 전기전자공학과

I. 서 론

전자산업에서 점점 휴대가 용이하고 성능이 뛰어나면서도 값싼 제품을 원함에 따라서 칩 설계자들은 SOC(SOC : System-On-a-Chip)이라는 새로운 IC 설계의 시대를 준비하고 있다. 공정기술의 발전으로 하나의 완전한 시스템을 구성하는 모든 컴포넌트들을 하나의 칩으로 구현할 수가 있게 되었으며, 이를 위하여 CAD 업계에서는 IC 설계를 위한 새로운 방법론들을 개발하는 노력을 경주하고 있다. 기존의 SOB(SOB : System-On-a-Board) 설계 환경에서의 시스템은 인쇄회로기판(PCB : Printed Circuit Board) 상에 다수의 칩을 사용하여 구성되었고, 각각의 칩들은 규정된 기능을 수행하였다. SOC의 설계 환경에서는 보드상의 칩을 소위 IP(Intellectual Property)라고 하는 하나의 코어가 그 역할을 대신한다. 이런 설계 방법은 결과적으로 고성능, 저전력, 짧은 설계 시간, 저가의 설계·제조 비용 등의 장점을 가져 올 것이 예상된다.[1]

SOC의 설계는 기존의 칩 설계 기법과 비교할 때 여러 가지 특징을 갖는다. 하나의 칩에 여러 개의 코어를 내포함에 따른 복잡도의 증가로 인하여 테스트 비용이 늘어날 뿐만이 아니라 그에 따라 테스트 시간 또한 증가하게 되어 SOC를 효과적으로 테스트하는 문제가 중요하게 생각되어지는 것은 SOC의 설계 경향이 갖는 중요한 특징 중의 하나이다. SOC가 비록 SOB의 축소판으로 생각되어질 수 있지만 각각을 테스트하는 문제에 있어서는 많은 어려움이 파생된다. 초고집적 회로설계 기술, 설계 자동화 기술, 초미세 선폭 공정기술, 그리고 반도체 재료기술의 발전은 2GHz 이상의 클록 주파수, 1.2V 이하의 낮은 공급 전압을 갖는 고품질의 코어 생산을 가능하게 하였지만, 이런 코어를 테스트하는 문제에 있어서는 점점 더 오랜 시간이 필요하게 되고 결과적으로 고가의 테스트 비용을 초래한다. 왜냐하면 노이즈, 신호 지연, 그리고 간섭 등의 문제가 더욱 중요하게 생각되어 짐에 따라서 기존의 고장 모델과 이에 연관된 테스트 패턴 생성 툴이 적합하지 않게 되기 때문이다. 또한 코어 기반의 설계 경향에서 파생된 코어 제공자와 코어 사용자 계층 간의 테스트를 위한 정보의 원활하지 않은 소통은 SOC의 테스트를 어렵게 한다. 시스템 칩을 테스트할 때는 코어 내부에 사용된 테스트 용이화 설계에 대한 정보, 테스트 모드, 테스트 프로토콜, 고장 검출율, 그리고 테스트 패턴 데이터 등에 관한 정보를 필요로 하는데 이 두 계층 간의 표준화되지 못하고 또 충분하지 못한 정보의 공유는 전체 시스템의 효과적인 테스트를 불가능하게 한다. 그 다음 SOC를 테스트하는 데 발생하는 어려움은 칩의 입출력에서 코어의 입출력으로의 테스트에 필요한 접점을 얻기가 용이하지 않다는 점에 있다. 하나의 칩에 여러 개의 코어가 내장된 경우 각각의 코어에 대해서 테스트 용도의 핀을 부여하는 것은 불가능하고 최소한의 핀으로 칩의 내부 깊숙이 존재하는 코어에 대해서 테스트에 필요한 조절용이도(controllability)와 관측용이도(observability)를 얻는 과정이 가능해야 SOC를 효과적으로 테스트할 수 있다.[2]

IEEE P1500은 코어 기반의 칩들에 대해 테스

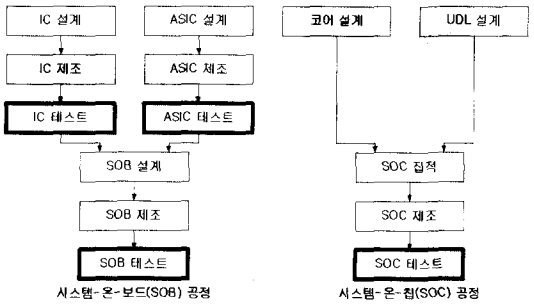
트의 관점에서 재사용을 수월하게 하고, 코어 제공자와 코어 사용자간의 상호처리 기준을 마련할 목적으로 결성된 IEEE 워킹 그룹이다. P1500에서는 SOC의 입출력으로부터 코어의 입출력까지의 테스트 접근을 얻기 위한 wrapper의 설계와 테스트 정보의 전달을 위한 CTL(Core Test Language)의 표준화 움직임을 보이고 있다. 대신에 코어 내부의 테스트 방법, 전체적인 SOC의 테스트 구조의 집적 및 최적화에 관한 내용은 SOC의 종류나 설계 특징에 따라 매우 상이한 일이기 때문에 논외 대상으로 한다.[3]

결과적으로 SOC를 효과적으로 테스트하기 위해서는 P1500의 표준과 맞물려 코어 사용자와 코어 제공자 각각의 충실한 역할 수행이 중요하다. SOC가 갖는 특징들은 테스트를 수행함에 있어서 여러 가지 어려움을 만들어 내고, 이는 코어 설계 단계에서 테스트 용이화를 고려하지 않으면 전체 시스템의 테스트가 불가능하게 되는 결과를 낳기도 한다. 코어 제공자는 기존의 코어 수준에서 고려되는 여러 가지 테스트 용이화 기법을 적용하여 테스트가 용이한 코어를 설계하고 코어 사용자에게 적절하게 기술된 형태의 전달물과 함께 제공한다. 코어 사용자는 이를 이용하여 설계하는 SOC의 특징에 따라 코어로의 테스트 접근을 효과적으로 얻을 수 있는 테스트 구조를 설계하고 전체적인 테스트 절차를 스케줄링 한다.

II. SOC 테스트 특징

1. 제조 공정에 따른 SOC 테스트 특징

SOC는 기존에 보드 상에 구현되는 시스템의 구성요소를 그대로 하나의 칩 안에 구현하는 기술을 의미한다. 그러므로 SOC의 설계는 많은 부분이 SOB와 공유할 수 있다. 하지만 각각을 테스트하는 문제에 있어서는 두 개의 설계환경이 커다란 차이점을 갖는다. 이것은 하나의 칩 안에 여러 개의 기능 블록들이 칩의 입출력에 대해서 깊숙하게 존재하여 테스트 접근이 어렵다는 SOC의 본질적인 특성이외에 각각이 갖는 제조 공정상의 차이점에 기인한다. <그림 1>은 각각 SOB 환경의 제조 공정과 SOC 환경의 제조 공정을 나타낸다.



<그림 1> 그림 SOB와 SOC의 제조 공정

<그림 1>에서 볼 수 있는 것과 같이 SOB의 설계 환경은 개별적인 기능요소들을 하나의 칩 또는 시스템 설계자의 의도에 의한 ASIC칩으로 설계하여 전체적인 하나의 보드에 집적되는 제조 공정을 갖는다. 테스트의 관점에서 SOB의 제조 공정이 갖는 특징은 다단계의 테스트가 이루어진다는 점이다. 보드에 사용되는 칩이나 ASIC 칩은 집적되기 전에 물리적인 공정을 사용하여 실제로 제조된다. 적당한 고장 모델과 높은 고장검출율을 가지고 칩 자체의 고장 유무에 관한 테스트를 거치며 이 칩을 사용하여 보드 시스템을 구성한 후에 다시 한번 시스템에 관한 테스트를 실시한다. 그러므로 보드 시스템을 테스트하는 시점에서 보면 각각의 구성요소인 칩들은 이미 고장이 없다는 가정이 가능하다. 실제로 SOB의 테스트 과정은 집적된 칩들의 고장 유무를 판단하는 과정보다는 보드자체 상호연결선의 고장유무를 판단하는 과정으로 집중된다. IEEE P1149.1인 경계 주사 구조는 사용자의 의도에 따라서 INTEST와 같은 칩의 내부를 테스트할 수 있는 명령어를 지원하고 있으나 이것은 의무적인 이행사항은 아니고 필요에 따라서 선택적으로 사용할 것을 권하고 있다. 대신에 보드의 상호 연결선을 테스트하는데 사용되는 EXTEST의 명령어는 반드시 지원해야 함을 규정하고 있다. 이 점을 보더라도 기존의 보드 수준의 시스템 환경에서는

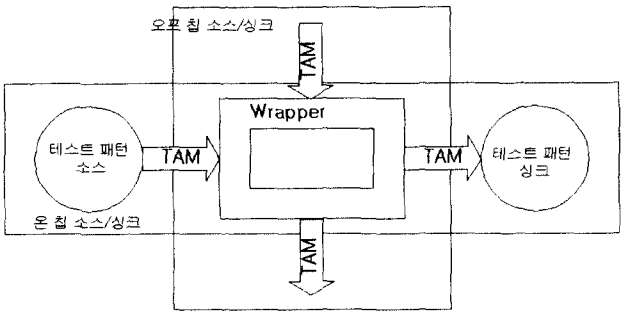
구성 요소간의 상호 연결선을 테스트하는 것이 중요한 관심사항인 것을 알 수 있다.[4]

SOC의 제조 과정에서는 실제적인 테스트의 과정이 모든 시스템이 구성되고 난 후에 단 한번 이루어진다. SOC의 개별 블록인 코어는 코어 제공자에 의해서 설계되고 이 설계물은 전달물의 형태로 코어 사용자에게 제공되어진다. 중요한 것은 이때의 설계물은 아직 기술 단계일 뿐이고 제조되거나 테스트되지 않은 상태라는 점이다. SOC의 제조는 모든 기능 블록들을 집적하고 난 후에야 가능한 일이고 테스트도 비로소 이때부터 가능한 것이다. 그러므로 SOC의 시스템을 테스트 할 때는 코어간의 상호 연결선뿐만이 아니라 코어 내부의 테스트 또한 중요한 의미를 갖는다. 이 점은 코어의 내부를 테스트하는데 필요한 테스트 벡터가 상호 연결선을 테스트하는데 필요한 테스트 벡터보다 그 양이 많기 때문에 테스트 시간을 고려하여 효과적인 테스트 벡터인가의 메커니즘을 필요로 한다. SOC의 테스트는 또한 고장 검출율, 테스트 비용, 시장 진입 시간 등의 전통적인 deep-submicron 칩이 갖는 테스트 문제는 여전히 갖고 있다.

2. SOC 테스트 특성과 일반적인 테스트 구조

SOC는 비록 인쇄회로기판의 축소된 형태로서 생각되어질 수 있으나 SOC를 테스트하는 문제에 있어서는 몇 가지 다른 성질을 갖는다. 그러므로 SOC를 테스트하기 위한 테스트 구조는 IEEE 표준 1149.1과 비교해 볼 때 독특한 테스트 구조를 필요로 한다. 이것은 SOC에 내재하는 모든 코어에 대해 별도의 테스트 용도의 입출력을 마련할 수 없는 핀 수의 제약, 칩의 입출력에서 모든 코어의 입출력에 직접 접근을 얻을 수 없는 코어 접근의 제약 등 SOC가 갖는 특징에 기인한다. 〈그림 2〉는 일반적으로 SOC가 갖추어야 할 테스트 구조의 개념적인 도식을 나타낸다.

테스트 패턴 소스(source)는 내장된 코어에 대한 테스트 벡터를 생산한다. 반대로 테스트 패턴 싱크(sink)는 테스트 결과를 예상된 결과값과 비교를 한다. 테스트 패턴 소스와 싱크는 외부의 자동 테스트 장치(ATE : Automatic Test Equipment)에 의해서, 온 칩의 내장된 자체 테스트 장치(BIST : Built In Self Test)에 의해서 또는 두 가지의 혼용에 의해서 구현될 수 있다. 온 칩, 오프 칩 혹은 두 가지의 혼용 중에 소스와 싱크가 구현되는 형태의 선택은 코어가 갖는 회로적 특성, 미리 정의된 테스트 벡터의 형태, 테스트 비용 등을 고려하여 결정한다. 기존 SOB의 시스템 환경에서 로직, 메모리, 아날로그 또는 혼성 신호의 테스트는 매우 다른 고장 검출 특성을 갖기 때문에 근본적으로 서로 상이한 테스트 특성을 가진다. 테스트 패턴을 생성하는 소스와 테스트 결과를 비교하는 싱크의 형태도 회로 기술에 따라 매우 다르기 때문에 기존의 자동 테스트 장치와 내장된 자체 테스트의 형태는 회로의 종류에 따라서 매우 다양한 형태를 가지고 발전하였다. 그러나 SOC의 출현으로 인해서 하나의 칩에 내장되는 코어의 회로 종류가 매우 다양해졌고 목적에 따라서는 이 세 가지의 회로 종

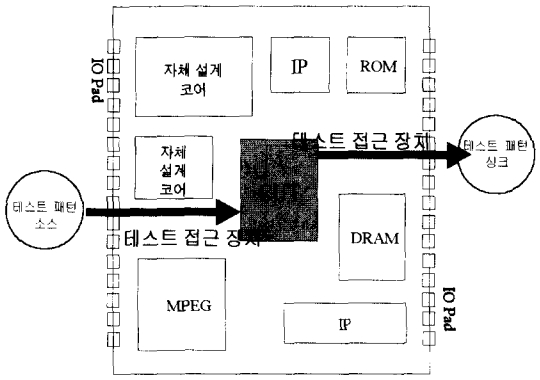


〈그림 2〉 SOC 테스트 구성 요소

류를 모두 하나의 칩에 포함하는 경우도 빈번하게 발생한다. SOC의 온 칩, 오프 칩 테스트 패턴 소스와 테스트 패턴 싱크는 기존 세 종류의 테스트 기술을 하나의 장치에 집적하는 형태를 갖추어야 한다.

전통적인 SOB와 비교해서 SOC의 설계 환경이 갖는 테스트 상의 커다란 차이점은 바로 칩의 입출력에서 코어의 입출력으로의 접근이 용이하지 않다는 점이다. 일반적으로 코어 제공자가 코어 설계물과 함께 제공하는 코어 테스트에 관한 정보는 코어의 입출력 단자의 관점에서 기술된다. 그러나 코어 사용자가 SOC를 제조한 후 전체 시스템을 테스트할 때에의 모든 테스트 과정은 칩의 입출력을 통하여 이루어진다. SOB에서 칩은 개별 단위로서 테스트되고, 테스트가 진행되는 동안에 칩의 물리적 입출력에 직접 접근이 가능하다. 이와 비교해서 시스템 칩의 내부 깊숙하게 자리잡은 코어는 SOC의 입출력에서 코어의 입출력으로 직접적인 물리적 접근이 불가능하다. 그러므로 SOC의 테스트에서는 시스템 칩의 입출력에서 코어의 입출력으로 접근이 가능하도록 전기적인 접근 장치가 필요하고 이것을 테스트 접근 장치(TAM: Test Access Mechanism)라 한다. 〈그림 3〉은 테스트 접근 장치의 사용을 보여 주고 있다.

SOC의 테스트가 기존의 설계 환경과 비교해서 갖는 특징 중의 하나는 다양한 테스트 동작을 필요로 한다는 점이다. 코어 테스트 wrapper는

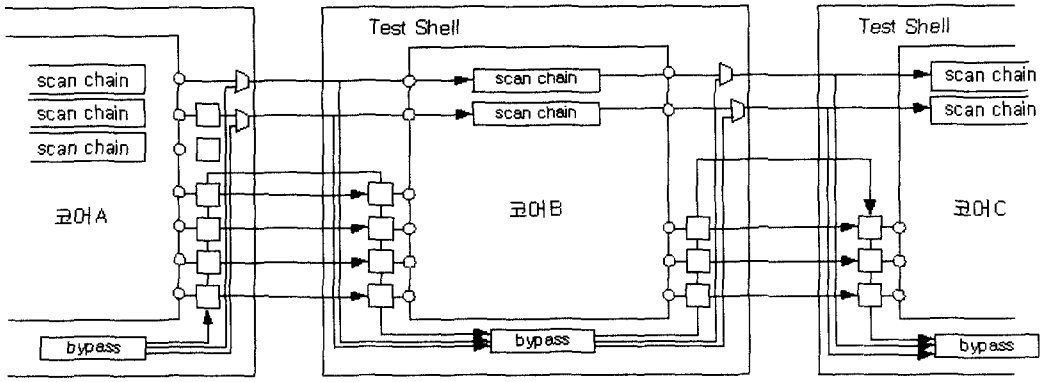


〈그림 3〉 SOC 테스트 접근 장치

내장된 코어와 SOC 환경 그리고 테스트 접근 장치와의 인터페이스를 담당하는 테스트 구조를 말하고 다음과 같은 테스트 모드간의 스위칭 메커니즘을 제공한다.

- 비 테스트 모드 : 비 테스트 모드에서 코어는 시스템 칩의 입출력에 연결되고 wrapper는 시스템 칩의 기능적 동작에 아무런 영향도 미치지 않는다.
- 코어 내부 테스트 모드 : 코어 내부 테스트 모드에서 테스트 접근 장치는 코어에 연결되고, SOC 입력으로부터의 테스트 입력이 코어의 입력으로 전달된다. 그리고 코어의 내부를 테스트 한 테스트 응답이 코어의 출력을 통하여 SOC의 출력에서 관찰된다.
- 코어 외부 테스트 모드 : 코어 외부 테스트 모드에서 테스트 접근 장치는 코어간의 상호 연결선에 연결되고 SOC 입력으로부터의 테스트 벡터를 코어의 출력에서 인가할 수 있다. 상호 연결선을 테스트 한 테스트 응답은 다시 코어의 입력을 통하여 SOC의 출력에서 관찰할 수 있다.
- 코어 바이패스 모드 : 테스트 대상이 아닌 코어에 대해서는 테스트 경로에서 제외되도록 한다.

Wrapper는 코어의 입출력 단자를 테스트 접근 장치에 연결한다. 이때 테스트 접근 장치의 데이터 폭은 코어의 입출력 폭과 일치하지 않을 수 있다. 왜냐하면 코어의 입출력 폭은 코어의 기능과 응용에 의해서 결정되는 반면에 테스트 접근 장치의 데이터 폭은 테스트 패턴 소스와 싱크의 데이터 폭, 혹은 테스트 접근 장치가 차지하는 실리콘 면적 등에 의해서 결정되기 때문이다. 만일 코어 입출력 단자의 폭이 테스트 접근 장치의 데이터 폭보다 크다면 코어의 입력에서의 데이터 직-병렬 변환, 코어의 출력에서의 데이터 병-직렬 변환이 wrapper에서 이루어져야 한다. 지금까지 발표된 wrapper의 예로서는 테스트 collar[5]와 TestShell[6]이 존재하고 〈그림 4〉는 테스트 TestShell을 나타낸다.



〈그림 4〉 Wrapper의 예 : TestShell

Ⅲ. IEEE P1500

IEEE P1500 워킹 그룹은 내장된 코어 테스트의 표준을 목적으로 한다. 표준화의 필요성은 코어 기반의 설계 및 테스트 환경이 코어 제공자와 코어 사용자 두 개의 그룹으로 분리되는 것에 기인한다. P1500은 코어 사용자와 코어 제공자 모두에게 효율성을 향상시키기 위하여 다양한 분야에서 코어 테스트의 상호 운용을 편리하게 하는 표준의 정의에 노력하고 있다. P1500은 코어의 내부 테스트 방법, 코어 내부에 존재하는 테스트 용이화 설계, 그리고 전체적인 시스템에 대한 테스트 구조의 집적 및 최적화에 관한 내용은 논외로 한다. 왜냐하면 이런 문제들은 SOC의 종류나 설계 특징에 따라서 매우 상이한 일이고 표준화를 통한 공통 분모를 찾기 어려운 문제이기

때문이다. 대신에 P1500은 테스트 시 plug-and-play의 용이함을 위한 제반 사항의 표준화에 초점을 맞추고 있다.

P1500은 크게 8개의 전략팀으로 구성되고 그 각각은 다음과 같다.

- Core Test Language 전략팀
- Scaleable Architecture 전략팀
- Compliance Definition 전략팀
- Terminology/Glossary 전략팀
- Documentation 전략팀
- Mergeable Cores Test 전략팀
- Benchmarking 전략팀
- Industry & Media Relations 전략팀

이 8개의 전략팀은 정기적인 미팅과 학술대회를 통하여 내장된 코어 테스트의 표준을 진행 중에 있으며 지금까지의 큰 흐름은 다음의 두 가지로 요약될 수 있다. 코어 테스트에 관련된 모든 사항을 기록하는 데 필요한 언어의 표준화를 통하여 코어 제공자와 코어 사용자간의 원활한 의사소통을 목적으로 하는 CTL(Core Test Language) 표준화 노력이 그 하나이고, 테스트 벡터를 코어 주변의 환경으로부터 코어의 입출력까지 효과적으로 전달하는데 필요한 표준적이고 유연적인 하드웨어를 제공하는 것이 목적인 Scalable Architecture 표준화 노력이 다른 하나이다.[7]

1. Core Test Language

CTL은 재사용 가능한 코어에 대한 테스트 관련 정보를 표현하는데 사용되는 언어이다. CTL은 코어를 SOC에 내장하는 시점에서 코어와 주변환경과의 테스트 인터페이스 상에 부여되는 테스트 요구와 테스트 제한을 포함하고, 코어의 테스트 구조를 제어하는데 필요한 직관적이고, 간결하고, 범용 적인 정보를 나타낸다. P1500 CTL은 IEEE 1450인 STIL(Standard Test Interface Language)에 기초를 두고 있다. STIL

은 특정한 칩의 테스트 벡터와 그에 따른 파형을 묘사하는데 사용된다. 코어 기반의 설계 환경에서는 다양한 코어 제공자와 코어 사용자가 전체 SOC 설계 공동체를 형성할 것으로 생각되기 때문에 코어 제공자는 코어 사용자가 테스트에 필요한 충분한 정보를 얻을 수 있도록 전달물을 제공해야 한다. 이 때, 이 전달물은 P1500 CTL로 표현된 다음의 사항을 포함해야 효과적인 테스트가 이루어질 수 있다.

- 스캔 사슬의 수, 길이 등의 코어에 포함된 테스트 용이화 설계
- 가능한 테스트 모드와 각 테스트 모드에 해당하는 테스트 프로토콜
- 메모리에 대한 마치(march) 알고리듬, BIST ready 코어에 대한 primitive polynomials을 포함한 테스트 패턴 데이터
- 고장 모델과 고장 검출율
- 스캔 사슬, 하드웨어 브레이크 포인트(break-point)등의 코어에 내장된 디버깅 용도의 하드웨어 정보
- 프로브(probe) 포인트의 물리적 위치 등의 고장 진단 정보

CTL은 다음과 같은 방법으로 기술된다. 최상위 수준에서 CTL은 코어가 지니고 있는 테스트 모드에 따라 모든 정보를 분류한다. 예를 들어, 두 개의 테스트 모드를 지원하는 코어에 대한 CTL 구조는 다음 〈그림 5〉와 같이 표현된다.

각 테스트 모드는 해당 테스트 모드에서의 코어 내부의 구성과 테스트 벡터의 정보를 포함하

```
Environment {
        CTL mode1 modeType {
                //Information about mode1
        }
        CTL mode2 modeType{
                //Information about mode2
        }
}
```

〈그림 5〉 P1500 Core Test Language 예

```
CTL modeName modeType {
        Internal {
                //Describe the information
                //about the core itself
        }
        PatternInformation {
                //Describe pattern
                //information for the mode
        }
        External {
                //Describe instructions to
                //the system integrator
        }
}
```

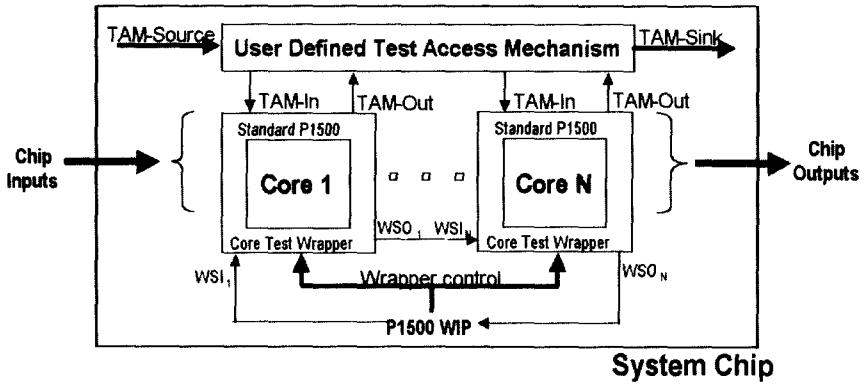
〈그림 6〉 P1500 Core Test Language의 예

여 코어의 주변 단자에서의 신호의 정적 상태를 묘사하도록 구성된다. 이 정보는 다음의 〈그림 6〉의 절차에 따라 각각 'Internal'과 'Pattern-Information'이라고 표시된 블록에 포함된다.

Internal 블록에서는 'Wrapper', 'Isconnected', 'IsEnabledBy'와 같은 키워드를 통하여 wrapper 정보가 포함되고, 'Electrucal-Property', 'DataRate', 'DataType'과 같은 키워드를 통하여 코어 신호의 특성이 묘사된다. PatternInformation 블록에서는 해당 테스트 모드에서 사용되는 모든 프로토콜을 포함하고, External 블록에서는 SOC로의 효과적인 코어 집적을 위하여 코어의 외부 환경 등의 코어 사용자에게 필요한 정보를 나타낸다.

2. P1500 Scalable Architecture

P1500 Scalable Architecture는 코어의 주변에 존재하여 코어와 테스트 접근 장치간에 다양한 스위칭 메커니즘을 제공하는 테스트 하드웨어이다. P1500 테스트 구조는 코어의 기능적 동작을 위하여 SOC 자체의 상호 연결을 유지하는 동작 이외에 테스트 모드에서 코어의 입력과 출력을 1비트의 직렬 테스트 접근 장치에 연결하거나 한 번에 여러 비트의 테스트 벡터를 인가하기 위한 병렬 테스트 접근 장치에 연결하는 동작을

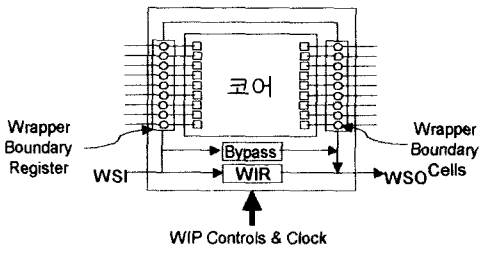


〈그림 7〉 P1500 테스트 구조를 적용한 SOC

수행한다. 〈그림 7〉은 P1500 테스트 구조를 적용한 SOC의 예를 보여 준다. 그림에서 볼 수 있는 것과 같이 P1500의 코어 테스트 구조는 wrapper, 테스트 제어 메커니즘, 테스트 접근 메커니즘의 3가지로 구성된다.

Wrapper는 코어 테스트 데이터 접근과 코어 테스트 고립 메커니즘을 제공하는 테스트 하드웨어이다. Wrapper는 코어의 입력에 대해서는 조절용이도를, 코어의 출력에 대해서는 관측용이도를 향상시켜 SOC의 입출력으로부터 코어 내부 테스트를 가능하게 한다. 또, 테스트 대상이 아닌 코어에 대한 wrapper는 주변의 테스트 대상 코어로부터의 테스트 고립 기능을 제공함으로서 예기치 못한 코어 손상을 방지한다. 〈그림 8〉은 wrapper를 적용한 코어의 예를 보여 준다.

테스트 제어 메커니즘은 SOC에 존재하는 테스트 모드와 테스트 신호를 제어하는 것을 목적으로 한다. P1500의 테스트 제어 메커니즘은 크게 직렬 테스트 접근 장치를 통한 테스트 제어와 병렬 테스트 접근 장치를 통한 테스트 제어로 나눌 수 있다. 직렬 테스트 제어는 SOC 내에 존재하는 하나의 테스트 경로 제어를 의미한다. 직렬 테스트 경로는 근본적으로 많은 쉬프트 이동으로 인하여 긴 테스트 시간을 요구하므로 직렬 테스트 제어는 동일 테스트 모드에서 자주 변하지 않는 테스트 신호의 제어를 목적으로 한다. 예를 들어 코어에 존재하는 테스트 자원을 코어 내부 테스트, 혹은 코어간의 상호 연결선 테스트 등의 테스트 목적에 따라 테스트 구성하는 과정은 하나의 테스트 동작에서는 일정하게 유지되므로 이 경우에는 직렬 테스트 제어를 통하여 테스트 제어를 달성한다. 이에 비해서 병렬 테스트 제어는 테스트 동작 중에 빈번하게 변화하는 테스트 신호를 제어한다. 예를 들어 scan_Enable 신호나 테스트 클록 신호가 이에 해당한다.



〈그림 8〉 P1500 wrapper 구조를 적용한 코어

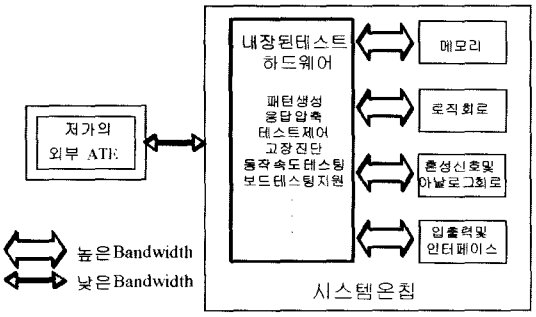
테스트 접근 메커니즘은 SOC의 입출력으로부터 코어의 입출력까지 테스트 데이터를 전달할 경로를 제공한다. 테스트 접근 메커니즘은 SOC 설계 환경, 가용한 제조 공정, 실리콘 면적 등에 따라서 매우 상이하게 적용될 수 있다. 즉, 모든 SOC에 일률적이고 공통적인 표준의 사항과는 거리가 있다. 그러므로 P1500은 테스트 접근 메커니즘에 대해서는 표준화 과정에서 제외하고 코어 사용자, 즉 SOC 제조자의 정의에 의하여 유동성 있는 적용이 가능하도록 유도하고 있다.

3. Dual Compliance Concept

P1500은 SOC에 내재되는 코어에 대해서 다음의 두 가지로 종류로 구분하고 있다. 그 두 가지는 P1500 Compliance 코어와 P1500 Ready 코어이다. P1500 Compliance 코어는 wrapper 디자인을 포함하고 wrapper의 단자 관점에서 CTL이 기술된다. 반면에 P1500 Ready 코어는 코어 설계 단계에서는 P1500 wrapper를 포함하지 않으며 코어에 대한 CTL 기술은 wrapper 단자가 아닌 코어 단자의 관점에서 기술된다. 그러므로 코어 사용자는 수작업으로 혹은 툴을 통하여 P1500 Compliance 코어로 전환할 수 있다. 결국 P1500에서는 코어를 SOC에 탑재 할 때 P1500 Compliance하게 되어야만 테스트 구조의 이점을 극대화 할 수 있음을 강조하고 있다. 그럼에도 불구하고 두 가지 코어를 구분하는 이유는 코어 사용자에 대하여 테스트의 유연성을 제공하기 위함이다. 즉 많은 부분을 설계물의 특성에 따라서 적당한 테스트 접근을 달성하고 최적의 테스트 스케줄링이 가능하도록 유도하고 있다. 이 경우 SOC의 환경에 따라 특별한 파라미터를 갖는 wrapper의 사용이 가능하거나 코어가 SOC 내에 자리잡는 위치에 따라서 적절한 route 시도를 할 수 있다는 장점이 제공된다. 이상으로 보아 P1500이 표준으로서 자리를 잡게 되면 지금의 많은 CAD 툴들이 IEEE 1149.1 테스트 구조의 합성을 지원하는 것과 같이 다양한 Ready-to-Compliance 자동 툴이 생겨날 것이 예상된다.

Ⅳ. SOC 테스트 기술

SOC는 하나의 칩에 로직, 내장된 메모리, 그리고 아날로그 회로와 같이 다양한 회로 기술을 포함한다. 또 칩의 집적도가 향상될수록 내장된 FPGA, 플래시 메모리 등의 진보된 회로 기술이 이에 포함될 것이 예상된다. 이런 혼합된 회로 기술이 포함된 SOC를 테스트하기 위한 쉽고도 효과적인 방법은 로직, 내장된 메모리, 그리고 아날로그 회로 등의 각 전자회로 기술 형태에 따라 테스트에 필요한 하드웨어를 SOC의 내부에 삽입하는 것이다. 이 경우에는 각 회로 기술에 따라 복수개의 외부 자동 테스트 장치(ATE: Automatic Test Equipment)를 필요로 하지 않고 〈그림 9〉와 같이 저가의 단일 자동 테스트 장치만으로 충분하다.



〈그림 9〉 SOC의 자동 테스트 장치

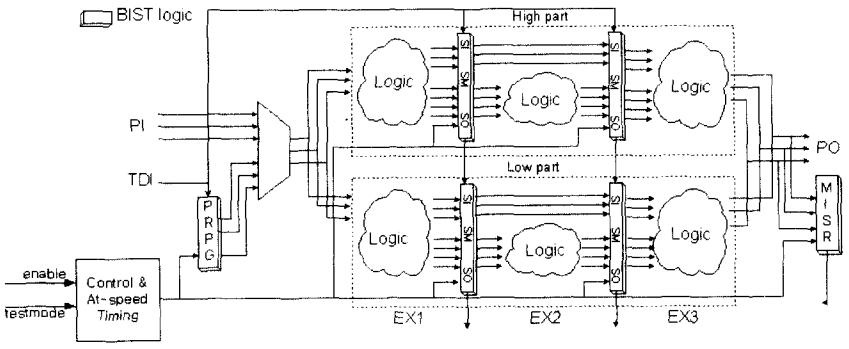
SOC 설계 경향은 기존의 IC 디자인 그룹을 다양한 코어 제공자 그룹과 코어 사용자 그룹으로 분리한다. 이로 인해 코어 제공자가 코어 사용자에게 제공하는 전달물의 측면에서나, 기본적인 SOC 테스트 구조에서 표준화의 필요성이 대두되었고 P1500은 이런 취지에서 결성된 IEEE 워킹 그룹이다. 코어 기반의 설계를 테스트하기 위한 현재까지의 진행 결과를 보면 여러 가지 SOC 테스트의 특성들 중에 부분적으로는 P1500의 표준에 의한 해결을 모색하는 한편, 다른 부분들은 코어 제공자와 코어 사용자에게 수정의 여지를 남겨 놓아 테스트의 유연성을 제공하고 있다. 효과적인 SOC 테스트를 위해서 코어 제공자는 코어 설계 시 테스트 용이화를 위한 설계를 고려해야 하고, 이를 코어 사용자에게 전달할 때에서는 적당한 기술을 통하여 테스트에 필요한 충분한 정보를 제공해야 한다. 또한 코어 사용자는 코어 제공자의 전달물을 이용하여 SOC의 특징에 맞게 최적의 테스트 접근 방법과 테스트 스케줄링을 개발한다.

1. 다양한 회로 기술에 따른 SOC 테스트 용이화 설계

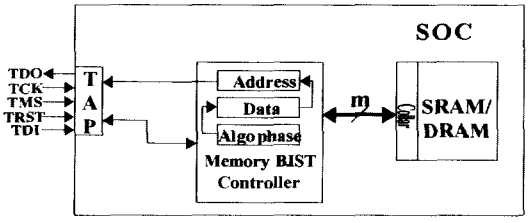
SOC 제조 과정 특징으로 인하여 코어 제공자는 코어 내부 테스트를 실시하지 않는다 하더라도 코어의 설계 단계에서 SOC를 테스트하는데 용이하도록 준비를 하여야 한다. 코어의 종류에 따른 대표적인 테스트 용이화 설계는 다음과 같다.

우선, 내장된 로직 블록의 테스트는 자체 테스트 기능을 가진 스캔 회로의 확장에 기초를 두고 있다. 이 내장된 자체 테스트(BIST : Built-In Self Test)는 테스트 벡터 발생기로서 PRPG (Pseudo Random Pattern Generator)와 테스트 결과 압축기로서 MISR(Multiple input signature register)에 의하여 수행되는 것이 일반적이다. 내장된 로직 코어에 대한 자체 테스트 회로는 완전한 동작속도에서 테스트 가능하여야 한다. 하나의 완전한 시스템 칩에 사용되는 로직 블록들은 그 목적에 따라 다양한 동작 주파수를 가지고 구현된다. 그러므로 SOC의 로직 BIST는 서로 다른 주파수에서 동작하는 클록 도메인들에 대하여 인터페이스를 제공하고 이를 통하여 각 로직 블록의 동작 주파수에서 테스트 가능함이 중요한 테스트 요구로 대두된다. 이는 SOC에 존재하는 모든 로직 코어에 대하여 동시에 테스트를 가능하게 하여 전체 테스트 시간을 줄일 수 있기 때문에 중요하다. 다음의 〈그림 10〉은 로직 코어들에 대한 내장된 자체 테스트의 예를 보여 준다.

미래의 SOC는 대용량의 고밀도 메모리를 내장할 것으로 기대된다. 이러한 고밀도의 메모리는 SRAM, DRAM, 그리고 Flash 메모리 등에 적용된다. SOC에 사용되는 내장된 메모리를 위한 테스트 구조 또한 테스트 소스와 테스트 싱크가 칩 내에 존재하는 자체 테스트가 바람직하다. 내장된 메모리의 자체 테스트 구조는 고장의 검출뿐만이 아니라 고장이 난 비트의 분석과 자체 고장 수리의 능력이 중요시된다. 메모리는 여분의 셀을 포함하고 고장이 발생한 메모리 셀과 교체하는 작업을 통해 재사용이 가능하도록 제조된다. 기존의 메모리 칩은 테스트 과정에서 발생하는 많은 양의 고장난 셀에 대한 정보를 외부의 자동 테스트 장치에 리포트하고 이후 레이저 퓨징을 통하여 이를 달성하였다. 그러나 SOC의 환경에서는 메모리만을 테스트하기 위한 전용의 자동 테스트 장치를 마련하는 것은 바람직하지 않고, 또한 테스트 용도의 입출력 핀에 대한 제한으로 인하여 외부로 전달할 수 있는 고장난 셀에 대한 정보에 한계가 있기 때문에 내장된 메모리의 자체 테스트 패턴 싱크는 고장난 셀에 대한 위치를 분석할 수 있는 능력이 중요하다. 즉, 내장된 메모리의 자체 테스트 구조는 메모리의 테스트 응답 벡터를 바탕으로 고장의 검출뿐만이 아니라, 메모리 재사용을 위하여 실제로 고장이 발생한 셀의 열과 행을 분석할 수 있도록 확장이 필요하다. 최근에는 고장 분석 후 고장 수리의 과



〈그림 10〉 로직 코어의 내장된 자체 테스트

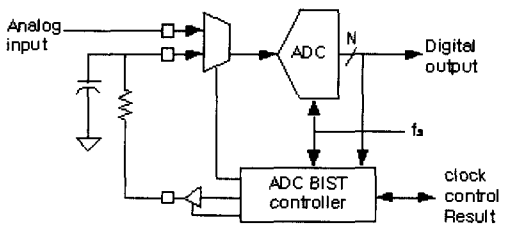


〈그림 11〉 내장된 메모리의 자체 테스트 장치

정 또한 고가의 외부 레이저에 의존하지 않고 SOC의 내부 테스트 패턴 소스와 싱크에서 가능하도록 하는 재구성 메카니즘에 대한 연구도 진행 중에 있다. 〈그림 11〉은 내장된 메모리 테스트 장치에 대한 예를 보여 준다.

내장된 혼성신호 코어에 대한 테스트 용이화 설계는 로직과 메모리에 대한 접근 방법과 많은 부분에서 유사하다. 혼성신호, 그 중에서도 아날로그 회로에 대해서는 테스트 용도의 하드웨어를 첨가하여 내부 노드를 변경하는 것은 코어 회로가 가지고 있는 특성 자체를 변화시키기 때문에 용이하지 않다. 그러므로 이 경우에 대해서는 테스트 입력을 가하고 그 결과를 관찰할 수 있는 자체 테스트 회로만으로 그 테스트를 실시한다. 다음의 〈그림 12〉는 그 개념을 보여 주고 있다.

혼성신호 코어를 위한 ADC BIST는 로직 회로와 같은 고착 고장 모델에 대한 테스트 입력을 가하기보다는 ADC 자체의 기능을 검증할 수 있는 사인파와 같은 기준 입력을 발생한다. 이에 대한 ADC의 응답 결과는 다시 ADC BIST의 입력으로 받아 들여져 기준 입력에 대하여 미리 계산된 여러 가지 예상값과 비교된다. 예를 들어 기준 입력에 대한 ADC 전압 출력의 최대값, 최소값 그리고 전류 출력의 최대값, 최소값 등을 측정하고 미리 예상되는 값과 비교를 통하여 고장 유무를 판단, 그 결과를 리포트 함으로서 혼성신호의 내장된 자체 테스트를 달성한다.



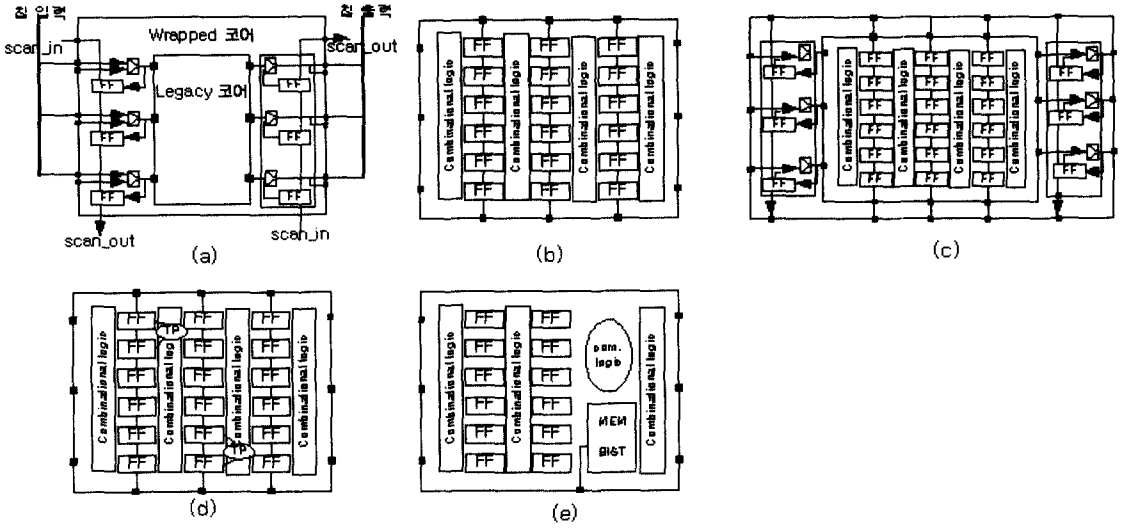
〈그림 12〉 혼성신호 회로의 자체 테스트 장치

2. 테스트 용이화 설계에 따른 코어의 분류

코어 제공자가 코어 설계 시에 고려한 테스트 용이화 설계는 전체 SOC 테스트 구조의 적용을 결정한다. 왜냐하면, 코어의 테스트 용이화 설계를 시스템 칩 수준에서 재사용하는 것은 빠른 시장 진입 시간의 SOC 설계 목적에 부합되기 때문이다. SOC에 내장되는 코어는 코어 설계 시에 고려된 테스트 용이화 정도에 따라서 분류가 가능하며 〈그림 13〉은 다양한 코어의 종류를 나타낸다.

먼저 Legacy 코어는 일반적으로 레이아웃(layout)의 형태로 제공되는 하드 코어를 의미하며 코어 제공자에 의하여 테스트에 관한 아무런 정보도 제공되어지지 않은 경우이다. 즉 코어 사용자는 SOC를 구성할 때 코어의 테스트 벡터를 알고 있지 못하며 또 설계물의 수정이 용이하지 않기 때문에 어떤 테스트 용이화 설계도 코어에 삽입할 수 없다. 그리고 코어 내부 구성에 대한 정보도 충분하지 않기 때문에 ATPG에 의한 테스트 벡터 산출도 가능하지 않다. Legacy 코어에 대한 SOC 수준의 테스트 구조는 칩의 입출력에서 코어로 직접 접근이 가능하도록 코어 주변에 wrapper를 구성한다. Legacy 코어에 대한 테스트 과정은 코어의 기능 벡터(functional vector)를 SOC의 입력에서 코어의 입력으로 직접 인가하는 과정으로 가능하다.

ATPG Enabled 코어는 설계 시 스캔을 적용하였고, 코어 테스트를 실시할 때 주변의 로직과 연계되어 테스트가 가능한 mergeable 코어를 의미한다. 일반적으로 RTL 수준으로 기술된 소프트 코어나 게이트 수준으로 기술된 펌 코어가 이에 해당한다. ATPG Enabled 코어는 테스트 시 UDL과 연계되어 동시에 테스트가 가능하기 때문에 코어 사용자는 UDL과 ATPG Enabled



〈그림 13〉 코어 테스트 용이화 설계에 따른 내장된 코어의 분류
(a) Legacy 코어 (b) ATPG Enabled 코어 (c) Reusable ATPG 코어
(d) BIST Enabled 코어 (e) Embedded BIST 코어

코어를 하나의 테스트 단위로 가정이 가능하고 이 merge된 테스트 단위에 대하여 ATPG를 실시한다.

Reusable ATPG 코어는 코어 설계시에 스캔 구조를 적용하였으나 다른 로직과는 연계되어 테스트될 수 없는 non-mergeable코어이다. ATPG에 의한 테스트 벡터도 코어 제공자에 의하여 제공되어 진다. Reusable ATPG 코어는 주변의 블록과는 동시에 테스트 될 수 없기 때문에 독립된 테스트 단위를 갖는다. 그러므로 SOC의 테스트구조 또한 Reusable ATPG 코어만의 wrapper 구조를 필요로 한다. 코어 제공자에 의한 테스트 벡터가 테스트 시 재사용된다.

BIST Enabled 코어는 코어 사용자에 의해서 내장된 자체 테스트를 구성할 수 있도록 모든 준비를 마련한 코어를 의미한다. 즉 코어 제공자에 의해서 테스트 벡터 발생기나 테스트 결과 압축기 등은 설계되지 않았지만 스캔 구조를 포함하고 있으며 의사 무작위 패턴(pseudo-random vector)에 의해서 테스트가 달성되지 않는 부분에 대해서는 테스트 포인트를 설정한다. BIST Enabled 코어가 갖는 의미는 내장된 메모리처럼 내장된 자체 테스트 구조를 공유할 수 있는 SOC 테스트 구조가 가능하다는 점이다. 즉 코어 사용자는 하드웨어 오버헤드와 테스트 시간의 상호 절충관계를 고려하여 최적의 구성이 가능하도록 다수의 BIST Enabled 코어에 대한 내장된 자체 테스트 제어기의 구성이 가능하다.

Embedded BIST 코어는 BIST Enabled 코어와는 달리 코어 내에 내장된 자체 테스트 제어기, 테스트 벡터 발생기, 테스트 결과 압축기 등의 모든 테스트 구조를 포함한다. 즉 내장된 자체 테스트를 실시함에 있어서 다른 Embedded BIST 코어나 BIST Enabled 코어와 적절한 테스트 스케쥴링에 의한 하드웨어 공유는 발생하지 않는다. 그러므로 이 코어에 대한 SOC 테스트 구조는 SOC의 입력에서 BIST 제어를 달성할 수 있도록 테스트 접근이 필요하다.

3. SOC 수준의 테스트 용이화 설계

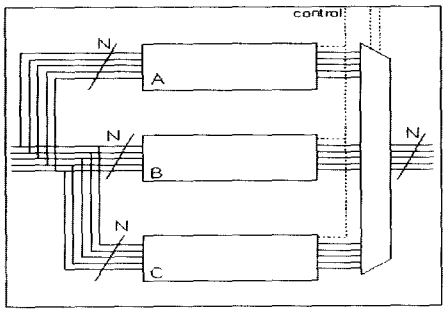
코어 기반의 설계 환경에서 전체적인 시스템의 테스트 구조는 SOC에 내재하는 코어의 수, 각

코어의 입출력 수, SOC의 가용한 입출력 핀 수 등의 여러 가지 요인에 의해서 최적화 된 형태를 달리한다. 코어 사용자, 즉 SOC 제조자는 이런 시스템 칩의 특징을 이해하고 전체적인 테스트 구조를 형성하는 역할을 담당한다.

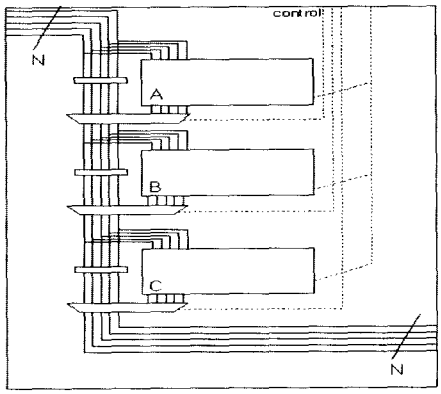
SOC 테스트 구조에서 테스트 접근 장치는 테스트 패턴 소스와 코어, 그리고 코어와 테스트 패턴 싱크간 물리적 거리간의 가교 역할을 한다. 코어에 대한 테스트 접근을 얻는 과정은 SOC 설계가 갖는 특징에 따라서 다음의 3가지 구성으로 구현될 수 있고, 코어 사용자는 이를 고려하여 테스트 접근 장치의 구조를 설계한다.[8]

멀티플렉싱 구조는 모든 코어가 최대의 테스트 접근 폭을 형성하도록 구성되며 각 코어는 동일한 SOC의 핀에 시간적으로 선택되어 연결된다. 데이지 체인 구조에서 모든 코어는 동일한 테스트 접근 폭을 가지며 하나의 테스트 접근 장치로 사슬 연결된다. 선택적인 멀티플렉서는 테스트의 목적에 따라서 테스트 대상이 아닌 코어에 대해 바이패스 할 수 있도록 한다. 분산 구조에서 각 코어는 각각의 테스트 접근 폭을 갖는다. 즉 유용한 총 테스트 접근 폭은 각 코어의 테스트 벡터 양에 따라 각각의 내장된 코어에 분배된다.

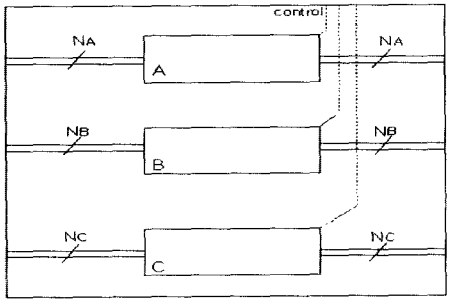
SOC 테스트를 위한 테스트 접근 장치는 내장된 코어의 특성과 SOC 자체의 특성을 고려하여 선택하여야 한다. 다양한 테스트 접근 장치 구조 중에 하나를 선택하는 일은 테스트의 질과, 실리콘 면적 등과 같은 설계비용, 그리고 자동 테스트 장치, 테스트 시간과 같은 테스트 적용 비용간의 상호 절충과정으로 이해될 수 있다. 만일 어떤 코어는 비교적 그 크기가 작고 혹은 내장된 자체 테스트 장치를 포함하고 있어 외부에서 인가해야 할 테스트 벡터의 크기가 작다면 이 코어에 대한 테스트 접근 장치의 폭은 최소화 할 수 있을 것이다. 반대로 어떤 코어는 많은 양의 테스트 벡터를 필요로 한다면 하나의 비트만의 테스트 접근은 테스트 비용 면에서 적합하지 않고 가능하면 최대한의 테스트 접근 폭을 얻을 수 있는 구조가 적합할 것이다.



(a)



(b)



(c)

〈그림 14〉 그림 TAM 구성 예
(a) 멀티플렉싱
(b) 데이지 체인
(c) 분산 구조

V. 결 론

SOC의 설계 시 내장된 코어의 재사용은 점점 더 증가되고 있다. 이러한 코어 기반의 SOC 설계에서 코어의 설계 시에 포함된 테스트 용이화 설계 구조를 재사용하는 것은 매우 바람직하다.

일반적으로 SOC 제조 과정에서 있어서 중요하게 생각되어지는 것 중의 하나는 테스트가 전체 디자인의 개발 시간에 있어 병목 현상으로 작용하여서는 안된다는 것이다. 짧은 시장 진입 시간을 목적으로 하는 코어 기반 설계 환경에 있어서 이 문제는 더욱 부각될 것이다. 코어 기반의 설계를 통하여 제조되는 SOC를 테스트하는 문제는 기존의 SOB 환경의 시스템을 테스트하는 것과는 다른 몇 가지 특징을 갖는다. SOC 설계 환경에서 파생된 코어 제공자와 코어 사용자의 두 그룹은 SOC를 설계하는 문제에 있어서 뿐만이 아니라 효과적으로 SOC를 테스트하는 문제에 있어서도 각각의 역할이 존재한다. SOC가 갖는 특징들은 테스트를 수행함에 있어서 여러 가지 어려움을 만들어내고, 이는 코어 설계 단계에서 테스트 용이화를 고려하지 않으면 SOC의 테스트가 불가능하게 되는 결과를 낳기도 한다. 코어 제공자는 기존의 칩 수준에서 고려되는 여러 가지 테스트 용이화 기법을 적용하여 테스트가 용이한 코어를 설계하고 코어 사용자에게 적절하게 기술된 형태의 전달물과 함께 제공한다. 코어 사용자는 이를 이용하여 설계하는 SOC의 특징에 따라 코어로의 테스트 접근을 효과적으로 얻을 수 있는 테스트 구조를 설계하고 전체적인 테스트 절차를 스케줄링 한다. 결과적으로 SOC를 효율적으로 테스트하기 위해서는 코어 제공자가 테스트 용이한 코어를 제공하는 것 못지 않게 코어 사용자가 이런 특징들을 올바르게 이해하고 전체적인 시스템 테스트 전략을 세우는 것이 중요할 것이다.

참 고 문 헌

[1] R. K. Gupta and Y. Zorian, "Introducing Core-Based System Design", IEEE Design & Test of Computers, Oct.-Dec., pp.15-25, 1997

[2] E. J. Marinissen, Y. Zorian, R. Kapur, T. Taylor, and L. Whetsel, "Towards a Standard for Embedded Core Test : An Example", IEEE International Test Conference, pp.616-627, 1999

[3] IEEE P1500 Web Site, http://grouper.ieee.org/groups/1500

[4] IEEE Computer Society. IEEE Standard Test Access Port and Boundary-Scan Architecture-IEEE Std. 1149.1-1990. IEEE, New York, 1990

[5] P. Varma and S. Bhatia, "A Structured Test Re-Use Methodology for Core-Based System Chips," IEEE International Test Conference, pp.294-302, 1998

[6] E. J. Marinissen, "A Structured And Scalable Mechanism for Test Access to Embedded Reusable Cores," IEEE International Test Conference, pp. 284-293, 1998

[7] E. J. Marinissen, R. Kapur, Y. Zorian, "On Using IEEE P1500 SECT for Test Plug-n-Play", IEEE International Test Conference, pp.770-777, 2000

[8] Y. Zorian, "System-Chip Test Strategies", Proc. of Design Automation Conference, pp.752-757, 1998

저 자 소 개

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 전기 및 컴퓨터공학과 졸업(공박), 1989~1992 : Schlumberger Inc. Research Scientist, 1992~1994 : Motorola Inc. Research Scientist, 1994~1999 : 연세대학교 공과대학 기계전자공학부 조교수, 1999~현재 : 연세대학교 공과대학 기계전자공학부 부교수, 〈주관심 분야 : 테스팅 및 DFT, CAD, SOC 설계〉

宋 東 燮

1974년 7월 24일생, 2000년 건국대 전기공학과 졸업, 2000년~현재 : 연세대 전기전자공학과 석사과정, 〈주관심 분야 : CAD 및 VLSI 테스팅〉