논문 2008-45SD-4-15

효율적인 SoC 테스트를 위한 온/오프-칩 버스 브리지 활용기술에 대한 연구

(Exploiting an On/Off-Chip Bus Bridge for an Efficiently Testable SoC)

송 재 훈*, 한 주 희*, 김 병 진*, 정 혜 란*, 박 성 주**

(Jaehoon Song, Juhee Han, Byeongjin Kim, Hyeran Jeong, and Sungju Park)

요 약

오늘날의 시스템-온-칩(SoC)은 짧은 제품 생산 주기를 맞추기 위하여 재사용 가능한 IP 코아들을 이용하여 설계한다. 그러나 고집적 칩을 생산하는데 있어 증가한 칩의 테스트 비용은 큰 문제가 된다. 본 논문에서는 Advanced High-performance Bus(AHB)와 Peripheral Component Interconnect(PCI) 버스를 위한 온/오프-칩 버스 브리지를 이용한 효율적인 테스트 접근 메커니즘을 제시한다. 본 기술은 독립적인 테스트 입력 경로와 출력 경로를 제공하고 버스 방향 전환을 위한 턴어라운드 지연 시간을 없앰으로써 테스트 시간을 매우 줄였다. 실험 결과는 면적 오버헤드와 기능적 · 구조적 테스트 모두 에서의 시간이 줄어들었음을 보여준다. 제안하는 기술은 다른 종류의 온/오프-칩 버스 브리지에도 적용 가능하다.

Abstract

Today's System-on-a-Chip (SoC) is designed with reusable IP cores to meet short time-to-market requirements. However, the increasing cost of testing becomes a big burden in manufacturing a highly integrated SoC. In this paper, we propose an efficient test access mechanism that exploits an on/off-chip bus bridge for the Advanced High-performance Bus (AHB) and Peripheral Component Interconnect (PCI) bus. The test application time is considerably reduced by providing dedicated test stimuli input paths and response output paths, and by excluding the bus direction turnaround delays. Experimental results show that area overhead and testing times are considerably reduced in both functional and structural test modes. The proposed technique can be applied to the other types of on/off-chip bus bridges.

Keywords : AMBA, bridge, PCI, SoC, testability

Ⅰ. 서 론

반도체 미세 공정 기술의 빠른 발전으로 인하여 기존에 사용되고 있던 여러 종류의 IP(Intellectual Property)를 이용한 SoC(System-on-a-Chip)의 설계 및 생산이 가능해 짐으로써 점점 더 짧아지는 제품 출시 기간을 만족 시킬 수가 있다. 설계 시간은 IP 코아를 재활용함으로써 줄일 수 있지만 칩의 테스트 시간은 SoC의 높은 복잡도로 인하여 증가하게 되었다. 테스트 비용은 메모리 크기, 테스트 장비(Automatic Test Equipment) 이용 시간, 코아 테스트 래퍼의 구조, 테스트 접근 메커니즘(Test Access Mechanism), 테스트 방법에 큰 영향을 받는다[1]. 최근의 실리콘 시장에서 낮은 테스트 비용으로 높은 테스트 질을 보장하는 기술은 경쟁에서 살아남는 결정적인 요소가 되었다.

Advanced Microcontroller Bus Architecture (AMBA)는 재사용 되는 IP 코아들의 연결을 강하게 하

* 학생회원, 한양대학교 컴퓨터공학과
(Department of Computer Science & Engineering, Hanyang University)
** 정회원, 한양대학교 전자컴퓨터공학부
(Department of Computer Science & Engineering, Hanyang University)
※ 본 연구보고서는 정보통신부 출연금으로 MIC/ IITA/ ETRI, SoC산업진흥센터에서 수행한 IT SoC 핵심설계인력양성사업의 연구결과입니다.
※ 본 연구는 한국과학재단 특정기초연구(R01-2006-000-11038-0(2008))지원으로 수행되었음.
접수일자: 2007년11월23일, 수정완료일: 2008년3월28일

는 버스 아키텍쳐 이며 그 중 AMBA Advanced High-performance Bus(AHB)는 주로 고성능 시스템 back-bone 버스로 사용 된다[2].

AMBA 기반 SoC에 임베디드되어 있는 코아를 테스트하기 위해서는 Test Interface Controller(TIC), External Bus Interface(EBI), 테스트 래퍼와 같은 장치들이 코아에 추가되어 사용된다[2~5]. TIC[2]는 기본적인 읽기/쓰기 트랜잭션을 발생시키는 AMBA 버스 마스터로써 AMBA 기반 시스템을 테스트 할 때 쓰이는 인터페이스 컨트롤러이다. EBI는 테스트 데이터를 전송하는 외부 테스트 버스(TBUS)로 사용되고 테스트 래퍼는 온-칩 버스에 직접 연결되어 있지 않은 코아의 입·출력으로의 접근을 가능하도록 한다.

TIC[2]는 임베디드 코아로의 주소와 테스트 데이터 혹은 테스트 응답 전송을 위한 TBUS와 AHB 사이의 경로를 하나만 제공하기 때문에 쓰기와 읽기 트랜잭션은 반드시 상호 배타적으로 실행되어야 한다. 그리고 TBUS 상의 버스 충돌을 피하기 위해서 추가적인 턴어라운드 싸이클이 필요하며 이는 기능적·구조적 테스트 시간을 증가 시키는 원인이 된다. [3]에서의 스캔 테스트 래퍼는 각각의 코아에 맞도록 설계 되지만 스캔 입력과 스캔 출력이 반드시 상호 배타적으로 수행되어야 하기 때문에 많은 수의 테스트 싸이클이 필요하다. [4]에서는 스캔 체인에서의 스캔 입·출력은 동시에 수행될 수 있으나 AHB-APB 브리지 로직의 수정은 AMBA 시스템과의 호환성을 손상시킨다. [6]에서는 AMBA 프로토콜과의 호환성은 완벽하게 유지하면서 스캔 쉬프트 입·출력을 동시에 지원하는 방법을 제안하였지만 기능적 테스트를 수행 할 때는 TIC의 특징을 그대로 사용하였기 때문에 기능적 테스트 시간은 증가한다.

Peripheral Component Interconnect(PCI) 버스는 칩과 보드 상의 어댑터를 연결하기 위해 광범위 하게 이용되어 왔다[7]. 내부에 AMBA 버스가 있는 SoC를 포함하는 PCI 보드에는 온/오프-칩 버스 브리지로 불리는 AHB-PCI 버스 브리지가 필수 구성요소 이다. 이 브리지는 온-칩 버스와 오프-칩 버스를 연결하기 위해 사용된다. AHB-PCI 버스 브리지가 있는 시스템 칩은 사운드, 그래픽, 네트워크 카드와 같은 곳에 다양하게 적용되어 왔다[8~10].

본 논문에서는 AHB-PCI 브리지, EBI, 테스트 래퍼를 이용하는 효율적인 테스트 접근 메커니즘을 제시한다. 테스트 입력 경로와 출력 경로를 독립적으로 제공하여 기존 방식에서 필요하던 버스 방향 전환에 따른 턴어라운드 지연을 배제시킴으로써 테스트 시간을 감소시켰다. 제안하는 방법은 외부 테스트 데이터를 ATE 장비로부터 SoC 까지 전송 시키는데 있어 온/오프-칩 버스 브리지를 최대한 재활용 하여 테스트 인터페이스 로직으로 생기는 실리콘 오버헤드를 최소화 한다.

본 논문은 다음과 같이 구성되어 있다. Ⅱ장에서는 AMBA 기반 SoC의 일반적인 테스트 접근 메커니즘을 소개하고 Ⅲ장에서는 제안하는 방식을 설명한다. Ⅳ장의 실험 결과에서는 제안 하는 방법의 면적 오버헤드 및 테스트 인가 시간의 우수성을 보여주며 Ⅴ장에서 결론을 제시한다.

Ⅱ. AMBA 기반 SoC의 테스트 인터페이스

일반적인 AMBA 기반 시스템은 그림 1에서처럼 Advanced High-performance Bus(AHB), Advanced Peripheral Bus(APB)로 구성된다. AHB는 마이크로 프로세서와 같은 고성능 코아와의 인터페이스를 위해 사용되고 상대적으로 느린 속도의 디바이스들은 APB에 연결된다. AHB-APB 브리지는 AHB와 APB 버스의 서로 다른 속도와 프로토콜을 조절하기 위해 반드시 사용 되어야 한다.

그림 1의 TIC IP 코아는 AMBA 버스 마스터로써 기본적인 읽기/쓰기 트랜잭션을 발생시키는 테스트 인터페이스 컨트롤러이다[2]. AMBA 기반 SoC의 외부에 있는 메모리 모듈에 접근하기 위해서는 EBI의 단방향 32 비트 주소 핀과 양방향 32 비트 데이터 핀이 일반적으로 사용된다[11~13]. EBI와 AMBA의 기능적 버스를 테스트 버스로 이용하기 때문에 추가적인 테스트 접근 메커니즘은 필요로 하지 않는다. 테스트 모드 동작 시에는 해당 코아의 고립, 조절 용이도, 관측 용이도를 높이기 위하여 그림 2와 같은 테스트 래퍼(혹은 테스트 장치)

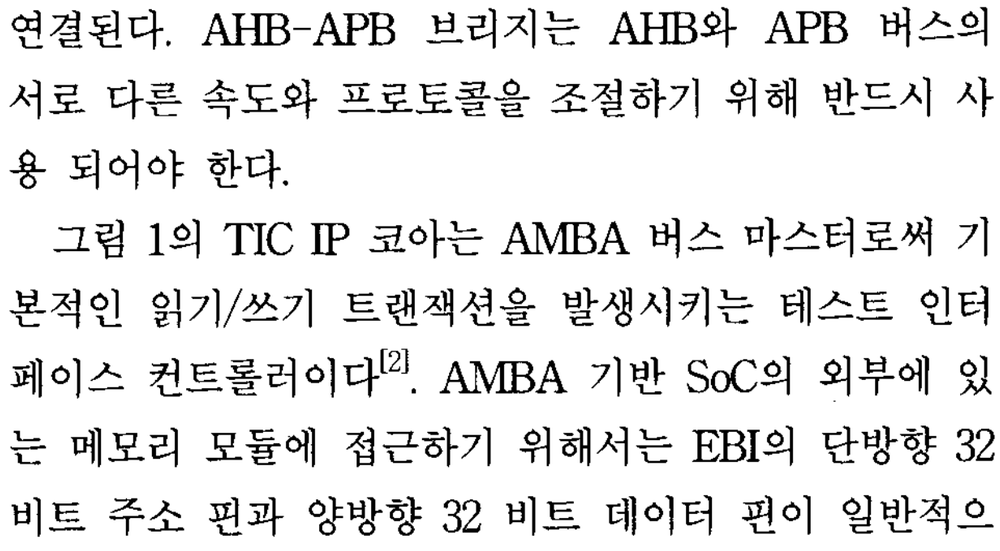


그림 1. TIC가 있는 AMBA 시스템
Fig. 1. Example of AMBA system with the TIC.

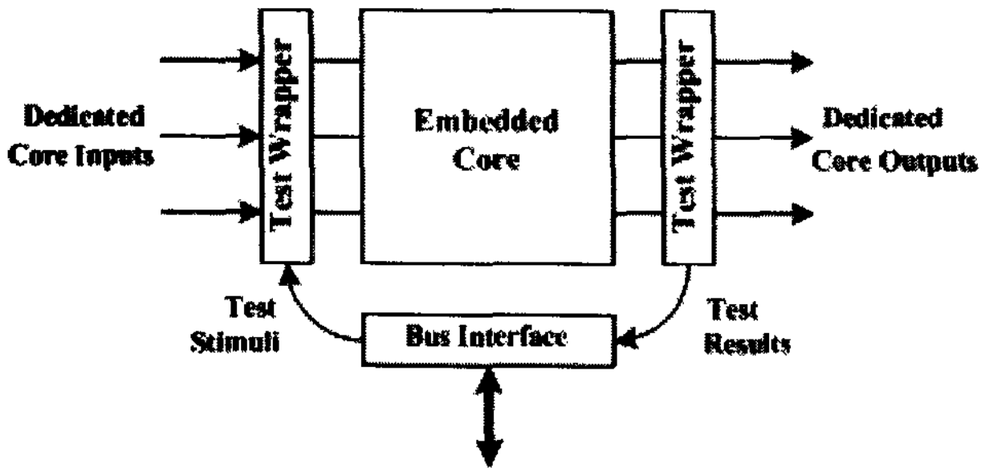


그림 2. 테스트 래퍼
Fig. 2. Test wrapper.

가 정의된다. 테스트 래퍼는 각 코아의 입·출력과 테스트 전략에 따라 구성되며 테스트 대상 코아가 AMBA 규격에 맞지 않아도 테스트 래퍼를 통해 접근 가능하도록 한다[2].

테스트 성능을 높이기 위해서는 기능적·구조적 테스트를 모두 실행 할 필요가 있다. 기능적 테스트 시 모든 TIC 기반 기술은 그림 1에서처럼 동일한 테스트 버스(TBUS) 상에서 읽기/쓰기 전송을 하기 때문에 버스 충돌을 피하려면 버스 방향 전환을 위한 추가적인 턴어라운드 지연이 필요하다. 구조적 테스트를 위한 스캔 테스트 장치가 [3]에서 제안되었다. 임시 레지스터들을 포함하는 스캔 테스트 장치는 버스 폭 보다 많은 수의 스캔 체인과 I/O를 가진 코아를 지원하도록 설계되었다. 하지만, 스캔 데이터는 스캔 시프트 입·출력을 동시에 할 수가 없어 긴 테스트 인가 시간을 초래한다. [4]에서는 EBI의 주소 버스를 스캔 출력 경로로 사용함으로써 스캔 시프트 입력과 출력의 동시 수행이 가능하다. 그러나 AHB-APB 브리지 로직의 수정은 테스트 모드 시에 AMBA와의 호환성을 깨트려 결과적으로 기능적 테스트가 불가능하게 된다. [6]은 스캔 시프트 입력과 출력이 동시 수행 가능할 뿐만 아니라 AMBA 프로토콜과의 호환성을 완벽하게 유지하여 일반적인 기능적 테스트도 가능한 설계 방법을 제시하였다. 하지만 읽기와 쓰기 트랜잭션 모두를 동일한 TBUS를 통해 실행하는 TIC를 그대로 사용하기 때문에 버스의 방향 전환을 위한 추가적인 턴어라운드 싸이클을 필요로 하게 되어 기능적 테스트 시간을 증가시킨다.

그림 1의 온/오프-칩 버스 브리지는 SoC를 오프-칩 버스와 연결하기 위해 꼭 필요한 컴포넌트이다. 본 논문에서는 AMBA 기반 SoC를 위한 새로운 설계 기술을 소개한다. 제안하는 기술은 온/오프-칩 버스 브리지를 테스트 인터페이스로 재활용함으로써 테스트 인가 시간을 상당히 감소시킨다.

Ⅲ. 제안하는 AMBA 기반 SoC를 위한 테스트 접근 메커니즘

제안하는 기술의 가장 큰 기여는 테스트 모드 시 온/오프-칩 버스 브리지를 테스트 인터페이스로써 재사용한다는 것이다. 브리지 상의 AHB 마스터 컴포넌트는 ATE와 테스트 대상 칩 사이의 인터페이스로 재사용되고, ATE는 가상의 버스 마스터로 동작 한다. 정상 동작 시 사용 하는 기능적 버스를 독립적인 스캔 경로로 지정하고 버스 방향 전환에 따른 턴어라운드 지연을 없애므로 기능적·구조적 테스트 시간 모두 확연하게 줄었다. 제안하는 방법은 재활용 하는 코아와 기능적 버스 구조의 내부 수정을 필요로 하지 않아 AMBA 프로토콜과의 호환성을 유지한다. 본 논문에서 소개하는 테스트 조절 용이도를 가지는 브리지를 Test Ready Bridge(TR-brdige)라 하겠다.

1. 제안하는 테스트 접근 메커니즘

그림 3은 SoC 레벨에서의 제안하는 테스트 접근 구조이다. TR-bridge는 외부에서 오는 테스트 입력을 내부 버스 프로토콜에 맞도록 변환시키고 기능적·구조적 테스트 모드 모두 제공한다. 기능적 테스트는 디자인 검증을 위하여 사용 되었던 테스트 데이터를 사용한다.

TR-bridge는 Test Request(TREQ), Test Acknowledgement (TACK), Command Byte Enbale (CBE) 신호를 테스트 벡터 인가를 조절하기 위한 핸드세이크 신호로 사용한다. CBE 신호는 테스트 모드 조절뿐 아니라 PCI의 기능적 인터페이스로 사용되는 신호이다[7]. 기능적 혹은 구조적 테스트 모드 동작 시 그림 3의 PCI 인터페이스 AD 버스와 EBI의 EBIDATA

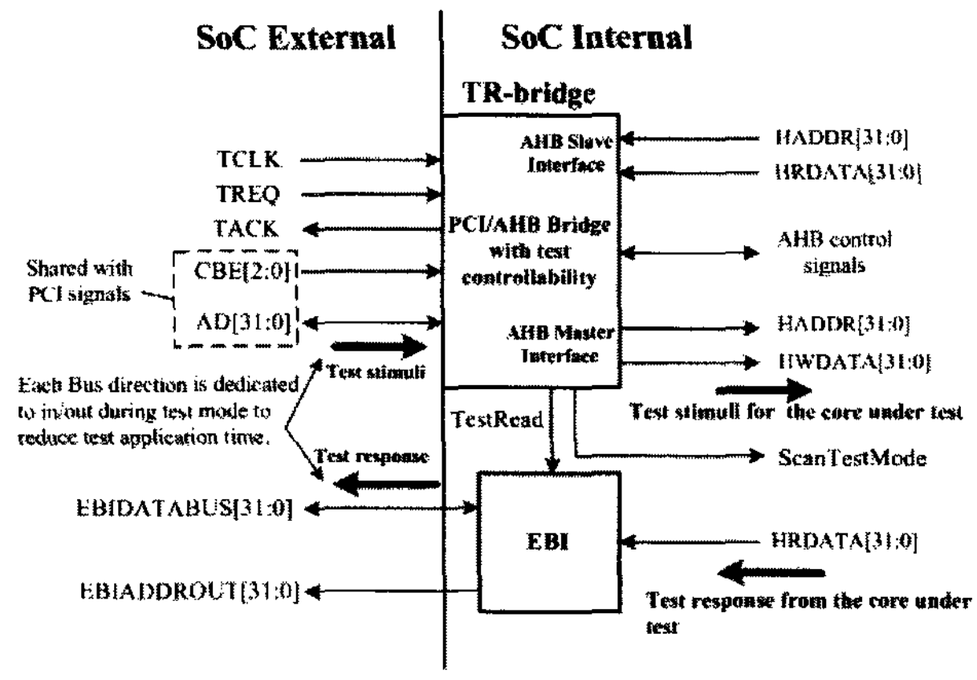


그림 3. 제안하는 테스트 접근 구조
Fig. 3. Proposed test access architecture.

버스는 각각 테스트 입력을 제공하고 테스트 출력을 관찰하는 경로로 사용된다. EBI는 정상 동작 시 주로 오프-칩 TCM(Tightly Couppled Memory)과의 통신을 위해 사용 된다. TR-bridge에서 나와 EBI로 들어가는 TestRead 신호는 EBIDATABUS 방향을 테스트 응답을 관찰할 수 있도록 바꿔준다. ScanTestMode 신호는 구조적 스캔 테스트를 위하여 테스트 래퍼를 동작시킨다.

2. 테스트 조절 용이도를 가지는 온/오프-칩 버스 브리지

이번 절에서는 그림 4와 같은 구조를 가지는 테스트 조절 용이도가 있는 AHB-PCI 버스 브리지를 제안한다.

브리지의 주요 구성요소는 AHB Master, PCI Target, AHB Slave, PCI Initiator, Test Controller 블록이다.

정상 동작 중에 AHB Master와 PCI Target 블록은 각각 AHB 버스 마스터와 PCI 버스 슬래이브로 동작한다. 이 컴포넌트들은 SoC가 PCI 버스에 대해서 슬래이브가 될 때에 활성화 된다. 또한 AHB Slave와 PCI Initiator 블록은 각각 AHB 버스 슬래이브와 PCI 버스 마스터 컴포넌트로 동작하는데 이 두 컴포넌트는 SoC가 PCI 버스에 대하여 버스 마스터로써 동작할 때 활성화 된다. 그림 4에서 어둡게 표현된 영역 HTIC (Hybrid Test Interface Controller)는 테스트 제어 로직으로써 정상 동작 시에는 테스트 요구 신호인 TREQ 신호가 인가되지 않음으로 활성화 되지 않는다.

그림 5는 테스트 모드 시 HTIC와 AHB Master 블록이 활성화되어 TR-bridge가 테스트 인터페이스 제어기로써 동작하는 것을 보여준다. 하나의 멀티플렉서와 간단한 Test Controller 블록으로 이루어진 HTIC는 ATE와 AHB Master 블록을 인터페이스 하는 주요 테스트 제어 블록이다. 멀티플렉서는 AHB Master 블록의 데이터 경로를 제어한다. 테스트 모드 시 PCI 인터페이스인 AD 버스는 AHB Master 블록으로 바로 연결되어 PCI Target과 PCI Write FIFO 블록과는 연결되지 않는다. 그리하여 테스트 모드 시에는 복잡한 PCI 프로토콜과의 호환성을 유지할 필요가 없다. 이러한 구조는 테스트 시퀀스를 간단히 하고 AHB에 비해 훨씬 느린 PCI의 속도 제한(33 또는 66 MHz)을 받지 않는다.

정상 동작 시 사용되는 AHB Master 블록은 테스트 모드 시에 재사용 되는 블록으로 AD 버스를 통해 외부로부터 들어온 테스트 벡터를 AHB 프로토콜과의 호환성을 완벽히 유지시키면서 내부 AHB 버스로 전송시킨다.

외부 테스트 제어 인터페이스는 테스트 클락(TCLK), 테스트 모드를 나타내는 두 제어 신호(TREQ, TACK), PCI 인터페이스와 공유하는 제어 신호(CBE[2:0])로 이루어진다.

표 1과 2는 HTIC의 외부 테스트 제어 인터페이스 신호의 동작을 보여준다. 신호들은 현재 동작 중인 모드에 따라 다른 기능을 가지게 된다. 독립된 핀인 TREQ와 TACK는 각각 Test Bus Request, Test Bus Acknowledge를 의미한다. TACK 신호는 칩의 외부에

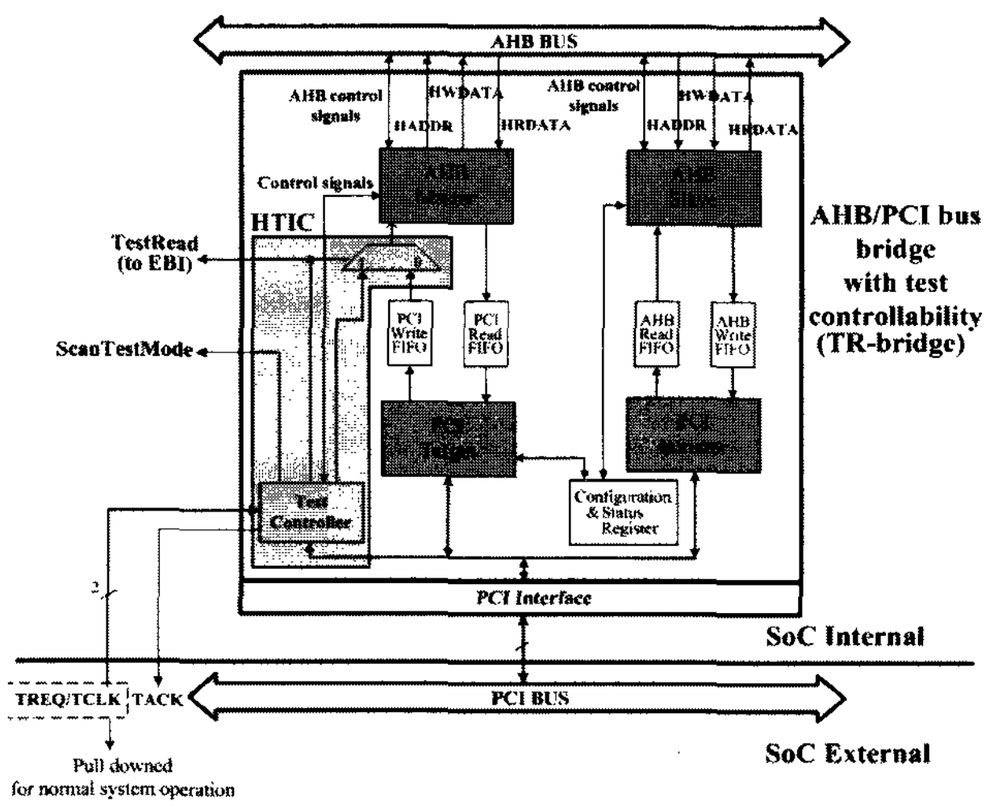


그림 4. 정상 시스템 동작 시의 테스트 조절 용이도를 가지는 AHB-PCI 버스 브리지

Fig. 4. AHB-PCI bus bridge with test controllability during normal system operation.

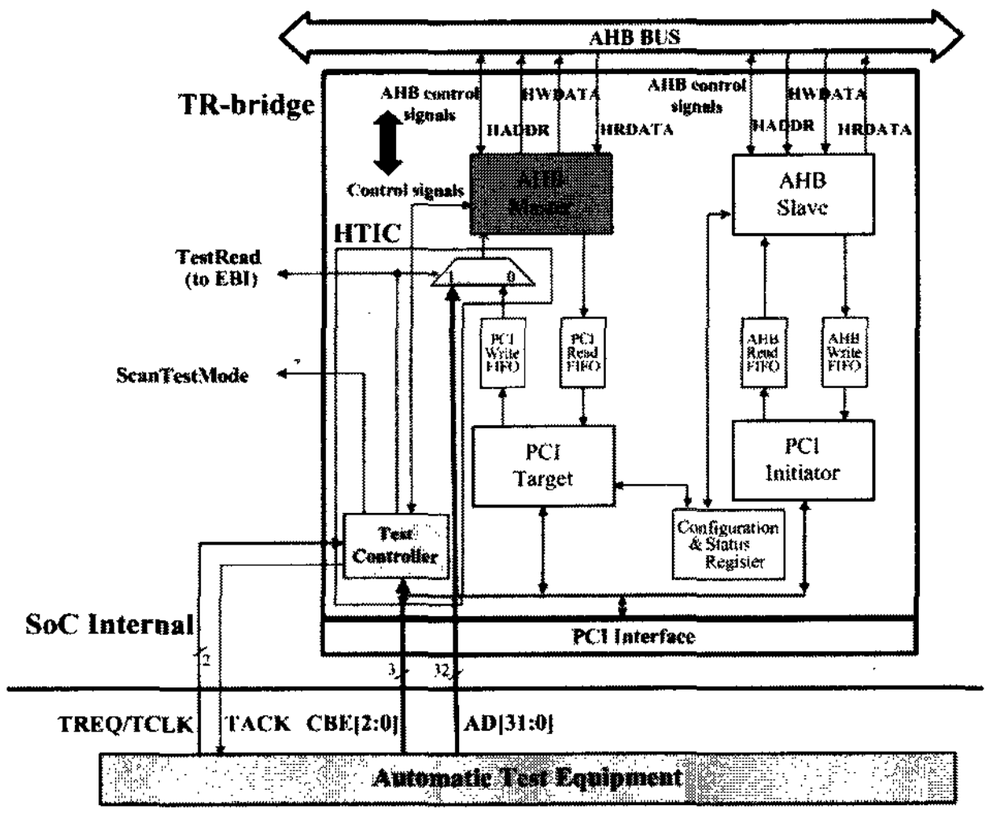


그림 5. 테스트 모드 시의 테스트 조절 용이도를 가지는 AHB-PCI 버스 브리지

Fig. 5. AHB-PCI bus bridge with test controllability during test mode.

표 1. 정상 동작 시 테스트 제어 신호
Table 1. Test control signals during normal system operation.

Input				Output	Description
TREQ	CBE[2]	CBE[1]	CBE[0]	TACK	
0	-	-	-	0	Normal operation
1	0	-	-	0	Request for entering a functional test mode
1	0	-	-	1	Functional test mode entered
1	1	-	-	0	Request for entering a structural test mode
1	1	-	-	1	Structural test mode entered

표 2. 기능적·구조적 테스트 동작 시 테스트 제어 신호
Table 2. Test control signals during either functional or structural test mode.

Input				Output	Description
TREQ	CBE[2]	CBE[1]	CBE[0]	TACK	
-	-	-	-	0	Current access is incomplete
1	-	1	1	1	Address vector
1	-	1	0	1	Write vector
1	-	0	1	1	Read vector
1	-	0	0	1	Control vector
0	-	-	-	1	Exit test mode

테스트 버스가 버스를 점유하였음을 알리거나 테스트 접근이 완료되었음을 알린다. TACK 신호가 low 일 때 현재의 테스트 벡터는 TACK 신호가 high가 될 때까지 값을 유지하여야만 한다. CBE[2:0] 신호는 TACK가 high 일 때만 HTIC에 의해서 샘플 된다.

테스트 인터페이스 결합으로 만들어진 서로 다른 네 가지 종류의 테스트 벡터에는 주소, 쓰기, 읽기, 제어 벡터가 있다. 테스트 모드 시에 CBE[1:0] 신호는 다음 싸이클에 인가되는 테스트 벡터의 종류를 나타내고 $CBE^{[2]}$ 신호는 현재 테스트의 종류가 구조적 혹은 기능적 테스트 인지의 여부를 나타낸다. 주소 벡터는 테스트 대상 코아를 선택하는데 사용 된다. 쓰기 벡터는 기능적·구조적 테스트 모두에서 테스트 입력을 위해 사용 되고 읽기 벡터는 테스트 응답 관찰을 위해 사용된다. 제어 벡터는 HSIZE, HPROT, HTRANS, HLOCK[2] 과 같은 AHB 제어 신호의 값을 갱신하기 위하여 사용된다.

SoC 외부로부터 혹은 외부로의 테스트 벡터 전송을 효율적으로 하기 위하여 PCI 인터페이스 AD[31:0]와 EBI의 EBIDATABUS[31:0]는 외부 32비트 테스트 버스로 재사용 된다. 정상 동작 중에는 주소와 데이터 모두를 전송하는 양방향인 AD[31:0] 버스는 테스트 모드 시는 테스트 입력 경로로만 사용되어 방향이 고정된다. 정상 동작 중에는 SoC 외부 디바이스와의 인터페이스를 위해 사용 되는 EBIDATABUS[31:0]는 테스트 모드 동작 시에는 테스트 응답을 관찰하는 출력 경로로써 ATE와 연결된다. 테스트 버스를 읽기와 쓰기 경로로 명확하게 나눔으로써 버스 방향을 바꾸기 위한 턴 어라운드 시간이 불필요하고 스캔 입·출력이 동시 수행 가능함으로써 테스트 인가 시간을 두드러지게 줄일 수 있다.

3. TR-bridge 동작

그림 6의 천이도는 TR-bridge의 동작을 보여준다. TACK 신호는 IDLE 상태에서 START 상태로 변하는 경우를 제외한 모든 경우의 천이를 제어한다. TREQ 신호가 인가되고 TACK 신호 응답이 high로 오면 HTIC는 테스트 모드로 들어간다. START 상태는 주소가 초기화되기 이전에 읽기와 쓰기 벡터가 입력되는 것을 막기 위한 상태이다. 인가되는 첫 번째 벡터는 주

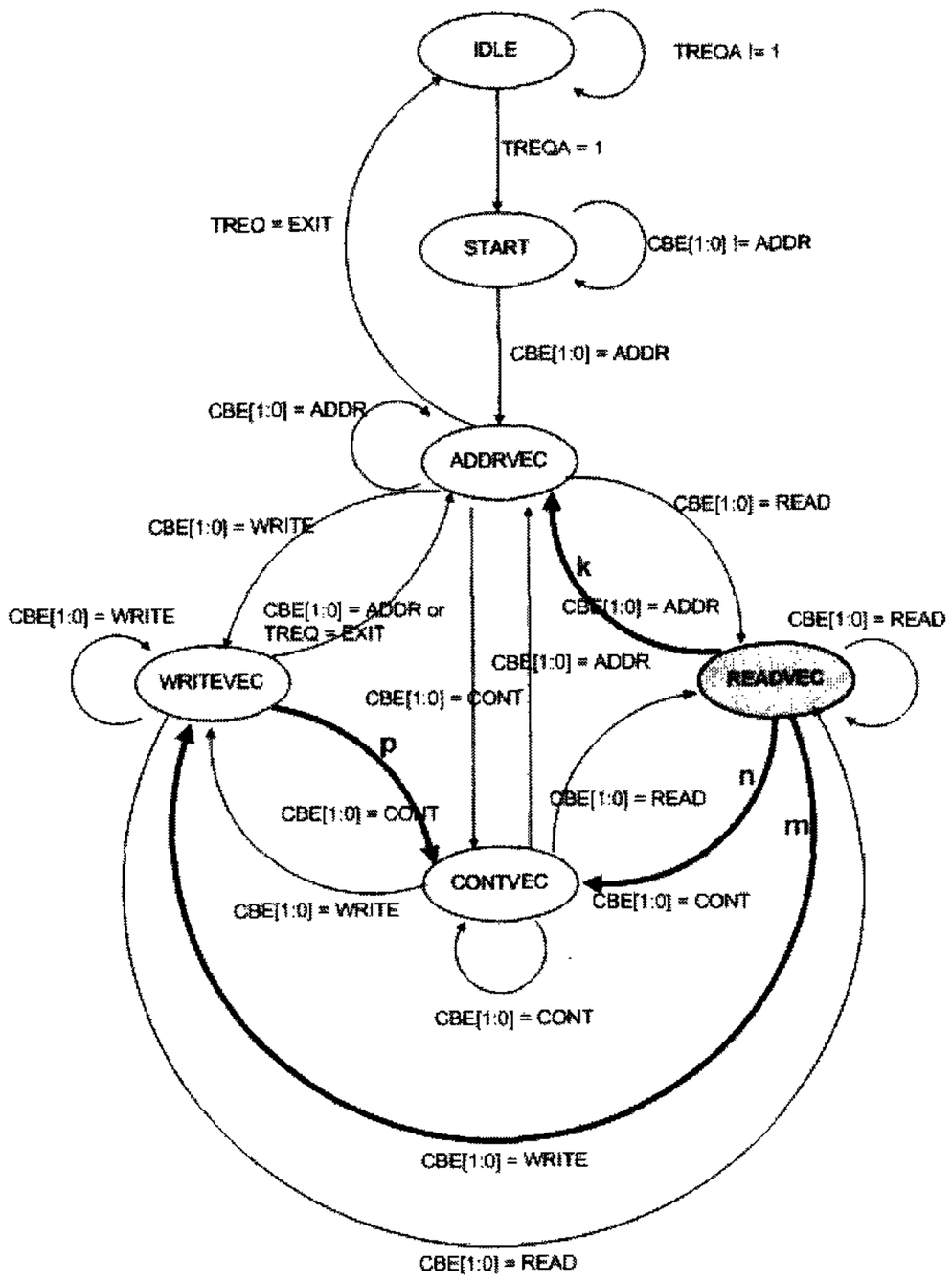


그림 6. 제안하는 HTIC의 상태도
Fig. 6. State diagram of the proposed HTIC.

소 벡터가 되도록 하기 위하여 START 상태는 CBE [1:0] 신호가 주소 벡터라고 가리킬 때만 빠져 나온다. 이 후 상태 머신은 CBE[1:0]의 신호 값에 따라 ADDRVEC, WRITEVEC, READVEC, CONTVEC 상태로 움직인다.

그림 7은 [2]에서 사용되는 TIC의 상태도를 보여준다. TIC 기반 접근에서 읽기 벡터나 버스트 읽기 벡터는 테스트 버스 방향을 바꾸기 위한 턴어라운드 시간을 위하여 그림 7에서 보이듯 항상 두 개의 추가적인 상태 LASTREAD, TRUNAROUND 상태가 이후에 따라온다. 왜냐하면 TIC 기반 구조는 어떤 종류의 전송이든 SoC 외부와의 인터페이스에 같은 테스트 버스를 사용하기 때문에 테스트 버스의 드라이버가 바뀔 때 발생할 수 있는 버스 충돌을 막기 위한 추가적인 싸이클이 필요하다[2]. 이것은 기능적 테스트 시간을 증가 시키는 요인이 되는데 실제 시스템에서는 읽기 트랜잭션이 다른 트랜잭션들 보다 더 자주 일어나기 때문이다. 그러나 제안하는 방법에서는 읽기와 쓰기 테스트 버스를 분리하였기 때문에 턴어라운드 상태와 같은 상태들이 필요하지 않다. 그림 6에서의 네 개의 상태(ADDRVEC, WRITEVEC, READVEC, CONTVEC)는 complete directed graph를 구성하며 이들 상태간의 트랜지션은 오직 한 클락 싸이클만 요구한다. 게다가 TIC 기반 접근의 제어 벡터는 언제나 제어 벡터 이전에 적어도 하나 이상의 주소 벡터를 필요로 한다[2]. 그러나 제안하는 방법에서의 제어 벡터는 그림 6에서처럼 CONTVEC 상태를 추가해 줌으로써 독립적으로 인가될 수 있어 테스트 인가 시간의 추가적인 감소를 가져 온다.

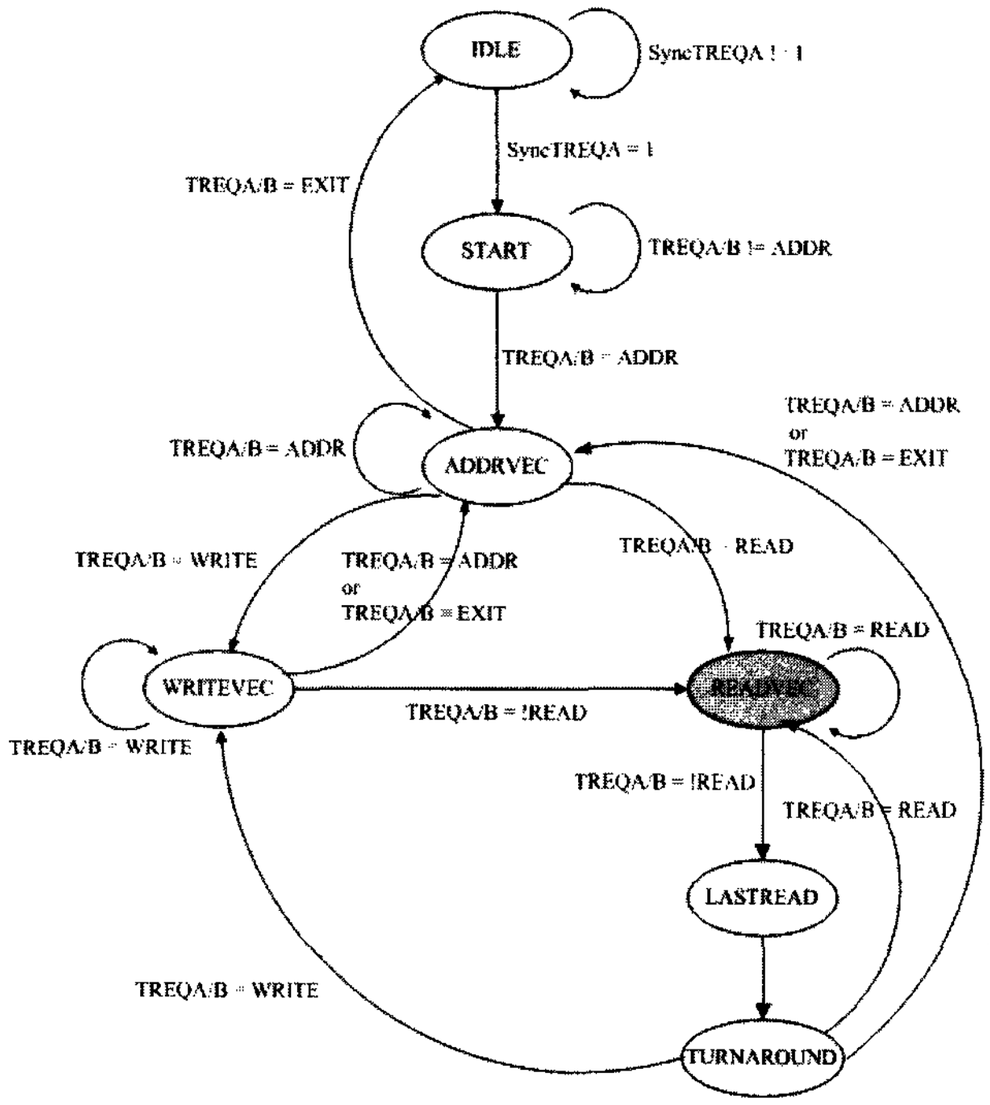


그림 7. TIC[2]의 상태도
Fig. 7. State diagram of the TIC.[2]

제안하는 방법과 TIC 기반 구조에서의 테스트 클락 싸이클 수를 비교하기 위하여 연속된 테스트 시퀀스에 대한 상태도에서의 트랜지션 수를 계산하였다. k, m, n, p 가 그림 6에서의 필요한 상태 변화의 수라면 그림 7에서는 얼마나 많은 수의 상태 변화가 필요할까? 세 가지 주요 케이스에 대하여 분석을 하였다.

case 1: 그림 7에서 세 클락이 필요한 READVEC에서 WRITEVEC로 혹은 READVEC에서 ADDRVEC 로의 전환은 그림 6의 제안된 방법에서는 한 클락만 필요하다. 그래서 TIC 기반 구조는 3(k+m) 클락 싸이클이 걸리는 것에 비해 제안하는 방법은 오직 (k+m) 클락 싸이클만 필요로 한다.

case 2: 그림 6에서 READVEC로부터 CONTVEC로의 전환은 한 클락이 필요하지만 그림 7에서는 하나 이상의 성공적인 주소 벡터 이후가 제어 벡터로 고려되기 때문에 같은 동작을 위해 적어도 네 클락을 필요로 한다[2]. 그래서 TIC 기반 구조는 4(n) 클락 싸이클이 필요하고 반면 제안하는 방법은 (n) 클락 싸이클만이 필요하다.

case 3: 그림 6에서 WRITEVEC로부터 CONTVEC로의 전환은 한 클락을 필요로 하지만 그림 7에서는 case 2에서 설명한 것과 같은 이유로 같은 동작을 위해 적어도 두 클락을 필요로 한다. 그러므로 TIC 기반 구조는 2(p) 클락 싸이클이 필요하지만 제안된 방법은 (p) 클락 싸이클만 필요하다.

case 1, 2, 3 로부터 얻어진 TIC 기반 구조에서와 제안된 HTIC 기반 구조에서의 전체 테스트 인가 시간은 각각 (1)과 (2)로 계산 될 수 있다.

$$T_{TIC} \cong 3(k+m)+4n+2p \quad (1)$$

$$T_{HTIC} \cong k+m+n+p \quad (2)$$

그리고 (3)에 있는 것처럼 TIC 기반 구조는 제안된 접근 보다 세배 이상의 클락 싸이클을 필요로 한다.

$$T_{TIC} \cong 3T_{HTIC}+n-p \quad (3)$$

읽기, 쓰기, 주소 트랜잭션은 다양하고 빈번하게 발생하기 때문에 기능적 테스트 시간이 구조적 스캔 테스트 시간 보다 (3)에 의한 영향을 더 받는다.

그림 8과 9는 각각 제안하는 방식과 TIC 기반 구조에서 예제 시퀀스를 실행 시켰을 때 볼 수 있는 타이밍도 이다. 사용된 시퀀스는 다음과 같다: Address ➔ Read 1 ➔ Read 2 ➔ Read 3 ➔ Address ➔ Write 1

그림 8에서 CBE 신호는 이어지는 싸이클에서 어떤 종류의 테스트 벡터가 오는지를 나타낸다. 초기에 CBE 신호는 주소 벡터 이후에 읽기 전송이 이어지도록 사용되었다. HTIC는 주소 벡터 Addr A를 T2 싸이클 이후에 샘플링 한 후 TR-bridge의 AHB Master 블록을 재사용함으로써 AHB 프로토콜에 맞는 전송을 발생시킨다. 이어지는 싸이클에서 읽기 벡터 Read 1이 온-칩 버스 AHB HRDATA로부터 EBIDATABUS로 전송되고 이어 외부 테스트 장비에 의해 값이 샘플 된다. 만약 내부 전송이 완료되지 못했다면 TACK 신호가 low가 되어 T5 이후 시간은 해당 전송을 완료하기 위한 추가적인 사이클로 사용될 것이다. 그림 8에서 버스트 읽기 전송 이후 새로운 주소 벡터 Addr B가 인가된다. 이어지는 싸이클에서 쓰기 데이터 Write 1이 AD 버스에 실리면 이후에 내부 AHB 버스로 전송된다.

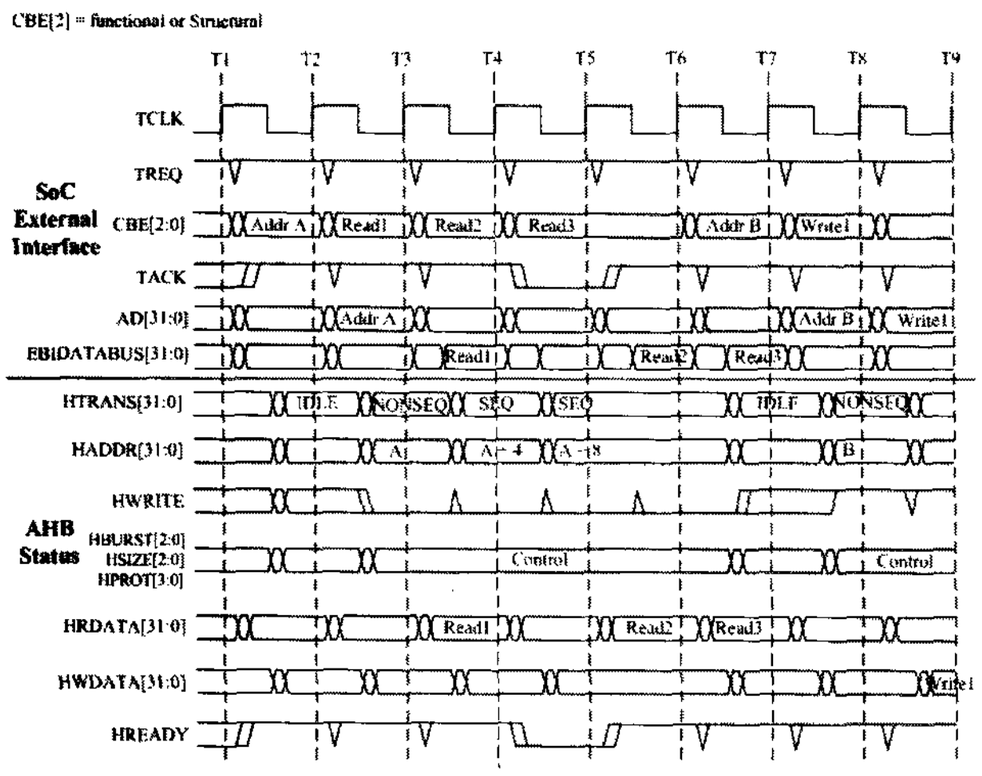


그림 8. 제안하는 TR-bridge의 타이밍도
Fig. 8. Timing diagram for the proposed TR-bridge scheme.

그림 9에서 첫 읽기 싸이클이 오기 전에 TBUS는 테스트 벡터를 전송하기 위해 반드시 high impedance 상태에 있어야 한다. 왜냐하면 읽기 벡터 싸이클의 버스 방향은 이전 주소 벡터 싸이클과는 반대이기 때문이다. 버스트 읽기의 마지막 실행에서 TIC는 버스 턴어라운드를 위해서 반드시 내부 버퍼를 꺼줘야 한다.

그림 8과 9는 TIC 기반 기술을 사용 하였을 때 읽기 전송을 위해서는 두 개의 추가적인 싸이클이 더 필요함을 보여준다[2]. 만약 모든 다른 종류의 트랜잭션들이 고려된다면 좀 더 확연한 테스트 싸이클 감소를 볼 수 있을 것이다.

그림 9. TIC 기반 구조[2] 에서의 타이밍도
Fig. 9. Timing diagram for the TIC-based scheme.[2]

4. 구조적 테스트 인터페이스

스캔 테스트와 같은 구조적 테스트 시 벡터 인가와 관찰을 위해 논문 [6]의 AMBA용 테스트 래퍼(harness) 기술을 적용하였다. 그리고 [6]의 AHB-APB 브리지 bypass multiplexer 기술도 적용하였는데 이것은 느린 APB 버스의 임베디드 코아를 구조적 테스트 할 때 속도를 높이기 위해서 사용된다. 그러나 우리의 구조에서는 독립적인 스캔 출력 경로를 위한 ad-hog 로직은 사용하지 않았다.

스캔 입·출력 경로는 각각 TR-bridge의 AD 버스와 EBI의 EBIDATABUS를 사용한다. 스캔 시프트 동안 어떤 읽기 트랜잭션도 요구하지 않으면서 스캔 입력과 출력의 동시 수행이 가능하다. 제안하는 방법은 스캔 테스트 동안 AMBA 버스 상에서 오직 쓰기 트랜잭션만을 발생시키기 때문에 읽기 트랜잭션을 위한 추가적인 싸이클이 불필요하여 스캔 테스트 시간을 더욱 줄일 수 있다.

5. TR-bridge 무결성 테스트

그림 10에서 어두운 부분으로 표현된 TR-bridge 내부를 테스트하기 위하여 스캔 체인과 테스트 래퍼가 AHB Master와 HTIC 블록을 제외한 TR-bridge에 삽입되어 임베디드 코아 테스트를 통하여 검증한다.

AHB 버스 상의 TR-bridge를 위한 자체 테스트 방법을 그림 11에서 볼 수 있다. TR-bridge를 위한 테스트 시퀀스는 다음과 같다. ATE로부터 나온 주소 벡터가 AD 버스를 통하여 HTIC와 AHB Master 쪽으로 전달되면 브리지 AHB slave 쪽의 해당 스캔체인과 Primary Inputs(PIs)이 테스트 래퍼에 의해 선택된다.

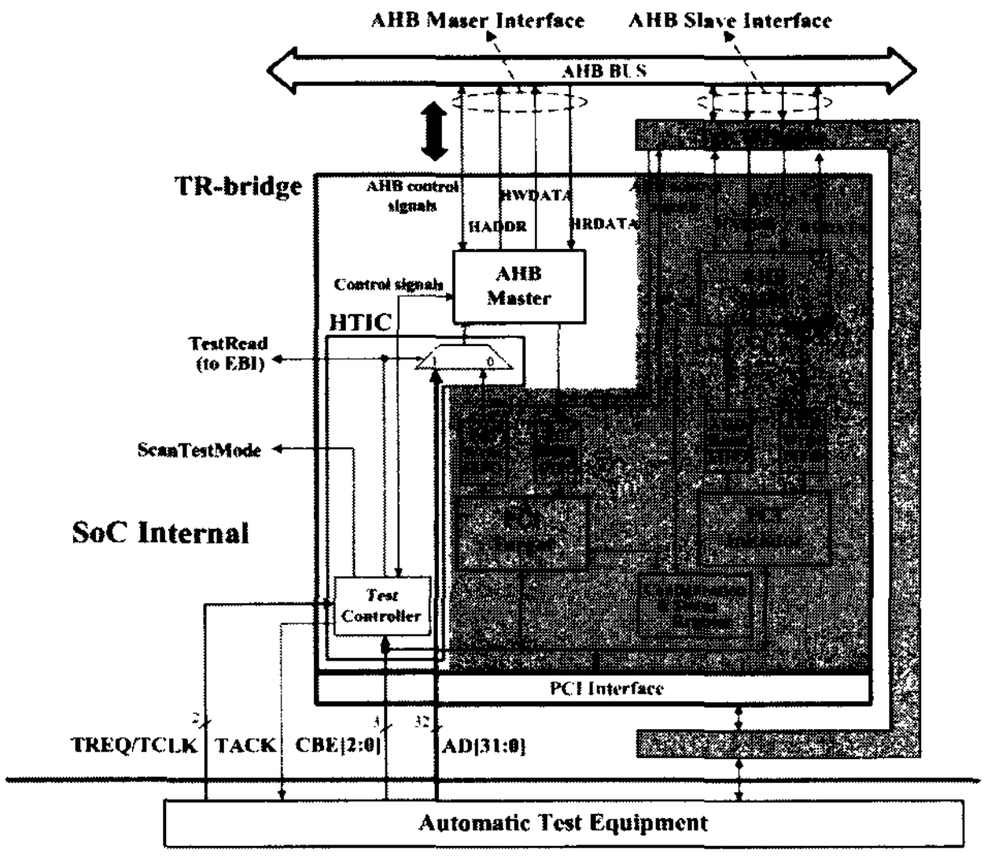


그림 10. TR-bridge 무결성 테스트를 위한 구조

Fig. 10. Design for testability for integrity test of the TR-bridge.

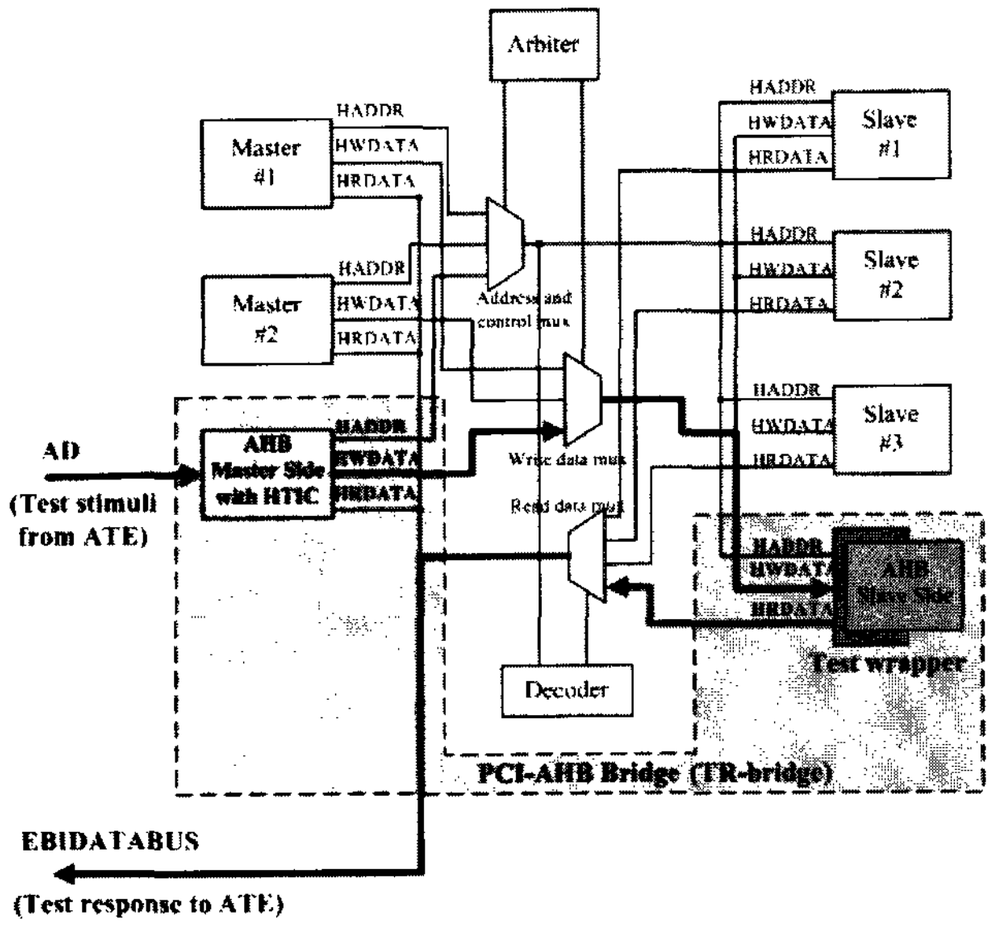
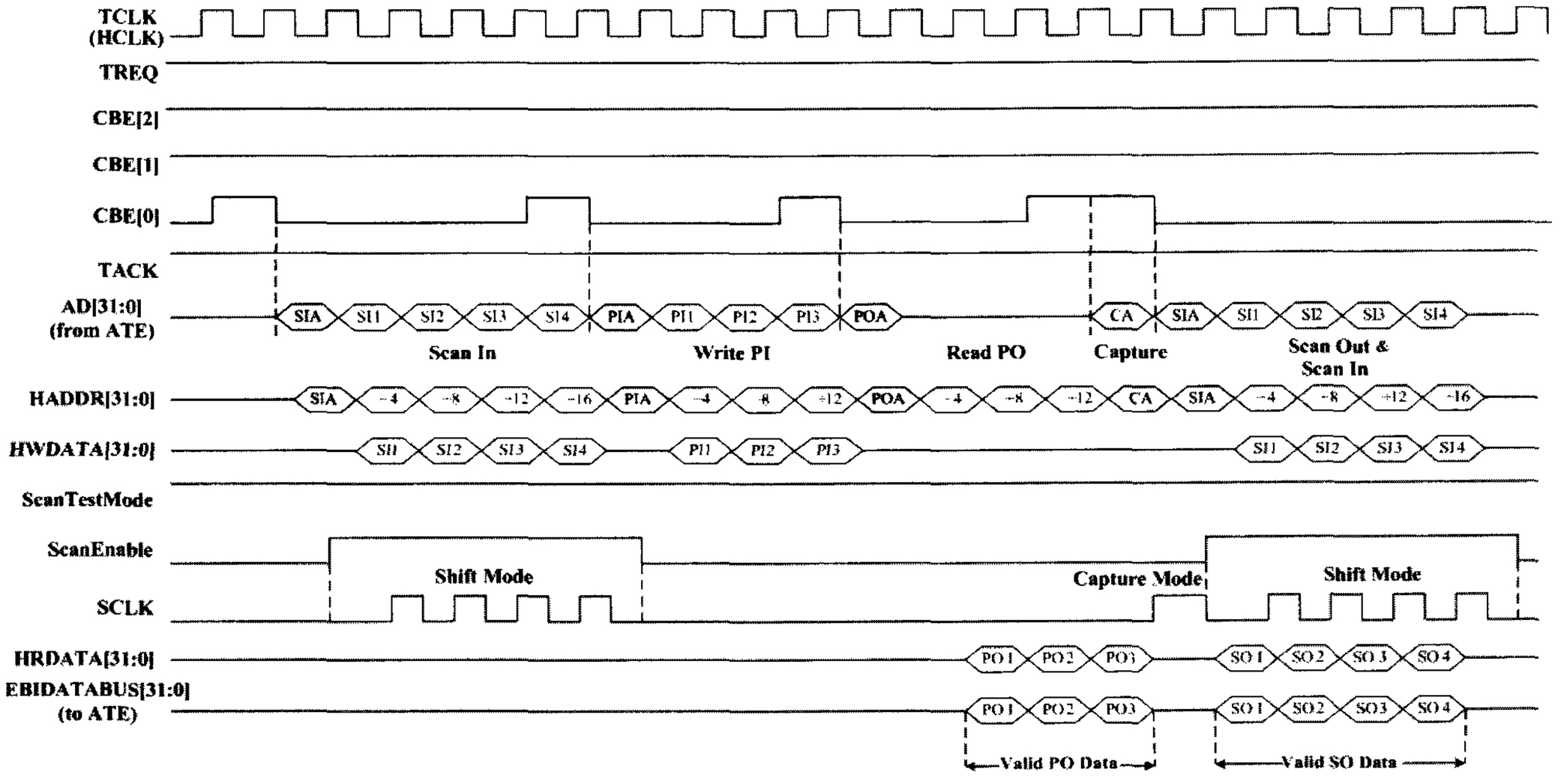


그림 11. TR-bridge 테스트 도식

Fig. 11. TR-bridge test scheme.



TCLK : Test clock

TREQ: Test request signals to the bridge

CBE[2:0] : Test control signals to the bridge

TACK : Test request acknowledgement from the bridge

AD[31:0] : Test bus to the bridge

HADDR : AMBA address bus

HWDATA : AMBA write data bus

ScanTestMode : Structural test enable-signal to test wrapper

ScanEnable : Scan shifting enable-signal from test wrapper to core under test

SCLK : Scan shifting clock

HRDATA : AMBA read data bus

EBIDATABUS : EBI data bus to ATE

그림 12. TR-bridge에 의한 스캔 테스트 타이밍도

Fig. 12. Timing diagram of scan test by the TR-bridge.

스캔 입력 혹은 기본 입력 데이터는 AD ➜ HWDATA 경로를 통하여 테스트 대상 회로에 쓰여 진다. 그 후 타겟 Primary Outs(POs)와 스캔 체인이 선택되어 캡쳐된 내부 응답 신호 및 POs의 결과가 HRDATA ➜ EBIDATABUS 경로를 통하여 관찰된다.

6. 구조적 테스트 동작

그림 12는 명확한 시뮬레이션 과정을 보여주기 위해

적용된 길이 4의 싱글 스캔 체인이 삽입된 간단한 테스트 대상 코아에 대한 구조적 스캔 테스트 시의 타이밍도이다.

구조적 테스트 모드는 CBE[2] 신호가 high 일 때 요청된다. 구조적 테스트 동작 중에는 TR-bridge에서 나오는 ScanTestMode 신호가 테스트 래퍼와 AHB-APB 우회 멀티플렉서를 동작시키기 위해 인가된다. 스캔 입·출력 경로는 각각 TR-bridge의 AD 버스와 EBI의 EBIDATABUS로 지정되어 스캔 시프팅을 하는 동안 AMBA 버스 상에 읽기 트랜잭션을 요청하지 않고 테스트 입력 인가와 응답 관찰 수행을 동시에 할 수 있다.

테스트 데이터와 결과를 전송하기 위해 관계되는 주요 신호는 AD, HWDATA, HRDATA, EBIDATABUS이다. TREQ, CBE, TACK 신호는 TR-bridge의 신호로써 포괄적인 테스트 동작을 제어한다. CBE[1:0] 신호는 이어지는 싸이클의 AD 또는 EBIDATABUS의 내용이 무엇인지 나타낸다. 동작 시퀀스는 다음과 같다. 먼저 AD 버스를 통해 전송되는 주소는 해당 스캔 체인과 PIs를 선택한다. 스캔 체인과 PIs에는 AD ➜ HWDATA 경로를 통하여 테스트 대상 코아로 온 값이 쓰여 진다. 다음 POs가 선택되어 그 값들이 HRDATA ➜ EBIDATABUS 경로를 통하여 관찰되면 ClockGenerator에 의해 생성되는 캡쳐 명령이 수행된다. 마지막으로 스캔 체인이 선택되어 캡쳐된 결과를 같은 경로를 통하여 시프트 시키면서 관찰하고 스캔 입·출력 경로를 분리하였기 때문에 동시에 스캔 입력 동작을 수행한다. 결국 제안하는 방법은 스캔 테스트 동안 AMBA 버스에 오직 쓰기 트랜잭션만을 발생시키기 때문에 읽기 트랜잭션을 위한 추가적인 싸이클을 일으키지 않는다.

7. 구조적 스캔 테스트 시간 추정

독립적인 스캔 입·출력 방법[3]과 현재 제안하는 방법의 구조적 스캔 테스트 시간 비교를 위하여 다음의 각 항에서는 수학적인 식으로 설명 하였다.

가. 독립적 스캔 입·출력 방법의 테스트 인가 시간

$$TC_{AHBCore} \cong 2SCL_M + PIR_N + POW_N + 7 \quad (4)$$

$TC_{AHBCore}$: Time to apply a test patter for each AHB core with [3]
SCL_M : Maximum scan chain length
PIR_N : The number of 32-bit register set of the wrapper at the core primary input side
POW_N : The number of 32-bit wire set of the wrapper at the core primary output side

$$TAT_{AHBCore} \cong TP_N(2SCL_M + PIR_N + POW_N + 7) \quad (5)$$

$TAT_{AHBCore}$: Total test application time of [3] for each AHB core
TP_N : The number of test patterns

$$TAP_{APBCore} \cong 2(TP_N(2SCL_M + PIR_N + POW_N + 7)) \quad (6)$$

$TAT_{APBCore}$: Total test application time of [3] for each APB core

AHB에 연결된 코아를 독립적인 스캔 입·출력 방법[3]으로 테스트 할 때 하나의 테스트 패턴에 대한 테스트 시간 계산은 (4)와 같다. 스캔 입력과 출력은 동시에 수행될 수 없으므로 두배의 스캔 쉬프트 시간이 필요하다((4),(5),(6)에서 $2SCL_M$으로 표시). 버스 폭보다 길이가 긴 코아 래퍼 레지스터의 값들은 한 번에 전송시킬 수 없어 이를 옮기기 위해 몇 번의 클락을 사용한다.

테스트 제어와 테스트 버스 방향 전환을 위해 추가적으로 7개의 싸이클이 더 필요하다. [3]의 기술을 적용한 각 AHB 코아의 전체 테스트 인가 시간은 식 (5)와 같다. APB는 느린 속도의 주변 장치를 위한 인터페이스이며 [2]에 따라 APB 주변장치는 각각의 쓰기 혹은 읽기 전송을 위해 두 개의 시스템 클락을 사용한다. 따라서 [3]의 기술을 적용한 각각의 APB 주변장치의 전체 테스트 인가 시간은 식 (6)처럼 계산될 수 있으며 이는 AHB 코아 보다 약 두 배 정도 더 걸린다.

나. 제안하는 기술의 테스트 인가 시간

$$TC_{prop} \cong SCL_M + PIR_N + POW_N + 5 \quad (7)$$

TC_{prop} : Time to apply a test pattern for each AHB core using the proposed technique

$$TAT_{prop} \cong TP_N(SCL_M + PIR_N + POW_N + 5) \quad (8)$$

$TATprop$: Total test application time of the proposed techniqe for both AHB and APB cores

제안된 방법을 적용한 AHB 코아의 스캔 테스트 벡터 인가와 응답 캡쳐에 필요한 시간은 (7)과 같이 계산할 수 있다. 스캔 시프팅의 동시 진행으로 인해 (4)에 비해 $2SCL_M$ 항은 SCL_M으로 줄어들었고 버스 방향 전환에 사용되었던 두 클락 싸이클을 없애므로써 7개의 클락 싸이클이 5개로 줄었다. [6]에서의 스캔 테스트 모드시에 AHB/APB 브리지를 우회시키는 기술을 적용하였기 때문에 AHB와 APB 코아의 테스트 인가 시간은 동일하다. 그러므로 AHB, APB 모두를 위한 전체 테스트 시간은 식 (8)과 같다.

표 3. 테스트 대상 코아들의 특성
Table 3. Characteristics of cores under test.

Cores		Total Area (No. of NAND Gates)		No. of PIs	No. of POs	No. of DFFs	Max. Scan Chain Length	No. of Vectors	Fault Coverage (%)
		Original	Full Scan						
AHB	Leon3 Processor	41901	46303	252	148	1166	37	386	99.75
	SDRAM Controller	3701	4115	93	119	212	7	68	99.45
	AHB-PCI Bridge	6364	7055	40	145	275	9	79	99.92
	Ethernet MAC	32737	35580	109	243	1339	42	485	99.99
APB	UART	9308	10523	69	32	524	17	231	98.99
	GPIO	4922	5107	78	104	96	3	13	100
	RTC	7566	9067	47	32	340	11	130	99.99

이러한 예측 식들은 TR-bridge의 정확한 테스트 제어 시퀀스를 포함하지 않는다. 그러나 각각의 AHB 코아를 위한 테스트 인가 시간은 반으로 줄어듦을 쉽게 관찰할 수 있고 APB 코아를 위한 시간은 원래 필요하던 시간보다 약 1/4정도 줄었음을 알 수 있다.

Ⅳ. 실 험

그림 13과 같은 AMBA 기반 SoC가 임베디드 코아의 면적 오버헤드와 테스트 인가 시간을 측정하기 위하여 사용되었다[14]. PLL을 제외한 각각의 코아에 AMBA 기반 SoC와 ATE의 스캔 입력과 출력 경로를 최대한으로 이용하기 위하여 풀-스캔 기술로 32개의 스캔 체인을 삽입하였다. 기능적 테스트를 위하여 디자인 검증을 위하여 사용하였던 테스트 데이터를 사용하였다. 구조적 테스트를 위해서는 상업용 자동 테스트 패턴 생성(ATPG) 툴을 사용하여 테스트 패턴을 생성하였다. RTL 코드는 0.25 ㎛ 공정 라이브러리를 사용하여 합성하였다. 각 테스트 대상 코아의 특성[14]과 관련되는 테스트 정보가 표 3에 나와 있다. 세 번째와 네 번째 열은 각 코아의 2개의 입력을 가지는 NAND 게이트에 대한 gate count 이다. 열 5, 6, 7은 각각 PIs, POs, D-flipflop의 수이며 최대 스캔 체인 길이, 테스트 벡터의 수, fault coverage가 각각 행 8, 9, 10에 있다.

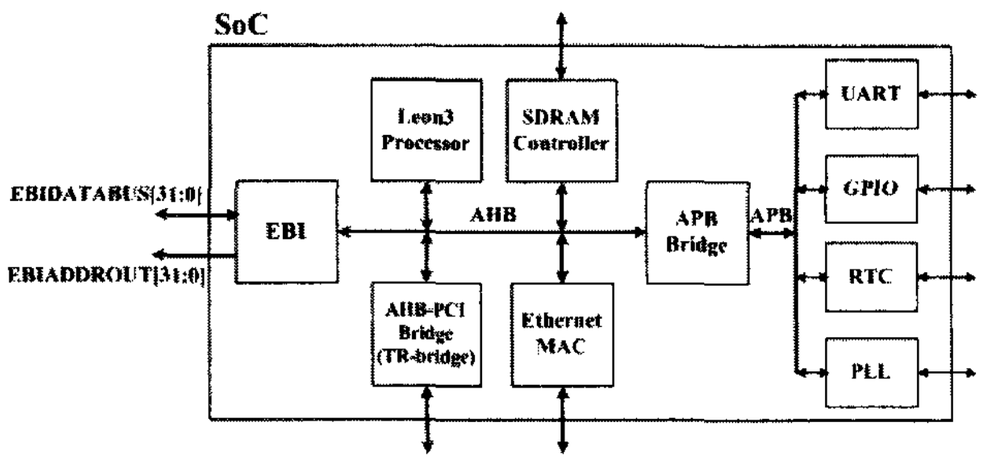


그림 13. 실험을 위한 AMBA 기반 시스템
Fig. 13. Example of AMBA-based system for experiments.

$$\frac{(OverheadofTheOtherTechique - Overheadofproposed)}{OverheadofTheOthterTechnique} \times 100(\%) \quad (9)$$

표 4는 TIC[2]와 비교한 제안된 HTIC의 면적 오버헤드이며 감소 비율은 식 (9)와 같이 계산하였다.

표 4에서 처럼 HTIC는 TIC[2]와 비교하여 면적이 약 76.23% 줄었다. 이것은 브리지의 기능을 테스트 제어 블록으로 공유함으로써 이루어 진 것이다.

표 5는 첫 번째 열에 있는 몇몇 주요 트랜잭션에 대한 기능적 테스트 시간을 비교한 것이다. 기능 검증 패턴이 인가될 때 TIC기반 구조인 [3], [6]과 비교하여 평균 35.72%의 테스트 싸이클이 감소됨을 관찰 할 수 있다.

구조적 스캔 테스트 시간 비교는 표 6에 있다. 표의 마지막 열에서 볼 수 있듯이 AHB와 APB 코아의 스캔 테스트 인가 시간은 [3]보다 확연이 줄어들었는데 AHB, APB 코아 각각 평균적으로 43.27%, 69.37%, 전체적으로는 50.26% 줄었다. [6]과 제안하는 TR-bridge

표 4. 면적 오버헤드 비교
Table 4. Comparison of area overhead.

Cores		Area Overheads (No. of NAND Gates)	Area Red. (%)
Test interface controller	TIC [2]	2983	76.23
	Proposed (HTIC)	709	

표 5. 기능적 테스트 시간 비교
Table 5. Comparison of functional test time

Key Transactions	Test Time (No. of Clocks)		Red. (%)
	TIC [2] - based [3], [6]	Proposed	
READVEC-> WRITEVEC	27720	9240	66.67
READVEC-> ADDRVEC	23643	7881	66.67
READVEC-> CONTVEC	860	215	75
WRITEVEC-> CONTVEC	278	139	50
Others	45578	45567	0
Total	98079	63042	35.72

표 6. 구조적 테스트 시간 비교
Table 6. Comparison of structural test time.

Cores		Test Time (No. of Clocks)		Red. over [3] (%)
		Exclusive Scan-in/out [3]	Concurrent scan-in/out Prop.	
AHB	Leon3 Processor	36284	20881	42.45
	SDRAM Controller	1904	1231	35.29
	AHB-PCI Bridge	2528	1220	51.74
	Ethernet MAC	49955	28172	43.60
APB	UART	20790	5792	72.14
	GPIO	520	185	64.23
	RTC	8320	2353	71.73
Total		120301	59834	50.26

는 모두 스캔 입·출력이 동시에 되는 기술을 사용하기 때문에 비슷한 인가 시간을 보여 [6]의 결과는 표에 추가하지 않았다.

AHB-APB 브리지를 우회하는 기술을 적용하고 스캔 입·출력을 동시에 가능하도록 하여 구조적 테스트 뿐만 아니라 기능적 테스트 일 때도 제안하는 기술이 최소의 면적 오버헤드를 가지고 테스트 인가 시간 감소에 효과적임을 보여준다.

Ⅳ. 결 론

본 논문은 온/오프-칩 버스 브리지를 활용한 보다 효율적인 테스트 접근 메커니즘을 제안하였다. 기능적 온/오프-칩 버스 브리지에 간단한 로직만을 추가하여 테스트 인터페이스 제어기로써의 기능을 가지게 하였다. 버스 방향 전환을 위한 턴어라운드 지연을 없애고 EBI를 독립적인 테스트 응답 출력 채널로 사용함으로써 기능적·구조적 테스트 시간 모두 확연히 줄였다. 본 논문에서는 AHB-PCI 브리지에 SoC 테스트 제어를 위한 온/오프-칩 재활용 기술을 적용하였으며 최소한의 면적 오버헤드로 테스트 비용을 줄이기 위해, 다른 종류의 온/오프-칩 버스 브리지에도 본 기술은 광범위하게 적용 가능하다.

참 고 문 헌

[1] Y. Zorian, E. J. Marinissen, and S. Dey, "Testing embedded-core based system chips," in Proc. IEEE Int. Test Conf., pp. 130-143, Oct. 1998.

[2] ARM, "AMBA specification (rev. 2.0)," May 1999.

[3] C. Feige, J. T. Pierick, C. Wouters, R. Tangelder, and H. G. Kerkhoff, "Integration of the scan-test method into an architecture specific core-test approach," Springer J. ElectroicTesting: Theory and Applicat., vol. 14, pp. 125-131, Feb. 1999.

[4] C. Lin and H. Liang, "Bus-oriented DFT design for embedded cores," inProc. IEEE Asia-Pacific Conf., pp. 561-563, Dec. 2004.

[5] ARM, "AHB example AMBA system technical reference manual," www.arm.com, ARM DDI 0170A, Aug. 1999.

[6] J. Song, P. Min, H. Yi and S. Park, "Design of Test Access Mechanism for AMBA Based System-on-a-Chip," in Proc. IEEE VLSI Test Symp., pp. 375-380, May 2007.

[7] PCI Special Interest Group, "PCI local bus specification," rev. 2.2, Dec. 1998.

[8] Z. Wang, Y. Ye, J. Wang, and M. Yu, "Designing AHB/PCI bridge," inProc. IEEE Int. Conf. on ASIC, pp.578-580, Oct. 2001.

[9] PLDA Ltd., AMBA-AHB PCI bridge IP core introductory document, http://www.plda.com/download/press_release/launch_ahb_pci_jan_2002.doc.

[10] HiTech Global, AHB-PCI bridge IP Core introductory document.

[11] http://www.hitechglobal.com/ipcores/ahbpci.htm.

[12] ARM, "ARM PrimeCell External Bus Interface (PL220)," www.arm.com, ARM DDI 0249B, Dec. 2002.

[13] ALTERA, "Excalibur devices hardware reference manual," www.altera.com, Vol. 1, ver. 3.1, Nov. 2002.

[14] Atmel, "AT91 ARM thumb microcontrollers," www.atmel.com, AT91R40807, Jan. 2002.
[15] J. Gaisler and E. Catovic, "Gaisler research IP core's manual," ver. 1.0.1, Jun. 2005.

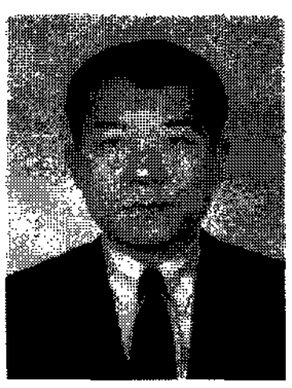
저 자 소 개

송 재 훈(학생회원)
2000년 한양대학교 전자컴퓨터 공학과 학사 졸업.
2002년 한양대학교 컴퓨터공학과 석사 졸업.
2003년 서울대학교 SoC 설계 센터 연구원.
2004년~현재 한양대학교 컴퓨터공학과 박사 재학 중.
<주관심분야 : SoC 설계, 테스트를 고려한 설계, 저전력 설계, 신호 무결성>

한 주 희(학생회원)
2005년 한양대학교 컴퓨터공학과 학사 졸업.
2008년 한양대학교 컴퓨터공학과 석사 졸업.
<주관심분야 : 통신, 컴퓨터, 신호처리, 반도체>

김 병 진(학생회원)
2007년 한양대학교 컴퓨터공학과 학사 졸업.
2007년~현재 한양대학교 컴퓨터 공학과 석사 재학 중.
<주관심분야 : SoC 테스팅, ASIC 설계>

정 혜 란(학생회원)
2005년 한양대학교 컴퓨터공학과 학사 졸업.
2007년~현재 한양대학교 컴퓨터 공학과 석사 재학 중.
<주관심분야 : SoC 테스팅, ASIC 설계>



박 성 주(정회원)
1983년 한양대학교 전자공학과 학사 졸업.
1983년~1986년 금성사 소프트웨어개발 연구원.
1992년 Univ. of Massachusetts 전기 및 컴퓨터공학과 박사 졸업.
1992년~1994년 IBM Microelectronics 연구스텝.
1994년~현재 한양대학교 전자컴퓨터공학부 정교수
<주관심분야 : 테스트 합성, Built-In Self Test, Scan Design, ATPG, ASIC 설계, 고속 신호처리>