논문 2008-45SD-2-10

# IEEE 1500 래퍼를 이용한 효과적인 AMBA 기반 시스템-온-칩 코아 테스트

## ( Efficient AMBA Based System-on-a-chip Core Test With IEEE 1500 Wrapper )

이 현 빈*, 한 주 희*, 김 병 진*, 박 성 주**

( Hyunbean Yi, Juhee Han, Byeongjin Kim, and Sungju Park )

요 약

본 논문에서는 Advanced Microcontroller Bus Architecture (AMBA) 기반 System-on-Chip (SoC) 테스트를 위한 임베디드 코어 테스트 래퍼를 제시한다. IEEE 1500 과의 호환성을 유지하면서 ARM의 Test Interface Controller (TIC)로도 테스트가 가능한 테스트 래퍼를 설계한다. IEEE 1500 래퍼의 입출력 경계 레지스터를 테스트 패턴 입력과 테스트 결과 출력을 저장하는 임시 레지스터로 활용하고 변형된 테스트 절차를 적용함으로써 Scan In과 Scan Out 뿐만 아니라 PI 인가와 PO 관측도 병행하도록 하여 테스트 시간을 단축시킨다.

Abstract

This paper introduces an embedded core test wrapper for AMBA based System-on-Chip (SoC) test. The proposed test wrapper is compatible with IEEE 1500 and can be controlled by ARM Test Interface Controller (TIC). We use IEEE 1500 wrapper boundary registers as temporal registers to load test results as well as test patterns and apply a modified scan test procedure. Test time is reduced by simultaneously performing primary input insertion and primary output observation as well as scan-in and scan-out.

**Keywords**: AMBA, IEEE 1500, System-on-Chip Test, Scan Test

## Ⅰ. Introduction

System-on-a-Chip (SoC) 설계 기술의 발달로, 재사용 가능한 Intellectual Property (IP)를 이용하여 설계 시간은 획기적으로 단축되었지만, SoC의 복잡도가 높아짐에 따라 테스트와 검증의 비중은 점점 더 증가하고 있다. SoC 테스트 비용은 Automatic Test Equipment (ATE)의 테스트 패턴 메모리의 용량 및 주입시간, 내장 코어 (embedded core)의 테스트 래퍼 (test wrapper), SoC Test Access Mechanism (TAM)과 테스트 방법에 의해 결정된다[1]. 내장 코어 테스트 표준으로써 IEEE 1500이 있다[2]. 칩의 내부 테스트라는 특성에 따라, 코아를 둘러싼 테스트 래퍼를 정의하고 있으며, 테스트 접근 메커니즘은 사용자가 정의 하도록 하여 보다 많은 유연성을 가지고 있다.

본 논문에서는 요즘 널리 사용되고 있는 Advanced Microcontroller Bus Architecture (AMBA) 기반의 내장 코어 테스트에 대해서 논한다. AMBA는 ARM사에서 개발한 온-칩-버스로써, AMBA를 통한 IP 코아의 기능적 테스트 (Functional Test)를 위해서는 Test

* 학생회원, 한양대학교 컴퓨터공학과
(Department of Computer Science & Engineering, Hanyang University)
** 정회원, 한양대학교 전자컴퓨터공학부
(Department of Computer Science & Engineering, Hanyang University)
※ 본 논문은 산업자원부가 지원하는 국가 반도체연구개발사업인 "시스템집적반도체기반기술개발사업(시스템IC2010)"을 통해 개발된 결과임을 밝힙니다.
접수일자: 2007년7월13일, 수정완료일: 2008년1월16일

Interface Controller (TIC), External Bus Interface (EBI), Test Harness가 사용된다[3]. 기능적 테스트는 테스트 시간이 매우 많이 소모되므로, 보다 적은 수의 패턴으로 높은 고장 검출이 가능한, Built-In-Self-Test (BIST)나 Scan Test와 같은 구조적 테스트 (structural test)가 요구된다[4].

본 논문에서는, AMBA 기반 내장 코어의 스캔 테스트를 위한 테스트 래퍼를 설계한다. IEEE 1500 래퍼를 활용하고 기존의 방식을 개선하여 추가되는 오버헤드를 줄이고, 테스트 시간을 줄일 수 있는 방법을 제시한다. II 장에서는 AMBA Test Interface와 기존 연구를 소개하고, III 장에서 IEEE 1500을 간략하게 소개한다. IV 장에서 본 논문에서 제안하는 AMBA 기반 코어 테스트 래퍼를 자세히 설명하고, V 장에서 테스트 시간 분석을 통한 비교 결과를 제시하며, VI 장에서 결론을 맺는다.

## II. AMBA Test Interface and Related Works

AMBA System은 그림 1과 같이 Advanced High-performance Bus (AHB)와 Advanced Peripheral Bus (APB)로 구성된다. AHB는 고속의 데이터 송수신을 위해 설계 되어 마이크로프로세서와 같은 고성능 모듈간의 연결에 사용되며, APB는 전송속도가 느린 장치들의 인터페이스에 사용된다. AHB와 APB의 연결을 위해서는 AHB-to-APB 브릿지가 필요하다[3].

TIC는 AMBA 시스템의 기능 테스트를 위한 인터페이스 제어기이다. 테스트 시 AHB 마스터가 되어 기본적인 AMBA Read/Write 트랜잭션을 수행함으로써 기능 테스트를 한다. 별도의 TAM 없이 AMBA 버스를 TAM으로 사용함으로써 소모 되는 면적을 줄일 수 있다. 테스트 데이터 Read/Write는 EBI의 32 비트 TBUS를 통해 이루어진다. 추가적으로, AMBA 규격에서, 테스트 모드 시 각 테스트 대상 코어의 Isolation, Controllability와 Observability를 위하여 그림 2와 같은 Test Harness를 정의하고 있다. Test Harness는 각 코어의 입출력과 테스트 전략에 맞게 구성될 수 있으며, 이를 통하여 non-AMBA I/O에 대한 접근도 가능하다[3].

논문 [5]에서는, 임시 레지스터를 사용하여 버스 폭 이상의 스캔 체인과 주입력 (Primary Input)을 지원할 수 있는 Scan Test Harness를 제시하였다. 하지만, 외부에서 테스트 입출력 경로를 공유하는 TIC의 한계 때문에 스캔 입력과 스캔 출력을 동시에 수행하지 못하므로 테스트 시간이 오래 걸린다. 논문 [6]에서는, EBI로부터 칩 외부로 나가는 어드레스 버스 중 일부를 테스트 응답 관측 경로로 사용함으로써 스캔 입력과 스캔 출력을 동시에 처리할 수 있도록 했다. 그러나 패턴 인가와 응답 관측을 위해 AMBA 이외의 버스를 추가하였으며, PI의 폭이 32 비트 이하, PO의 폭이 26 비트 이하인 APB 코어 테스트에만 적용 가능하다는 문제점이 있다. 논문 [7]에서는 스캔 체인의 수를 32개로 한정시키고, TBUS를 테스트 입력으로, EBI의 어드레스 버스를 테스트 출력으로 사용하여 스캔 입력과 스캔 출력을 동시에 수행할 수 있도록 하였으며, 테스트 모드 시에 데이터 패스가 AHB-to-APB 브릿지를 거치지 않고 APB 코어에 직접 연결시킴으로써 테스트 시간을 축소시켰다.

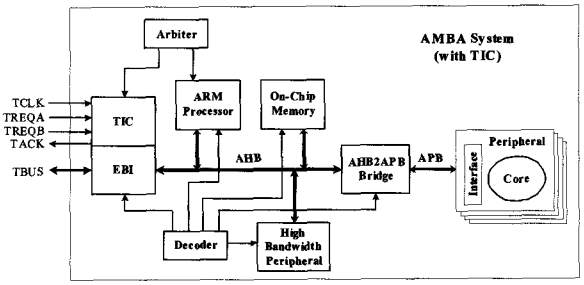



그림 1. AMBA 시스템
Fig. 1. AMBA System.

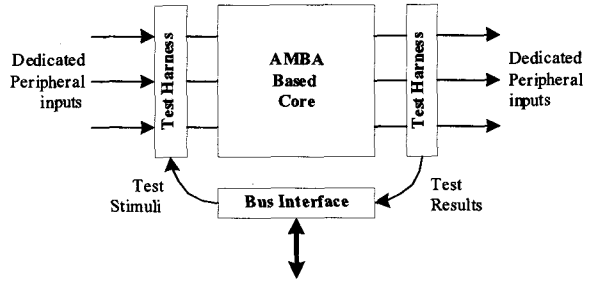



그림 2. 테스트 하네스
Fig. 2. Test Harness.

## III. IEEE Standard 1500

IEEE Standard 1500은 내장 코어 테스트 표준으로써, 크게 래퍼 명령 레지스터 (Wrapper Instruction Register (WIR)), 래퍼 바이패스 레지스터 (Wrapper Bypass Register (WBY)), 래퍼 경계 레지스터 (Wrapper Boundary Register (WBR))로 구성되어 있다. 테스트 접근 단자로는 (Test access terminal)로는

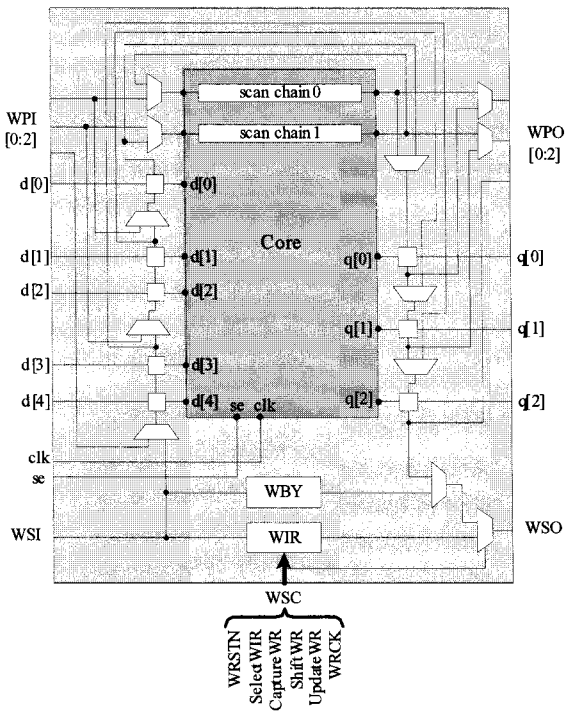
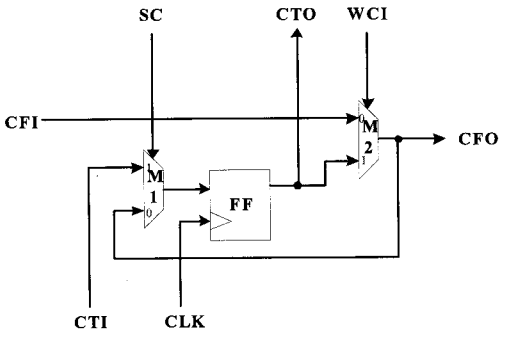


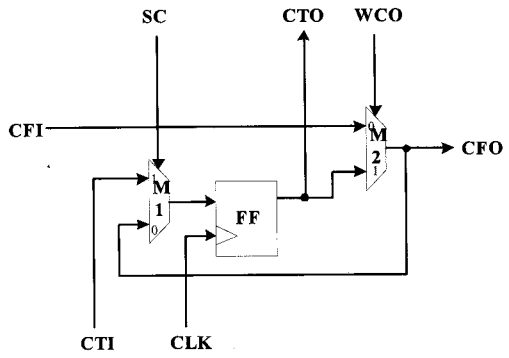

그림 3. IEEE 1500 이 적용된 코아
Fig. 3. IEEE 1500 Wrapped Core.

필수적인 래퍼 직렬 포트 (Wrapper Serial Port (WSP))와 선택사항인 래퍼 병렬 포트 (Wrapper Parallel Port (WPP))를 정의하고 있다[2]. IEEE 1149.1의 TAP 제어기와 같은 테스트 제어부 및 TAM은 SoC 설계자에 의해 설계 되어야 한다. 필수 테스트 명령어로는 WS_BYPASS, WS_EXTEST, Wx_INTEST가 있으며 그밖에 여러 선택적인 명령어를 정의하고 있다. 그림 3이 내부 스캔체인을 가지고 있는 IEEE 1500 랩드 코어 (wrapped core)의 한 예이다[2]. WSI와 WSO를 이용한 직렬 테스트 도메인뿐만 아니라 WPI와 WPO를 통한 병렬 테스트 도메인도 포함하고 있다. 여러 MUX를 이용하여 테스트 모드에 따라서 테스트 패턴의 경로를 다양하게 설정 할 수 있도록 구성 되어있다. 코어내의 스캔 체인 또한 WPI-WPO를 통한 독립적인 경로로 설정되거나, WBR과 같은 경로로 설정이 가능하다. 이러한 테스트 래퍼는 WSC 신호에 의해, 스캔 체인은 se와 clk에 의해 제어된다.

IEEE 1500 표준에 코어 테스트 목적에 따라서 다양한 구조의 래퍼 경계 셀 (Wrapper Boundary Cells (WBC))이 정의되어 있지만, 본 논문에서는 면적 오버헤드를 고려해서 그림 4와 같은 작은 크기의 래퍼 경계 셀을 사용한다[2, 8]. Update 레지스터가 없으며 하나의 레지스터로 Shift 및 Capture를 수행할 수 있다.




(a) 래퍼 입력 셀 (Wrapper Input Cell (WIC))




(b) 래퍼 출력 셀 (Wrapper Output Cell (WOC))

그림 4. 래퍼 경계 셀
Fig. 4. Wrapper Boundary Cell.

## IV. AMBA Based Core Test Wrapper

이 장에서, IEEE 1500 래퍼를 이용한 AMBA 기반 코어 테스트 래퍼를 제시한다. TIC와 EBI를 통한 테스트 접근 메커니즘은 논문 [7]에서 제안한 ATAM (AMBA based Test Access Mechanism)을 따르고 IEEE 1500 래퍼의 Wrapper Boundary Register (WBR)을 PI와 PO의 임시 레지스터로 이용한다. 이 구조의 장점을 이용하여 스캔 테스트 시간을 줄일 수 있는 방법을 제시한다.

스캔 체인의 수를 *SCn*, PI와 PO의 폭을 각각 *PIn*과 *POn*이라고 할 때, 본 논문에서 제시하는 AMBA Based core test wrapper는 그림 5와 같이 구성된다. AMBA 버스 폭을 고려하여 논문 [7]과 같이 스캔 체인의 수를 32개 이하로 제한하고 스캔 입력과 출력을 위한 임시레지스터는 별도로 두지 않는다. PI와 PO는 32비트씩 그룹화하고 IEEE 1500 wrapper의 WBR을 임시 레지스터로 활용한다.

각 테스트 절차마다 TIC로부터 받은 어드레스를 디

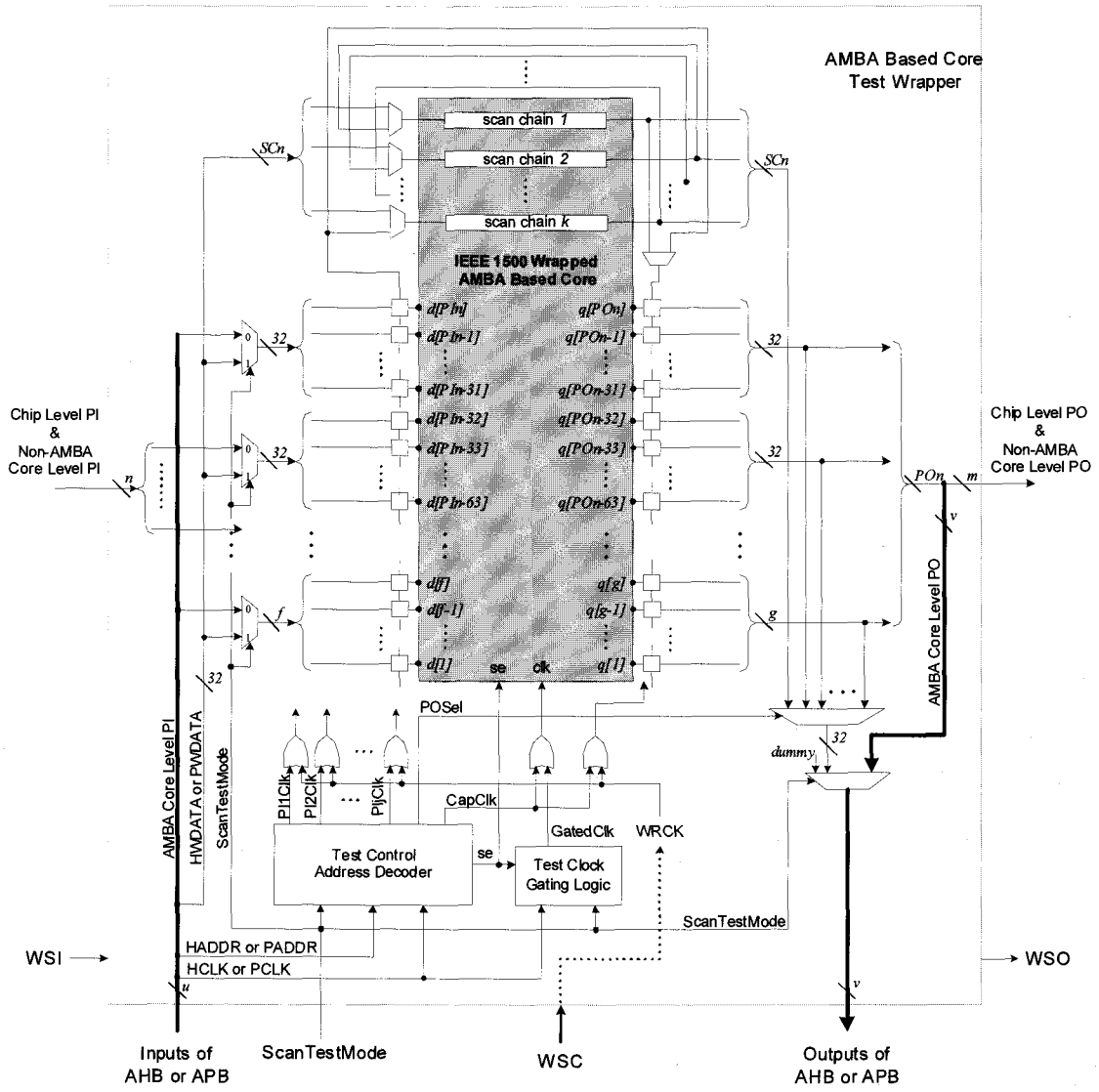



그림 5. IEEE 1500 래퍼를 사용한 AMBA 기반 코아 테스트 래퍼
Fig. 5. AMBA Based Core Test Wrapper using IEEE 1500 Wrapper.

*SCn*: The number of scan chains (≤*32*)
*PIn*: The width of PI
(= "*n* + *u*" for the width of chip level & non-AMBA core level PI '*n*' and the width of AMBA core level PI '*u*')
*POn*: The width of PO
(= "*m* + *v*" for the width of chip level & non-AMBA core level PO '*m*' and AMBA core level PO '*v*')
*f*: The width of last group in PI
*g*: The width of last group in PO

코딩하여 필요한 동작을 수행하기 위하여 Test Control Address Decode와 Test Clock Gating Logic을 포함한다. WSI, WSO 및 WSC와 연결되는 IEEE 1500 circuitry는 생략한다.

### 1. PI/PO Grouping

PI/PO는 크게 chip level PI/PO와 core level PI/PO로 구분 할 수 있으며, core level PI/PO는 AMBA PI/PO (data bus, address bus and control signals)와 non-AMBA I/O로 구분 할 수 있다[5, 7]. TIC를 이용한 스캔 테스트 모드 시에, PI에 테스트 패턴을 인가하고 PO결과를 확인하기 위해서 32비트 데이터 버스를 사용하므로 PI와 PO를 32비트 단위로 그룹화 해야 한다. 그림 5에서 PI가 *j*개의 그룹으로 나누어진다고 할 때, *PIn*은 32의 배수 일수도, 아닐 수도 있으므로 마지막 *j* 번째 그룹의 폭 *f*은 32이하가 된다. PO의 마지막 그룹

의 폭 $g$도 이와 마찬가지로 32 이하가 된다.

### 2. Test Control Address Decoder

스캔 테스트를 위해서 어드레스 지정방식을 사용하여 스캔 입출력, PI 적재, PO 관측, Capture동작을 수행하도록 한다. 각각의 동작 수행을 위한 Test Control Address Decoder의 기능은 아래와 같이 간단하게 요약할 수 있다.

● 스캔 입출력: 스캔 인에이블신호인 se를 생성하고 Test Clock Gating Logic을 동작 시켜 스캔 쉬프트 클럭을 생성하도록 한다. (4.3 참조)

● PI 적재: PI의 각 그룹에 특정 어드레스를 할당하고 각 어드레스에 해당되는 그룹의 WBR에서만 적재할 수 있도록 클럭 펄스 (PI1Clk, PI2Clk, …, PIjClk)를 생성한다.

● PO 관측: PO의 각 그룹에 특정 어드레스를 할당하고 PO Sel 신호를 통하여 PO 각 그룹을 선택하여 관측 할 수 있도록 한다. 스캔 입출력 시에는 스캔 출력을 선택하도록 한다.

● Capture: CapClk를 생성하여 스캔 체인에서 한번의 Capture가 수행되도록 한다. 스캔 체인 Capture 수행 시에 PO의 WBR에서도 동시에 Capture를 수행하도록 한다. (4.4 참조)

### 3. Test Clock Gating Logic

스캔 입출력 수행 시에 원하는 횟수만큼 쉬프트를 수행하기 위해서는 스캔 인에이블신호 생성과 더불어서 안정된 클럭 게이팅 테크닉이 필요하다. 그림 6이 Test Clock Gating Logic이다. Clock Gating Cell (CGC)을 사용함으로써 글리치(glitch) 없이 안정된 Gated 클럭을 생성할 수 있다[9~10].

### 4. Test Procedure

(1)과 같은 기본적인 스캔 테스트 절차를 따르기 위해서는, PI는 동시에 인가 또는 유지될 필요가 있으므로 반드시 레지스터가 필요하며, PO는 코어의 출력에 래치되어 있는 상태에서 32비트 단위로 어드레싱 하여 읽을 수 있으므로 별도의 레지스터가 없어도 된다[5, 7]. 본 논문에서 제시한 바와 같이 IEEE 1500 래퍼의 출력 WBR을 PO 임시 레지스터로 사용하여 스캔 캡쳐 시에 PO 임시레지스터에 PO 캡쳐를 함께 수행하면 (2)와 같이 변형된 절차로써 스캔 테스트를 수행 할 수 있다.

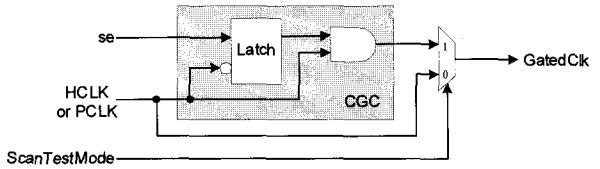



그림 6. 클락 게이팅 셀을 이용한 테스트 클락 게이팅 로직

Fig. 6. Test Clock Gating Logic using Clock Gating Cell.

*Scan In* → *PI Application* → *PO Observation* → *Scan Capture* → *Scan Out* (1)

*PI Application* → *Scan In* → *Scan & PO Capture* → *PO Observation* → *Scan Out* (2)

(2)에서는, *Scan In*전에 *PI Application*을 먼저 수행하고, *PO Observation*을 수행하기에 앞서 *Scan Capture*와 *PO Capture*를 수행하며, 마지막으로 *Scan Out*을 수행한다. 입력 패턴에 대한 테스트 결과 PO를 레지스터에 한번 저장했다가 관측하므로 스캔 테스트 결과는 (1)과 동일하게 된다. 이와 같은 절차를 따르면, TIC와 EBI에 의해 테스트 입출력 경로가 분리 되어 있으므로, Read나 Write중 어떤 transaction을 사용하더라도 PO 그룹에 해당되는 어드레스를 인가하면 PO 관측이 가능하다. 따라서 PI 그룹과 PO 그룹에 같은 어드레스를 할당함으로써 *PI Application*과 *PO Observation*을 동시에 수행 할 수 있다. *PI Application* 부터 *Scan Out*까지를 하나의 테스트 싸이클이라고 하면, 현재 테스트

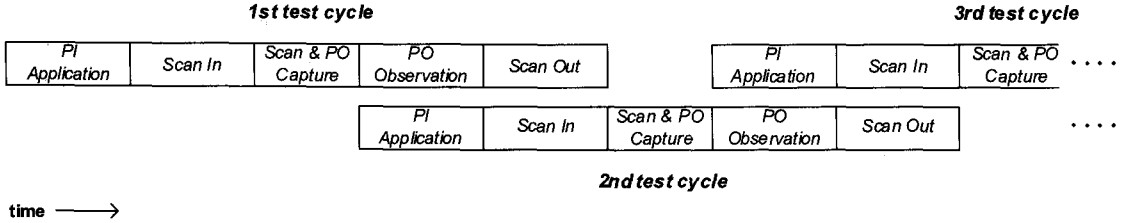



그림 7. 오버랩 되는 테스트 싸이클

Fig. 7. Overlapped Test Cycles.

싸이클의 *PO Observation*과 다음 테스트 싸이클의 *PI Application*이 동시에 수행 되도록 할 수 있다. PI의 폭이 PO의 폭보다 작으면 dummy 데이터를 인가하고, PI의 폭이 PO의 폭보다 크면 유효한 데이터만 관측한다. 결과적으로, 그림 7과 같이 정수 $n$에 대해서, $n$번째 테스트 싸이클의 *PO Observation*과 *Scan Out*단계가, $(n+1)$ 번째 테스트 싸이클의 *PI Application*과 *Scan In*단계와 동시에 수행 될 수 있으므로 테스트 시간을 훨씬 더 많이 줄일 수 있다.

## V. Scan Test Time Analysis & Experiment

본 논문에서 제시한 테스트 래퍼 및 방법을 평가하기 위하여, 테스트 시간을 분석하여 논문 [7]과 비교한다. 테스트 시간을 분석하기 위하여 *PIGn*을 PI 그룹의 수, *POGn*을 PO 그룹의 수, *SCl*을 스캔체인의 길이, *TPn*을 테스트 패턴의 수라고 정의하자. AMBA의 Burst 모드로 테스트를 수행한다고 하면, 테스트 절차의 각 단계는 한 클럭의 어드레스 단계와 여러 클럭의 데이터 단계로 구성된다. *Scan Capture*시에는 데이터 단계가 필요하지는 않지만, 다음 단계인 *Scan Out* 수행에 해당되는 새로운 어드레스 입력을 위해서 TIC 인터페이스에 의해 한 클럭이 소모된다. 따라서 테스트 절차(1)을 따르는 [7]에서 한 번의 테스트 싸이클 동안 소모되는 테스트 클럭 수 $T_{1cycle}$은 다음과 같다.

$$T1_{cycle} = (1+SCl)+(1+PIGn)+(1+POGn)+(1)+(1)+(1+SCl) = 2SCl + PIGn + POGn + 6. \quad (3)$$

연속된 테스트 싸이클 수행시에, *Scan Out*과 *Scan In* 단계만 겹치므로 총 테스트 시간은 *TPn*번의 *PI Application, PO Observation, Scan Capture*, TIC 인터페이스에 의한 한 클럭과 *(TPn+1)*번의 *Scan In/Out*의 합으로 이루어진다. 따라서 논문 [7]에서 제시한 방법의 총 테스트 클럭 수 $T_{total}$ 은 다음과 같다.

$$T_{total} = TPn\{(1+PIGn)+(1+POGn) + (1) + (1)\} + (TPn+1)(1+SCl) = TPn(SCl+PIGn+POGn+4)+SCl+1. \quad (4)$$

본 논문에서 제시한 방법에서, *PO Capture*는 *Scan Capture*와 동시에 발생하고, 테스트 절차의 순서만 바뀌었으므로 한 번의 테스트 싸이클을 수행하는데 걸리는 시간은 (3)과 같다. 그러나 연속된 테스트 싸이클 수행 시에는 *Scan Out*과 *Scan In* 뿐만 아니라 *PO Observation*과 *PI Application*이 동시에 수행되고, *POGn*과 *PIGn*의 크기는 서로 다를 수 있으므로 *PO Observation*과 *PI Application* 시간 중 길게 걸리는 단계에 맞추어 주어야 한다. 따라서 제안한 방식의 $T_{total}$ 은 다음과 같이 구할 수 있다.

$$T_{total} = TPn\{(1) + (1)\} +(TPn+1)[MAX\{(1+PIGn),(1+POGn)\}+ (1+SCl)] = TPn(MAX\{PIGn,POGn\} + SCl + 5) + MAX\{PIGn,POGn\} + SCl + 3. \quad (5)$$

두 방식의 총 테스트 시간의 차를 구하면 (6)과 같다. 이 결과는, 제안한 방식이 [7]에 비해 얼마나 많은 테스트 시간을 줄일 수 있는지를 나타내고 있다. 일반적으로 *TPn*은 *PIGn*이나 *POGn*에 비해서 매우 크므로 $PIGn \geq POGn$이고 $POGn \neq 1$ 이거나 $PIGn \leq POGn$이고 $PIGn \neq 1$ 이면, 제안한 방식을 사용함으로

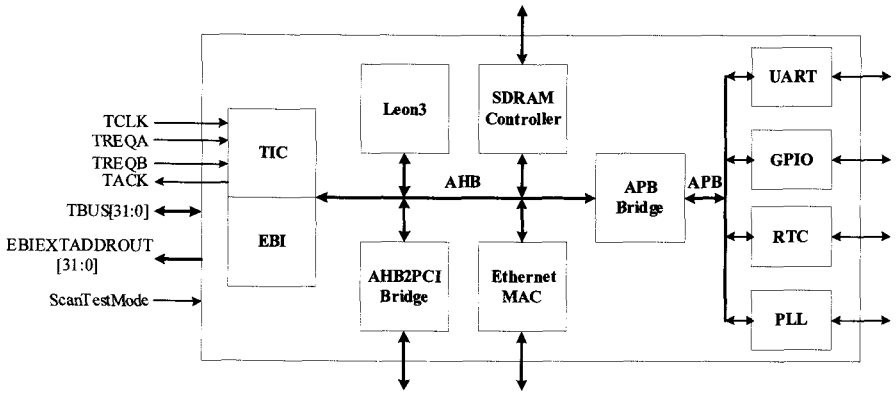



그림 8. AMBA 시스템의 예
Fig. 8. Example of AMBA system.

표 1. 스캔 테스트 시간 비교표
Table 1. Comparison of Scan Test Time

| Cores | Test Time (# of TCLK) | | Reduction rate (%) |
|---|---|---|---|
| | ATAM [7] | Proposed | |
| Leon3 Processor | *35936* | *31345* | *12.78* |
| SDRAM Controller | *2116* | *1792* | *15.31* |
| AHB2APB Bridge | *2538* | *2318* | *8.67* |
| Ethernet MAC | *32053* | *30130* | *6.00* |
| UART | *6024* | *5798* | *3.75* |
| GPIO | *225* | *194* | *13.78* |
| RTC | *2482* | *2356* | *5.08* |

써 상대적으로 더 많은 테스트 시간을 줄일 수 있음을 알 수 있다. *SCI*은 두 방식의 테스트 시간 차이에 영향을 주지 않는다.

$$(4) - (5) = TPn(PIGn + POGn - 1) - (TPn+1)MAX\{PIGn,POGn\} - 2 \quad (6)$$

논문 [7]과 테스트 시간을 비교 평가하기 위하여 그림 8과 같은 AMBA 시스템을 구성하였다. 테스트 래퍼가 Core level PI/PO뿐만 아니라 Chip level PI/PO도 지원 할 수 있다는 전제하에 (4)와 (5)를 이용하여 테스트 시간 비교하였다. 표 1이 논문 [7]과 제안한 방식과의 비교 결과이다. 제안된 방식을 적용함으로써 테스트 시간을 평균 9.34 % 단축시킬 수 있다.

## VI. Conclusions

본 논문에서는, AMBA 기반 내장 코어의 테스트 래퍼를 제시하였다. 기존에 제시된 ATAM을 사용하되 테스트 시간을 단축시킬 수 있는 테스트 래퍼와 테스트 방법을 제시하였다. IEEE 1500 래퍼의 WBR을 PI 임시 레지스터와 PO 임시 레지스터로 활용하고 변형된 테스트 절차를 적용함으로써 Scan In과 Scan Out 뿐만 아니라 PI 인가와 PO 관측도 병행 할 수 있도록 하였다. 주어진 AMBA 기반 코어들에 대한 테스트 시간 분석 결과 평균 9.34 % 테스트 시간을 단축시킬 수 있었으며, SDRAM Controller에 대해서는 15.31 %까지 테스트 시간을 단축시킬 수 있었다. IEEE 1500 래퍼가 있는 AMBA 기반 코어에 제안한 래퍼를 사용함으로써, 테스트 장비의 능력에 따라서 IEEE 1500 기능이나 AMBA 코어 테스트 래퍼 기능을 선택적으로 사용할 수 있을 뿐만 아니라, 적은 면적 오버헤드로 테스트 시간을 단축시킬 수 있으므로 테스트 비용을 절감 할 수 있을 것이다.

## 참 고 문 헌


[1] Y. Zorian, E. J. Marinissen and S. Dey, "Testing Embedded-core based System Chips," Proceedings of IEEE International Test Conference, pp. 130-143, Oct. 1998.

[2] IEEE Computer Society, "IEEE Standard Testability Method for Embedded Core-based Integrated Circuits," Aug. 2005.

[3] ARM IHI 0011A, "AMBA Specification (Rev 2.0)". May 1999.

[4] M. Abramovici, M. Breuer, and A. Friedman, "Digital Systems Testing and Testable Design," IEEE Press, New York, 1990.

[5] C. Feige et al, "Integration of the Scan-Test Method into an Architecture Specific Core-Test Approach," Journal of Electronic Testing, Volume 14, pp. 125-131, July 1998.

[6] C. Lin and H. Liang, "Bus-Oriented DFT Design for Embedded Cores," IEEE Asia-Pacific Conference, Volume 1, pp. 561-563, Dec. 2004.

[7] 민필재, 송재훈, 이현빈, 박성주, "AMBA 기반 SoC 테스트를 위한 접근 메커니즘 설계," 대한전자공학회 논문지, Vol. 43, No. 10, Oct. 2006.

[8] E. J. Marnissen, S. K. Goel and M. Lousberg, "Wrapper Design for Embedded Core Test," IEEE International Test Conference, pp. 911-920, Oct. 2000.

[9] Matthias Beck, Olivier Barondeau, Martin Kaibel, Frank Poehl, Lin Xijiang, Ron Press, "Logic Design For On-Chip Test Clock Generation - Implementation Details and Impact on Delay Test Quality," Proceedings of the Design, Automation and Test in Europe, 2005.

[10] Christian Piguet, "Low-Power CMOS Circuits Technology Logic Design and CAD Tools," Taylor & Francis. 2005.

[11] J. Gaisler and E. Catovic, "Gaisler Research IP Core's Manual," version 1.0.1, Jun. 2005.

## 저 자 소 개


**이 현 빈**(학생회원)

2001년 한양대학교 전자컴퓨터 공학과 학사 졸업.
2003년 한양대학교 컴퓨터공학과 석사 졸업.
2007년 한양대학교 컴퓨터공학과 박사 졸업.

<주관심분야 : SoC 테스팅, ASIC 설계, 네트워크 시스템 설계>

**김 병 진**(학생회원)

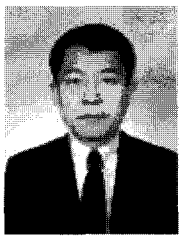
2007년 한양대학교 컴퓨터공학과 학사 졸업.
2007년~현재 한양대학교 컴퓨터 공학과 석사 재학 중.

<주관심분야 : SoC 테스팅, ASIC 설계>

**한 주 희**(학생회원)

2005년 한양대학교 컴퓨터공학과 학사 졸업.
2006년~현재 한양대학교 컴퓨터 공학과 석사 재학 중.

<주관심분야 : VLSI 설계, SoC DFT, 테스팅 Crosstalk>



**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>