반도체디스플레이기술학회지 제18권 제3호(2019년 9월)
Journal of the Semiconductor & Display Technology, Vol. 18, No. 3. September 2019.

멀티클럭 모드를 이용한 병렬 테스트 성능 향상 기법

홍찬의[*]·안진호[*†]
[*†]호서대학교 전자디스플레이공학부

The Method of Parallel Test Efficiency Improvement using Multi-Clock Mode

Chan Eui Hong[*] and Jin-Ho Ahn[*†]
[*†]Hoseo University, School of Electronics and Display Engineering

ABSTRACT

In this paper, we introduce the novel idea to improve parallel test efficiency of semiconductor test. The idea includes the test interface card consisting of NoC structure able to transmitting test data regardless of ATE speed. We called the scheme “Multi-Clock” mode. In the proposed mode, because NoC can spread over the test data in various rates, many semiconductors are tested in the same time. We confirm the proposed idea will be promising through a FPGA board test and it is important to find a saturation point of the Multi-Clock mode due to the number of test chips and ATE channels.

Key Words : Network-On-Chip, Multi-Clock Mode, Parallel Test, ATE, Multi-Site Test

1. 서 론

최근 반도체 공정의 발전과 함께 테스트의 중요성이 커지고 있다. 테스트는 단계마다 실행하며 다양한 테스트가 존재한다. 웨이퍼 테스트를 통과하면 패키지로 만들고, 만들어진 패키지는 여러 테스트를 거쳐 양품의 판별 후 제품으로 판매하게 된다. 패키지 테스트는 테스트 장치(Automatic Test Equipment: ATE)와 테스트 대상 패키지(Device Under Test: DUT) 그리고 핸들러(handler)를 사용하고 ATE와 DUT를 전기적으로 연결해주는 인터페이스 보드가 있다. 이 인터페이스 보드는 패키지의 구성과 특성에 따라 다양하게 제작이 가능하고 외장형 자체 테스트(Built-Out Self Test: BOST) 기능을 수행하기도 한다. 네트워크 온 칩(Network-On-Chip: NoC) 인터커넥트는 적용 분야와 기능에 따라 다양한 형태로 구현이 가능하다. 이를 이용하여 인터페이스 보드를 제작하여 멀티사이트 테스트(Multi-Site Test)를 할 수 있다. 멀티사이트 테스트는 ATE에서 동시에 복수의 DUT를 테스트하는 것이다. 멀티사이트 테스트를 하는 이유는 동시에 여러 개의 패키지를 테스트하여 전체 테스트 시간을 줄임으로써 생산 비용을 줄이기 위해 사용되고 있다[1][2]. 현재 시판 중인 고속 메모리 테스터의 경우 DDR4 기준으로 512개의 메모리를 동시 테스트 가능하며 동작 속도는 2.4Gbps이다[3]. NoC는 대량의 DUT를 병렬로 테스트하기 위해 도입된 데이터 전송 구조이다. 기존의 데이터 버스 구조는 모델링이 편리하며 전송 지연시간이 짧지만 연결할 수 있는 DUT의 수가 제한적이고 연결된 DUT의 수가 증가할수록 제어가 복잡해지는 단점이 있다. 그리고 ATE와 DUT와의 동작 속도에 차이가 있을 경우 이를 보상할 수 있는 정밀한 클럭 제어와 동기화 회로를 설치해야 한다. 하지만 NoC는 인터커넥트를 사용하여 테스트 병렬성을 높일 수 있고 그 구조로 인해 클럭 제어와 동기화 회로를 설치할 필요없이 저속의 ATE를 이용하여 테스트가 가능하다. NoC를 사용한 인터페이스 보드의 병렬 테스트구조는 Fig. 1과 같다. ATE에서 발생한 테스트 데이

[†]E-mail: jhahn@hoseo.edu

터를 인터커넥트 구조를 사용하여 DUT에 인가하고 인가된 DUT에서 테스트 결과를 다시 ATE로 돌려보내 불량을 검출한다.

이에 본 논문에서는 NoC을 이용하여 테스트 병렬성을 높이고 고속으로 데이터 전송이 가능한 NoC의 특성을 이용하여 저속으로 동작하는 ATE로 DUT를 테스트할 수 있는 방법과 구조를 제안하였다.

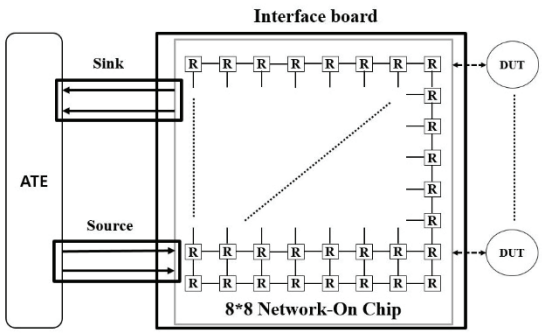

Fig 1. Semiconductor test using an ATE-NoC combination.

2. NoC의 특징

NoC 개발을 위해서 선행되어야 하는 것은 NoC의 동작을 위한 내부 구조와 동작 조건이다. Fig. 2와 같이 NoC는 라우터와 인터커넥트 채널, 네트워크 연결부로 구성된다. Fig. 2의 IP 코어는 테스트하고자 하는 DUT로 볼 수 있다. 인터커넥트 채널은 물리적인 데이터 연결 통로이며 네트워크 인터페이스(Network Interface: NI) 블록은 IP 코어를 라우터와 연결하는 역할을 하며 라우터의 버퍼링 효과를 사용하여 각 모듈 간 동작 속도 차이를 보상할 수 있다[4,5]. 라우터는 채널과 채널을 연결하고 데이터가 입력됐을 때 목적 포트로 전송하는 역할을 하며 이는 라우팅 알고리즘에 따라 달라진다[7].

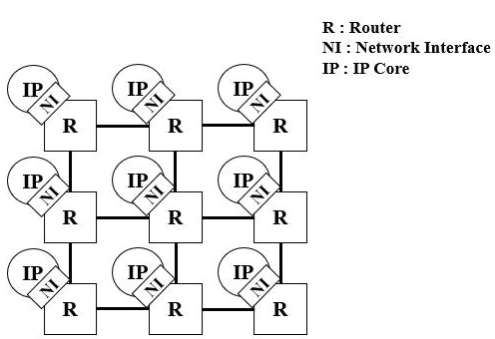

Fig. 2. NoC communication structure.

Fig. 3은 본 논문에서 사용하는 라우터의 내부를 표현한 것이며 테스트 데이터가 입력됐을 때 데이터의 목적지 비트에 따라 해당 방향으로 데이터를 전송한다. 만약, 복수 개의 데이터가 서로 다른 위치에서 전송되어 입력됐을 경우 우선순위가 높은 방향의 데이터를 먼저 전송하고, 우선순위가 낮은 방향은 버퍼에 데이터를 저장해 두었다가 차례대로 전송하는 구조이다.

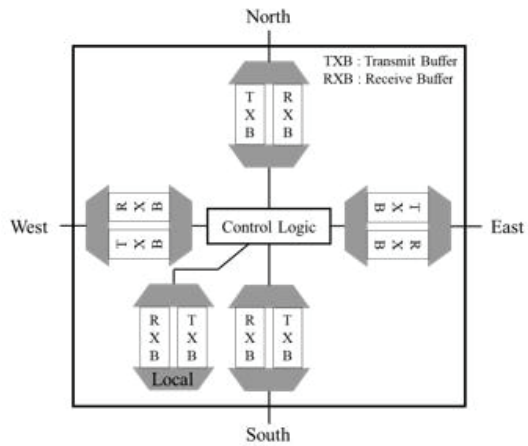

Fig. 3. Internal structure of NoC router.

NoC는 라우터를 배치하고 연결하는 방법에 따라 다양한 토폴로지를 구성할 수 있다. Fig. 4는 정규 형태의 토폴로지를 나타내고 있다. 각 토폴로지마다 다른 기능을 가지고 있으며 Fig. 4 (a)는 메쉬 토폴로지이고 양방향 링크를 사용한다. Fig. 4 (b)는 토러스 토폴로지이고 단방향 링크를 사용한다. Fig. 4 (c)는 트리 기반의 토폴로지이고, 트래픽을 분산시켜 유용하게 사용할 수 있다는 장점이 있다. 토폴로지는 데이터의 전송 및 트래픽 등을 고려하여 선택한다[6,9].

반도체 테스트를 위해서 ATE는 크게 테스트 소스와 싱크 역할을 한다. 테스트 소스는 테스트용 데이터를 발생시키며 생성된 데이터는 NoC와 같은 전송 매체를 통해 테스트하고자 하는 DUT로 이동한다. DUT별 테스트 결과는 다시 전송 경로를 따라 테스트 싱크로 전달된다. 테스트 싱크는 테스트 결과를 저장하고 이를 분석하여 고장의 원인 및 위치 등을 파악한다. 테스트 소스는 ATE 이외에도 자동 테스트 패턴 생성기(ALgorithmic Pattern Generator: ALPG)나 마이크로프로세서 등을 사용할 수 있으며, 테스트 싱크는 분석의 종류와 범위, 시간 등을 고려하여 ATE나 별도의 전용 시그니처 분석기(signature analyzer) 등을 사용한다.

NoC를 전송 구조로 사용하는 경우 모든 데이터는 통신 규격에 따라 헤더(header)와 페이로드(payload), 그리고 트레일러(trailer) 형태로 구성해야 한다. 헤더는 테스트 패턴 데이터

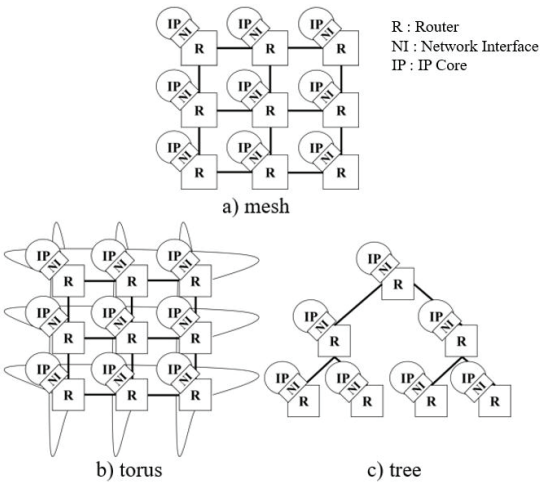

Fig. 4. Configuration of NoC topology.

와 테스트 결과 데이터를 구분하는 값과 목적지 주소, 패킷 전송방식 등으로 구성된다. 페이로드는 테스트하고자 하는 DUT의 테스트 패턴 값 또는 DUT별 테스트 결과 값이 포함된다. NoC의 테스트 데이터 전송 방식은 크게 유니캐스트(unicast) 방식과 멀티캐스트(multicast) 방식으로 구분한다. 유니캐스트 방식은 DUT별로 상이한 패턴을 전송할 때 사용하는 1:1 통신 방식이며, 멀티캐스트 방식은 여러 DUT에 동일한 패턴을 전송할 때 사용하는 1:N 통신 방식이다[8].

3. NoC를 이용한 멀티클럭 테스트 모드

3.1 ATE 와 NoC 의 동작속도가 같은 경우 ($f_{ATE} = f_{NoC}$)

ATE 동작속도(f_{ATE})와 NoC의 동작속도(f_{NoC})가 동일하면 테스트 소스와 NoC는 클럭 단위로 테스트 데이터를 생성하고 전송하게 된다. 싱글 클럭 사용시 ATE의 테스트 소스는 테스트를 해야 하는 DUT와 1:1 로 연결되어 해당 DUT의 테스트가 끝날 때까지 클럭에 맞춰 테스트 데이터를 전송한다. 싱글 클럭이기 때문에 한번에 복수 개를 테스트할 수 없으며 한 개의 테스트가 완벽히 끝나야 다음 DUT를 테스트할 수 있다. NoC는 파이프 라인과 같은 연결을 제공하며 DUT는 테스트 소스로부터 데이터를 받아 테스트하고 테스트 결과 패턴을 싱크로 보내 결과를 분석하게 한다. 한 개의 DUT 테스트가 끝나면 새로운 DUT를 찾고 서로 연결하여 전과 동일한 동작을 한다.

Fig. 5의 테스트 소스는 하나의 DUT와 연결하고 테스트를 시작한다. 테스트 데이터는 매 클럭 연속으로 보내지며 DUT 또한 해당 데이터의 테스트가 끝날 때마다 매 클럭 테스트 결과를 싱크로 보낸다. 싱글클럭 모드는 ATE와 NoC가 동일 클럭을 사용하기 때문에 순차적으로 동작하는 방법이다.

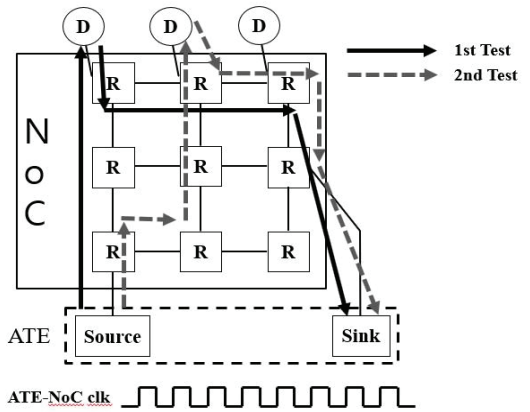

Fig. 5. Single-Clock mode operation example.

3.2 ATE 와 NoC 의 동작속도가 다른 경우 ($f_{ATE} \neq f_{NoC}$)

ATE의 동작속도와 NoC의 동작속도가 다르면 ATE와 NoC는 서로 다른 클럭 소스에 맞춰 동작한다. 클럭의 속도 차이에 따라 사용자가 따로 회로를 만들 필요는 없으며 이는 상기 설명한 NoC내의 네트워크 인터페이스 블록에 의해 보상된다. 본 논문에서 제안하는 멀티클럭 모드는 저속의 ATE를 고속으로 동작하는 NoC와 연결하여 복수개의 DUT를 동시에 테스트하는 방법이다. 예를 들어 NoC의 속도가 2배 빠르다면 테스트 소스와 싱크를 최대 2개까지 연결 가능하고, 동시에 2개의 DUT를 병렬 테스트할 수 있다(그림 6 참조). 이 때 테스트 소스와 싱크의 동작 타이밍은 그림 7과 같다. 그림과 같이 NoC 클럭을 2분주하고 분주된 클럭의 위상차를 180도로 만들어 각 테스트 소스와 싱크의 동작 타이밍을 분리한다.

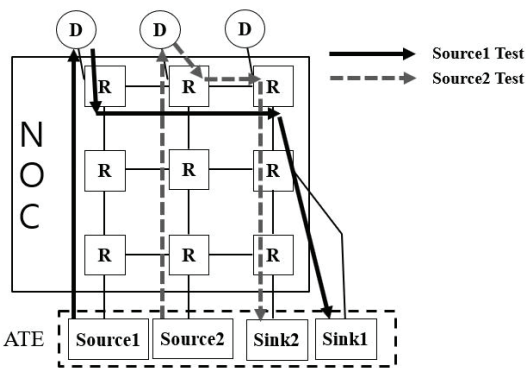

Fig. 6. Multi-Clock mode operation example.

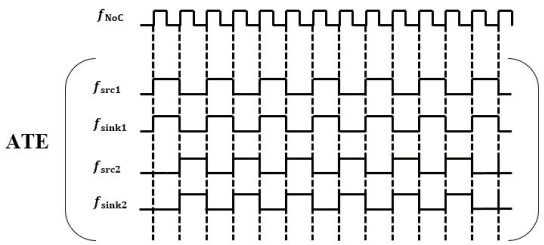

Fig. 7. ATE-NoC operation timming.

4. 실험 결과

멀티클럭 모드를 이용한 병렬 테스트 효과성 분석은 상용 DE2-115 FPGA보드를 이용하였으며, 상기 보드에 Simple Parameterizable NoC[11]을 이식하여 실험하였다. 실험에 사용된 NoC의 주요 파라미터는 다음과 같다.

- 채널 폭: 64bit
- 크기: 8*8
- 토폴로지: Mesh
- 라우팅 기법: XY routing
- 패킷전송 방법: Unicast mode

상기 NoC의 크기와 채널 폭 등은 DE2-115보드의 구조를 고려하여 실험적으로 결정하였다. 실험에 사용한 DUT는 96Mb(32Mb*3) 크기의 메모리를 가정하였으며, NoC에 연결된 DUT 총 수는 48개로 하였다. 각 DUT별 포트에는 LED를 연결하여 테스트패턴 및 결과 입출력 여부를 확인할 수 있도록 하였다.

테스트 소스, 싱크, 그리고 DUT의 NoC 배치 구조는 Fig. 8과 같다. 각 테스트 소스 별로 테스트할 DUT의 위치와 테스트 순서를 사전에 입력하였다. 본 실험에서 첫번째 소스의 경우 두번째 소스의 전 열까지 테스트하고, 두번째 소스의 경우 해당 열부터 다음 소스, 또는 마지막 열까지 테스트를 진행한다.

Table 1은 실험 결과를 나타낸 것이며 NoC 속도 50MHz, 25MHz일 때를 기준으로 ATE의 속도를 계속 낮추면서 실험하였다. 테스트 소스와 싱크의 개수는 최대 4개로 제한하였으며 모든 실험은 최대 개수의 소스를 연결하였다. 비교 결과로 사용된 예상시간은 싱글클럭 모드에서 측정된 시간을 기준으로 하였다. 이를 기준으로 멀티클럭 모드의 효과성을 분석할 수 있도록 예상시간 대비 테스트시간 개선율을 실험결과로 제시하였다.

Table 1의 실험 결과를 보면 NoC의 속도가 50MHz일 때 라우터에서 데이터의 전송이나 DUT의 테스트 결과 전송 등 동작이 25MHz일 때보다 빠르므로 측정시간이 더 빠른 것을

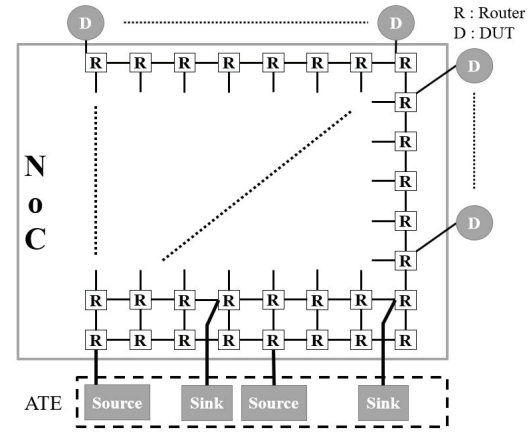

Fig. 8. NoC layout for experiments.

Table 1. Experimental results

NoC 속도 (MHz)	25		
예상시간(s)	566		
ATE 속도 (MHz)	소스의 개수(개)	측정시간(s)	개선율(%)
25	1	566	0
12.5	2	368	35
10	2	368	35
8.3	3	230	59.4
6.25	4	322	43.1
5	4	322	43.1
NoC 속도 (MHz)	50		
예상시간(s)	283		
ATE 속도 (MHz)	소스의 개수(개)	측정시간(s)	개선율(%)
50	1	283	0
25	2	184	35
16.67	3	138	51.2
12.5	4	138	51.2
10	4	184	35
8.3	4	230	18.7
7.14	4	276	2.5

볼 수 있다. 또한, 모든 경우에서 멀티클럭을 사용할 때 테스트시간이 개선됨을 확인하였다.

그러나, 분석 결과 NoC의 속도에 따라 최적의 ATE속도가 있으며 대략 2~4배의 속도 차이가 났을 때 테스트시간 개선율이 가장 증가하는 것을 알 수 있다. 표에서 보면 동작속도가 50Mhz일 때는 속도의 차이가 5배 이상 날 때부터 테스트시간 개선율이 둔화되는 것을 볼 수 있다. 이는 테스트 싱크

와 소스의 개수를 최대 4쌍으로 제한했기 때문이기도 하지만 NoC와 ATE의 큰 속도 차이로 인하여 NoC의 동작 유휴 시간이 증가하는 것도 그 원인이 될 수 있다. 따라서 최적의 테스트 결과를 얻기 위해서는 ATE-NoC구조별로 적정한 멀티클럭 모드와 테스트 소스 및 싱크 수를 결정해야 한다.

5. 결 론

본 연구에서는 멀티클럭 모드를 사용하여 병렬 테스트의 성능을 향상시키는 방법에 대하여 제안하였다. 멀티클럭 모드는 NoC의 특성을 고려, NoC와 ATE 동작 속도를 분리하여 동시에 여러 개의 DUT를 동시에 테스트할 수 있는 방법이다. 실험 결과, 기존 싱글클럭 방식 대비 대부분의 멀티클럭 환경에서 테스트시간이 감소됨을 확인할 수 있었다.

그러나 멀티클럭 모드에서는 NoC에 연결된 ATE나 DUT의 위치, NoC와 ATE의 속도 차이 정도, 테스트 소스와 싱크의 수 등에 따라 테스트시간이 변화할 수 있다. 따라서 ATE-NoC구조 별 최적의 테스트 환경에 대한 실험적 탐색이 요구된다.

감사의 글

This research was supported by the MOTIE (Ministry of Trade, Industry & Energy (project number G3-37) and KSRC (Korea Semiconductor Research Consortium) support program for the development of the future semiconductor device.

참고문헌

1. J. Rivoir, “Lowering Cost of Test: Parallel Test or Low-Cost ATE?”, Proc. of ATS(ATS`03), 2003.
2. N. Velamati and R. Daasch, “Analytical Model for Multi-site Efficiency with Parallel to Serial Test Times, Yield and Clustering”, Proc. of VTS, 2009.
3. EXICON Corp, http://www.exicon.co.kr/sub/sub02_01.php.
4. Arteris, “A comparison of Network-on-Chip and busses”, white paper, 2005.
5. D. Bertozzi, A. Jalabert, S. Murali, R. Tamhankar, S. Stergiou, L. Benini and G. D. Micheli, “NoC synthesis flow for customized domain specific multiprocessor systems-on-chip”, *IEEE Trans. on TPDS*, Vol. 16, pp. 113-129, 2005.
6. S. Murali and G. D. Mucheli, “SUNMAP: a tool for automatic topology selection and generation for NoCs”, Proc. of DAC, 2004.
7. E. Bolotin, I. Cidon, R. Ginosar and A. Kolodny, “QNoC: QoS architecture and design process for network on chip”, Journal of Systems Architecture, Vol. 50, pp. 105-128, 2004.
8. C. Hong and J. Ahn, “Improving Parallel Testing Efficiency of Memory Chips using NOC Interconnect”, *Trans. of KIEE*, Vol. 68, No. 2, pp. 364-369, 2019.
9. T. Bjerregaard and S. Mahadevan, “A survey of research and practices of Network-on-chip“, *ACM Computing Surveys*, Vol. 38, No. 1, pp. 1−51, 2006.
10. W. Zhang, L. Hou, J. Wang, S. Geng and W. Wu, “Comparison Research between XY and Odd-Even Routing Algorithm of a 2-Dimension 3X3 Mesh Topology Network-on-Chip”, Proc. of WRI GCIS, 2009.
11. Simple Parameterizable Network-on-Chip, https://github.com/gtarawneh/simpnoc

접수일: 2019년 8월 31일, 심사일: 2019년 9월 19일, 게재확정일: 2019년 9월 25일