

경계스캔 구조를 사용한 시스템의 온라인 버스 모니터링

論 文

49D-12-5

On-line Bus Monitoring of a System Using Boundary-Scan

宋東燮*·裴相民*·姜成昊**·朴英鎬***

(Dong-Sup Song · Sang-Min Bae · Sungho Kang · Young-Ho Park)

Abstract - When a system is composed of multi-boards, an efficient bus arbitration method for the data transfer bus must be provided for guaranteeing proper operations. In this paper, a new test methodology is developed which is used for testing on-line bus arbitration. In the new test methodology, events that are occurred during bus arbitration are defined, and expected signals during fault-free bus arbitration are compared with the signals captured during on-line bus arbitration using boundary-scan cells. For this, a new test architecture is proposed which is efficient for the maintenance and the repair of multi-board systems. In addition, the new methodology can be used with off-line interconnect test using boundary-scan.

Key Words : boundary scan, bus arbitration, on-line monitoring, interconnect test

1. 서 론

실용화되는 칩의 고집적화와 인쇄 회로 기판의 표면 실장 기술(SMT: Surface Mounting Technology)의 점증적 발달은 디지털 시스템의 복잡성과 밀도를 증가시켰다. 이런 변화는 시스템의 성능을 향상시키고 인쇄 회로 기판(PCB: Printed Circuit Board)의 크기를 감소시키는 장점을 가져온 반면 시스템 테스트를 어렵게 만들었다. IEEE 표준 P1149.1 경계 주사 구조(boundary-scan architecture)는 더 이상 전통적인 인서트(in-circuit) 테스트 방법으로 인쇄 회로 기판을 테스트하기가 불가능 해짐에 따라 고안된 테스트 용이화(DFT: Design For Testability) 기법중의 하나이다[1][2][3].

경계 주사 구조는 보드 내 각 소자의 입·출력 핀에 주사 이동 레지스터(scan shift register)를 첨가하고 이들을 하나의 주사 사슬(scan chain)로 연결하여 각 소자들의 제어 용이도(controllability)와 관측용이도(observability)를 향상시키는 역할을 한다[4][5]. 경계 주사 구조는 다양한 종류의 테스트에 적용될 수 있다. 보드에 장착된 칩들의 이상 유무를 판단하는 칩 테스트, 보드 내 칩들 간 상호연결선(interconnect)의 단선·합선 여부를 테스트하는 상호 연결선 테스트, 그리고 시스템 온 칩(SOC: System-On-a-Chip) 경향에 보조를 맞추어 칩 내의 모듈에 대한 접근 수단으로의 사용 등 다양한 연구가 진행되어 왔다[6][7].

최근 전자회로의 구성은 규모가 커지고 기능이 다양해짐에 따라 한 개의 보드로 전체 시스템을 구성하는 것은 불가능할 뿐만 아니라, 비효율적이다. 그러므로, 시스템의 구성이 다중 보드를 통해서 이루어진다. 이런 다중보드 환경을 갖는 시스템을 테스트하기 위해서는 기존에 IEEE 표준 P1149.1 경계 주사 구조의 확장이 필요하게 되고 이에 따라 BT-Link (Backplane-Test bus Link)나, IEEE 표준 P1149.5 MTM(Module Test and Maintenance) 버스 프로토콜 등이 연구되었다[8][9]. 하지만, 이런 연구들은 시스템이 동작 중이지 않는 오프라인(off-line)에서의 방법들로, 잠재되어 있는 고장을 고려하지는 못한다. 잠재되어 있는 고장이란, 시스템에 존재하는 결함이 오프라인 상에서는 고장으로 나타나지 않다가, 시간이 경과한 후에 비로소 오동작으로 영향이 나타나는 고장을 의미한다. 이런 문제는 시스템이 운행중인 온라인(on-line) 상에서 발생하는 높은 온도로 인해 인쇄 회로 기판의 상호 연결선이 손상을 입는 등의 원인으로 발생할 수 있다.

다중의 보드를 갖는 시스템은 하나의 시스템 버스(bus)를 여러 개의 보드가 공유하고, 적절한 버스 중재(arbitration) 스케줄링에 의해서 데이터를 전송한다. 물론 시스템을 운행하기 전에 오프라인 테스트에서 보드내의 상호 연결선뿐만 아니라, 시스템 버스의 상호 연결선에 대해서도 테스트를 실행한다. 그러나 만일 시스템 버스를 중재하는 상호 연결선 중에 오프라인에서는 검출되지 않은 잠재되어 있는 고장이 존재해서, 정상적인 운행 동작 중에 그 영향이 발생한다면, 이 고장을 감지·진단하고 오프라인으로 전환 후 고장 수리를 할 수 있는 시스템 유지·보수의 수단이 필요하다. [10]에서는 온라인 테스트의 필요성과 함께, 시스템 버스가 아이들(idle)상태에 존재할 때 테스트 벡터를 인가해서 온라인 테스트를 수행할 수 있음을 주장했으나, 이 방법은 시스

* 準 會 員 : 延世大 工大 電氣電子工學科 碩士課程

** 正 會 員 : 延世大 工大 電氣電子工學科 副教授 · 工博

*** 正 會 員 : 韓國電子通信研究院 前任技術員

接受日字 : 2000年 10月 14日

最終完了 : 2000年 11月 11日

템 버스란 여러 개의 보드가 공용으로 사용하는 자원이므로, 버스의 아이들 상태가 자주 존재하지 않을 수밖에 없기 때문에 문제가 있다. 또 버스가 아이들 상태에 존재한다 하더라도 언제 버스를 통해서 데이터 전송을 시작할 지 예상할 수 없기 때문에 적절한 테스트 삽입에는 무리가 있다.

본 논문에서는 경계 주사 구조를 사용한 다중 보드 시스템에서 시스템 버스 중재 동작을 온라인 모니터링 하는 방법에 대해서 연구한다. 경계 주사 구조의 데이터 온라인 캡처능력과 직렬 이동을 통한 데이터 분석능력을 통해서 버스 중재 신호선에 존재하는 고장을 검출할 수 있는 테스트 구조와 방법을 개발한다. 이후의 구성은 다음과 같다. 2장에서는 온라인 시스템 버스 중재 모니터링에 사용되는 경계 스캔 구조에 대해서 간단하게 살펴보고, 3장에서는 오프라인 테스트와는 달리 온라인 테스트가 갖는 어려움에 대해서 알아본다. 그리고 4장에서는 본 논문의 대상이 되는 시스템에 대해서 모델링 함과 동시에, 온라인 모니터링의 대상을 결정하고, 그 타당성을 제시한다. 그리고 5장은 이런 시스템에서 경계 스캔 구조를 사용하여, 버스 중재를 모니터링 하는데 필요한 구조와 방법을 설명한다. 마지막으로 6장에서는 결론과 함께 기대되는 테스트에서의 개선점을 짚어본다.

2. IEEE 표준 P1149.1 경계 주사

경계 주사는 조절용이도와 관측용이도를 향상시키기 위해 칩의 모든 입·출력 핀에 경계 주사 이동 레지스터를 덧붙인 접근 방법을 사용한다. 그림 1은 경계 주사 표준 구조를 나타내고 있다. 이 중 Test Access Port(TAP), TAP 제어기(TAP controller), 명령어 레지스터는 경계 주사에 있어서 필수적인 하드웨어 요소이다. 또한 여러 가지 테스트 데이터 레지스터군중의 경계 주사 레지스터와 바이패스 레지스터(bypass register)는 필수적인 구성요소이다[11].

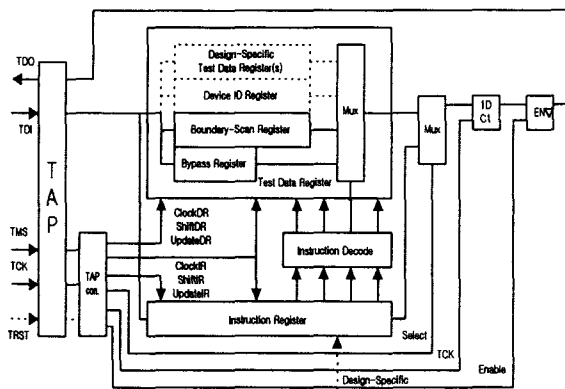


그림 1. 표준 경계 주사 구조
Fig 1. IEEE standard boundary scan architecture

TAP은 칩의 핀들로 구성되며 Test Data Input(TDI), Test Data Output(TDO), Test Mode Select(TMS), Test Clock(TCK)의 네 신호선을 입력으로 갖는다. 여기에 선택적으로 테스트 중인 회로를 비동기로 초기화하기 위한 Test Reset(TRST)가 사용될 수 있다. TAP제어기는 TAP으로

부터 제어 신호를 받는 순차회로로 이루어져 있으며 이는 테스트 버스를 통해서 입·출력되는 테스트 데이터에 의해서 제어된다. 그림 1에서 보듯이 각각의 레지스터들은 멀티플렉서를 통해서 TDI로부터 TDO까지의 직렬 사슬을 구성하도록 선택된다. 또한 오른쪽의 마지막 멀티플렉서는 직렬 사슬을 명령어 레지스터로 연결할 것인지 아니면 테스트 데이터 레지스터 군으로 연결할지를 제어한다. 이러한 사슬의 선택은 명령어 레지스터에 전달된 명령을 통해서 제어된다.

그림 2는 경계 주사 레지스터(boundary scan register: BSR)셀을 보여준다.

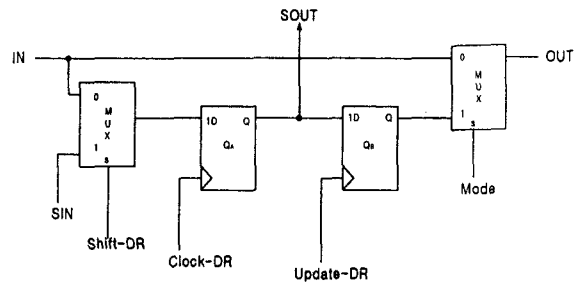


그림 2. 경계 주사 레지스터
Fig 2. Boundary scan register

모드신호는 정상적인 기능신호(functional signal)와 주사 이동 레지스터 레치 중 하나의 경로를 선택한다. 모든 주사 이동 레지스터는 직렬 경로로 연결되어 있다. 여기에서 한 가지 주의 깊게 볼 것은 기능신호는 주사 이동 레지스터에 의해서 샘플 되어 직렬 주사 이동 레지스터를 통해 칩의 핀으로 관찰되어 질 수 있다는 점이다.

IEEE 표준 P1149.1은 반드시 실행되어야 할 명령어로 다음의 세 가지를 규정하고 있다.

- EXTEST 명령어
칩간의 상호 연결선을 테스트 할 수 있는 명령어로 칩의 입력에 붙은 셀에서 테스트 응답을 관찰하고, 칩의 출력에 붙은 셀이 테스트 입력으로 동작하는 명령어이다.
- SAMPLE / PRELOAD 명령어
기능적인 신호를 주사 이동 레지스터로 샘플 하거나, 경계 주사 레지스터를 원하는 값으로 세팅 할 때 사용하는 명령어이다.
- BYPASS 명령어
이 명령어에서 모든 경계 주사 구조는 1비트의 레지스터로 동작하며, 경계 주사 사슬을 구성하는 칩들 중에서 테스트에 관계하지 않는 칩들은 통과시키는 역할을 한다.

이 외에도 IEEE 표준 P1149.1은 여러 가지의 선택적으로 사용할 수 있는 명령어들을 규정하고 있고, HIGHZ, CLAMP, RUNBIST, IDCODE, USERCODE 명령어가 그것들이다.

3. 경계 주사 구조를 이용한 온라인 테스트의 어려움

일반적으로 모든 전자회로는 제조 시에 높은 고장 검출율

을 기준으로 오프라인 테스트의 단계를 거친다. 칩에 대해서는 완전스캔(full scan), 부분스캔(partial scan), 내장된 자체 테스트(BIST: Built-In-Self-Test), 전류 테스트 등의 여러 가지 테스트 용이화 설계를 적용하여 기능을 검증하고, 보드나 시스템 또한 상호 연결선에 대해서 단락과 합선의 테스트 과정을 거친다. 그림 3은 일반적인 전자회로의 테스트 과정을 나타낸다. 그림 3에서 보듯이 테스트하려는 전자회로에 대해서 고장검출 가능한 알고리즘이 적용된 테스트 벡터(test vector)를 인가하여 얻은 응답 벡터(circuit response)와 고장이 없을 경우의 전자회로에 대해 미리 예상이 된 응답 벡터(true response)를 비교하여 고장의 유·무를 판단한다. 하지만, 이 고장 검출의 방법은 전자회로가 오프라인 상태에 존재할 때 사용되는 방법으로, 전자회로 자체의 기능을 수행중인 온라인 상태에서 동일하게 적용하는 데는 여러 가지 어려움이 존재한다. 그 이유는 다음과 같다.

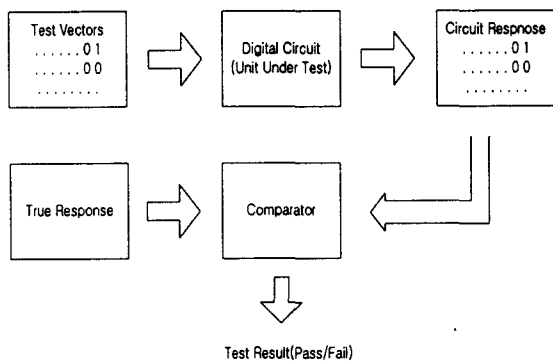


그림 3. 일반적인 테스트 절차
Fig 3. General testing process

우선 고장을 검출할 수 있는 알고리즘이 내재된 테스트 벡터를 삽입하는데 어려움이 존재한다. 온라인 동작중인 전자회로는 그 본연의 기능을 수행하기 위해서 계속적으로 변화하는 데이터 값을 처리한다. 그 데이터는 일정한 패턴이 존재하는 것이 아니고 처리중인 프로세스의 종류에 따라 예측을 할 수 없다. 이런 기능적 데이터를 고장 검출을 위한 테스트 벡터로 사용하기에는 고장 검출의 알고리즘이 내재되어 있지 않으므로 적당하지 않다. 또한 온라인 동작중인 시스템에 오프라인과 같이 테스트 벡터를 삽입하는 것도 문제가 있다. 왜냐하면, 이 때의 시스템 데이터는 타이밍에 대해서 매우 민감한 동작을 수행하므로 이런 타이밍을 무시하고 데이터의 흐름 중간에 테스트 벡터를 삽입하는 것은 어려운 문제이다. 이와 연계하여 테스트 대상의 전자회로에 대해서 고장이 없을 경우의 테스트 응답을 예측할 수 없다는 어려움이 존재한다. 왜냐하면, 온라인 상에서 테스트 벡터에 해당하는 기능적인 벡터는 앞에서 밝힌 바와 같이 처리중인 프로세스의 종류에 따라서 예측이 어렵기 때문에, 가능한 모든 입력 조합에 대해서 고장이 없을 경우의 테스트 응답을 정해 둔다는 것은 불가능하다.

일반적으로 오프라인 테스트에서 보드나 시스템의 상호 연결선 테스트에 대한 완결성을 가늠하는 척도로 고장 검출

을 사용한다. 상호연결선 상에서 고장이라고 가정된 모든 결합 중에서 인가한 테스트 벡터로 검출이 가능한 것이 어느 정도 인가를 나타내는 수치이다. 그러나, 시스템이 기능적으로 동작을 하고 있는 상황에서 오프라인과 동일한 범위의 고장 모델과 고장 검출율을 적용하기는 앞에서 밝힌 온라인 상에서 테스트를 적용하는데 겪는 어려움 때문에 무리가 있다. 더욱이 경계 주사 구조는 직렬 사슬이 갖는 특성 때문에 테스트 벡터의 적재와 그 응답의 분석에 많은 수의 데이터 이동을 필요로 한다. 그러므로, 빠른 속도로 변화하는 데이터를 모두 실시간에 분석한다는 것은 불가능하다. 결론적으로 온라인 상에서는 시스템 운행에 치명적인 손상을 줄 수 있는 상황의 고장 모델을 통해서 적당한 모니터링을 통한 테스트 방법이 필요하다고 할 수 있다.

최근 전자회로로 구성된 하나의 시스템은 다양한 기능을 기본적으로 추구한다. 이런 환경에서는 시스템의 유지·보수의 편리를 위해 보통 보드 단위로 기능을 모듈화 시켜서 구성을 하게 된다. 예를 들면, 퍼스널 컴퓨터의 그래픽 보드나, 사운드 카드 등이다. 다중 보드로 구성된 하나의 시스템은 백플레인(backplane)의 버스를 통해서 서로 간에 데이터 이동을 하게 되는데, 이때 보드의 수는 여러 개가 존재하는 반면에 이들간의 연결선인 버스는 한 개만이 존재하므로 버스에서의 데이터 충돌을 막기 위해서는 버스 사용의 권한을 효과적으로 보드들에게 중재(arbitration)하는 계획이 필요하고, 모든 버스 규격은 이런 계획을 포함한다. 시스템이 정상적으로 운용되고 있는 상황의 버스 중재 계획에서 발생하는 고장은 그 파급이 충분히 예상되는 바와 같이 치명적이다. 이것은 시스템의 국지적인 손실이 아니라, 전체 시스템의 엄청난 손상으로 이어질 수 있다. 결과적으로 오프라인에서는 고장으로 검출되지 않은 결함으로 인해서 온라인 중 버스 중재 계획에 고장이 발생하는 지의 여부를 모니터링하고, 그 결과를 리포트 하는 테스트 모듈이 필요하다.

4. 버스 중재 모니터링 모델링

이번 장에서는 버스 중재 모니터링의 대상신호와 모니터링 대상이벤트를 모델링한다. 버스 중재시 일어나는 일련의 과정을 분석하여 온라인 모니터링 대상 신호를 결정하고, 오프라인 테스트에서 알고리즘이 담긴 테스트 벡터와 동일한 역할을 버스 중재 신호가 온라인에서 달성할 수 있는 방법을 설명한다.

4.1 버스 중재 모니터링 대상 신호

버스 중재는 각 보드로부터 버스 사용 요청을 받아 우선 순위가 높은 보드에게 먼저 제어권을 넘겨주는 역할을 하는 중재기(arbiter)와 보드 내의 데이터 처리 장치(data processing device)가 버스의 사용을 원할 때 동일 보드의 리퀘스터(requester)에게 알리거나 혹은 중재기로부터 버스 사용 허락을 받은 상태에서 버스로의 드라이버를 활성화시키는 마스터(master), 그리고 마스터와 중재기 사이에 존재하여 두 모듈 사이의 데이터를 받아 처리하는 리퀘스터의 3 가지 기능 모듈로 구성된다. 그림 4는 버스 중재 모듈들 간의 관계를 보여 준다.

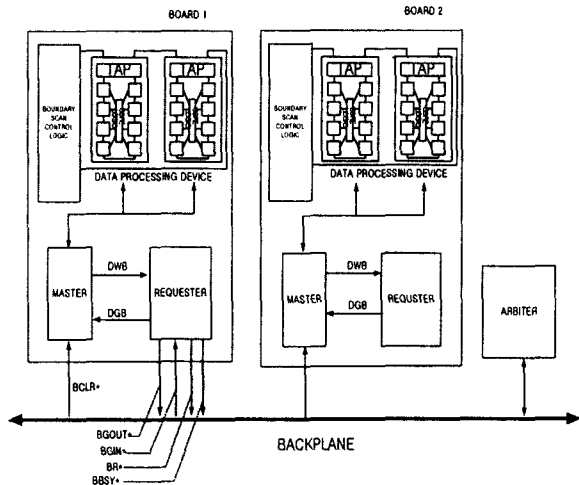


그림 4. 버스 중재 모듈
Fig 4. Bus arbitration modules

그림 4에서 볼 수 있는 것과 같이 각 보드에 존재하는 버스 중재 모듈들 사이에는 정해진 프로토콜에 따라서 정보를 전달하는 신호선들이 존재한다. 이 신호선들의 조합에 따라 동시에 두 개의 마스터에 의해서 버스의 사용이 없도록 하고, 다중의 마스터가 버스의 사용을 요청했을 때 하나의 마스터에게 우선권을 부여하는 스케줄링이 이루어진다. 이런 역할을 하는 버스 중재 신호들과 그 기능은 다음과 같다. 여기서 *표시는 액티브 로우(active low)의 신호를 의미한다.

■ Device Wants Bus (DWB)

데이터 처리 장치가 버스를 사용해서 데이터의 전송을 할 필요가 발생했을 때 마스터는 이를 감지하여 리퀘스터에게 DWB신호를 통해서 이를 알린다.

■ Device Granted Bus (DGB)

중재기로부터 버스의 사용을 허락 받은 보드의 리퀘스터는 버스의 제어권이 이제 자신의 보드에 있고, 버스로 데이터를 내보내도 됨을 알리는 수단으로 DGB 신호를 사용한다.

■ Bus Request (BR*)

DWB가 트루(true)로 활성화되는 것을 감지한 리퀘스터는 중재기에게 버스의 사용을 요청하는데 이것은 BR*신호를 로우 레벨로 활성화시킴으로 가능하다.

■ Bus Grant Line (BGIN*, BGOUT*)

입력되는 모든 BR*신호 중에 최우선 순위의 리퀘스터 신호에 대해서 중재기는 버스의 사용권한을 BGIN*을 통해서 허락한다. 만일 DWB 신호가 액티브 되지 않은 상태에서 중재기로부터 BG*의 액티브 신호가 들어오면, 보드에서 이 신호를 캡처하지 않고 BGOUT*를 통해서 통과시킨다.

■ Bus Busy Line (BBSY*)

중재기로부터 버스의 사용을 허락 받은 보드의 리퀘스터는 버스의 제어권을 가지고 있다는 표시를 다른 모든 시스템에게 알려야 하는데, 이는 BBSY* 신호선을 로우레벨로 드라이브하여 이루어진다.

■ Bus Clear Line (BCLR*)

현재 버스를 사용해서 데이터를 전송중인 보드의 버스 사용

우선권 보다 높은 우선 순위의 버스 사용 요청이 발생했을 때, 중재기는 보드의 마스터에게 BCLR*를 로우레벨로 드라이브하여 이를 알린다.

이 신호들 중에 모니터링 대상 신호는 버스 중재 동작이 발생하는 원인이 되는 신호들로서 DWB, DGB, BR*, BGIN*, BCLR*의 5개로 정한다. 즉, 버스 중재 동작은 이들 5개의 신호 조합에 의한 이벤트에 따라 발생하는 BBSY*나 BGOUT*의 값에 의해서 이루어진다.

4.2 버스 중재 절차

오프라인 테스트와 다르게 온라인 모니터링은 시스템의 동작에 중요한 영향을 미칠 수 있는 고장의 상황을 모니터링 대상 이벤트로 정하고, 이벤트가 발생했을 때 생기는 시스템의 반응 관찰을 통해 고장의 유무를 판단한다. 버스 중재시 발생하는 고장을 검출하기 위해서는 고장이 없을 경우에 버스 중재 프로토콜에 따라 발생하는 버스 중재 신호들의 값을 정의해야 한다. 그림 5는 다중 보드의 시스템에서 임의의 보드가 버스 사용을 원할 때 발생하는 버스 중재 과정을 나타낸다. 그림 5에서 볼 수 있는 일련의 과정들은 시스템의 종류가 무엇이던지, 혹은 보드의 종류가 무엇이던지 상관없이, 버스를 사용해서 데이터를 전송하려고 하는 보드에서는 동일하게 발생하는 절차이다. 또 이 일련의 과정들은 전송하려고 하는 데이터의 종류가 무엇이던지 혹은, 전송하는 데이터의 길이가 얼마이든지 간에 과정의 생략 없이 하나하나 진행되어야 한다.

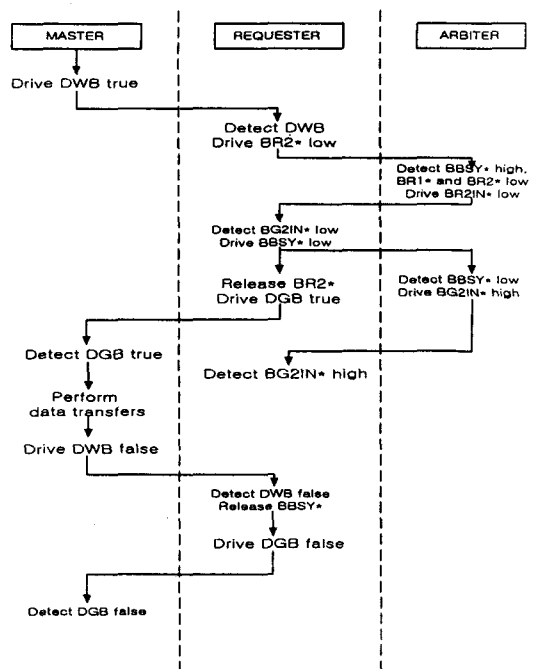


그림 5. 버스 중재 절차
Fig 5. Bus arbitration process

앞서 3장에서 밝힌 바와 같이 온라인 테스트를 하는 데 발생하는 어려움 중의 하나가 바로 버스를 통해서 전달되는 데이터인 기능적 벡터는 어떤 값을 가질지 예상할 수 없다는 것이다. 즉, 버스를 통해서 다른 보드로 전송되는 기능적 데이터는 시시각각으로 변화하고, 또 일정한 규칙이 있는 것이 아니기 때문에, 어떤 벡터가 버스로 전송되는지를 알기는 불가능하다. 그러나, 그림 5에서 볼 수 있는 것과 같이 버스 중재에 관여하는 신호선 들은 그렇지가 않다. 버스를 통해서 전달하려고 하는 데이터에 상관없이 이 일련의 이벤트들은 진행되어야 하는 것이므로 이 버스 중재 신호들은 모니터링을 통해서 발생하는 이벤트에 따라 다음 상태의 값을 예측할 수가 있다. 오프라인 테스트에서는 인가한 테스트 벡터에 해당하는 응답 벡터와 고장의 없을 경우에 예상되는 벡터를 비교해서 이상의 유무를 판단하지만, 온라인 모니터링은 발생하는 이벤트에 따라 예상되는 다음 상태의 값과 실제로 발생하는 값을 비교하여 고장의 유무를 판단한다.

4.3 버스 중재 이벤트 모델링

본 논문의 버스 중재 모니터링은 버스 중재 절차동안의 신호 변화 관찰을 통해서 실제로 발생하는 신호들의 조합과 고장이 없을 경우에 예상되는 버스 중재 신호들의 조합의 비교를 통해서 이루어진다. 그러므로, 고장이 없을 경우의 버스 중재 절차 시에 발생 할 수 있는 버스 중재 신호 조합을 먼저 살펴본다. 그림 6은 버스 중재 절차에서 발생할 수 있는 두 가지의 경우에 대해서 버스 중재 신호들의 조합을 나타내었다.

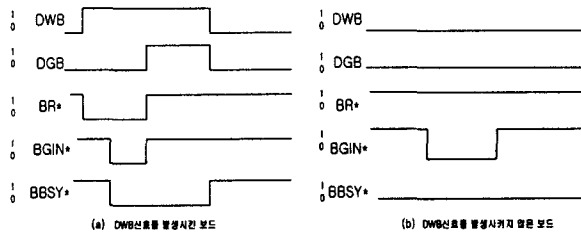


그림 6. 버스 중재 과정시 고장이 없을 경우 예상되는 신호 변화

Fig 6. Expected bus arbitration signals in fault free cases

동일한 BR* 선에서 보드가 갖을 수 있는 동작은 (a)의 경우가 나타내는 것과 같이 보드가 DWB 신호를 발생시킨 경우이거나, (b)의 경우가 나타내는 것과 같이 다른 보드가 BR* 신호를 발생시킨 경우에 BGIN*신호를 BGOUT*를 통해서 다른 보드로 통과시키는 동작이고, 각각의 경우에 대해서 버스 중재 신호들은 고장이 없을 경우에 그림 6과 같은 신호 변화를 가진다.

경계 스캔 구조가 갖는 커다란 단점 중의 하나는, 직렬 경로가 갖는 특성 때문에 테스트 벡터를 인가하고 또 그 응답을 관찰하는데 많은 수의 데이터 이동을 필요로 한다는 점이다. 특히, 시스템이 동작중인 온라인 상에서 그림 5와

표 2. 버스 중재 모니터링 대상 신호와 모니터링 대상 이벤트

Table 1. Bus arbitration monitoring signals and events

DWB	DGB	BR*	BGIN*	BCLR*	분류
0	0	0	0	0	발생하지 않는 신호조합
0	0	0	0	1	발생하지 않는 신호조합
0	0	0	1	0	발생하지 않는 신호조합
0	0	0	1	1	발생하지 않는 신호조합
0	0	1	0	0	이벤트3
0	0	1	0	1	이벤트4
0	0	1	1	0	이벤트3
0	0	1	1	1	정상 동작
0	1	0	0	0	발생하지 않는 신호조합
0	1	0	0	1	발생하지 않는 신호조합
0	1	0	1	0	발생하지 않는 신호조합
0	1	0	1	1	발생하지 않는 신호조합
0	1	1	0	0	발생하지 않는 신호조합
0	1	1	0	1	발생하지 않는 신호조합
0	1	1	1	0	이벤트3
0	1	1	1	1	이벤트2
1	0	0	0	0	이벤트3
1	0	0	0	1	정상 동작
1	0	0	1	0	이벤트3
1	0	0	1	1	정상 동작
1	0	1	0	0	발생하지 않는 신호조합
1	0	1	0	1	발생하지 않는 신호조합
1	0	1	1	0	이벤트3
1	0	1	1	1	정상 동작
1	1	0	0	0	발생하지 않는 신호조합
1	1	0	0	1	발생하지 않는 신호조합
1	1	0	1	0	발생하지 않는 신호조합
1	1	0	1	1	발생하지 않는 신호조합
1	1	1	0	0	발생하지 않는 신호조합
1	1	1	0	1	발생하지 않는 신호조합
1	1	1	1	0	이벤트3
1	1	1	1	1	이벤트1

같은 버스 중재 신호의 변화는 경계 스캔 구조를 사용해서 그 값을 관찰하는데 필요한 데이터의 이동시간 보다 짧은 주기로 발생하므로 변화하는 모든 신호 조합을 모니터링 한다는 것은 불가능하다. 그러므로, 버스 중재 과정에서 발생하는 일련의 신호 변화 중에서 시스템의 운용에 중요한 이벤트를 모니터링 대상 이벤트로 설정하여야 한다. 보드가 버스를 사용해서 데이터를 전송하는 행동을 다음의 4가지로 모델링하고 모니터링 대상 이벤트로 설정한다.

- 이벤트1 : 버스를 사용해서 데이터의 전송을 시작하려고 하는 이벤트.
- 이벤트2 : 버스를 사용해서 데이터의 전송을 끝마치는 이벤트.
- 이벤트3 : 버스를 사용해서 데이터를 전송 중에 BCLR* 신호를 중재기로부터 받은 이벤트.
- 이벤트4 : 버스의 사용 요청이 없는 보드가 중재기로부터 버스 사용 허용 신호(BGIN*)를 받았을 때, 이 신호를 BGOUT*로 통과시키는 이벤트.

이제, 모니터링 대상신호 DWB, DGB, BR*, BGIN*, BCLR*의 5개가 갖을 수 있는 신호조합 32개 각각에 대해서 표 1과 같은 분류를 할 수 있다. 여기서 발생하지 않는 신호조합은 버스 중재 동작에서 고장이 없을 경우에 갖을 수 없는 신호 조합을 의미한다. 예를 들어, DWB와 BR*가 동시에 0인 상황은 고장이 없을 경우에 절대로 발생할 수 없다. 왜냐하면, BR*는 DWB가 1인 것을 감지해서 0으로 드라이브되는 신호이기 때문이다. 그러므로 표 1에서 발생하지 않는 신호조합 또한 버스 중재 과정에서 실제로 발생된다면 고장의 상황이 된다. 그리고 분류에서 정상 동작으로 나타나는 신호 조합은 버스 중재 과정에서 발생할 수 있는 신호조합이나, 위에서 분류한 모니터링 대상 이벤트에는 속하지 않는 신호 조합들이다.

5. 경계 주사 구조와 버스 중재의 온라인 모니터링

5.1 온라인 버스 중재 모니터링 블록과 경계 주사 구조의 적용

온라인에서 시스템 버스 중재 과정의 고장을 모니터링 하기 위해서 테스트의 용이함과, 저렴한 테스트 비용을 달성하는 내장된 자체 테스트 회로와 보드 전체의 아키텍처를 그림 7에 나타내었다.

오른쪽 박스 안의 블록들은 버스 중재에 관계되는 블록들로 빅칩(big-chip) 내에 존재한다. 여기서 빅칩이란 각 보드 내에서 다른 모든 칩들이 버스로 데이터를 전송할 때 거치게 되는 칩을 말한다. 실제로 PCI를 비롯한 버스 구조들은 브리지(bridge)를 통해서 데이터를 버스로 내 보내는데, 이런 역할을 하는 칩을 빅칩이라 하며, 이 빅칩은 버스와 보드 내 다른 데이터 처리 칩들의 연결 역할을 담당한다. 그림에서 보는 바와 같이 빅칩에서 백플레인 버스로 나가는 모든 핀에는 경계 주사 셀들이 부착되어 있다.

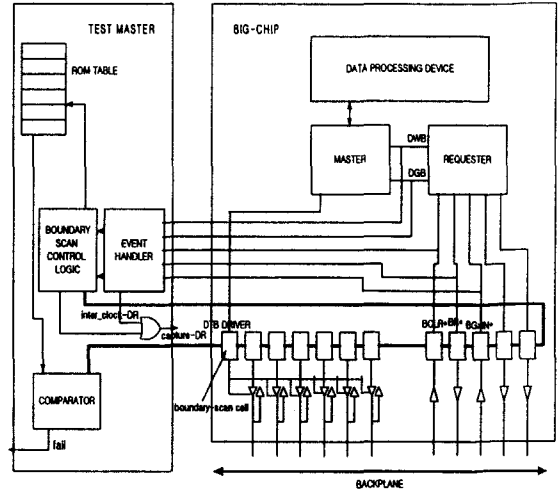


그림 7. 온라인 버스 중재 모니터링 구조
Fig 7. Architecture of on-line bus arbitration monitoring

그리고 오프라인 상호 연결선 테스트와는 달리 경계 주사 사슬이 이 셀들에 대해서만 구성되어 있다. 이것은 온·오프 테스트 모드에 따른 간단한 멀티플렉싱 기법으로 이루어질 수 있다. 이런 사슬 구성의 장점은 버스 중재에 관여하는 신호는 모두 이 셀들에서 캡처가 가능하기 때문에 경계 주사 사슬에 부착된 셀들의 수를 감소시킴으로서 직렬 이동 데이터의 수가 감소한다. 온라인의 상황에서는 모니터링 대상의 이벤트가 발생함에 따라 다음 이벤트가 발생하기 전까지 캡처된 신호를 이동해야 하므로 경계 주사 사슬의 직렬 이동 데이터의 수를 감소시키는 것은 아주 중요한 문제이다. 그림 7의 각 블록 기능은 다음과 같다.

■ 이벤트 핸들러(event handler)

표 1의 버스 중재 모니터링 대상 신호선들을 입력으로 하여, 모니터링 대상 이벤트에 해당하는 신호 조합이 발생했을 때, 이벤트 발생을 알리는 신호를 이벤트의 종류에 따라서 001, 010, 011, 100의 코드화된 신호로 발생한다. 이 때 이벤트 핸들러는 모니터링 대상 신호에 대해서 이벤트의 검출뿐만 아니라, 표 1에 나타난 버스 중재 절차에서 발생하지 않는 신호의 조합이 발생하는지도 모니터링을 한다. IEEE 표준 P1149.1 경계 주사 구조의 TAP 제어기는 16개의 상태를 갖는 FSM으로서 특정한 상태로 천이를 위해서는 적당히 TCK와 TMS를 인가해야 한다. 온라인 버스 중재 모니터링을 수행할 때 중요한 사항은 모니터링 대상 이벤트 신호 조합이 검출된 후 바로 다음 시스템 클럭에서 버스의 값을 경계 주사 셀로 캡처 해야 한다는 점이다. 즉 이벤트가 발생한 후 다음 클럭에서의 버스값이 테스트 벡터로서 의미가 있는 것이다. 빅칩의 경계 주사 셀에서 버스의 값을 캡처하는 동작은 제어 신호 clock-DR에 의해서 이루어진다. 이벤트 핸들러는 이벤트 발생을 감지함과 동시에 inter_clock-DR신호를 발생시켜 clock-DR신호를 활성화함으로써 이벤트 발생 다음 클럭의 버스 값이 경계 주사 셀에 캡처될 수 있도록 한다. 그림 7은 IEEE 표준 1149.1 기능을

지원하기 위해서, 경계 주사 컨트롤 로직에 존재하는 TAP 제어기의 출력 신호와 이벤트 핸들러의 inter_clock-DR을 논리합(OR)함으로써 clock-DR신호를 발생시키는 것을 의미한다.

■ 롬 테이블(ROM table)

버스 중재시 발생하는 각 이벤트에 따라서 고장이 없을 경우에 예상되는 값들을 저장하고 있는 블록이다. 모니터링 대상 이벤트에 따라 고장이 없을 경우에 이벤트 발생 다음 시스템 클럭에서 버스 중재 신호선들의 예상되는 값은 그림 5의 관찰을 통해서 이루어지고 4가지의 이벤트에 대해서 표 2에 나타내었다. 여기서 DTB_driver는 그림 7에서 보듯이 버스 드라이버의 인에이블(enable)신호를 의미한다.

표 3. 고장이 없는 경우 예상되는 신호 상태
Fig 2. Expected bus arbitration signals without faults

	DWB	DGB	BGIN*	BGOUT*	BCLR*	DTB_driver
event1	1	1	0	1	1	1
event2	0	0	1	1	1	0
event3	0	0	1	1	1	0
event4	0	0	0	0	1	0

■ 경계 주사 컨트롤 로직(boundary-scan control logic)

이벤트 핸들러로부터 코드화된 이벤트 발생 입력을 받으면, 경계 주사 컨트롤 로직은, clock-DR에 의해서 경계 주사 셀에 캡처된 버스 중재 신호를 주사 이동하고 동시에 롬 테이블에서 이벤트 종류에 따라 고장이 없을 경우에 예상 가능하여 저장된 값을 불러오도록 하는 제어 신호를 발생한다.

■ 비교기(comparator)

빅칩의 경계 주사 셀에서 캡처된 값과 롬 테이블에서 읽어들인 값을 비교하여 고장의 유무를 리포트하는 역할을 한다.

그림 7의 구조는 보드간 버스를 사용해서 데이터를 전송하는 중간에 테스트 데이터를 삽입하는 방식이 아니라, IEEE P1149.1에서 나타낸 표준 경계 주사 셀의 캡처기능을 사용하여 전송 데이터를 샘플링(sampling)하는 방식이므로 타이밍에 민감한 버스 중재 동작에는 아무런 영향을 미치지 않는다. 또한 모니터링에 필요한 대부분의 기능블록이 IEEE P1149.1 표준의 것을 따르고 있으며, 이벤트 핸들러도 간단한 조합회로로 구성이 가능하기 때문에 하드웨어 오버헤드는 매우 작음을 예상할 수 있다.

5.2 시스템 버스 온라인 모니터링 절차

시스템은 고가의 장비이므로 다른 VLSI 테스트와는 다르게 유지·보수의 수단으로서 중요한 의미를 갖는다. 그림 8

은 시스템 테스트의 절차를 보여 준다.

시스템은 전원이 투입되어 정상적으로 동작을 수행하기 전에 오프라인 상에서 상호 연결선에 대한 테스트를 거친다. 경계 주사 구조를 사용한 보드에서 각 소자의 입·출력 핀에 존재하는 경계 주사 셀들은 상호 연결선에 대해서 테스트 벡터를 인가하고, 응답 벡터를 관찰하는데 필요한 단자역할을 수행한다. 상호 연결선에서 발생할 수 있는 고장에 대해서 고장 검출 알고리즘이 담긴 테스트 벡터를 경계 주사 구조의 직렬 사술을 통하여 이동시키고, 이를 상호 연결선에 인가한 후 다시 경계 주사 구조의 직렬 사술을 통하여 응답 벡터를 관찰한다. 오프라인 테스트는 온라인 테스트와는 달리 고장 검출 알고리즘이 담긴 테스트 벡터를 인가할 수 있으므로 대부분의 고장은 이 시점에서 검출 가능하다. 상호 연결선에서 발생할 수 있는 고장의 대부분인 단락(open)과 합선(short), 고착 고장(stuck-st fault)이 오프라인 테스트에서 검출이 가능하고, 또한 고장 위치의 분석이 가능하다. 그러므로 검출된 고장 종류와 고장 위치에 대해서 적당한 고장 수리를 통하여 시스템은 재사용 가능하다.

오프라인 테스트를 무사히 통과하고 정상운행 중인 시스템에서는 온라인 버스 모니터링을 실시한다. 오프라인 테스트 과정에서 이미 대부분의 고장은 검출이 되었고, 온라인에서는 잠재된 고장의 영향이 나타나는지를 모니터링 한다. 버스 신호의 모니터링을 통하여 발생하는 이벤트에 따라 버스 값을 캡처하고 미리 예상이 가능한 고장이 없을 경우의 버스 값과 비교해 고장의 유·무를 판단한다. 만일 온라인 버스 중재 모니터링에서 고장이 검출되면 오프라인으로 전환하여 다시 테스트를 하고, 마찬가지로 고장 수리의 과정을 거친다.

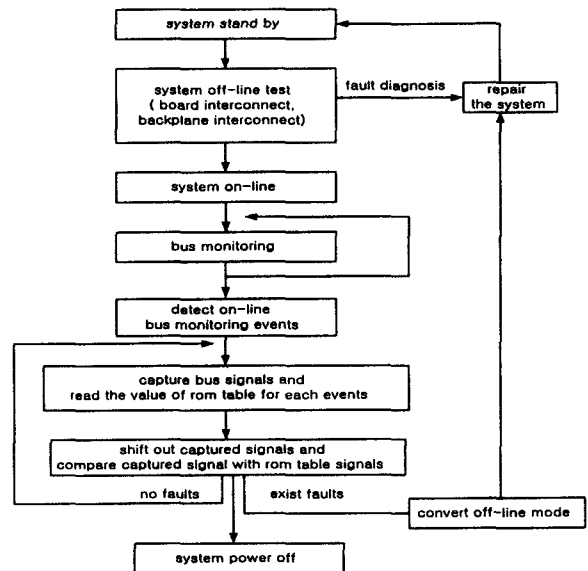


그림 8. 시스템 유지·보수 절차
Fig 8. Process of system maintenance and repair

6. 결론

시스템이 동작중인 온라인 상황에서 고장을 검출할 수 있는 테스트 구조는 시스템 유지·보수의 측면에서 매우 중요

하다. 특히, 시스템 버스를 다중의 보드가 공유하여 사용하는 동작에 있어서는 고장의 영향이 시스템 전체에 미치기 때문에 더욱 더 그러하다. IEEE P1149.1 경계 주사 구조와 확장은 오프라인에서 시스템 테스트를 훌륭하게 수행할 수 있다. 이를 발전시켜, 본 논문은 IEEE P1149.1의 구조를 사용하여 온라인 동작중인 시스템 버스의 고장 여부를 모니터링 하는 테스트 구조와 테스트 방법을 제시하였다. 이를 위하여 온라인 버스 중재 신호 변화의 관찰을 통해서 모니터링 대상이 되는 이벤트를 모델링하였고, 각 이벤트에 대하여 버스 중재 신호를 캡처하고 이 캡처된 값과 고장이 없을 경우의 예상된 신호와 비교함으로써 고장을 검출할 수 있는 내장된 자체 테스트 구조를 개발하였다. 본 논문에서 사용한 버스 중재 모델은 특정한 버스 구조에 한정되는 것이 아니라, 일반적으로 모든 버스 구조가 공통적으로 사용하고 있기 때문에 VME 버스나, PCI 버스 같은 특정한 버스 구조로 적용 될 수 있다.

참 고 문 헌

- [1] H. Bleeker, P. Eijinden and F. Jong, Boundary-Scan Test: A Practical Approach, Kluwer Academic Publishers, pp1-5, 1993.
- [2] K. P. Parker, The Boundary-Scan Handbook, Kluwer Academic Publisher, pp1-8, 1992.
- [3] C. Maunder, The board users guide to testable logic circuits, Addison-Wesley Publishing Company, pp4-23, 1992.
- [4] I. Ghosh, N. K. Jha, and S. Dey, "A low overhead design for testability and test generation technique for core-based systems", Proc. of International Test Conference, pp50-59, 1997.
- [5] 김현진, 신종철, 임용태, 강성호, "다중 주사 경로 회로 기판을 위한 내장된 자체 테스트 기법의 연구", 대한 전기공학회논문지, pp14-25, 1999.
- [6] C. Su, S. W. Jeng, and Y. T. Chen, "Boundary Scan BIST Methodology for Reconfigurable Systems", Proc. of International Test Conference, pp774-783, 1998.
- [7] A. Hassan, J. Rajski, and V. K. Agarwal, "Testing and Diagnosis of Interconnects using Boundary Scan Architecture", Proc. of International Test Conference, pp126-136, 1988.
- [8] C. Poirier, "IEEE P1149.5 to 1149.1 data and protocol conversion", Proc. of International Test Conference, pp527-535, 1993.
- [9] L. Whetsel, "A proposed method of accessing 1149.1 in a backplane environment", Proc. of International Test Conference, pp206-216, 1992
- [10] J. A. Floyd, "Real-time on-board bus testing", Proc. of International Test Conference, pp140-145, 1995.
- [11] K. P. Parker, The boundary-scan handbook, Kluwer Academic Publisher, pp11-52, 1993.

저 자 소 개



송 동 섭 (宋 東 燮)
1974년 7월 24일생. 2000년 건국대 전기 공학과 졸업. 2000년~현재 연세대 전기 전자공학과 석사과정
E-mail : dssong@cowboys.yonsei.ac.kr



배 상 민 (裵 相 民)
1973년 10월 15일생. 1998년 광운대 제어 계측공학과 졸업. 2000년~현재 연세대 전기전자공학과 석사과정
E-mail : warmice@cowboys.yonsei.ac.kr



강 성 호 (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(공학박). 미국 Schlumberger연구원. Motorola 선임연구원. 현재 연세대학교 공과대학 전기전자공학과 부교수.
Tel : 02-2123-2775, Fax : 02-313-8053
E-mail : shkang@yonsei.ac.kr



박 영 호 (朴 英 鎬)
1985년 대전산업대학교 전자계산학과 졸업 1983~현재 한국전자통신연구원 / 교 환전송기술연구소/ 시스템종합팀 / 선임 기술원 관심분야 : CAD,컴퓨터네트워크
E-mail : yhpark@etri.re.kr