C 언어를 이용한 PCI Express 동작 모델 설계 및 검증

예 상 영, *현 유 진, **성 광 수
(주) 다이나릿시스템, *영남대학교, **영남대학교
전화 : 019-575-4006 / 핸드폰 : 042-862-6411

Design And Verification Of A PCI Express Behavioral Model Using C Language

Sang-Young Ye, *Eugin Hyun, *Kwang-Su Seong
Dynalith Systems Company, Yeungnam University
E-mail : syye@dynalith.com

Abstract

Today's and tomorrow's processors and I/O devices are demanding much higher I/O bandwidth than PCI 2.3 or PCI-X can deliver and it is time to engineer a new generation of PCI to serve as a standard I/O bus for future generation platforms. According to this demand the PCI SIG proposed PCI Express. This paper describes about the design of PCI Express Behavioral Model. A Behavioral Model enables the designers to test whether the design specifications are met by performing computer simulations rather than experiments on the physical prototype. In the proposed Model, we can verify whether our design concept satisfies the PCI Express functional protocol.

I. 서론

지난 10여 년 간 PCI 버스는 고속 인터페이스로서의 제 역할을 해 왔다. 하지만, 현재와 미래에는 더 많은 대역폭을 가지는 프로세서와 I/O 장치가 필요하다. 이에 따라 표준 I/O 버스에 대해서도 새로운 플랫폼이 필연적으로 대두되었다.[1][2][3]

이에 대해 PCI 표준을 정하는 PCI SIG에서는 기존의 PCI를 계승하는 새로운 I/O 버스 표준으로 3GIO으로도 일컬어지는 PCI Express를 발표했다. 이 새로운 고속 I/O 인터페이스는 2.5GB/s(단방향)의 전송 속도를 가지며 기존의 PCI 장치에 사용된 운영체제와 드라이버 소프트웨어를 그대로 사용할 수 있다는 장점을 가지고 있다.[3][4]

오늘날 새로운 제품의 수명은 짧은 시장 주기에 의해 좌우된다. 이러한 경제 환경에 있어 제품을 개발하고 테스트하는데 있어 실질적 하드웨어 모델의 설계와 테스트는 고비용과 시간적 소모가 너무 크다.[5][6]

반면에 소프트웨어를 통한 가상 모델의 설계는 설계자로 하여금 실질적 하드웨어 모델 설계에 비해 훨씬 저렴하며 테스트 시간의 효과적인 단축이 가능하다. 그리고, 설계자는 즉각적인 피드백이 가능하며 이를 곧바로 디자인에 적용할 수 있다.[6]

본 논문에서는 앞에서 언급한 PCI Express 규격으로 동작하는 장치를 C언어를 이용해 동작 모델(Behavioral Model)을 설계하여 기능 시뮬레이션을 통해 시스템의 구조, 동작 원칙 등의 해석을 하고자 한다. 소프트웨어로 구현되는 동작 모델은 정확한 타이밍과 동작속도에 대한 검증은 어려우나 기능적 설계 규격이 기본 PCI Express 규격을 잘 만족하는지에 대해 검증하기가 편리하다. 또한 피드백 된 문제점에 대한 수정이 용이하다.[7]

본 논문은 서론에 이어 PCI Express에 대한 소개와 이를 C로 구현한 모델에 대해 다룬다. 그리고, 해당 모델을 통한 실험을 통해 결론에 이른다.

II. PCI EXPRESS에 대한 소개

그림 1은 기본적인 PCI Express 링크를 표현한 것이다. 링크(Link)란 두 구성요소(Component) 사이의 dual- simplex 통신 채널을 의미한다. 기본 PCI Express 링크는 두 개의 쌍(transmit pair와 receive pair), 저전압(low-voltage), 차동 전압 신호 쌍(differentially driven signal pair)으로 구성되어진다.

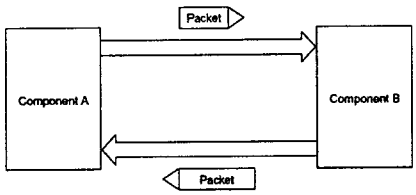

그림. 1 PCI Express Link

다음으로 PCI Express의 Layer들에 대해서 살펴보자. 그림 2는 각 Layer들을 도형화 한 것이다.

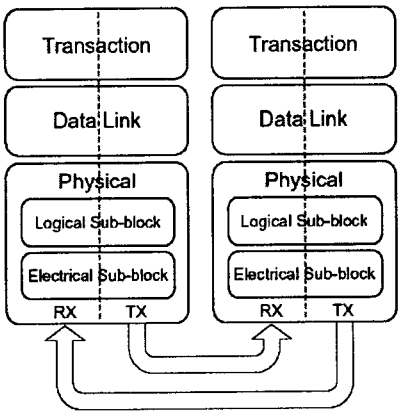

그림. 2 High-Level Layering Diagram

PCI Express는 각 구성요소간의 정보를 전송하기 위해 패킷을 사용한다. 패킷들은 각 Layer에 따라 형태가 변한다. 그림 3은 이러한 변화를 표현한 것이다.

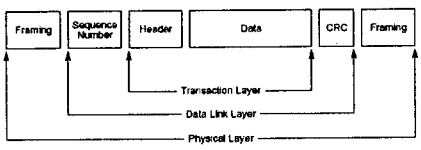

그림. 3 Packet Flow Through the Layers

먼저 최상위 Layer로 정의된 Transaction Layer에서는 데이터와 메시지를 Transaction Layer Packets(TLPs)을 생성한다. 이는 상대 구성요소에게 전달되기 전에 Data 링크 Layer로 전송된다.

중간에 위치한 Data 링크 Layer의 경우 상위 Layer(Transaction Layer)에서 전달한 TLP를 상대 장치에게 제대로 전달하는 일을 담당한다. 이 과정에서 TLPs에 일련의 번호를 붙이고 데이터 보호 코드를 첨가한다. 뿐만 아니라 이 Layer에서는 링크를 관리하는 일도 담당한다.

마지막으로 언급할 Layer은 Physical Layer이다. 이 Layer에서는 parallel-to-serial과 serial-to-parallel conversion, PLL(s), 그리고 임피던스 매칭 회로와 같은 모든 인터페이스 관련 회로부분을 포함한다. 이는 Data 링크 Layer에서 전송된 모든 패킷들을 시리얼 통신으로 상대 구성요소에 전달하는 역할을 하며 이 과정에서 발생한 에러를 Data 링크 Layer에 보고한다.

III. 소프트웨어 모델 연동 신호

본 논문에서는 그림 4와 같은 구조로 PCI Express 구성요소의 동작 특성을 나타내는 소프트웨어 모델을 C언어를 이용하여 구현하였다.

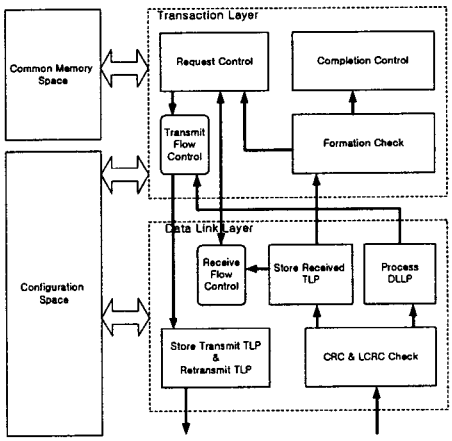

그림. 4 PCI Express Interface Controller Block Diagram

각 구성요소와 각 Layer들 모두가 독립적으로 동작해야 하므로 이를 구현하기 위해 프로세스를 사용하였다. 이는 유닉스 시스템에서 각 프로세스가 독립적으로 동작하는 점을 이용하였다. 이러한 각 프로세스간의 통신을 위해서는 공유 메모리와 FIFO가 이용되었다. 공유메모리의 경우 하나의 구성요소 내에서 여러 Layer에서 사용되어질 수 있는 파라미터들에 대해서

서로 다른 Layer에서의 접근 및 값 변환이 가능하게 한다. FIFO의 PCI Express의 기본 데이터 전송방식인 패킷을 전송하기 위해서 사용되었다. 패킷을 전송하는 과정에서 패킷의 종류와 시작과 끝을 알리는 신호를 FIFO를 통해 패킷과 함께 전송한다.[8]

PCI Express에서는 링크의 과부하를 줄이기 위해 Flow Control이라는 개념을 도입하였다. 이는 상대 구성요소의 여유 공간에 대해 정보를 주고받음으로써 불필요한 데이터 전송을 막는 것이다. 기본적으로 Transaction Layer에서는 상대 구성요소로부터 받은 Flow Control 정보를 가지고 현재 전송하고자 하는 TLP의 전송을 관장하게 된다.

Transaction Layer에서 전송하게 되는 TLP에는 기존의 PCI 버스에서 수행하였던 Memory Read/Write, I/O Read/Write, 그리고 Configu- ration Read/Write를 수행하는 것뿐만 아니라 Message도 포함된다. 두 구성요소를 연결하는 링크에는 추가적인 신호 선이 없기 때문에 특정 신호 선을 대신하는 메시지를 전송하게 된다. 이러한 기능을 담당하는 Transaction Layer에 대해 소프트웨어 모델이 구현된 형태를 그림 5와 그림 6의 순서 흐름도로 나타내었다.

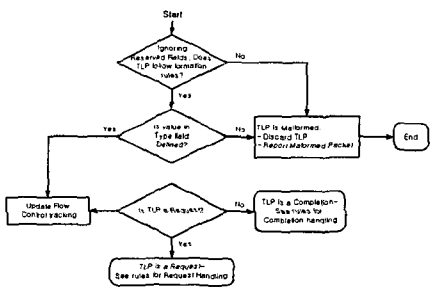

그림. 5 Flowchart for Handling of Received TLPs

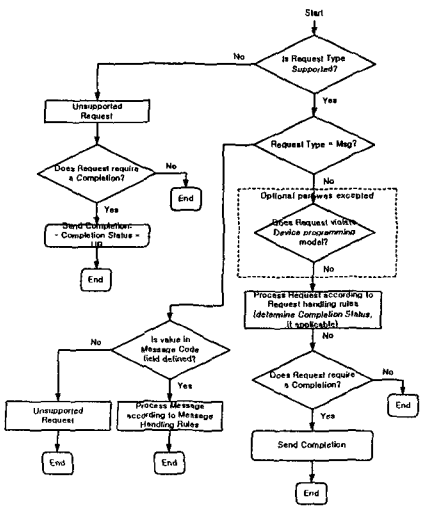

그림. 6 Flowchart for Handling of Received Request

Transaction Layer와 함께 구현된 Data 링크 Layer의 경우 Transaction Layer에서 전송된 TLP를 그대로 상대 구성요소에게 전달해야 한다. 이를 위해 각 TLP에는 일련 번호와 데이터 보호 코드(CRC)를 첨부하여 Physical Layer에 전송한다. 그리고, 문제가 발생한 TLP에 대해 재전송 및 에러보고 등을 담당하기도 한다. 그림 7은 전송된 TLP에 대해 Data 링크 Layer가 처리하는 과정을 나타낸 것이다.

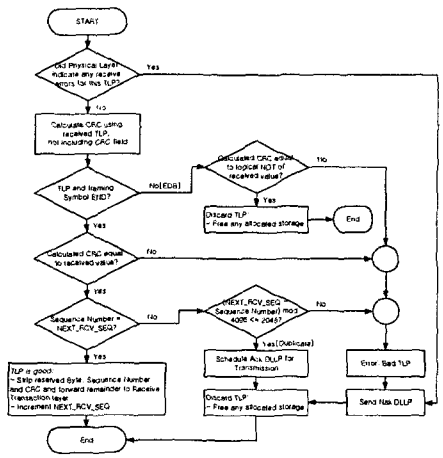

그림. 7 Receive Data Link Layer Handling of TLPs

또한, Data Link Layer는 TLP와 관련된 패킷과 자체적으로 Link를 관리하기 위한 패킷들을 전송한다. 이를 DLLP(Data Link Layer Packet)라 한다. 이러한 DLLP에 대해서는 그림 8과 그림 9에서와 같은 과정을 통해 처리하게 된다.

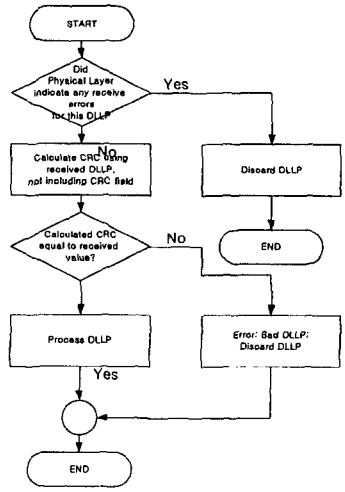

그림. 8 Received DLLP Error Check Flowchart

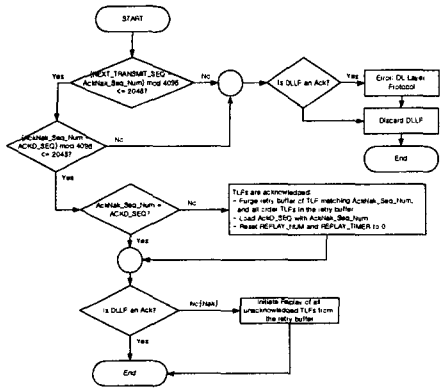

그림. 9 Ack/Nak DLLP Processing Flowchart

IV. 실험 및 고찰

설계된 인터페이스 컨트롤러에 대한 검증을 위해 기본적인 PCI Express Device 골격을 가지는 두 개의 구성요소(Component)를 만들었다. C 언어로 설계한 Legacy Endpoint (PCI Express Component)에 대해 Link가 처음 물리적으로 동작 가능한 상태를 만들어 Data Link Layer가 Flow Control을 거쳐 제대로 정상 동작 상태가 되는지를 검증하였다.

그리고, Endpoint의 정상동작 상태에서의 데이터 전송과 인위적인 오류를 통해 Endpoint가 에러를 제대로 처리하는지에 대해 테스트 하였다.

그림 10과 그림 11은 이에 대한 결과 화면이다.

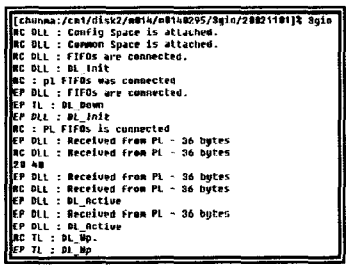

그림. 10 Data Link Layer Test Result

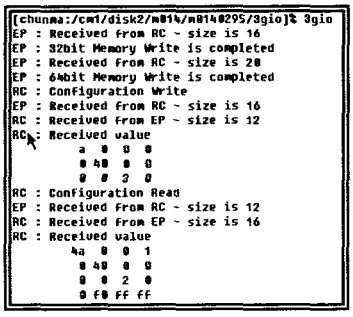

그림. 11 Transaction Layer Test Result

소프트웨어로 구현된 동작 모델의 경우 테스트하는 Endpoint의 Configuration Space 및 기본 메모리 영역에 대한 직접적인 데이터 접근이 가능하여 결과 검출이 용이하였다. 또한 기능적 동작 특성을 나타내도록 설계하였으므로 이에 대한 수정이 쉬웠다.

V. 결론

본 논문에서는 새로운 인터페이스 컨트롤러를 알고리즘 레벨에서 설계하고 이를 검증하였다. 소프트웨어를 이용한 설계 검증의 경우 전체적인 시뮬레이션 환경의 구축이 용이하다. 이는 복잡한 구조의 실질적인 하드웨어 설계에 앞서 소프트웨어를 통해 PCI Express 동작 모델을 설계하므로 최종적으로 설계하고자 하는 칩에 대해 프로토콜에 대한 검증을 할 수 있었다.

본 논문에서 프로토콜 알고리즘에 대해 검증된 소프트웨어 동작 모델은 실질적인 타이밍과 딜레이에 대한 정보를 제공할 수 없다. 그러므로, 본 연구에 이어 다음으로 HDL을 이용한 저 수준의 설계와 검증이 필요하다. 이 때 소프트웨어 동작 모델은 설계 및 검증에 있어 비교 모델로서의 역할을 할 것으로 보인다.

참고문헌

[1] PCI SIG, PCI Local Bus Specifications, Rev. 2.3, PCI SIG, March 2002.

[2] PCI SIG, PCI-X Addendum to the PCI Local Bus Specification, PCI SIG, July 2000.

[3] PCI SIG, 3GIO White Paper, PCI SIG, 2001.

[4] PCI SIG, PCI Express Base Specification, Revision 1.0, PCI SIC, April 2002.

[5] 최 훈, "구조적 C를 사용한 마이크로 프로세서의 설계", 한국과학기술원, 1995.

[6] Rajarishi Sinha, Christiaan J.J. Paredis, and Pradeep K. Khosla, "Behavioral Model Composition in Simulation-Based Design", Proc. IEEE Computer Society, 2002.

[7] 이기준, "IC 설계를 위한 논리와 회로 시뮬레이션", 전자공학회지, v.22, n.8, pp.98-110, 1995.

[8] 조유근 역, "UNIX 시스템 프로그래밍", 홍릉과학출판사, 1999.