직렬 링크 방식의 주변 장치 통합 인터페이스 설계

김도석* · 정훈주* · 이용환**

Design of General Peripheral Interface Using Serial Link

Do-Seok Kim* · Hoon-Ju Chung* · Yong-Hwan Lee**

요 약

최근 주변 장치의 성능은 사용자들이 요구하는 멀티미디어 데이터를 충족하기 위해 급속히 증가하고 있으며 고성능 장치에 실시간으로 데이터를 제공하기 위해 주변 장치의 인터페이스는 넓은 대역폭과 높은 전송 속도가 필요하게 되었다. PCI Express는 고속의 직렬 전송 인터페이스로 이전의 PCI와 PCI-X와 상호 호환이 되는 인터페이스이다. 본 논문에서는 직렬 링크 방식의 주변 장치 통합 인터페이스 설계하였다. TC/VC 매핑 기법과 VC 중재 기법을 사용해 우선순위에 의한 패킷 전송이 가능하도록 하였고, 4개의 레인을 사용하여 패킷을 전송하도록 하였다. Verilog HDL을 사용하여 인터페이스를 설계하였고 이를 Modelsim으로 검증하였다. FPGA 검증은 Xilinx ISE와 SPARTAN XC3S400을 사용하였으며 합성은 Synopsys Design Compiler를 사용하여 검증하였다.

Key Words : Interface, PCI Express, Flow Control, Virtual Channel, PCI-X

ABSTRACT

The performance of peripheral devices is improving rapidly to meet the needs of users for multimedia data. Therefore, the peripheral interface with wide bandwidth and high transmission rate becomes necessary to handle large amounts of data in real time for multiple high-performance devices. PCI Express is a fast serial interface with the use of packets that are compatible with previous PCI and PCI-X. In this paper, we design and verify general peripheral interface using serial link. It includes two kinds of traffic class (TC) labels which are mapped to virtual channels (VC). The design adopts TC/VC mapping and the scheme of arbitration by priority. The design uses a packet which can be transmitted through up to four transmission lanes. The design of general peripheral interface is described in Verilog HDL and verified using ModelSim. For FPGA verification, Xilinx ISE and SPARTAN XC3S400 are used.We used Synopsys Design Compiler as a synthesis tool and the used library was MagnaChip 0.35um technology.

* 금오공과대학교 전자공학부
** 교신저자: 금오공과대학교 전자공학과 교수 (yhlee@kumoh.ac.kr)
접수일자 : 2011년 01월 05일, 수정일자 : 2011년 01월 25일, 심사완료일자 : 2011년 02월 03일

Ⅰ. 서 론

현재 컴퓨터의 CPU 처리 속도 및 메모리의 성능은 급속히 향상되고 있으며 더욱 많은 장치의 연결이 필요하게 되었다. 또한 다양한 주변 장치의 성능이 인터넷과 네트워크 사용량의 증가 및 전송기술의 발전으로 인해 급속히 향상되고 있다. 이러한 기술 향상으로 인하여 고화질 영상 등의 대용량 데이터에 대한 실시간 처리가 필요하게 되었다. 즉 다양하고 고성능화 되어가는 장치로 인해 데이터를 실시간 처리하며 이러한 장치를 통합 연결하여 사용할 수 있는 넓은 대역폭을 가지는 인터페이스가 필요하게 되었다. PCI-SIG(PCI Special Interest Group)[1]는 컴퓨터의 주변 장치와 CPU가 직접 연결되어 고속으로 데이터를 전송할 수 있는 로컬 버스 표준 규격인 PCI를 정의하였고 90년대 PC의 표준 버스로 널리 사용되었다. 그러나 PC의 CPU와 여러 주변장치 및 네트워크 성능이 급속히 향상되면서 PCI에는 여러 문제점이 발생하게 된다. 각 장치의 동작 속도 증가로 인하여 병렬 버스 구조를 가지는 PCI 구조에서는 여러 장치를 연결하기 위한 확장을 위해 많은 선의 증가가 필요하였으며 특히 병목 현상을 가져와 시스템 성능을 저하시키게 되었다[2]. 이로 인해 PCI-SIG에서는 1998년 PCI의 확장된 규격인 PCI-X 버스를 제안하게 되었다. 이는 기존의 PCI에서의 동작 클록을 향상시킨 것으로 데이터의 전송 속도 및 대역폭을 증가 시켰으나 여전히 병렬 버스 구조를 가지며 시스템에 추가할 수 있는 장치의 수가 제한되어 있었다. 또한, 확장 시 선의 수가 많아져 EMI 잡음 문제 또한 여전히 발생 되었다[3]. 이러한 PCI 버스의 문제점을 극복하기 위해 PCI-SIG에서는 기존 PCI의 하드웨어 및 소프트웨어에 대한 호환성을 유지하면서도 높은 성능을 갖는 PCI Express를 제안하게 되었다.

PCI Express는 패킷(Packet)을 사용한 통신 및 컴퓨터 등의 입출력 장치 표준 규격으로 점대 점(Point-to-Point) 방식을 사용한 직렬 전송 프로토콜이다. 하나의 전송 선로(Lane)는 full-duplex로 방식으로 양방향 전송이 가능하도록 되어 있으며 하나의 링크를 다중 선로로 구성할 경우 x2, x4, x8, x12, x16, x32로 선로 개수의 가변이 가능하게 함으로써 선형적으로 전송속도 및 대역폭 확장이 가능하다. 따라서 다양한 주변기기의 속도에 맞추어 쉽게 대응 가능하다. 또한 PCI Express의 버전 역시 높여 현재 향상되는 주변장치의 성능에 대응하여 발전하고 있다.

본 논문에서는 설계되는 직렬 링크 방식의 장치 통합 인터페이스는 PCI Express의 계층 구조를 기반으로 하며 직렬 전송이 가능하도록 구현한다. 또한 4개의 전송 레인으로 구성하여 보드가 작은 디바이스에서도 효과적으로 장착할 수 있도록 하였다. 이로써 그림 1과 같이 주변장치와 통합이 가능하도록 하며 패킷 기반의 전송 형태를 가져 데이터를 효과적으로 전송할 수 있다.

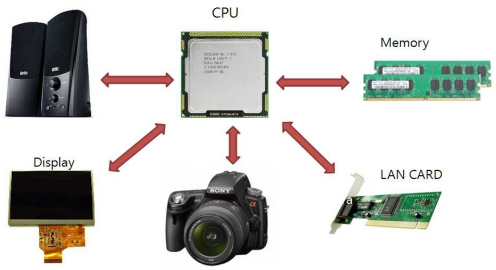

그림 1. 직렬 링크 방식 인터페이스의 예
Fig 1. Example of Peripheral Interface Using Serial Link.

Ⅱ. 인터페이스 구조

주변 장치 통합 인터페이스는 패킷을 사용한 통신 및 컴퓨터의 입력, 출력 장치에서 표준규격으로 현재 널리 사용되고 있다. 하나의 전송 선로는 송신과 수신이 가능하며 점 대 점 방식을 가지는 직렬 인터페이스이다. 또한 최대 32개까지의 선로 확장이 가능하여 선형적인 전송속도의 증가 및 대역폭 확장이 가능하다. 그림 2는 주변장치 통합 인터페이스[4]의 계층 구조이다.

Transaction Layer에서는 상위 계층에서 전송된 유효한 요청 정보를 가지는 헤더 및 전송 데이터 그리고 ECRC로 구성된 TLP(Transaction

Layer packet)를 생성하여 하위 계층인 Data Link Layer로 전송한다. Data Link Layer에서는 Transaction Layer에서 전송된 TLP에 전송 패킷의 신뢰성을 위한 Sequence Number와 LCRC(Data Link Layer CRC Code)를 붙여 Physical Layer로 전송한다. 또한 DLLP(Data Link Layer Packet)[5]의 생성 및 관리를 수행한다. Physical Layer는 최하위 계층으로서 직렬-병렬 변환, 병렬-직렬변환과 같은 물리적 인터페이스와 관련된 작업과 8b/10b 인코딩 및 10b/8b 디코딩 및 이를 실시하기 위한 패킷의 분할 같은 논리적 작업을 담당한다. 이때 Data Link Layer에서 전송 받은 TLP 및 DLLP에는 시작(Start)과 끝(End)을 나타내는 프레임(Frame) 정보를 붙여 전송한다. 그림 3은 각 계층의 세부 구조를 타나낸다. 각각의 계층은 수신부와 송신부로 구분되어 양방향 통신이 가능하도록 하였다.

1. Transaction Layer

Transaction Layer는 디바이스의 소프트웨어 계층과 Data Link Layer사이에 위치한다. 송신부는 헤더와 데이터 및 ECRC를 생성하여 TLP를 구성한다. 이때 구성된 TLP는 TC(Traffic Class)와 VC(Virtual Channel)[6] 맵핑을 통해 VC버퍼에 저장한다. 이후 수신단에서 해당 VC버퍼에 저장된 패킷을 받을 수 있는 상황이 되면 TLP를 전송한다.

수신부는 Data Link Layer에서 전송된 TLP를 TC와 VC 맵핑을 통해 VC 버퍼 저장 위치를 결정한 후 해당 버퍼에 저장한다. 이후 VC 우선순위에 따라 선택되어진 버퍼의 저장 데이터를 전송한다. 이때 데이터는 데이터 버퍼의 내용이 그대로 전송이 이루어지나 헤더버퍼에서 출력되는 정보는 상위계층으로 전송될 유효한 컨트롤 정보를 선택하여 전송한다. 또한 현재 전송중인 데이터에 대한 오류 유무를 판단하기 위해 CRC를 생성하여 수신된 TLP의 ECRC와 비교하여 결과를 소프트웨어 계층으로 전송한다.

2. Data Link Layer

Data Link Layer의 송신단은 Transaction Layer에서 전송된 패킷에 신뢰성을 위한 Sequence Number와 LCRC를 TLP에 붙여 Link 패킷을 생성하고 재전송(Replay) 버퍼에 저장과 함께 Physical Layer로 전송한다. 수신단의 Data Link Layer에서는 Physical Layer에서 전송된 패킷에 포함된 LCRC 체크를 통해 전송 패킷의 오류를 확인한다. 이와 함께 또한 Sequence Number 또한 확인하여 현재 전송된 패킷의 오류를 판단한다. Sequence Number에 대한 확인은 현재 전송중인 요청이 순차적으로 이루어지고 있는 지에 대한 체크를 담당한다. 이후 전송된 패킷의 이상이 없을 경우 해당 패킷의 Sequence Number와 LCRC 필드를 제거한 후 Transaction Layer로 전송한다.

3. Physical Layer

Physical Layer의 송신단에서는 전송될 TLP 또는 DLLP에 1 바이트의 시작 및 끝 프레임을 붙여 전송한다. 이후 바이트 striping을 실시하여 각각의 8b/10b 인코더를 통해 10비트로 데이터 인코딩을 실시한다. 생성된 10비트 데이터는 병렬-직렬 컨버터를 통해 직렬로 레인으로 전송한다. 수신단에서 각 레인을 통해 수신한 데이터는 직렬-병렬 컨버터를 통해 변환되고 디코더를 통해 10비트에서 8비트 데이터로 디코딩 한다. 이후 un-striping을 실시하여 패킷을 복원하고 시작과 끝 정보를 제거 후 Data Link Layer로 패킷을 전송한다.

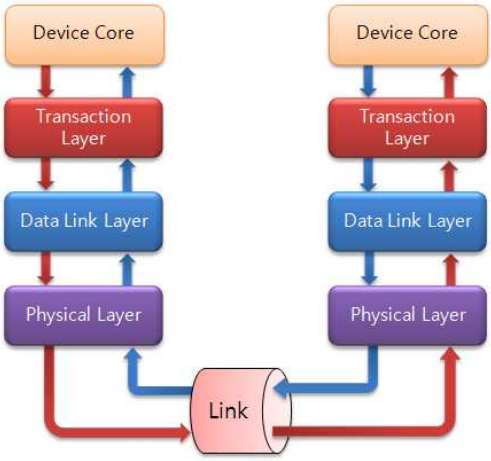

그림 2. 인터페이스 계층 구조
Fig 2 Layer architecture of interface.

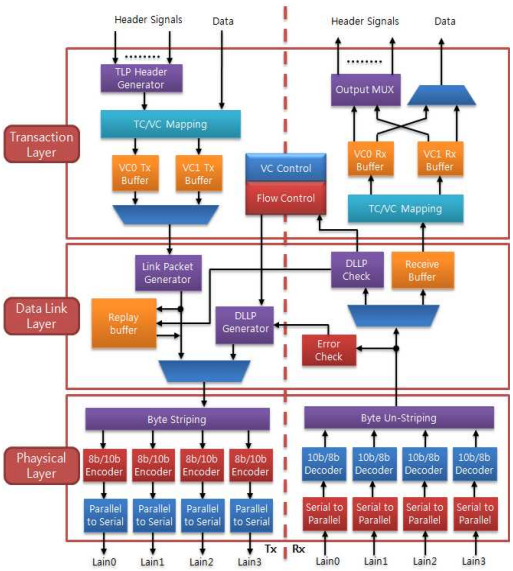

그림 3. 세부 계층 구조
Fig 3. Detailed layer architecture.

Ⅲ. 인터페이스 성능 보장 기법

1. Traffic Class와 Virtual Channel

하나의 전송선로로 구성 장치 통합 인터페이스는 다수의 디바이스 또는 단일 디바이스 인터페이스로 사용 가능하다. 이때 다수의 디바이스 또는 다양한 종류의 요청이 들어올 경우 이에 대한 전송 우선순위가 높을수록 우선적으로 요청을 실행한다. 이를 위해 사용되는 것이 TC와 VC 맵핑 전송 기법이다. TC는 소프트웨어 계층에서 전송된 정보로 TC0에서 TC7까지 총 8개의 레벨중 하나가 부여되어 전송된다. VC는 하나의 전송선로로 구성된 직렬 전송 인터페이스에서 여러 가지 기능을 전송할 경우 이에 따른 각각의 전송 통로를 제공하는 것처럼 하기 위해 가상 채널 ID를 부여해 전송 선로를 사용한다. 또한 VC 우선순위를 통해 각각의 VC 레벨에 따른 전송 우선권 부여 및 전송 대역폭을 가변할 수 있다[7]. 그림 4에서는 2가지의 장치를 사용하기 위한 통합 인터페이스의 전송 방법을 나타낸다. 장치1과 장치2는 2개의 TC를 가지며 통합 인터페이스 또한 2개의 VC을 갖기 때문에 이를 각각 맵핑한다. 이후 TC와 VC 맵핑을 통해 각각의 VC 버퍼에 저장하고 이후 VC 우선순위에 따라 전송하는 방식을 사용하게 된다.

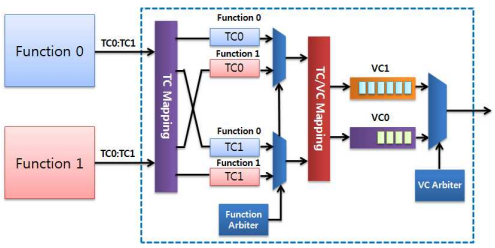

그림 4. 다중 장치의 TC와 VC 맵핑 기법
Fig 4. Mapping between TC and VC for multiple devices.

예를 들어, VC를 사용하지 않을 경우 1/20초당 15MB의 데이터 전송량을 요구하는 영상 데이터는 일반적 데이터인 이더넷1, 이더넷2와 함께 라운드로빈 방식에 의해 동일한 빈도의 전송이 이루어진다. 그로인해 영상 데이터는 원하는 전송폭에 미치지 못하여 디스플레이 출력 영상이 정상적이지 않을 수 있다. 반면, 그림 5와 같이 VC 기법을 사용하면 실시간 영상 데이터에 더 높은 레벨의 VC1를 맵핑하고 각 이더넷 데이터를 낮은 레벨의 VC0에 맵핑하여 VC의 전송 비율을

3:1로 전송한다. 따라서 1/20초 동안 영상 데이터를 15MB 이더넷 데이터를 5MB 전송하게 되며 이에 따라 영상 데이터가 요구하는 전송 속도를 만족하게 된다.

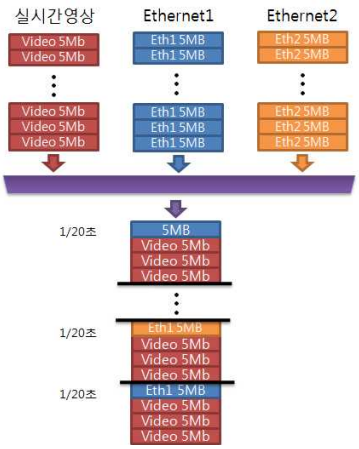

그림 5. VC를 사용한 우선순위에 의한 데이터 전송
Fig 5. Data transmission on priority using VC.

2. 플로우 컨트롤

플로우 컨트롤은 수신단에서 패킷을 수신할 수 있는 VC 버퍼의 여유 공간을 주기적으로 송신단에 보고한다. 이를 통해 송신단은 패킷의 전송 전에 수신단의 VC 버퍼에 충분한 저장 공간이 있는지 확인한 후 패킷을 전송한다. 따라서 수신단의 버퍼가 충분하지 않아 송신단에서 보내는 패킷이 버려지는 것을 방지하고 전송 대역폭의 낭비를 줄일 수 있다.

그림 6은 플로우 컨트롤 방법에 대한 흐름을 보여준다. 플로우 컨트롤은 송신단 디바이스의 Transaction Layer에서 FCC(Flow Control Credits)를 단위로 한 버퍼의 현재 여유 공간의 정보를 Data Link Layer에 전송하며, Data Link Layer에서는 이 정보를 통해 FC DLLP를 생성한다. 생성된 패킷은 송신단의 전송되어 현재 수신단의 버퍼 상태정보를 업데이트 한다. 이후 수신단에서 송신단으로의 패킷 전송이 발생할 경우 플로우 컨트롤을 사용하여 수신단의 VC 버퍼 상태 정보와 전송될 패킷의 전송량 정보에 대한 계산을 실시함으로써 현재 전송될 TLP가 저장될 수신단의 여유 버퍼가 있는지 확인한 후 패킷의 전송을 실시한다.

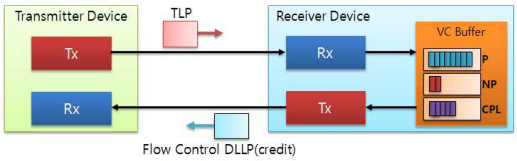

그림 6. 플로우 컨트롤 과정
Fig 6. Flow control process.

3. 재전송 기법

Data Link Layer에서 송신단에서 수신단으로 패킷의 전송 시 Link 패킷을 생성하여 전송한다. 이때 전송되는 패킷은 재전송 버퍼에도 저장된다. 이후 수신단에서 전송하였을 때 수신된 패킷에 대한 오류검사를 통해 오류가 발견되지 않을 경우 그에 대한 응답으로 ACK DLLP를 발신한다. 그렇지 않을 경우 Sequence Number가 포함된 NAK DLLP를 전송한다. 이때 수신단에서 NAK DLLP를 받을 경우 패킷의 재전송[8]을 실시하게 되며 이때 Sequence Number에 대응되는 어드레스의 재전송 버퍼 데이터를 전송한다. 그림 7은 재전송 버퍼의 구조를 나타낸다.

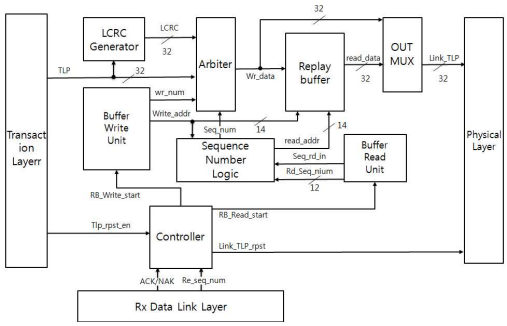

그림 7. 재전송 버퍼의 구조
Fig 7. Architecture of replay buffer.

Ⅳ. 구현 및 검증

주변 장치 통합 인터페이스의 동작을 검증하기 위하여 Verilog를 사용한 인터페이스 설계하였으며 이를 Modelsim으로 시뮬레이션하여 기능 검증을 하였다. 이후 FPGA 구현을 통해 하드웨어 검증을 실시하였다. 또한 Design Compiler를 통해 구현된 인터페이스의 면적 및 동작속도를 측정하였다.

시뮬레이션을 위해서는 2개의 동일한 인터페이스를 구성하여 양방향으로 연결하고 시뮬레이션하였다. 시뮬레이션을 위해 사용된 테스트 벤치는 일정하게 증가하는 데이터가 출력되도록 하였으며 TC0과 TC1의 값이 교대로 입력이 되도록 하였다. 수신단의 테스트 벤치는 최종 결과나 나오는 것을 확인하기 위해 구성하였다. 그림 8은 인터페이스 시뮬레이션 검증을 위한 연결 구성도를 나타낸다.

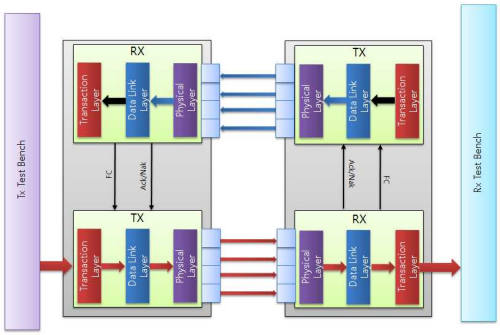

그림 8. 시뮬레이션 환경
Fig 8. Simulation environment.

1 시뮬레이션 검증

테스트벤치에 의해 TC0과 TC1의 요청이 교대로 발생하는 데이터 스트림을 입력으로 시뮬레이션을 수행하였다. 그림 9는 Tx Transaction Layer에서 진행되는 메모리 쓰기 요청 시 동작 과정의 시뮬레이션이다. 먼저 입력되는 데이터는 TC1의 값을 가지며 이후에 들어온 데이터는 TC0으로 설정되어 입력되었다. 입력된 TC와 VC 맵핑을 통해 해당 데이터가 저장될 VC 위치를 결정하였으며 메모리 쓰기 요청 이므로 메모리 헤더를 구성하여 헤더 버퍼에 저장되고 이와 함께 데이터를 데이터 버퍼에 저장한다. 데이터 저장이 완료된 후 생성된 ECRC를 헤더 버퍼에 저장한다.

Rx Transaction Layer는 Data Link Layer에서 전송된 TLP에서 Header와 데이터를 구분하여 VC 버퍼에 저장한다. 이후 VC 우선수위에 따라 수신 디바이스로 헤더 정보에 구성되어 있던 유효한 컨트롤 정보 및 데이터를 전송한다. 그림 10은 Rx Transaction Layer는 최종적으로 TLP가 도착하여 이를 디바이스로 출력하는 파형을 보여준다.

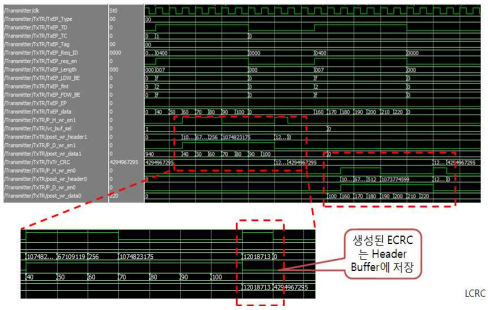

그림 9. Tx Transaction Layer 신호 파형
Fig 9. Signal waveform of Tx Transaction Layer.

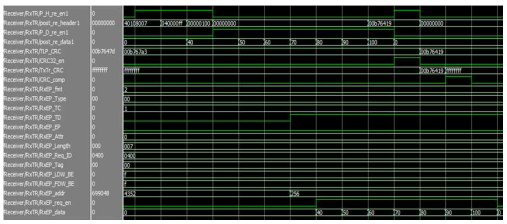

그림 10. Rx Transaction Layer 신호 파형
Fig 10. Signal waveform of Tx Transaction Layer.

2. FPGA 검증

FPGA를 이용한 검증은 Xilinx SPARTAN XC3S400을 사용하여 주변 장치 통합 인터페이스를 구현하였다. FPGA 합성 툴은 Xilinx ISE 11.1을 사용하여 합성하였다. 그림 11은 FPGA 검증 구조를 나타내며 표 1은 합성결과를 보여준다. 요청 발생기는 카운터로 구성되어 있으며 주기적으

로 데이터 전송을 요청한다. 이 요청을 받은 송신단의 패킷 인터페이스에서는 이를 패킷으로 변환하여 수신단으로 전송하며 수신단에서는 전송받은 패킷을 세그먼트 디코더로 전송하고 세그먼트에 출력한다. 한편 송신단에서도 이 데이터를 세그먼트에 표시하여 전송된 패킷과 동일한지를 검증한다. 그림 12는 FPGA 최종 결과에 따른 출력결과를 보여준다.

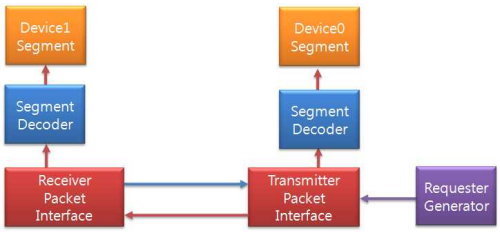

그림 11. FPGA 검증 환경
Fig 11. Co-validation environment for FPGA.

표 1. FPGA 합성 결과.
Table 1. FPGA Synthesis results

Device	Xilinx SPARTAN XC3S400
Total Logic elements	2081
Total memory bits	227,072(227,072/262,144 87%)
Total Pins	37(37/141 26%)
동작 속도	125.548 MHz

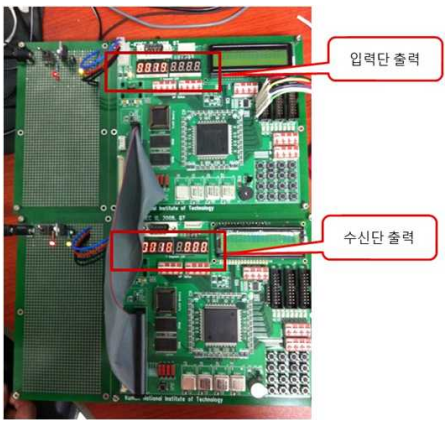

그림 12.FPGA 보드 구현
Fig 12. Implementation on FPGA Board.

표 2는 Synopsys Design Compiler 합성 결과를 나타낸다. 합성 시 사용된 공정은 MagnaChip 0.35um 표준 셀 라이브러리를 사용하였다. 이를 이용해 표 2와 같은 동작 속도 및 게이트를 구하였다.

표 2. Design compiler 합성 결과.
Table 2. Design compiler FPGA Synthesis results

Device		0.35um Standard Cell
합성 조건	Voltage	3.3 V
	Process	1
	Temperature	25 ℃
동작 속도		84 MHz
전송 속도		840 Mbps
셀 면적(게이트 수)		25,708

V. 결 론

본 논문에서는 주변장치 통합 인터페이스를 설계하였다. Transaction Layer, Data Link Layer, Physical Layer를 설계하여 OSI 상위 계층과의 높은 호환성을 제공한다. 또한 직렬 전송 방식의 패킷 전송 및 4개의 전송선로를 구성하여 고속의 데이터 전송 동작을 가능하도록 하였다. TC와 VC 맵핑 기법을 사용하여 전송 우선순위가 높은 패킷을 우선 전송시켜 실시간 전송이 필요한 패킷의 전송이 가능하도로 구성하였다. 또한 플로우 컨트롤을 사용해 수신단의 버퍼 정보를 실시간으로 확인하여 데이터를 전송함으로써 전송 선로의 효율성을 높였다. 전송 패킷의 오류 확인을 위한 Data Link Layer의 CRC 및 Transaction Layer의 CRC를 확인함으로써 현재 전송되는 패킷의 유효성을 확보하였으며 오류 발생이 있어도 송신단의 재전송 버퍼를 사용하여 패킷의 재전송을 구현하였다. 주변장치 통합 인터페이스는 Verilog HDL로 설계하였으며 Modelsim을 이용하여 시뮬레이션을 실행 하였다. SPARTAN XC3S400을 사용하여 FPGA 검증을 통한 하드웨어의 동작을 확인하였다. MagnaChip 0.35 um 공정을 이용하여 합

성한 결과 메모리를 제외하고 25,708개의 게이트와 84 MHz의 동작 주파수로 동작이 가능하며 전송 속도는 840 Mbps로 측정되었다.

후 기

본 논문은 금오공과대학교 학술연구비의 지원에 의한 연구결과입니다.

참 고 문 헌

[1] PCI-SIG Alliance, http://www.pcisig.com/
[2] 김영우, 박경, 김명준, “입출력 연결 규격의 기술 동향”, 정보통신연구진흥원, 주간기술 동향 1101호, 2003.
[3] Dhawan, S.K, "Introduction to PCI Express a new high speed serial data bus", IEEE Nuclear Science Symposium Conference Record, pp.687-691, 2006.
[4] Ravi Budruk, Don Anderson, Tom Shanley, PCI Express System Architecture, Addison-Wesley , 2003.
[5] PCI-SIG, PCI Express Base Specification Revision 2.0, 12, 20, 2006.
[6] A. Wilen, J. Schade and R. Thornburg Introduction to PCI Express, Intel Press, 2003.
[7] 변상봉, 문병인, 이용환, “모바일 기기를 위한 칩간 범용 인터페이스 구현”, 한국정보기술학회논문지, 제8권, 제11호, 2010.
[8] 장형식, 현유진, 성광수, “PCI-Express의 재전송 버퍼 관리 기법” 대한전자공학회, 2004.

저자약력

김 도 석 (Do-Seok Kim) 학생회원

2009년 금오공과대학교 전자공학부 학사
2009년-현재 금오공과대학교 전자공학과 석사과정

<관심분야> SoC 설계, 임베디드 시스템, 네트워크 인터페이스

정 훈 주 (Hoon-Ju Chung) 종신회원

1994년 경북대학교 공학사
1997년 한국과학기술원 공학석사
2002년 한국과학기술원 공학박사
2002-2004년 LG.Philips LCD 선임연구원
2004년-현재 금오공과대학교 전자공학부 부교수

<관심분야> 디스플레이, 인터페이스 설계

이 용 환 (Yong-Hwan Lee) 정회원

1993년 연세대학교 전자공학과 공학사
1999년 연세대학교 전자공학과 공학박사
1999-2002년 하이닉스반도체
2003-2004년 삼성전자
2004년-현재 금오공과대학교 전자공학부 부교수

<관심분야> SoC, 임베디드 시스템, 비전 마이크로프로세서 구조