논문 2010-47SD-1-12

효율적인 네트워크 사용을 위한 온 칩 네트워크 프로토콜

(On-chip-network Protocol for Efficient Network Utilization)

이 찬 호*

(Chanho Lee)

요 약

반도체 공정 및 설계 기술의 발전에 따라 SoC에 보다 많은 기능이 포함되고 데이터 전송량 또한 급격히 증가하고 있다. 이에 따라 SoC 내부의 온 칩 네트워크에서 데이터 전송 속도가 전체 시스템의 성능에 큰 영향을 미치게 되어 이와 관련된 연구가 활발하게 진행되고 있다. 기존의 AHB를 대체하기 위한 온 칩 네트워크 프로토콜로 AXI와 OCP가 대표적으로 거론되고 있으나 전송 성능을 증가시키기 위해 신호선의 수가 크게 증가하여 인터페이스와 네트워크 하드웨어 설계가 매우 어렵고 기존에 널리 사용되던 AHB와 다른 프로토콜과의 호환성도 좋지 않다. 본 논문에서는 이를 개선하기 위한 새로운 온 칩 네트워크 프로토콜을 제안한다. 제안된 프로토콜은 신호선의 수를 기존의 AHB보다 줄이고 AXI 등 다른 프로토콜과의 호환성도 고려하였다. 성능 분석결과 AXI보다는 조금 떨어지는 성능을 보여주고 있으나 8-버스트 이상의 전송에서는 큰 차이가 없고 신호선 수 대비 성능에서는 월등히 우수함을 확인하였다.

Abstract

A system-on-chip (SoC) includes more functions and requires rapidly increased data bandwidth as the development of semiconductor process technology and SoC design methodology. As a result, the data bandwidth of on-chip-networks in SoCs becomes a key factor of the system performance, and the research on the on-chip-network is performed actively. Either AXI or OCP is considered to a substitute of the AHB which has been the most popular on-chip-network. However, they have much increased number of signal wires, which make it difficult to design the interface logic and the network hardware. The compatibility of the protocols with other protocols is not so good. In this paper, we propose a new interface protocol for on-chip-networks to improve the problems mentioned above. The proposed protocol uses less number of signal wires than that of the AHB and considers the compatibility with other interface protocols such as the AXI. According the analysis results, the performance of the proposed protocol per wire is much better than that of the AXI although the absolute performance is slightly inferior.

Keywords : on-chip-bus, on-chip-network, interface protocol, SoC, AMBA, AHB, AXI

I. 서 론

반도체 공정 기술과 시스템 설계기술의 발달로 인하여 SoC(System-on-Chip) 기술은 지난 10 년간 급속도로 발전해왔다. 이러한 발전은 시스템을 구성하는 기술과 방법을 다양화시켜 SoC 내부구조를 더욱 복잡하게 만들었다. 또한, 고품질의 다양한 멀티미디어 데이터에 대한 요구로 인해 처리해야 할 데이터의 양은 기하급수적으로 늘어났다. 이에 따라 시스템의 성능이 연산기의 연산 능력보다는 데이터 통신 성능에 의해 더 큰 영향을 받게 되었고 이러한 문제를 해결하기 위해 다양한 온 칩 네트워크(on-chip-network) 구조와 프로토콜에 대한 연구가 진행되었다. AMBA AHB[1], AMBA AXI[2], WISHBONE[3], CoreConnect[4], OCP[5], SNP[6], XSNP[7] 등이 대표적인 인터페이스 프로토콜이다. 대표적인 온 칩 네트워크 구조는 Nostrum[8~9], Hermes[10], QNOC[11], aSoC[12], Octagon[13], AEthereal[14], SoCBus[15], SNA[16] 등이 있다.

* 정회원, 숭실대학교 정보통신전자공학부
(School of Electronic Engr., Soongsil University)
※ 본 논문은 교육과학기술부의 재원으로 한국연구재단의 기초연구사업(2009-0072664)의 지원을 받아 수행되었으며 IDEC의 SW 지원을 받았습니다.
접수일자: 2009년10월13일, 수정완료일: 2009년12월11일

복잡한 시스템을 위한 플랫폼 기반의 SoC 설계에서는 검증된 IP의 확보가 중요한데, 플랫폼의 성능 향상을 위해 새로운 인터페이스 프로토콜을 도입하는 경우 기존의 IP가 사용하는 프로토콜과 호환되지 않으면 IP 재설계가 불가피하므로 IP 재활용이 어렵다. 따라서, 처음부터 서로 다른 인터페이스 프로토콜 사이의 호환성을 고려하여 인터페이스를 설계한다면 이러한 문제를 해결할 수 있다. 기존 인터페이스 프로토콜은 상호간 호환성을 고려하고 있지 않기 때문에 서로 다른 프로토콜을 사용하는 다양한 IP(intellectual properties)를 하나의 시스템에 적용하는 작업은 쉽지 않다. 일반적으로 두 개의 서로 다른 프로토콜을 사용할 때 프로토콜 변환기를 사용하면 일부 문제를 해결할 수 있으나, 세 개 이상이 되면 각 프로토콜 신호가 의미하는 내용을 정확히 반영하기 어려워 호환성을 보장할 수 없다. 따라서 프로토콜 변환기를 사용한다 하더라도 세 개 이상의 서로 다른 프로토콜을 사용하는 시스템에서는 효율적인 통신이 불가능하다. 시스템 수준에서 프로토콜 호환성을 지원하는 것은 플랫폼 기반 설계에서 IP의 재활용이라는 면에서 중요한 요소이다.

한편, 기존의 인터페이스 프로토콜은 매스터와 슬레이브 인터페이스의 비대칭적 구조로 인해 하나의 IP가 두 가지 기능 모두를 가지기 위해서는 두 배의 신호선이 요구된다. 연산 과정에서 많은 양의 데이터를 요구하는 IP는 매스터(master) 와 슬레이브(slave) 인터페이스를 동시에 가지면 성능을 크게 향상시킬 수 있다. 이는 대칭적인 인터페이스를 이용하면 신호선을 늘리지 않고도 해결 가능하다. 또한, 데이터 전송시 다양한 기능을 제공하고 성능을 개선하기 위해 많은 제어 신호선을 포함한 프로토콜이 많은데, SoC 규모가 커지고 많은 기능 블록들이 하나의 시스템에 통합되면 많은 수의 신호선은 상당한 배선 혼잡(routing congestion)을 야기시킬 수 있다. 따라서, 부가 신호선의 수를 최소화하면서 원하는 효과를 얻을 수 있는 기술이 필요하다. 이러한 방식의 프로토콜은 SNP[6]와 XSNP[7]에서 이미 사용한 바가 있다.

SoC 시스템의 통신 패턴이 다양해지면서 온 칩 네트워크 상에서도 컴퓨터 네트워크와 같이 QoS(quality of service)를 지원하는 차등화된 서비스를 제공하고 있다[11~15]. 기존 구조에서는 대부분 온 칩 네트워크에서 QoS를 결정하며, 실제 통신을 수행하는 IP가 QoS를 선택할 수 있는 기회를 제공하지는 않는다. 이것은 기존의 온 칩 네트워크 구조가 특정 시스템 전용으로 사용되거나 다양한 시스템의 특성에 대한 고려 없이 구조적인 지원만 하기 때문이다. 또한, 기존 프로토콜은 QoS 지원에 대한 고려를 하지 않고 있기 때문에 IP 설계자는 QoS를 선택적으로 구현할 수 없다. MPSoC (Multi-Processor SoC)와 같이 병렬처리로 인한 다양한 통신 트래픽이 발생될 수 있는 구조에서 QoS를 IP 수준에서 결정할 수 있다면 보다 효율적인 통신을 수행할 수 있다.

오프 칩 통신은 온 칩 통신에 비해 신호선 수에 대한 제약이 많은 편이다. 인터페이스 신호 선의 수는 패키지의 입출력 핀 수 및 PCB의 배선 수와 직접적으로 연관이 있기 때문에 물리적 비용과 동작 주파수에 민감하다. 일반적으로 오프 칩 통신용 프로토콜은 PCI[17], PCI-X[18], PCIe[19]와 같은 규격을 따른다. 최근 다중 코어를 이용한 프로세서 설계가 활발해지면서 프로세서 간(processor-to-processor) 또는 칩셋 간 연결과 같은 오프 칩 통신에 다양하게 적용될 수 있는 상업용 기술들이 개발되고 있다[20~21]. 이러한 설계 추세는 온 칩 통신과 오프 칩 통신이 유기적으로 연결되면서 오프 칩 통신의 고성능화를 요구한다. 온 칩에서 사용되는 프로토콜과 오프 칩에서 사용되는 프로토콜이 서로 다를 경우 프로토콜 변환 과정은 추가적인 성능 저하를 유발할 수 있으므로 이를 최소화하는 것이 중요하다.

본 논문에서는 서로 다른 인터페이스 프로토콜간 호환성 지원, 인터페이스의 효율성, IP 수준의 QoS 지원, 온/오프 칩 통신 지원을 고려한 온 칩 네트워크 프로토콜을 제안한다. 제안하는 프로토콜은 기존의 고성능 온 칩 네트워크 프로토콜과 비슷한 성능을 제공하면서 효율성에 중점을 둔 시스템 설계를 가능하게 한다. 그 특성을 검증하기 위해 제안한 프로토콜 기반의 IP를 설계하여 시뮬레이션을 통해 결과를 분석하였다.

Ⅱ. 제안하는 프로토콜 구조

1. 인터페이스 신호

제안하는 프로토콜은 기본적으로 이종 프로토콜 간 호환성을 지원하며, 대칭적 구조로 인한 단순한 인터페이스를 갖는 점대점(point-to-point) 프로토콜이다. 그림 1에 나타난 것처럼 BABEL 프로토콜의 인터페이스는 valid(VLD), ready(RDY), command(CMD), channel (CHN)의 네 가지 신호로 구성되며, 핸드셰이킹

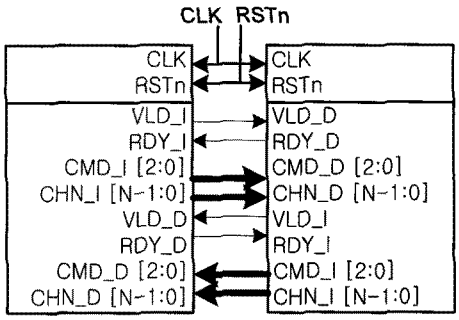

그림 1. 제안한 인터페이스 프로토콜을 위한 신호선
Fig. 1. Signals for the proposed interface protocol.

(handshaking) 통신을 한다. VLD는 현재 통신이 유효함을 의미하고 RDY는 유효한 통신이 정상적으로 수신될 수 있음을 의미한다. VLD와 RDY가 동시에 활성화되면 통신이 이루어진 것으로 간주한다. CHN은 데이터와 주소, 그리고 제어정보가 전달되는 물리 채널이다. CMD는 현재 CHN에 전달되는 정보의 의미와 특징을 나타낸다. 각각의 신호는 단방향 신호로 들어오는 신호와 나가는 신호의 한 쌍으로 구성되며 각 채널은 기본적으로 양방향 독립 통신(duplex)을 수행하는 것으로 정의된다. 제안한 프로토콜에서는 하나의 CHN을 데이터, 제어 정보(SBI: Side Band Information), 주소 정보 전달용으로 사용한다. 이는 PCI[17], SNP[6], XSNP[7] 등에서 사용되는 방식으로 실제 온 칩 통신에서 하나의 트랜잭션(transaction)내에서 주소 정보는 단순 증감하고, 제어 정보는 일정 기간 동안 유지되는 특성을 가지기 때문에 공유 사용이 가능하다.

제안한 프로토콜에서는 통신을 시작하는 인터페이스를 개시자(initiator)라 하고 이에 응답하는 인터페이스를 목적지(destination)라 하는데 AMBA와는 달리 서로 대칭구조이므로 IP의 물리적인 인터페이스만으로는 개시자와 목적지를 구분할 수 없고 실제 데이터 전송이 발생할 때 그 역할이 구분된다. 따라서 모든 인터페이스는 개시자와 목적지로서의 역할을 수행할 수 있다., 논리적인 기능면에서 개시자는 AMBA의 매스터 역할을 하고 목적지는 슬레이브 역할을 한다. AMBA의 경우 매스터와 슬레이브의 기능이 동시에 필요한 DMAC 등은 두 개의 인터페이스가 필요하나 제한한 프로토콜에서는 하나의 인터페이스로 개시자와 목적지 역할을 수행할 수 있어 인터페이스 수가 IP 기능에 따라 증가하지 않는다.

제안한 프로토콜에서는 64 비트 핸드셰이킹 통신을 기본으로 수행하여 138 개의 신호선을 요구한다. 기존

표 1. 제안한 프로토콜에서 사용되는 명령어
Table 1. Descriptions of commands in the proposed protocol.

Signal	Command	Description
000	IDLE	IDLE
001	AWT	Address and SBI for write transaction
010	ART	Address and SBI for read transaction
011	DATA	Data of transaction(s)
100	RSP	Response
101	Reserved	Reserved for future use
110	EXT	Extension Phase
111	LAST	Last Transfer

의 AMBA AHB가 64 비트 매스터와 슬레이브 인터페이스를 동시에 가지려면 416 개, AMBA AXI는 600 개의 신호선을 요구하므로 네트워크 효율성이 매우 뛰어남을 알 수 있다. 채널의 신호선은 필요에 따라 128 비트 이상으로 늘어나거나 32 비트 이하로 줄어들 수도 있다. 신호선이 128 비트로 늘어날 경우 두 개의 채널값이 한번에 전송되고, 줄어들 경우에는 하나의 채널값이 두 번 또는 그 이상으로 나누어 전송된다.

한편, 채널을 통해 주소와 데이터, 제어 정보가 전달되므로 채널에 실린 정보의 의미와 특성을 나타내기 위한 명령어가 CMD를 통해 함께 전달된다. 표 1에 나타난 바와 같이 명령어는 모두 7개로 구성된다. IDLE은 현재 통신을 하고 있지 않음을 의미하고 AWT와 ART는 각각 쓰기와 읽기 주소를 의미하는데 주소값과 함께 제어 정보 신호(SBI)가 함께 전달된다. DATA와 LAST는 모두 데이터를 의미하지만 LAST는 단일 또는 마지막 데이터를 나타내는데 사용된다. RESP는 응답을 위해 사용되며, 개시자의 통신이 목적지에 전달되면 목적지는 RESP 명령어를 이용하여 개시자에 응답한다. EXT는 다른 명령어를 확장하기 위한 목적으로 사용되는데 바로 앞의 명령어가 계속됨을 나타내며, 주로 데이터 폭이 넓은(wide) 영역(domain)에서 좁은(narrow) 영역으로 통신을 할 때 사용된다.

앞서 언급한 바와 같이 제어 정보 신호가 별도의 신호선을 갖지 않고 채널을 통해 전송된다. 이러한 제어 신호는 개시자로부터는 SBI를 통해 전달되고 목적지는 응답(RESP)을 통해 전송된다. 그림 2는 SBI와 RESP에 포함된 정보를 보여준다. 각 신호의 위의 숫자는 시작과 끝의 비트 위치를, 아래 숫자는 해당 신호의 비트 수를 각각 나타낸다. SBI의 신호중 Burst Length는 트랜

잭션의 버스트 크기를 의미하며, 최대 16개까지 가능하다. BurstType은 증가(increment)와 래핑(wrapping) 두 가지를 지원한다. BurstSize는 전송되는 데이터의 크기를 의미하며, 8 - 1,024 비트까지 가능하다. RW는 전송이 읽기 또는 쓰기임를 나타내고 LockType은 분리할 수 없는(atomic) 통신을 지원하기 위해 사용되며 AMBA AXI와 동일한 기능을 갖는다. ReqResp는 쓰기 동작의 경우 응답 신호의 유무를 지시하는 신호이며, ReqResp=1이면 목적지로부터 반드시 응답 신호를 받아야 하고 ReqResp=0이면 응답을 필요로 하지 않는다. TransactionID는 개시자가 내보내는 트랜잭션의 ID를 의미하고 outstanding address 방식이나 out-of-order completion 방식의 트랜잭션에서 이용된다. LinkID는 현재 개시자 IP가 연결된 포트의 ID(identification)를 의미한다. LinkID는 IP 동작과는 무관하며, 네트워크에 의해 결정되는 부분이다. 따라서 IP는 LinkID 부분은 비워둔 채로 통신을 진행한다. ProtocolID는 다수의 이종 프로토콜을 사용할 때 각각의 프로토콜에 ID를 부여하여 구분할 목적으로 사용된다. 최대 4 개의 이종 프로토콜을 동시에 지원할 수 있으며, IP가 아닌 프로토콜 변환기와 같은 유닛에서 사용하는 신호이다.

그림 2(b)는 RESP 명령을 사용할 때 응답 정보 포맷을 보여준다. SBI의 신호와 동일한 의미를 갖는 세 개의 ID 값과 목적지의 응답신호를 의미하는 Response, 그리고 읽기 또는 쓰기 전송에 대한 응답신호임을 나타내는 RW로 구성된다. 세 개의 ID 값의 위치를 SBI와 동일하게 하여 인터페이스 로직을 간단하게 설계할 수 있도록 하였다. Response는 2 비트 신호로 표 2에 나타

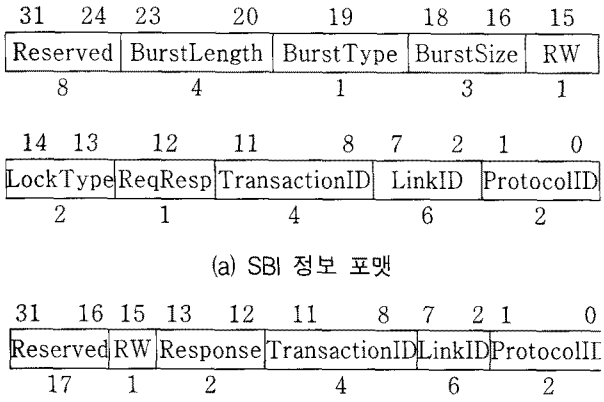

그림 2. SBI와 RESP 정보 포맷
Fig. 2. Information field formats of SBI and RESP.

표 2. 응답신호(Response)의 종류와 의미
Table 2. Description of Response signals.

Code	설명
000	Normal 통신을 정상적으로 수행. Exclusive 통신을 시도했을 경우 실패를 의미
001	Exclusive 통신 정상 수행
010	목적지 오류. 목적지 IP의 동작 오류
011	디코더 오류. 주소의 오류를 의미.

난 바와 같은 의미를 갖는다. RW 신호는 RSP 명령어 이후 추가로 읽어야할 데이터가 있는지 없는지를 알려주기 위해 필요하다.

2. 통신 방식

제안한 프로토콜에 의한 통신은 주소, 제어 정보, 데이터가 차례로 CHN 신호선을 통해 명령어와 함께 핸드셰이킹 방식으로 전달된다. 주소와 SBI는 CHN을 통해 동시 전송되는데 상위 32 비트에 주소가 위치하고 하위 32 비트에 SBI가 위치한다. SBI의 상위 9 비트가 사용되지 않는 영역이므로 필요시 주소를 확장하여 32 비트 이상의 주소 공간을 사용할 있다. 그림 3은 기본 통신의 예를 보여준다. 그림 3(a)는 주소와 SBI가 먼저 전달되고 4개의 데이터가 뒤따르는 4 버스트 쓰기 동작이다. 마지막 데이터는 LAST 명령어와 함께 전달되는데 LAST 명령어는 SBI의 버스트 길이만큼 데이터가 전달되지 않은 경우에도 나타날 수 있고 LAST 명령어가 나오면 전송이 종료되므로 조기 종료(early termination)가 가능하다. 목적지는 SBI의 Response 값이 1인 경우에만 응답 신호를 생성한다. 그림 3(b)는 4 버스트 읽기 동작을 보여주는데 Response 값에 관계없이 항상 응답을 발생시킨다.

그림 3(c)에서는 wide-to-narrow 통신의 예를 보여준다. LAST 명령을 제외한 모든 명령은 EXT 명령으로 확장될 수 있어 채널폭이 좁아진 만큼 EXT 명령으로 채널 데이터가 확장된다. 64 비트 통신이 기본이나 32 비트로 좁아지면 AWT와 ART가 32비트씩 두 번에 걸쳐 전송되는데 상위 32 비트가 먼저 AWT(ART) 명령어와 함께 전송되고 하위 32 비트는 EXT 명령어와 함께 전송된다. 목적지에서는 이를 64 비트로 모아 적절히 해석한다. 모든 데이터 통신의 끝은 LAST 명령을 사용하는데 마지막 64 비트의 경우 상위 32 비트 데이터는 DATA 명령어와 함께 전송되고 하위 32 비트만 LAST 명령어와 함께 전송된다. 16 비트 이하의 채널

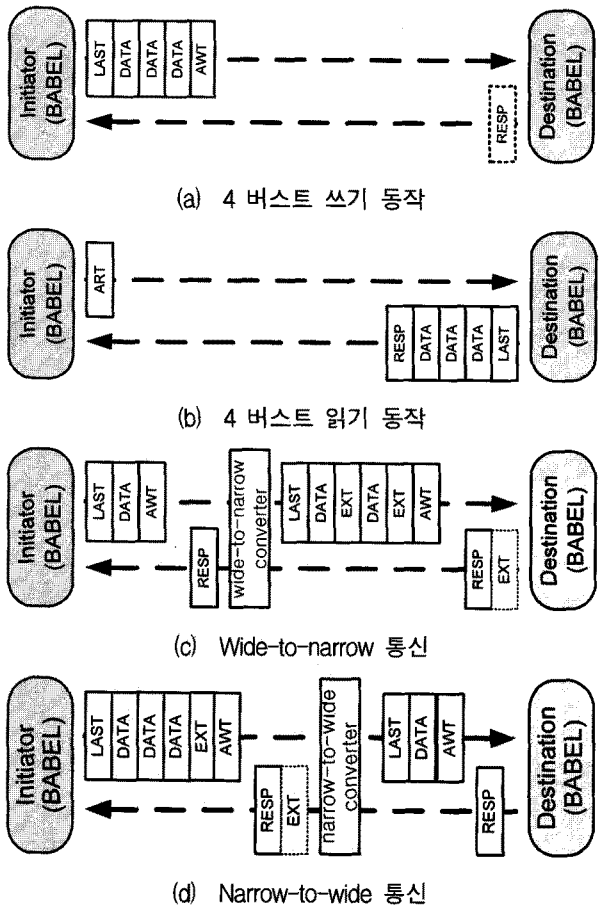

그림 3. 제안한 프로토콜에 의한 기본 통신
Fig. 3. Basic transactions in the proposed protocol.

폭을 갖는 경우에도 64 비트 중에서 첫 번째 전송부만 원래의 명령어가 따라가고 이후에는 EXT 명령어와 함께 한다. 목적지는 항상 64 비트 단위로 데이터를 처리한다. RESP 명령어의 경우에는 하위 14 비트만 이용하므로 16 비트까지는 EXT 명령으로 확장할 필요가 없다. 8 비트 이하의 채널을 사용할 경우에만 EXT 명령어를 사용한다. 그림 3(d)에서는 narrow-to-wide 통신의 예를 보여준다. 데이터 폭이 좁은 채널에서 데이터 폭이 넓은 채널로 연결될 경우 데이터를 하나의 채널로 결합할 수 있기 때문에 효율적인 물리적 채널 사용이 가능하다. 이때, 64 비트보다 좁은 채널에서 64 비트로 변경되는 경우에는 EXT 명령어를 제거하면 변환이 완료되지만 128 비트 이상으로 변환하는 경우에는 주소와 데이터를 하나로 모으지는 않는다. 이는 명령어에서 주소와 SBI, 데이터가 함께 섞여있는 경우를 나타낼 수 없기 때문이다.

그림 4에 AHB 기반 IP와 AXI 기반 IP가 제안한 프로토콜을 이용하여 통신하는 과정이 나타나 있다. 대부분의 온 칩 통신에서 쓰기 통신은 비교적 단순하게 처리할 수 있는 반면 읽기 통신은 회신을 기다려야 하기 때문에 프로토콜의 동작에 따라 상당한 성능 저하가 발생할 수 있다. 일반적으로 AHB에서는 4 개 이하의 (경우에 따라서는 16 버스트 이상) 버스트 전송은 미결정 길이 버스트(unspecified length burst read)로 처리하는데 이는 성능 저하 문제를 초래할 수 있다. AHB와 AXI간의 통신에서 AHB IP가 미결정 길이 버스트 읽기를 발생시키면 일반적으로 AXI bridge[22]에서는 AHB의 미결정 길이 버스트를 단일 전송(single transfer)으로 변환하여 처리하거나, 하위 프로토콜 호환성(backward protocol compatibility)을 위해 AHB-Lite[23]와 같이 축소된 규격을 사용한다. AHB 프로토콜 변환기는 이러한 미결정 길이 버스트 전송을 응용 시스템에 따라 적절한 버스트 크기로 변경하여 성능 저하를 최소

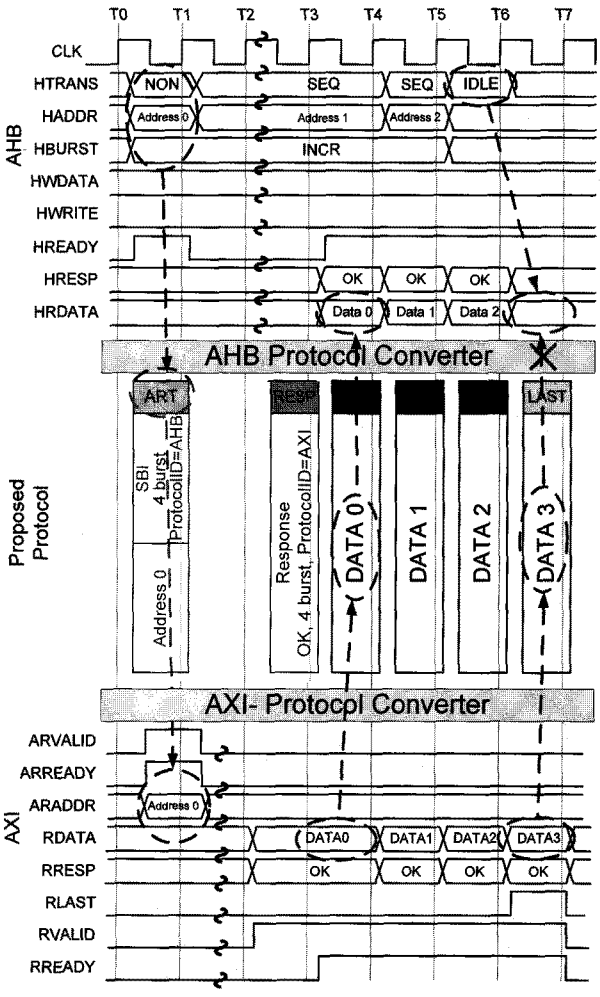

그림 4. 제안한 프로토콜을 이용한 AHB-to-AXI 변환 예
Fig. 4. Example of protocol conversion between AHB and AXI using the proposed protocol.

화시킬 수 있다. 그림 4에서 보듯이 실제 AHB IP에서 요구한 버스트 크기는 3 개 이지만 처음에는 몇 개의 데이터를 요구할지 알 수 없으므로 AHB 프로토콜 변환기에서 4 버스트로 전송 요청을 하여 AXI IP에서 4 개의 데이터를 전송했으며, 프로토콜 변환기는 4 개의 데이터 중에서 AHB IP가 요구한 3 개의 데이터만 공급하고 통신을 종료한다. 이를 통해 버스트 3의 전송의 효과를 낼 수 있다. 제안한 프로토콜로 변환하는 과정에서 1 사이클의 지연이 발생하지만 네트워크의 신호선 수가 1/2 ~ 1/5로 줄어드는 장점이 더 크게 작용하고 버스트 길이가 커질수록 1 사이클 지연은 시스템 성능에 큰 영향을 미치지 않는다.

제안하는 프로토콜에서는 IP 재활용을 위해 다른 프로토콜 인터페이스 기반의 IP와의 호환성을 보장하는 ProtocolID를 이용한다. 두 개의 프로토콜만 존재할 경우에는 성능 저하를 감수해서라도 어느 정도 프로토콜 변환이 가능하지만 셋 이상의 프로토콜이 섞여 있을 경우 프로토콜 변환기가 이미 한번 변환된 신호를 받게 되어 신호의 해석에 어려움이 따른다. 이 경우 통신 자체가 안 되거나 성능 저하가 매우 커질 수 있다. 따라서 SBI의 ProtocolID 값을 이용하여 처음 신호를 발생시킨 IP가 사용하는 프로토콜을 알 수 있고 필요할 경우 SBI의 미사용 영역을 이용하여 프로토콜 변환에 필요한 추가 정보를 제공할 수 있다. 미사용 영역의 정보는 ProtocolID에 따라 서로 다른 의미를 가지므로 적은 비트수로도 많은 정보를 전달할 수 있다. 따라서 시스템 설계자는 시스템에 포함되는 프로토콜을 파악하여 ProtocolID를 부여하고 필요할 경우 SBI의 미사용 영역을 ProtocolID에 따라 필요한 정보를 전달하도록 정의하여 가장 효율적인 통신이 가능하도록 시스템을 설계할 수 있다. 다양한 프로토콜 기반의 IP가 섞여 있는 경우 모든 IP는 프로토콜 변환기를 통해 신호를 전달하는데 변환기의 네트워크 방향은 제안한 프로토콜을 사용한다. 따라서 통신하는 상대편 IP가 어느 프로토콜을 사용하는지 알 수 없다. ProtocolID는 상대편 IP가 사용하는 프로토콜에 대한 정보를 제공하여 적절한 행동을 취할 수 있도록 해준다. 예를 들어, AXI 기반 IP가 AHB 기반 IP와 통신하는 경우 AHB 기반 IP는 전송 요청이후 응답이 올 때까지 다음 전송을 시작하지 못하므로 응답을 빨리 보내주어야 전체 시스템의 성능 저하를 막을 수 있다. 또한 사용자가 미사용 영역을 사용하기를 원하지 않으면 AHB기반 IP가 이해하지 못하는 기능을 사용하지 않음으로써 원활한 통신이 이루어지도록 할 수 있다.

III. 성능 분석

제안하는 프로토콜은 보다 적은 인터페이스 신호선을 이용하여 효율적인 네트워크 구조 설계를 가능하게 하고 제트워크 사용 효율을 높이는 것이 목적이다. 또한 기존에 널리 이용되는 AMBA AHB와 점차 사용 범위가 넓어지고 있는 AXI와의 호환성을 유지하면서 그 외의 프로토콜 기반의 IP도 이용하여 하나의 시스템을 설계할 수 있는 환경을 제공하는데 중점을 두었다. 따라서 제안하는 프로토콜이 다른 프로토콜과 비교하여 절대적으로 가장 우수한 성능을 보여줄 수는 없으나 신호선 수 대비 성능과 인터페이스 로직 및 관련 네트워크 구조의 간단함과 프로토콜 간의 통신 호환성에서는 강점을 가진다.

AXI와 비교하여 제안하는 프로토콜이 지원하지 않는 기능은 인터리빙(interleaving)과 보호(protection) 모드, 그리고 캐시(cache) 기능이다. 인터리빙은 IP와 네트워크가 모두 지원해야하고 하드웨어를 매우 복잡하게 하는데 비해 성능 향상은 특정한 경우에만 나타나 효율이 좋지 않다. 보호 모드와 캐시 기능도 용도가 제한되어 있는데 비해 IP 설계를 매우 복잡하게 하는 요인이다. 따라서 세 가지 기능은 제외하여도 프로토콜 변환기에서 해결 가능하므로 호환성 유지에 큰 문제가 없어 제외하였다. 제안하는 프로토콜의 ProtocolID 기능은 이러한 면에서 상당한 이점을 갖는다.

성능면에서는 다섯 개의 독립적인 채널을 갖는 AXI가 태생적으로 더 좋은 성능을 보인다. 즉 주소와 데이터가 별도의 채널로 분리되어 파이프라인 동작이 시작되면 최대 성능은 AXI가 더 좋을 수밖에 없다. 다만 버스트 길이가 커지면 그 차이는 10% 이내로 줄어든다. 이러한 성능 차이는 네트워크를 무시한 1:1 통신을 가정한 것으로 다대다 통신의 복잡한 상황이 되면 통신 패턴과 네트워크 구조에 따라 성능 차이는 더 좁혀지거나 역전될 수 있다. 즉 다섯 개의 복잡한 채널이 네트워크 하드웨어에는 상당한 부담을 주어 네트워크 내부의 채널 형성과 데이터 전달 과정에서 발생하는 잠복기가 성능 저하의 원인이 될 수 있다. 또한 인터페이스 로직도 다섯 개의 독립적인 채널을 제어하기 위한 다섯 개의 제어 유닛이 필요하다. 반면에 제안하는 프로토콜은

두 개의 독립적인 채널을 가지므로 두 개의 제어 유닛이면 충분하다. 신호선 하나의 효율성을 비교해보면 제안하는 프로토콜이 월등히 좋은 것을 알 수 있다. 제안하는 프로토콜과 비슷한 구조인 SNP의 경우 신호선당 효율이 AXI에 비해 최대 40% 이상 우수하므로 제안하는 프로토콜도 동일한 신호선당 효율을 보인다.[6]

구조가 비슷한 SNP와 비교해 보면 제안하는 프로토콜의 명령어가 SNP의 페이즈와 비슷한 역할을 한다. SNP에서는 읽기와 쓰기 데이터 페이즈가 분리되어 있으나 제안하는 프로토콜은 데이터 명령어만 있고 데이터의 마지막을 나타내는 LAST 명령어와 명령어의 확장을 나타내는 EXT 명령어가 추가된 것이 다르다. 읽기와 쓰기 데이터를 분리하면 네트워크에서 페이즈만으로 읽기와 쓰기를 구분할 수 있는 장점이 있으나 시스템이 1:1 통신이 아닌 경우 LAST 명령어가 없으면 네트워크는 버스트 길이를 알 수 없다. 또한 EXT 명령어가 없으면 데이터 폭이 네트워크 중간에 바뀌는 통신을 하기 어려운 문제가 발생한다. 즉, SNP는 1:1 통신에는 최적화되어 있지만 다수의 IP가 존재하는 시스템에서 네트워크 하드웨어 설계를 고려하지 않아 네트워크 설계를 어렵고 복잡하게 만든다. 또한 주소와 제어정보, 그리고 데이터로 구성되는 패킷 형식은 주소와 제어 정보를 한번에 보내는 제안하는 프로토콜에 비해 잠복기와 전송 대역폭면에서 불리하다. 버스트 길이가 작을수록 이 차이가 커진다. 따라서 제안하는 프로토콜은 AXI와 SNP의 중간에서 SNP의 단점을 어느 보완하면서 AXI에 근접하는 성능과 SNP의 신호선당 효율성이라는 장점을 그대로 가지는 프로토콜이다. SNP는 또한 제어 정보에 대한 정의를 사용자에게 맡겨 놓았다. 이는 자유도 측면에서는 장점이 될 수도 있지만 시스템 설계자에게 프로토콜 설계의 일부를 맡기는 부담을 주어 설계를 어렵게 만드는 요인이 되고 프로토콜의 특성상 표준화의 중요성을 고려하면 제어 정보는 프로토콜에서 정의하고 사용자에게 제한적인 자유도를 주는 것이 올바른 방향이다. 제안하는 프로토콜은 SBI에 제어 정보를 정의하고 일부 미사용 영역은 사용자가 정의해 사용할 수 있도록 하였다. 따라서 네트워크를 SBI 정의에 따라 설계하고 네트워크 구조에 영향을 주지 않는 제어 정보는 미사용 영역을 이용하여 특정 IP 또는 시스템에 적용되도록 할 수 있다. 표 3에 AXI, AHB, SNP와 신호선을 비교한 결과가 나타나 있다. 신호선을 나타내는 숫자중 괄호안의 값은 매스터와 슬레이브 기능을 동시에 수행하는 경우 필요한 신호수이다. 신호선당 상대 전송효율은 1:1 통신을 가정한 것으로 실제 시스템에서는 네트워크 구조와 통신 패턴에 따라 달라질 수 있다. 특히 AHB의 경우에는 실제 상황에서 상당한 성능 및 효율 저하가 예상된다.

표 3. 프로토콜간의 신호선수 및 신호선당 효율 비교
Table 3. Comparison results of the number of wires and transfer efficiency per wire.

Protocol	AXI	AHB	SNP	Proposed
신호선	268(536)	176(352)	138(138)	138(138)
신호선당 상대 전송효율	1	0.78	1.65	1.83

V. 결 론

본 논문에서는 신호선의 수를 줄여 네트워크 효율성을 강화하고 서로 다른 프로토콜 간의 호환성을 강화한 SoC용 새로운 인터페이스 프로토콜을 제안한다. 기존의 SoC용 프로토콜과는 달리 주소와 데이터 채널을 공유하도록 하였고 기존의 방식과는 달리 주소와 제어 정보를 한 번에 보내어 전송 성능을 향상시켰다. 또한 AMBA AHB 및 AXI와 호환성이 유지되도록 제어 정보를 정의하고 제어 정보에 IP의 프로토콜 정보를 포함하도록 하여 서로 다른 프로토콜 간의 통신에서 데이터 전송을 용이하게 하여 IP의 재활용을 쉽게 하도록 하였다. 성능 분석 결과 절대 성능은 AXI에 비해 다소 떨어지지만 신호선 하나당 효율은 83%정도 더 좋고 기존의 비슷한 방식의 프로토콜에 비해서도 10% 정도 성능 향상이 있었다. 제안하는 프로토콜은 기존 IP나 시스템을 재활용하여 시스템을 설계하는데 유용하게 이용할 수 있다.

참 고 문 헌

[1] ARM, "AMBA AHB Specification", IHI0011A, 1999.
[2] ARM, "AMBA AXI Specification", IHI0022B, 2004.
[3] Silicore Corporation, WISHBONE SoC Architecture Specification Revision B.3, 2002.
[4] IBM, CoreConnect Bus Architecture, 1999.
[5] OCP-IP, Open Core Protocol Specification 2.0, 2003.

[6] Jaesung Lee, Hyuk-Jae Lee, Chanho Lee, "A Phase-Based Approach for On-Chip Bus Architecture Optimization," The Computer Journal, vol.52 , No.6 , pp.626-645, 2009.10
[7] S. Lee and C. Lee, "A High Performance SoC On-chip-bus with Multiple Channels and Routing Processes", IFIP Very Large Scale Integration (VLSI-SOC), pp. 86-91, 2006. 10.
[8] S. Kumar, et al., "A network on chip architecture and design methodology", IEEE Computer Society Annual Symposium on VLSI (ISVLSI'02), pp. 105-112, April 2002.
[9] M. Millberg, E. Nilsson, R. Thid, S. Kumar, A. Jantsch, "The Nostrum backbone a communication protocol stack for networks on chip", Proceedings of the VLSI Design Conference, 2004. 1.
[10] F. Moraes, A. Mello, L. M. oller, L. Ost, N. Calazans, "A low area overhead packet-switched network on chip: architecture and prototyping", IFIP Very Large Scale Integration (VLSI-SOC), pp. 318-323, 2003.
[11] E. Bolotin, I. Cidon, R. Ginosar, A. Kolodny, "QNoC: QoS architecture and design process for network on chip, The Journal of Systems Architecture", Special Issue on Networks on Chip 50, Vol. 2. pp. 105-128, 2004.
[12] J. Liang, S. Swaminathan, R. Tessier, "aSOC: A scalable, single-chip communications architecture," IEEE International Conferenceon Parallel Architectures and Compilation Techniques, pp.37-46, 2000.10.
[13] F. Karim, A. Nguyen, S. Dey, "An interconnect architecture for network systems on chips", IEEE Micro 22, Vol. 5. pp. 36-45, 2002.
[14] E. Rijpkema, K. Goossens, A. Radulescu, "Trade offs in the design of a router with both guaranteed and best-effort services for networks on chip", Design, Automation and Test in Europe(DATE'03), pp.350-355, 2003.3.
[15] D. Wiklund, D. Liu, "SoCBUS: switched network on chip for hard real time systems", International Parallel and Distributed Processing Symposium (IPDPS), 2003. 4.
[16] 이상헌, 이찬호, 이혁재, "효율적인 다중 채널 On-Chip-Bus를 위한 SoC Network Architecture," 전자공학회 논문지, 제42권 SD편 제2호, pp.143-150, 2005.2
[17] PCI-SIG , Conventional PCI Specification 3.0
[18] PCI-SIG , PCI-X Specification 2.0.
[19] PCI-SIG , PCI Express Base Specification 2.0
[20] HyperTransport Technology Consortium, HyperTransport I/O Link Specification 1.1, 2003. 8.
[21] Intel, Intel QuickPath Architecture White Paper, 2008. 3.
[22] ARM, "AMBA 2 AHB to AMBA 3 AXI Bridges", DTO0008B, 2006. 2.
[23] ARM, "AHB-Lite Overview", DVI0044A, 2001.

저 자 소 개

이 찬 호(정회원)
대한전자공학회 논문지
제43권 SD편 제9호 참조
현재 숭실대학교 정보통신전자공학부 교수