XSNP: 고성능 SoC 버스를 위한 확장된 SoC 네트워크 프로토콜

(XSNP: An Extended SoC Network Protocol for High Performance SoC Bus Architecture)

이 찬 호 † 이 상 헌 †† 김 응 섭 ††† 이 혁 재 ††††

(Chanho Lee) (Sanghun Lee) (Eung-Sup Kim) (Hyuk-Jae Lee)

요 약 최근, SoC 설계연구가 활발히 진행되고 있으며, 하나의 시스템에 보다 많은 수의 IP가 포함되고 있다. 많은 IP간의 효율적인 통신과 재사용율을 높이기 위해 다양한 프로토콜과 버스 구조들이 연구되고 있다. 기존의 공유 버스 구조의 문제점을 해결하기 위해 제안된 SNP(SoC Network Protocol)와 SNA(SoC Network Architecture)는 각각 peer-to-peer 방식의 프로토콜과 버스 구조이다. 한편 AMBA AHB는 대규모 SoC 시스템에 다소 부적절한 구조를 가짐에도 불구하고 산업 표준으로 자리매김 해왔다. 따라서 기존의 많은 IP들이 AMBA 인터페이스를 가지고 있으나 SNP와는 프로토콜과 완벽하게 호환되지 않는 문제점을 가지고 있다. 기존의 IP들의 인터페이스를 SNP로 바꾸기 전까지는 새로 제안된 버스 구조에서도 AMBA AHB와의 호환성을 완전히 배제할 수가 없다. 본 논문에서는 기존의 SNP가 확장된 XSNP(extended SNP) 스펙과 SNA 기반 시스템에서 이를 지원하는 SNA 컴포넌트를 제안한다. AMBA AHB와 SNP 사이의 프로토콜 변환을 지원하기 위해서 기존 SNP의 페이즈를 1 비트 확장하여 새로운 8개의 페이즈를 추가하였다. 따라서 AMBA 호환 가능한 IP는 SNP를 통해 성능 감쇠 없이 AHB-to-XSNP 변환기를 통해 통신할 수 있다. 또한 이러한 확장 방법은 AMBA AHB 뿐 아니라 SNP와 다른 버스 프로토콜 사이의 신호 변환에도 이용하여 SNP의 유연성과 성능을 향상시킬 수 있다. 제안된 구조의 검증/평가를 위해 다양한 시뮬레이션을 수행하였으며, AMBA AHB와의 호환성에 있어 문제가 없다는 것을 검증하였다.

키워드 : SNA, SNP, 온 칩 버스, SoC 버스, 온 칩 네트워크, 크로스바 라우터, AMBA

Abstract In recent years, as SoC design research is actively conducted, a large number of IPs are included in a system. Various bus protocols and bus architectures are designed to increase IP reusability. Among them, the AMBA AHB became a de facto standard although it is somewhat inadequate for a large scale SoC. We proposed SNP and SNA, high performance on-chip-bus protocol and architecture, respectively, to solve the problem of the conventional shared buses. However, it seems to be imperative that the new on-chip-bus system support AMBA-compatible IPs for a while since there are a lot of IPs with AMBA interface. In this paper, we propose an extended SNP specification and a corresponding SNA component to support ABMA-compatible IPs used in SNA-based system. We extend the phase of the SNP by 1 bit to add new 8 phases to support communication based on AMBA protocol without penalty of elongated cycle latency. The AHB-to-XSNP converter translates the protocol between AHB and SNP to attach AMBA-compatible IPs to SNA based system. We show that AMBA IPs can communicate through SNP without any degradation of performance using the extended SNP and AHB-to-XSNP converter.

Key words : SNA, SNP, on-chip-bus, SoC bus, on-chip-network, crossbar router, AMBA

· 본 연구는 한국과학재단 목적기초연구(R01-2003-10582-0)의 연구비와 IDEC의 CAD 툴 지원으로 수행하였습니다.

† 정 회 원 : 숭실대학교 정보통신전자공학부 교수
chlee@ssu.ac.kr

†† 학생회원 : 숭실대학교 전자공학과
blueys76@engineer.ssu.ac.kr

††† 학생회원 : 서울대학교 전기 · 컴퓨터공학부
eskim@capp.snu.ac.kr

†††† 정 회 원 : 서울대학교 전기 · 컴퓨터공학부 교수
hjlee@ssu.ac.kr

논문접수 : 2006년 1월 25일
심사완료 : 2006년 4월 19일

1. 서 론

공정기술과 EDA 툴의 발전에 따라서 하나의 실리콘 다이(die) 에서 보다 많은 IP 블록의 통합이 기술적으로 가능하게 되었다. 이런 기술의 발달로 멀티미디어와 통신 등 연산 작업량이 많은 곳에서 필요로 하는 병렬 처리 연산이 요구되고 있다. 멀티프로세서 SoC(System-on-Chip)에서는 상호 통신구조(communication architecture)에서 병목현상이 발생하며, 공유 통신 자원에 대한 접근을 동시에 요청할 때 시스템의 성능 저하를 막을 수 있는 효율적인 방법이 필요하게 된다. 이러한 문제는 기존의 SoC 버스가 공유버스(shared bus)구조를 가짐으로써 병목현상이 발생하기 때문이다.

현재 세계 시장의 70% 이상을 차지하고 있는 ARM 사의 AHB는[1] 다른 버스에 비해서 훨씬 간단한 프로토콜을 가지기 때문에 소규모 SoC에서는 비교적 쉽게 사용할 수가 있었다. 이에 따라 많은 IP 벤더들이 AHB 호환 IP를 개발하여 SoC 시장에서 AHB는 온 칩 버스의 대표자리를 굳혀 왔으나, 이처럼 간단한 버스 프로토콜은 SoC의 규모가 방대해지면서 그 효율성의 한계를 드러내고 있다. 즉, AHB는 단순한 SoC 설계에는 적합하지만 복잡한 SoC용으로는 그 기능이 부족한 것이다. 따라서 AHB의 문제점을 극복하는 새로운 SoC 버스 표준의 필요성이 절실하다.

이에 대한 대안의 하나로 연구되고 있는 NoC (Network on Chip)의 경우도 아직까지 구현 형태가 지나치게 복잡하고 중재 알고리즘 처리를 위한 계산량이 많아 오히려 전력 소모 증가와 성능 저하를 가져올 수 있다. 이는 기존의 슈퍼컴퓨터나 서버 제작을 위한 네트워크 구현법을 그대로 모방하기 때문에 생기는 문제이다[2]. Gurrier 등은 버스와 네트워크 구조를 사용할 때 장/단점에 대해서 언급했는데 네트워크 구조를 사용할 때 내부구조의 특성상 추가적인 잠복기(latency)가 발생하며, 상당한 실리콘 면적을 차지하게 된다는 것을 주요 단점으로 지적했다[3].

대부분의 기존 SoC 버스구조는 중재 토폴로지(topology)를 갖는 다중 버스 분할 연결을 지원하고 동시에 적절한 확장성을 지원하도록 설계되었다. AMBA[4], Wishbone[5], CoreConnect[6] 등이 적절한 예로 볼 수 있다. 이들은 중재정책 및 추가적인 몇몇 특징 이외에는 서로 유사한 구조를 가지고 있기 때문에 AMBA 버스와 유사한 문제점을 갖고 있다.

따라서 이러한 문제점들을 해결하기 위해서 단순한 구조를 가지면서 동시에 다중 마스터에 대해서 다중 채널을 제공함으로써 대역폭을 확보하며, 버스에 추가되는 IP의 증가를 고려하여 확장이 용이한 SNP(SoC Network Protocol)와 SNA(SoC Network Architecture)가 제안되었다[7,8].

SNA는 주소와 대부분의 제어 신호를 데이타 선에 포함시키고 제어 신호선의 추가를 최소화 하도록 하여 연결선의 수를 대폭 줄인 SNP를[9] 인터페이스 프로토콜로 사용한다. SNA 내부의 물리적 구성요소는 4N 개의 채널을 갖는 크로스바 라우터(Crossbar Router)와 이들의 상호연결을 중재하는 광역 중재자(Global Arbiter), 그리고 IP 또는 sub-system을 크로스바 라우터와 연결시키는 스위치 래퍼/브리지(Switch Wrapper/Bridge)로 구성된다.

현재까지 다수의 버스 구조와 프로토콜이 발표되었지만, ARM 프로세서와의 쉬운 연결성, 공개된 라이센스, 단순한 프로토콜의 특징으로 AMBA는 여전히 학계에서나 산업계에서 많이 쓰이고 있는 버스구조이다. 그리고 기존의 무수한 IP들이 AMBA를 지원하기 때문에 이를 버리고 새로운 온 칩 버스를 채택하기 위해서는 기존의 IP를 다시 설계해야 하는 까다로운 단계가 필요하게 된다. 따라서 대부분의 IP 사업자들은 AMBA 호환성을 가지는 구조로 설계해왔고, 우리는 이러한 시장 특성을 무시할 수 없었다. 이를 위해 AMBA기반 IP를 SNA에 쉽게 연결할 수 있는 AMBA-to-SNP 변환기가 필요하다. 하지만 SNP가 근본적으로 AMBA 프로토콜과 호환성이 없기 때문에 변환과정에서 약간의 성능 감소가 발생되고 메모리의 추가로 면적이 커지는 문제가 발생한다. 데이타 교환에 따른 대역폭의 한계로부터 시스템의 성능을 향상시키기 위해 기존의 시스템을 유지한 채 버스만을 SNA로 교체하려는 경우 응용 시스템에 따라서는 큰 효과를 보지 못하는 결과를 가져올 수도 있다. 이러한 문제는 SNP와 AMBA AHB와의 프로토콜 호환성의 결여에서 오는 문제이므로 이를 근본적으로 해결해야 한다.

본 논문에서는 이러한 문제를 해결하기 위해 AMBA AHB와 SNP간의 호환성을 지원하는 XSNP(eXtended SNP)를 제안한다. XSNP는 기존 SNP의 큰 특징인 페이즈(phase)를 1 비트 확장시켜 8 개의 페이즈를 추가시킴으로써 AMBA AHB와 호환성을 확보할 수 있고 프로토콜 변환기에서 메모리를 제거하여 면적을 줄일 수 있다. 이에 따라 프로토콜 변환에 따른 페널티를 없애고 성능 감소를 막을 수 있다. 또한 XSNP는 기존 SNP의 페이즈를 단순히 1 비트 확장시켰기 때문에 기존 SNP와의 호환성에도 전혀 문제되지 않는다. SNP 인터페이스를 가지는 기존 SNA 시스템을 페이즈 신호 1 비트만 추가되도록 변환하면 어떤 영향도 받지 않는다. 따라서 XSNP 프로토콜을 사용하는 SNA 시스템은 AMBA AHB IP와도 완벽하게 호환 가능하다. 이러한

확장 방법은 AMBA AHB 뿐만 아니라 다른 버스 프로토콜에 기반한 IP를 SNA/SNP 시스템에서 이용할 때도 큰 성능 저하 없이 적용할 수 있어 SNA/SNP의 활용도를 크게 높일 수 있다. 확장된 프로토콜을 검증하기 위해 AHB-to-XSNP 프로토콜 변환기를 설계하고 FPGA 상에서 동작을 검증하였다.

2. SNP 기반의 프로토콜 변환기 구조

SNP는 IP 증가에 따라 늘어나는 물리적 선(wire) 수를 프로토콜 레벨에서부터 줄일 수 있게 고려되었다. SNP는 데이타, 주소, 제어 세 그룹의 신호들이 모두 하나의 32 비트 채널을 통해 버스 전송이 가능하게 고안되었다. 따라서 양방향(duplex)통신을 위해서 64 비트의 버스 선이 필요하며, 기타 사이드 밴드(side-band) 신호를 모두 포함한다 해도 74개의 선만으로 양방향 통신을 완벽히 수행할 수 있다. 사이드 밴드 신호에는 페이즈(phase) 신호 6비트(각 방향에 따라 3 비트씩)가 포함되어있다. 페이즈 신호는 IDLE, RP(*response*), RD(read data), SP(special phase), CO(control information), WA(write address), WD(write data), RA(read address)의 8 개의 채널 상태를 나타내기 위해 사용된다. 이외에 리셋(reset), 클럭(clock), VALID, READY 신호 4 종류이다. 여기서 VALID/READY(각 방향에 따라 각각 1 비트씩)는 peer-to-peer 통신을 하기 위해 기본적으로 필요한 validation/ack를 나타내는 신호이다[8].

그림 1은 기존 SNA의 스위치 래퍼의 개념적인 내부 구조를 보여준다. SNP는 peer-to-peer 프로토콜이기 때문에 버스 요청/승인 관련 신호는 포함하지 않는다[8]. 따라서 스위치 래퍼는 크로스바 라우터와 버스 요청/승인을 보내고 받는 작업을 대신 수행해야 한다. 이는 스위치 래퍼 컨트롤러에 의해 생성되고 전달된다. 그리고 스위치 래퍼 컨트롤러는 현재 다이렉트 라우팅 시 어떠한 라우팅도 허용하지 않는다는 것을 상태신호를 이용하여 크로스바 라우터에 알리는 역할을 수행한다.

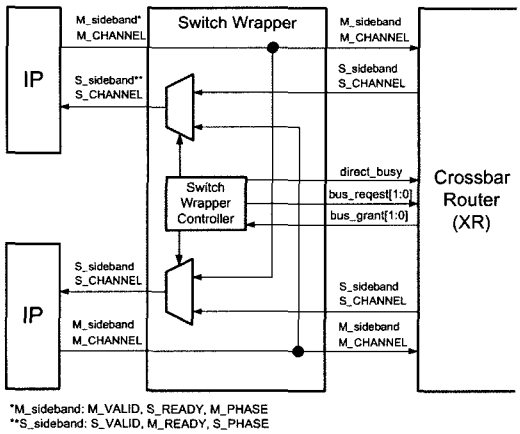

그림 1 SNP기반의 SNA 스위치 래퍼 구조

SNP 기반의 AMBA AHB 호환용 스위치 래퍼는 내부에 프로토콜 변환기를 가지고 있고 버스 요청과 다이렉트 라우팅 모드를 지원한다. 그림 2에서 기존의 SNP 기반의 AMBA AHB 호환용 스위치 래퍼 구조를 보여준다. AMBA IP를 스위치 래퍼에 연결할 때, SNA와 SNP 인터페이스를 따르기 위해 포트 매핑과 프로토콜 변환이 필수적으로 요구된다. 플로우 컨트롤러(flow controller)는 이러한 역할을 수행한다.

AMBA AHB는 버스트 전송 시 SNP와 서로 다른 응답 정책을 사용한다. AMBA AHB의 경우 버스트 전송시 마스터는 매 읽기/쓰기 트랜잭션 사이클마다 슬레이브로부터 응답을 필요로 한다. 만약 응답이 없을시 다음 트랜잭션이 이어질 수 없는 프로토콜 특징을 가지고 있다. 반면, SNP는 전체 버스트 전송이 끝난 뒤에 응답을 요구한다. 이는 SNP가 버스트 전송을 강화하여 백본(back-bone) 통신용 프로토콜에 적합하게 설계되었기 때문이다. 따라서 플로우 컨트롤러는 이러한 프로토콜상의 비호환성 부분을 극복하기 위해 FIFO(first-in first-out)와 같은 내부 메모리를 포함하여 버스트 전송시 데이타를 모두 저장한다. 이때, 마스터는 플로우 컨트롤러에 데이타를 전송하고 플로우 컨트롤러는 AMBA AHB 프로토콜에 맞는 응답을 보내며, 이후 저장된 데이타는 버스트로 목적지 슬레이브에 저장된다. 그러나 이 방법에는 여러 가지 문제점이 내포된다. 먼저 내부 메모리로 인한 플로우 컨트롤러의 면적문제이며, 두 번째로 내부 메모리로 인한 전송 잠복기(transfer latency)가 증가한다는 것이다. 마지막으로, 이러한 방법으로는 AMBA AHB의 버스트 전송모드 중 "unspecified length" 버스트 전송을 완벽히 변환할 수 없다는 것이다.

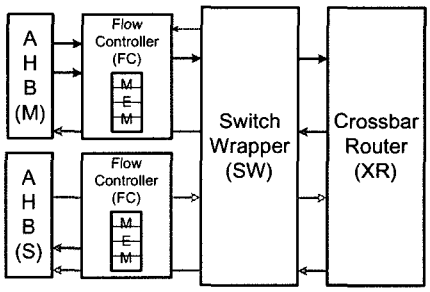

그림 2 SNP기반의 AMBA AHB 호환용 스위치 래퍼

3. XSNP 기반의 프로토콜 변환기 구조

AMBA AHB 트랜잭션 신호를 SNP로 변환할 때 발

생하는 문제를 해결하기 위해 기존 SNP의 MSB를 1 비트 확장하여 XSNP를 정의하였다. 확장된 비트는 AHB-to-XSNP 변환기에 의해서만 인식되며, 다른 SNA 컴포넌트에서는 확장된 비트를 이용해 적당한 트랜잭션(transaction)으로 변환한다. 따라서 SNP 기반의 IP와도 아무런 문제없이 통신이 가능하다.

표 1에서 추가된 XSNP의 페이즈 타입을 보여준다. 위쪽 흰 바탕에 표시된 것은 기존 SNP의 페이즈를 나타내며, 아래쪽 어두운 바탕 위에 표시된 것은 확장된 페이즈를 나타낸다. 기존 SNP 페이즈가 3 비트가 모여 하나의 페이즈를 나타내는 것과는 달리, 확장된 XSNP 페이즈는 비트별로 의미가 정의된다. 만약 확장된 비트가 '0'일 경우 XSNP는 기존 SNP와 정확히 일치하며, '1'일 경우 AHB-to-XSNP 변환기는 AMBA AHB IP로부터 오는 신호를 인식하여 페이즈 신호를 AMBA AHB의 적당한 신호로 변환한다.

확장된 페이즈 신호의 하위 3 비트 신호는 AMBA AHB의 사이드 밴드 신호인 HWRITE, HSIZE, HLOCK의 상태를 나타낸다. PHASE[0]은 '0' 또는 '1' 이냐에 따라 각각 읽기와 쓰기 트랜잭션을 나타낸다. PHASE[2:1]는 트랜잭션의 HSIZE를 나타내며, "00"은 "byte", "01"은 "half-word", "10"은 "word"를 의미하고 "11"은 "locked word" 전송을 나타낸다.

AMBA AHB 프로토콜에서의 다양한 전송 타입은 XSNP에서는 "unspecified length" 버스트 전송으로 변환된다. 래핑(wrapping) 버스트 타입은 두 개의 "unspecified length" 버스트 전송으로 나눠진다. 이는 XSNP에서는 주소값이 항상 증가한다고 가정하기 때문이다. 버스트 전송의 AHB 사이드 밴드 신호가 소스와 같은 버스트 타입으로 전송되지는 않지만, 타이밍 순서는 같은 순서로 전송되어 결과는 동일하다. XSNP에서는 기존 SNP에서 시스템 설계자가 사용할 수 있게 남겨둔 "SP(special phase)"를 단일 전송 또는 "unspecified length" 버스트 전송의 끝을 나타내는 데 사용한다. XSNP와 프로토콜 변환기는 어떤 AMBA AHB 스펙에도 위배되지 않는다.

XSNP의 트랜잭션 과정은 기존의 SNP를 기반으로 하여, SNP IP는 다른 차이점을 인지하지 못하고 AMBA AHB 기반의 IP는 프로토콜 변환기를 통해 AHB 신호를 수신하여 AMBA AHB 시스템에서와 같이 동작할 수 있다. AHB-to-XSNP 변환기는 단일 채널의 AMBA AHB 프로토콜과 같은 성능을 낼 수 있고 SNA의 스위치 래퍼는 기존 구조에서 저장되어야 했던 AHB 트랜잭션 신호를 저장하지 않아도 된다. 또한 기존 AMBA AHB 기반의 시스템에서 AMBA AHB 버스를 SNA로 교체할 경우 시스템을 수정하지 않아도 정상 동작할 수 있고 SNA의 다중 채널 지원으로 인하여 버스 시스템은 대역폭이 증가하여 전체적으로 시스템 성능을 향상시킬 수 있다. 또한 XSNP와 프로토콜 변환기를 이용하여 AMBA AHB IP간 통신 시 성능 감소 없이 76 개의 와이어만을 사용하기 때문에 인터커넥트 와이어를 대폭 줄일 수 있다.

표 1 XSNP의 페이즈 타입

Phase Type	PHASE[3:0]
IDLE (ID)	0000
Response (RP)	0001
Read Data (RD)	0010
Special (SP)	0011
Control Information (CO)	0100
Write Address (WA)	0101
Write Data (WD)	0110
Read Address (RA)	0111
Read phase (R)	1xx0*
Write phase (W)	1xx1
Byte transfer (B)	100x
Half-word transfer (H)	101x
(Full) Word transfer (F)	110x
Locked word transfer (L)	111x

* x : don't care condition

제안된 시스템에서 제약이 있다면 HSIZE가 32 비트까지 지원되며, HLOCK 신호가 "word" 전송에 대해서만 지원된다는 것이다. 또한 32 비트 데이타 채널이 하나만 존재하므로 back-to-back 전송에서 한 사이클 잠복기가 발생한다. 그러나 이러한 상황은 빈번히 발생되지 않아 무시될 수 있는 조건이다. 또한 반드시 필요하다면 페이즈 신호를 1 비트 추가하여 그러한 제한 조건도 없앨 수 있다. 현재로서는 그러한 제한 조건이 성능에 영향을 거의 주지 않으므로 페이즈 신호를 추가할 필요성은 없다. 따라서 제안된 구조는 AMBA AHB 프로토콜과의 호환성 부분에서 어떤 문제점도 갖지 않는다.

4. 동작 및 검증

버스의 검증은 버스에 연결된 IP의 수가 많기 때문에 시뮬레이션 할 때 모든 신호를 설계자가 일일이 확인하는 것은 불가능에 가깝다. 따라서 AHB 규격을 정확히 따르는 시뮬레이션용 AHB IP를 테스트용으로 개발하였다. 이 테스트 IP들 간에 통신이 제대로 이루어지는지를 통해 버스 모듈의 동작을 검증한다.

테스트 IP는 마스터와 슬레이브로 구성된다. 기본적인 개념은 그림 3과 같다. 마스터 IP는 파일로부터 지정된 AHB 트랜잭션 기술(description)을 읽어서 AHB 트랜잭션을 시작한다. 슬레이브 IP는 트랜잭션 신호를 받아서 지정된 주소에 값을 쓰거나 저장되어 있는 값을 반

환한다. 마스터 IP는 반환된 값과 기대값을 비교하여 다를 경우에 이를 사용자에게 알린다. 또한 슬레이브로부터 OKAY가 아닌 응답(response)이 왔을 때는 이를 알려주고 각종 비정상적인 조건이 발생했을 때 이를 사용자에게 알린다. 슬레이브 IP는 트랜잭션 제어 신호가 AHB 규격을 제대로 지키고 있는지를 검사하여 위반(violation)이 발생했을 때와 각종 비정상적인 신호가 들어왔을 때 이를 알린다. 슬레이브에는 또한 응답을 바꿀 수 있는 기능을 제공하여, 지정된 주소에 값을 쓰면 지정된 응답을 하도록 할 수 있다. RETRY나 SPLIT의 경우에는 지정된 주소에 숫자를 쓰면 그 수만큼의 사이클 뒤에 RETRY나 SPLIT 상황을 해제하도록 할 수 있다.

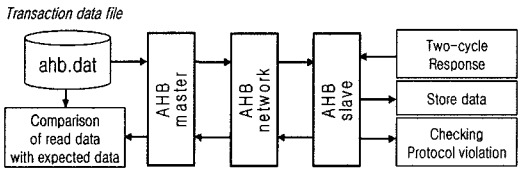

그림 3 AHB 트랜잭션 테스트용 IP의 개념도

그림 4는 SNA에서 XSNP를 사용했을 때의 프로토콜 변환과 신호의 흐름 예를 보여준다. 먼저, AHB 마스터는 트랜잭션을 시작하고, AHB 신호를 프로토콜 변환기로 보낸다. 수신된 AHB 신호는 XSNP로 변환되며, M_CHANNEL(요청자가 내보내는 채널)을 통해 크로스바 라우터로 출력된다. 크로스바 라우터는 해당 목적지까지 신호를 전달하며, AHB 슬레이브의 프로토콜 변환기는 이를 다시 AHB 신호로 재생한다. AHB 슬레이브의 응답신호는 XSNP 신호로 변환되어, S_CHANNEL(목적지에서 내보내는 채널)을 통해 AHB 마스터로 전달된다. SNP IP는 기존 SNP 프로토콜을 사용하며, 스위치 래퍼는 상위 한 비트에 '0'을 추가하여 XSNP 신호로 변환한다. AHB 마스터가 트랜잭션을 시작할 때, 프로토콜 변환기는 M_CHANNEL로 첫 번째 주소(A1)을 생성하여 내보낸다. 이때 적당한 XSNP 페이즈 신호(R/W)가 같은 시기에 출력된다. "wait state"가 없다고 가정했을 때, 쓰기 데이타는 "W" 페이즈와 함께 M_CHANNEL을 통해 전송되며, 마지막 전송 데이타는 "SP"와 함께 전송되어 트랜잭션의 끝을 알린다. 목적지에서는 S_PHASE를 통해 "RP"를 전달한다. 읽기 트랜잭션의 경우, 읽기 데이타는 "RP"와 함께 S_CHANNEL을 통해 전송된다. M_PHASE는 "ID"를 유지하고, 마지막 전송에 "SP"를 전달한다.

AHB IP를 SNA에 연결하기 위해서는 AHB 신호를 XSNP로 바꾸고, 전송된 XSNP를 다시 AHB 신호로

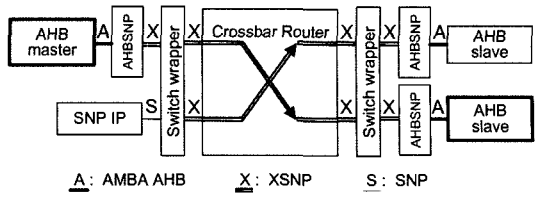

그림 4 SNA를 통한 AHB 기반 IP 사이의 통신

재생시키는 작업이 필요하다. 이 작업이 제대로 이루어지지 않으면, SNA를 이용한 AHB IP간에 통신이 이루어질 수 없다. 따라서 AHB와 XSNP 프로토콜 간의 상호 변환이 정상적으로 이루어지는지를 검증하는 것이 중요하다. 그림 5는 XSNP 프로토콜의 트랜잭션 타이밍도로서, 매 사이클 마다 변환되는 신호를 자세하게 나타낸다. 맨 위에 나타난 AHB 신호는 AHB 마스터와 마스터에 연결된 변환기 사이에 나타나는 AHB 신호를 나타낸 것이다. {HTRANS, HBURST, HADDR, HWIRTE, HWDATA}는 AHB 마스터에서 내보내는 신호이고, {HREADY, HRESP, HRDATA}는 마스터에 연결된 변환기가 AHB 마스터에게 응답하는 신호이다. 그 아래 M_CHANNEL은 AHB 마스터의 신호을 받아서 마스터쪽에 있는 AHB-to-XSNP 변환기가 변환한

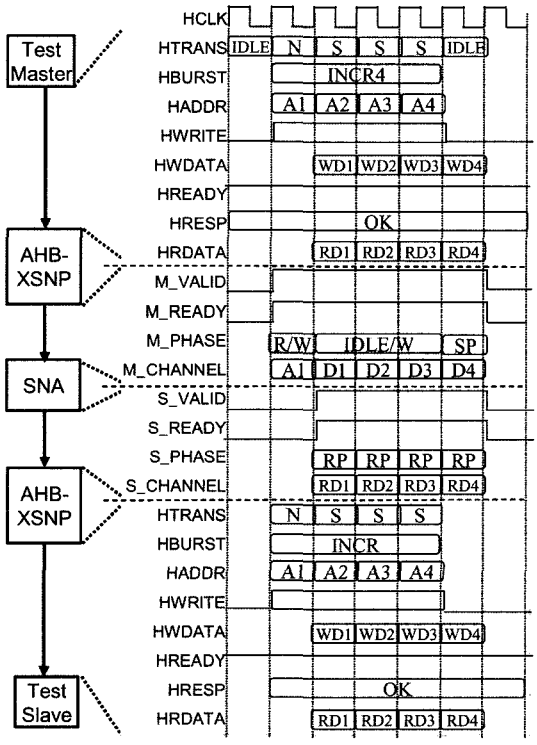

그림 5 AHB와 XSNP간의 신호 변환

신호이다. 그 아래의 S_CHANNEL은 AHB 슬레이브의 응답을 받아서 슬레이브쪽에 있는 변환기가 XSNP로 변환하여 마스터쪽에 있는 변환기로 전송하는 신호이다. 맨 아래 AHB 신호는 AHB 슬레이브와 슬레이브에 연결된 변환기 사이에 나타나는 AHB 신호를 표시한 것이다. {HTRANS, HBURST, HADDR, HWRITE, HWDATA}는 슬레이브쪽 변환기가 M_CHANNEL의 신호를 보고 AHB 마스터 신호를 재생한 것이다. {HREADY, HRESP, HRDATA}는 AHB 슬레이브가 슬레이브쪽 변환기가 재생한 AHB 마스터 신호에 대해 응답한 신호이다.

XSNP에서는 모든 버스트가 길이가 정해져 있지 않고 주소는 트랜잭션이 진행됨에 따라 증가한다고 정의하고 있다. 하지만 AHB의 래핑 버스트의 경우에 주소가 증가하다가 일정 경계(boundary)를 넘어서면 베이스 주소(base address)로 변경된다. 이 경우는 XSNP로는 표현할 수가 없다. 왜냐하면, XSNP에서 주소는 처음에 한 번만 전송되고 수신측에서 주소를 증가시켜 사용하며, 그림 6의 왼쪽에 나타난 것처럼 XSNP 페이즈는 래핑 버스트를 표현할 수 없기 때문이다. 따라서 하나의 AHB 래핑 버스트는 두 개의 XSNP 증가(increment) 버스트로 나누어서 전송된다. 이 과정에서 약간의 변환 오버헤드가 발생한다. 그림 6의 오른쪽에 보여주고 있는 것처럼 주소가 래핑되기 전까지가 하나의 XSNP 트랜잭션이, 래핑된 후가 또 하나의 XSNP 트랜잭션이 된다.

AHB는 파이프라인 버스(pipelined bus) 프로토콜로서 데이타가 나오기 전에 그에 해당하는 주소와 제어 신호가 바로 직전 사이클에 나오게 된다. AHB HTRANS 신호에는 BUSY가 존재한다. 이것은 마스터 IP가 다음 beat를 바로 진행할 수 없을 때, 다음 데이타 사이클을 한 번 건너뛴다는 의미이다. 그런데 XSNP는 주소와 데이타가 같은 채널을 통해 전송되기 때문에 파이프라인 구조를 갖지 않는다. AHB-to-XSNP 변환기에서 AHB 마스터 IP로부터 BUSY 신호를 받았을 때는 BUSY 이전 beat에 대한 데이타를 전송 중이다. 따라서 다음 사이클에 M_VALID를 "0"으로 내림으로써 BUSY 신호를 전송할 수 있을 것이다. 그러나, BUSY가 반영된 XSNP 신호는 슬레이브쪽 변환기에서 AHB 신호로 BUSY를 반영하는 데는 문제가 있다. 그림 7의 왼쪽에 나타난 것처럼, M_VALID가 내려간 사이클을 받았을 때는 이미 슬레이브쪽 변환기에서 주소를 내보낸 뒤이다. 따라서 주소를 내보낸 뒤에 기대했던 데이타가 도착하지 않은 경우가 된다. AHB에서 이미 내보낸 주소는 다음 사이클에서 취소할 수가 없으므로 변환기에서는 슬레이브쪽에 데이타를 제공해야 하지만, 유효한 데이타를 받지 못한 상태가 되므로 기대하지 않았던 데이타가 슬레이브쪽으로 전달된다. 이를 방지하기 위해서, 마스터쪽 변환기에서 BUSY를 받았을 때, 현재 보내고 있는 데이타가 트랜잭션의 마지막 데이타임을 SP로 표시한다. BUSY 사이클이 끝나고 다시 주소가 마스터로부터 나올 때 변환기는 다시 새로운 트랜잭션을 시작한다. 이렇게 하면, 하나의 AHB 트랜잭션은 BUSY 전 후의 두 개의 트랜잭션으로 나뉘게 되지만, 그림 7의 오른쪽에 나타낸 것처럼 BUSY 사이클이 의도한 동작을 슬레이브쪽 변환기에서 정확히 재생할 수 있다. 그리고 BUSY 때문에 트랜잭션이 나뉘게 되더라도 BUSY 때문에 쉬는 사이클을 이용해서 나눠진 두 번째 트랜잭션의 주소를 전송하기 때문에 사이클 손해는 없다.

AHB와 XSNP 프로토콜 변환상의 정확한 성능비교를 위해 시뮬레이션을 수행하였다. 버스 프로토콜은 그 특성상 프로토콜 자체만의 비교는 무의미하며, 버스와 연

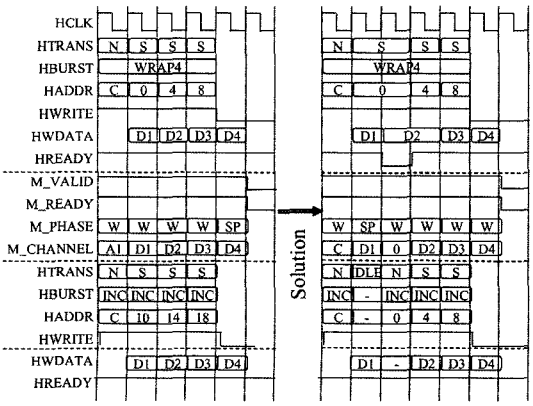

그림 6 AHB의 래핑 버스트를 XSNP로 변화시킬 때의 문제점과 해결 방법

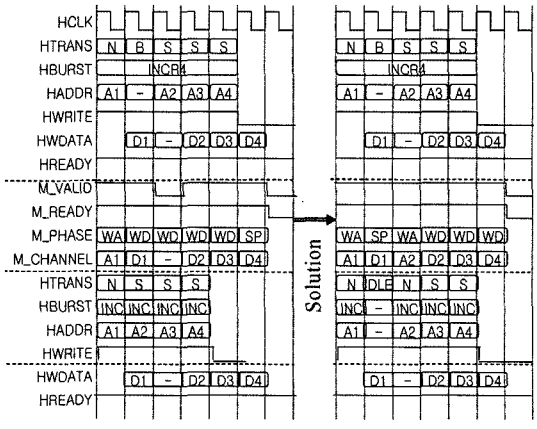

그림 7 AHB BUSY 신호를 XSNP로 변화시킬 때의 문제점과 해결 방법

계되어 동작하였을 때의 성능을 비교하여야 한다. 따라서, 동일한 AHB 시뮬레이션 모델을 그림 3, 4와 같이 AMBA 버스와 SNA 구조에 각각 적용하여 시뮬레이션한 결과를 표 2에서 보여주고 있다. AHB는 74 사이클(request/grant 사이클 고려, back-to-back 통신 배제), XSNP는 79 사이클을 소비하였다. 시뮬레이션 결과 약 5 사이클의 성능차이를 보이고 있는데, 이는 그림 6에 나타난 바와 같이 래핑 버스트의 변환 과정 중에 발생되는 오버헤드 때문이다. 표 2의 두 번째 시뮬레이션에서 INCR4(증가형 4 버스트)와 Single(단일 전송)을 섞어서 시뮬레이션을 100 회 반복하였다. 실험에서 back-to-back 통신은 배제되었다. AMBA AHB에서 디폴트 마스터가 지정될 경우 요청(request) 사이클이 생략되는데 이는 일반적인 상황이 아닌 특수 마스터에만 발생하는 경우이기 때문에 본 실험에서는 매 전송마다 요청 사이클이 필요하다고 가정하였다. 실험에 사용된 두 구조의 일반적 성능은 동일함을 알 수 있다.

기존의 SNP를 이용하여 AHB 통신을 수행할 경우 모든 버스트 통신을 단일전송으로 변환하여 전송해야한다. 이는 SNP와 AHB와의 응답 정책의 차이 때문이다. 따라서 5 버스트를 전송할 경우 AHB에서는 8 사이클로 전송 완료가 가능하지만 SNP를 이용할 경우 20 사이클이 필요하다. 따라서 SNP를 이용하여 AHB 통신을 지원할 경우 2 배 이상의 성능 감소가 발생할 수 있다. 그러나 XSNP는 변환기에서 응답을 생성하여 보냄으로써 버스트 전송이 가능하여 8 사이클에 전송을 완료할 수 있다.

래핑 버스트 통신의 경우 XSNP에서 오버헤드로 인해 AHB보다 한 사이클 더 소모하였으나 실제 AHB에서 래핑 버스트는 일반적인 슬레이브 통신에 사용되는 것보다 캐쉬(cache) 라인을 채우기 위해 주로 사용된다[9]. 즉, 조건 2의 실험결과에서 보는 것처럼 일반적인 슬레이브와의 통신에서는 XSNP가 AHB와 동등한 성능을 보일 수 있음을 알 수 있다. 따라서 XSNP는 AHB보다 적은 수의 버스 와이어로 큰 성능 감쇠 없이 호환성을 유지할 수 있음을 실험을 통해 알 수 있었다.

표 2 프로토콜 성능비교

	AHB	*Protocol Converter* (for XSNP)
조건 1*	74	79
조건 2**	900	900

* : AHB의 8가지 전송을 한 번 씩 수행

** : AHB의 Increment 4 burst/Single 전송 100 회 반복(Back-to-Back 통신이 아님을 가정)

5. 설계 및 구현

AHB-to-XSNP 프로토콜 변환기는 Verilog HDL을 이용하여 설계되었으며, Xilinx FPGA와 0.35um CMOS 셀 라이브러리에서 합성되었다. 프로토콜 변환기는 기존의 플로우 컨트롤러를 대체한다. 기존 3 비트의 페이즈 신호가 4 비트로 확장되었으며, 두 개의 주소 변환기와 요청기/라우터로 변경되었다. 이러한 변화는 기본적으로 면적증가를 야기 시킬 수 있지만, 프로토콜 변환기의 면적 감소와 기존 주소 변환기(address converter)의 개선된 정책을 사용함으로서 전체 면적은 감소하였다.

표 3에서는 기존 SNP와 XSNP용 스위치 래퍼의 면적비교를 보여준다. 플로우 컨트롤러와 프로토콜 변환기는 SNA 시스템에 AMBA AHB IP를 수용하기 위해 사용된다. 표 3에서 합성된 조건은 하나의 스위치 래퍼에 두 개의 플로우 컨트롤러 또는 프로토콜 변환기가 결합되었을 때의 조건이다. 또한 기존의 플로우 컨트롤러가 스위치 래퍼에 포함되어 SNP IP와 AHB IP에 따라 서로 다른 스위치 래퍼를 사용해야 했던 것과는 달리 AHB-to-XSNP 변환기는 스위치 래퍼와 분리되어 AHB IP가 연결되는 부분에만 사용하면 되어 설계가 쉬워졌다. XSNP를 사용함으로서 거의 85%의 면적 감소 효과를 볼 수 있으며, 잠복기가 감소하였기 때문에 전체 성능 개선 효과도 기대할 수 있다.

표 3 기존 SNP와 XSNP를 위한 AMBA 호환용 스위치 래퍼의 면적 비교

Target technology	Original SNP (1 Switch wrapper with 2 Flow controller)	XSNP (1 Switch wrapper + 2 Converters)
Xilinx Virtex 800 (FPGA)	~ 1,200 LUT	~ 775 LUT
CMOS 0.35 um (@100MHz)	~ 13,500 gates	~ 1,900 gates

6. 결 론

본 논문에서 고성능 SoC 버스 시스템인 SNA/SNP에서 AMBA 기반의 IP를 성능 저하없이 사용할 수 있는 XSNP 프로토콜을 제안하고 이를 이용한 AHB-to-XSNP 프로토콜 변환기를 설계하였다. 제안된 XSNP는 AHB 버스트 전송에서 기존 SNP를 이용하여 AHB 신호로 변환할 때 발생하는 여러 잠복기와 플로우 컨트롤러에 포함되었던 큰 면적의 버퍼를 제거할 수 있었다. XSNP는 기존 SNP와 완벽하게 호환가능하며 기존 SNP 인터페이스를 가지는 SNA 시스템에 전혀 영향을 주지 않는 특징을 가진다. 따라서 XSNP를 사용함으로서 SNA 시스템은 SNP와 AMBA 기반의 IP를 모두 사용할 수 있다. 설계된 AHB-to-XSNP 프로토콜 변환기는 기존의 SNP 기반 변환기 면적의 15% 정도로

감소하고 프로토콜 변환에 따른 성능 저하는 없다. XSNP 기반의 SNA 시스템을 기존의 AMBA AHB 기반의 시스템에 적용하면 시스템 재설계 없이 AHB 대비 약 40% 정도 감소한 연결선 수만으로 동작이 가능하고 다중 채널 제공에 따른 성능 향상까지 기대할 수 있다.

참 고 문 헌

[1] Inside the New Computer Industry, issue 138, Jan 2001.

[2] F. Moraes, et al. "Hermes: an Infrastructure for Low Area Overhead Packet-switching Networks on Chip," Integration the VLSI Journal, 38(1), Oct. 2004, pp. 69-93.

[3] P. Gurrier, A. Greiner, "A Generic Architecture for On-Chip Packet-Switched Interconnections," Proceedings of the conference on Design, Automation and Test in Europe, Paris, France, pp. 250-256, 2000.

[4] ARM, "AMBA Specification, Revision 2.0," 1999.

[5] W. Peterson, "WISHBONE SoC Architecture Specification, Revision B.2," Silicore Corporation, 2001.

[6] IBM, "CoreConnect Bus Architecture," 1999.

[7] Sanghun Lee, Chanho Lee, Hyuk-jae Lee, "A New Multi-Channel On-Chip-Bus Architecture for System-On-Chips," IEEE-SoC Conference, Santa Clara, CA., pp. 305-308, September 2004.

[8] 이재성, 이혁재, 이찬호, "SNP: 시스템 온 칩을 위한 새로운 통신 프로토콜", 정보과학회논문지, 시스템 및 이론 제32권 제9호, pp. 465-474, 2005.10.

[9] ARM, Technical Support FAQs [Online], Available: http://www.arm.com/support/faqip/577.html

이 찬 호

1987년 서울대학교 전자공학과(학사). 1989년 서울대학교 전자공학과(석사). 1994년 UCLA, 전자공학과(박사). 1994년~1995년 삼성전자 반도체연구소 선임연구원 1995년~현재 숭실대학교 정보통신전자공학부 부교수. 관심분야는 SoC on-chip-network, 3D 그래픽 프로세서 설계, 채널코덱의 구현, SoC 설계방법론, H.264 codec 구현

이 상 헌

2003년 숭실대학교 전자공학과(학사). 2005년 숭실대학교 전자공학과(석사). 2005년~현재 숭실대학교 전자공학과 박사과정. 관심분야는 SoC 설계 방법론, On-Chip-Network/SoC Bus 연구, 3D 그래픽 프로세서 설계

김 응 섭

2004년 숭실대학교 정보통신 전자공학부(학사). 2006년 숭실대학교 전자공학과(석사). 2006년~현재 서울대학교 전자공학과 박사과정. 관심분야는 SoC 설계 방법론, 멀티미디어 프로세서 설계

이 혁 재

1987년 서울대학교 전자공학과(학사). 1989년 서울대학교 전자공학과(석사). 1996년 미국 퍼듀대학교 전기컴퓨터공학과(박사). 1996년~1998년 루이지애나 공과대학 컴퓨터공학과 조교수. 1998년~2001년 인텔 선임연구원. 2001년~현재 서울대학교 전기컴퓨터공학부 부교수. 관심분야는 컴퓨터 아키텍처 및 멀티미디어용 SOC 설계