SNP: 시스템 온 칩을 위한 새로운 통신 프로토콜
(SNP: A New On-Chip Communication Protocol for SoC)

이 재 성 † 이 혁 재 †† 이 찬 호 †††
(Jaesung Lee) (Hyuk-Jae Lee) (Chanho Lee)

요 약 고집적 SoC 설계시에 버스방식의 온칩 통신은 대역폭이 제한되는 문제점이 있고 NoC (Network-on-Chip) 방식에서는 구현의 복잡도가 증가하는 문제점이 있다. 본 논문에서는 이러한 문제점을 극복하는 새로운 온칩 통신 규격인 SNP(Soc Network Protocol)를 소개한다. SNP는 기존 버스의 신호선들을 세 가지 그룹인 제어(control), 주소(address), 데이타(data)로 나눈 뒤 하나의 채널을 통해 전송함으로써 신호선의 수를 줄인다. SNP 채널은 대칭구조로 사용되기 때문에 마스터-슬레이브 통신 방식뿐만 아니라 마스터-마스터 통신도 효율적으로 지원한다. 하나의 전송에 필요한 신호 그룹의 진행 규칙을 SNP 규격으로 정의하고, 동일한 정보가 반복적으로 전달되는 것을 방지하는 페이즈 복원 기능을 제안하여 통신대역을 효율적으로 사용할 수 있도록 한다. 산업계 표준 규격인 AMBA AHB와 비교한 결과 멀티미디어 타입의 데이타 전송시에 54%의 신호선수만으로도 대등한 대역폭을 지원할 수 있음을 보인다.

키워드 : 시스템 온 칩, 멀티미디어, 온칩 연결 구조, 통신 프로토콜, 버스 구조

Abstract For high density SoC design, on-chip communication based on bus interconnection encounters bandwidth limitation while an NoC(Network-on-Chip) approach suffers from unacceptable complexity in its implementation. This paper introduces a new on-chip communication protocol, SNP (SoC Network Protocol) to overcome these problems. In SNP, conventional on-chip bus signals are categorized into three groups, control, address, and data and only one set of wires is used to transmit all three groups of signals, resulting in the dramatic decrease of the number of wires. SNP efficiently supports master-master communication as well as master-slave communication with symmetric channels. A sequencing rule of signal groups is defined as a part of SNP specification and a phase-restoration feature is proposed to avoid redundant signals transmitted repeatedly over back-to-back transactions. Simulation results show that SNP provides about the same bandwidth with only 54% of wires when compared with AMBA AHB.

Key words : System-on-Chip, multimedia, on-chip interconnection, communication protocol, bus architecture

1. 서 론

최근 나노 시대의 도래와 더불어 하나의 반도체 칩안에 집적시킬 수 있는 하드웨어 IP(Intellectual Property) 수가 급격히 증가함에 따라 이들 상호간 효율적인 연결 방법을 제공하기 위한 연구가 널리 이뤄지고 있다. 버스 구조를 사용하여 칩 내부 IP들간의 연결을 하기위한 많은 온칩 버스들이 제안되었다. 대표적인 온칩 버스들인 CoreFrame, CoreConnect, AMBA, Wishbone, Super-Hyway, OCP 및 VSI 등이 있다[1-7]. 이들 기존의 온칩 버스들은 모두 프로토콜을 간단하게 하고, 통신 용량을 최대화 하기위한 노력을 기울인 반면 신호선을 줄이는 노력은 기울이지 않고 있다. 그 결과 기존 온칩 버스 기반 연결 방법은 소수의 IP들을 비교적 저렴한 비용으로 연결하는 경우에 널리 사용되는 반면 수십 개의 IP가 하나의 단일 칩에 집적되는 향후 SoC(System-on-Chip)에서 기존 버스는 신호선의 급격한 증가를 초래해 칩의 크기가 증가하고, 초미세(deep sub-micron) 공정의 최대 이슈인 신호선간 cross-talk 노이즈와 capacitance에 따른 신호 지연 문제가 발생한다[1,8-11]. 따라

· 본 연구는 한국과학재단 목적기초연구(R01-2003-000-10582-0)지원으로 수행되었음
† 학생회원 : 서울대학교 전기컴퓨터공학부
jslee@capp.snu.ac.kr
†† 정 회 원 : 서울대학교 전기컴퓨터공학부 교수
hjlee@capp.snu.ac.kr
††† 정 회 원 : 숭실대학교 정보통신전자공학부 교수
: chlee@ssu.ac.kr
논문접수 2005년 4월 20일
심사완료 2005년 7월 15일

서, 온칩 버스 신호선들의 숫자를 줄일 필요성이 대두되고 있으나, 아직까지 프로토콜의 관점에서 획기적으로 신호선 수를 줄이고자 하는 노력은 제시되지 않고 있다.

기존 오프칩(off-chip) 버스의 경우 주소와 데이타 버스등을 공유해 사용하는 PCI나 제어신호까지 함께 공유해 사용하는 시리얼 버스인 I^2C, SPI 등 신호선 수를 줄이기 위한 프로토콜들이 개발되었다[12-14]. 그러나, 이들은 PCB 기판위의 신호선을 줄이고자 tri-state 버퍼를 사용하거나 혹은 양방향 신호를 많이 사용한다. 하지만, 온칩 버스의 경우는 tri-state 버퍼를 사용하는 것이 로직 설계시 검증이 까다로워서 지양하고 있으며 또한 양방향 신호의 사용도 제한된다. 또한, 시리얼 버스처럼 극단적으로 신호선 수를 줄이는 경우도 대역용량에 제한이 발생하여 칩 내부에서는 사용하기가 어렵다. 따라서, 온칩 버스에서 신호선 수를 줄이기 위해서는 기존 오프칩 버스에서 사용하는 것과는 다른 새로운 방법을 사용할 필요성이 있다.

최근 개별 컴퓨터 또는 컴퓨팅 노드간 네트워킹을 위해 사용되어 온 개념을 칩 내부 IP간 연결용으로 간략화하여 구현한 NoC(Network-on-Chip) 기술이 제안되었다[15,16]. 이 기술은 폭넓은 대역폭을 제공하고 기존 버스에 비해 훨씬 적은 신호선을 사용하기 때문에 주목을 받고 있다. 또한, 고속 직렬 통신 기술[17]과 결합하여 신호선 문제에 따른 통신 장애를 극복할 수 있는 기술로서도 각광 받고 있다. 그러나 패킷 기반의 복잡한 프로토콜과 스위칭 노드의 스토어 앤 포워딩 매커니즘 및 각 링크에서의 흐름 제어 방법은 NoC를 구성하는 컴포넌트의 크기를 증가 시킬 뿐만 아니라 패킷 전달 지연을 증가시키고, 통신 시간을 예측하기 어렵게 된다. 더욱이 큰 버퍼가 각 링크마다 준비되어져야 하고[15,18,19] 직렬 통신의 경우 큰 면적을 차지하는 전용 아날로그 회로가 추가되어야 하므로[20] 각각의 IP 마다 모두 NoC 인터페이스를 통하여 연결한다는 것은 무리이다. 특히, 모바일 기기가 주 응용분야인 멀티미디어 시스템 온 칩은 전력 소모, 비용등의 측면에서 매우 민감하기 때문에 그러한 네트워크 기반 연결 구조는 적합하지 못하다고 할 수 있다[21,22].

기존의 버스방식이나 새로운 NoC 방식에서 드러나는 문제점을 극복하고자 본 논문에서 새로운 온칩 통신 규격인 SNP(Soc Network Protocol)를 제안하다. SNP는 앞서 언급된 기존 방식들에 있어 최대한 장점들을 취하고 단점들을 배제하는 방향으로 개발되었다. 특히, SNP는 온칩 환경이라는 제약 조건하에서 신호선을 줄이는 방식으로써 기존의 오프칩 버스들의 장점을 최대한 활용하면서도 온칩 버스로서의 역할을 극대화하기 위하여 기존 버스들보다 훨씬 적은 신호선을 사용하여 대역폭이 증가하더라도 신호선의 급격한 증가를 방지하도록 한다. 또한, 프로토콜을 가능한 단순하게 함으로써 프로토콜 처리를 간단하게 하여 통신 노드간 전송시간이나 통신 노드의 복잡도 증가를 기존 온칩 버스 수준으로 유지한다. 또한, 새로운 IP 설계와 IP의 재사용이 용이하도록 하며 전체 개발 비용도 버스기반 설계 수준의 저비용에 맞출 수 있도록 하였으며 향후 NoC 기반의 상위 수준 연결시에도 지역 연결구조로서 상위연결구조와 이음매 없이 연결될 수 있도록 대칭 쌍방향 채널기반 아키텍쳐를 제공한다.

본 논문의 구성은 다음과 같다. 2장을 통하여 SNP의 신호체계를 소개하고, 3장에서는 SNP에서의 정보 전달 순서를 설명한다. 4장에서는 기존 산업계 표준 버스인 AMBA의 AHB 및 AXI를 대상으로 다양한 종류의 트래픽 시뮬레이션을 수행하여 SNP와 비교한다. 마지막으로 5장에서 결론을 맺는다.

2. SNP(SoC Network Protocol)

2.1 신호 체계

SNP 개발의 핵심 동기 가운데 하나는 IP의 증가에 따라 늘어나는 신호선 수에 따른 라우팅 부담을 줄이고자 하는 것이다. SoC를 구현할 때 기존 온칩 버스 아키텍쳐가 직면하게 되는 문제점은 많은 수의 IP에 원하는 만큼의 대역폭을 제공하지 못할 뿐만 아니라 대역폭을 만족시키려면 추가되어야 하는 신호선수가 급격히 늘어난다는 데 있다. 예를 들어 산업계 표준인 AMBA AHB의 경우 41개의 제어, 32 비트의 주소, 32 비트의 읽기 데이타, 32 비트의 쓰기 데이타 신호들이 각각 독립적으로 존재하기 때문에 하나의 IP마다 갖춰야할 신호선수가 137개에 이르게 된다. 그러나 자세히 살펴보면 각각의 신호 그룹이 독립적으로 존재해야하는지에 대해서는 의문을 가지게 된다. 더욱이 멀티미디어 데이타나 멀티프로세서간의 데이타 전송은 대부분의 경우 버스트 전송이기 때문에 주소 정보와 제어 정보는 통신 초기에만 보내면 되는 경우가 많기 때문이다.

본 논문에서 제안하는 SNP는 기존 버스의 신호선들을 세가지 신호 그룹인 제어(control), 주소(address), 데이타(data) 그룹으로 나눈 뒤 하나의 32 비트 신호선들을 통해 전송한다. 이 신호선들을 채널이라 칭하며 CHANNEL[31:0]로 표기한다. 하나의 채널에 서로 다른 종류의 신호들을 전송하게 되므로 신호의 종류를 구별하기 위해서 페이즈라고 명시한 3 비트 신호 PHASE[2:0]가 추가된다. 표 1은 채널에 전송되는 신호의 종류를 나타내고 있다. PHASE[2:0] 신호의 값에 따라서 8가지의 서로 다른 정보가 채널을 통해 전송된다. 이들 8가지 정보는 현재 채널에 정보가 실리지 않은 상태를

나타내는 IDLE(idle), 읽고 쓰기에 대한 슬레이브의 응답이 실렸음을 나타내는 RP(Response), 읽기 데이타가 실렸음을 나타내는 RD(Read Data), 인터럽트 등 특수 목적 또는 향후 추가되는 사양들에 대한 예약용 페이즈로 사용되는 SP(Special), 제어 정보가 실렸음을 나타내는 CO(Control), 쓰기 데이타에 대한 주소 정보가 실렸음을 나타내는 WA(Write Address), 쓰기 데이타가 실렸음을 나타내는 WD(Write Data), 읽기 데이타에 대한 주소 정보가 실렸음을 나타내는 RA(Read Address)이다. 각 페이즈에서 CHANNEL[31:0]에 전송되는 각 신호들의 정의는 참고문헌 [23]에 자세히 설명되어 있고, 본 논문에서는 지면관계상 생략한다.

표 1 채널에 전송되는 신호의 종류

페이즈 종류	PHASE[2:0] 값	전송되는 신호의 종류
IDLE	000	No information
RP	001	Response
RD	010	Read Data
SP	011	Special Transaction
CO	100	Control information
WA	101	Write Address
WD	110	Write Data
RA	111	Read Address

그림 1은 송신자(sender)에서 수신자(receiver)로 데이타가 전송될 때 사용되는 기본 SNP 신호의 결선을 나타낸다. 이미 설명한 CHANNEL[31:0]와 PHASE[2:0]이외에 VALID 및 READY 신호가 추가되어 총 37개의 신호로 하나의 SNP 채널이 구성된다. VALID는 CHANNEL[31:0] 및 PHASE[2:0]에 유효한 신호들이 전송됨을 나타낸다. READY 신호를 제외한 모든 신호들은 모두 같은 방향을 향하며 READY는 전송방향이 유일하게 수신자 쪽에서 송신자 방향으로 가는 신호로서 수신자가 신호를 받아들일 상태가 됨을 의미한다.

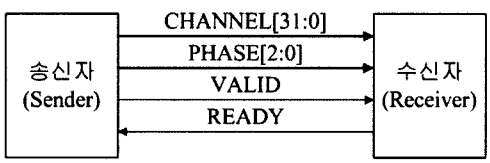


그림 1 SNP 신호선

그림 2는 SNP 신호선들을 이용하여 통신을 수행하는 예를 보여준다. 채널의 초기 상태는 IDLE에서 시작한다. 먼저 싸이클 1에서 송신자는 일련의 데이타를 보내는 하나의 전송과정(transaction)을 진행하기 위하여 데이타가 저장될 0123h이라는 32비트의 16진 주소를 전송한다. 이때 페이즈 신호는 WA(Write Address)를 나타내며 VALID신호를 활성화하여 채널의 현재 상태가 액티브 상태 즉, 유효한 정보가 전달됨을 알린다. 수신자는 READY신호를 활성화하여 수신준비가 됐음을 알린다. 이렇게 하여 하나의 페이즈가 전송되면 그 다음의 CO 페이즈에서 INCR4라는 제어 정보를 보낸다. 여기서 INCR4는 4개의 데이타가 전송되며 어드레스는 연속적으로 증가한다는 것을 나타내는 제어정보이다. 싸이클 3부터 싸이클 9까지 4개의 32비트데이타가 전송된다. 싸이클 3에서부터 채널의 페이즈는 WD상태가 되어 쓰기 데이타 전송 페이즈임을 알린다. 싸이클 4에서처럼 수신단이 준비상태가 안되었을 때는 READY신호를 이용하여 전송 제어가 가능하며 송신자 입장에서도 싸이클 6에서처럼 VALID 신호를 비활성화(disable)하여 전송 제어가 가능하다. 그림의 각 부분에 빗금 친 신호값들은 무의미한 값임을 나타내는 표시이다.

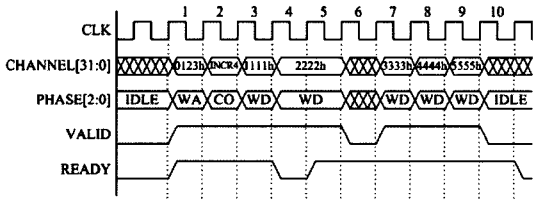


그림 2 SNP 통신 타이밍도의 예

SNP 신호들이 READY이외에는 모두 동일 방향을 향하는 점은 향후 대용량 SoC에서의 사용을 고려한 결과이다. 왜냐하면 하드웨어 IP간 전송시간이 동작 주파수 한 싸이클을 초과하는 경우 IP들 사이에 staging register를 삽입하는 register slicing 또는 interconnection serialization 기법이 필요한데 그 기법들은 이러한 동일 방향 채널에서만 가능하게 하기 때문이다.

2.2 마스터-슬레이브 연결

기존의 SoC에서 널리 사용되는 통신 방식은 마스터-슬레이브 통신 방식이다. 즉, 마스터에서 하나의 전송을 시작하여 슬레이브에게 보내면, 슬레이브가 그에 대한 동작을 수행하고 응답을 마스터에게 보냄으로써 그 전송을 마무리하는 방식이다. 대표적인 예로서는 프로세서(마스터)가 메모리(슬레이브)에 데이타 읽기(혹은 쓰기)를 위한 전송을 시작하면(즉, 요청 (request)을 메모리에 보내면) 메모리가 그 데이타를 보내고(혹은 저장하고) 정확한 데이타를 보냈다는(혹은 저장되었다는) 응답 신호를 프로세서에게 보낸다. 이러한 마스터-슬레이브 통신을 지금까지 설명된 SNP 채널 두 개를 이용하여 구현할 수 있다. 그림 3은 SNP를 이용한 마스터-슬레이브 통신 채널 결선을 보여준다. 그림에서 상위 채널은

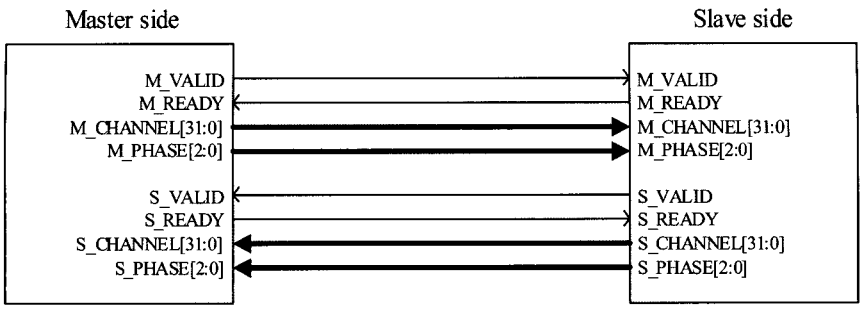
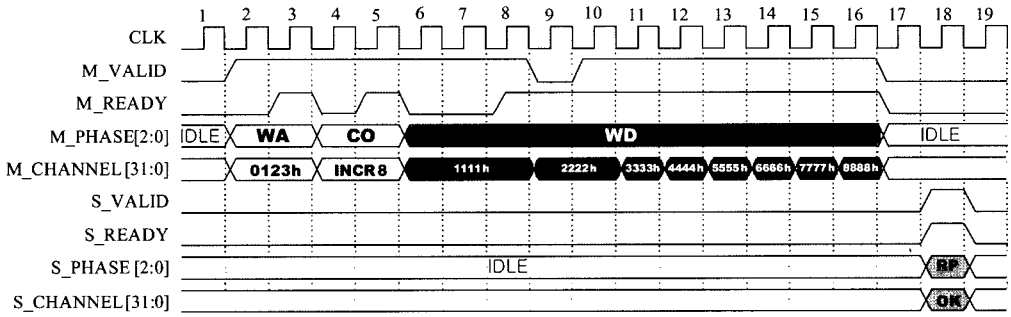


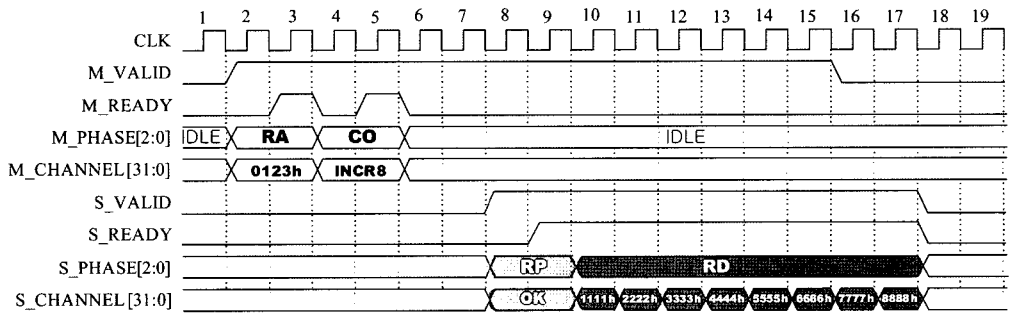
그림 3 SNP를 사용한 마스터-슬레이브 연결

마스터로 부터 슬레이브로의 신호 전송을 위한 채널이고 하위 채널은 슬레이브에서 마스터로의 신호 전송을 위한 채널이다. 상하위 채널 모두 37개의 동일한 신호로 연결된다. 단지 접두어로 M과 S를 붙여 구분한 것은 그 신호선의 송신자(마스터인지 슬레이브인지)를 표시하기 위한 것이다. 먼저 상위 채널의 M_CHANNEL에는 주소, 쓰기 데이타와 제어 정보가 전송될 수 있고, 하위 채널인 S_CHANNEL은 읽기 데이타와 응답 정보가 슬레이브에서 마스터로 전송될 수 있다.

그림 4는 마스터-슬레이브 연결을 사용하여 기본적인 쓰기 및 읽기에 대한 타이밍도 예제를 나타내고 있다. 먼저 그림 4(a) 경우 마스터가 쓰기 전송을 수행하기 위해 싸이클 2에서 현재 페이즈 상태를 IDLE에서 WA로 변화시킴으로써 채널에 쓰기 데이타에 대한 주소(0123h)가 실렸음을 알린다. 싸이클 3에서 슬레이브는 M_READY 신호를 활성화함으로써 그 주소 정보를 수신하겠다는 표시를 한다. 같은 방법으로 제어 정보 INCR8(8개의 데이타를 주소를 증가시키며 전송)를 싸이클 4에서 보낸 뒤 싸이클 6에서부터 보내고자하는 쓰기 데이타를 보내게 된다. 각각의 페이즈에서 VALID와 READY 신호는 변함없이 핸드쉐이킹(handshaking)을 통하여 정보 전달을 제어한다. 싸이클 16에 이르면 슬레



(a) 쓰기



(b) 읽기

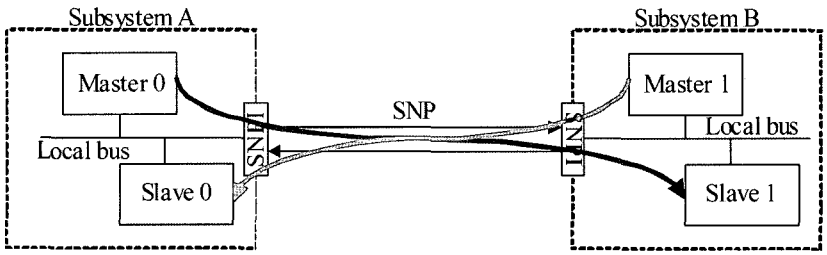
그림 4 마스터-슬레이브 통신 타이밍도의 예

이브는 마스터가 보내고자하는 모든 데이타를 수신하게 된다. 슬레이브는 쓰기 결과에 대한 응답을 마스터에게 보낼 수 있으며 그림의 경우 싸이클 18에서 RP 페이즈를 통하여 쓰기가 잘 됐음을 나타내는 OK로서 응답을 하고 있다. 그림 4(b)에서는 마스터가 데이타 읽기를 수행하고 있다. 싸이클 2에서 마스터는 페이즈를 IDLE에서 RA로 바꿈으로써 읽기의 시작을 알리고 이에 관련된 제어 정보를 보내면 슬레이브는 싸이클 9에서 그에 대한 응답을 보내며, 싸이클 10에서 17을 거쳐 읽기 데이타를 보낸다.

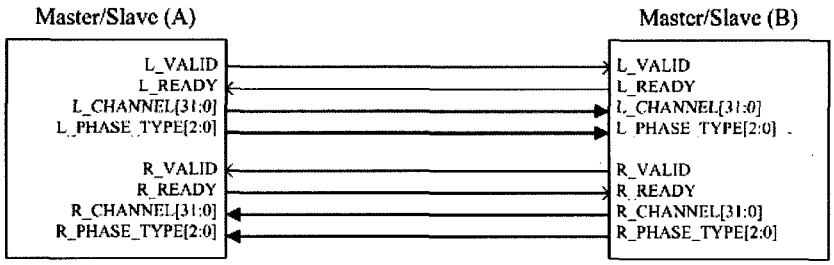
2.3 마스터-마스터 연결

그림 5(a)는 두개의 서브시스템 A와 B로 구성된 시스템에서 각각의 써브시스템이 모두 마스터와 슬레이브를 포함하는 경우를 보여준다. 서브시스템 A의 마스터 0과 서브시스템 B의 슬레이브 1간의 통신과 서브시스템 B의 마스터 1과 서브시스템 A의 슬레이브 0간의 통신이 동시에 필요한 경우 기존의 버스 구조를 이용하여 연결하려면 마스터-슬레이브 방식의 인터페이스가 두 쌍이 필요하게 된다. 이러한 시스템을 SNP를 사용하면 그림 5의 (b)와 같이 연결된다. 그림에서도 알 수 있듯이 그림 4의 마스터-슬레이브간 통신시 필요로 하는 결선만으로도 구현이 가능함을 알 수 있다. 이는 하나의 SNP채널로 마스터와 슬레이브 정보를 번갈아 보낼 수가 있기 때문에 가능하다. 이 그림에서 상위채널과 하위채널을 구별하기 위해서 상위 채널의 신호는 접두어 'L_'로 시작하고, 하위 채널의 신호들은 'R_'로 시작하도록 신호들의 이름을 표기하였다.

그림 6은 마스터-마스터간 통신을 예시하고 있다. 그림에서 L 채널을 통해 하나의 마스터가 싸이클 1에서 6에 걸쳐 데이타 쓰기를 진행하고 싸이클 9에서 R 채널을 통해 해당 슬레이브로부터 응답을 받는다. 이와 동시에 R 채널에서 또 다른 마스터가 싸이클 3에서 8에 걸쳐 또 다른 쓰기를 진행하고 그에 해당하는 슬레이브가 싸이클 9를 통해 응답을 수행한다. 읽기의 경우도 마찬가지로 싸이클 10에서 11에 걸쳐 쓰기 요청을 하는 동안 상대편의 또 다른 마스터는 같은 싸이클 기간 동안



(a) 마스터-마스터 통신을 필요로하는 시스템 구성 예



(b) 마스터-마스터 연결

그림 5 SNP를 사용한 마스터-마스터 통신

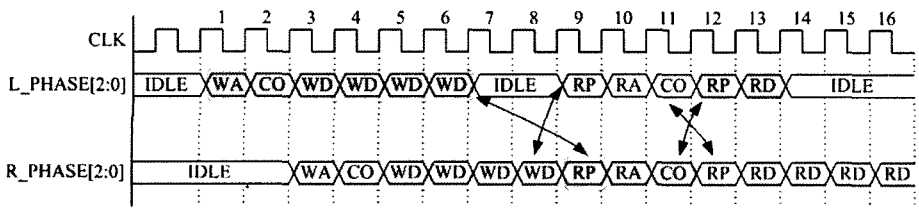


그림 6 마스터-마스터 통신의 타이밍도의 예

다른 쓰기 요청을 수행할 수가 있다. 그림에서 보듯이 L채널과 R채널이 서로 다른 전송을 동시에 수행할 수 있으므로 매우 효율적인 마스터간 통신이 이루어짐을 알 수 있다.

3. 페이즈 진행 규칙 및 최적화

본 절에서는 SNP 채널을 통한 전송에서 자주 사용되는 페이즈들의 진행 순서를 알아보고, 이를 바탕으로 가능한 페이즈 진행 순서를 결정한다. 그리고, 페이즈 진행 순서를 제한하고 이를 효과적으로 이용함으로써 전송시간을 줄이는 방법을 살펴본다.

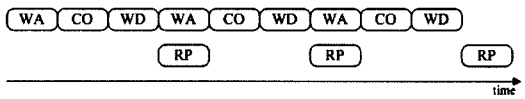
3.1 페이즈 진행 규칙(phase sequencing rule)

SNP 채널이 수행하는 역할은 송신자로서 정보를 전달하거나 수신자로서 응답을 보내는 두 전송과정만이 존재한다. 각 전송과정(transaction)이 시작하여 그 과정이 끝날 때까지 중간에 IDLE 상태가 발생하면 안된다. 즉, 초기 IDLE에서 시작하여 다양한 페이즈 변화를 통해 전달하고자하는 내용을 전송하고 나면 다시 IDLE 상태에 머무르게 되고 이때 하나의 전송과정이 완료됐다고 말할 수 있다. 표 2는 SNP를 통해서 발생되는 전송과정의 종류 및 각 전송과정을 구성하는 페이즈들의 조합을 보여준다. SNP의 전송과정은 크게 쓰기, 읽기 및 특수(special) 전송으로 나뉜다. 하나의 쓰기 전송과정(또는 읽기 전송과정)은 각각 요청 부전송과정(sub-transaction)과 응답 부전송과정으로 구성된다. 표 2에서 보는 바와 같이 쓰기 요청 부전송과정은 쓰기 주소를 먼저 전송하고(WA 페이즈), 제어 정보를 다음으로 전송하며(CO 페이즈) 마지막으로 데이타를 전송한다(WD 페이즈). 필요에 따라 데이타는 여러 번 전송할 수 있으며, 이는 기호 [WD]+로 표시된다. 쓰기 응답 부전송과정의 경우 하나의 RP 페이즈로 구성된다. 읽기 요청 부전송과정의 경우도 유사한 방법으로 읽기 주소를 먼저 전송하고(RA 페이즈), 제어 정보를 다음으로 전송한다(CO 페이즈). 읽기 응답 부전송과정의 경우 RP를 통해 응답 상태를 먼저 알려주고 이어서 해당 주소의 데이타를 보내준다(RD 페이즈). 마찬가지로 [RD]+는 읽기 데이타를 여러 번 전송 가능하다는 것을 나타낸다. 특수 전송과정은 하나의 SP 페이즈로만 구성된다. SNP에서 정확한 정보 전달을 위해서는 페이즈의 순서가 지켜져야 한다.

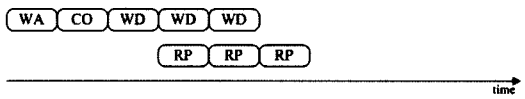
3.2 페이즈 복원 기능(phase-restoration)

SNP는 하나의 채널을 통해 어드레스, 제어 정보 및 데이타를 모두 보내기 때문에 전송과정당 필요한 싸이클 수가 증가하게 된다. 연속되는 전송과정간 중복되는 정보를 보내지 않음으로써 기존 버스 전송방식에서 요구되는 싸이클 이외의 추가 싸이클 수를 극소화할 수 있다. 간단한 예를 도시하면 그림 7과 같다. 그림 7(a)는 데이타를 연속적으로 세 번 전송하는 SNP의 페이즈 진행을 나타내고 있다. 단일 쓰기 데이타 전송이 세 번 연속된다고 가정할 때 각 전송은 WA, CO 및 WD의 세 페이즈로 구성된다. 여기에서 WA의 경우 연속되는 데이타는 대부분 주소가 전송되는 데이타 수만큼 자동 증가되어도 무방하므로 첫 전송이후의 각 전송들에 대해서는 WA 페이즈를 생략할 수가 있다. 제어 정보 역시 반복되는 경우가 대부분이므로 굳이 CO 페이즈를 매번 보낼 필요가 없다. 그러한 이유로 불필요한 페이즈를 보내지 않고 WD만 보내게 되면 그림 7(b)와 같이 줄어들게 된다.

그림 7(b)와 같이 전송하는 경우 표 2의 페이즈 진행 규칙을 위반하게 된다. 왜냐하면, 하나의 쓰기 부전송은 WA 페이즈 및 CO 페이즈를 요구하는데, 그림 7(b)에서는 두 번째 및 세 번째 전송에서 WA 및 CO 페이즈를 생략했기 때문이다. 본 논문에서는 이렇게 생략된 페이즈를 처리해주기 위해서 '페이즈 복원(phase-restoration)' 기법을 제안한다. 이 기법이 적용되는 경우 슬레이브는 WA와 CO 정보를 저장하고 있다가 페이즈 순서 규칙에서 생략된 부분이 발생하면 저장하고 있던 가



(a) 주소 및 제어정보를 반복하여 전송하는 경우



(b) 주소 및 제어 정보를 생략하고 전송하는 경우

그림 7 연속적인 3회 쓰기 전송과정의 페이즈 진행

표 2 SNP의 전송과정 페이즈 진행 순서

Transaction	Sub-transaction	Phase sequence	Comments
Write	Request	WA-CO-[WD]+	[WD]+는 하나 이상의 WD 페이즈를 의미함
	Response	RP	·
Read	Request	RA-CO	·
	Response	RP-[RD]+	[RD]+는 하나 이상의 RD 페이즈를 의미함
Special	·	SP	·

장 최근 정보로 생략된 페이즈를 복원하여 전송을 정상화한다. 이렇게 페이즈 복원 기법을 자주 이용하는 경우 전송 시간을 많이 줄일 수가 있다. 특히, 멀티미디어 데이타 처리나 프로세서간 캐쉬 블록 전송같은 경우 페이즈 복원을 자주 이용할 수 있다. 따라서, 이러한 응용에서는 다량의 신호선을 사용하는 기존 버스 통신 방식과 비교하여도 SNP의 통신 방식은 유사한 전송 시간을 가질 수 있게 된다.

표 2의 SNP 페이즈 진행 규칙에 페이즈 복원 기능을 적용시키면 다음의 표 3과 같이 수정된다. 기본적인 내용은 표의 2와 같으나 기호 { }로 표기된 페이즈들은 생략 가능함을 나타낸다. 예를 들어 {WA}-{CO}-[WD]+는 WA 페이즈 및 CO 페이즈가 생략가능함을 나타낸다. 이때 만약 주소가 매번 증가하는 경우 CO 페이즈를 사용하여 이러한 정보를 슬레이브에 전달하면, 생략된 페이즈의 정보는 가장 최근에 진행된 전송과정의 동일 페이즈로부터 복원한다. 따라서, 슬레이브는 항상 가장 최근의 WA, CO, RA 정보들을 저장하는 레지스터를 가지고 있어야 하며 마스터는 가장 최근의 RP 정보를 저장하는 레지스터를 가지고 있어야 한다. 예를 들어 WA-CO-WD-RP의 쓰기 전송과정이 완료된후에 연속적인 주소로 데이타를 쓰기 원하는 경우 WA, CO를 생략하고 WD만 여러 번 보내어도 가장 최근에 저장된 WA와 CO 정보는 슬레이브쪽에서 저장된 상태로 각 전송과정마다 복원되어 사용된다. 이때 WA의 정보는 CO 정보가 제공하는 바에 따라 슬레이브 내에서 자체적으로 주소를 증가시키어 갱신되어 복원된다.

하나의 부전송과정에 IDLE 페이즈가 포함되지는 않는다. 즉, IDLE 페이즈는 하나의 부전송과정의 종료를 나타내는데 사용될 수 있다. IDLE 없이 하나의 부전송과정의 종료는 연이은 새로운 부전송과정의 시작으로 나타날 수도 있다. 또한, 데이타 페이즈의 길이를 미리 알고 있는 경우에는 정해진 길이만큼 데이타 페이즈가 진행되면 자동으로 종료됨을 알 수 있다.

4. SNP의 성능 분석

4.1 전송 성능 비교

본 장에서는 SNP 규격의 기능 모델(functional model)들을 구현하고 이를 이용하여 SNP 성능을 AMBA AHB 및 AXI와 비교한다. 원래 AXI는 버스 구조 규격이 아니고 IP 인터페이스 규격이지만 기존 버스 방식의 신호선들을 유지면서 읽기/쓰기가 동시에 가능한 특성이 SNP와 유사하므로 AXI를 그 비교 대상으로 정하였다. 그림 8(a)는 AMBA AHB, AXI 및 한쌍의 SNP 채널 기반의 전형적인 버스 아키텍처인 마스터-슬레이브 연결 구조를 구현한 뒤 하나의 마스터와 다른 하나의 슬레이브간에서 같은 양의 데이타를 전송하는데 걸리는 시간을 비교 한 것이다. 세로축은 AHB에 대한 AXI 및 SNP의 상대적 전송 시간을 나타낸다. 따라서, AHB는 항상 1이다. 가로축은 싱글 전송과정과 버스트 전송과정의 배합 비율을 나타낸다. 여기서 싱글 전송과정이란 단 하나의 데이타 페이즈(RD 혹은 WD)만 존재하는 전송과정을 말하며 버스트 전송과정이란 여러개의 데이타 페이즈가 존재하는 경우를 말한다. 가로축 값이 0이면 항상 싱글 전송과정만이 존재함을 나타내며 1이면 항상 버스트 전송과정만이 존재하는 것을 의미하고 0.5인 경우는 각각 반반씩이 된다. 버스트 전송에서 버스트 전송 데이타의 크기는 64 bytes(영상처리 표준에서 널리 사용되는 8×8 macro block 단위의 데이타)이다. 본 시뮬레이션을 위한 시나리오는 MPEG4 영상 압축 및 복원 알고리즘을 기반으로 하고 있으며 전송 배합 비율이 1에 가까운 경우는 이미지 센서를 통한 영상 입력 데이타들을 DMA를 통하여 메모리에 저장하거나 저장된 데이타를 출력하는 입출력 동작의 경우와 DCT/IDCT(Discrete Cosine Transform/Inverse Discrete Cosine Transform), ME(Motion Estimation) 연산을 위한 경우이며 모두 8x8 블록단위 전송의 버스트전송 위주이다. 전송 배합 비율이 0에 가까운 경우는 주로 메인 컨트롤러인 ARM7 프로세서가 명령어를 인출하다가 분기하는 경우에 해당하나 전체 데이타 전송에서 차지하는 부분은 극히 미미하다. 그 밖에 싱글, 버스트가 혼합적으로 일어나는 경우는 VLC(Variable Length Coding)이나 MC(Motion Compensation) 연산의 경우처럼 하드웨어 모듈과 ARM7에서 동작하는 펌

표 3 페이즈 복원 기능을 사용한 SNP 페이즈 진행 순서

Transaction	Sub-transaction	Phase sequence	Comments
Write	Request	{WA}-{CO}-[WD]+	{WA}, {CO}는 WA, CO가 각각 생략될 수 있음을 의미함, [WD]+는 하나 이상의 WD 페이즈를 의미함
	Response	RP	·
Read	Request	{RA}-CO	{RA}는 RA가 생략될 수 있음을 의미함
	Response	{RP}-[RD]+	{RP}는 RP가 생략될 수 있음을 의미함, [RD]+는 하나 이상의 RD 페이즈를 의미함
Special	·	SP	·

웨어가 함께 각각의 연산 처리를 수행하는 경우에 해당한다. 덧붙여 본 시나리오에서 버스트 전송은 실제 다른 어플리케이션에서도 영상 데이타 블록 전송이나 내장 프로세서의 캐쉬 라인 전송의 형태로 매우 빈번히 발생하는 형태이며 싱글 전송은 마이크로 프로세서가 분기하거나 특정 레지스터들을 억세스하는 경우에 발생하는 형태이다. 모든 인터페이스는 32-bit 크기의 데이타 신호와 주소 신호들로 구성되었다.

그림 8(a)의 그래프를 보면 SNP가 AHB(신호선수 137개) 보다 신호선 수가 훨씬 적음에도 불구하고 오히려 성능이 좋음을 알 수 있다. 이는 AHB가 동시에 읽기/쓰기 데이타 전송을 허용하지 않는 반면, SNP는 쌍방향의 독립적 채널을 사용하여 읽기/쓰기 데이타 전송이 동시에 가능하기 때문이다. 따라서, 버스트 전송의 배합이 많을수록(즉, 가로축의 값이 1에 가까워질수록) 거의 두 배 가까이 차이가 나게 됨을 알 수 있다. AXI의 경우는 전송 종류에 상관없이 읽기/쓰기가 동시에 가능하므로 전반적으로 SNP보다 성능이 좋다. 하지만, 버스트 전송위주의 환경에서는, 즉 burst rate가 1에 가까운 경우에는 SNP의 성능이 AXI에 매우 근접함을 알 수 있다. 이는 AXI가 SNP의 74개의 신호선에 비해 훨씬 많은 110개의 신호선을 사용한다는 점을 고려하면 SNP가 높은 효율을 가짐을 보여주는 결과이다.

그림 8(b)는 각 신호선들이 얼마나 효율적으로 사용되는 지 보여준다. 이를 위하여 'wire 효율'을 정의하여 비교하였다. wire 효율이라 함은 하나의 버스 싸이클당 평균적으로 전송된 데이타양을 버스 신호수만큼으로 나눈 것으로서, 하나의 wire가 얼마나 낭비되지 않고 효율적으로 사용되고 있는가를 나타낸 척도가 되겠다. 그림 8(b)의 세로축은 wire 효율을 나타내며, 본 그래프도 역시 AHB를 기준으로 하여 상대적인 효율값을 나타낸 결과이다. 따라서, 기준이 되는 AHB는 항상 1을 가진다. 그림에서 보는 바와 같이 SNP가 효율이 더 좋음을 알 수 있다. 버스트 전송의 비중이 큰 환경에서 SNP는 기존 AHB 규격에 비하여 최대 3.5배까지 그리고, AXI 대비 1.4배까지 효율이 좋아짐을 발견할 수 있다.

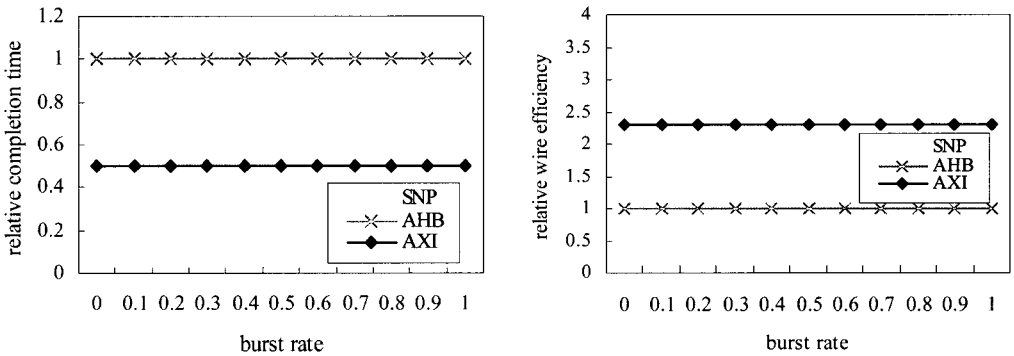
4.2 하드웨어 구현 복잡도 비교

본 절은 기존 버스기반의 IP들이 SNP 인터페이스를 위하여 추가적으로 필요로하는 하드웨어 부담에 대하여 알아본다. 기존의 버스 인터페이스는 아래 그림 9(a) 처럼 구성된다. 마스터 쪽 인터페이스에서는 제어(ctrl), 어드레스(addr), 쓰기 데이타(w_data) 버스가 순서대로 출력되며 슬레이브로 부터의 읽기 데이타(r_data)와 응답(resp)을 받기 위한 버스들이 입력된다. 유사 형태로서 슬레이브 쪽 인터페이스에서도 마스터와 그 5가지의 버스를 통해 연결된다.

반면, SNP의 경우 인터페이스 구조가 그림 9(b) 처럼 변형된다. 즉, 제어, 주소, 쓰기 데이타는 멀티플렉스되어 하나의 싱글 SNP 채널에 연결되고 그 3가지 신호 버스들을 페이즈 진행 규칙에 따라 중재하기 위한 제어 상태기가 필요하게 된다. 그러나, 전송 정보의 종류가 기존 버스의 그것과 같기 때문에 제어 상태기는 단순한 상태 머신(state machine)으로 구현이 가능하여 추가 설계 부담이 매우 작다. 마찬가지로 슬레이브 쪽에서도 응답 신호와 읽기 데이타 두가지를 중재할 수 있는 상태기 하나만 추가 되면 된다. 더욱이, 슬레이브 쪽 설계가 단순해질 수 있는 것은 phase 신호가 마스터로부터 전송되어 오기 때문에 그때마다의 phase에 따라 반응하도록 설계하면 되기 때문이다. 따라서, 기존 버스기반 IP들을 SNP 기반으로 이용하고자 하는 경우에도 추가 설계 부담이 크지 않다. 만약 IP가 설계 단계에서부터 SNP 기반으로 설계 되는 경우는 그 부담이 더 경감된다.

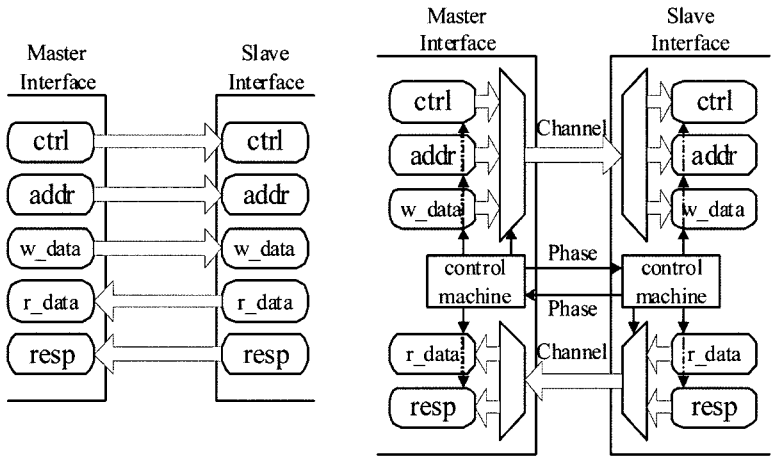
5. 결 론

본 논문에서는 SoC 설계시 주로 사용되어 오던 버스 기반 통신 방식을 대체할 수 있는 SNP라는 새로운 통



(a) 전송 시간 (b) wire 효율

그림 8 SNP와 AMBA AHB 및 AXI의 성능 비교



(a) 기존 버스 (b) SNP

그림 9 하드웨어 인터페이스 구성

신 규격을 제안하였다. 기존 산업계 표준인 AHB와 비교하면 버스트 전송 위주의 통신에서 54%의 신호선수로도 거의 유사한 성능을 나타냄을 알 수 있었다. 멀티미디어 응용이나 대용량 SoC의 backbone 버스에서 이러한 버스트 전송이 많이 이루어지므로 SNP는 멀티미디어 칩이나 고집적 SoC에서 효율적으로 사용될 수 있을 것으로 기대된다. 실제로 최근 SoC화 되고 있는 칩들이 주로 시스템 레벨의 DTV 칩셋이나 셋톱박스칩셋, 휴대용 멀티미디어 솔루션칩 등 멀티미디어 환경 즉, 버스트 전송위주의 어플리케이션이 주류이므로 이러한 가능성을 뒷받침해준다. 특히, 하나의 하드웨어 모듈이 마스터 및 슬레이브 기능을 동시에 수행하는 경우 SNP가 더욱 효율적으로 사용될 수 있다. 왜냐하면, 기존의 버스로 지원하려면 두 배의 신호수를 필요로 하기 때문이다. 대표적인 예로 DMA controller는 마스터/슬레이브의 역할을 동시에 수행하는 IP로서 어느 한 때에 마스터/슬레이브 둘 중 한 가지 모드로만 동작함에도 불구하고 두 쌍의 방대한 신호선들을 필요로 하게 된다. SNP 방식을 사용하면 단 한 쌍의 신호선만으로도 그러한 역할들을 모두 수용 할 수 있으므로 효율적인 연결이 가능하다. 또한, SNP는 IP가 전송지연이 커서 한 싸이클 이상 걸리는 대용량 SoC에도 적합하다. 왜냐하면, 하나의 채널을 구성하는 신호가 READY를 제외하고는 모두 같은 방향이기 때문에 register slicing 또는 interconnection serialization 기법을 적용할 수 있기 때문이다.

위에서 열거한 이러한 사실들로 보았을 때 SNP는 기존 방식의 한계를 극복하고 나아가 버스 방식의 연결구조와 네트워크 방식 연결구조 사이의 차이를 좁혀 향후 멀티미디어용 반도체 설계의 실질적 규격으로 발전할 수 있을 것으로 기대된다. 본 논문에서 소개한 내용이외에도 채널 최적화 모드(Reduced Channel Mode), 가상채널(Virtual Channel), 계층간 독립적으로 분리된 정의 등 여러 유용한 SNP 부가 기능들이 있으며 이들 기능은 참고문헌 [23]에 자세히 설명되어 있다. 향후에는 보다 실질적인 성능 비교를 위해 AMBA AHB기반으로 제작된 MPEG-4 코덱을 SNP기반으로 개정하는 연구가 진행중이며 SNP기반 전용 개발 툴도 개발 진행 중에 있다.

참 고 문 헌

[1] ARM, *AMBA Specification*, Rev. 2.0, 1999.

[2] IBM, *CoreConnect Bus Architecture*[Online]. Available: http://www-306.ibm.com/chips/techlib/product-families/CoreConnect_Bus_Architecture.

[3] VSI Alliance, *Virtual component interface standard* [Online]. Available: http://www.vsi.org.

[4] OCP International Partnership, *Open core protocol specification* [Online]. Available: http://www.ocpip.org.

[5] *CoreFrame architecture* [Online]. Available: http://www.palmchip.com.

[6] *Wishbone specification* [Online]. Available: http://www.opencores.org.

[7] *SuperHyway datasheets* [Online]. Available: http://www.superh.com.

[8] L. Benini and G. De Micheli, "Network on Chips: A New SoC Paradigm," *IEEE Computer*, pp. 70-78, Jan. 2002.

[9] K. K. Ryu, E. Shin, and V. J. Mooney, "A Comparison of Five Different Multiprocessor SoC Bus Architectures," in *Proc. EUROMICRO Symp. on Digital Systems Design*, pp. 202-209, Sept. 2001.

[10] ARM, *Multi-layer AHB Overview*, 2001.
[11] ARM, *AMBA AXI protocol specification*, 2003.
[12] *PCI specification* [Online]. Available: http://www.pcisig.com.
[13] *I2C specification* [Online]. Available: http://www.semiconductors.philips.com.
[14] *Serial Peripheral Interface* [Online]. Available: http://www.mct.net/faq/spi.html.
[15] T. T. Ye, L. Benini, G. D. Micheli, "Packetized On-Chip Interconnect Communication Analysis for MPSoC," in *Proc. IEEE Design Automation and Test in Europe*, pp. 344-349, Mar. 2003.
[16] D. Wingard, "MicroNetwrok based integration for SoCs," in *Proc. 38th Design Automation Conference*, pp. 673-677, June 2001.
[17] K.-M. Lee, S.-J. Lee, and H.-J. Yoo, "Low Energy Transmission Coding for On-Chip Serial Communications," in *IEEE Int'l SOC Conference (SOCC)*, Sept. 2004.
[18] M. B. Taylor, "The Raw Processor A Scalable 32-bit Fabric for Embedded and General Purpose Computing," in *Proc. Hotchips XIII*, Aug. 2001.
[19] W. H. Ho and T. M. Pinkston, "A Methodology for Designing Efficient On-Chip Interconnects on Well-Behaved Communication Patterns," in *Proc. of 9th. Int'l Symp. on High-Performance Computer Architecture (HPCA-9)*, pp. 377-388, Feb. 2003.
[20] K.-M. Lee, et al., "A 51mW 1.6GHz On-Chip Network for Low-Power Heterogeneous SoC Platform," in *IEEE Int'l Solid-State Circuits Conf. (ISSCC)*, pp. 152-153, Feb. 2004.
[21] S.-J. Lee, et al., "An 800MHz Star-Connected On-Chip Network for Application to Systems on a Chip," in *IEEE Int'l Solid-State Circuits Conf. (ISSCC), Dig. Tech. Papers*, pp. 468-469, Feb. 2003.
[22] K.-M. Lee, "Design and implementation of Low-Power Network-on-Chip for Application to High-Performance System-on-Chip Design," Ph.D dissertation, KAIST, Daejeon, Korea, 2005.
[23] *SNP specification* [Online]. Available: http://capp.snu.ac.kr

이 재 성

1999년 2월 아주대학교 전자공학부 학사 2001년 2월 아주대학교 전자공학과 석사 2001년 2월~현재 한국전자 통신연구원(ETRI) 연구원. 2002년 9월~현재 서울대학교 전기컴퓨터공학부 박사과정. 관심분야는 Advanced On-chip Interconnection Architecture, Protocol Processing, 병렬처리 프로세서 구조 연구, 영상처리

이 혁 재

1987년 2월 서울대학교 전자공학과 졸업 1989년 2월 서울대학교 전자공학과 석사 1996년 12월 퍼듀대학교 전기컴퓨터공학부 박사. 1996년 8월~1998년 8월 루지에나 공과대학 조교수. 1998년 11월~2001년 2월 Intel Corporation, 선임 연구원. 2001년 3월~현재 서울대학교 전기컴퓨터공학부 부교수. 관심분야는 Multimedia Processor, Computer Architecture, System-on-Chip(SoC), VLSI Design

이 찬 호

1987년 2월 서울대학교 전자공학과 졸업 1989년 2월 서울대학교 전자공학과 석사 1994년 6월 UCLA, 전자공학과 박사 1994년 8월~1995년 2월 삼성전자 반도체연구소 선임연구원. 1995년 3월~현재 숭실대학교 정보통신전자공학부 부교수 관심분야는 채널코덱의 구현, SoC on-chip-bus, SoC 설계 방법론, 3D 그래픽 프로세서 설계, MPEG codec 구현