FPGA 구현을 위한 IP 통합 기법의 비교

컴퓨터 및 전화통신의 초기 시절 이후로 인터커넥션 네트워크는 전기 공학의 매우 중요한 부분이 되었다[1]. 그러던 것이 초대규모 집적(VLSI) 회로 시대에 이르러서는 MOS 트랜지스터의 구동 특성[2]과 온칩 인터커넥트[3]의 비교적 높은 커패시턴스가 결합함에 따라서 더욱 중요하게 되었다.

자료제공 : 알테라 코퍼레이션 / www.altera.com

칩 내에서 기능 유닛을 연결하기 위해 이용되는 인터커넥션 네트워크는 칩의 성능에 중대한 아니 실로 절대적인 영향을 미친다. 버스는 가장 단순한 형태의 인터커넥트이나 밀도나 전력 측면에서 좋지 않은 선택이다. 버스를 최대의 속도로 구동하기 위해 요구되는 전력 및 공간이 버스의 커패시턴스에 따라서 급격히 늘어나기 때문이다[4]. 뿐만 아니라 멀티포인트 연결 네트워크는 특정한 시점에 단일의 대화만 이뤄지고 있거나 아니면 바로 인접한 요소 사이에 통신이 이루어지는 경우라 하더라도 버스의 전체 길이를 구동해야 하므로 좋지 않은 선택이다. 크로스바가 토대 디바이스 및 배선 기술에 의해 결정되는 최대 크기까지 최적의 솔루션이다. 일반적으로 다자간 통신을 위한 최적의 솔루션은 크로스바로 구축된 네트워크이다[5].

현황

오늘날 보는 것과 같은 온칩 버스는 컴퓨터 시스템에 이용되는 시스템 레벨 버스의 단순한 형태 혹은 축소판이다. 그러한 버스들이 잘 작동한다는 것은 확실히 입증되고 있으나 이들은 단지 당시의 상업적 및 기술적 한계 내에서 설계된 것들이다. 다시 말해서 와이어는 저렴하고, 회로는 비싸고, 인터커넥트는 로직보다 더 빨랐던 것이다.

그런데 오늘날에는 이러한 모든 것들이 변화했다. 첨단의 대규모 IC는 로직에 의해서가 아니라 인터커넥트에 의해서 속도가 제한되고 있다. 다중 클록 도메인이나 실험적인 파장 전달 기법을 이용한 칩들에서 그러한 징후를 쉽게 찾아볼 수 있다. 이제 로직 게이트는 풍부하다. 관용구처럼 인용되는 (그리고 흔히 잘못 인용되곤 하는) '무어의 법칙'에 따라서 대다수 엔지니어들이 어떻게 해야 할지 알 수 있는 것보다 더 높은 게이트 밀도가 제공되고 있다. 이러한 당황스러운 풍부함 때문에 회로 및 시스템 설계가 거꾸로 되돌아가고 있다. 로직 및 인터커넥트가 저렴하고 와이어를 절약하는 것이 비생산적인 것이 되었다.

버스는 버스 표준의 개발로 이어졌다. 이 역시 현재의 상업적 및 기술적 동향의 자연스러운 결과물이다. 각기 다른 업체로부터의 이산 IC가 인쇄 회로 기판 상에서 통신할 필요가 있었고 그래서 로직 레벨, 전류 구동, 시그널링 극성을 위한 표준이 요구되었다. 개별 핀 및 신호에 대한 표준(트랜지스터-트랜지스터 로직 또는 TTL 등)은 일단의 핀 또는 버스에 대한 표준으로 이어졌다. 시그널링 표준은 관련이 없는 부품들을 연결하는 것을 수월하게 했고, 버스 표준은 관련이 없는 마이크로프로세서, 주변장치, 메모리를 연결하는 것을 수월하게 했다. 이는 다시 전체적인 회로 보드를 플러그인 모듈로 다루는 보드 레벨 표준(VME, S-100, Futurebus, PCI 등)으로 이어졌다. 많은 경우에 그리고 실제로 대다수의 경우에 이들 보드 레벨 표준은 업체들 고유의 칩 레벨의 준표준을 확장한 것에 지나지 않았다.

핀 레벨, 칩 레벨, 보드 레벨 버스들이 전기적 호환성을 보장한다 할지라도 이들 각자가 실제로 상대방과 대화할 수 있다고 보장하지는 않는다. 말하자면 전류는 그러고 싶은데 프

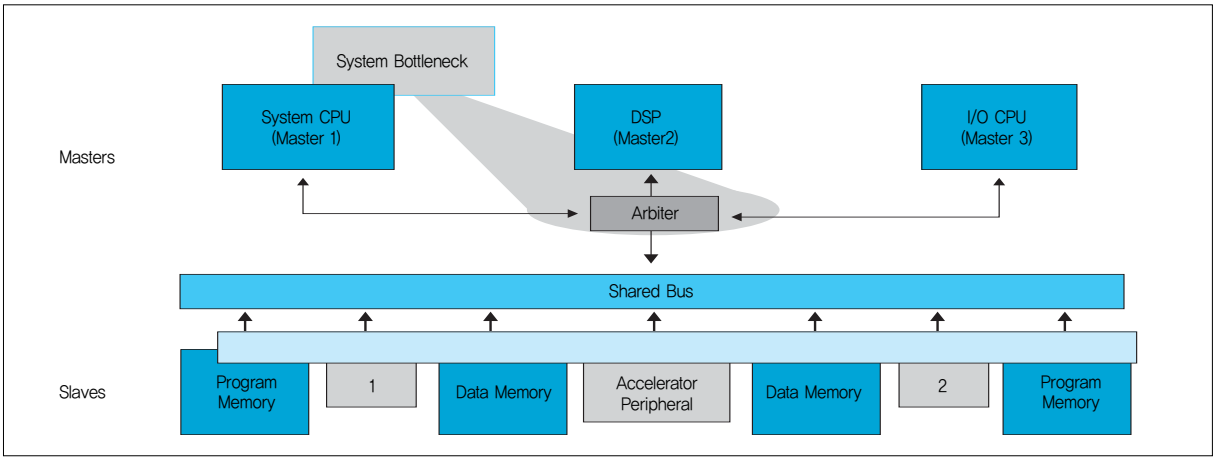

그림 1. 전통적 버스 토폴로지

로토콜이 취약하다. 버스 표준은 시스템 아키텍처 및 데이터 흐름의 더 높은 차원의 문제들을 해결하지는 못하지만 예를 들어서 1과 0의 연속적 열을 어떻게 정확히 시그널링할지 같은 자잘한 문제들을 해결하는 것에 있어서는 자신의 할 일을 잘 수행한다.

많은 법규나 정책이나 규정이 그러하듯이 버스 표준 역시 자신의 유용성을 반감시키는 측면이 있다. 버스는 채택되자마자 곧바로 시대에 뒤떨어지기 시작한다. 버스가 강제하는 바로 그 엄격함이 자신의 취약성이 된다. 버스 표준은 시그널링, 프로토콜, 대역폭, 활용 모델 등에 있어서 변화를 허용하지 않는다. 버스는 특성적으로 변화하는 것이 느리고 예기치 못한 것을 억제하는 것은 신속하다. 버스는 혁신을 불가능하게 하고 독창성을 저해한다. 그러면서도 버스는 끈질기게 물고 늘어지는 힘이 있다. 버스는 일치되지 않는다면 아무 것도 아닌 것이다. 버스는 끊임없이 변화하고 진화하는 대해에서 꾸준한 표준화의 등대로 남아 있다. 고정적인 규격이 없다면 부품 업체들이 어떻게 표준을 준수할 것인가?

대안 검토

버스에 대한 대안은 다양하며 이들 모두가 다양한 컴퓨터, 칩, 보드, ASIC, FPGA에 성공적으로 이용되고 있다. 버스가 모든 인터커넥션 문제에 대한 만병통치약이 아니듯이 이들 대안도 결코 만병통치약이 아니다. 표준 버스의 고정적 루팅과 타임테이블을 피함으로써 설계를 위한 새로운 가능성을 열고 그렇고 그랬을 프로젝트에 활기와 창의성을 불어넣을 수 있을 것이다.

버스와 네트워크

현행 버스 아키텍처에 대한 대안은 단순히 다른 유형의 버스일 뿐이다. 더 정확히 말해서는 네트워크, 스위치 패브릭, 크로스바 등의 다른 인터커넥트 토폴로지이다. 이들 인터커넥트 토폴로지는 다수의 칩, 보드, 시스템 레벨 제품에 성공적으로 이용되고 있으며 인기가 높아지고 있다. 칩 레벨 및 시스템 레벨 아키텍처가 변화함에 따라서 인터커넥트 전략이 변화해야 하는 것은 당연하다.

그림 1에서 보는 것과 같은 버스 토폴로지는 일반적으로 일대일 또는 일대다수 통신을 선호한다. 이들 버스 역시 각기 다른 업체의 칩들이 상호운용할 수 있도록 핀 레벨 또는 보드 레벨 인터페이스를 표준화해야 하는 필요성에 의한 당연한 결과였다. 버스는 다중의 마스터(트랜잭션을 발행하고 데이터를 소싱 또는 싱크하는 당사자)가 있을 수 있으나 한 시점에 한 마스터만 작동할 수 있다. 버스의 정의에 있어서 근본적인 것이 이의 배타적 특성이다. 한 시점에 한 마스터만 버스를 이용할 수 있으며 다른 모든 잠재적 마스터들은

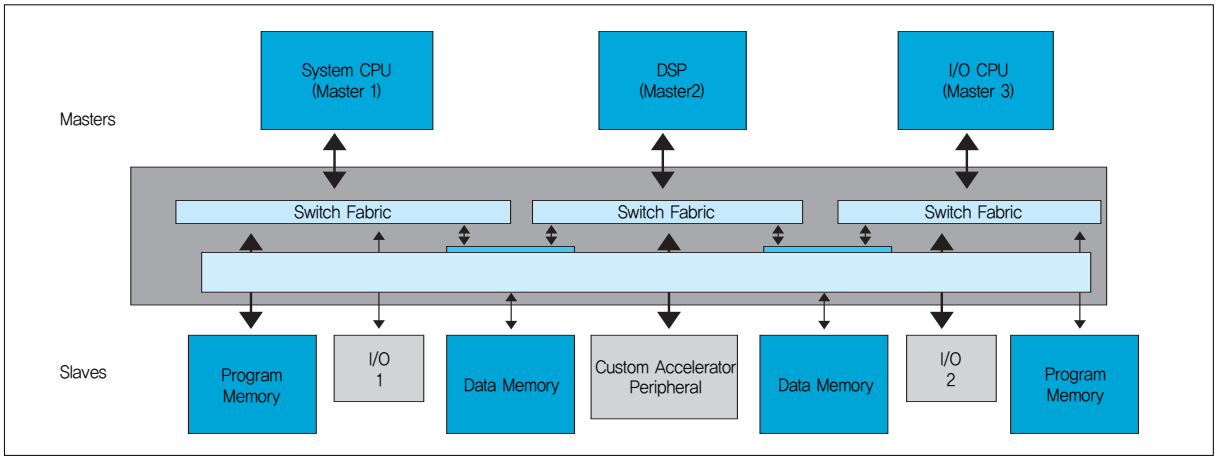

그림 2. 스위치 패브릭 토폴로지

기다려야 한다. 그러므로 버스 중재(공유 메커니즘 등)가 모든 버스 표준의 중요한 부분이 되었다.

대다수 버스가 다중의 마스터를 지원할 수 있으나 역시 특정 시점에 한 마스터만 작동할 수 있다. 마스터는 버스에 대한 액세스를 놓고 경쟁하고, 트랜잭션을 발행하고, 슬레이브("broadcall" 트랜잭션의 경우에는 다중의 슬레이브)가 응답하기를 기다리고, 그리고 버스를 해제한다. 그런 다음 마스터는 두 번째 트랜잭션을 발행하거나, 아니면 중재를 통해서 버스에 대한 제어권을 다른 마스터에게 잃는다. 이러한 구성은 액세스하기 위해 기다리는 것이 시스템 성능을 심각하게 저하시키는 시스템 병목 문제로 이어질 수 있다.

일정하게 향상된 버스는 다음 트랜잭션의 시작을 이전 트랜잭션의 끝과 겹치게 함으로써 중재 오버헤드 또는 트랜잭션 지연을 완화하는 분할 트랜잭션을 지원한다. 그렇게 함으로써 전체적인 트랜잭션 시간으로부터 귀중한 몇몇 사이클을 절약할 수는 있으나 모든 버스에 근본적인 단일 마스터 문제를 완화하는 것에 있어서는 전혀 효과적이지 못하다[6].

ASIC 또는 FPGA로 온칩 배선 자원을 이용해서 칩 레벨 버스를 편리하게 구현할 수 있다. 표준적 칩 제조 기법들이 칩 상에 버스를 구현하기 위해(그리고 전력 및 전역 클록 신호를 분배하기 위해) 편리한 비교적 길고 곧은 금속 층을 제공한다.

네트워크 토폴로지는 전통적 버스와 매우 유사하다. 네트워크 역시 공유 매체를 통해서 일대일 또는 일대다수 트랜잭션으로 설계되었다. 이들 역시(흔히 충돌 탐지 및 재시도 알고리즘을 이용해서) 버스에 대한 제어권을 중재한다. 뿐만 아니라 이들 프로토콜은 시간을 절약하기 위해서 인접한 트랜잭션을 어느 정도 겹치게 할 수 있다. 물리적 구현을 별개로 한다면 네트워크와 버스는 매우 유사하다.

스위치 패브릭

크로스바 스위치와 이의 좀더 일반화된 형태인 스위치 패브릭은 한때 표준 버스보다 더 단순하면서도 더 복잡했다. 크로스바 스위치는 다중 마스터 및 다중 슬레이브를 이용한 칩 또는 시스템을 위해서 다수-다수 통신 메커니즘을 제공한다. 버스나 네트워크와 달리 크로스바 및 스위치 패브릭은 다중 동시 트랜잭션을 지원한다. 이는 대역폭에 있어서 확실한 향상을 가능하게 한다. 다만 특정 시점에 한 마스터만 트랜잭션을 수행하고자 하는 경우는 예외이다. 그러한 경우에는 기존의 버스 또는 네트워크도 마찬가지로 잘 작동할 것이다. 다중의 마스터가 예측하지 못한 간격으로(그리고 흔히 동시적으로) 트랜잭션을 발행하는 좀더 일반적인 경우에는 스위치 패브릭(그림 2)이 더 나은 결과를 달성한다[7].

단일의 마이크로프로세서 또는 프로세서 코어를 이용한 시스템이라 하더라도 다중 마스터 시나리오는 꽤 일반적이

다. 가트너/데이터퀘스트의 조사에 의하면 시스템 온 칩(SoC)의 평균적인 프로세서의 수가 약 3.5개이며 이는 점점 늘어나고 있다. 다시 말해서 대다수 칩이 하나 이상의 프로세서를 포함하며, 여기서 '프로세서'는 소프트웨어를 실행하는 RISC, CISC, 비디오, 네트워크 프로세서 등으로 정의된다.

독립형 마이크로프로세서 칩조차도 흔히 하나 이상의 프로세서 '코어'를 포함한다. 인텔의 유명한 코어 2 듀오 및 이와 유사한 프로세서는 단일 실리콘 칩 내에 2개 이상의 이종 프로세서를 포함하는 대표적인 예이다. 프리스케일(이전 모토로라)은 십년 넘게 듀얼 프로세서 QUICC 및 PowerQUICC 통신 칩을 생산하고 있다. 네트워킹 및 통신 시장의 팹리스 칩 업체들은 각각의 칩에 4개, 10개, 혹은 그 이상의 프로세서를 이용한 디바이스를 꾸준히 생산하고 있다. 당연히 이들 디바이스 역시 내부적으로 스위치 패브릭을 이용한다.

크로스바 스위치란 용어는 글자 그대로 금속 바가 직각으로 서로 교차되도록 탑재된 데서 유래한 것이다. 스위치 또는 릴레이가 직교 바 쌍을 연결해서 전기적 접속을 이루었다. X 방향의 어떠한 바든 Y 방향의 어느 바에나 연결될 수 있으므로 이들을 흔히 X-Y 크로스바라고 하기도 한다. 크로스바 스위치는 통신 장비에 있어서 매우 일반적이었으며(지금도 여전히 일반적이며) any-to-any 연결이 가능한 특성 때문에 매우 유용하다.

전체적인 시스템 대역폭을 향상시키는 것 이외에도 스위치 패브릭은 또한 많은 중재 지연 및 버스 오버헤드를 방지한다. 버스는 단일 자원인 반면에 스위치 패브릭은 공유된다. 2개 마스터가 동일한 슬레이브를 어드레싱하거나 또는 그 반대가 아닌 한 어떠한 숫자의 트랜잭션이라도 동시에 처리할 수 있다. 자원 충돌이 발생하면 스위치 패브릭이 다른 공유 자원과 마찬가지로 중재한다. 그러한 충돌을 방지하기만 한다면 중재가 불필요하다.

스위치 패브릭은 그러므로 더 우수한 대역폭 및 더 낮은 지연시간을 제공한다. 프로세서가 코드를 가져오거나 데이터를 저장하거나 데이터를 검색하고 공유 버스에 근본적인 시간 지연을 발생시키지 않고자 하는 많은 고성능 디자인의 경우에 메모리 지연시간이 특히 중요하다. 다중의 마스터가 동시에 다중의 슬레이브를 어드레스할 수 있으면 총 대역폭(다시 말해서 동시적 마스터/슬레이브 트랜잭션 수) 역시 향상된다.

암달(Amdahl)의 법칙에 의하면 프로세서의 수가 증가하는 것에 따라서 메모리 대역폭이 증가되어야 한다[8]. 하지만 반도체 메모리 대역폭은 프로세서가 요구하는 만큼 빠르게 증가하지 못하고 있다. 메모리가 흔히 더 높은 성능을 달성하는 것을 방해하는 병목 지점이 되고 있으며 이러한 불일치로 인해서 멀티 레벨 캐시, 와이드 데이터 패스, 쓰기가능 코드 저장, 다양한 명령 셋 등의 다양한 처방이 등장하게 되었다. 그러므로 메모리에 대한 병목 문제를 해소하는 것이 모든 칩 디자이너들의 시급한 해결 과제이다. 그 모든 트래픽을 공유 버스를 통해 보내는 것은 그러한 목표에 위배된다.

인터커넥트의 구현

어느 인터커넥트와 마찬가지로 스위치 패브릭은 토대 실리콘의 강점에 중점을 두고 구현해야 한다. 초기의 크로스바 스위치는 단순히 그러한 교차 바였을 뿐이다. 최근의 스위치 패브릭은 반도체 게이트 및 금속 루팅 층을 이용한다. 버스와 비교해서 스위치 패브릭은 긴 금속 트레이스에 덜 의존하고 로직 게이트에 더 의존한다. 이 점은 인쇄 회로 기판에는 부적절하고 FPGA 같은 로직이 풍부한 칩에는 적합하다. 하드 ASIC은 그 둘의 중간 지점에 있다. 금속이 풍부하고 비교적 로직이 부족하므로 특별히 스위치 패브릭을 중심으로 특별히 설계하지 않으면 버스가 더 적합하기 때문이다.

FPGA의 로직이 풍부한 환경에서는 스위치 패브릭이 매우 적합하다. 스위치 패브릭이 FPGA의 특성에 적합할 뿐만 아니라 업계의 전반적인 경향에 적합하다. 디자이너들이 '소프트' 하드웨어 IP(intellectual property)로 전환함에 따라서 스위치 패브릭이 근본적으로 더 소프트하며 디자인에 포함되는 다른 소프트 IP와 잘 작동한다. 반면에 버스는 엄격하게 정의되고 신중하게 제어된다. 그러므로 스위치 패브릭이 통합하기가 더 편리하며 기존의 (그리고 미래의) 기능 블

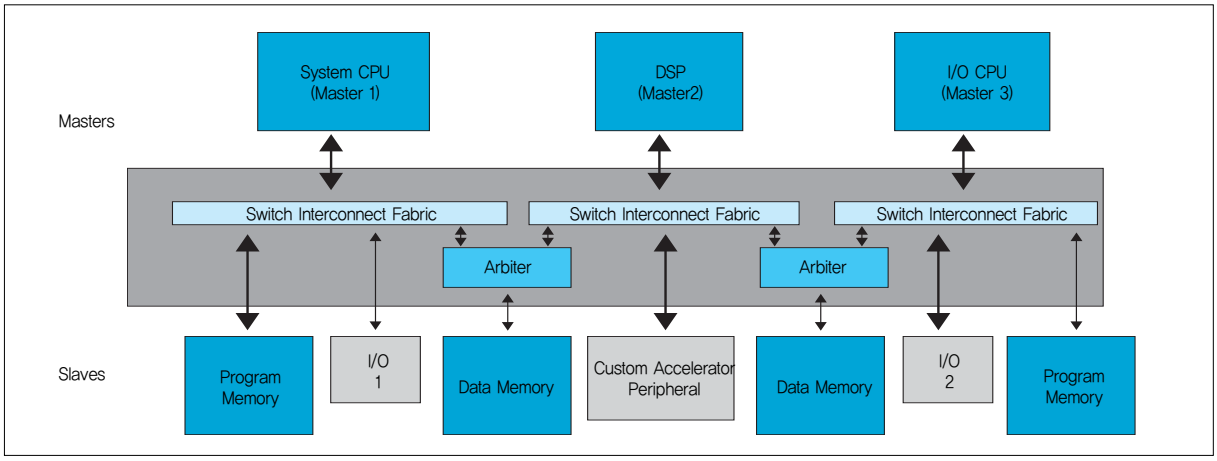

그림 3. SOPC Builder 시스템 인터커넥트 패브릭

록에 적응시키기가 더 용이하다.

고정적인 버스 인터페이스를 이용하지 않으면 기능 블록 혹은 컴포넌트가 더 신속하게 진화할 수 있으며 버스 정의에 의해서 제한되지 않는다. 이는 IP 디자이너에게나 IP 사용자에게나 IP 자신에게나 좋은 소식이다. 스위치 패브릭은 버스가 아니라 'IP 통합 백본'이 된다. 이는 칩의 여러 구역을 연결하는데 더 유연한 방식으로 그렇게 한다[9].

스위치 패브릭은 또한 EDA 툴의 발전을 활용한다. 스위치 패브릭은 더 추상적인 형태의 인터커넥션으로서 예를 들어서 Verilog 버스 모델이 제공하는 것보다 더 높은 추상화 수준으로 정의된다. 연결의 디테일이 추상화되는 한편 디자이너는 버스 타이밍 및 지연시간이 아니라 데이터 흐름 및 아키텍처에 집중할 수 있다.

추상화 레벨의 향상

소프트웨어 개발자들은 자신들의 작업의 디테일을 툴로 전가하는 데 익숙해지고 있다. 이들은 1과 0을 왔다갔다하는 대신에 명령 니모닉을 이용하며 니모닉 대신에 C나 자바 같은 고수준 언어를 이용한다. 이들 디자이너들은 모든 변수가 정확히 어디에 저장되고 어떻게 정렬되는지 지정하는 대신에 이러한 디테일을 컴파일러로 전가한다. 디테일을 전가함으로써 프로그래머들은 흐름 제어, 연산, 로직에 집중할 수 있으며, 한편 컴파일러는 모든 변수, 레지스터, 산술 연산, 저장 등을 정확하게 구현한다. C 프로그래머들은 '워드'가 16비트인지 아니면 64비트인지도 모르고 그렇다고 해서 문제가 되지도 않는다.

우수한 컴파일러와 마찬가지로 우수한 하드웨어 개발 툴은 최소의 작업으로 디자이너가 원하는 것을 목표 하드웨어로 맵핑할 수 있다. 그리고 우수한 컴파일러와 마찬가지로 이들 툴은 토대 플랫폼의 구조 및 자원을 이해하면서 그렇게 한다. 그렇게 함으로써 이들 툴은 두 가지를 달성한다. 하나는 디자이너에 대해서 디테일의 부담을 완화하고, 둘째는 디자이너 자신이 할 수 있는 것보다 더 우수한 맵핑을 달성하는 것이다. 구현을 위한 과잉 사양이나 과도한 제한은 흔히 최악의 하드웨어(또는 소프트웨어)로 이어진다. 컴파일러가 자신의 일을 수행할 수 없기 때문이다. 우수한 컴파일러는 단순히 변환기가 아니라 최적화기이다.

오늘날의 상황에서는 버스 규격이 하드웨어 툴이 제공할 수 있는 "솔루션 공간"을 과도하게 제한한다. 툴들이 토대 실리콘에 더 적합한 솔루션을 탐험할 수 있도록 자유롭게 하는 것이 더 편리할 뿐만 아니라 더 나은 방법이다. 특히 FPGA의 로직이 풍부한 환경에서 버스는 이 디바이스의 강점을 활용하기 위해서 적합하지 않다. 인터커넥트는 결코 완벽해질 수는 없으나 언제나 필요한 것이다. 인터커넥트는 버스를 이용하든 스위치 패브릭을 이용하든 네트워크를 이용하든 아

니면 다른 토폴로지를 이용하든 그 위에 칩의 나머지 부분(그리고 시스템의 나머지 부분)이 설계되는 토대를 제공한다. 인터커넥트가 잘 이루어졌을 때 시스템의 많은 목표를 이룰 수 있다. 인터커넥트가 불량하게(또는 부적절하게) 이루어졌을 때는 디자이너들이 생산적인 작업을 수행하는 것이 아니라 인터커넥트 문제와 싸워야 한다.

솔루션

알테라의 SOPC Builder 소프트웨어는 IP 및 디자인 블록의 통합을 자동화함으로써 더 높은 추상화 수준으로 작업해야 하는 디자이너들의 필요성을 충족하는 솔루션을 제공한다. SOPC Builder는 컴포넌트들을 위해 스트리밍 및 메모리 맵핑 인터페이스를 제공함으로써 디자인의 데이터 플레인과 제어 플레인을 통합할 수 있는 시스템 디자인 인터커넥트 패브릭을 제공한다. 이 인터커넥트 패브릭은 슬레이브 측 중재를 지원함으로써 동시 다중 마스터 버스 액세스를 가능하게 하므로 마스터 측 중재 방식을 통해서 시스템 성능을 향상시킨다.

브리지를 이용해서 토폴로지를 관리하고 성능을 향상시키기 위한 추가적인 기능들이 가능하다. 이들 기능을 이용해서 디자이너 또는 시스템 설계자는 더 높은 성능 및 쓰루풋을 필요로 하는 시스템 구역을 성능 요구가 낮은 구역과 분리시킬 수 있다. SOPC Builder의 스트리밍 인터페이스는 패킷 프로세싱, 다중화 스트림, DSP 데이터 같은 고대역폭 애플리케이션을 위해 높은 쓰루풋 낮은 지연시간 연결을 제공한다. 메모리 맵핑된 시스템 인터커넥트 패브릭은 최소한의 FPGA 로직 자원을 이용해서 데이터패스 다중화, 어드레스 디코딩, 대기 상태 생성, 주변장치 어드레스 정렬(원시 또는 동적 버스 크기 정렬에 대한 지원 포함), 인터럽트 우선순위 지정을 지원한다. 그림 3은 그림 2의 스위치 패브릭 아키텍처에 상응하는 SOPC Builder 시스템 인터커넥트 아키텍처의 다이어그램을 보여준다.

시스템에 새로운 컴포넌트를 추가할 때마다 또는 주변장치 액세스 우선순위를 변경할 때마다 SOPC Builder가 자동으로 새로운 최적화된 시스템 인터커넥트 패브릭을 생성한다. SOPC Builder가 자동으로 시스템 인터커넥트 패브릭을 생성하므로 성능을 향상시키거나 기능을 추가하기 위해서 시스템을 신속하고 편리하게 변경할 수 있다. SOPC Builder는 가입 및 웹 이디션 Altera Quartus II 설계 소프트웨어에 포함되어 제공된다. EP&C

관련 정보

- SOPC Builder: www.altera.com/products/software/products/sopc/sop-index.html
- Quartus II 안내서, Volume 4: SOPC Builder: www.altera.com/literature/quartus2/lit-qts-sopc.jsp

참고 문헌

1. V.E. Benes, "Mathematical Theory of Connecting Networks and Telephone Traffic," Academic Press, 1965.
2. Foty, "MOSFET Modeling With Spice," Prentice Hall, 1996.
3. Sung-Mo Kang, "CMOS Digital Integrated Circuits," McGraw-Hill, 2003.
4. Johnson and Graham, "High Speed Digital Design: a Handbook of Black Magic," Prentice Hall, 1993.
5. A. Brinkmann, J.-C. Niemann, I. Hehemann, D. Langen, M. Porrmann, U. R?ckert, "On-Chip Interconnects for Next-Generation System-on-Chips," in Proceedings of the 15th Annual IEEE International ASIC/SoC Conference, IEEE, 2002.
6. David J. Kuck, "The Structure of Computers and Computations, Volume 4," Wiley, 1978.
7. Brinkmann, et al.
8. Gene Amdahl, "Validity of the Single Processor Approach to Achieving Large-Scale Computing Capabilities," AFIPS Conference Proceedings, pp. 483-485, 1967.
9. Weste, Eshraghian, "Principles of CMOS VLSI Design," Addision Wesley, 1992.

감사의 말

- Chris Balough, 알테라의 소프트웨어 및 임베디드 마케팅 이사
- Kent Orthner, 알테라의 SOPC Builder 이사
- Ying Sosic, 알테라의 소프트웨어 및 임베디드 마케팅 임베디드 파트너 이사
- Richard Venia, 알테라의 소프트웨어 및 임베디드 마케팅 선임 제품 마케팅 엔지니어