주소 버스 공유를 통한 AXI 크로스바 스위치의 면적 및 전력 소모 감소

이 상 택　　전민제　　정의영
연세대학교 전기전자공학과

요약

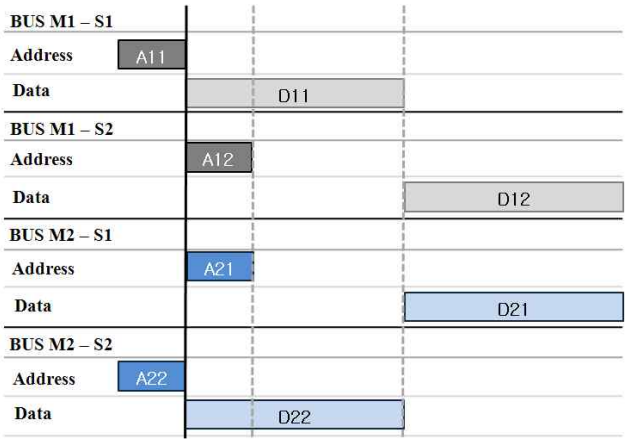
The crossbar switch (also called bus matrix) solutions are rapidly replacing the traditional shared bus-based solutions in the modern SoC designs to meet the ever increasing bandwidth requirement and low latency real-time constraint. The abundant channel resources, i.e. multiple buses, give this performance gain, but they also incur non-trivial overhead in hardware perspective. Also, since the amount of address and control information transfer is typically much smaller than that of data, the address channel utilization is much lower than the data channel. To tackle this point, we propose a new crossbar switch architecture which uses the shared bus for the address transfer and the bus matrix for the data transfer. The result shows that the proposed architecture saves 33.1% area and 45.4% power consumption only with negligible performance loss.

1. 서론

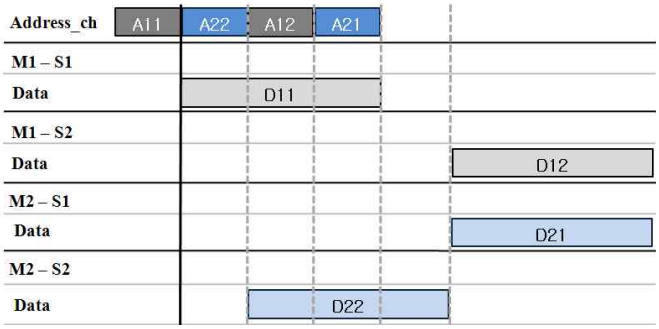
반도체 제조 공정의 발달은 내장형 시스템의 단일칩화(System-on-Chip, SoC)를 가능하게 하였으며, SoC의 출현은 다시 내장형 시스템의 고성능화를 가속시켜 그 적용 범위를 고성능 멀티미디어 및 다기능 시스템으로 확장할 수 있게 하였다. 이에 따라 고성능 내장형 프로세서, 메모리에 대한 요구가 커졌을 뿐 아니라 늘어난 데이터 전송량을 지원하기 위한 온칩 통신망에 대한 요구 또한 증가하고 있다. 간단한 구현과 값싼 설계비용으로 많이 사용되었던 공유버스 구조는 부착되는 모듈이 많아지고 데이터 전송량이 많아짐에 따라 전기적 특성 및 성능의 한계로 인해 최신 SoC 설계에는 더 이상 적합하지 않게 되었다. 공유 버스의 제한된 통신 자원을 해결하기 위해 제안된 버스 크로스바 (혹은 버스 매트릭스) 구조는 모듈간 점대점(point-to-point) 방식으로 전용의 버스를 연결하여 공유 버스의 대역폭 한계를 극복하였다. 그러나 하드웨어적인 비용 증가가 매우 크고 동작 주파수 또한 그 사이즈에 의해 제약되기 때문에 이를 해결하기 위해 부분 크로스바(partial crossbar)와 같은 해결책이 제시되기도 하였다 [1][2]. 또한 주소 및 제어 신호의 전송량은 데이터의 전송량에 비해 현저히 낮아 버스 매트릭스의 증가된 채널수를 제대로 활용하지 못하는 경우가 많다. 이러한 문제점을 해결하기 위해 본 논문에서는 AMBA3 AXI [3] 크로스바 스위치에서 주소 채널을 공유 버스 형태로 사용하고 데이터 채널은 버스 매트릭스를 유지하여 버스 매트릭스 수준의 성능을 유지하면서 면적과 전력소모를 감소시키는 방법을 제안한다.

2. 버스 크로스바에서 전용 주소 채널의 비효율성

최근의 버스 프로토콜들은 대부분 주소 및 제어 신호에 데이터 크기와 버스트 길이 등을 포함하여 한 싸이클 만에 전송을 마치게 되며, 여러 싸이클의 데이터 페이즈가 뒤따르게 된다. 이러한 프로토콜의 버스를 크로스바 형태로 구현할 경우 주소 채널의 활용도가 데이터 채널에 비해 현저히 떨어지는 현상이 발생하게 된다. 예를 들어, 2x2 스위치에 연결된 두 개의 마스터 M1과 M2가 두 개 의 슬레이브 S1과 S2로 하나씩의 버스트 길이 3의 데이터 전송을 요청하는 경우를 생각해보자. 이 경우 네 개의 주소 버스는 각각 그림 1(a)에서 보이는 것처럼 한 싸이클 동안만 사용될 뿐 데이터 전송이 이루어지는 나머지 싸이클 동안은 유휴상태가 지속된다.



(a) 기존 2x2 크로스바 스위치



(b) 제안된 주소 채널이 공유된 스위치

그림 1. 주소 버스 공유를 통한 효율 개선

반면, 그림 1 (c)는 크로스바에서 데이터 채널들은 그대로 유지한 채 주소 채널만을 공유 버스로 간소화하여 사용하였을 때의 동작을 나타낸다. 주소 채널이 네 개에서 한 개로 줄어들었지만 한 번에 한 마스터로부터의 데이터 요청만을 처리할 수 있는 슬레이브의 특성상 모든 데이터 전송이 종료되는데 단지 한 싸이클의 오버헤드만 있는 것을 확인할 수 있다.

3. 제안된 스위치의 구조

그림 2는 제안된 스위치 구조 중 쓰기 채널을 예로 보여주고 있다. 데이터 채널은 점대점 연결로 유지되며 주소 채널은 공유 버스로 연결된다. 점대점 연결을 위해서는 각 포트마다 목적지에 맞는 버스를 선택하기 위한 라우터와 여러 입력 버스 중 하나를 선택하기 위한 아비터가 필요하다. 주소 채널을 점대점 형태에서 공유 버스 형태로 가져가게 되면 각 포트마다 필요했던 라우터와 아비터가 각각 하나씩만 필요하게 되어 그 만큼의 하드웨어적인 비용을 절약할 수 있다.

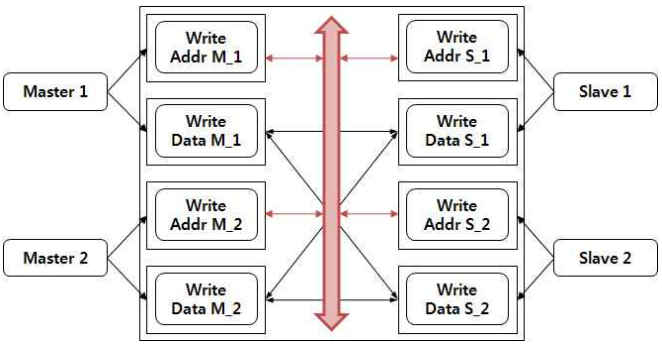


그림 2. 제안된 스위치 구조의 예

4. 실험

제안된 구조의 효율성을 검증하기 위해 4x4 AXI 크로스바 스위치와 주소 채널을 공유 버스 형태로 바꾼 스위치의 성능과 면적, 그리고 전력 소모를 비교해보았다.

제안된 구조는 데이터 전송이 주소 전송에 비해 버스 점유 시간이 큰 점에 착안한 것이므로 데이터의 버스트 길이가 매우 짧을 경우 성능 저하가 커지지만 반대로 충분히 긴 버스트 길이에 대해서는 크로스바 스위치와 거의 동일한 성능을 낼 것으로 예측해 볼 수 있다. 그림 3은 지정된 개수의 전송을 버스트 길이 1, 4, 8, 16에 대해 수행했을 때 모든 전송이 끝날 때까지 걸리는 시간을 비교한 것으로. 각 버스트 길이에 따른 크로스바 스위치와 제안된 구조의 성능 차이를 보여준다. 그림 3에서 확인할 수 있듯이 버스트 길이가 길어질수록 제안된 구조와 크로스바와의 성능 차는 줄어드는 것을 확인할 수 있다. 버스트 길이가 16일 때는 제안된 구조는 크로스바 스위치보다 단지 1 싸이클의 성능 저하만을 보여주었다.

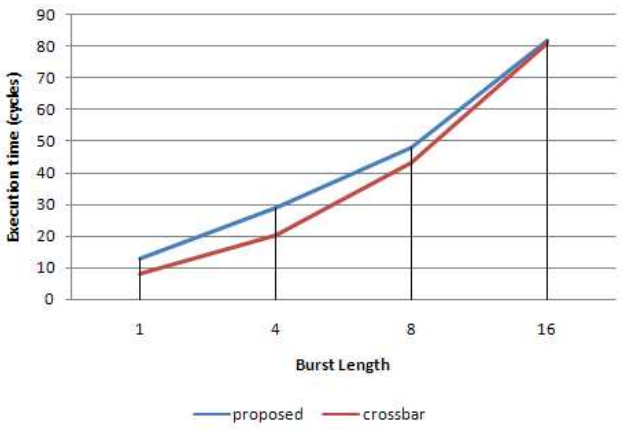


그림 3. 버스트 길이에 따른 성능 차이

다음으로 하드웨어 비용 측면에서의 이득을 확인하기 위해 4x4 크기에 대해 크로스바 스위치와 제안된 스위치를 0.13um 공정 라이브러리를 이용해 합성하였다. 표 1은 두 가지 구조에 대한 면적과 전력 소모를 나타낸다.

구조	면적 (gate count)	전력소모 (mW)
Crossbar	118,608	7.38
Proposed	79,357	4.03
Improve (%)	33.1	45.4

표 1 주소 버스 공유 스위치와 기존 크로스바 스위치의 면적 및 전력 소비 비교

4. 결론

고성능 프로세서 기술이 발달하면서 버스 성능의 향상은 필수가 되었다. 하지만 면적 측면에서의 효율과 저전력 설계 또한 중요한 인자가 되었다. 이번에 제안한 주소 버스의 공유 구조는 burst의 길이가 길어지면 실행 시간에 큰 영향을 주지 않고 약 30%의 면적 감소와 전력 감소의 이점을 얻는 것을 확인할 수 있었다.

감사의 글

본 논문은 지식경제부가 지원하는 국가 반도체 연구 개발 사업인 “시스템IC2010사업”을 통해 개발된 결과입니다.

참고 문헌

[1] S. Murali and G. De Micheli, "An Application-Specific Design Methodology for STbus Crossbar Generation", *DATE 2005*, pp. 1176-1181

[2] S. Pasricha, N.Dutt,M.Ben-Romdhane, "Constraint-Driven Bus Matrix Synthesis for MPSoC", *ASPDAC 2006*, pp. 30-35

[3] ARM, www.arm.com