

동적 재구성 가능한 고성능 시스템온칩 버스 구조에 관한 연구

서병현, 김규철
 단국대학교 전자컴퓨터공학과

A High Performance System-on-Chip Bus Architecture for Dynamic Reconfiguration

Byunghyun Seo, Kuychull Kim
 Dept. of Electronics and Computer Engineering, Dankook University

Abstract - 본 논문에서는 IDLE 전송만을 수행하거나 버스접근빈도가 낮은 디폴트 마스터(Default Master)를 버스에 대한 접근빈도가 가장 높은 마스터로 제정의 하고, 버스접근빈도가 가장 높은 마스터를 찾기 위한 블록을 제작하여 추가하였다. 이 블록을 이용하여 버스에 대한 접근빈도와 데이터의 특성에 따라 디폴트 마스터를 재설정 해줄 수 있다. 이로써 버스에 대한 접근시간을 줄이고, 다중버스구조에서 단일버스구조와 동일한 전송이 가능하게 하여, 기존의 디폴트 마스터를 사용한 버스 구조에서 보다 효율적인 전송이 가능하다.

1. 서 론

일반적으로 SoC(System-on-Chip)는 하나의 칩 내부에 프로세서, 버스, 메모리를 비롯한 레지스터 등을 모두 포함하게 된다. 설계자가 원하는 동작을 실현하기 위한 IP(Intellectual Property)와 이것을 연결하는 통신구조 [1]-[9]로 이루어지며 IP가 증가함에 따라 통신구조의 복잡도가 증가하게 된다. 지금까지 SoC 설계에 있어 가장 널리 쓰이는 통신구조는 버스구조이며 대부분의 경우 중앙 처리를 담당하는 CPU(Central Processing Unit)가 버스를 통해서 주변 IP를 제어하는 방식을 취해 왔으나 통신 패턴이 다양화되고 그 양이 증가함에 따라 많은 문제점이 제기되었다.

현재 가장 많이 사용되는 규격은 AMBA구조로 Central Multiplexer Interconnection Scheme과 함께 사용하기 위해 디자인 되었다. 하지만 이는 병렬 처리에 대한 한계가 있기 때문에 다양한 다중버스구조를 통한 병렬처리를 위한 해결책들이 나왔다. 이러한 구조에서도 전송을 시작할 수 있는 마스터들은 버스를 사용하기 위해서 아비터에게 버스사용권한을 요청하고, 아비터는 때때 다른 마스터에게 부여한 버스권한을 우선순위가 높은 마스터에게 재부여하는 버스 중재시간이 필요하다. 또한, 서로 다른 버스에 놓인 마스터와 슬레이브간의 통신을 위해서 추가적인 버스접근시간이 요구된다. 이러한 문제는 AMBA-AHB Specification[2]에 나와 있는 디폴트 마스터의 정의를 본 논문에서 제안하는 새로운 방식으로 해결함으로써 보다 효율적인 데이터 전송이 가능하다.

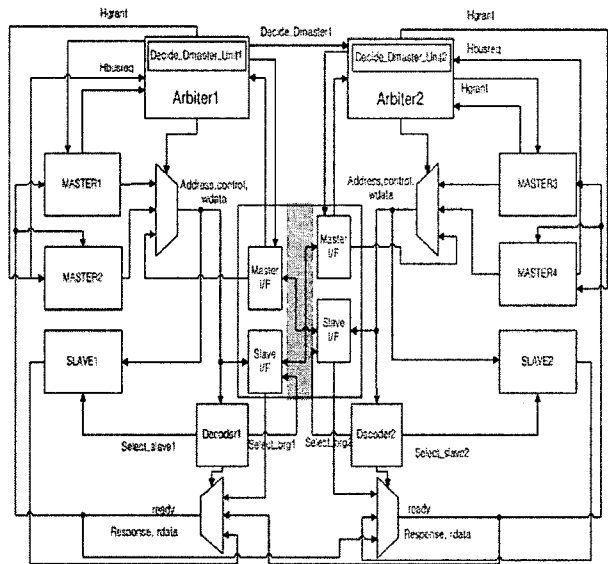
기존의 아비터는 다른 마스터들의 버스 요청이 없는 경우 가장 낮은 우선순위를 가지고 있으며, IDLE 전송만을 수행하는 디폴트 마스터에게 버스권한을 부여한다. 단, 디폴트 마스터가 없는 경우에는 버스에 대한 접근빈도가 가장 낮은 마스터로 대체한다. 이러한 디폴트 마스터를 사용함으로써 버스에 대한 데이터 전송이 없기 때문에 저전력 모드로 동작이 가능하다는 장점이 있다. 하지만 기존의 디폴트 마스터를 사용하는 경우 모든 마스터는 다른 마스터에게 양도되었던 권한을 재획득하는 과정에서 매번 긴 버스접근시간을 필요로 하게 되며, 브리지를 통과하여 서로 다른 버스로 전송하는 경우에는 추가적인 중재시간이 요구된다. 예로, 두개의 버스를 가로지르는 single transaction의 경우에는 대기상태가 없을 시 버스접근시간은 전송시간과 동일하게 된다. 보다 많은 브리지를 통과하여 다른 버스로 전송되는 경우에는 데이터전송보다 버스접근에 더 많은 시간이 소요되어 데이터 전송의 비효율성을 초래한다. 기존의 디폴트 마스터를 사용함으로써 저전력 모드로 동작하게 하는 장점이 있어서도 일부 다중버스구조에서는 보다 효율적인 전송을 위해서 마스터 자체를 변형시키거나 마스터에 버퍼를 연결함으로써, 버스요청신호와 동시에 제어신호를 내보도록 하고 있다. 이는 오히려 중재시간이 늘어남에 따라 불필요한 데이터를 내보내는 시간을 늘리는 결과를 초래하고 보다 많은 전력을 소모하게 된다. 또한, 일반적으로 전송할 데이터가 없는 마스터는 버스에 대한 권한을 획득하지 못한 상태에서 IDLE 전송을 하기 때문에 기존의 디폴트 마스터를 사용함으로써 얻는 이득이 사라지게 된다.

2. 동적 재구성 가능한 버스 구조

<그림1>은 본 논문에서 제안하는 두개의 버스로 이루어진 다중버스구조를 간략화 하여 나타낸 것이다. 일반적인 다중버스구조와 유사한 구조를 갖는다. 기본적인 다중버스구조는 전송을 시작할 수 있는 마스터, 이에 응답할 수 있는 슬레이브, 각 마스터의 버스사용 요청을 중재할 수 있는 아비터, 슬레이브를 선택할 수 있는 디코더 그리고 서로 다른 버스로 데이터를 전송할 수 있는 브리지로 구성된다.

브리지가 디폴트 마스터로 선택되었을 경우 안정적인 동작을 위하여 일반적으로 사용되는 "Cross Bridge"에 [3]에서 사용한 By-Pass Bridge 기능

을 추가한 브리지를 제작하여 사용하였으며, 아비터로 들어오는 정보를 이용하여 버스에 대한 접근빈도가 가장 높은 마스터를 찾아내어 아비터에게 알려주는 Decide_Dmaster_Unit이 추가되었다.



<그림 1> 제안된 버스 구조의 구성도

2.1 브리지

브리지는 서로 다른 버스에 위치한 마스터와 슬레이브를 연결해주는 역할을 하는 구성요소이다.

첫 번째 버스에 놓인 마스터로부터 두 번째 버스에 놓인 슬레이브를 향해 전송이 요청된 데이터는 브리지의 슬레이브 인터페이스에 저장되고, 브리지의 마스터 인터페이스는 아비터에게 버스권한을 요청하고 권한을 부여받기까지 마스터를 대기상태로 유지시킨다. 권한을 부여받은 브리지는 최종 목적 슬레이브로 전송을 시작하고 응답을 마스터로 전송한다. 브리지가 두 번째 버스의 디폴트 마스터로 지정되어 있다면 브리지의 슬레이브 인터페이스는 마스터 인터페이스와 직접 연결된다. 이는 다중버스구조를 유지하며 단일버스구조와 동일하게 동작할 수 있도록 해줄 수 있다. 예로, 두 번째 버스에 위치한 슬레이브를 향해 전송을 시도하는 첫 번째 버스에 있는 마스터는 브리지를 통과하기 위한 추가적인 대기시간 없이 바로 최종 목적지인 두 번째 버스의 슬레이브로 데이터 전송이 가능하다.

2.2 아비터

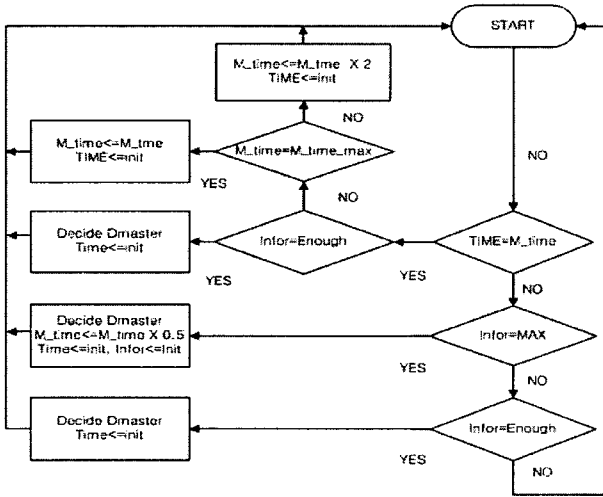
아비터는 특정 시간에 오직 하나의 마스터만 버스에 접근할 수 있도록 보장해 주는 구성요소이다. 이러한 기능을 수행하기 위해서 아비터는 마스터들의 버스요청을 관측하여 우선순위가 높은 마스터에게 버스권한을 부여한다. 본 논문에서 제안한 버스구조에 사용된 아비터는 AMBA-AHB구조를 사용하는 시스템에서 가장 많이 사용하는 중재방식인 라운드로빈 방식을 따른다. 라운드로빈 방식은 하나의 마스터에게 계속해서 버스권한이 부여되는 문제를 방지할 수 있다. <그림1>에서 사용된 아비터는 Decide_Dmaster_Unit으로부터 받은 제어신호를 이용하여 기존의 디폴트 마스터가 아닌 본 논문에서 제안하는 디폴트 마스터에게 버스 권한을 주는 것 이외에는 기존의 아비터와 동일하게 동작한다.

아비터는 사용빈도가 가장 높은 마스터에게 권한을 부여함으로써, 버스 중재 횟수를 줄일 수 있으며, 브리지가 디폴트 마스터로 선택된 경우에는 브리지를 통과하며 발생하는 대기 시간을 제거할 수 있다. 아비터2에 의해 브리지가 디폴트 마스터로 선택된 경우, 다중버스구조가 단일버스구조처럼 동작하기 위해서는 하나의 마스터에만 버스권한이 부여 되어야 한다. 이를 위해서 아비터는 브리지로부터 오는 신호들을 확인할 수 있어야 한다.

2.3 Decide_Dmaster_Unit

Decide_Dmaster_Unit은 모니터시간을 조건에 따라 재설정하면서 버스에 대한 사용 빈도가 가장 높은 마스터를 결정하고 이 정보를 아비터로 전해주는 구성요소이다.

<그림 2>에서는 Decide_Dmaster_Unit의 동작에 대한 흐름도를 나타내었다. 모니터 시간(M_time) 동안 감지할 수 있는 양보다 많은 정보가 입력되면 모니터 하는 시간을 반으로 줄이고 입력된 정보로 버스에 대한 접근빈도가 가장 높은 마스터를 디폴트 마스터로 지정한다. 입력된 정보가 디폴트 마스터를 정하는 최소 정보량보다 작은 경우 허용 범위(M_time_max) 내에서 모니터 시간을 두 배로 늘리고 입력된 정보를 버린다. 이와 같은 작업을 반복함으로써 마스터들의 다양한 버스접근빈도에 대해서 디폴트 마스터를 찾아 재설정해 줄 수 있다.



<그림 2> Decide_Dmaster_Unit의 흐름도

2.4 데이터 발생기

데이터 발생기는 실제 버스의 구성요소가 아닌, 본 논문에서 제안한 버스 구조를 검증하기 위하여 제작한 블록이다. 각 백-엔드 로직[9]을 가진 마스터 인터페이스에 연결되어 로드를 부여할 수 있다. 각 버스에 대하여 세 개의 스테이지와 하나의 패턴 카운터를 가지고 있다. 스테이지1에서는 마스터들을 그룹별로 나누고, 전송할 슬레이브를 선택한다. 스테이지2에서는 하나의 데이터 패턴에 대하여 마스터의 빈도를 결정해 준다. 스테이지3에서는 마스터의 온오프 상태를 설정해주며, 패턴 카운터는 각 스테이지를 통해 생성된 데이터 패턴의 반복 횟수를 설정한다.

3. 실험 및 결과

Altera Quartus 4.2를 이용하여 구현하였으며, Altera사의 ARM코어와 PLD가 결합된 Excalibur를 Target Device로 설정하였다. 설계된 모듈을 시뮬레이션 하기 위하여 Modelsim 6.0d를 사용하였다. 각각의 버스에는 세 개의 마스터, 한 개의 슬레이브와 브리지를 사용한다. 제작된 마스터는 실제 마스터가 아닌 인터페이스만을 가지며, 마스터의 백-엔드 로직에 데이터 발생기를 연결하여 각 마스터가 다양한 패턴의 전송을 하도록 설계하였다.

<표 1> 각 버스 구조에 사용된 Logic Cell 비교

버스구조	Logic Cell	차이(LC)	중요 추가 블록
제안버스	2026	기준	Decide_Dmaster_Unit(179 LC)
다중버스	1842	183	-
단일버스	1753	273	-

<표 1>은 Quartus 4.2를 이용하여 얻은 결과로 각 버스구조에 사용된 Logic Cell의 수를 보여주고 있다. 제안된 버스구조에서는 단일버스구조나 다중버스구조보다 많은 Logic Cell이 사용되었다. Decide_Dmaster_Unit의 추가로, 첫 번째 버스에서 85 Logic Cell, 두 번째 버스에서 94 Logic Cell이 추가되었다. 데이터 발생기 블록에는 430 Logic Cell이 사용되었다.

<표 2>은 Quartus4.2에서 설계한 모듈을 Modelsim 6.0d에서 각 버스구조를 동일한 조건에서 10ms동안 시뮬레이션 한 결과이다. <표 2>의 데이터비율은 단일버스 내의 전송인 local transaction에 대한 서로 다른 버스 전송을 하는 cross transaction의 비율이다. 데이터비율의 값이 큰 경우 즉, cross transaction의 전송비율이 높은 경우에는 다중버스구조보다 단일버스구조가 효율적인 전송이 가능하며, 데이터비율의 값이 작은 경우에는 다중버스구조가 단일버스구조보다 효율적인 전송이 가능하다. 제안된 구조는 새롭게 정의된 디폴트 마스터를 사용함으로써 버스접근빈도가 높은 마스터에 대한 중계시간을 단축하였으며, 브리지를 디폴트 마스터로 선택함으로써 다중버스구조를 유지한다. 그리고 필요에 따라 단일버스 모드와 동일하게 동

작하게 동작할 수 있으므로 다중버스나 단일버스에 비하여 다양한 데이터 비율에서 모두 효율적인 전송을 하고 있다는 것을 확인할 수 있었다.

<표 2> 각 버스 구조별 데이터 전송률 비교

구분	버스구조	데이터비율	데이터 전송량(워드)	이득
1	제안버스	0.2	103904	기준1
	다중버스	0.2	88012	18.1%
	단일버스	0.2	70560	47.3%
2	제안버스	0.5	97915	기준2
	다중버스	0.5	82980	18.0%
	단일버스	0.5	74300	31.8%
3	제안버스	1	96833	기준3
	다중버스	1	79412	21.9%
	단일버스	1	77208	25.4%
4	제안버스	2	92670	기준4
	다중버스	2	75943	22.0%
	단일버스	2	81065	14.3%
5	제안버스	5	90053	기준5
	다중버스	5	72712	23.8%
	단일버스	5	84523	6.5%

4. 결 론

본 논문에서는 버스에 대한 접근빈도가 가장 높은 마스터를 디폴트 마스터로 정의하였다. 즉, 다른 마스터들로부터 버스요청이 없는 경우 버스접근빈도가 가장 높은 마스터에게 버스사용권한을 부여함으로써 버스접근시간을 줄이고, 다중버스구조를 유지하면서 단일버스구조처럼 동작할 수 있는 버스구조를 설계 제작하였다. 그리고 실험을 통하여 같은 시간동안 기존의 버스구조보다 더 많은 데이터 전송이 가능하다는 것을 확인하였다. 이와 같은 구조는 다양한 버스접근빈도나, 데이터 전송패턴을 가진 시스템에 유용하리라 생각된다.

[참 고 문 헌]

- [1] "CoreConnect Bus Architecture", <http://www.chips.ibm.com/products/coreconnect/>.
- [2] "AMBA 2.0 Specification", <http://www.arm.com/armtech/AMBA>.
- [3] K.Sekar, K.Lahiri, A.Ragunathan, S.Dey, "FLEXBUS: a high-performance system-on-chip communication architecture with a dynamically configurable topology", Design Automation Conference, 2005. Proceedings. 42nd 13-17, 571 - 574, (2005).
- [4] 공민철, "다중 마스터 설계를 이용한 AMBA 인터페이스 구현", 한국산업기술대학, (2004).
- [5] S.Han, A.Baghdadi, M.Bonaci, S.Chae, and A.A.Jerraya, "An Efficient Scalable and Flexible Data Transfer Architecture for Multiprocessor SoC with Massive Distributed Memory.", Design Automation Conf., 370-371, (2004).
- [6] K.Lahiri, A.Ragunathan, S.Dey, "Design of High-Performance System-on-Chips Using Communication Architecture Tuners", IEEE Trans. on CAD, vol23, 919-932, (2004).
- [7] T.Meyerowitz, C.Pinello, and A. Sangiovanni-Vincentelli, "A Tool for Describing and Evaluating Hierarchical Real-Time Bus Scheduling Policies", Design Automation Conf., 312-317, (2003).
- [8] S.Pasricha, N.Dutt, and M.B.Romdhane, "Fast Exploration of Bus-based On-chip Communication Architectures", Symp. HW/SW Codesign, (2004).
- [9] "multi_master_reference_design", <http://www.altera.com>.