멀티미디어용 다중작업이 가능한 동기 세그먼트 구조

전 치 훈, 연 규 성, 황 태 진, 위 재 경+
숭실대학교 전자공학과 SIP 연구실
+wjk@ssu.ac.kr

Synchronous Segmented Bus Architecture for Multitasking on Multimedia System

Chi-Hoon Jun, Gyu-Sung Yeon, Tae-Jin Hwang, Jae-Kyung Wee+
School of Electronic Engineering Soongsil University, Seoul, Korea
+wjk@ssu.ac.kr

본 논문은 OCP(Open Core Protocol)에 호환되는 파이프라인 구조를 가진 시스템 버스와 MPEG 시스템에 적합한 메모리 버스를 갖는 계층 구조를 가지는 새로운 동기 세그먼트 버스를 제안한다. 이 구조는 MPEG 시스템의 모바일 제품에 사용되는 영상 데이터 처리를 위한 메모리 인터페이스에 기반을 둔 버스 구조와 Multi-master와 Multi-slave를 사용하여 고성능의 다중 처리를 위한 양방향 다중 버스 구조(bi-direction multiple bus architecture)를 가진다. 효율적인 데이터 처리를 위하여 파이프라인 stage와 결합된 Master와 Slave의 주소번지가 latency를 결정하며, 시스템의 특성에 따라서 IP 코어를 배치하였다. 제안된 버스는 저 전력 구현을 위하여 세그먼트 버스 구조를 가지고, 멀티미디어 SoC 시스템의 성능 저하 없이 다중 작업이 가능한 구조를 갖는다. Wirability를 고려하여 양방향 구조를 채택하였고, Testablility를 위하여 단방향(uni-direction) 구조와 대체 가능하다. 또한, Local arbiter의 수정만으로 Master의 추가가 가능한 확장 구조를 가진다. Latency를 줄이기 위하여 직접 제어 방식과 단순한 구조의 Central arbiter로 구현 되었다.

Keyword: 버스, MPEG, 파이프라인, 계층구조, 양방향, 단방향, Wirability, Testablity, Latency

Ⅰ. 서 론

VLSI 기술의 발전으로 보다 많은 양의 회로를 단일 칩에 집적이 가능하게 되었고, 이는 SoC가 가능하게 되었다. 고집적의 시스템을 단 기간에 계발하기 위해서, IP(Intellectual Property)로 통용되는 기존에 잘 설계된 IP 코어를 이용하는 것이 절실히 요구된다[1]. 이러한 재사용 가능한 IP 코어와 IP 코어 사이의 통신을 위하여 On-Chip 통신은 주요한 이슈로 부각되었다. On-chip 통신을 위하여 많은 버스[2]-[4]가 개발되었다.

본 논문에서는 이러한 IP 코어의 재사용의 효율을 높이고자 VCI(Virtual Component Interface)와 실제 많이 사용 있는 OCP(Open Core Protocol)에 적용 가능한 버스와 MPEG 시스템과 같은 멀티미디어 모바일 제품에 적합한 버스구조를 제안한다.

본 논문의 제 Ⅱ장에서는 기존의 버스의 시스템에 MPEG 시스템 적용 시의 문제점과 제안된 버스구조와의 차이점과 제안된 버스 구조의 통신 프로토콜 특징에 대하여 다룬다. 제 Ⅲ장에서는 제안된 버스구조의 시뮬레이션의 결과와 제안된 버스 구조의 간단한 요약과 기대효과를 제시하며 결론을 맺는다.

Ⅱ. 본 론

1. MPEG 시스템 적용 시 기존버스의 문제점

본 논문에서는 일반적으로 가장 많이 쓰이는 ARM사의 AMBA와 Sonics사의 Sonics 버스를 MPEG 시스템에 적용 했을 시의 문제점을 알아보고, 그 외의 samba-bus[7]와 segment-bus[6]의 차이점을 다루었다.

ARM사의 AMBA의 경우 MPEG 시스템에 적용 했을 때의 대표적인 AMBA의 특징과 문제점은 다음과 같다. AMBA는 ROM이나 RAM과 같은 on chip IP들

로 구성된 시스템에 좋은 성능을 보이는 반면 외부 메모리와 같은 긴 latency를 가지는 시스템에서 성능 저하를 보인다. 이러한 버스는 전체 버스를 점유하게 되는 문제점을 해결하기 위하여, AMBA는 split과 Multi-layer방식을 지원한다. 하지만 동작과 구조의 특성상 Master의 수가 늘어나면 버스의 면적이 증가하게 되고, Routing이 많아지고, 복잡해지는 wirability문제가 발생한다.

Sonics의 버스는 하드웨어 적으로 AMBA보다 구현이 더 쉽고, arbitration 방식으로 Time schedule(TDMA)을 이용한다. Sonics의 버스를 사용하여 SDRAM을 사용할 경우, Master가 요청 후에 데이터의 입출력이 일어날 때까지 수 사이클 이상의 시간이 필요 할 때 발생하는 성능의 저하를 막기 위하여 SDRAM 인터페이스가 직접 해당하는 Master에게 Read/Write를 요구하게 함으로 성능 향상을 요구 할 수 있게 하는 등 AMBA와 비교하여 좀더 Optimize가 쉬운 버스 구조를 가진다. 하지만 SDRAM의 효율을 높이려면 TDMA등의 경우 여러 개의 Request들을 모아 burst의 연관성을 찾아서 수행해야 하는 정교한 SDRAM 인터페이스의 구현이 필요로 하는 문제점이 있다. 본 논문에서 제안한 버스는 기존의 세그먼트 버스[5]와 비동기 SAMBA-BUS[6] 개념과 다른 다중 작업을 위한 양방향 버스와 기존의 칩 설계와 호환성을 가지는 동기 시스템을 적용하였다.

2. 제안하는 버스 구조

가. 전체 구조

앞 절의 기존 버스의 문제점을 해결하고자 MPEG 시스템의 모바일용 데이터 처리를 위한 메모리 시스템을 고려한 버스 구조와 multitasking 처리를 가능하기 위한 고성능 시스템을 고려한 계층적 버스 구조를 제안한다. 각각의 버스를 시스템 버스와 메모리 버스라 칭한다.

제안하는 버스구조는 다음의 특징을 가진다. 첫째, MPEG 시스템의 모바일 제품에 사용되는 영상 데이터 처리를 한다. 둘째, MPEG와 같은 data-width가 큰 시스템은 메모리버스와 인터페이스하고, 성능 저하 없이 다중작업이 가능한 시스템 버스 구조를 가진다. 셋째, 각 IP 코어의 어드레스번지를 latency에 따라 고정하여 데이터 통신 효율을 높이고, 빠른 버스 채널의 선택을 위하여 arbiter를 단순화 하였다. 그림1은 제안된 전체 버스 구조를 나타낸다. 그림1에서는 시스템 버스와 메모리 버스의 계층적인 버스 구조를 보인다. 또한 여러 arbiter와 버스와 버스사이를 연결하는 Bridge의 구조를 보인다. 시스템 버스는 양방향 버스구조와 다중작업을 위한 구조로 제안되었고, 메모리 버스는 MPEG 시스템의 각 IP들과의 인터페이스를 위한 버스로 메모리의 점유율이 높은 IP들로 구성된 시스템을 갖고, 효율적인 통신을 위하여 멀티 Master와 Single Slave의 구조를 가진다.

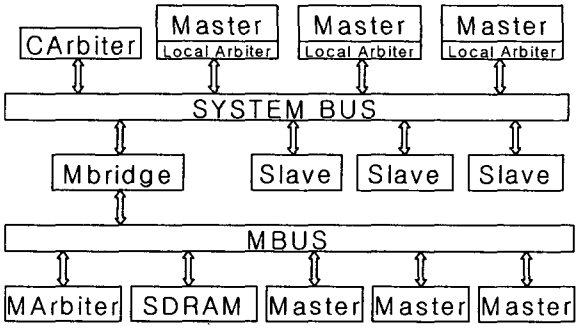

그림1 제안된 버스 구조

Local arbiter는 그림1에서 각 Master에 연결되어 있고, 설계 전 고려된 Master들의 사용빈도에 따라 TDMA 방식으로 arbitration을 한다. Central arbiter는 Local arbiter에서의 요청에 대하여 버스의 채널 상태에 따라 Master의 승인 여부 결정한다. 메모리 버스(MBUS) arbiter는 MPEG 각 IP 코어에서의 요청에 따라 MPEG 시스템의 효율적인 동작을 하도록 arbitration한다. Mbridge는 System 버스에서 slave로 MBUS에서 Master로서의 역할을 한다.

나. 메모리 버스 (MBUS) 구조

MPEG는 Data-width가 크고, 메모리의 접근 빈도가 높은 특성을 가진다. SDRAM은 Master에서 요청 후 실제로 데이터 입출력이 일어날 때까지 수 사이클 이상의 시간이 걸리는 긴 latency를 갖고 있다. SDRAM 인터페이스가 해당 Master 에게 읽기/쓰기를 요구하고, 데이터 arbitration 신호는 미리 대기하고 있을 수 있다. 미리 메모리 접근 할 타이밍을 알고 있어야 하므로 MPEG의 각 블록들의 동작과 데이터의 전송 타이밍을 정확히 알고 있어야 한다. SDRAM의 동작 특성에서 Bank-interleaving을 burst동작과 결합할 경우 효율을 극대화 할 수 있다. 메모리 접근빈도를 예상하여 메모리 접근 가능한 타이밍에 다중 작업을 가능하게 하는 버스를 다음 절에 설명한다.

다. 시스템 버스 구조

제안된 버스 구조 중 그림2는 Master에서 Slave로의 전송 요청 시 다중 작업을 위한 시스템 버스의 구조를 보여준다. Master에서 요청이 일어 날 경우 Local arbiter에서 arbitration을 한 후 Dones, Requests, Accepts의 직접 제어 신호와 central arbiter의 버스 채널 신호로 버스 채널을 제어한다. 이후 승인된 Master에 한하여 양방향 버스 구조를 가지는 사각형 점선의 블록을 통하여 전송이 이루어진다. 이 블록은 Mux, F/F, 3-State 버퍼로 구성되어 있다. 성능 향상과 데이터가 정확한 전송을 위하여 F/F을 삽입한 파이프라인 구조를 가지고, wirability를 높이고자 양방향 전송이 가능한 구조로 설계 되었다. 양방향 구조의 버스에 사용되는 3-State 버퍼는 central arbiter에서 제어하여 버스의 채널을 열고 닫는 스위치의 역할을 수행한다. Local arbiter에서 요청된 어떤 동작에 대하여 각 Master와 Slave의 주소번지를 central arbiter에서 ROM으로 저장하고 이전 Master와 Slave의 주소번지를 비교하여 3-state 버퍼가 스위치로서 동작해서 전송을 수행한다.

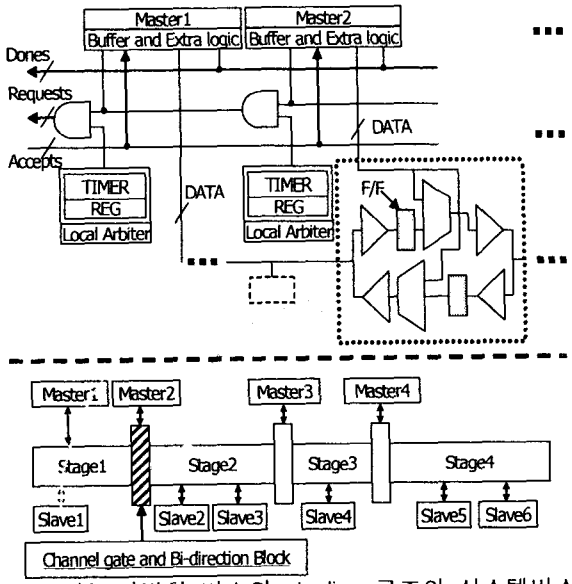

그림2. 양방향 버스와 pipeline 구조의 시스템버스

표 1. 다중 작업을 위한 Master ID와 Slave ID의 고정

Master ID	001	010	011	100
Slave ID	001	010/011	100	101/110
Bus Channels	001	010	011	100

표1은 다중 작업을 위한 Master와 Slave의 주소번지를 나타낸다. "Master ID"와 "Slave ID"는 설계 전 고려된 IP의 특성, Master와 Slave의 관계, Latency에

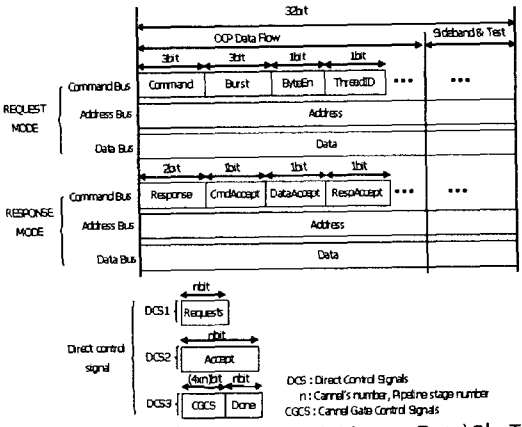

그림3. Pipeline 버스(Command, Address, Data)와 직접 제어를 위해 할당된 신호와 제어 정보 형식

따라 결정된다. "Bus Channels"은 버스 채널의 번호를 나타내고, Pipeline stage수에 비례한다. Central arbiter는 표1에 주어진 ID로 각 Stage의 3-State 버퍼의 스위치를 On/Off 시킴으로 버스의 채널을 제어한다. Central arbiter는 ROM으로 "Master ID", "Slave ID"(Bus Channels)가 저장하고, 각 ID만으로 채널을 제어하는 간단한 구조를 가진다. 시스템 버스의 구조로는 최대 4개의 다중 작업이 가능하며, 각각의 Master와 Slave의 배치에 따라서 그 시스템의 효율이 결정된다. 이러한 3-state 버퍼의 on/off를 위한 스위치는 latency를 줄이고, 성능 향상을 위하여 직접 제어를 한다. 높은 효율을 위해서는 버스 점유율에 따라 그림2와 같은 파이프라인 stage에 Master와 slave를 배치하고, 표1에서와 같이 ID를 지정한다. 같은 "Master ID"와 "Bus Channels"를 가지는 경우가 가장 빈번히 사용되는 IP들로 배치되는 것이 효율이 좋다.

그림3은 제안된 버스에 할당된 신호의 형식을 나타내는 것으로 Request와 Response를 위한 파이프라인 버스로 나누어진다. Request는 Master에서 Slave로의 전송을 의미하고, Response는 slave에서 master로의 전송을 나타낸다. 이들 전송은 앞서 설명한 양방향 버스의 구조를 가지고 있으며 Request시 Command, Address, Data의 3개의 버스 구조로 설계되었고, response시에도 동일한 3개의 버스 구조로 설계된다. 이들 버스들은 그림3과 같은 OCP를 지원하는 신호들로 구성되어 있다. Response시의 버스 구조는 request와 동일한 구조를 가지고 있으며, 서로 대칭적이다. 이러한 버스의 제어를 위한 제어 신호의 형태는 그림3에서 보여준다. DCS1, DCS2, DCS3(Direct control signals, Done)는 master와 local arbiter, local arbiter와 central arbiter, central arbiter와 slave, central

arbiter와 channel gate block(그림2 참고) 으로 연결되며, 각 bit는 pipeline의 stage수에 의해 결정 된다.

본 논문에서 구현된 구조는 4 master와 6 slave의 IP를 가지며, 4 stage pipeline의 4개 채널을 가지는 버스로 구현 되어있다. 이 구조는 pipeline의 추가하거나 central arbiter의 수정 없이 Local arbiter의 수정으로 Master와 Slave의 확장이 가능한 구조이다

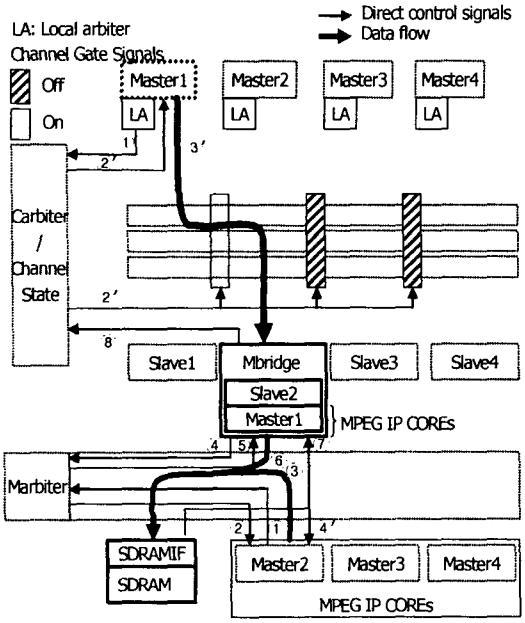

그림4. 시스템 버스에서 MBUS로의 데이터 흐름과 직접 제어 신호의 흐름

그림4는 메모리 버스와 시스템 버스와의 전송의 흐름을 간략히 보여준다. 그림4에서 MPEG의 동작이 이루어지는 ①의 Request가 있고, 다른 시간상에서 시스템 버스의 Master에서 ①'의 request가 발생하였다면, MBUS의 ②의 승인 신호와 ③의 데이터 쓰기 동작이 이루어진다. 이 MPEG의 동작이 계속 일어나는 동안 시스템 버스의 ②'의 승인 신호와 채널 게이트 제어 신호를 보내고, ③'의 데이터 쓰기 동작이 발생한다. Mbridge는 이러한 Request 신호를 버퍼에 저장하고 다시 MBUS의 arbiter로 Request를 하게 되고, 만약 ④의 MPEG동작이 끝나거나, 메모리를 접근할 수 있는 순위를 할당 받게 되면, Mbridge의 버퍼에 저장된 데이터 ⑥ 이 출력되어 메모리를 쓰게 된다. 메모리에 데이터 쓰기가 완료하게 되면, ⑦의 완료 신호를 보내게 되고, 다시 Mbridge를 통하여 ⑧의 완료 신호를 시스템 버스의 arbiter로 보내면서 전송이 종료된다.

Ⅲ. 실험 및 결론

제안된 버스구조는 OCP의 데이터 흐름 신호들 중 기본 신호들을 이용하여 시뮬레이션을 하고 검증 하였다. 그림5는 Master1이 채널4를 동작하는 worst case의 결과를 보여준다. 이 동작에서 최대 4사이클의 latency가 필요하다. 하지만 시스템의 특성에 따른 IP 배치에 따라 성능이 달라지므로 IP 배치의 고려가 시스템의 성능에 영향을 끼친다는 것을 알 수 있다.

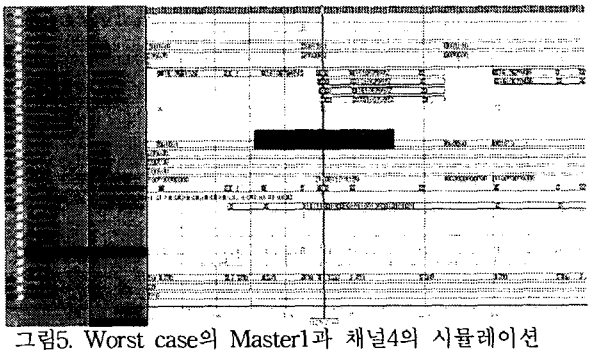

그림5. Worst case의 Master1과 채널4의 시뮬레이션

본 논문은 MPEG 시스템에서 MPEG 동작의 성능 감소 없이 다중 작업이 가능한 새로운 동기 세그먼트 버스 구조를 제안하였다. 제안된 버스 구조는 IP의 재사용 가능한 구조로 이루어져 있다. 직접 제어 신호와 파이프라인 버스 구조는 높은 성능을 얻을 수 있고, 양방향 버스의 구조로 wires의 감소를 얻을 수 있다. Testability를 해결하고자 단 방향 버스 구조로 대체 가능하도록 설계 되었다. 또한, Master의 추가가 필요할 때는 Local arbiter의 수정만으로 확장이 가능한 구조이다.

참 고 문 헌

[1] Michael keating, Pierre Bricaud, "Reused methodology manual for system-on-a-chip designs", KAP, 1998

[2] IBM, The CoreConnectTM Bus Architecture, http://www.arm.com, 1999

[3] Sonics, SONICS uNetwork Technical Overview, http://www.sonicsic.com, 2002

[4] OCP International Partnership. Open Core Protocol Specification, http://www.ocpip.org , July 2002

[5] Plosila, J, Seceleanu, T. Liljeberg, P, "Implementation of a self-timed segmented bus", Design & Test of Computers, IEEE , Volume: 20 , Issue: 6 , Nov.-Dec. 2003

[6] Ruibing Lu; Cheng-Koh Koh,"SAMBA-bus: A high performance bus architecture for system-on-chips", Computer Aided Design, 2003. ICCAD-2003. International Conference on , 9-13 Nov. 2003