32 비트 멀티미디어 RISC CPU 를 위한 버스 인터페이스 유닛의 설계

조 영 록, 안 상 준, 이 용 석
연세대학교 전자공학과
james@dubiki.yonsei.ac.kr

VLSI Design of a Bus Interface Unit for a 32bit RISC CPU

Young-Rog Jo, Sang-Jun An, Yong-Surk Lee
Dept. of E.E. Yonsei Univ.

ABSTRACT

This paper describes a bus interface unit which is used in a 32bit high-performance multimedia RISC CPU including DSP unit. The main idea adopted in designing is that the bus interface unit enables the processor to provide on-chip functions for controlling memory and peripheral devices, including RAS-CAS multiplexing, DRAM refresh and parity generation and checking. The number of bus cycles used for a memory or I/O access is also defined by the processor, thus, no external bus controllers are required. All memories and peripheral devices can be connected directly, pin to pin, without any glue logic. That is the key point of the design.

1. 서 론

본 논문은 32 비트 고성능 멀티미디어 RISC 형 CPU 를 위한 버스 인터페이스 설계를 주제로 한다. 버스 인터페이스의 기초적인 기능은 외부에서 데이터를 읽고 다시 계산된 결과를 외부로 쓴다거나 외부에서 발생한 중요 신호들, 예를 들면 리셋(reset)이나 인터럽트(interrupt) 등을 수용하여 내부의 처리 장소로 보내 주는 것이다. 이러한 것 외에 여러 기능들이 있으나 연구의 가장 중요한 점은 내부 코어가 어떠한 외부 환경과 연결 된다 하더라도 추가적인 논리 회로(glue logic) 가 필요 없이 바로 핀과 핀(pin to pin)으로 동작 할 수 있다는 것이다. 그러기 위해서 타이머(timer)와 버스 콘트롤러(bus controller), watch dog timer 등을 내장한다. 구체적으로는 독립된 주소 버스(address bus)와 데이터 버스(data bus), 4 개의 주소 영역(address space)을 두어서 다양한 종류의 메모리를 다양한 버스크기(bus size)로 지원 하며 이러한 동작에 대한 자세한 기술(description)이 전역 레지스터(global register)내에 위치한 레지스터에 정의 된다. 모든 메모리와 외부 주변 장치(I/O)를 위한 제어 신호가 내부적으로 프로세서에서 자체 생성되는 기능을 지원한다. 또한 최대 7 개까지의 벡터 인터럽트(vectored interrupts)를 받아들이도록 설계 했다.

2. 버스 인터페이스 유닛의 구조

버스 인터페이스는 크게 세 부분으로 나누어 질 수 있다.

1. 어드레스 로직
2. Data I/O
3. 어드레스 제어단

어드레스 로직은 정수 연산 유닛(ALU)이나 인스트럭션 프리페치를 위한 26bit 의 주소(address)가 통과하는 경로(data path)이다. 주소는 외부 SRAM, ROM 에 접근하

※ 본 연구의 일부는 과학 재단의 특정 기초 연구 과제 연구비 지원으로 이루어 졌음. (97-0100-0701-2)

거나 DRAM 접근을 위한 경로, 그리고 내부 메모리를 접근하기 위한 경로 등 크게 세가지 경로로 구분된다. Data I/O 는 크게 데이터를 읽어오거나 쓰기 위한 두 개의 부분으로 이루어진다. 칩 내부의 비트 수는 32 비트로서 워드(word) 크기의 데이터를 사용할 수 있지만 외부의 버스는 8, 16, 32 비트 등이 될 수 있기 때문에 데이터를 칩 밖으로 써 주거나 읽어온 데이터를 사용하기 전에 정열(align) 해 줄 필요가 있다. 어드레스 로직과 data I/O 의 동작은 서로 밀접하게 연관되어 있고 특히 주소가 나가는 시간과 데이터가 나가는 시간이 규정된 시간 내에 동작하도록 해 줄 필요가 있으며 이것은 버스 인터페이스에서 대단히 중요한 요소이다. 따라서 이러한 타이밍 관련 제어단으로서의 어드레스 제어단의 역할은 매우 중요하다. 그림 1 은 버스 인터페이스 유닛의 간단한 데이터 흐름을 나타낸 그림이다.

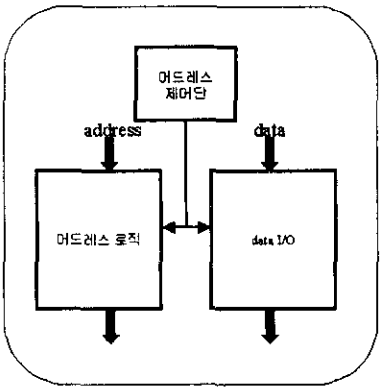


그림 1. 버스 인터페이스 유닛의 간단한 데이터 흐름

어드레스 로직과 data I/O 는 실제적으로 주소와 데이터(명령어 포함)가 이 곳을 통해서 나가고 들어오는 경로가 된다. 어드레스 제어단이 앞의 두 유닛을 제어해서 데이터가 원하는 곳으로 원하는 시간에 갈 수 있도록 해준다. 따라서 이 세 부분은 어느 하나를 따로 떼어서 생각할 수 없으며 각종 신호를 서로 주고 받게 된다.

데이터의 흐름에 있어서 실제로는 파이프라인의 각 단계에 맞도록 수행되어야 하므로 어드레스 제어단에서 언제 데이터를 읽고 써야 하는 지를 알고 있어야 하며 이때에 필요한 신호들을 모두 만들어 주어야 하므로 실제로는 어드레스 제어단이 하는 일이 가장 많다고 볼 수 있다. DRAM 을 포함한 외부 메모리를 접근하는데 필요한 모든 신호들, 즉 외부로 나가는 RAS#, CAS#, WE#, OE#, CHIP SELECT, IORD#, IOWR# 등이 모두 어드레스 제어단에서 만들어 진다. 각 부분을 좀더 구체적으로 살펴보면 다시 8 개 정도로 나눌 수 있으며 먼저 data I/O 는 그림 2 에서 보는 바와 같이 읽기와 쓰기(LOAD/STORE)로, 다시 읽기 유닛은 데이터를 내부 메모리(IRAM), 외부 메모리(EMEM), 명령어 캐쉬(I-CACHE) 등 어느 곳에서 읽어오는 가에 따라서 3 부분으로 어드레스 제어단은 메모리의 종류에 따라서 SRAM, DRAM 제어 유닛과 주변 장치(I/O) 접근을 위한 주소 제어 유닛 등 3 개의 유닛으로 구성된다.

버스 인터페이스는 이러한 주요 기능 외에 부가적인 기능들이 있다. 4 개의 인터럽트 핀(INT1~INT4)을 통해서 들어오는 신호를 받아서 해당 처리 블록으로 보

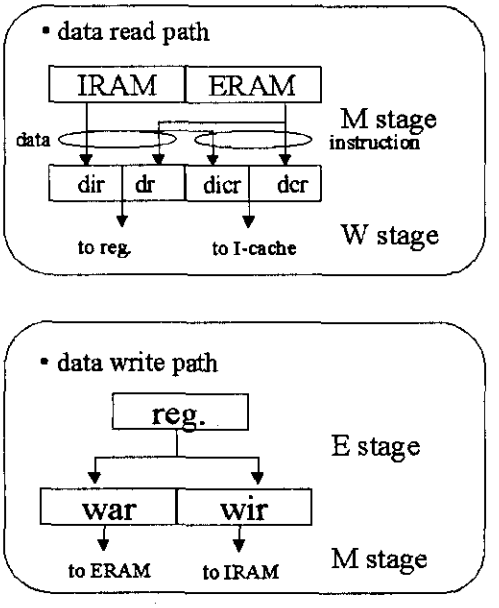


그림 2. 읽기와 쓰기(LOAD/STORE)

내주는 것과 3 개의 입출력 핀(IO1~IO3)을 이용해서 기본 입출력 외에 인터럽트 처리 신호로 해석해 주는 것

이 가장 대표적인 것이며 이외에도 외부에 DMA(Direct Memory Access)가 있는 경우나 멀티 프로세스 환경에서 외부 버스 사용의 권한을 부여 받는 버스 중재(bus arbitration) 등이 추가적인 기능이다.

3. 시뮬레이션 검증 및 합성

본 논문의 버스 인터페이스 유닛을 검증하기 위해서 버스 인터페이스 유닛을 Verilog-HDL 을 이용하여 기술하고 시뮬레이션 하였다. 버스 인터페이스 유닛의 전체 10 개 모듈을 behavioral 및 structural 수준에서 기술하고, 그 모델들의 시뮬레이션을 수행하여 각 모듈을 검증한 후 다시 전체를 통합하여 전체 버스 인터페이스 유닛에 대한 검증을 하였다.

버스 인터페이스 유닛의 HDL 은 어드레스 로직을 위한 하나의 모듈과, data I/O 와 관련하여 5 개의 모듈, 어드레스 제어를 위한 3 개의 모듈, 나머지 버스 중재를 위한 1 개 등 총 10 개의 모듈로 되어 있다. 각 모듈은 게이트(gate)와 MUX, 플립플롭(flip flop)을 이용하여 behavioral 과 structural 수준이 혼합되어서 기술되었으며 시뮬레이션 모듈에 의해서 발생된 입력 벡터를 각 모듈의 HDL 모델로 입력하고, 그에 따른 신호들의 파형을 관찰하여 검증을 수행하였다. 실제적인 검증을 위해서는 프로세서의 나머지 부분들도 필요하고 그에 따라서 명령어를 해석하고 발생되는 주소로서 메모리에 접근해서 계산된 데이터를 가지고 하는 검증이 필요하나 여기서는 우선 주소 값과 그에 해당하는 데이터 값을 각 모듈의 와이어(wire)와 레지스터(register)를 이용하여 임의로 입력시켰다.

각 모듈별로 검증을 수행한 후 통합 모듈을 만들어 버스 유닛 전체에 대한 검증을 수행하였다. 통합 시뮬레이션을 위해서는 반드시 버스 인터페이스 유닛 외에도 명령어 해석기를 비롯한 여러 다른 유닛과 내부 메모리, 외부 메모리들이 필요하나 그렇게 하지 못하고 외부 입력 신호들을 통해서 타이밍에 관계없이 필요한 값들을 입력시켰다. 시뮬레이션의 결과로 그림 3 에 2 개의 파형(wave)을 나타내었다. 처음의 파형은 주소 영역 3 의 외부 SRAM 을 단일 싸이클 내에 접근해서 8 비트의 데이터 버스를 이용해서 읽거나 쓰는 그림으로 각기 WE0#~WE3#과 OE# 신호가 필요하다. 두 번째 파형은 외부 DRAM 접근을 보여주는 파형으로 ras precharge time, ras to cas delay time, cas access time 이 모두 4 싸이클인 경우이다.

4. 버스 인터페이스 유닛의 합성

HDL 을 자동 합성하여 표준 셀(standard cell) 방식의 배치와 배선(placement & route)을 거쳐서 레이아웃을 완성했다. 합성된 네트리스트를 이용하여 측정한 최대 지연 경로(critical path)는 상승 시 13.40ns, 신호 하강 시 13.77ns 로서 평균적으로 13.58ns 로 최대 73.6Mhz 의 동작 주파수가 가능하다. 최적화된 레이아웃을 수행하기 위해서 비교적 게이트 수가 많은 어드레스 로직과 data I/O 부분은 하위 모듈을 각각 레이아웃 했으며 나머지는 하나의 블록으로 묶어서 전체 레이아웃을 수행하였다. 레이아웃 결과 Gate Count 는 어드레스 로직이 1500.7, data I/O 부분이 3654.3 이고 나머지가 982.5 로 전체

그림 3-1. 시뮬레이션 파형

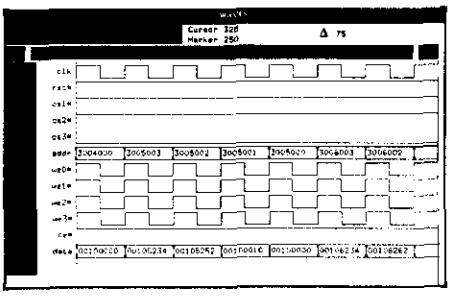
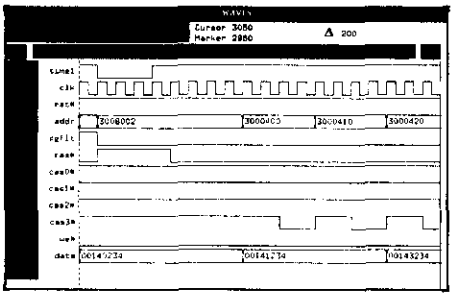


그림 3-2. 시뮬레이션 파형



6137.5 이며 net 당 2.3 개의 평균 핀 수(Total number of pins / Total number of nets)를 가지며 패드를 제외한 전체 유닛의 크기는 1791 × 1586.7 um^2 이다.

버스 인터페이스 유닛의 합성 결과는 표 1 과 같다

표 1. 버스 인터페이스 유닛의 합성 결과

내용	결과
최대 동작 주파수	73.6MHz@5.0volt
게이트 수	어드레스 로직:1500.7 개
	data I/O:3654.3 개
	그 외:982.5 개
	전체:6137.5 개
면적	1.791mm × 1.587mm
최대 지연 시간	13.40ns(rising edge) 17.77ns(falling edge)
동작 전압	5.0 volt

5. 결 론

본 논문은 32 비트 고성능 멀티미디어 RISC 형 CPU 를 위한 버스 인터페이스 설계를 주제로 하였다. 설계의 핵심으로 프로세서가 어떠한 환경에서도 외부에 추가적인 회로를 필요로 하지않고 바로 동작할 수 있도록 했으며 이를 위해서 내부에 2 개의 32 비트 레지스터를 두어서 시스템 초기화 시 레지스터에 실리는 값에 따라서 하드웨어의 수정 없이 프로그램의 수정만으로도 동작이 가능 하도록 했다. 이 들을 뒷받침하기 위해서 버스 인터페이스 유닛은 RAS to CAS 멀티플렉싱과 DRAM refresh, 패리티 비트 생성과 검사를 포함한 메모리와 외부 주변 장치들을 제어하기 위한 모든 기능들을 온 칩(On Chip)화 시켰다. 메모리와 I/O 의 접근을 위해서 사용되는 버스 싸이클은 프로세서 내에 정의되므로 어떠한 외부 버스 제어기도 필요로 하지 않는다. 메모리와 주변 장치와는 추가 회로 없이도 핀과 핀으로 바로 연결이 가능하도록 했다. 본 인터페이스 유닛의 HDL 모델은 로직 수준의 설계를 고려하여 캐드 툴(CAD tool)을 이용하여 자동 합성이 가능하도록 기술 되었으며, 회로 합성과 P & R 을 거치면서 설계된 네트리스트(netlist)를 이용하여 0.6um triple metal layer CMOS 공정을 이용하여 레이아웃 하였다. 이렇게 설계된 전체 1.791mm × 1.587mm(패드 제외)의 크기로 총 6137.5 개의 게이트를 가지며 최대 73.6MHz 에서 안정적으로 동작한다.

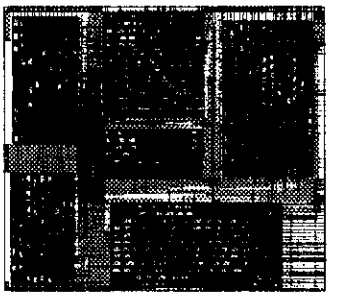


그림 4. 버스 인터페이스 유닛의 레이아웃

참고 문헌

[1] Michael Dolle, Satwinder Jhand, Walter Lehner, Otto Muller, Manfred Schlett, "A 32-b RISC/DSP Microprocessor with Reduced Complexity", *IEEE Journal of Solid –State Circuits*, vol. 32, No. 7, July. 1997.

[2] Jim Turley, "Hyperstone Merges CPU and DSP Cores – Fixed Point DSP Shares Instructions and Registers With Image Core", *Micro Processor Report*, April 21, 1997.

[3] Michael Dolle, Manfred Schlett, "A Cost Effective RISC/DSP Microprocessor for Embedded Systems", *IEEE Micro*, October, 1995.

[4] 최병윤, "명령어 축소형 32 비트 마이크로프로세서의 VLSI 구현", 연세대학교 전자공학과 박사학위 논문, 1992 년.

[5] 유회준, *DRAM 의 설계*, 홍릉 과학 출판사, pp 71-107, 1996.

[6] 송진흥, 김남영, 김동욱, 이재철, *VLSI 설계, 이론과 실습*, 홍릉 과학 출판사, pp 295-357, 1997.

[7] COMPASS Design Automation, *ASIC synthesizer for Verilog Design*, 1991.