1999 年度 大韓電子工學會 夏季綜合學術大會 論文集 第 22 卷 第 1 號 99/6

32 비트 RISC 마이크로프로세서를 위한 버스 인터페이스 제어기의 설계

허 상 경, 안 상 준, 정 우 경, 김 영 준, 이 용 석
연세대학교 전자공학과
서울시 서대문구 신촌동 134 번지
Tel 02)361-2872, Fax 02)312-4584
skyong@dubiki.yonsei.ac.kr

VLSI Design of a Bus Interface Controller for 32-bit RISC microprocessor

Sangkyong Heo, SangJun An, Wookyeong Jeong, Youngjun Kim, Yongsurk Lee
Dept. of Elec. Eng., Yonsei University
134, Shinchon-dong, Seodaemun-gu, Seoul
Tel 02)361-2872, Fax 02)312-4584
skyong@dubiki.yonsei.ac.kr

요 약

본 논문에서는 DSP 기능을 내장한 32 비트 RISC 마이크로프로세서를 위한 버스 제어기를 설계하였다. 연구의 초점은 버스 타이밍, 주소 멀티플렉싱, 리프레쉬, 버스 중재 등을 제어하는 버스제어기를 온칩화 하여 CPU 로 하여금 외부 램과 추가적인 장치없이 직접 연결될 수 있도록 한 것이다. 버스 제어기가 관리하는 메모리의 종류는 SRAM, ROM, DRAM, EDO DRAM 이며 고속 모드(Fast page mode, EDO page mode 및 RAS-down mode)기능을 지원하며 다양한 Wait 를 넣을 수 있다. 주소 영역은 4 가지(EMA0~EMA3)이며 내부적으로 7 개의 레지스터가 있고 이들을 이용하여 서로 연결된 세 개의 상태 머신으로 모든 램과의 타이밍을 제어함으로써 공유블록을 활용할 수 있었다. Verilog HDL 로 기술하고 Synopsys 로 합성한 후 타이밍 검증을 수행한 결과 최악조건에서 53.1Mhz 로 동작할 수 있었다. 그 후 0.6um single poly triple metal process 공정으로 레이아웃 되었고 면적은 1.44mm x 1.21mm 이다.

1. 서 론

마이크로프로세서의 고속화 추세에 부응하여 연세대학교 아식설계공동연구소는 1989 년부터 RISC 마이크로프로세서 설계 및 연구를 수행해 왔으며 이중 본 연구의 CPU 인 YS-RDSP(Yonsei-RISC DSP)마이크로프로세서는 DSP 유닛을 내장한 32 비트 RISC 마이크로프로세서이다[1]. 마이크로프로세서는 외부 장치와 밀접한 관계를 가지므로 코어의 성능향상만으로는 전체 시스템의 성능향상에 한계가 있다. 따라서 느린 외부 장치들을 효율적으로 관리할 수 있고 구조와 기능이 특정 CPU 의 사양에 최적화된 버스 인터페이스 제어기의 설계가 중요한 이슈로 작용한다.

본 논문에서 제 2 장은 설계된 버스 인터페이스 제어기의 구조에 대한 내용이 기술된다. 이어 제 3 장에서는 합성 및 타이밍 검증을 하고 레이아웃한 결과를 기술한다. 그리고 제 4 장에서 결론을 맺는다.

2. 버스 인터페이스 제어기의 구조

2.1. 전체 구조

설계된 버스 제어기의 전체 블록도를 그림 1 에 도시하였다. 내부적으로, 코어와의 데이터와 어드레스를 주고받는 IAB IDB interface block, 버스 제어기를 포함한 주변장치들의 레지스터들의 읽고 쓰기를 관리하는 Register R/W control block, 주변장치들의 어드레스와 데이터 버스인 PAB(Peripheral address bus), PDB(Peripheral data bus)와의 연결을 관리하는 PAB PDB interface block, 각 메모리 영역의 칩 선택 신호를 관리하는 EMA control block, 선택된 메모리 영역에 대한 세부적인 신호와 칩 외부의 어드레스, 데이터 버스를 관리하는 Individual EMA control block, 리프레쉬를 제어하는 Refresh control block, 그리고 이 모든 블록들의 타이밍을 상태머신으로 제어하는 State machine block 으로 나누어서 설계하였다. 또한 연결된 램종류, 버스폭등의 정보를 7 개의 내부 레지스터에 기록해서 참조하도록 설계하였다.

※ 본 연구의 일부는 1998 년도 한국 학술진흥재단 대학부설 연구소과제 연구비에 의하여 연구되었음.

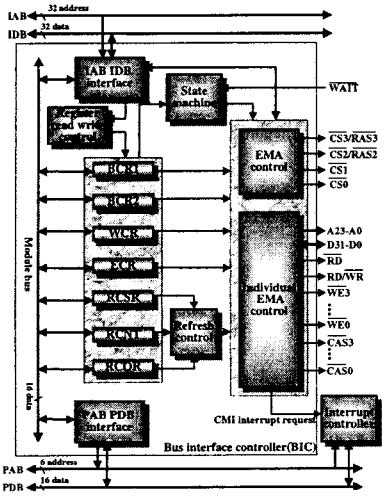

그림 1. 버스 인터페이스 제어기의 블록도

2.2. 상태머신

본 연구에서는 SRAM, ROM, DRAM, EDO DRAM 의 각 메모리 인터페이스에 대한 버스 타이밍과 주소 재생성, 주소 멀티플렉싱, 데이터 정렬 등의 모든 타이밍 정보를 중앙의 상태머신이 제공하도록 설계되어 공유 블럭을 최대한 활용하는 구조로 설계되었다[2].

상태머신은 세부적으로 Master 상태머신, SRAM 상태머신, DRAM 상태머신으로 나누어지는데 그림 2 는 이 중 DRAM 상태머신이다. 이것은 DRAM 뿐 아니라 EDO DRAM 에도 적용시킬 수 있도록 설계되었으며 Fast page mode, EDO page mode, RAS-down mode, Refresh 등의 DRAM 타이밍을 통합적으로 제어한다. 또한 DRAM, EDO DRAM 엑세스 타이밍에 삽입되는 wait state 와 리프레쉬 타이밍에 삽입되는 wait state 를 공유함으로써 하드웨어적인 복잡도를 최소화 하도록 설계되었다.

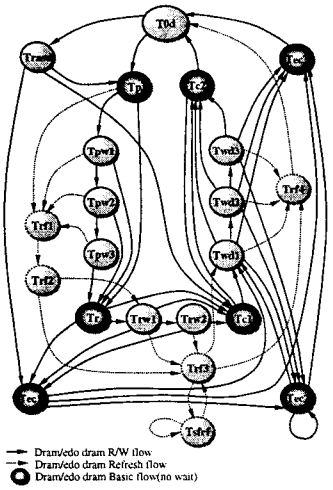

그림 2. DRAM 상태머신의 상태천이도

2.3. 주소 재생성/멀티플렉싱

설계된 버스 제어기는 엑세스 크기와 버스 폭에 따른 데이터의 정렬과 어드레스의 재생성을 제어하게 된다[3]. 예를들어 32 비트 엑세스인데 버스폭이 8 비트이면 주소의 최하위 두 비트를 00, 01 10, 11 로 재생성해서 4 번 엑세스해야 한다. 이러한 멀티 엑세스의 경우마다 상태머신을 참조하여 그림 3 과 같이 IAB 의 최하위에 1'b1(addr+1), 2'b10(addr+2), 2'b11(addr+3)을 넣어 A[23:0]를 구성하게 된다.

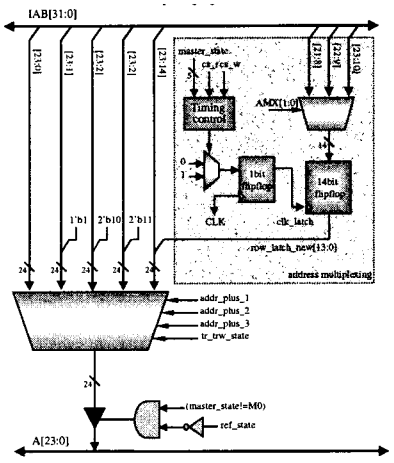

그림 3. 주소 재생성/멀티플렉싱 블록도

그림 3 의 검은 부분은 DRAM 엑세스에서 필요한 주소 멀티플렉싱을 수행하는 블록이다. 내부 레지스터 중 AMX 비트가 00 이면 컬럼주소가 8 비트, 01 이면 9 비트, 10 이면 10 비트로 정의하였다. 표 1 에 주소 멀티플렉싱을 나타내었다.

표 1. 주소 멀티플렉싱

AMX [1:0]	No. of Column Addr. bits	Page size	Row Addr. Output on A13~A0	Column Addr. output on A13~A0
00	8	256B	Addr. bit 21~8	Addr. bit 7~0
01	9	512B	Addr. bit 22~9	Addr. bit 8~0
10	10	1KB	Addr. bit 23~10	Addr. bit 9~0

2.4. 데이터 정렬

램에서 데이터를 읽어올 때나 쓸 때 엑세스폭과 버스폭의 관계에 따라 정렬과정이 필요하다. 본 연구의 프로세서는 빅 엔디언(Big endian)방식으로 데이터가 처리된다. 즉 데이터를 읽을 때 00 번지의 데이터가 Most Significant Byte 즉 IDB[31:24]로 들어가고 11 번지의 데이터가 Least Significant Byte 즉 IDB[7:0]로 들어가는 방식이다[4]. 쓸 때도 마찬가지로 빅 엔디언 방식이 적용된다.

그림 4 에 읽기 정렬 블록도를 도시하였다. Timing control 블록에서는 상태머신의 정보를 이용하여 래치할

타이밍을 제어한다. 8 비트 플립플롭 3 개가 읽어온 데이터를 래치하는데 사용된다. 예를들어 32 비트 엑세스인데 버스폭이 8 비트이면 4 번 엑세스를 해야 하는데, 3 번째까지의 읽어온 데이터를 각각 래치하고 마지막 엑세스의 데이터와 함께 32 비트를 만들어 IDB 에 실어주도록 설계하였다.

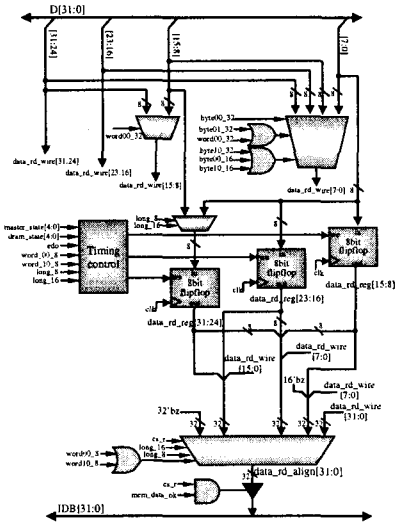

그림 4. 데이터 읽기정렬 블록도

데이터를 램에 쓸 때도 빅 엔디언 방식으로 정렬되어서 쓰여지도록 설계하였다. 예를들어 엑세스폭이 32 비트인데 버스폭이 8 비트이면 사용가능한 외부 데이터 버스는 8 비트뿐이므로 32 비트의 데이터를 바이트별로 나누어서 4 번에 걸쳐 엑세스를 해야한다. 이 과정에서 재생성된 주소와 나누어진 데이터를 매번 엑세스마다 각각 어드레스와 데이터 버스에 실어주도록 설계하였다.

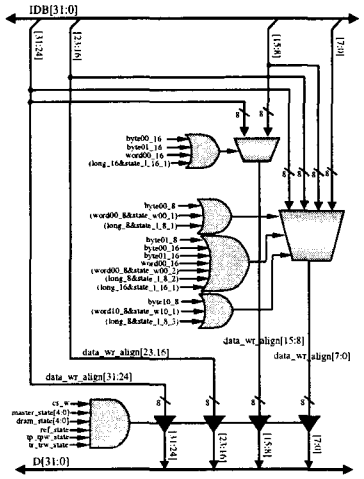

그림 5. 데이터 쓰기정렬 블록도

3. 검증 및 결과고찰

그림 6 에 버스 인터페이스 제어기의 설계 흐름도를 도시하였다. CPU 와 주변 장치와의 입출력과 스펙을 정의한 후 Verilog HDL 로 기술된 버스제어기와 CPU 를 합체한 후 CADENCE 의 SimWaves 를 이용해서 시뮬레이션 및 검증을 하였다. 기능 검증이 완료된 HDL 을 Synopsys 를 이용해서 합성하고 여기에서 타이밍 정보를 SDF 파일로 뽑아내고 게이트 레벨의 HDL 을 추출해서(HDL conversion) 타이밍 검증(Back annotation)을 하였다[5][6]. HDL 시뮬레이션 파형과 타이밍 검증 파형을 비교하여 다르면 다시 HDL 을 수정하고 위의 과정을 반복하였다. 최종적으로 통과된 네트리스트를 COMPASS 툴을 사용하여 0.6um single poly triple metal process 공정으로 레이아웃 하였다.

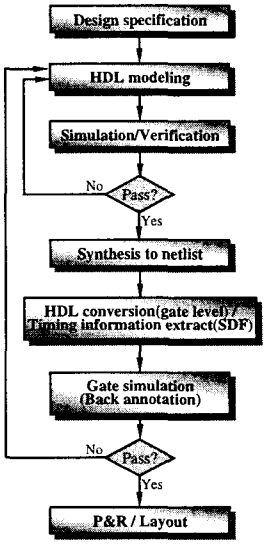

그림 6. 설계 흐름도

표 2 에 합성 및 레이아웃 결과를 나타내었다.

이 결과는 본 논문의 CPU 인 YS-RDSP 의 최악조건 동작주파수 30Mhz 이상, 정상조건 동작주파수 40Mhz 이상을 만족한다.

표 2. 합성 및 레이아웃 결과

항 목	내 용
게이트 수	6002 개
최대지연시간	18.83 ns(최악) / 10.51 ns(정상)
최대동작주파수	53.1 Mhz(최악) / 95.2 Mhz(정상)
면적(mm x mm)	1.44 x 1.21
동작 환경	3.0volt,85°C,slow process (최악조건) 3.3volt,25°C,typical process(정상조건)

본 논문의 버스제어기와 다른 버스제어기들과의 비교와 성능평가를 표 3 에 나타내었다.

YSKY16 의 버스제어기는 제어할 수 있는 램이

DRAM 뿐이고 어드레스 영역이 1 개 이므로 비교적 간단하고 게이트 수가 제일 작다. 그리고 GMS 의 버스 제어기는 제어하는 램이 SRAM, ROM, DRAM 으로써 본 논문에서의 EDO DRAM 은 빠져 있지만 주소 영역이 5 개로써 본 논문보다 많다. 그러나 주소 영역의 추가는 이미 존재하는 램 콘트롤러를 해당 메모리 영역으로 스위칭하면 되지만, 메모리 종류를 추가한다는 것은 그 메모리의 스펙에 따른 버스 타이밍들을 모두 생성시켜야 하므로 추가되는 로직이 아주 많게 된다.

또한 wait 사이클의 삽입면에서, GMS 버스 제어기는 각 주소 영역에 따라 wait 사이클이 고정되어 있다. 이에 비해 본 버스제어기는 S/W 적인 방법으로 최대 10 개의 wait 를 삽입할 수 있고, 또한 H/W 적인 WAIT 신호를 이용해서 wait 를 추가할 수 있게 설계되었다.

종합하면, 본 논문의 버스 제어기는 대부분의 제어 회로를 중앙의 상태 머신으로 압축하는 구조를 채택하여 공유블록을 최대한 활용함으로써 비교적 적은 게이트 수와 높은 동작 주파수를 얻을 수 있었다.

표 3. 비교 및 성능평가

항목	본 논문의 버스제어기	GMS 버스 제어기	YSKY16 버스제어기
제어하는 RAM 종류	SRAM, ROM, DRAM, EDO DRAM	SRAM, ROM, DRAM	DRAM
어드레스영역	4 개	5 개	1 개
버스타이밍 제어	상태 머신	카운터	상태 머신
최대 동작주파수	95.2 Mhz	73.6 Mhz	74.6 Mhz
게이트 수	6002 개	6138 개	4191 개
합성 라이브러리	0.6 um 표준 셀 라이브러리		
동작 환경	3.3 volt, 25°C, typical process(정상조건)		

그림 7 은 코어와 함께 HDL 레벨에서 시뮬레이션한 결과 파형이다. SRAM 이며 엑세스폭과 버스폭은 32 비트이고 wait 가 삽입되지 않은 경우이다. 코어에서 실행된 명령어는 Rf 레지스터의 값(0001ff70)이 지정하는 번지에서 데이터를 읽어와서 MACL 레지스터에 쓰는 것이다. M1 단에서 어드레스가 나가고 M2 단에서 읽어온후 WB 단에서 MACL 에 쓰여짐을 볼 수 있다.

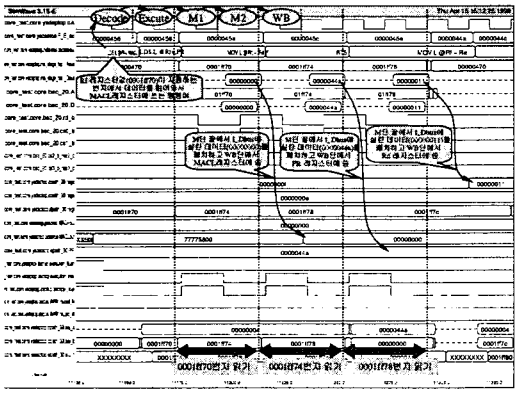

그림 7. HDL 레벨 시뮬레이션 파형

그림 8 은 게이트 레벨의 시뮬레이션 파형이다.

EDO DRAM 이고 엑세스폭은 32 비트, 버스폭은 8 비트이며 wait 가 삽입되지 않은 경우이다.

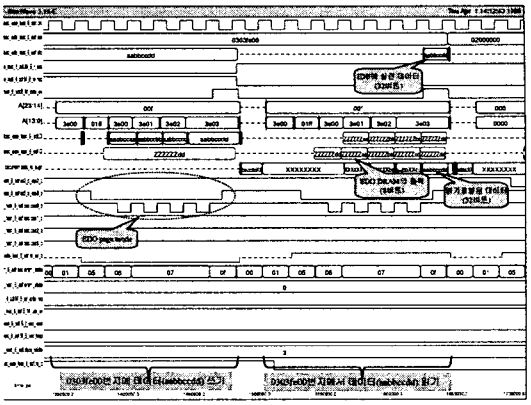

그림 8. 게이트 레벨 시뮬레이션 파형

4. 결 론

본 논문에서는 32 비트 RISC 마이크로프로세서를 위한 버스 인터페이스 제어기를 설계하였다. 외부의 추가적인 회로 없이 여러 종류의 램을 제어하며 여러 가지 고속모드를 지원하는 버스 제어기를 마이크로프로세서에 내장되는 형태(On chip)로 구현하였다. 또한 서로 연결된 세 개의 상태머신으로 램들을 제어함으로써 공유블록을 활용하여 면적을 최소화하였다.

본 연구의 버스 제어기는 Verilog HDL 로 기술되고 기능검증과 타이밍 검증을 거친 후 0.6um single poly triple metal process 공정으로 레이아웃 한 결과 크기는 1.44 x 1.21 (mm x mm)이고 총 6002 개의 게이트수를 가지며 최악조건에서 최대 53.1MHz, 정상조건에서 95.2Mhz 로 동작하는 것을 확인하였다. 이는 본 연구의 CPU 인 YS-RDSP 의 최악조건 동작 주파수인 33MHz 를 만족한다.

5. 참 고 문 헌

[1] 이용석, "RISC Microprocessor Overview", 고성능 마이크로프로세서 구조와 설계 비디오 강좌 시리즈, 1997, http://mpu.yonsei.ac.kr/yslee/RiscOverview.html

[2] Douglas J. Smith, *HDL Chip Design*, Doone Publications, pp.195-262, 1996.

[3] Nikitas Alexandridis, *Design of Microprocessor-Based Systems*, Prentice-Hall, New Jersey, pp. 51-120, 1993.

[4] J.L.Hennessy & D.A.Patterson, "*Computer Architecture, A Quantitative Approach*", 2nd edition, Morgan Kaufmann Publishers, 1996

[5] Pran Kurup & Taher Abbasi, *Logic Synthesis Using Synopsys*, Kluwer Academic Publishers, pp.33-88, 98-196, 1997.

[6] *Design Compiler Refernce Manual : Optimization and Timing Analysis*, Version 1997.01, Synopsys, Inc.,1997.