ARM7호환 32비트 RISC 프로세서의 설계 및 검증

배 영 돈(裵 映 敦), 서 보 익(徐 補 益), 이 용 석(李 龍 錫), 박 인 철(朴 仁 哲)
한국과학기술원 전기 및 전자공학과 VLSI Systems Lab.
전화 : (042) 869-4403 / 팩스 : (042) 869-4410

Design and Verification of an ARM7 Compatible 32-bit RISC Processor

Young-Don Bae, Bo-ik Seo, Youngseok Lee, In-Cheol Park
Department of Electrical Engineering, Korea Advanced Institute of Science and Technology
E-mail : donny@vslab.kaist.ac.kr

Abstract

This paper describes a 32-bit RISC processor, which has instruction level compatibility with the ARM7 microprocessor. The processor is fully synthesizable, and its performance is evaluated based on 0.35-μm CMOS library.

This paper focuses on the implementation of the processor and the reliable verification strategy ensuring the complete instruction level compatibility.

The processor has successfully verified using a FPGA chip.

I. 서론

현재 고속 그래픽 처리기, 네트워크, 가전, 멀티미디어 기기, 그리고 이동통신 등의 많은 시스템에서 사용되는 마이크로프로세서는 데스크 탑용의 범용 프로세서와는 달리 embedded core용으로 개발된 프로세서가 필요하다. 이들 embedded 프로세서는 기본적으로 작은 면적에 집적될 수 있으며 전력소모가 적어야 한다[1]. 또한 target chip을 제작하는 공정에 포팅(porting)이 가능해야 하며 함께 집적하는 블록들과의 동작 검증이 가능해야 한다. 이러한 이유로 다양한 공정에서 제공하며 설계 환경이 잘 구축되어있는 Advanced RISC Machines사의 ARM계열이 많이 사용되고있다[2,3,4,5].

본 논문에서는 Verilog를 사용하여 ARM7 프로세서와 명령어 수준의 호환성을 갖는 마이크로프로세서를 설계하였다. 따라서, 셀 라이브러리를 사용하여 합성하여 어느 공정에나 쉽게 사용할 수 있으며 필요에 따라 수정하여 사용할 수 있다. 따라서, 마이크로프로세서기반의 ASIC 설계에 유용하다. 최근 이와 같은 목적으로 softcore의 개발이 이루어지고 있으며, Advanced RISC Machines사에서도 합성 가능한 형태의 ARM 프로세서를 공급하기 시작하였다[6,7,8].

또한 본 논문에서는 RT(Register Transfer) 레벨 및 게이트 레벨에서 명령어 단위의 호환성 검증과 시뮬레이션을 동시에 빠른 속도로 수행할 수 있는 방법을 제시한다.

설계된 프로세서는 0.35-μm 공정 라이브러리를 사용하여 합성하여 성능을 측정하였으며, FPGA를 사용하여 검증되었다.

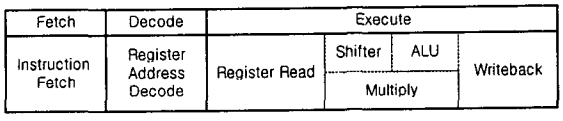

그림 2 명령어 파이프라인 구조
Fig. 1. Instruction Pipeline Architecture

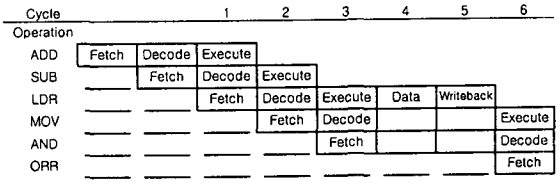

그림 1 명령어 파이프라인의 예
Fig. 2. Example of Instruction Pipeline

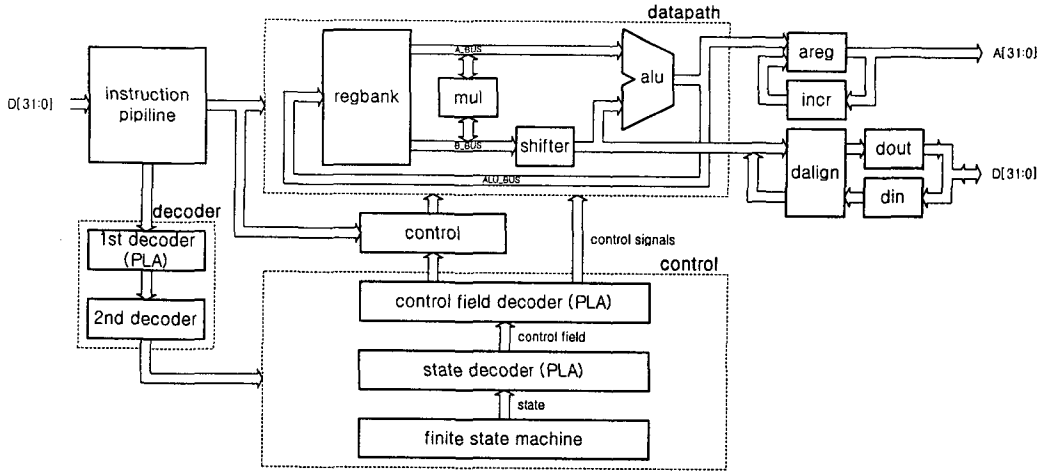

그림 3 설계된 프로세서의 구조

Fig. 2. Processor Architecture

II. 프로세서 구조

명령어 구조는 ARM architecture V4와 동일하다[9].

명령어 파이프라인 구조는 그림에 보여진 것과 같이 세 단계로 이루어져있으며 대부분의 동작은 Execute 단계에서 이루어진다 (그림1). Execute단계는 multi-cycle 구조를 가지고 있으며 대부분의 명령어는 1 cycle에 수행되며 branch, load/store, load/store multiple, multiply 명령어는 최대 17 cycle까지 필요로 한다 (그림2).

프로세서의 전체 구조는 그림에 보여진 것과 같이 크게 세 부분으로 이루어져있다 (그림3).

데이터패쓰 블록은 32비트 레지스터 뱅크와 32비트 배럴 쉬프터와 ALU 그리고 32×8비트 곱셈기로 이루어져있다. 32×8비트 곱셈기는 4 cycle에 걸쳐 32×32비트 곱셈을 수행한다.

디코더 블록은 명령어로부터 레지스터 주소와 컨트롤 블록의 FSM(Finite State Machine)에 필요한 정보 등을 추출한다.

컨트롤 블록은 multi-cycle 동작에 필요한 FSM과 각 상태로부터 데이터 패쓰와 파이프라인을 제어하는 신호를 생성한다.

표 1 합성 결과(0.35 μm 3-metal 공정)

Table 1. Synthesis Results

Block Name	Area [μm^2]
Datapath	1,720,150
- 32-bit register bank	946,736
- 32×8-bit multiplier	284,004
- ALU	139,692
- 32-bit shifter	94,836
Decoder	83,272
Control	329,616
- 1st control	178,052
- 2nd control	151,564
Total	2,184,110

III. 설계 및 검증

설계 및 검증의 흐름은 그림4에 나타난 것과 같다.

전체 설계는 기본적으로 Verilog를 사용하여 기술하였으며 그림 3의 디코더와 컨트롤 블록 중 PLA로 표시되

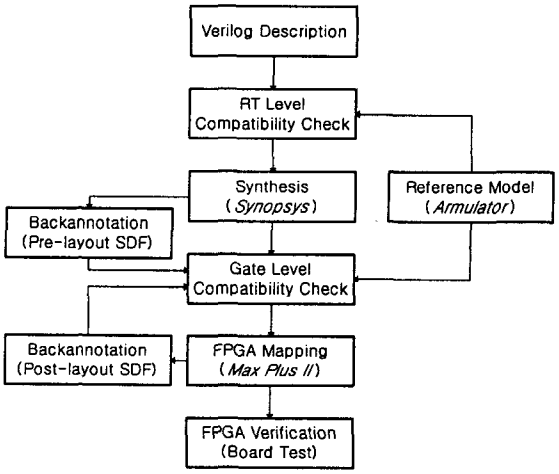

그림 4 설계 및 검증 흐름도

Fig. 2. Design and Verification Flow

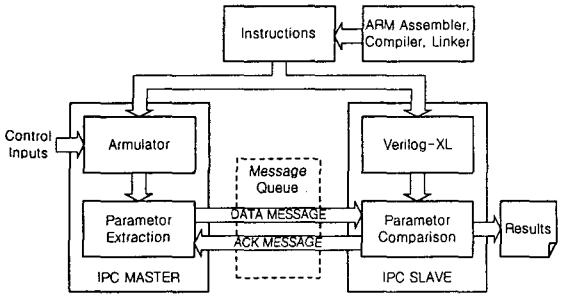

그림 5 명령어 단위의 호환성 검증

Fig 5. Instruction Compatibility Verification

어있는 블록은 PLA(Programmable Logic Array) 형식으로 기술하고 espresso를 사용하여 optimize하였다. PLA형식으로 기술된 블록과 Verilog로 기술된 블록 모두 SYNOPSYS Design Compiler를 사용하여 합성되었다[10].

모든 블록은 합성과정의 편의를 위하여 single rising edge clock구조로 설계하였다. 표1은 0.35-μm 3-metal CMOS공정에서 합성한 결과이다. 합성결과 전체 크기는 약 1.5 mm × 1.5 mm이었다. 만일 데이터패쓰 부분을 표준 셀 라이브러리(standard cell library)를 사용하지 않고 데이터패쓰 라이브러리를 사용하였다면 전체 크기는 1.2 mm × 1.2 mm정도의 결과를 얻을 수 있을 것으로 예상된다.

본 논문에서는 ARM Software Development Toolkit에서 제공하는 ARM 프로세서 시뮬레이터 Armulator를 reference 모델로 사용하여 명령어 단위의 호환성 검증과 Verilog 시뮬레이션과 동시에 수행할 수 있도록 하였다[11].

별개의 프로그램인 Armulator와 Verilog-XL의 결과를 비교하기 위하여 IPC (Inter-process Communication)를 사용하였다[12]. IPC는 UNIX에서 기본적으로 지원한다. 그리고 Verilog-XL의 인터페이스는 PLI(Program Language Interface)를 사용하였다. 호환성 검증을 위한 시뮬레이션 환경은 그림 5와 같다.

각 명령어 단위로 Verilog-XL시뮬레이션 결과(레지스터, 상태 레지스터, 프로그램 카운터 값, 명령어 코드)를 PLI(Program Language Interface)루틴을 통하여 전달하고 동일한 명령어에 대하여 Armulator에서 수행한 결과를 IPC 메시지 큐를 통하여 비교하였다. 이를 통하여 빠른 속도로 정확한 동작을 검증할 수 있으며, 오류가 발생한 위치를 정확히 파악할 수 있으므로 디버깅이 매우 쉽게된다.

표 2 성능평가

Table 2. Performance Evaluation

	ARM7TDMI	This paper
technology	0.35 μm	0.35 μm
die area	2.1 mm^2 (include test circuit)	2.18 mm^2
operating frequency	66 MHz	75 MHz (worst case simulation)

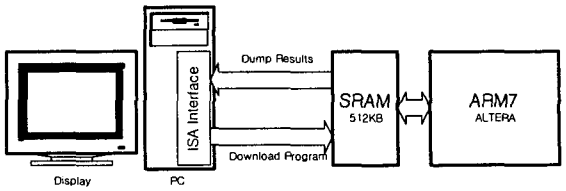

그림 6 FPGA 검증 환경

Fig. 6. FPGA Verification Environment

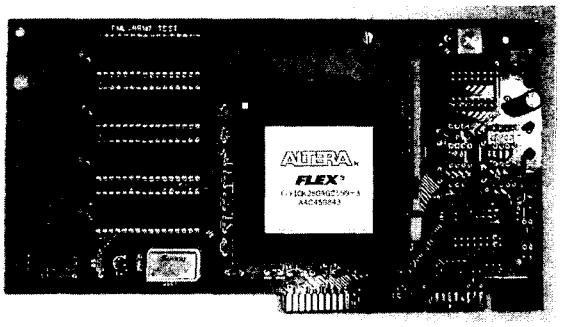

그림 7 FPGA 검증 보드

Fig. 7. FPGA Verification Board

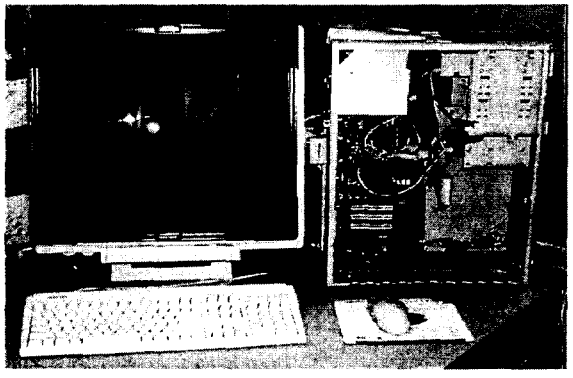

그림 8 테스트 프로그램 수행

Fig. 8. Test Program Results

호환성의 검증은 RT(Register Transfer) 레벨 및 pre-layout 및 post-layout 게이트 레벨 시뮬레이션에서 수행하였다. 시뮬레이션 입력으로는 다양한 어셈블리 및 C코드를 어셈블러, 컴파일러를 통하여 만든 바이너리 코드를 사용하였다.

테스트를 위한 회로를 포함하고 있는 ARM7TDMI의 die area가 0.35-μm공정에서 2.1 mm^2이며 테스트 회로가 전체 면적의 약 30% 차지하는 것을 고려하여 비교하면 설계된 프로세서가 실제 ARM7보다 30%정도 크다. 하지만 앞에서 설명한 것과 같이 데이터패쓰 블록을 optimize하면 비슷한 크기의 결과를 얻을 수 있음을 알 수 있다. 또한 ARM7TDMI의 동작주파수가 66 MHz인 반면 설계된 프로세서는 worst-case 시뮬레이션 결과 75 MHz에서 동작하였다 (표 2)[13].

따라서, 본 논문에서 설계된 프로세서는 semi-custom 방식으로 설계된 점을 감안하면 ARM7TDMI에 비교 가능한 크기를 가지고 있으며 동작 주파수 면에서는 더 우수한 성능을 가지고 있다.

IV. FPGA 검증

설계된 프로세서는 ALTERA의 EPF10K250ARC599-3을 이용하여 검증하였다 (그림 7). 검증용 프로그램을 메모리에 다운로드하여 프로세서가 실행 할 수 있도록 그림 6과 같은 검증환경을 개발하였다. PC에서 ISA버스를 이용하여 SRAM에 테스트 프로그램을 다운로드하고 프로세서에서 수행하였다. 테스트 프로그램은 결과를 다시 메모리의 특정 주소에 저장하도록 제작하여 메모리의 내용을 다시 PC에서 읽어 들여 수행결과를 확인 할 수 있도록 하였다.

C 언어로 이미지 프로세싱 프로그램을 작성하고 이를 컴파일하여 검증 보드상에서 수행한 결과 정확한 결과를 확인할 수 있었다 (그림 8).

V. 결 론

최근 SOC(System-On-a-Chip)와 IP(Intellecutal Property)의 중요성이 부각되는 가운데 embedded 프로세서의 설계기술은 가장 핵심적인 기술 중 하나이다. 프로세서의 성능뿐만 아니라 편리하고 강력한 개발자 환경이 embedded 프로세서의 경쟁력을 결정짓는다. 이러한 상황에서 새로운 명령어 집합을 사용하는 프로세서의 개발보다는 이미 많은 사용자를 확보하고 있는 프로세서와 명령어 수준의 호환성을 갖으며 이를 발전시켜 경쟁력 있는 성능의 프로세서를 개발하는 것은 embedded-market에 접근하는 가장 빠른 방법 중 하나일 것이다. 본 논문에서는 이러한 목적으로 ARM7과 명령어 수준의 호환성을 갖는 32비트 RISC프로세서를 설계하였으며 어느 공정에서나 사용할 수 있는 합성 가능하도록 하였다. 그리고 같은 technology의 ARM7TDMI와 성능을 비교한 결과 동작주파수 측면에서 향상되었으며 die area측면에서도 경쟁력을 갖고 있었다.

또한 본 논문에서는 빠른 속도로 명령어의 호환성 검증과 시뮬레이션을 동시에 수행할 수 있는 방법을 제시하였으며 이를 이용하여 C 컴파일러를 사용하여 제작한 것을 포함하여 다양한 프로그램을 수행하여 RT레벨, 게이트 레벨에서 명령어 단위로 호환성을 검증하였으며 FPGA를 사용하여 검증한 결과 정확한 동작을 확인하였다.

참고문헌

[1] Schlett, M,."Trends in embedded-microprocessor design," Computer, Volume: 31 8, Aug. 1998, p. 44-49

[2] Simon Segars, "ARM7TDMI Power Consumption", IEEE Micro, July 1995, p. 12-19.

[3] ARM7TDMI Datasheet, Advanced RISC Machines, August 1995.

[4] Montanaro, J. *et al.*, "160-MHz, 32-b, 0.5-W CMOS RISC microprocessor," Solid-State Circuits, IEEE Journal of Vol. 31 11 , Nov. 1996 , p. 1703-1714

[5] Segars, S., "The ARM9 family-high performance microprocessors for embedded applications," Computer Design: VLSI in Computers and Processors, 1998. ICCD '98. Proceedings. International Conference on, 1998, p. 230-235

[6] ARM7TDMI-S Datasheet, Advanced RISC Machines, October 1998.

[7] http://www.arm.com/Pro+Peripherals/Cores/ARM9E

[8] http://www.dspg.com/prodtech/core/teak.htm

[9] D.V Jaggar, "Advanced RISC Machines Architectural Reference Manual," Prentice Hall, London, 1996.

[10] SYNOPSYS, Design Compiler Reference Manual, Version 1998.02, February 1998.

[11] Advanced RISC Machines, ARM Software Development Toolkit, Version 2.50, 1998.

[12] W. R. Stevens, "Advanced Programming in the UNIX Environment", Addison Wesley, 1992.

[13] ARM7 Thumb Family, Advanced RISC Machines, ARMDOI 0035B