논문 2009-46SD-1-10

OpenRISC 프로세서와 WISHBONE 버스 기반 SoC 플랫폼 개발 및 검증

(Development and Verification of SoC Platform based on OpenRISC Processor and WISHBONE Bus)

빈 영 훈*, 류 광 기**

(Younghoon Bin and Kwangki Ryoo)

요 약

본 논문에서는 교육적 활용과 어플리케이션 개발에 응용 가능한 SoC 플랫폼을 제안한다. 플랫폼 하드웨어는 OpenRISC 프로세서, 범용 입출력장치, 범용 직렬 인터페이스, 디버그 인터페이스, VGA/LCD 제어기 등의 주변장치와 온 칩 SRAM 및 WISHBONE 인터커넥터로 구성되며 전체 합성 가능하도록 설계 되었다. 모든 하드웨어 구조는 재구성 가능하여 매우 유연한 구조로 되어있다. 또한 개발된 SoC 플랫폼의 하드웨어/소프트웨어 디버깅과 플랫폼 상에서 구현될 소프트웨어 개발을 위해 컴파일러, 어셈블러, 디버거, 운영체제 등의 SW 개발환경이 구현 및 검증되었다. 설계된 IP와 SoC는 Verilog HDL로 기술된 테스트벤치를 이용한 모듈 수준 기능검증, 최상위 블록 수준 기능검증, ISS를 이용한 구조적, 명령어 수준 검증, FPGA 프로토타입을 이용한 시스템 수준 에뮬레이션 방법을 통해 검증 되었다. 검증된 플랫폼을 이용한 멀티미디어 SoC를 Magnachip 0.18 um CMOS 라이브러리를 이용하여 ASIC으로 구현하여 91MHz의 클록 주파수에서 동작을 확인 하였다.

Abstract

This paper proposes a SoC platform which is eligible for education and application SoC design. The platform, fully synthesizable and reconfigurable, includes the OpenRISC embedded processor, some basic peripherals such as GPIO, UART, debug interface, VGA controller and WISHBONE interconnect. The platform uses a set of development environment such as compiler, assembler, debugger and RTOS that is built for HW/SW system debugging and software development. Designed SoC, IPs and Testbenches are described in the Verilog HDL and verified using commercial logic simulator, GNU SW development tool kits and the FPGA. Finally, a multimedia SoC derived from the SoC platform is implemented to ASIC using the Magnachip cell library based on 0.18 um 1-poly 6-metal technology.

Keywords: SoC 플랫폼, IP 개발 및 검증, 온 칩 버스, 임베디드 프로세서

I. 서 론

집적도가 증가하고 설계 및 공정 기술의 발달로 인하여 시스템 온 칩 (System-on-a-Chip)이 보편화 되고 있다. 이러한 반도체 기술의 발달로 인해 휴대폰, 디지털 카메라, DVD 플레이어, 디지털 TV등 다양한 전기, 전자 부품에 SoC 기술을 적용한 제품들이 상용화 되고 있지만 SoC를 이용한 전자 제품의 개발은 상당히 어렵고 많은 시간과 노력이 필요하다. 특히, 대부분의 소비 전자 제품들은 짧은 라이프사이클을 가지고 있기 때문에 조금이라도 빠른 시간에 하나의 제품을 출시하고 즉시 또 다른 제품을 만들어야 하는 Time-To-Market의 어려움에 처해있다. SoC 설계의 여러 문제점을 해결하기 위해 반도체 IP를 재사용 하는 방법이 있다. 다양한 IP는 SoC 설계자들이 이전의 제품 개발에서 사용한 설계물이나 혹은 팹리스 IP 전문 업체, 혹은 소스가 공개

* 학생회원, ** 평생회원, 한밭대학교 정보통신전문대학원
(Graduate School of Information and Communication, Hanbat National University)

※ 본 연구는 IDEC의 지원 및 지식경제부 출연금으로 ETRI, 시스템반도체산업진흥센터에서 수행한 IT SoC 핵심설계인력양성사업의 연구결과임.
접수일자: 2008년11월3일, 수정완료일: 2009년1월5일

된 IP 개발 그룹을 통해 제공 받을 수 있다. IP 기반 SoC 설계방법론은 IP 시장의 발전을 이루게 하였고 설계 전반에서 다양한 IP들이 사용되고 있다. 하지만 IP 기반 설계 방법은 기존의 IP를 재사용하기 때문에 설계하고자 하는 SoC에 집적화되기 위해 설계 과정의 각 수준에서 충분히 검증되고 분석되어야 한다. 이와 더불어 완벽한 설계물이라 하더라도 시스템에 집적하기 위해서 인터페이스의 호환성과 스펙의 적합성 등의 다양한 조건을 검사해야 하기 때문에 상대적으로 SoC 설계를 위해 복잡한 과정을 거친다.

이에 반해 프로세서, 메모리, 시스템 구성을 위해 필요한 주변 하드웨어 블록들을 하나의 플랫폼에 구현해 놓고 플랫폼을 활용한 어플리케이션 SoC를 개발하는 플랫폼 기반 SoC 설계 방법론이 발전되고 있다. 하나의 SoC 플랫폼은 검증된 하드웨어 IP 블록뿐만 아니라 플랫폼에서 운용될 소프트웨어의 개발환경도 구축되어있다. 이런 플랫폼 기반 설계방법의 장점은 하드웨어 IP를 쉽게 수정/추가/제거 할 수 있다는 것과 하드웨어 소프트웨어의 동시개발이 가능하다는 점이다. 따라서 플랫폼은 설계자가 설계에 필요한 노력과 시간을 최소로 줄이면서 차별화된 제품을 개발할 수 있는 다양한 기능을 제공한다. 본 논문에서는 OpenCores 그룹에서 공개한 IP를 활용하여 교육 및 어플리케이션 개발에 활용 가능한 SoC 플랫폼을 제안한다. 플랫폼은 32비트 RISC 프로세서, WISHBONE 버스, 온 칩 메모리, 주변장치, 그리고 소프트웨어를 위한 개발 환경으로 구성된다. 플랫폼 개발을 위해 프로세서를 비롯한 각 IP에 대한 분석과 검증, IP의 SoC 플랫폼 통합, 소프트웨어 개발환경 구축, 전체 시스템 통합, 최종적으로 개발된 플랫폼을 이용한 멀티미디어 SoC 칩 제작을 수행하였다. 개발 과정을 통해 충분히 검증된 플랫폼의 활용 가치와 재사용성을 확인하고 어플리케이션 시스템 개발을 위한 필요한 성능을 얻을 수 있었다.

II. 제안된 SoC 플랫폼의 기능 및 구성

1. OpenRISC 1200 임베디드 프로세서

OpenRISC 1200 프로세서는 쉽게 재사용 할 수 있고 수정 가능한 LGPL을 기반으로 라이센스 되어 있다. 따라서 다양한 측면에서 SoC 설계에 활용이 가능하다는 장점이 있다. 또한 합성 가능한 Verilog HDL 상위수준 언어로 기술되어 있기 때문에 특정 공정을 사용하여

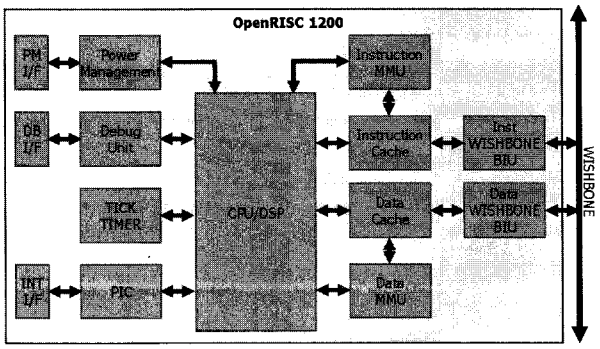

그림 1. OpenRISC 1200 프로세서 구조
Fig. 1. Architecture of OpenRISC 1200 Processor.

SoC 및 ASIC 설계에 활용할 수 있다. 또한 다양한 파라메타들이 하나의 Verilog file에 포함되어 있어 파라메타의 적절한 수정을 통해서 쉽게 컨피규레이션이 가능하다. 그림 1은 제안된 플랫폼의 프로세서 구조를 보여준다. OpenRISC 1200 프로세서는 명령어/데이터 패스 및 메모리 등이 분리된 하버드 구조로 설계 되었으며, 32비트 스칼라 RISC 프로세서다. RISC 프로세서의 특징에 맞게 5단 파이프라인 구조를 채택 하였으며, 임베디드시스템을 타깃으로 실시간 운영체제 지원을 위한 메모리 관리 장치를 지원하고 프로세서 내에 곱셈 및 누산기 유닛을 통한 기본적인 DSP 기능도 지원한다.

또한 프로세서 내부의 온 칩 메모리를 제외한 로직들의 게이트 수가 비교적 작고, 저 전력을 위한 파워 관리 블록과 외부 시스템과의 인터페이스를 통한 쉽고 빠른 디버깅 환경, 프로그램 가능한 인터럽트 인식 및 처리, WISHBONE 표준 인터페이스를 채택, 명령어 및 데이터 인터페이스를 구성하여 쉽게 IP들의 추가 및 수정이 가능하다는 점에서 저 전력, 저비용, 작은 크기의 임베디드, 포터블 시스템에 적합하다는 특징이 있다.

2. WISHBONE 온 칩 버스 아키텍처

OpenRISC 1200 프로세서는 외부 메모리 및 주변장치와의 연결을 위해 2개의 WISHBONE 호환 32비트 버스 인터페이스를 포함한다. WISHBONE 버스는 포터블 IP 코어를 위한 SoC 인터커넥터로 비교적 간단하고 쉽게 IP들을 연결할 수 있는 온 칩 버스 프로토콜을 지원한다. WISHBONE 인터페이스는 다중 마스터/슬레이브 아키텍처를 지원한다. 또한, 마스터와 슬레이브 사이의 연속적인 데이터 전송을 위한 버스트 데이터 전송과 같은 주소로의 읽기/쓰기 접근이 가능한 RMW 전송도 지원한다. 이밖에 핸드셰이크 방식의 통신 프로토콜을

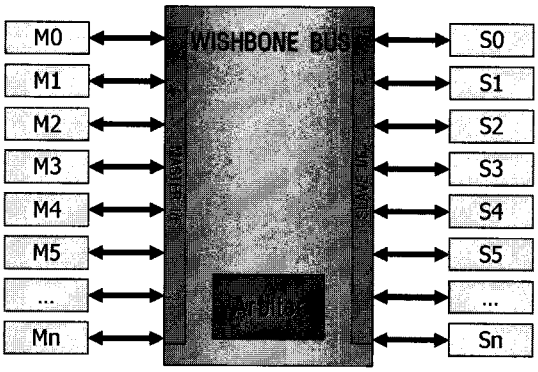

그림 2. 8x8 WISHBONE 인터커넥터의 블록도
Fig. 2. 8x8 Block Diagram of WISHBONE Interconnection.

사용하여 컴포넌트 상호간에 데이터 송/수신의 정보를 공유하는 특징을 가지고 있다. 그림 2는 8x8 멀티 마스터/슬레이브 구조의 WISHBONE 인터커넥터를 보여준다. 최대 8개의 마스터와 슬레이브를 연결할 수 있으며 버스 내부의 여러 마스터 중 하나의 마스터에게 버스 사용권을 허락하는 버스 중재기가 내장 되어 있다. 이 밖에도 선택된 마스터 신호를 슬레이브에게 전달하기 위해서 멀티플렉서와 디코더를 포함한다.

3. 제안된 SoC 플랫폼 구성과 특징

제안된 SoC 플랫폼은 32비트 RISC 프로세서 코어와 시스템 구성을 위해 필요한 기능 블록을 제공하기 위해 다양한 주변 장치들로 구성되며, IP들을 연결하고 공유 버스의 접근을 제어하며 높은 버스 전송 폭을 지원하기 위해 다중 버스 마스터/슬레이브 구조의 WISHBONE 온 칩 버스 구조가 선택되었다. 플랫폼을 구성하는 IP 블록은 UART, 디버그 인터페이스, 범용 입출력 장치, DMA 모듈을 포함한 VGA/LCD 제어기, 온칩 RAM, WISHBONE 인터커넥터이고, 플랫폼의 기본 통신 채널인 WISHBONE 온 칩 버스는 소스 클록에 동기화 되어 데이터, 주소 그리고 제어 신호를 전달하는 동기 버스로, 버스 내부의 버스 중재기가 매 순간 공유 버스의 사용권을 부여하기 위해 각 마스터와 슬레이브로 부터 신호를 전달받아 정해진 우선순위에 의해 버스 사용을 제어한다. 그림 3은 제안된 SoC 플랫폼의 블록도를 보여준다.

UART는 RS232 프로토콜을 사용하여 플랫폼과의 직렬 통신 기능을 제공한다. 플랫폼에 구현된 UART 모듈은 산업화 표준인 16550 직렬통신 프로토콜과 호환되며 최대 32비트 데이터 전송모드를 지원한다.

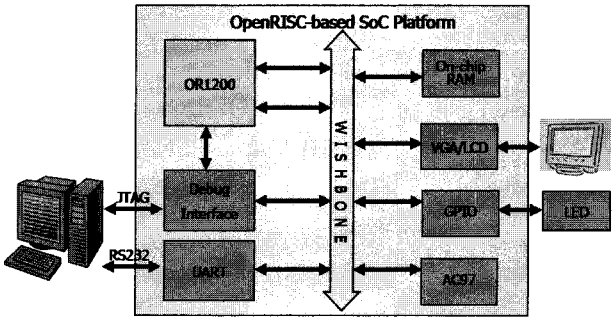

그림 3. 제안된 SoC 플랫폼의 전체 구성 블록도
Fig. 3. Block Diagram of Proposed Platform Architecture.

디버그 인터페이스 블록은 메모리 초기화, 프로세서 및 주변장치 컨피규레이션, 시스템 트레이스 및 디버깅 등 다양한 기능을 수행한다. 이 블록은 크게 탭 컨트롤러와 SoC 디버그 모듈로 나누어져 있는데, 외부 JTAG 포트로부터 전달된 TAP 제어 신호 및 데이터를 통해 SoC 내부 하드웨어 블록을 제어하고, 데이터 전송을 수행하며 SoC 디버그 모듈로 하여금 시스템 내부의 메모리, 레지스터 값이나 현재 수행중인 명령어를 시스템 디버깅을 위해 외부로 전달하는 기능을 하고 있다. 또한 플랫폼에는 쉽게 접근 및 제어가 가능한 온 칩 SRAM이 포함되어 있고, SRAM은 4K Byte로 구성되어 있으며 플랫폼 초기 구성 시 프로세서의 명령어나 연산에 필요한 데이터를 로드하여 시스템을 동작시키기 위해 사용된다. 또한 사용자 정의 가능한 범용 입/출력 장치 컨트롤러가 구성된다. 특정 목적을 위해 사용되는 고정 포트 이외에 플랫폼 내부와 외부 환경과의 데이터 전송을 위해 사용된다. 이밖에 LCD 디스플레이 장치를 통해 영상을 출력하기 위한 VGA 컨트롤러를 내장하고 있다. 32bpp, 24bpp, 16bpp, gray scale 등의 컬러 모드를 지원하며 임베디드 시스템에서 GUI를 필요로 하는 어플리케이션을 위해 사용된다.

4. 소프트웨어 개발 플랫폼

OpenRISC는 여러 가지 조합으로 구성된 개발 툴 킷을 제공한다. 그 툴 킷에는 바이너리 유틸, C/C++ 컴파일러, 어셈블러, 링커, 디버거 등 GNU 프로그램의 대부분이 포팅 가능하고 OpenRISC 그룹에서 개발된 ISS의 소스가 공개되어 사용자 환경에 맞추어 쉽게 컴파일 하여 실행할 수 있다. 또한 eCos, uClinux, Linux, RTEMS, microC/OS 등 많은 임베디드용 운영체제가 다양한 시스템 버전에서 OpenRISC 아키텍처를 위해

포팅 되었다. 소프트웨어 개발, ISS 기반 HW/SW 시뮬레이션, 디버깅 및 프로그램 다운로딩 등 통합 소프트웨어 개발 환경을 통해 설계자는 특정 목적에 맞게 임베디드 운영체제, 어플리케이션, 디바이스 드라이버 등을 개발하고 검증할 수 있다.

III. 시스템 검증 및 테스트

IP 코어 및 SoC 플랫폼의 재사용 가능함과 신뢰성을 보증하기 위한 가장 중요한 요소는 검증 절차다. OpenCore 그룹에서 OpenRISC 프로세서에 대해 여러 가지 테스트 환경에서 검증과정을 수행 하였고 칩 수준에서의 동작이 입증 되었지만 플랫폼을 구현하기 위해서는 더 많은 요소를 검증하여야 한다. 따라서 OpenRISC 프로세서와 주변 IP 모듈의 다양한 수준의 검증과 분석을 통해 신뢰성을 향상 시키고 누구나 쉽게 플랫폼을 사용하여 응용 할 수 있도록 하기위해 컴포넌트 수준, 아키텍처 수준, 시스템 수준 검증과 HW/SW 통합 시뮬레이션 등 다양한 검증과정이 SoC 플랫폼 설계에서 필요하다. 본 논문에서는 4가지 단계를 거쳐 구현된 SoC 플랫폼과 IP를 검증 하였다.

1. 블록 수준 검증

블록 수준 검증 방법에서는 전체 SoC 플랫폼을 기능 블록으로 분할하고 분할된 각 하드웨어 단위로 시뮬레이션을 수행한다. 시뮬레이션을 위해 Verilog HDL로 기술된 테스트벤치를 사용하며 테스트벤치 내에 다양한 테스트 벡터를 DUT에 인가하기 위한 태스크 및 함수가 포함된다. 블록 수준 검증은 하드웨어의 주요 기능을 테스트하고 각 모듈 사이의 인터페이스 및 신호관계를 검증하게 된다. 또한 블록 내의 특정 이벤트의 발생에 따라 내부 상황을 모니터링 할 수 있도록 Verilog 코드 안에 모니터링 코드를 삽입하여 디버깅 가능하도록 하였으며, 상용 로직 시뮬레이터를 통해 시뮬레이션 한 결과를 비교하여 기능을 검증하였다. 그림 4는 프로세서 블록을 검증하는 과정을 보여준다. OpenRISC 소프트웨어 개발환경을 이용하여 컴파일 된 바이너리 이미지를 16진 형식으로 바꾼 후 Verilog로 모델링된 FLASH, SRAM 시뮬레이션 모델에 명령어를 초기화하여 시뮬레이션을 실행 하였다.

시뮬레이션 결과로부터 프로세서내의 캐시 메모리가 사용 안 될 경우, 3클록 사이클마다 명령어를 플래시 메모리 모델에서 인출하여 각 명령어의 오퍼레이션을 수행하고 로드, 스토어 명령어 실행 시 데이터 메모리로 접근하여 데이터 읽기/쓰기를 수행함을 볼 수 있다. 시뮬레이션을 통해 OpenRISC 프로세서의 WISHBONE 인터페이스의 동작과 기능을 검증 하였다.

그림 5는 프로세서 내부의 캐시 메모리가 인에이블되어 CPU가 캐시 메모리로부터 명령어를 패치 하여 연산을 수행하는 과정을 시뮬레이션 한 결과를 보여준다.

캐시 메모리가 인에이블되면 CPU에서 필요한 명령어를 인출하기 위해 플래시 메모리로의 상대적으로 긴 접근 시간을 줄일 수 있다. 시뮬레이션 결과에서 보듯이 캐시를 사용 할 경우 플래시 메모리에서 명령어를 인출 할 때보다 캐시 데이터 히트가 발생 할 경우 1/3의 빠른 명령어 인출 속도를 얻는다. 따라서 매 클록 마다 명령어가 인출되어 파이프라인 레지스터를 통해 각 단계를 거쳐 매 클록마다 하나의 명령어를 실행하는 결과를 볼 수 있다. 그림 6은 SoC 플랫폼의 각 컴포넌트의 연결을 위해 설계한 8X8 WISHBONE 인터커넥터의

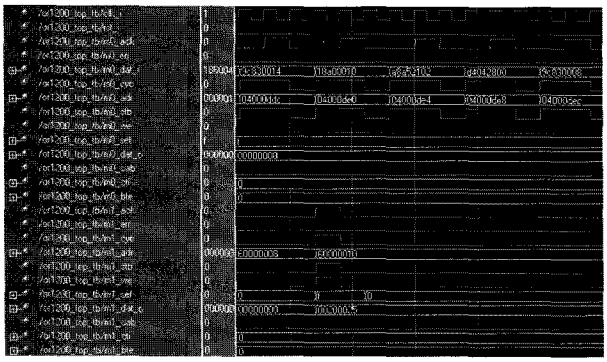

그림 4. OpenRISC 프로세서 블록 시뮬레이션 결과
Fig. 4. Logic Simulation Result of the OpenRISC Processor Block.

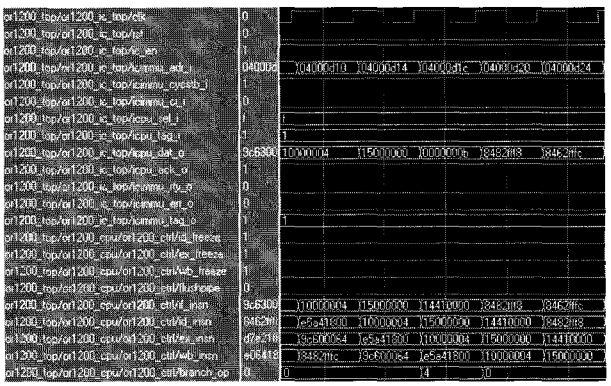

그림 5. 캐시가 사용된 프로세서 동작 시뮬레이션
Fig. 5. Logic Simulation of Processor including Caches.

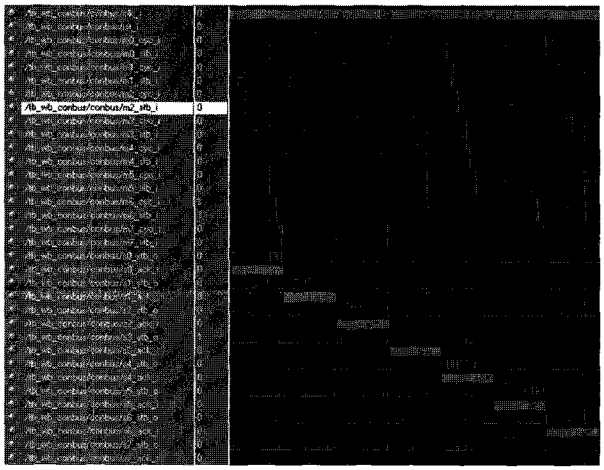

그림 6. 8X8 WISHBONE 버스 시뮬레이션
Fig. 6. Simulation for 8X8 WISHBONE Bus.

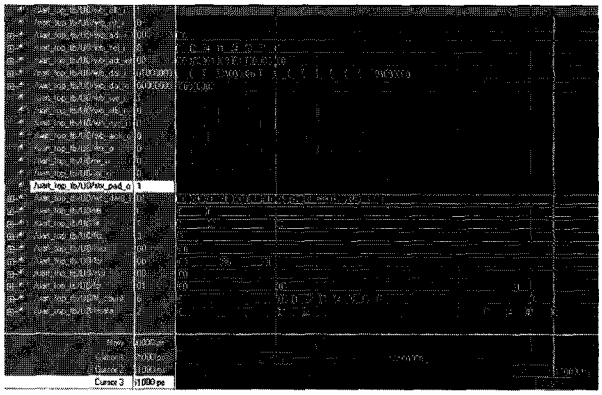

그림 7. UART 컨트롤러의 데이터 송신 시뮬레이션
Fig. 7. Simulation for Transmission Procedure of UART.

버스 중재와 제어 신호의 전달과정을 시뮬레이션 한 결과다. 시뮬레이션을 위해 최대 8개의 마스터와 슬레이브 블록을 연결 하였다. 버스 내부 중재기는 매 클록 마다 8개의 마스터로부터 버스 접근 요청 신호를 확인하고 라운드 로빈 방식으로 하나의 마스터에게만 32비트 주소, 데이터 버스와 각 제어 버스를 사용 할 수 있게 한다. 버스 사용이 허락된 마스터의 데이터 전송이 완료된 후 다음으로 우선권을 가진 버스 마스터에게 버스 사용권이 넘어감을 확인 할 수 있다. 시뮬레이션을 통해 특정 사이클 동안 하나의 마스터가 버스 중재기로부터 버스 사용 권한을 부여받고 해당 마스터 신호가 공통 버스에 올라오고 주소 디코더에 의해 선택된 슬레이브로 전달되어 마스터와 슬레이브 사이에 데이터 전송이 수행됨을 확인하였다.

그림 7은 UART 모듈의 외부 직렬 포트와의 데이터 송신 과정을 시뮬레이션 한 결과이다. 시뮬레이션을 위해 UART의 송수신 FIFO를 리셋하고, 수신 FIFO의 트리거 레벨을 14Byte로 설정하였다. UART에서 WISHBONE 마스터로 데이터 송수신에 대한 모든 인터럽트를 신호를 허용하고 송신 조건을 데이터 8비트, 정지비트 1비트, 홀수 패리티를 사용하도록 설정한 후 레지스터에 8비트 병렬 데이터를 입력하여 하위 비트부터 직렬로 STX_O 포트를 통해 외부 장치로 출력된다.

2. 통합 하드웨어 블록 검증

통합 하드웨어 블록 검증에서는 각 IP와 프로세서가 연결된 전체 SoC 하드웨어 블록을 상용 시뮬레이터로 시뮬레이션 하여 하드웨어 통합 검증을 수행한다. 이전 단계에서 검증된 OpenRISC 프로세서와 IP들이 통합된 플랫폼의 정상적인 동작을 검증하고, 실제 프로세서에서 동작하는 명령어를 기반으로 검증 하기위해 테스트 소프트웨어 프로그램을 OpenRISC GNU 툴 체인을 통해 컴파일 하여 실행코드를 추출하였다. 생성된 바이너리 이미지를 16진 코드로 변환하기 위해 C 프로그램을 작성하여 데이터 변환을 수행 하였으며 변환된 16진 코드는 테스트 벤치 안에서 $memreadh 함수를 사용하여 플랫폼의 SRAM 메모리 모델에 초기화된다. 그림 8은 통합 하드웨어 블록 검증의 테스트 환경을 보여준다.

시뮬레이션을 통해 프로세서는 온 칩 SRAM으로부터 프로그램 실행을 위한 명령어를 인출 하여 내부의 정의된 하드웨어에 의해 연산을 수행하고 플랫폼 안의 IP들의 레지스터나 메모리를 접근하여 데이터 읽기/쓰기 기능을 수행한다. 이러한 과정을 통해 프로세서와 WISHBONE 온 칩 버스, 주변 모듈의 동작과 컴포넌트 사이의 인터페이스를 검증 할 수 있다. 그림 9는 통합 하드웨어 블록 검증의 테스트 환경을 이용하여 설계된 SoC 플랫폼을 시뮬레이션 한 결과다.

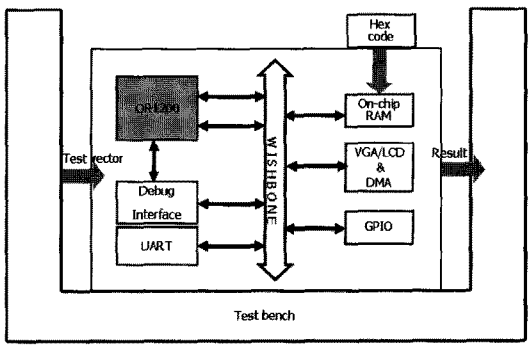

그림 8. 통합 하드웨어 블록 검증을 위한 테스트 환경
Fig. 8. Test Environment for Integrated HW Block Verification.

그림 9. SoC 플랫폼의 통합 하드웨어 시뮬레이션 결과
Fig. 9. Result of Simulation for SoC Platform Top Block.

프로세서에서 메모리에 저장된 프로그램을 실행하여 생성된 결과 데이터를 데이터 영역에 저장한 후 데이터 메모리로부터 UART 모듈을 통해 전달하여 외부 UART 포트를 통해 데이터 송신과정을 나타낸다. UART 모듈 내부의 FIFO 블록에 프로세서로부터 외부로 전달될 데이터가 저장되며, FIFO에 저장된 데이터는 UART 포트로 정해진 타이밍에 따라 송/수신 된다. 그림 9의 웨이브 파형에서 UART FIFO에 저장된 데이터의 아스키코드를 확인할 수 있다.

3. ISS를 이용한 아키텍처/명령어 수준 검증

그림 10은 ISS를 이용한 HW/SW의 아키텍처, 명령어 수준 검증 방법을 보여준다.

ISS 기반 검증은 하드웨어 구현이 완성되지 않았거나 실제 물리적인 디바이스가 없을 경우 유용하게 사용된다. 설계자는 프로세서의 구조적, 명령어 수준의 기능과 동작을 확인 할 수 있으며 다양한 주변장치 IP 모델을 연결하여 프로세서뿐만 아니라 주변 IP가 포함된 SoC를 테스트 할 수 있다. 또한 하드웨어뿐만 아니라,

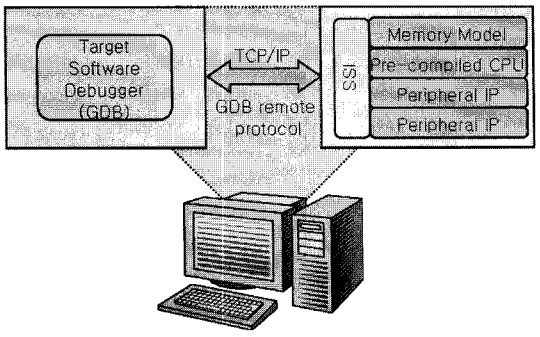

그림 10. 아키텍처/명령어 수준 검증 환경
Fig. 10. Test Environment of Architecture/Instruction Level Verification.

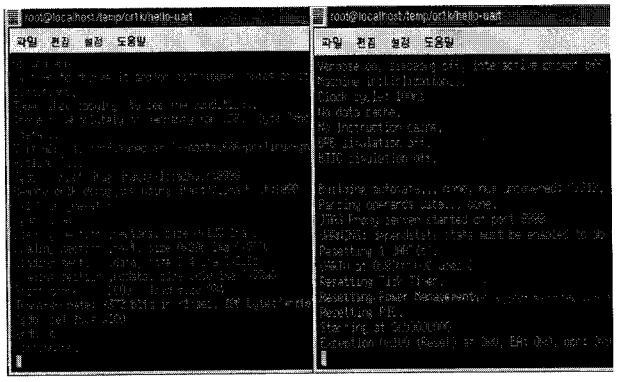

그림 11. GDB-ISS를 이용한 구조적/명령어 수준 시뮬레이션
Fig. 11. Architectural/Instructional Level Simulation Using GDB-ISS Connection.

응용 프로그램의 테스트도 가능하다. OpenCores 그룹에서 개발된 ISS를 Linux 기반 개발환경에 포팅한 후 필요로 하는 하드웨어 블록을 구성하고 ISS 환경을 구성하여 GDB 디버거를 통해 실행 및 디버깅을 수행한다. 또한 ISS와 GNU GDB를 이용하여 SoC의 가상 HW에서 실제 SW를 미리 검증할 수 있다.

이러한 테스트 환경은 플랫폼의 하드웨어 블록을 각 IP의 C 언어 수준의 추상화 수준으로 기술한 시뮬레이션 모델과 플랫폼을 테스트 하기위한 소프트웨어 프로그램을 GDB- ISS 시뮬레이션 환경에 통합하여 구현할 수 있다. 그림 11은 ISS와 주변 장치의 상위수준 모델, GDB 디버거를 이용하여 소프트웨어를 로딩하고 시뮬레이션 한 결과다.

4. FPGA 프로토타입을 이용한 시스템 수준 검증

SoC 플랫폼의 시스템 수준 검증을 위해서 플랫폼을 구성하는 하드웨어 IP를 SoC 플랫폼에 통합하여 FPGA에 구현한다. 플랫폼의 칩 수준 동작을 테스트 하기위해 테스트 프로그램과 어플리케이션 프로그램이 사용된다. 검증된 주변장치 IP들을 OpenRISC 프로세서가 포함된 SoC에 통합하여 FPGA 개발 보드를 이용해 최종 테스트를 하였다. Xilinx ISE 툴을 이용하여 SoC의 로직 합성을 수행하며 FPGA 컨피규레이션을 위해 실제 JTAG 연결핀을 이용해 FPGA에 프로그램 하였다. 플랫폼 내부의 VGA IP를 이용하여 640X480의 테스트 이미지를 출력하고 프로세서 및 하드웨어 블록의 연결 및 기능을 테스트하기 위한 어셈블리 테스트 프로그램과 SoC 하드웨어/소프트웨어 검증을 위한 어플리케이션 프로그램이 OpenRISC GNU 툴 체인을 통해 크로스

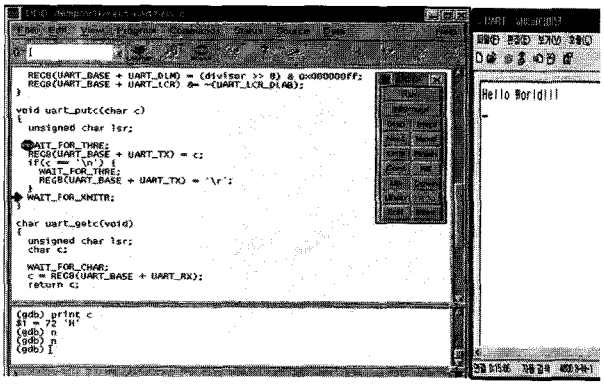

그림 12. SoC 플랫폼을 테스트하기 위한 시리얼 통신
Fig. 12. Serial Communication for Test of SoC Platform.

컴파일 되어 바이너리 이미지로 생성된다. 이 실행 코드를 호스트 환경의 GDB/DDD 디버거를 통해 구현된 SoC의 온 칩 RAM에 로딩하고 전체 SoC의 동작을 확인한다. 그림 12는 FPGA에 구현된 SoC 플랫폼의 UART 기능을 테스트하기 위한 시리얼 프로그램이고, 두 개의 uart_putc와 uart_getc 함수를 통해 프로세서로부터 UART 제어기를 통해 외부 환경과 직렬 데이터 송/수신을 처리하는 과정을 테스트한다. 실행 결과는 하이퍼터미널을 통해 확인한다.

IV. 시스템 구현

1. FPGA를 이용한 플랫폼 환경

FPGA를 이용해 SoC 플랫폼을 구현하기위해 Xilinx Virtex-4 Family XC4VLX80 디바이스를 사용하였다. FPGA를 통해 새로운 IP를 설계하여 통합 할 수 있고 빠르게 검증 할 수 있는 장점을 가지고 있으며 플랫폼을 이용한 어플리케이션 소프트웨어를 개발하기 위한

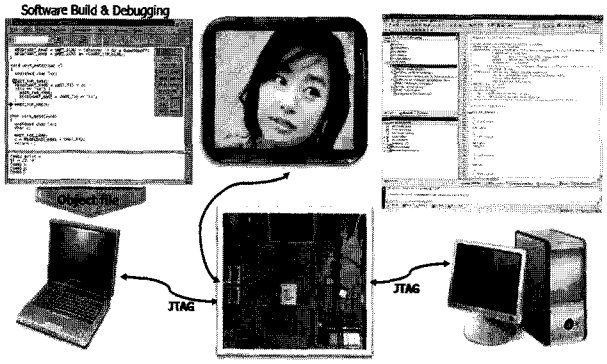

그림 13. SoC 플랫폼 개발 환경
Fig. 13. Development environment of proposed SoC platform.

표 1. FPGA를 이용한 SoC 플랫폼 구현 결과
Table 1. Summary of SoC platform implementation in FPGA.

디바이스	XC4VLX80
Slice	3347 (43%)
LUTs	6034 (39%)
최소 요구 주기	31.524 ns
최소 셋 업 시간	7.757 ns
최소 홀드 시간	10.605 ns
최대 동작 클록주파수	31.722 MHz

환경을 함께 구현하여 하드웨어/소프트웨어 동시 개발이 가능하다. 플랫폼 FPGA 구현 결과 약 40 %의 로직을 사용하였고, 최대 동작 클록 주파수는 약 32 MHz의 결과를 보였다. 그림 13은 SoC 플랫폼 개발 환경을 보여주며 표 1은 Xilinx XC4VLX80 FPGA를 이용한 플랫폼 구현 결과를 나타낸다.

2. Magnachip 0.18 um standard cell 라이브러리를 이용한 멀티미디어 SoC 칩 설계

개발된 SoC 플랫폼 개발 환경을 바탕으로 멀티미디어 SoC를 Magnachip 0.18 um cell 라이브러리를 이용하여 ASIC으로 설계하였다. 멀티미디어 기능을 포함한 RTL 형태의 SoC를 Magnachip 0.18 um cell 라이브러리를 사용하여 Synopsys 사의 Design Compiler를 통해 로직 합성하였다. 합성 시 worst/best 두 가지 동작 환경을 고려하여 동작조건과 와이어로드 모델을 적용 하였으며 두 가지 경우 모두 주어진 제약조건을 만족하였다. 로직 합성 후 생성된 Verilog netlist 파일과 추출된 지연 정보인 SDF 파일을 가지고 SoC 검증 초기 단계의 기능 검증 시 사용했던 테스트 벤치를 이용하여 프리-레이아웃 시뮬레이션 검증을 수행하여 타이밍과 기능을 검증하였다.

시뮬레이션 검증 후 netlist 파일은 Synopsys Astro P&R 툴을 사용하여 배치 및 배선되며, P&R에 사용된 매크로 블록은 대부분 온 칩 메모리만을 사용하였으며 플루플랜 (Floorplan) 과정에서 메모리의 위치를 특정 메탈 레이어와 측면에 위치시켜 P&R을 수행하였다. P&R에 사용된 타이밍 스펙은 로직 합성 시 사용 하였던 것의 약 7 %의 마진을 제거하여 적용 하였으며 최종 배선 후 설계 타이밍과 설계 규칙이 주어진 스펙에 만족하는 결과를 얻었다. 최종 P&R 된 셀로부터

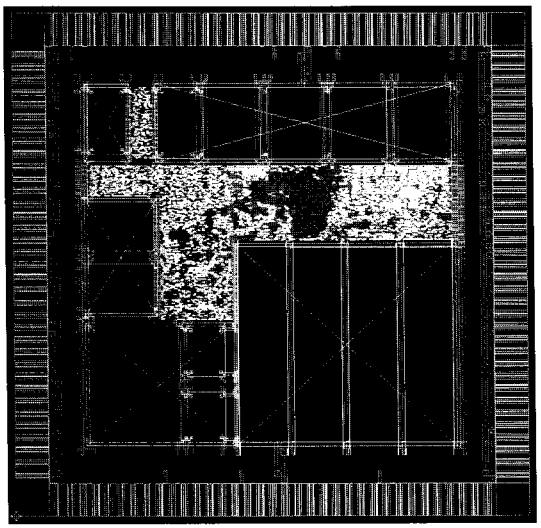

그림 14. SoC 플랫폼의 배치 및 배선 결과
Fig. 14. P&R Result of SoC Platform.

표 2. ASIC 라이브러리를 이용한 플랫폼 설계 결과
Table 2. Summary of SoC platform implementation in ASIC.

공정	Magnachip 0.18 um CMOS
셀 라이브러리	M18GM180S
합성 클록 주파수	96 MHz
P&R 클록 주파수	91 MHz
면적	2.6 mm^2 core-region out of 4 mm^2
동작 전압	1.8 V Core, 3.3 V I/O

SPF(Standard Parasitic Format)을 추출하고 새로운 SDF 파일과 포스트 netlsit를 가지고 시뮬레이션 하였다. 프리 레이아웃 시뮬레이션과 동일한 수준의 테스트 입력을 통해 시뮬레이션 한 결과 동일한 출력을 확인 하였다. 그림 14는 Astro 툴을 이용한 P&R 결과이고, 표 2는 ASIC 설계 사양에 대한 요약이다.

V. 결 론

본 논문에서는 OpenRISC 프로세서와 WISHBONE 버스 기반 합성 가능한 SoC 플랫폼이 제안되었다. SoC 플랫폼은 다양한 주변장치를 포함하여 기본적인 프로세서를 이용한 소프트웨어 프로그램 실행은 물론 주변 장치를 이용한 어플리케이션 응용도 가능하다. 구현된 플랫폼은 블록 수준 기능검증, 하드웨어 통합 검증, 구조적/명령어 수준 기능 검증, FPGA 프로토타입을 통한 시스템 HW/SW 통합검증, 총 4단계의 검증과정을 거쳐 동작 및 타이밍을 테스트 하였으며, Xilinx XC4VLX80 FPGA를 이용한 프로토타입을 거쳐 최종적으로 Magnachip 0.18 CMOS cell 라이브러리를 이용하여 ASIC 칩으로 설계되었다. FPGA 프로토타입 결과 최대 동작 클록 주파수는 약 32 MHz이며, ASIC 칩 제작을 통해 최대 91 MHz의 동작 클록 주파수에서 정상적으로 동작함을 확인 하였다.

참 고 문 헌

[1] Damjan Lampret, OpenRISC1200 IP Core Specification Rev. 0.7, September 6, 2001.
[2] Richard Herveille, WISHBONE SoC Architecture Specification, Revision B.3, September 7, 2002.
[3] Jacob Gorban, UART IP Core Specification, Rev. 0.6 August 11, 2002.
[4] Igor Mohor, SoC Debug Interface, Rev. 3.0 April 14, 2004.
[5] Richard Herveille, VGA/LCD Core Specification, Rev. 2.0 March 20, 2003.
[6] Damjan Lampret, OpenRISC1000 Architecture Manual, January 28, 2003.
[7] David A. Patterson, Computer organization and design : the hardware software interface. 3rd edition, Morgan Kaufmann Pub, 2004.
[8] M. Bolado, "Platform based on Open-Source Cores for Industrial Applications", IEEE Computer Society, 2004.
[9] OpenCores, http://www.opencores.org
[10] Daniel Mattsson, Evaluation of synthesizable CPU cores, December 21, 2004.
[11] Xilinx, XC4VLX80 Data Sheet
[12] Rudolf Usselmann, Verification Strategies, Rev. 0.1, February 4, 2001.
[13] OpenCores Coding Guidelines, Revision. 1.2, July 14, 2003.
[14] Richard Stallman, Debugging with GDB, Rev. 9, June 2002.
[15] Magnachip Semiconductor, LTD. 0.18-Micron 1.8V Standard Cell Library Datasheet, June, 2005
[16] Synopsys, Astro User Guide, version Y-2006.06, June 2006.
[17] Synopsys, Design Compiler User Guide, version 2002.05, June, 2002.

저 자 소 개

빈 영 훈(학생회원)

2006년 한밭대학교 컴퓨터공학과 학사

2008년 한밭대학교 정보통신전문대학원 석사

2008년~현재 한밭대학교 정보통신전문대학원 박사과정

<주관심분야 : 임베디드 프로세서, SoC 플랫폼 설계, 하드웨어/소프트웨어 통합설계, 멀티미디어 코덱 설계>

류 광 기(평생회원)

1986년 한양대학교 전자공학과 학사

1988년 한양대학교 전자공학과 석사

2000년 한양대학교 전자공학과 박사

1991년~1994년 육군사관학교 교수부 전자공학과 전임강사

2000년~2002년 한국전자통신연구원 집적회로연구부 시스템 IC 설계팀 선임연구원

2003년~현재 한밭대학교 정보통신공학과 전임강사, 조교수

<주관심분야 : SoC 플랫폼 설계 및 검증, 하드웨어/소프트웨어 통합설계 및 통합검증, 멀티미디어 코덱 설계>