The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 22, No. 5, pp.105-110, Oct. 31, 2022. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2022.22.5.105
JIIBC 2022-5-16

파이프라인 방식의 32 비트 ARM 프로세서에 대한 FPGA 구현 및 검증

FPGA Implementation and Verification of A Pipelined 32-bit ARM Processor

이종복*

Jongbok Lee*

요 약 국내의 메모리 반도체 설계 기술은 세계 최고의 수준이나, 아직까지 프로세서의 설계는 그에 미치지 못하여 메모리와 프로세서의 균형있는 발전을 이루지 못하고 있다. Xilinx에서 제공하는 Vivado 통합 환경을 이용하여 저렴한 비용으로 짧은 시간에 현장에서 즉석으로 쉽게 프로세서를 FPGA 반도체 칩에 구현할 수 있다. 본 논문에서는 유럽 및 전 세계의 대학 및 연구소에서 디지털시스템 설계에 널리 쓰이는 VHDL을 이용하여 32 비트 ARMv4 계열의 프로세서를 설계하고, Vivado에서 Xilinx FPGA로 구현 및 로직아날라이저로 검증하였다. 그 결과, FGPA로 구현된 ARM 프로세서가 ARM 명령어들로 구성된 프로그램을 성공적으로 수행하였다.

Abstract Domestically, we are capable of designing high-end memory semiconductors, but not in processors, resulting in unbalance. Using Vivado as a development enivronment and implementing the processor on a Xilinx FPGA reduces time and cost dramatically. In this paper, the popular language VHDL which is widely used in Europe, universities, and research centers around the world for the digital system design is used for designing a pipelined 32-bit ARM processor, implemented on FPGA and verified by Integrated Logic Analyzer. As a result, the ARM processor implemented on FPGA could execute ARM instructions successfully.

Key Words : ARM, FPGA, VHD

Ⅰ. 서 론

1980년대에 Acorn Computer Group에서 최초로 개발된 ARM 아키텍처는 Advanced RISC Machine으로 명칭을 바꿔서 오늘의 ARM이 되었다. ARM은 2016년에 SoftBank에 인수되었다가, 2020년에는 그래픽 프로세서 장치를 주력으로 하는 NVIDIA에 의한 흡수가 무산되었다. 2021년 기준 290억 개의 ARM 프로세서가 매년 출하되고 있으며, 스마트폰, 태블릿, 게임기, 카메라, 로봇, 자동차, 컴퓨터 서버 등 광범위하게 쓰이고 있다. ARM은 직접 완성된 프로세서 칩을 판매하지 않으며, 라이선스를 통하여 삼성, 애플, 퀄컴 등의 회사들에 시스템 온 칩의 형태로 공급한다. 이와같이 ARM 프로세서는 임베디드 기기에 이용되는 저전력 프로세서로서 Intel과 대비되며, 모바일 시장에서 큰 강세를 보이고 있다. 오늘날, Intel이 거머쥐고 있는 데스크탑 컴퓨터나

*정회원, 한성대학교 기계전자공학부
접수일자 : 2022년 9월 21일, 수정완료 : 2022년 10월 5일
게재확정일자 : 2022년 10월 7일

Received: 21 September, 2022 / Revised: 5 October, 2022 / Accepted: 7 October, 2022
*Corresponding Author: jblee@hansung.ac.kr
School of ME Engineering, Hansung University, Korea

서버 컴퓨터를 제외한 대부분의 스마트폰 및 임베디드 기기에는 전부 ARM 프로세서가 탑재되어 있다고 해도 과언이 아니다.

국내에서는 메모리 반도체 설계 및 공정 기술이 세계 최고의 수준이지만, 시스템반도체 중에서 특히 프로세서 설계는 아직 그렇지 못한 형편이다. 따라서 프로세서 설계 분야에 대한 연구를 활발히 하고 박차를 가할 필요가 있다. 본 논문에서는 VHDL을 이용하여 5 단계 파이프라인을 갖는 32 비트 ARM 프로세서를 설계하였으며, FPGA에 구현하고 검증하였다.

본 논문은 다음과 같이 구성된다. 2 장에서 ARM 프로세서 구조의 특징과 아울러 ARM 프로세서의 명령어 및 파이프라인 단계별 동작을 살펴 고찰한다. 3 장에서 ARM 프로세서의 FPGA 합성을 자세히 살펴보며, 4 장에서 FPGA 구현 환경과 로직아날라이저를 이용한 검증 결과를 보이고, 5 장에서 결론을 맺는다.

II. ARM 프로세서의 구조

표 1에 본 논문에서 설계한 ARM 명령어 집합을 나타냈다. ARM 명령어는 크게 산술 논리, 곱셈, 메모리, 분기의 4 가지 유형으로 구성된다[1].

표 1. 설계된 ARM 프로세서의 명령어 집합
Table 1. The instruction set of the Designed ARM processor

명령어 유형	명령어
산술 논리	AND EOR SUB RSB ADD ADC SBC RSC TST TEQ CMP CMN ORR MOV LSL LSR ASR RRX ROR BIC MVN
곱셈	MUL MLA UMULL UMLAL SMULL SMLAL
메모리	STR LDR STRB LDRB STRH LDRH LDRSB LDRSH
분기	B BL

산술 논리 명령어는 AND, EOR, SUB 등의 21 개 명령어로 구성된다. 곱셈 명령어는 6 개의 명령어로 이루어지며, 결과가 32 비트인 것과 64 비트인 것으로 나뉜다.

특히 MLA, UMLAL, SMLAL은 곱셈의 결과에 다시 덧셈을 수행하므로, 일반 명령어가 두 개의 피연산자 레지스터를 읽는 것과 달리, 세 개의 피연산자 레지스터를 동시에 읽어야 한다. 메모리 명령어는 STR, LDR 등의 스토어 및 로드 명령어로 구성되며 워드 단위 이외에 바이트 단위 또는 하프워드 단위로 수행한다. 마지막으로, 분기 명령어는 B와 BL로 구성된다.

그림 1은 ARM 프로세서의 5 단계 파이프라인이다. 이것은 인출(Fetch), 해독(Decode), 메모리 접근(Memory), 실행(Execute), 되쓰기(Write Back)의 5 단계로 나뉜다.

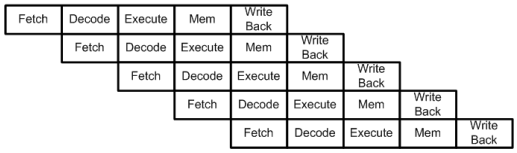

그림 1. ARM 프로세서의 5 단계 파이프라인
Fig. 1. The 5 stage pipeline of ARM processor

명령어 파이프라인에서 명령어 간에 쓰기 후 읽기(RAW) 종속이 발생하는 경우를 해저드라고 하며, 해저드 유닛에서 처리한다. 인접한 명령어 간에 발생하는 해저드는 메모리단계의 결과를 포워딩시키고, 한 개의 명령어만큼 떨어진 명령어 간에 발생하는 해저드는 되쓰기 단계의 결과를 포워딩시켜서 해결할 수 있다. 마지막으로, 두 개의 명령어만큼 떨어진 명령어 간에 발생하는 해저드는 레지스터 화일의 내부 포워딩에 의하여 해소된다[2,3].

그림 2에 본 논문에서 설계한 ARM 프로세서의 전체 블럭도를 보였다. 인출단계, 해독단계, 실행단계, 메모리 접근단계, 되쓰기단계의 각 사이마다 파이프라인 레지스터가 삽입되어 명령어 파이프라인 체계를 지원한다. 아래 각 파이프라인 단계를 상세히 기술한다.

1. 인출단계

명령어 메모리에는 ARM 프로세서가 실행할 명령어들이 기억되어있으며, 프로그램 카운터 (PC) 레지스터에 수록된 명령어 주소를 공급받아 명령어를 출력하는 기능을 수행한다. 명령어들을 순차적으로 인출할 때는 주소를 바이트 단위로 접근하므로 PC+4의 주소를 이용하나, BL 분기 명령어의 경우에는 단순증가된 PC 값과 ALU로 계산한 값 중에서 하나를 멀티플렉서로 선택하여 보낸다.

인출부에서 2 개의 덧셈기가 필요한데, 순차적인 다음 주소를 발생시키기 위하여 PC+4를 실행하는 덧셈기와, 특수 분기 명령어의 경우 복귀하는 명령어의 주소로 PC+8을 계산하는 덧셈기가 필요하기 때문이다.

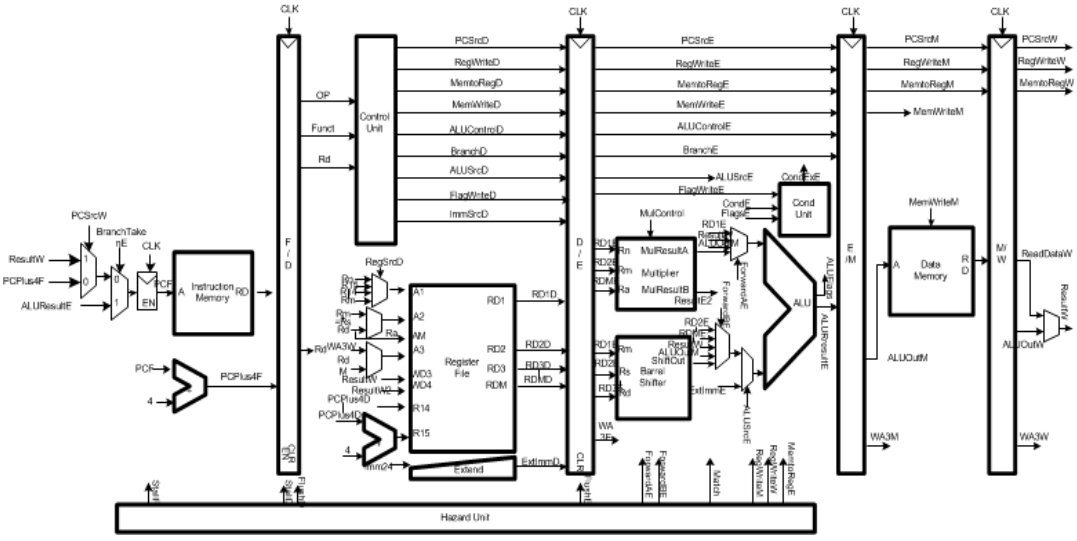

그림 2. ARM 프로세서의 블럭도
Fig. 2. The block diagram of ARM processor

2. 해독단계

레지스터 화일은 곱셈 명령어를 원활히 처리하기 위하여 동시에 3 개의 피연산자를 읽고, 2 개의 결과값을 쓸 수 있도록 설계되었다. 해독단계의 제어부는 명령어 메모리에서 인출한 명령어의 필드에 따라 9 개의 제어신호를 만들어낸다. 이것은 PC 값을 선택하는 제어신호, 레지스터 화일에 결과를 쓸 것인지 여부를 결정하는 제어신호, 메모리로부터 읽은 값을 레지스터에 쓸 것인지의 여부를 결정하는 제어신호, 레지스터의 데이터를 메모리에 쓰도록 하는 제어신호, ALU에서 어떤 유형의 연산을 할 것인지를 결정하는 제어신호, 분기타겟 주소를 PC에 적재하는 제어신호, ALU의 두 번째 입력을 선택하는 제어신호, 상태 플래그를 업데이트하는 제어신호, 명령어 영역의 직접 피연산자를 몇 비트로 확장할 것인지를 지정하는 제어신호로 구성된다.

3. 실행단계

실행단계는 ALU의 상단에 곱셈기와 쉬프터를 설치해서 산술논리연산 외에 곱셈과 쉬프팅 동작을 지원한다. ALU의 첫번째 입력은 레지스터화일에서 공급되는 값, 메모리단계에서 포워딩되는 값, 되쓰기 단계에서 포워딩되는 값, 마지막으로 곱셈기에서 출력되는 값 중에서 선택한다.

ALU의 두번째 입력은 레지스터화일에서 공급되는 값, 데이터 해저드가 발생했을 때 메모리단계와 되쓰기 단계에서 포워딩되는 값, 부호확장기로부터 공급되는 값, 그리고 쉬프터의 출력 중에서 선택한다.

분기가 채택되었을 때, 후속의 두 개 명령어가 파이프라인에서 삭제된다. 그리고 만일에 PC에 대한 쓰기 작업이 파이프라인 내에 존재하면, 이것이 완료될 때까지 파이프라인은 정지해야 한다. 이것은 인출단계를 정지함으로써 구현되며, 특정 단계를 정지시키면, 다음 단계를 삭제해야 명령어가 반복 실행되는 것을 방지할 수 있다.

4. 메모리단계

메모리단계는 데이터메모리와 실행단계에서 파이프라인 래치를 통하여 전달된 각종 제어신호들로 구성된다. 명령어 메모리의 주소 입력은 실행단계의 ALU 출력과 연결되며, 데이터 입력은 레지스터화일의 두 번째 출력으로부터 공급된 데이터와 연결된다. 명령어 메모리의 출력은 되쓰기 단계의 파이프라인 레지스터로 전달된다. 한편, 스토어 명령일 때 메모리 쓰기 제어신호를 이용하여 데이터메모리에 값을 기록할 수 있다.

5. 되쓰기단계

두 번째 명령어 파이프라인인 해독단계에서 레지스터 화일에서 데이터를 읽어들이는데, 되쓰기단계는 이것과는 반대로 레지스터 화일에 데이터를 기록하는 과정이

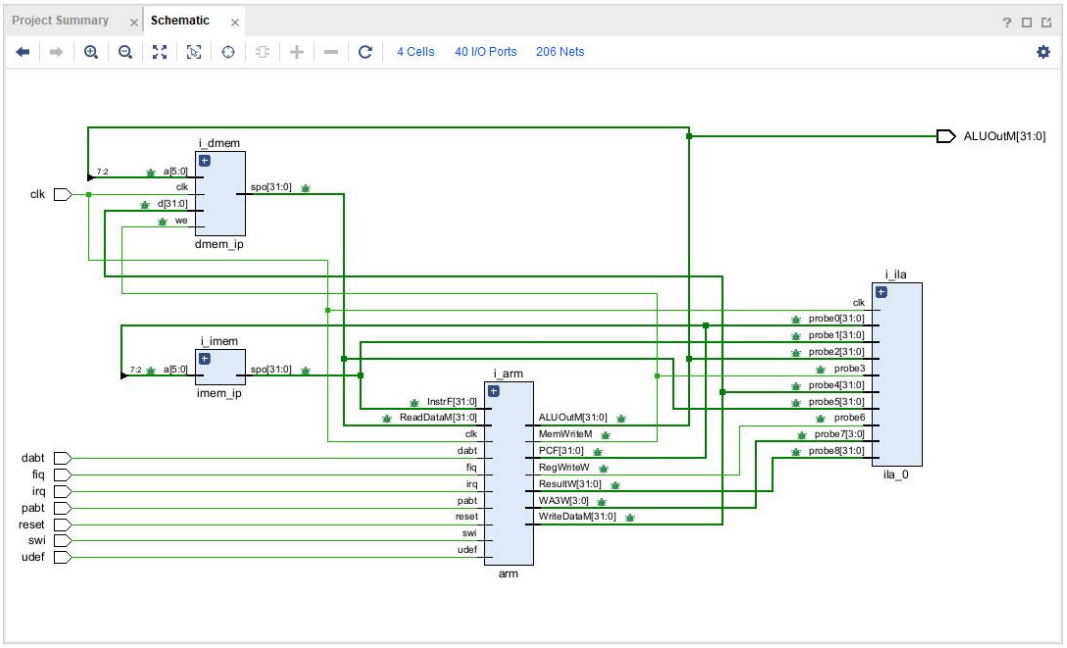

그림 3. 프로세서, 명령어메모리, 데이터메모리, ILA-IP의 최상위 수준 설계도
Fig. 3. Top level design of processor, instruction memory, data memory, and ILA-IP

다. 즉, 레지스터 화일은 유일하게 읽기와 쓰기를 위하여 명령어 파이프라인에서 두 번 접근되는 장치이다. 로드인 경우에 명령어 메모리로부터 읽은 값을 레지스터 화일에 써야 하며, ALU 연산인 경우에도 그 결과를 레지스터 화일에 역시 써야 한다. 따라서, 메모리를 레지스터에 연결하는 제어신호로 둘 중에 하나를 선택한다.

본 프로세서는 마이크로프로세서에 필수적인 7 가지 인터럽트 기능을 탑재하였다. 인터럽트에는 리셋에 의한 인터럽트, 데이터중단 인터럽트, 고속 인터럽트, 외부장치에 의한 일반 인터럽트, 선인출중단 인터럽트, 소프트웨어 인터럽트, 정의되지 않은 명령어에 의한 인터럽트로 구성된다.

III. ARM 프로세서의 FPGA 합성

본 논문의 전 구현 과정은 3.1GHz로 동작하는 Intel Core i7-950 데스크탑 컴퓨터에서 CentOS 및 Windows 10 운영체제하에서 시행하였다. Xilinx사의 Vivado 2022.1 버전을, VHDL 컴파일러는 2008 버전을 이용하였다. Vivado에서 VHDL로 프로그래밍한 파이프라인 방식의 32 비트 ARM 프로세서에 대한 시뮬레이션이 끝난 후에 RTL 분석, 합성, 구현, FPGA 프로그램의 순서로 연구를 진행하였다[4,5].

그림 3에 합성된 최상위 레벨의 회로도를, 그림 4에 합성된 ARM 프로세서의 회로도를 나타냈다. 최상위 레벨 회로도에서 명령어 메모리와 데이터 메모리는 Vivado의 IP-Catalog에서 제공하는 Distributed Memory Generator를 통해 각각 ROM과 Single Port RAM으로 구성했다. 명령어 메모리는 검증을 위한 ARM 프로그램을 32 비트 16 진수 기계어로 작성하여 COE 파일로 적재하여 초기화했다. 그리고 로직아날라이저를 통한 검증을 위하여 ILA 코어를 포함시켰다.

RTL 분석 결과, 본 프로세서는 4 개의 Cell, 40 개의 I/O Port, 206 개의 Nets로 구성됐다. 그림 5에 FPGA로 구현된 프로세서를 나타내고 있다. 합성 결과, LUT 939 개, Register 1050 개, Mux 160 개, DSP 3 개, IOB 40 개가 소요되었다. 전력 분석 결과, 클럭에 45 %, 신호에 30 %, 논리게이트에 20%의 전력이 소모되어 총 0.099 W가 소모되었다. 타이밍 분석 결과, 최장 셋업 시간이 26.5ns, 홀드 타임이 0.033ns, 펄스폭이 15.3 ns로 측정되었다.

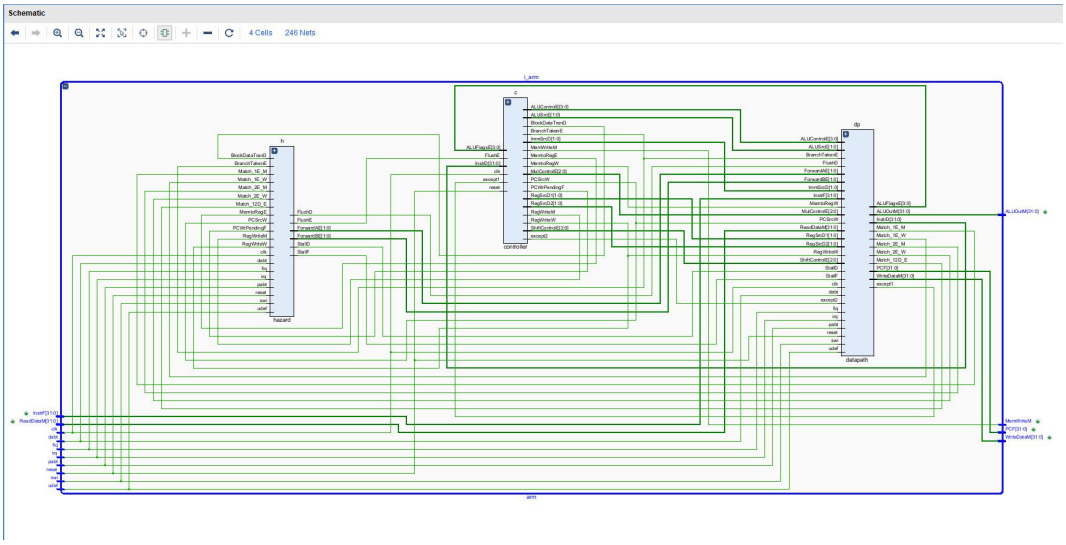

그림 4. 데이터부, 제어부, 해저드블럭으로 합성된 ARM 프로세서
Fig. 4. Synthesized ARM processor by Data Path, Control Path, and Hazard Block

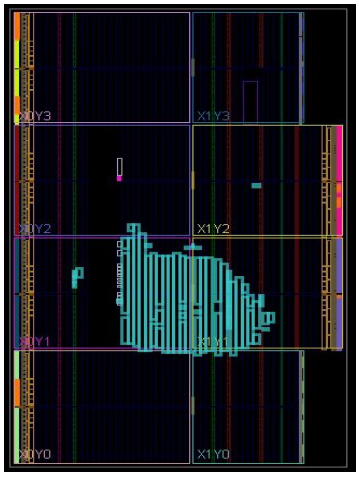

그림 5. FPGA로 구현된 ARM 프로세서
Fig. 5. The FPGA Implemented ARM processor

IV. FPGA에 대한 ILA를 이용한 검증

본 연구의 FPGA 구현 및 실행에 이용된 보드인 국내 리버트론사의 FPGA Starter Kit III를 그림 6에 나타냈다. 여기에 탑재된 FPGA는 xc7a75tfgg484-1이다.

그림 7에 ARM 프로세서가 실행되는 ILA (Integrated Logic Analyzer) 결과 파형을 나타냈다. 일반적으로 FPGA 검증을 위하여 IP-Catalog에서 ILA 코어를 설계에 포함시키는 방법과, 합성된 후에 관찰하고 싶은 노드를 찾아서 찍어보는 방법이 있다. 그러나, 합성 후에는 신호들의 이름이 바뀌거나 버스 신호가 분리되기 때문에 두 번째 방법이 적절치가 않다. 따라서, 본 논문에서는 ILA 코어를 설계에 포함시키는 첫 번째 방법을 채택하였다. 에뮬레이션에 이용된 클럭은 FGPA Starter Kit III에서 제공되는 100MHz 클럭을 이용하였다.

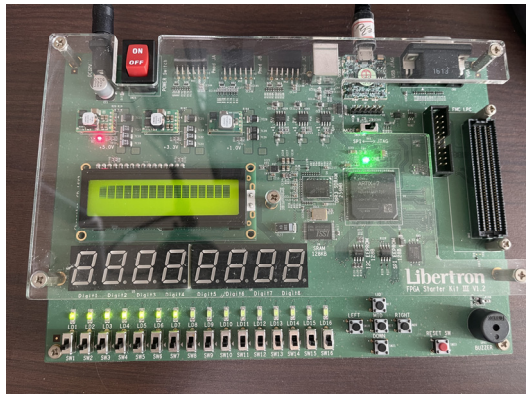

그림 6. 리버트론사의 FPGA Starter Kit III
Fig. 6. FPGA Starter Kit III of Libertron Ltd.

검증에 필요한 노드의 신호들로 프로그램카운터, 32 비트 명령어, 메모리에서 읽은 32 비트 데이터, 메모리에

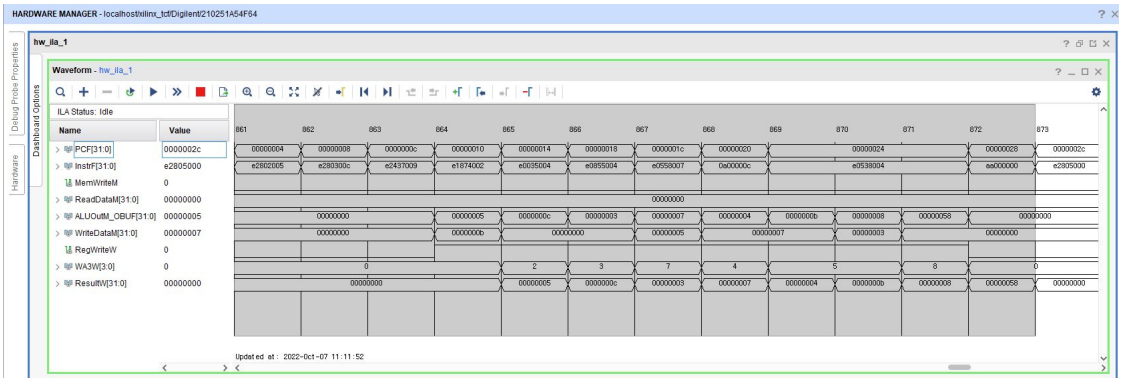

그림 7. Vivado Integrated Logic Analyzer의 파형
Fig. 7. The waveform of Vivado Integrated Logic Analyzer

기록하는 32 비트 데이터, ALU에서 계산한 32 비트 값, 레지스터화일에 기록하는 4 비트 레지스터의 번호와 32 비트값을 선택하였다. 이 값들을 관찰한 결과, 본 연구에서 설계한 파이프라인 방식의 32 비트 ARM 프로세서가 프로그램을 올바르게 실행하는 것을 확인하였다.

V. 결 론

본 논문에서는 Vivado 환경에서 VHDL을 이용하여 37 개의 명령어를 실행할 수 있는 ARM 프로세서를 설계, FPGA로 구현 및 검증하였다. 주어진 ARM 프로그램을 실행시키고 로직아날라이저로 측정한 결과, 올바르게 동작하는 것을 확인할 수 있었다.

추후로, FPGA로 검증된 본 프로세서를 Synopsys사의 Design Compiler로 합성하고, Prime Time으로 타이밍 분석의 수행 및 IC Compiler 2로 P&R을 수행하여 국내의 반도체 교육기관인 IDEC을 통하여 ASIC 칩으로 구현할 예정이다. 또한, 수퍼스칼라 프로세서와 멀티코어 프로세서에 대한 FPGA 구현 역시 가능하다.

References

[1] ARM Architecure Reference Manual, http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.subset.architecture.reference/index.html

[2] J. L. Hennessy, and D. A. Patterson, “Computer Architecture A Quantitative Approach”, 6th Edition; 2018. ISBN:978-012-811905-1

[3] S. L. Harris, and D. M. Harris, "Digital Design and Computer Architecture ARM Edition", Elsevier Korea LLC, 2016. DOI:https://doi.org/10.1016/C2018-O-14352-8

[4] M. M. Kinage and D.G. Khairnar, "Design and Implementation of FPGA Soft Core Processor for Low Power Multicore Embedded System using VHDL," Sep. 2016. DOI : https://doi.org/10.1109/ICACDOT.2016.7877603

[5] J. Lee, “Design and Simulation of ARM Processor using VHDL”, Journal of The Institute of Internet, Broadcasting and Communication, Vol. 18, No. 5, pp. 229-235, Oct 2018. DOI:https://doi.org/10.7236/JIIBC.2018.18.5.229

저 자 소 개

이 종 복(정회원)

- 1964년 8월 20일생.
- 1988년 서울대 컴퓨터공학과 졸업.
- 1998년 동 대학 전기공학부 졸업 (공박).
- 1998~2000 LG반도체 선임연구원.
- 2000년 ~ 현재 한성대 기계전자공학부 교수
- Tel : 02-760-4497
- Fax : 02-760-4435
- E-mail : jblee@hansung.ac.kr
- 관심분야 : 마이크로 프로세서, 멀티코어 프로세서, 인공지능 프로세서

※ 본 연구는 한성대학교 학술연구비 지원과제임.