JANT 한국항행학회논문지
Journal of Advanced Navigation Technology

응용 및 융합 기술

J. Adv. Navig. Technol. 21(4): 386-393, Aug. 2017

칩 테스트를 위한 UART-to-APB 인터페이스 회로의 설계

UART-to-APB Interface Circuit Design for Testing a Chip

서 영 호[1] · 김 동 욱[2*]
[1]광운대학교 인제니움학부대학
[2]광운대학교 전자재료공학과

Young-Ho Seo[1] · Dong-wook Kim[2*]
[1]Ingenium College of Liberal Arts, Kwangwoon University, Seoul, 01897, Korea
[2]Department of Electronic Materials Engineering, Kwangwoon University, Seoul, 01897, Korea

[요 약]

칩을 개발하는 과정에서 설계된 칩의 검증을 위해 FPGA (field programmable gate array)를 많이 이용한다. FPGA에 다운로드 된 회로를 검증하기 위해서는 FPGA로 데이터를 입력해야 한다. PC와 외부 보드를 통한 칩과의 통신을 위한 많은 방식이 있지만 가장 간단하고 쉬운 방법은 범용 비동기화 송수신기 (UART; universal asynchronous receiver/transmitter)를 이용한 방식이다. 최근 대부분의 회로는 AMBA (advanced microcontroller bus architecture) 버스에 연결되도록 설계되어 있다. 즉, 설계된 회로를 검증하기 위해서는 UART를 거친 후에 AMBA 버스를 통해 데이터를 전달해야 한다. AMBA 버스도 최근에 버전 4.0까지 거치면서 다양한 버전이 존재하는데 간단히 테스트를 하기 위한 용도로는 APB (advanced peripheral bus)가 적합하다. 본 논문에서는 UART-to-APB 인터페이스를 위한 회로를 설계하였다. Verilog HDL을 이용하여 설계된 회로는 Altera Cyclone FPGA에서 구현되었고, 최대 380 MHz의 속도에서 동작이 가능하였다.

[Abstract]

Field programmable gate arrays (FPGAs) are widely used for verification in chip development. In order to verify the circuit programmed to the FPGA, data must be input to the FPGA. There are many ways to communicate with a chip through a PC and an external board, but the simplest and easiest way is to use a universal asynchronous receiver/transmitter (UART). Most recently, most circuits are designed to be internally connected to the advanced microcontroller bus architecture (AMBA) bus. In other words, to verify the designed circuit easily and simply, data must be transmitted through the AMBA bus through the UART. Also the AMBA bus has been available in various versions since version 4.0 recently. Advanced peripheral bus (APB) is suitable for simple testing. In this paper, we design a circuit for UART-to-APB interface. Circuits designed using Verilog-HDL were implemented in Altera Cyclone FPGAs and were capable of operating at speeds up to 380 MHz.

Key words : Universal asynchronous receiver/transmitter, Advanced microcontroller bus architecture, Advanced peripheral bus, Field programmable gate array, Chip design.

https://doi.org/10.12673/jant.2017.21.4.386

Received 31 July 2016; **Revised** 1 August 2017
Accepted (Publication) 24 August 2017 (30 August 2017)

***Corresponding Author; Dong-wook Kim**

Tel: +82-2-940-5167
E-mail: dwkim@kw.ac.kr

Ⅰ. 서 론

최근 반도체는 디지털/정보화시대의 새로운 핵심기술 및 제품을 창조하여 인류기술 발전을 선도하는 21세기 디지털 기간산업으로서 메모리, 시스템반도체, 융합반도체, 그린반도체 기술로 발전하고 있다. 시스템반도체는 첨단 IT 수요에 연동된 미래 유망산업으로 휴대폰, 가전, 자동차 등 시스템산업의 밑바탕이 되고 있다. 최근 산업의 융・복합화가 가속화되면서 시스템반도체의 중요성이 크게 부각되고, 스마트폰, e-book, 멀티 미디어기기 등 IT 제품의 융합, 지능화, 경박단소가 가속화되고 있다. 시스템반도체는 지능형 자동차, 감성 정보기기, 헬스케어 등 異업종간 융합을 위한 요소기술로서 새로운 비즈니스 창출의 핵심이기도 하다. 에너지 분야에 반도체 기술을 융합하여 Smart Grid, 신재생에너지 등 녹색산업의 발전을 견인할 수 있으며, 저전력 반도체의 개발로 에너지 소비 절감 및 탄소배출 등 환경규제 대응이 가능하다 [1], [2].

반도체 기술은 시스템 구현에 필요한 다기능 집적반도체 기술인 시스템반도체, 고주파소자, 이차전지, 전력반도체 등을 포함하는 특화 디바이스, 정보기억 능력이 구현되는 메모리 및 반도체를 생산하기 위한 공정/장비/소재/패키지/PCB 기술을 포함한다. 시스템반도체는 제품 또는 시스템의 핵심기능을 하나의 칩에 집약하여 기기의 제어와 운용을 주도하는 반도체로, 응용 분야에 따라서 PC, 모바일, 가전, 자동차, 산업 등으로, 기능별로는 마이크로 프로세서, 로직 IC, 아날로그 IC, 센서 등으로 구분된다 [1].

본 논문에서는 칩을 개발하는 과정에서 설계된 칩을 간단한 방법으로 PC를 통해 외부에서 회로의 동작을 검증할 수 있도록 하기 위한 통신용 인터페이스 회로를 제안하고 설계하였다. 최근 거의 모든 시스템 반도체는 내부의 시스템 버스로써 AMBA 버스를 사용하고 있다 [3]. 따라서 원래 설계된 회로를 수정하지 않고, 이 버스를 최대한 활용하면서 가장 쉬운 방법의 통신 인터페이스 방식이 적합하다. 대부분의 컴퓨터는 USB를 이용한 UART 통신을 지원하고 있고, 현재에도 가장 많이 사용되는 방식이라 할 수 있다. 따라서 이 이 둘을 활용할 수 있다면 최소한의 수정만을 하면서 원래의 회로는 수정하지 않고 설계자가 설계한 회로를 매우 간단히 테스트할 수 있다. AMBA 버스에서 AXI 혹은 AHB 등의 인터페이스를 대상으로 한다면 버스 매트릭스 및 버스 제어기 등에 작업을 해야할 것이 많아지게 되어 테스트를 위한 연결 자체가 설계 과정의 부하를 가중시키고 오류의 가능성을 높이는 결과를 가져오게 된다. 따라서 AMBA 버스 중에서 회로의 상태 확인 및 프로그래밍을 위한 시스템 버스인 APB를 대상으로 인터페이스 회로를 구현하고자 한다.

본 논문은 다음과 같이 구성된다. 먼저 2장에서는 APB와 UART에 대한 칩과의 통신에 대해서 간단히 설명한다. 3장에서는 제안한 하드웨어 구조와 핵심 동작에 대해서 설명한다. 다음으로 4장에서는 구현 결과를 보이고 5장에서 결론을 맺는다.

Ⅱ. 칩과의 통신

본 논문에서 제안하는 회로는 칩 외부에서는 UART를 이용하고 칩 내부에서는 AMBA APB를 이용한다. 본 장에서는 제안하고자 하는 회로의 이해를 돕기 위해 UART와 APB에 대한 개요를 간단히 소개하고자 한다. 이 두 가지 프로토콜에 대한 자세한 설명은 참고문헌으로 대신하고자 한다.

2-1 APB

APB (advanced peripheral bus)는 AMBA (advanced microprocessor bus architecture) 3 프로토콜 제품군의 일부이고, 저비용의 인터페이스를 제공한다. 최소 전력 소비 및 인터페이스 복잡성을 최소화하도록 최적화되어 있다. AHB는 전이중 병렬 통신을 사용하고 온 칩 버스 표준을 사용한다. APB에는 파이프라이닝이 없고, APB는 주로 간단한 주변 장치에 연결하기 위해 제안되었다. APB는 주변 장치 기능을 지원하기 위해 인터페이스 복잡성을 줄이고 전력 소비를 최소화하기 위해 최적화되는 경우도 있다. 즉, APB는 낮은 대역폭을 필요로 하지 않는 모든 주변 장치와 인터페이스 한다. 고성능의 라인 된 버스 인터페이스를 필요로 하지 않는다. 모든 동작은 클럭의 상승 에지에만 동기가 되어있어 APB 주변 장치를 어떤 회로에도 쉽게 연결할 수 있다. 모든 동작은 최소 2싸이클 동작으로 구성되어 있다. APB는 AMBA Advanced High-performance Bus Lite (AHB-Lite)와 인터페이스 할 수 있고, AMBA AXI (advanced extensible interface)를 지원한다. 주변 장치의 프로그래밍 가능한 제어 레지스터에 대한 액세스를 위해 주로 사용된다. AHB의 기능을 살펴보면 단일 에지 클록 프로토콜, 여러 버스 마스터, 분할 트랜잭션, 단일 사이클 버스 마스터 핸드 오버, 버스트 전송, 대형 버스 폭 및 비 삼중 코드 구현이 있다. AHB에서 트랜잭션은 주소 단계와 데이터 단계로 구성된다 [4], [5].

2-2 UART

범용 비동기화 송수신기 (UART; universal asynchronous receiver/transmitter)는 병렬 데이터의 형태를 직렬 방식으로 전환하여 데이터를 전송하는 컴퓨터 하드웨어의 일종이다. UART는 일반적으로 EIA RS-232, RS-422, RS-485와 같은 통신 표준과 함께 사용한다. UART의 U는 범용을 가리키는데 이는 자료 형태나 전송 속도를 직접 구성할 수 있고 실제 전기 신호 수준과 방식(이를테면 차분 신호)이 일반적으로 UART 바깥의 특정한 드라이버 회로를 통해 관리를 받는다는 뜻이다. 통신 데이터는 메모리 또는 레지스터에 들어 있어 이것을 차례대로 읽어 직렬화 하여 통신한다. 최대 8 비트가 기본 단위이다. UART는 일반적으로 컴퓨터나 주변 기기의 일종으로 병렬 데이터를 직렬화 하여 통신하는 개별 집적 회로이다. 비동기 통신이므로 동기 신호가 전달되지 않는다. 따라서 수신 쪽에서 동기신호를 찾아내어 데이터의 시작과 끝을 시간적으로 알아 처리할 수 있도록 약속되어 있다. 디지털 회로는 자체의 클럭 신호를 사용하여 정해진 속도로 수신 데이터로 부터 비트 구간을 구분하고 그

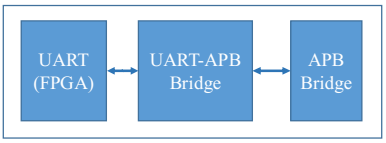

그림 1. 제안한 하드웨어의 전체 구조.
Fig. 1. Whole architecture of the proposed hardware.

비트의 논리 상태를 결정하여 데이터 통신을 한다. UART는 보통 마이크로컨트롤러에도 포함되어 있다. 듀얼 UART, 곧 DUART는 두 개의 UART를 하나의 칩에 합친 것이다. 수많은 현대의 집적 회로(IC)는 동기화 통신도 지원하는 UART와 함께 한다. 이러한 장치들은 범용 동기화 송수신기 (USARTs; universal synchronous/asynchronous receiver/transmitter)로 부른다 [6].

Ⅲ. 하드웨어 구조

본 장에서는 전체적인 하드웨어 구조와 핵심적인 동작에 대해서 나타낸다.

3-1 전체 하드웨어 구조 및 동작

본 절에서는 제안한 하드웨어의 전체 구조에 대해서 설명한다. 그림 1에 전체적인 하드웨어 구조를 간단히 나타냈다. 전체 하드웨어는 크게 세 가지 부분으로 나누어지는데, 여기에는 UART(FPGA), UART-APB 브리지(bridge), 그리고 APB 브리지(bridge)가 포함된다. UART(FPGA) 회로는 외부 장치로부터 UART 프로토콜에 따라서 입력된 데이터를 전송 받거나 외부 장치로 전송하는 역할을 수행한다. UART-APB 브리지 회로는 UART 데이터를 입력받아서 분석하고 적절한 동작을 결정하는 역할을 수행한다. 이 회로를 통해서 명령어(OPCODE)와 데이터, 그리고 제어 신호들을 UART 데이터로부터 추출하고, 이 신호들을 APB 브리지 회로로 전송하여 APB 브리지에 연결된 APB 슬레이브 회로들에 적절한 데이터를 전달한다. 또한 APB 슬레이브 회로들로부터 전송되어 오는 데이터들을 UART(PC)와 UART(FPGA) 사이의 상태를 확인하면서 UART(FPGA)를 거쳐서 UART(PC)로 전송할 수도 있다. UART 프로토콜은 저속으로 동작하고 APB 프로토콜은 그에 비해서는 고속으로 동작한다. 즉, UART-APB 브리지 회로는 클럭 속도가 다른 두 영역 사이를 버퍼링하는 역할도 수행한다.

그림 2에는 제안한 하드웨어의 자세한 구조를 나타냈고, 여기에는 중요한 신호들의 연결 상태를 모두 나타냈다. 또한 그림 1과 비교할 때 APB SLAVE 회로가 추가된 것을 볼 수 있는데 APB 슬레이브 회로는 실제의 APB 슬레이브가 아니고 APB 슬레이브로 사용될, 즉 테스트할 회로를 쉽게 연결할 수 있도록 APB 프로토콜을 단순한 읽고, 쓰기 동작으로 변환해 주는 역할을 담당한다. 즉, APB 슬레이브 회로는 일종의 APB Wrapper에 해당한다 [7]. 따라서 이를 통해 매우 쉽게 모든 회로를 제안한 회로에 연결할 수 있어서 검증을 위한 준비 시간을 줄여주고 생산성을 향상할 수 있도록 하였다. 본 논문에서 구현한 회로에서는 2개의 APB 슬레이브 회로를 내장하도록 하였으나 슬레이브를 선택하는 신호를 8 비트로 구현하였기 때문에 최대 256개의 슬레이브 회로를 연결할 수 있다.

3-2 UART와 APB 사이의 프로토콜 변환

그림 1과 그림 2에서 보인 전체 하드웨어의 구조에서 가장 핵심적인 회로는 UART-APB 브리지 회로이다. 그림 3에 UART-APB 브리지 회로를 자세하게 나타냈다. UART-APB 브리지 회로는 크게 두 부분으로 나누어지는데, 그림 3에서 윗 부분은 외부 장치에서 입력된 신호를 처리하기 위한 회로이고, 아래 부분은 APB 슬레이브에서 입력된 데이터를 외부 장치로 전송하기 위한 회로이다. 아래의 회로에서 각 전송 선로 상의 상태를 확인하기 위한 신호는 생략되어 있다. UART를 통해 입력

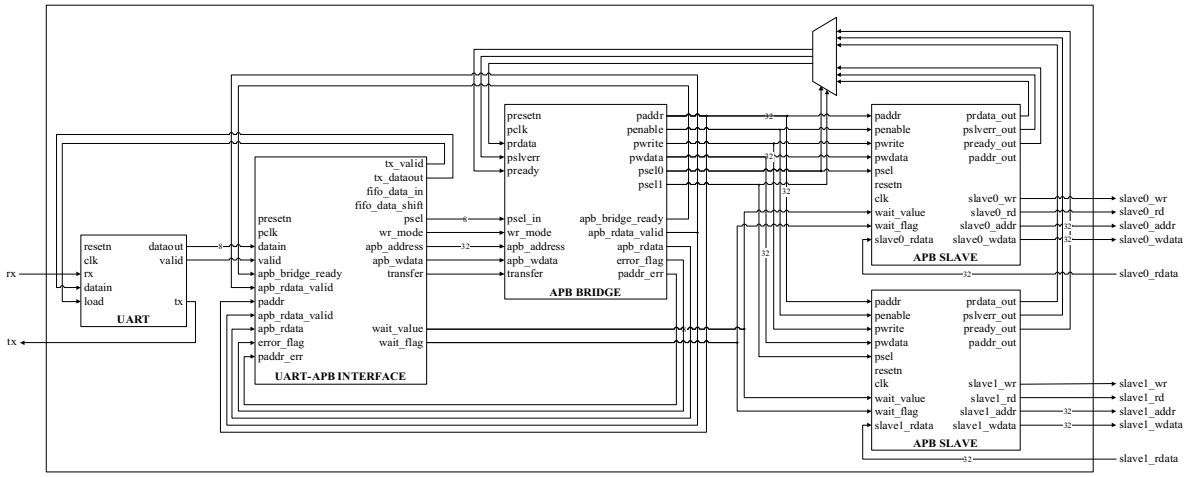

그림 2. 제안한 하드웨어의 세부 구조.
Fig. 2. Detail architecture of the proposed hardware.

된 데이터는 OPCODE Decoder를 거치면서 이것이 어떤 데이터인지 분석된다. 표 1에 OPCODE의 내용을 정의하였다.

먼저 프로그래밍 레지스터(programming register)와 동작 레지스터(operation register)를 프로그래밍하여 전체 회로가 어떤 모드가 될 것인지와 전체 회로가 어떤 동작을 할 것인지 결정한다. 프로그래밍 레지스터는 표 2와 같이 정의된다. 다음으로 주소 레지스터(address register) 혹은 데이터 레지스터(Data Register)에 주소 및 데이터가 입력된다. 프로그래밍 레지스터의 모드에 따라서 주소와 데이터는 8 비트부터 32 비트까지 다양한 비트로 정의되고 사용될 수 있다.

프로그래밍된 동작 조건에 맞추어서 주소 및 데이터가 모두 입력되면 Transfer 신호가 트리거 하여 APB 브리지 회로가 동작을 시작하게 된다. Transfer 신호는 하나의 동작을 위한 시작 신호에 해당하고, 이를 기준으로 그 이후의 회로들이 동작을 수행한다. Transfer 신호는 Write Operation Controller에 의해서 생성되는데 이 회로는 이전에 프로그래밍된 값들과 상태들을 관찰할 이후에 프로그래밍 조건에 따라 데이터들이 모두 입력된 것을 확인하면 곧바로 Transfer 신호를 출력하여 APB 브리지 회로가 읽기 혹은 쓰기 동작을 할 수 있도록 명령을 전달한다.

표 1에는 OPCODE를 정의하였는데, OPCODE에는 동작의 시작을 알리고 전송 에러가 발생하였을 때 시작점을 다시 회귀시키는 Start Code, 전체 회로의 모드를 결정하는 Programming Register, 전체 회로의 중요 상태를 저장해서 외부 장치가 확인할 수 있도록 하는 Status Register, 회로의 동작을 정의하는 Operation Register, APB 슬레이브에 쓸 데이터를 저장하는 Write Data Register, APB 슬레이브로부터 읽은 데이터를 저장하는 Read Data Register, 읽기와 쓰기를 위한 주소를 저장하는 Address Register 등의 동작을 위한 것들로 구성된다.

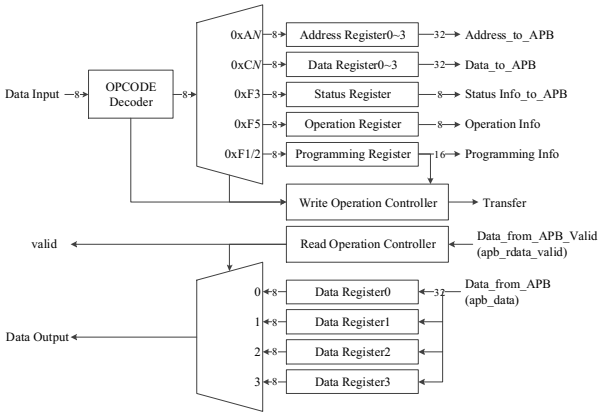

그림 3. UART-APB 브리지의 구조.
Fig. 3. Architecture of the UART-APB bridge.

표 1. OPCODE의 정의.
Table 1. Definition of OPCODE.

#	Code	Operation
1	0xFFFF	Start Code
2	0xF0	Programming Register0
3	0xF1	Programming Register1
4	0xF3	Status Register
5	0xF5	Operation Register
6	0xC3	Write Data Register 3
7	0xC2	Write Data Register 2
8	0xC1	Write Data Register 1
9	0xC0	Write Data Register 0
10	0xD3	Read Data Register 3
11	0xD2	Read Data Register 2
12	0xD1	Read Data Register 1
13	0xD0	Read Data Register 0
14	0xA3	Address Register 3
15	0xA2	Address Register 2
16	0xA1	Address Register 1
17	0xA0	Address Register 0
18	0xB0	Info Register

표 2. 프로그래밍 레지스터의 정의.
Table 2. Definition of programming register.

Register 0	7	6	5	4	3	2	1	0
	Data Width		Address Width		Operation Mode			RW
Register 1	7	6	5	4	3	2	1	0
	NA	Slave Select			Wait Offset			Reset

표 2에는 레지스터 중에서 가장 중요한 프로그래밍 레지스터에 대해 정의하였다. RW는 읽기 동작인지 쓰기 동작인지 나타낸다. Operation Mode는 크게 3가지로 구성되는데, 일반적인 읽기 및 쓰기 동작, 에코 동작(연속된 쓰기 및 읽기), 그리고 FIFO를 연결할 경우에 FIFO에 데이터를 저장한 후에 처리하는 모드로 구분된다. Address 및 Data Width는 각각 주소와 데이터의 비트 너비를 나타내고, 8, 14, 24, 32 비트의 네 가지 모드가 존재한다. Reset은 회로를 논리적으로 초기화시키기 위한 것이고, Wait Offset은 슬레이브들이 테스트를 위해서 임의의 값으로 APB 프로토콜을 지연시키기 위한 것이다. Slave Select는 테스트할 슬레이브를 임의로 지정하기 위한 것이다. 이러한 16 비트의 모드를 통해서 다양한 조건에서 다양한 슬레이브에 연결된 회로들을 테스트할 수 있다.

3-3 응답 확인 방식 동작

PC를 기준으로 크게 읽기와 쓰기의 두 가지 동작으로 구성된다. UART의 동작은 매우 느리고, APB의 동작은 그에 비해 빠른 동작이다. 따라서 둘 사이의 올바른 동작을 위해서는 서로의 동작 상태를 확인하면서 읽기와 쓰기 동작을 수행해야 한다. PC의 UART와 FPGA의 UART는 상대적으로 느리기 때문에 UART를 거쳐서 APB로 전송되는 동작은 확인없이 진행될 수 있다. UART에서 사용되는 클록이 한 사이클이 지나기 전에 대부분의 APB 측의 동작은 완료되기 때문이다. APB의 slave로부터 UART로 데이터가 전송될 경우에는, 즉 PC측의 읽기 동작이 수행되고 있는 상황에서는 APB는 UART의 상태를 확인해야 한다.

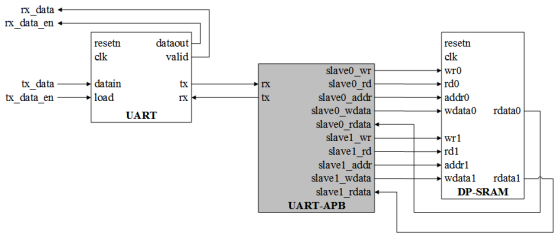

그림 4. 테스트 설계 환경.
Fig. 4. Test design environment.

Ⅳ. 구현 결과

본 논문에서 제안한 하드웨어는 Verilog-HDL을 이용하여 구현되어 Altera사의 Cyclone III EP3C16F484C6 FPGA를 타겟으로 합성하였고, Modelsim을 이용하여 시뮬레이션을 수행하여 동작을 검증하였다 [8], [9]. FPGA에서 합성된 회로는 약 105개의 셀을 이용하여 구현되었고, 총 217개의 입출력 포트를 이용한다. 또한 지정한 FPGA에서 최대 380 MHz의 클록 주파수에서 동작이 가능하였다.

그림 4에는 구현한 하드웨어를 테스트하기 위해서 회로의 입력과 출력에 두 개의 회로를 더 구현하여 연결하였다. 먼저 UART 통신을 위해 PC쪽의 UART 역할을 수행할 수 있도록 UART(PC)를 UART(FPGA)와 동일한 것으로 연결하였고, 2개의 슬레이브 회로를 위해서 듀얼 포트 SRAM(synchronous random access memory)를 사용하여 연결하였다. 즉, 2개의 슬레이브를 따로 구현한 것이 아니고 1개의 듀얼 포트 회로를 이용하여 2개의 역할을 할 수 있도록 하였다.

그림 5에는 Quartus 환경에서 합성한 결과의 RTL(register transfer level) 회로로 추출한 것을 캡쳐한 그림이다. 툴의 특성상 방향이 그림 2와 반대인 것을 확인할 수 있는데, 그림 2와 비교하면 구조가 정확히 동일하다는 것을 확인할 수 있다.

그림 6에서 그림 10까지는 구현한 하드웨어의 시뮬레이션 검증 결과를 자세히 나타냈다. 시뮬레이션은 ModelSim 시뮬레이터 환경을 이용하였고, 기능(functional) 시뮬레이션을 통해 설계된 회로의 동작을 수정하였다. 각각의 그림에는 다양한 쓰

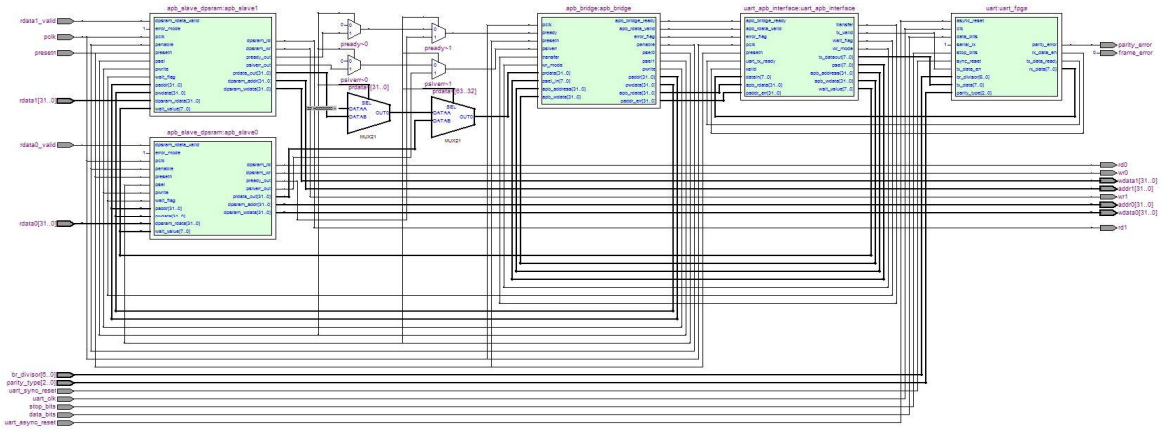

그림 5. FPGA 합성의 RTL 결과.
Fig. 5. RTL result from FPGA synthesis.

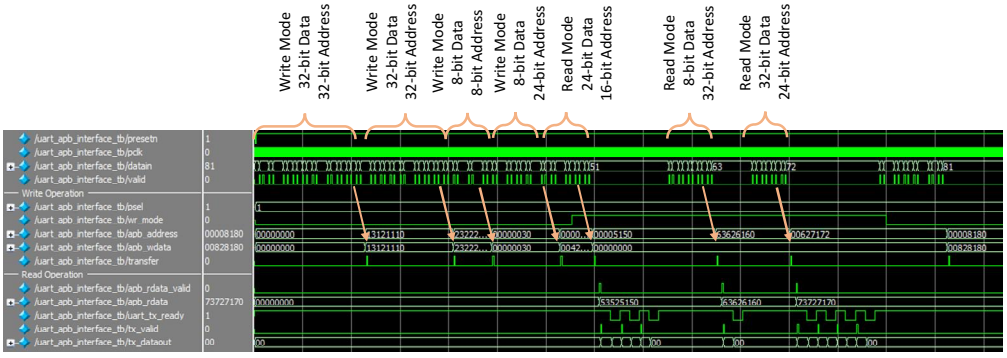

그림 6. UART-APB 브릿지의 동작.
Fig. 6. Operation of the UART-APB bridge.

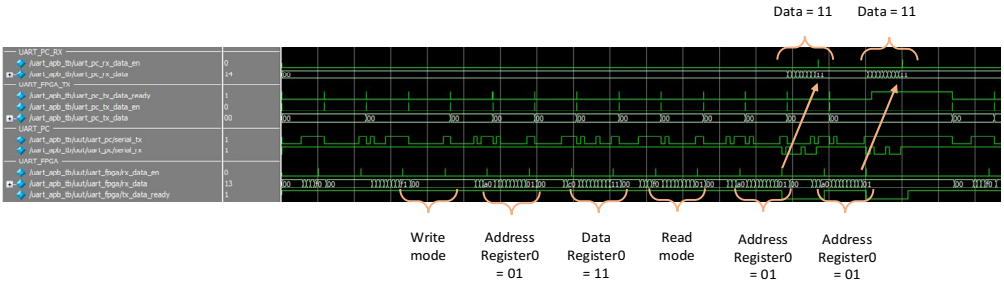

그림 7. 전체 하드웨어의 시뮬레이션 결과.
Fig. 7. Simulation result of the whole hardware.

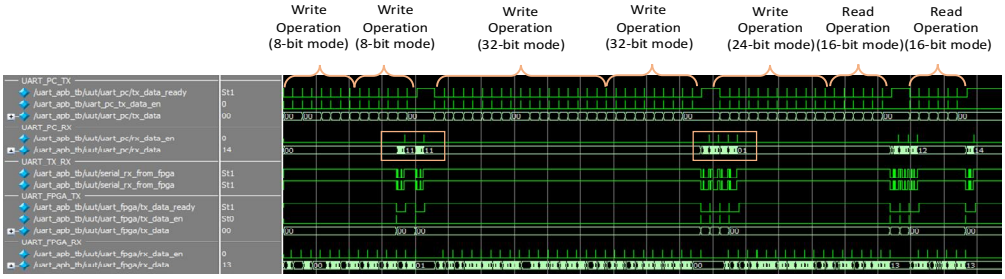

그림 8. 쓰고 읽기 동작 결과.
Fig. 8. Operation result of the write and read .

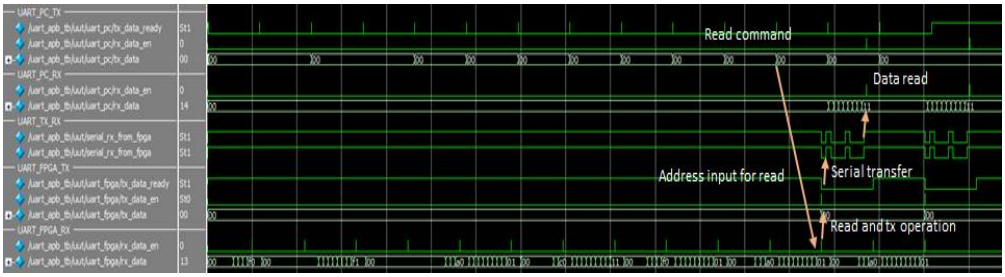

그림 9. 상세한 읽기 동작.
Fig. 9. Detail read operation.

기와 다양한 읽기 동작을 나타내고 있다. 그림들에서의 신호 이름은 그림 2와 그림 3에 나타낸 신호의 이름과 동일하다.

그림 6에는 가장 핵심 동작인 UART-APB 브리지의 동작을 시뮬레이션하였다. 주소와 데이터가 다양한 비트너비를 갖는 조건으로 세 가지 쓰기 동작(데이터비트/주소비트 - 32/32, 8/8, 8/24)을 수행하고 연속적으로 세 번의 읽기 동작(데이터비트/주소비트 - 24/16, 8/32, 32/24)을 수행하는 것을 보이고 있다.

그림 7은 전체 하드웨어의 시뮬레이션 결과를 보이고 있다. 신호들은 크게 UART_PC와 UART_FPGA에 대해 RX와 TX로 구분하였다. 또한 UART_TX_RX는 두 UART 사이에 전송되는 신호를 나타낸 것이다. UART_PC_TX를 통해서 사용자가 쓰기 동작을 위해 데이터를 입력하면 UART_TX_RX를 거친 후에 UART_FPGA_RX로 전송된다. 사용자가 읽기 동작을 하고자 한다면 읽기 동작을 위한 데이터들을 UART_PC_TX를 통해 입력하여 UART_FPGA_RX로 전달되고, 슬레이브를 통해 읽혀진 데이터들은 UART_FPGA_RX를 통해 입력된 후에 UART_PC_RX로 최종적으로 전달되어 사용자가 확인할 수 있게 된다.

그림 8은 그림 7에서 데이터를 쓰고 읽는 부분을 확대하여 한 부분만 표시한 것이다. 슬레이브로 사용된 DP-SRAM의 0x01 번지에 0x11 값을 쓰고 두 번 같은 주소를 읽는 동작을 나타내고 있다.

그림 9는 읽는 동작이 어떻게 이루어지는지 상세한 타이밍도를 나타냈다. 읽기 명령(read command)이 입력되면 APB 브리지가 데이터를 읽어온 후(address input for read)에 이를 UART_TX_RX를 통해 전송(serial transfer)하고 이것이 최종적으로 UART_PC_RX에 나타나는(data read) 과정을 보이고 있다.

그림 10에는 APB 브리지와 APB 슬레이브 간에 데이터를 송수신하는 것을 나타내고 있다. 그림 10(a)와 그림 10(c)는 APB 표준안에 정의된 프로토콜이고, 그림 10(b)와 그림 10(d)는 시

뮬레이션을 통해 검증된 결과이다. 각 시뮬레이션 결과를 살펴보면 표준안에서 정의된 프로토콜과 정확히 동일하게 동작하는 것을 확인할 수 있다.

Ⅴ. 결 론

본 논문에서는 칩을 개발하는 과정에서 설계된 칩을 간단한 방법으로 검증할 수 있도록 하기 위한 통신용 인터페이스 회로를 제안하고 설계하였다. 본 논문에서 제안한 하드웨어는 Altera사의 Cyclone III EP3C16F484C6 FPGA를 타겟으로 Quartus를 이용하여 합성하였고, Modelsim을 이용하여 동작을 검증하였다. FPGA에서 합성된 회로는 약 105개의 셀을 이용하여 구현되었고, 총 217개의 입출력 포트를 이용한다. 또한 지정한 FPGA에서 최대 380 MHz의 클록 주파수에서 동작이 가능하였다. 본 논문을 통해 구현한 회로는 PC와 외부 보드를 통한 칩과의 통신을 위한 많은 방식이 있지만 가장 간단하고 쉬운 방법인 UART와 대부분의 회로가 사용하고 있는 AMBA 버스에 연결되도록 설계되어 설계자가 설계한 회로를 매우 간단히 테스트할 수 있도록 하였다.

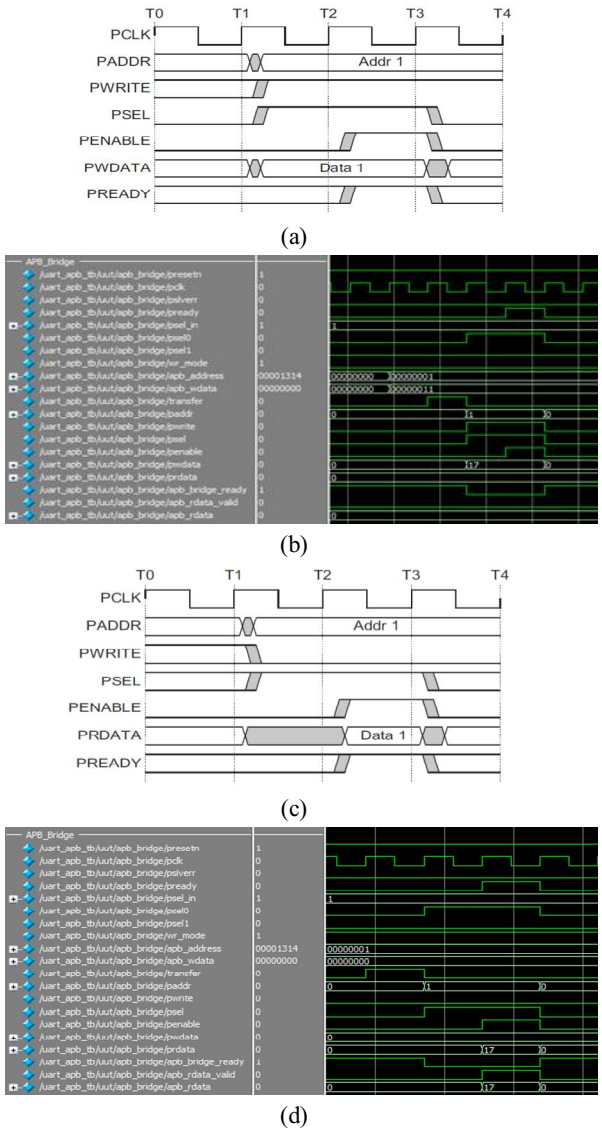

그림 10. APB 브리지의 동작 (a) 쓰기 동작 프로토콜, (b) 쓰기 동작 결과, (c) 읽기 동작 프로토콜, (d) 읽기 동작 결과.
Fig. 10. Operation of the APB bridge (a) write protocol, (b) write operation, (c) read protocol, (d) read operation.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT and Future Planning (NRF-2014R1A2A1A11052433)

References

[1] W. J. Kim, S. S. Park, and H. B. Jung, “Trends and forecasts of the system semiconductor industry,” *Weekly Technology Trends*, Vol. 1462, pp. 1-13, Sep. 2010.

[2] J. L. Burns, “Technology trends and implications on SoC design,” in *2011 IEEE International SOC Conference*, Taipai, pp.386-386, Sep. 2011.

[3] A. Paunikar, R. N. Umarikar, and K. Sivasankaran, “Design and implementation of area efficient, low power AMBA-APB bridge for SoC,” in *2014 International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE),* Coimbatore, pp.1-6, Mar. 2014.

[4] AMBA™ 3 APB Protocol Specification, ARM, ARM IHI 0024B, pp. 1-34, Sep. 2003.

[5] M. Roopa, R. M. Vani, and P. V. Hunagund, “UART controller as AMBA APB slave,” in *National Conference on Challenges in Research & Technology in the Coming Decades (CRT 2013),* Ujire, pp.1-6, Sep. 2013.

[6] Wikipedia. Universal asynchronous receiver/transmitter [internet]. Available : https://en.wikipedia.org/wiki/Universal_asynchronous_receiver/transmitter

[7] G. Ma, and H. He, “Design and implementation of an advanced DMA controller on AMBA-based SoC,” in *2009 IEEE 8th International Conference on ASIC*, Hunan, pp. 419–422, Oct. 2009.

[8] Intel. Cyclone FPGAs series [Internet]. Available : https://www.altera.com/products/fpga/cyclone-series.html

[9] Mentor. Modelsim [internet]. Available : https://www.mentor.com/products/fv/modelsim/

서 영 호 (Young-Ho Seo)
2004년 2월 : 광운대학교 전자재료공학과 (공학박사)
2003년 ~ 2004년 : 한국전기연구원 연구원
2005년 ~ 2008년 : 한성대학교 조교수
2008년 ~ 현재 : 광운대학교 인제니움학부대학 교수
※관심분야 : 실감미디어, 2D/3D영상 신호처리, 디지털 홀로그램

김 동 욱 (Dong-Wook Kim)
1983년 2월 : 한양대학교 전자공학과 졸업(공학사)
1985년 2월 : 한양대학교 공학석사
1991년 9월 : Georgia공과대학 전기공학과(공학박사)
1992년 3월 ~ 현재 : 광운대학교 전자재료공학과 정교수
※관심분야 : 3D 영상처리, 디지털 홀로그램, 디지털 VLSI Testability, VLSI CAD, DSP설계