Regular paper http://dx.doi.org/10.13067/JKIECS.2013.8.11.1697

SD 프로토콜 분석기 설계

문지훈* · 오재철**

Design of the SD Protocol Analyzer

Ji-Hoon Moon* · Jae-Chul Oh**

요 약

SD 슬레이브 IP 개발 시 CMD 및 데이터가 정상적으로 처리되는지 분석하기 위해서 프로토콜 분석기를 이용하고 있다. 본 논문에서는 윈도우 환경에서 Visual C++를 이용하여 SD 프로토콜을 분석할 수 있는 프로토콜 분석기를 개발하였다. SD 프로토콜 분석기는 SD 메모리 데이터를 저장하기 위한 임베디드 리눅스 소프트웨어와 이를 분석하기 위한 MFC 프로그램으로 구성되며, 프로토콜 분석은 SD 메모리 카드에서 호스트로 송수신되는 데이터를 리눅스 소프트웨어에서 수집하여 MFC에서 분석할 수 있도록 설계하였다. 실험 결과 개발된 보드를 이용하여 SD 메모리 카드에 데이터를 읽거나 기록할 때 발생되는 CMD 종류의 확인이 가능 하였으며, 문제가 발생되는 부분에 대한 디버깅이 가능하였다.

ABSTRACT

Protocol analyzer is being used to analyze proper processing of CMD & data when developing SD slave IP. In this thesis, a protocol analyzer was developed for analyzing SD protocol in Windows environment using Visual C++. SD protocol analyzer consists of embedded Linux software for storing SD memory data and MFC program for analyzing this. As for protocol analysis, it has been designed to collect data transmitted from SD memory card to host by Linux software for its analysis by MFC. It was found through the experiment that the CMD type could be confirmed that occurs when reading and writing data to SD memory card using the developed board, and debugging the problems that occur was possible.

키워드

Memory Card, Protocol Analyzer, Device Driver
메모리카드, 프로토콜 분석기, 디바이스 드라이버

Ⅰ. 서 론

IT 환경의 발달로 3D-TV, VOD등과 같은 다양한 영상정보매체 기술은 매우 다양한 분야로 발전을 거듭하고 있다[1]. 이러한 장비 및 서비스들의 발전 배경에는 이를 처리하는 마이크로 프로세서 발전 뿐만 아니라 위의 데이터를 처리할 수 있는 임베디드 기기의 발전으로 가능하게 되었다.

임베디드 기술은 특정한 제품이나 솔루션에 주어진 작업을 수행할 수 있도록 추가로 탑재되는 임베디드 시스템(차세대 성장동력 분야를 비롯한 정보가전 및 정보통신기기, 항공기, 차량, 로봇, 산업기기, 의료기기

* 순천대학교 컴퓨터과학과(p3161@naver.com)
**교신저자(corresponding author) : 순천대학교 컴퓨터과학과(ojc@sunchon.ac.kr)
접수일자 : 2013. 09. 16 심사(수정)일자 : 2013. 10. 21 게재확정일자 : 2013. 11. 15

등)에 내장되어 하드웨어의 제어, 통신, 멀티미디어, 인터넷, 게임, 인공지능, 유비쿼터스 컴퓨팅 등 기본 기능 및 다양한 부가기능을 제공하는 기술이다[2]. 이러한 임베디드 기기의 특성상 데이터 저장을 위해서 낸드 플래시 메모리를 필요로 하며, 특정 기기들은 사용자의 데이터를 저장하기 위해서 SD 메모리 카드를 이용한다. 특히 PDA, 모바일 단말기등은 소형의 내장형 기기로 이동중에도 데이터의 접근을 가능케 함으로 최근 인기를 끌고 있다[3].

휴대 저장 장치의 대중적인 보급으로 SD 메모리 카드의 사용이 확산되고 있다. SD 메모리 카드는 저장장치로 대부분 낸드 플래시 메모리를 이용하고 있다[4]. SD 카드는 복사 방지를 위한 SDMI 보안 표준에 만족하며 빠른 속도와 대용량의 메모리 카드이다. SD 메모리 카드는 상호 보안 알고리즘을 사용하여 보안을 유지할 수 있는 특징을 가지고 있다[5].

SD 메모리 카드 슬레이브 IP를 개발할 경우 여러 종류의 SD 호스트가 존재하는 문제로 인하여 모든 SD 호스트를 지원하는 카드를 개발하는 것이 쉽지 않다. 또한 SD 슬레이브 IP 사용 시 호스트와의 통신상의 오류가 발생한 경우, 프로토콜 분석기를 사용하지 않는다면 정확한 디버깅을 수행하기가 어렵다. SD 호스트는 CMD 라인을 이용하여 요청할 명령을 메모리 카드에 전달하며, 원하는 데이터를 읽거나 기록할 경우는 데이터 라인을 이용한다. 만일 메모리 카드가 호스트와 통신이 되지 않을 경우 정확한 원인을 찾기 위해서는 들어오는 데이터 및 CMD를 분석 하여야 한다.

이러한 문제점을 해결하기 위해, 본 논문에서는 SDIO 기능을 이용한 SD 프로토콜 분석기를 제안한다. SD 호스트 및 메모리 카드에서 전달되는 CMD 및 데이터 라인의 클럭 정보를 FPGA에서 받아들인 후, 위의 정보를 S3C2450 프로세서가 탑재된 임베디드 리눅스의 SROM 인터페이스를 이용하여 데이터를 수집한다. 수집한 정보는 TCP/IP를 통하여 Visual C++로 작성된 SD 프로토콜 분석 소프트웨어로 전달되어 호스트와 SD 메모리 카드 사이의 데이터를 분석하여 문제점 분석이 가능함을 확인 하였다.

이후 본 논문의 구성은 다음과 같다. 2장에서 관련된 연구들을 소개하고, 3장에서 본 논문에서 SD 프로토콜 분석기 설계를 기술하고, 4장에서 제안된 SD 프로토콜 분석기에 대한 실험을 나타내며, 마지막 5장에서 결론을 도출한다.

II. 관련연구

2.1SD메모리카드프로토콜

SD 카드는 SD(secure digital) 모드와 SPI(Serial Peripheral) 모드의 통신 프로토콜을 지원한다. 호스트는 이러한 통신 프로토콜을 리셋 커맨드를 읽어 들임으로서 알 수 있다[5][6][7].

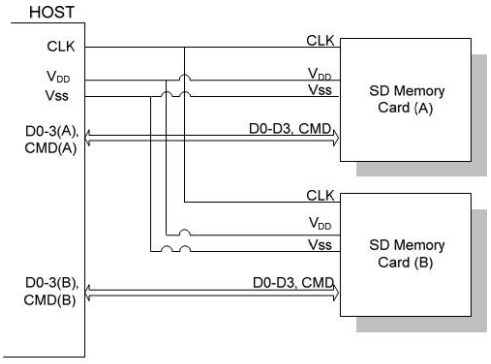

그림 1. SD 메모리 카드 시스템 버스 토폴로지
Fig. 1 SD memory card system bus topology

SD 버스는 그림 1과 같이 호스트와 카드간의 클럭 신호를 위한 CLK, 양방향으로 명령 송수신 및 응답을 받기 위한 커맨드, 4개의 양방향 데이터 라인, 전원 및 그라운드 라인이 존재한다. 커맨드와 데이터 라인은 여러 개의 카드에 일대일로 연결되고 전원 및 그라운드 라인은 공유하여 연결된다.

표1은 SD 버스 프로토콜 구성을 나타낸다. SD 버스에서의 통신은 스타트 비트에 의해서 초기화가 이루어지고 스톱 비트에 의해서 종료되는 커맨드와 데이터 비트 스트림에 의해 이루어진다. 스타트 비트에 의해서 커맨드의 시작이 이루어지며, 스톱 비트에 의해서 해당 커맨드의 종료를 알 수 있다. 리스폰스는 커맨드에 대한 응답을 나타낸다. 커맨드에 따라서 호출되는 리스폰스 타입이 결정 되며, 특정 커맨드는 리스폰스를 갖지 않는 타입도 존재한다. 리스폰스는 커맨드 라인을 이용하여 호스트에서 요구하는 정보를

전달한다. 데이터는 SD 인터페이스의 데이터 라인을 이용하여 전달하며, 호스트에서 요청하거나 전달하는 데이터는 SD 인터페이스의 데이터 라인을 이용한다.

표 1. SD 버스 프로토콜 구성
Table 1. Composition of SD bus protocol

Type	Description
CMD	Command is a token that starts an operation. A command is sent from the host either to a single card(addressed command) or to all connected cards(broadcast command). A command is transferred serially on the CMD line.
Response	Response is a token that is sent from an addressed card, or (synchronously) from all connected cards, to the host as an answer to a previously received command. A response is transferred serially on the CMD line.
Data	Data can be transferred from the card to host or vice versa. Data is transferred via the data lines.

2.2 SD커맨드및리스폰스

SD 메모리 카드를 제어하기 위한 4개의 커맨드 타입이 존재한다. broadcast command(bc)는 response가 존재하지 않는 형식이며, broadcast commands with response(bcr)는 SD 호스트에서 커맨드 전달시 SD 슬레이브로부터 리스폰스를 필요로 한다. address command(AC)는 SD 메모리 카드의 데이터 라인에 데이터를 전송하지 않는 타입이며, addressed data transfer commands(adtc)는 데이터 라인에 데이터를 전달하는 타입이다. SD 메모리 카드에서 모든 커맨드와 리스폰스는 카드의 커맨드 라인을 통해서 이루어진다.

모든 커맨드는 48비트의 고정된 크기를 가지며, 25MHz 사용시 1.92us를 50MHz 사용시 0.96us를 요구한다. SD 메모리 카드에서 25MHz 클럭 사용시 데이터 라인 하나만 사용하며, 50MHz 이용시 카드의 모든 데이터 라인을 이용하여 데이터를 전송한다. 커맨드 형식은 시작 비트를 항상 0으로 시작하며, 다음 6비트는 커맨드 인덱스를 나타낸다. 몇몇 커맨드는 아규먼트를 필요로 하며, 크기는 32비트이다[5]. 그리고 CRC7 7비트와 커맨드의 끝을 알리는 end bit 1비트로 구성이 된다.

리스폰스 또한 커맨드와 동일하게 SD 메모리 카드의 CMD 라인을 이용하여 전송한다. 리스폰스의 종류는 R1(normal response command), R2(CID, CSD register), R3(OCR register), R6(published RCA response), R7(card interface condition)으로 나누어진다. R1 리스폰스의 코드 길이는 커맨드와 동일하게 48비트로 구성된다. 커맨드와 다르게 리스폰스의 경우 transmission bit가 '0'이며, 45:40 비트에 리스폰스된 커맨드 인덱스를 나타낸다. 39:8 비트에 card status를 7:1 비트에 CRC 마지막 비트에 end bit인 '1'로 구성된다. R2의 코드 길이는 136비트이다. 133:128 비트는 '1111'값으로 예약되어 있으며, 127:1 비트에 CID 및 CSD 레지스터 값이 전송된다. CID 레지스터는 제조사 ID, 제품명, 제조일자, 시리얼 번호에 대한 정보를 포함한다[6]. CSD 레지스터는 SD 카드 레지스터 중에서 가장 중요한 레지스터이며, SD 카드 설정과 관련된 레지스터이다. 이 레지스터를 통하여 메모리 카드의 용량 설정 등이 결정되며, 이 레지스터의 초기 설정 값이 정확하지 않을 경우 SD 메모리 카드가 정상 동작하지 않는다.

R3은 OCR 레지스터에 대한 리스폰스이며, 코드 길이는 48비트의 크기를 갖는다. R3를 사용하는 커맨드가 ACMD41이며, 이 명령어는 SD 카드 초기화를 시작하는 커맨드이며, 초기화를 완료 하였는지 검사하는데 사용된다. 첫 번째 ACMD41 전에는 반드시 CMD8이 발행 되어야 한다. R3의 45:40 비트 값은 '111111'으로 reserved 되어 있으며, 7:1 비트 또한 '111111'으로 reserved 되어 있다.

R6은 48비트의 크기를 가지며, 45:40 비트는 리스폰스된 커맨드 인덱스를 나타낸다. R6의 경우 이 비트값은 항상 '000011'값 즉, CMD3을 의미한다. 비트 39:8은 아규먼트 필드를 나타내며, 상위 16비트는 published RCA number로 사용되고, 하위 16비트는 card status bit로 사용된다. 하위 8비트는 다른 리스폰스와 마찬가지로 CRC7 및 end bit로 사용된다[6].

R7의 코드 길이는 R6와 동일하게 48비트이다. 카드 지원 전압 정보는 CMD8의 리스폰스로 전송된다. 리스폰스 R7에서의 19:16 비트는 카드가 제공하는 전압 범위를 나타낸다. 45:40 비트는 커맨드 인덱스를 나타내는데, '001000'으로 고정되어 있다. 즉, CMD8에 해당된다. CMD8은 호스트에서 SD 카드로 커맨드

전송 후, R7의 응답이 있을 경우 CMD2, CMD3 명령을 SD 카드로 커맨드를 전송하게 된다. CMD8에 대한 리스폰스는 R7이며, 이는 OCR 레지스터 값을 필요로 한다. OCR 레지스터는 카드 동작에 관련된 동작 전압에 대한 프로필이 저장된다. 따라서 SD 펌웨어 소프트웨에서 OCR 레지스터 설정이 올바르지 않으면 카드는 정상적으로 동작되지 않는다.

2.3 SD카드슬레이브컨트롤러

SD 슬레이브 IP(Intellectual Property)는 SD 메모리 카드 인터페이스를 위한 블록과 ARM SRAM 인터페이스를 위한 블록으로 나누어진다. 표 2는 SD 슬레이브 IP 블록 다이어그램을 나타낸다.

SD 슬레이브 IP는 SD, user logic, static input 인터페이스로 나누어진다[6]. SD 인터페이스는 SD 호스트와 슬레이브 IP 사이의 인터페이스로 SD 클럭, 명령어, 데이터 라인이 사용된다.

user logic 인터페이스는 SD 슬레이브 IP가 SD 프로토콜을 해석하여 user logic에 데이터를 읽고, 쓰기 위한 인터페이스이다. static input은 기본적으로 SD 슬레이브 IP가 동작하기 위한 설정 값으로, 이 값을 설정하지 않으면 SD 슬레이브의 커맨드, 리스폰스가 정상적으로 동작하지 않는다. buffer는 SD 메모리 카드에 데이터를 읽거나 기록할 경우 데이터를 담을 기억 공간이 필요로 하다. 이러한 기능을 수행하며, 4킬로바이트의 데이터를 저장할 수 있다. register는 SD 슬레이브 컨트롤러에서 HDL 및 소프트웨어에 의해서 설정 하거나 읽을 수 있는 부분을 의미한다. 이 레지스터 제어를 통해서 SD 호스트에서 데이터를 읽거나 쓰기를 할 경우에 인터럽트 발생 설정할 수 있다. 또한 호스트에서 데이터를 전달할 경우 SD 슬레이브 IP에 의해서 register에 의해서 호스트의 데이터를 SD 슬레이브 IP가 제공하는 메모리 영역을 통해서 안전하게 데이터를 메모리 영역에 저장할 수 있는 역할을 제공한다. 위의 register가 정상적인 값이 설정 되지 않을 경우라면 SD 메모리 카드 시스템이 정상적으로 동작하지 않게 된다.

표 2. SD 슬레이브 IP 블록 다이어그램
Table 2. Block diagram of SD Slave IP

Name	Description
SD Slave IP	SD slave controller receives commands from the host through the SD interface. Most of the commands are processed locally by the controller without any help from the user logic.
static input	A number of read-only register data such as Manufacturer ID is provided to the SD Slave via static inputs which can be hardwired by the user to the proper value. Some registers are completely self-contained within the controller core and do not require static input.
buffer	The SD Slave Controller features a data buffer to store data for transfer between the SD DAT line and the user interface. Each user interface has its dedicated data buffer. This buffer is used for both read and write operations. The buffer can hold up to 1024 32-bit words, which is 4096 bytes.
register	All the SD registers are implemented within the SD Slave controller, except the SD Status Register. The optional Driver Stage Register(DSR) is also exists in the SD Save controller. Some of the register contents are device specific, such as manufacture ID, and is provided to the SD Slave controller by the user logic via static input.

표 3은 ARM_SET 레지스터를 나타내며, 이 레지스터는 SD 호스트로부터 메모리 읽기, 쓰기 요청을 나타낸다.

표 3. ARM_SET 레지스터
Table 3. ARM_SET register

Bit	Description
[15:1]	Reserved
[0]	setting the completion of transmission operations 0 : idle status 1 : operation completition

0번째 비트가 '1'로 설정 되면 SD 슬레이브 IP는 하나의 동작이 끝났음을 의미한다. 예를 들어 호스트에서 특정 번지의 데이터 읽기를 요청한 경우, SD 슬

레이브 IP에서는 필요로 하는 데이터를 요청 하게 되며 데이터는 전달된다. 슬레이브 IP가 필요로 하는 데이터를 모두 전달한 후 ARM_SET 레지스터의 0번째 비트의 값을 1로 설정하게 되면, SD 슬레이브 IP는 호스트에서 요청하는 다음 명령어를 처리할 수 있게 된다. 만일 위의 명령어 처리를 수행하지 않는다면 SD 메모리 카드는 정상 동작을 하지 못하게 된다.

표4는 인터럽트 레지스터를 나타내며, 이 레지스터는 SD 호스트에서 SD 슬레이브 IP에게 데이터 읽기/쓰기 요청이 있을 때 발생한다. 호스트에서 메모리 영역에 데이터 쓰기가 필요한 경우 SD 슬레이브 IP는 인터럽트를 발생 시키게 되며, 인터럽트 레지스터의 1:2 비트 값을 읽어 들여 SD 호스트가 요구하는 operation을 알 수 있다. 인터럽트 처리를 모두 끝낸 후, 이 레지스터의 0번째 비트 값을 '0'으로 설정하면 인터럽트가 해제 된다.

표 4. 인터럽트 레지스터
Table 4. Interrupt register

Bit	Description
[2]	memory read
[1]	memory writ
[0]	interrupt enable bit 0 : interrupt disable 1 : interrupt enable

표 5는 SD_Address 레지스터를 나타내며, 이 레지스터는 호스트에서 메모리 읽기, 쓰기를 수행할 메모리 위치를 나타낸다. 이 레지스터를 읽어 들여 호스트에서 원하는 메모리 위치에 데이터를 읽거나 쓸 수 있다. 표 6는 SD parameter 레지스터를 나타내며, 인터럽트 발생 시 위의 레지스터를 통해서 SD 호스트가 메모리를 읽을 것인지 쓸 것인지를 판단할 수 있다. 표 6의 [3:2] 비트의 normal access는 일반적인 메모리 읽기, 쓰기를 나타내며, special access는 SD 슬레이브 IP가 특별한 동작을 수행하기 위한 것으로 special access가 발생하면, ARM_SET 레지스터에 값을 '1'로 설정한다. [13:4] 비트는 SD 호스트에서 요청하는 데이트 크기를 나타낸다.

표 5. SD_Address 레지스터
Table 5. SD_Address register

Bit	Description
[47:0]	Byte address of the memory location to be accessed. 48 bits of address is provided to support the maximum size of memory card.

표 6. SD 파라메터 레지스터
Table 6. SD parameter register

Bit	Description
[15:14]	reserved
[13:4]	transmitted data size
[3:2]	special/normal access 01 : special access 10 : normal access
[1]	read/write operation 0 : read 1 : write
[0]	indicating whether it is memory space transmission or I/O 0 : memory transmission 1 : I/O transmission

III. SD 프로토콜 분석기 설계

3.1 시스템구성도

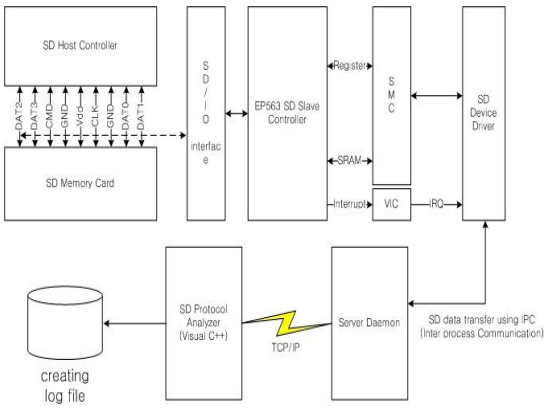

그림 2. SD 프로토콜 분석기 시스템 구조
Fig. 2 Architecture of the SD protocol analyzer system

그림 2는 SD 프로토콜 분석기 시스템 구조를 나타낸다. SD 프로토콜 분석기를 구현하기 위해서 SD 호스트 컨트롤러와 연결되는 SD 메모리 카드의 데이터 취득을 위한 SD Slave IP Controller FPGA 모듈이 필요하다. FPGA에서 전달해 주는 데이터를 처리하기 위하여 S3C2450 임베디드 프로세서를 이용하였다. 분석할 데이터를 전달받은 임베디드 모듈은 TCP/IP를 이용한 PC용 SD 분석기 소프트웨어에 데이터를 전달한다. 위의 소프트웨어는 전달 받은 데이터를 읽어들인 후, 들어온 데이터에 대해서 커맨드 및 리스폰스를 분석하여 로그 파일을 생성하게 된다.

3.2 SD Slave FIFO 디바이스 드라이버

그림 3은 SD 슬레이브 IP의 S3C2450 프로세서의 SMC 연결 구조를 나타낸다. SD 메모리 카드가 호스트 연결 후 들어오는 CMD 및 DAT0 -DAT3 라인의 값들을 임베디드 시스템에 전달해 주어야 한다. SD Slave IP의 FPGA 모듈은 S3C2450 프로세서의 SMC IP를 이용하여, 0x18000000 메모리와 0x10000000에 연결되어 있다. 0x18000000은 레지스터 영역으로 사용되며, 0x10000000은 데이터 영역으로 사용된다[8]. 그림의 SRAM_FSM 블록의 nRCS0,1 라인에 따라서 Dual-Port RAM 블록 및 Register File 블록이 선택된다. 프로토콜 분석기 특성상 SD 슬레이브 IP에서 SD 커맨드 및 데이터 라인의 값들을 프로세서에게 전달해 주어야 한다. 이러한 경우 Dual-Port RAM 블록에 CMD 및 데이터 라인의 값들을 저장한 후, 읽어야 할 데이터가 있음을 알리기 위해서 Register File블록을 이용하여 인터럽트를 S3C2450 프로세서에게 알린다.

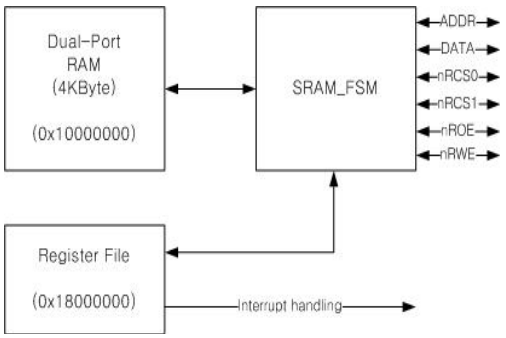

그림 3. SD 슬레이브 IP의 SMC 구조
Fig. 3 SMC structure of SD Slave IP

SD 슬레이브 FIFO 디바이스 드라이버는 Register File 영역의 인터럽트 레지스터를 읽어 들인 후, 인터럽트가 데이터 읽기인지 쓰기 인지를 분석한다. 데이터 쓰기인 경우, SD 호스트에서 SD 메모리카드의 특정 영역에 데이터를 기록하는 경우를 의미한다. 이런 경우 FPGA의 Dual-Port RAM 영역의 데이터를 읽어들여야 하기 때문에 0x10000000 메모리 영역의 데이터를 읽어 들인 후, DDR 메모리 영역에 저장한다. 모든 데이터를 읽어 들인 후, SD 슬레이브 IP에 데이터 읽기가 끝났음을 알리는 값을 Register File 영역에 값을 설정하여, SD 슬레이브 IP가 다음 동작 수행하도록 한다.

3.3 SD 프로토콜 분석 소프트웨어 설계

SD 메모리 카드의 CMD 및 DAT0부터 DAT3 까지의 데이터는 SD 슬레이브 FIFO 디바이스 드라이버로 전달된다. 위의 드라이버 FIFO 데이터는 TCP/IP를 통하여 윈도우의 SD 프로토콜 분석기 소프트웨어에 전달된다. 커맨드 라인을 이용하여 커맨드 토큰 및 리스폰스가 전달된다. SD 프로토콜 분석 소프트웨어를 통하여 커맨드 및 리스폰스 토큰을 분석한다. 커맨드는 아래 그림 4와 같은 토큰을 갖는다. 커맨드 토큰 프로토콜은 48비트의 크기를 갖는다. 커맨드 토큰은 시작비트 '0'에 의해서 시작되고 스톱 비트 '1'에 의해서 종료된다[5][9]. 커맨드 토큰은 MSB가 먼저 호스트로부터 전달되며, LSB가 가장 마지막에 전달된다. 처음 전달되는 비트 '0'은 커맨드 토큰의 시작 비트를 나타내며, 그 다음의 비트 '1'에 따라서 커맨드인지 리스폰스 인지를 나타낸다. 커맨드 토큰도 데이터 블록과 마찬가지로 에러 검출을 위한 패킷 정보에 CRC 비트 정보를 포함하고 있다[5][9]. 따라서 처음 들어오는 비트가 '0'이며, 그 다음 전달되는 비트가 '1'인 경우는 커맨드 토큰 형식을 나타낸다.

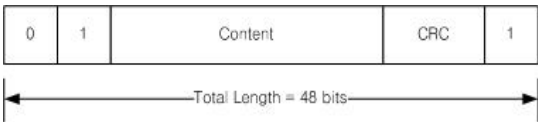

그림 4. 커맨드 토큰 형식
Fig. 4 Command token format

그림 4에서 Content는 커맨드 인덱스와 아규먼트로 구성된다. 커맨드 인덱스는 [45:40] 비트이며, 아규먼트는 [39:8] 비트를 나타낸다. 커맨드 인덱스의 값들을 BCD 코드로 표현하여, 실제 커맨드 타입을 얻을 수 있다. 그리고 [7:1]은 CRC7 코드를 나타내며, 마지막 비트는 end bit를 나타낸다. 리스폰스 토큰 형식은 커맨드 토큰 형식에서 리스폰스가 있는 커맨드에 대한 호스트에게 응답을 전달한다. 리스폰스는 두 가지 형식으로 나누어진다. 그림 5는 리스폰스 구조를 나타낸다. 리스폰스는 시작 비트로 '0'이 호스트에서 전달되며, 그 다음 비트가 '1'이 들어온다. 세 번째 비트에서 6비트가 어떤 커맨드에 대한 리스폰스인지를 나타내는 커맨드 인덱스를 나타낸다. 이 커맨드 인덱스에 따라서 리스폰스 형식이 결정된다. SD 메모리 카드에 대한 리스폰스 형식은 5가지로 나누어진다. 그림 6은 리스폰스 R1 토큰 형식을 나타낸다.

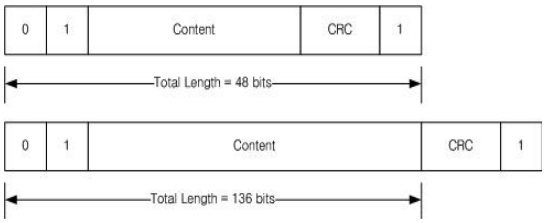

그림 5. 리스폰스 구조
Fig. 5 Response format

그림 6. 리스폰스 R1
Fig. 6 Response R1

리스폰스 R1은 다른 리스폰스 형식과 식별하는 부분을 card status를 이용한다. 가장 일반적인 리스폰스 형식으로서 48비트의 길이를 가진다. 아래 그림7은 리스폰스 R2를 나타낸다.

그림 7. 리스폰스 R2
Fig. 7 Response R2

R2 리스폰스는 코드 길이가 138비트이며, CSD 및 CID 레지스터에 이용된다. R2 리스폰스 형식의 구분 방법은 커맨드 인덱스를 읽어들인 후, 이 코드 값이 '111111'이면 R2 리스폰스 형식으로 판단한다. 만일 R2 리스폰스에 대한 command index인 '111111'라면 127 비트를 읽어 들인다. 위의 127비트는 CID 및 CSD 레지스터 정보를 나타낸다. 그림 8은 R3 리스폰스를 나타낸다.

그림 8. 리스폰스 R3
Fig. 8 Response R3

R3 리스폰스의 코드 길이는 48비트이며, OCR 레지스터의 항목을 전달하며, 이는 ACMD41를 나타낸다. R3 리스폰스 분석을 위해서는 [45:40] 비트가 '111111'이면서 [7:1] 비트가 '111111'인 경우 이를 리스폰스 R3로 판단한다. 그림 9는 리스폰스 R6를 나타낸다.

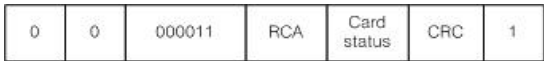

그림 9. 리스폰스 R6
Fig. 9 Response R6

R6 리스폰스의 판단 기준은 커맨드 인덱스인 [45:40] 비트의 값이 0x3이라면 R6 리스폰스를 나타내며, [39:24] 비트를 읽어 이를 RCA 레지스터 값을 식별하며, [23:8] 비트를 읽어 card status 값을 얻는다. 그림 10은 리스폰스 R7을 나타낸다.

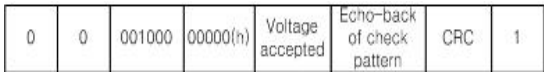

그림 10. 리스폰스 R7
Fig. 10 Response R7

리스폰스 R7의 코드 길이는 48비트이다. 커맨드 인덱스의 값이 0x8인 경우에 리스폰스 R7을 나타내며, R7을 통하여 카드가 제공하는 전압 범위값을 전달한다.

Ⅳ. 실험

SD 프로토콜 분석기 설계에 대한 시스템은 그림 11과 같다. SD 호스트와 연결되는 SD 인터페이스 부분에 실제 SD 메모리 카드가 삽입 되어 있다. 이때, 호스트에서 전달되는 데이터를 받기 위해서 작은 기판 회로가 구성되어 있으며, 이 기판을 통하여 FPGA 보드로 커맨드 및 데이터가 전달된다. 전달된 데이터는 FPGA의 SD 슬레이브 IP를 통하여 S3C2450 ARM 프로세서에 전달된다. 전달된 데이터는 TCP/IP를 이용하여 호스트 PC로 전달되어, 데이터 분석을 수행하는 구조로 되어 있다. 또한 FPGA 모듈과 ARM 프로세서와의 통신을 위해서 ARM 프로세서의 SMC IP 컨트롤러를 이용하였다.

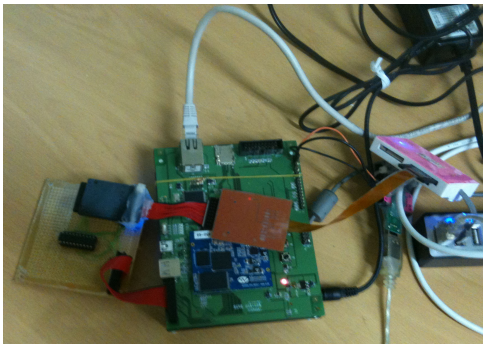

그림 11. SD 프로토콜 분석기 시스템
Fig. 11 System of SD protocol analyzer

그림 12에서 보는 것과 같이 FPGA에서 전달해준 데이터를 로그 파일 생성 결과를 보여준다. 1차적으로 생성된 데이터를 이용하여 커맨드 및 리스폰스 정보를 SD 프로토콜 형식에 맞추어 분리하는 과정을 수행한다.

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0
00000000	66	66	66	66	0A	66	66	66	66	0A	66	66	66	66	0A	66	ffff.ffff.ffff.f
00000010	66	66	66	0A	66	66	66	66	0A	66	66	66	66	0A	66	66	fff.ffff.ffff.ff
00000020	66	66	0A	66	66	66	66	0A	66	66	66	66	0A	66	66	66	ff.ffff.ffff.fff
00000030	65	0A	38	30	30	30	0A	30	30	30	30	0A	30	31	32	62	e.8000.0000.012b
00000040	0A	66	66	34	38	0A	30	30	30	30	0A	30	31	61	61	0A	.ff48.0000.01aa.
00000050	38	37	66	38	0A	34	30	30	30	0A	30	30	30	64	0A	35	87f8.4000.000d.5
00000060	30	39	66	0A	66	65	65	65	0A	30	30	30	30	0A	30	30	09f.feee.0000.00
00000070	30	30	0A	63	62	66	39	0A	62	38	30	30	0A	30	30	30	00.cbf9.b800.000
00000080	39	0A	30	34	31	66	0A	66	65	64	32	0A	38	30	37	38	9.041f.fed2.8078
00000090	0A	30	30	30	30	0A	32	62	66	33	0A	66	30	30	66	0A	.0000.2bf3.f00f.
000000A0	66	38	30	30	0A	30	66	66	66	0A	66	64	64	63	0A	30	f800.0fff.fddc.0
000000B0	30	30	30	0A	30	30	30	31	0A	39	37	66	39	0A	62	38	000.0001.97f9.b8
000000C0	30	30	0A	30	30	30	39	0A	30	34	31	66	0A	38	30	37	00.0009.041f.807
000000D0	38	0A	30	30	30	30	0A	32	62	66	33	0A	66	30	30	66	8.0000.2bf3.f00f

그림 12. 로그 파일 형식
Fig. 12 Log file format

그림 13은 SD 프로토콜 분석기로 들어오는 2진 데이터를 분석하여 SD 커맨드 및 리스폰스를 분석한 결과를 나타낸다. packet으로 시작하는 부분은 해당 프로토콜 패킷에 대한 비트 값을 나타내며, 그 다음 줄의 type으로 시작하는 부분은 비트 값들을 알아보기 쉽도록 16진수로 각 값들을 표현하도록 한 것이다. argument가 NULL인 경우는 type으로 시작하는 부분에 로그 데이터가 기록되어 있지 않고, argument에 특별한 데이터 값을 가지는 경우에는 해당 데이터 값들을 로그 파일에 기록하였다.

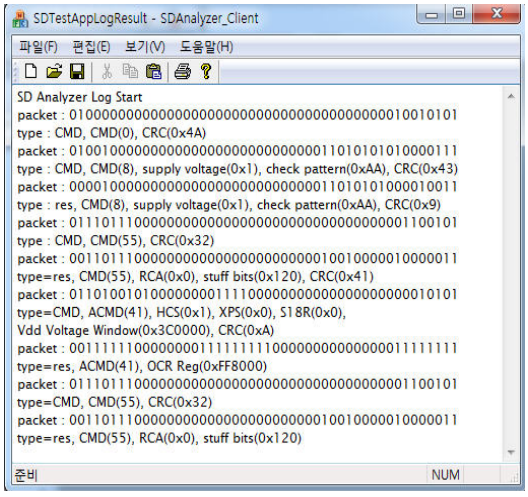

그림 13. SD 프로토콜 분석기 로그 데이터
Fig. 13 Log data of SD protocol analyzer

그림 14는 CMD0 대한 프로토콜 파형을 나타낸다. SD 메모리 카드를 호스트에 삽입할 경우, 발생하는 커맨드이다. 그림 12의 로그 파일 형식의 오른쪽을 보면, 카드가 연결하기 전에 연속적으로 '1'값이 들어오는 것을 확인할 수 있다. 이 상태에서 정상적인 CMD가 전송되면, 스타트 비트인 '0', 커맨드 인덱스, 아규먼트, CRC7 및 종료 비트인 '1'이 전달됨을 그림 14를 통하여 확인할 수 있다. CMD0의 경우는 리스폰스가 없는 형태 이므로 그림 13의 SD 프로토콜 분석기 로그 데이터에서 보는 것과 같이 CMD0에 대한 리스폰스 타입이 없는 것을 알 수 있다.

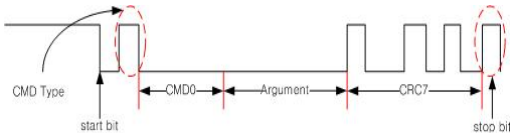

그림 14. CMD0에 대한 커맨드 프로토콜 파형
Fig. 14 CMD protocol wave for CMD0

그림 15는 CMD8의 커맨드 및 리스폰스 프로토콜에 대한 프로토콜 파형을 나타낸다. 커맨드의 경우 들어오는 두 번째 비트가 '1'이며, 리스폰스는 '0'임을 그림을 통하여 알 수 있다. 이 커맨드는 아규먼트를 통하여 supply voltage, check pattern 값들을 알 수 있다. 그리고 CMD8 다음에 CMD55가 오는 것을 확인할 수 있으며, 이는 그 다음에 오는 커맨드가 응용 커맨드임을 나타낸다. 따라서 그 다음에 오는 CMD41은 응용 커맨드임을 알 수 있다. 따라서 ACMD41를 의미한다. 이 로그 파일을 통하여 SD 메모리 카드가 삽입되어 초기화 루틴을 수행할 경우 CMD0 -> CMD8 -> ACMD41 순으로 수행됨을 알 수 있다. 또한 잘못된 커맨드 및 SD 슬레이브 IP에서 처리하지 못하는 커맨드가 존재하는 경우, SD 프로토콜 분석기 프로그램을 통하여 분석이 가능함을 확인 하였다.

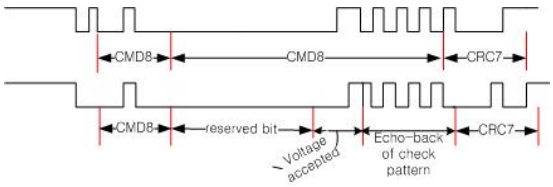

그림 15. CMD8에 대한 커맨드 프로토콜 파형
Fig. 15 Command protocol wave for CMD8

V. 결 론

SD 슬레이브 IP 개발 및 SD 메모리 카드 개발 시 IP 검증 및 기타 하드웨어 문제점을 검증하기 위해서 프로토콜 분석기가 필요하다. 본 논문에서는 SD 메모리 카드의 인터페이스를 분석하여 프로토콜의 일치 여부 및 전달 커맨드 및 리스폰스 등의 정보를 분석하였다. 이를 통하여 SD 호스트마다 전달해 주는 명령어 종류를 확인 할 수 있으며, 메모리 카드가 정상적으로 동작하지 않을 경우 문제점을 파악할 수 있음을 실험을 통하여 확인 하였다.

위의 시스템을 사용할 경우 FPGA 및 ASIC을 통해서 개발 중인 SD 슬레이브 IP가 하드웨어적으로 문제점이 없는지를 칩 생산 이전에 판단이 가능하며, 어떤 커맨드가 문제가 발생하는지를 손쉽게 판단이 가능하다. 대부분의 사용 판매중인 SD 프로토콜 분석기는 고가격 이지만 사용하는 기능은 대부분 SD 커맨드 및 리스폰스의 값들이 정상적으로 호스트와 통신되는 값인지를 판별하는 기능을 이용한다. 논문에서 개발된 프로토콜 분석기를 이용할 경우 저 비용으로 메모리 카드 오류 내용 분석이 가능하다.

참고 문헌

[1] Seon-Keun Lee, Woo-Yeol Jeong, "Design of the Entropy Processor using the Memory Stream Allocation for the Image Processing", The Journal of the Korea Institute of Electronic Communication Sciences, Vol. 7, No. 5, pp. 1017-1026, 2012.

[2] Hyun Hub, Jae-hak Lee, "A Study on Development of H8 MCU IDB(Integrated development board) for Embedded Education", The Journal of the Korea Institute of Electronic Communication Sciences, Vol. 4, No. 1, pp. 51-57, 2009.

[3] Kuk-se Kim, Gil-choon Kim, Joon Lee, "Embedded Linux System for Self-Control System of Car", The Journal of the Korea Institute of Electronic Communication Sciences, Vol. 2, No.1, pp. 62-66, 2007.

[4] C. T. Baik, Y. H. Lee,"A NAND Flash Controller for Mobile Devices", Proceeding of 5th KITT(Korean Institute of Information Technology) Summer Conference, pp. 667-670, gumi, south korea, Jun 2007.

[5] R. S. Kim, "A Study on Firmware Design for SD Memory Card Interface of CalmRISC," M. S. Theses, KyungHee University, Feb 2002.

[6] SD card protocol, http://wiki.seabright.co.nz/wiki/SdCardProtocol.html.

[7] EP563 SD Card Slave Controller, http://www.eurekatech.com/products/peripheral/ep563.htm.

[8] AP app part, "S3C2450 16/32-Bit RISC Micro-

processor User's manual," Samsung Electronics, Inc., 2009.
[9] SD Specification Part 1 Physical Layer Specification Ver 3.0, 2009.

저자 소개

문지훈(Ji-Hoon Moon)

2002년 동의대학교 컴퓨터공학과 졸업(공학사)
2004년 동의대학교 대학원 컴퓨터공학과 졸업(공학석사)
2009년 순천대학교 대학원 컴퓨터공학과 이학박사과정 수료
※ 관심분야 : 정보보안, 임베디드시스템

오재철(Jae-Chui Oh)

1978년 전북대학교 컴퓨터공학과 졸업(공학사)
1982년 전북대학교 대학원 컴퓨터공학과 졸업(공학석사)
1988년 전북대학교 대학원 컴퓨터공학과 졸업(공학박사)
1984년~1986년 기전대학교 전자계산학과 전임강사
1986년~현재 순천대학교 컴퓨터공학과 교수
※ 관심분야 : 임베디드시스템, USN, 네트워크 설계 및 분석