USB와 임베디드 시스템의 다중 통신 프로토콜 구현

김 성 수, 이 승 준, 문 철 홍
광주대학교 전자공학과
전화 : 062-670-2293 / 핸드폰 : 018-645-3525

Implementation of Multiplex Communication Protocol of USB and eMbedded System

Sung-Su Kim, Seung-jun Lee, Cheol-Hong Moon
Dept. of Electronics Engineering, Kwangju University
E-mail : hwaranghu@lycos.co.kr

Abstract

This paper use the USB Bulk Communication for interface of eMbedded system. But existing USB Bulk Communication have defects, it only connect point to point. And then this paper defines several command packet to improve such defects. So it improves Data Packet Layer of existing USB Bulk Communication.

And then this paper make up for it's defects and implements multiplex communication Protocol in order to connect many eMbedded systems.

I. 서론

컴퓨터의 발달과 이에 따른 주변 인터페이스 기술의 발달은 PC 보급을 확산시켰으며, 이러한 PC보급의 확산은 노트북과 같은 소형 PC의 보급에도 영향을 미쳤다. 또한 최근 유행어처럼 사용되는 PDA, 핸드헬 PC(HPC) 역시 지속적인 성장을 계속해 가고 있다. 이와 같이 소형 임베디드 시스템이 늘어가고, 점차 사용하는 데이터의 량이 증가하게 됨에 따라 이를 유지하고 관리하기 위해서는 HDD와 같은 저장매체를 이용해야 하고 이를 위한 인터페이스 구성이 필수적으로 이루어져야 한다. 이런 인터페이스의 구성을 위해 본 논문에서는 USB를 사용하였으며, 임베디드 시스템으로부터 전송되는 데이터의 관리를 위해 PC를 이용하였다. 이처럼 한대의 PC를 이용하여 여러 대의 임베디드 시스템을 다중 접속하게 함으로써 좀더 임베디드 시스템을 효율적으로 이용할 수 있게 하였다.

본 논문에서는 임베디드 시스템을 제어하는 CPU로 GMS320C7201을 사용하였으며, 이를 이용하여 외부로부터 데이터를 녹음하고 재생할 수 있는 녹음 장치를 구현하였다. 본 논문에서 사용하는 녹음장치의 모든 데이터는 압축하지 않은 원래의 데이터를 전송하므로 데이터의 전송 량이 많다. 이러한 대량의 데이터를 전송하고 처리하는 과정을 USB를 통해 구현 하였다. 또한 다수의 임베디드 시스템을 동시에 접속하여 처리하도록 USB다중 프로토콜을 구현 하였다. 이는 기존의 1대 1접속방법의 문제점을 해결하기 위한 것으로 동일한 드라이버를 사용하는 같은 USB디바이스의 다중 접속이 가능하도록 하였다.

II. 기존 USB 프로토콜

2.1 일반 USB 벌크 전송의 구조

본 논문에서 사용되는 USB 통신 방식은 벌크 전송방식을 채택하고 있으며, 본 논문에서 사용된

GMS320C7201이 벌크 전송방식만을 지원하고 있다. 기본적으로 USB 벌크 통신 방식은 그림 1과 같은 구조로 구성되어 있다.

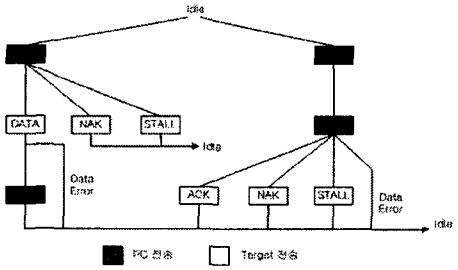


그림 1. 기존 USB 벌크 전송 방식

위의 기존 USB 벌크 전송 방식은 크게 3단계의 영역으로 분류할 수 있다. 먼저 IN과 OUT을 나타내는 Token Packet, Data Packet 그리고 ACK, NAK, STALL을 표시하는 Handshake Packet으로 분류된다. 이들 Packet의 형태는 그림 2에 각각 나타내었다.

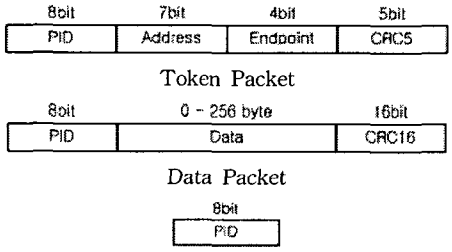


그림 2. USB 벌크전송의 주요 Packet

2.2 다중 통신상의 문제점

기존의 USB 벌크 전송 방식은 1대 1통신에서는 아무런 문제없이 동작 할 수 있다. 하지만 여러 대의 Target을 연결하여 사용하고자 할 때, 이는 각각의 드라이버를 가지고 있어야 하며, 제어할 수 있는 응용프로그램 역시 각각 제공 되어야 한다는 점과, 동일한 드라이버를 갖는 Target이 자신의 위치를 알리기 위한 방법 역시 매우 어려워진다는 문제가 있다.

III. 다중 통신 프로토콜의 설계 및 구현

3.1 전체 시스템의 구성

본 논문에서 사용된 전체 시스템의 구성은 그림 3과 같이 구성되어져 있다. 서론에서 언급 되었듯이 임베디드 시스템은 녹음 장치로 구성되어 있다.

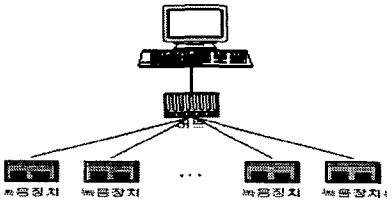


그림 3. 전체 시스템의 구성

3.2 명령 Packet의 정의

본 논문에서 사용된 프로토콜은 기본 USB 벌크 통신 프로토콜에서 사용된 data영역을 이용하여 좀더 세부적으로 구성되어졌다. 이는 송신과 수신을 효율적으로 운용하기 위해 정의된 기본 Packet들을 data영역에 추가시켰으며, 이들은 항상 수신과 송신 Packet을 한 쌍으로 구성하여 한 개의 명령 Packet을 구성하고 있다. 또한 명령 Packet을 구성하고 있는 기본 Packet은 32Byte를 기본으로 하고 있다.

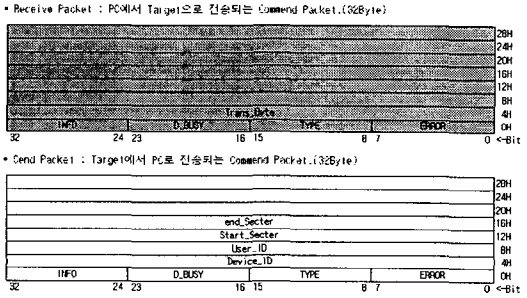
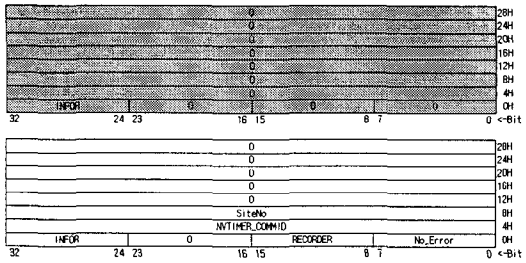


그림 4. 기본 Packet

그림 4에서 보여주는 Receive Packet은 그림 1에서 OUT Token Packet 이후에 Host에 의해서 전송되는 Data Packet에 해당한다. 또한 Send Packet은 IN Token Packet 이후에 Target에 의해 전송되는 Data Packet이다. 그림 4에 보여준 기본 Packet에 의해 만들어지는 Command Packet은 INFO Command Packet, INFO 예외Command Packet, Data Command Packet 그리고 X Command Packet으로 이루어져 있으며, 실제 데이터를 전송하는 데이터 영역으로 구분되어져 있다. 이들 각각의 Packet과 각 Packet에 사용된 메시지를 그림 5, 6, 7, 8에 차례로 나타내었다.

INFO Command Packet

: 이 Command Packet은 PC가 기본적으로 알아야 할 Target의 정보를 전송한다.

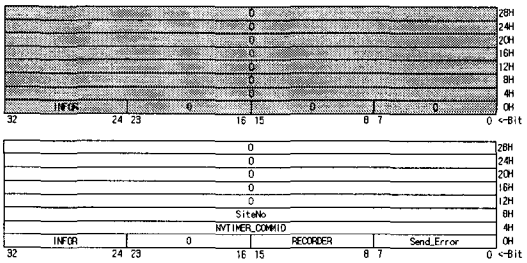


사용된 메시지	내 용
INFOR	Command Packet이 정보 Packet임을 알려 준다.
RECORDER	현재 장비가 RECORDER임을 알려 준다.
No_Error	에러가 없음을 알려 준다.
NVTIMER_COMMID	현재 장비의 고유 ID를 알려 준다.
SiteNo	현재 장비의 사용자 ID를 알려 준다.

그림 5. INFO Command Packet

INFO 예외Command Packet

: 이 Command Packet은 PC가 INFOR Command Packet을 Target으로 전송할 때 Target에서는 미리 보낼 데이터가 있는지 없는지를 PC에게 알려준다. 마약 PC로 전송할 데이터가 없는 경우 INFOR Command Packet의 예외가 동작하여 PC가 데이터를 받을 준비를 하지 않도록 해준다.

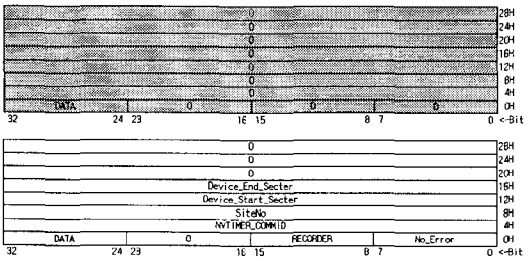


사용된 변수	내 용
INFOR	Command Packet이 정보 Packet임을 알려 준다.
RECORDER	현재 장비가 RECORDER임을 알려 준다.
Send_Error	보내기 위해 예약된 데이터가 없음을 알려 준다.
NVTIMER_COMMID	현재 장비의 고유 ID를 알려 준다.
SiteNo	현재 장비의 사용자 ID를 알려 준다.

그림 6. INFO 예외 Command Packet

DATA Command Packet

: 이 Command Packet은 INFO Command Packet에러가 발생하지 않은 경우 실제 데이터를 주고받기 전에 전송되는 Command Packet으로 INFO Command Packet이후에 전송되는 Command Packet이다.

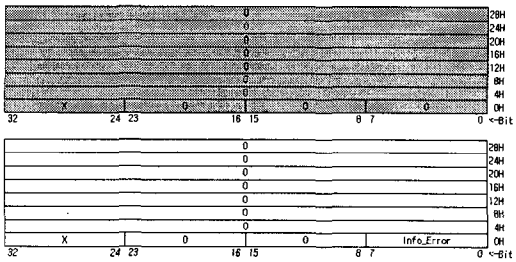


사용된 변수	내 용
DATA	Command Packet이 정보 Packet임을 알려 준다.
RECORDER	현재 장비가 RECORDER임을 알려 준다.
No_Error	에러가 없음을 알려 준다.
NVTIMER_COMMID	현재 장비의 고유 ID를 알려 준다.
SiteNo	현재 장비의 사용자 ID를 알려 준다.
Device_Start_Secter	현재 PC로 전송될 데이터 Track의 시작 Secter를 알려 준다.
Device_End_Secter	현재 PC로 전송할 데이터 Track의 마지막 Secter를 알려 준다.

그림 7. DATA Command Packet

X Command Packet

: 이 Command Packet은 PC가 Target으로 어떠한 정보를 전송하는지 알 수 없는 경우에 전송된다. 여기서 Receive Packet의 X는 INFOR또는 DATA이외의 값이 전송된 경우를 말하며, 이 경우는 어떠한 동작도 하지 않고 다음신호를 대기하도록 구현되어있다.



사용된 변수	내 용
Info_Error	현재 어떤 Command를 원하는지 알 수 없음을 PC로 알려 준다.

그림 8. X Command Packet

3.3 주요 세부 기능

(1) Target과 PC의 자동 연결

자동연결은 드라이버에 의해서 구현되며 각 장치가 연결 되고 끊어질 때마다 발생하는 메시지를 확인하여 해당 드라이버를 연결해 줌으로서 이루어진다. 이러한 과정을 그림 9에 나타내었다.

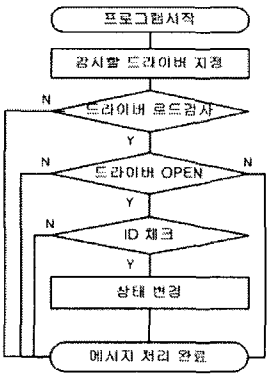


그림 9. 자동 연결 과정

(2) 폴링

자동 연결이 확인 되고 난 다음에는 연결된 장치의 정보를 얻고 Target의 연결이 유효한지를 판단하기 위한 과정이다. 이처럼 폴링의 과정에서 PC가 정보를 원하게 되면 Target은 현재 접속중인 장치의 ID와 사용자 ID 그리고 장치가 데이터를 받을 것인지, 혹은 전송할 것인지를 알려주게 된다. 이러한 정보 전달은 그림 5에 보여주는 것과 같은 INFO Command Packet을 사용한다. 그림 5에서는 에러가 발생하지 않은 경우만을 보여주고 있으며, 만일 에러가 발생한다면 그림 6의 INFO 예외 Command Packet이 발생하게 되어 Send_Error라는 메시지의 값에 의해 에러 처리 루틴을

수행하게 된다. 이러한 과정을 그림 10에 나타내었다.

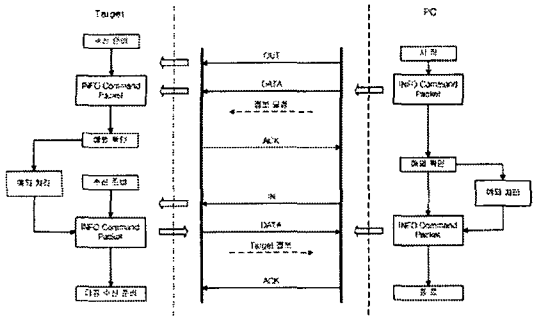


그림 10. 폴링 과정

(3) Data 전송

Data 전송을 하기 위해서는 우선 그림 7의 Data Command Packet이 전송되어야 한다. 이때에는 전송할 데이터의 길이, 파일명 등이 전송 된다. 이러한 Data Command Packet이 전송되고 나면 실제 유효한 데이터를 전송하는 부분은 기존의 USB 벌크 전송방식과 같다. 이는 32Kbyte단위로 데이터를 전송하며, 동기를 맞추기 위해 매 32Kbyte마다 32byte의 무효 데이터를 전송함으로서 데이터의 정렬이 흩어지는 것을 차단하였다. 이러한 Data 전송에 관한 신호의 흐름은 그림 11에 나타내었다.

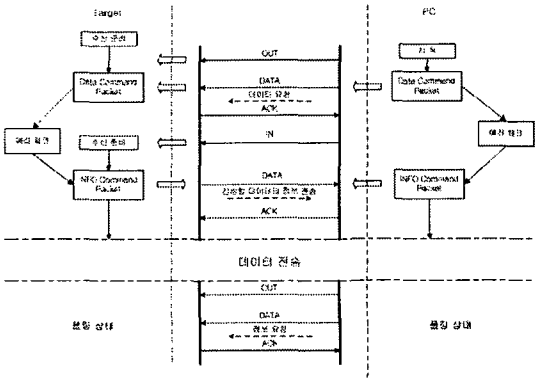


그림 11. Data 전송

이 이외에도 파일명, 데이터가 전송된 시간과 날짜, 데이터의 인증 확인 등을 위한 부가적인 기능들이 추가 되어 있으며, 이러한 기능은 데이터베이스와 연동하여 좀도 효율적으로 데이터를 관리하고, 데이터 전송에 신뢰성을 증대 시켰다.

Ⅳ. 결론

최근에는 고속의 인터페이스 기술이 발달하여 USB 이외의 여러 인터페이스 기술이 지원되고 있지만, USB가 다른 인터페이스 기술에 비하여 설치 및 개발이 용이하기 때문에 임베디드 시스템용 고속인터페이스의 기능을 충분히 수용할 수 있으리라 본다. 본 논문에서는 기존의 USB 벌크전송을 수정하여 다수의 임베디드 시스템을 연결하고 데이터 전송이 원활하게 이루어지도록 다중 프로토콜을 구현해 보았다. 이는 메인 PC 한 대를 이용하여 여러 대의 단말기(임베디드 시스템)를 연결할 수 있도록 하였다. 본 논문에서는 녹음 장치로서 저장된 데이터를 읽고 쓰는 단순한 형태의 데이터 전송만을 하지만, 이러한 USB 다중 프로토콜을 이용하면 보다 더 많은 응용으로 상용화에도 이용이 가능 하리라 추측해 본다.

참고문헌

[1] Intel Microsoft NEC, "Universal Serial Bus Specification", Intel Microsoft, Revision 1.1

[2] HYUNDAI, "GMS320C7201 32-bit RISC Microprocessor", HYUNDAI Micro Electronics, Ver 0.95

[3] John Hyde, "USB Design by Example A Practical Guide to Building I/O Devices", Wiley Computer Publishing