Embedded System을 위한 iRTOS™ 기반의 USB Mass Storage 설계 및 구현

안희중[0] 한성용 박희상 이철훈
충남대학교 컴퓨터공학과
(hjahn[0], syhan, hspark)@pplab.ce.cnu.ac.kr (chlee)@ce.cnu.ac.kr

Design and Implementation of USB Mass Storage Based on iRTOS™ for Embedded System

Hee-Jung Ahn[0], Sung-Yong Han, Hi-Sang Park, Cheol-Hun Lee
Dept. of Computer Engineering, Chungnam National Univ.

요 약

최근 PC 시장은 USB라는 새로운 인터페이스의 보급으로 USB를 이용한 장치들이 급속도로 증가하고 있다. 또한 USB를 이용하여 저장 장치를 접근할 수 있는 USB Mass Storage는 디지털 카메라나 MP3 Player 등에서 빠른 전송속도와 편리함 때문에 각광 받고 있다. 본 논문에서는 실시간 운영 체제인 iRTOS™에서 USB 인터페이스를 이용한 USB Mass Storage가 가능하도록 설계하고 이를 구현한 내용을 기술한다.

1. 서 론

저장 매체가 대용량화 되면서 빠른 전송 속도를 갖는 장치가 필요하게 되었다. 제한된 연결 장치 수나 사용의 편리성을 해결하기 위해서 USB가 등장하게 되었고, 최근 디지털 카메라와 MP3 Player등 휴대용 기기들에도 많이 응용되고 있다. 그러나 시스템이 복잡해질수록 이를 관리하고 제어하는 데에는 한계가 있기 때문에 일반적으로 실시간 운영체제 등의 임베디드 시스템(Embedded System)을 사용하여 안정성 및 용이한 프로그래머 인터페이스를 제공한다. 본 연구팀은 실시간 운영 체제 중에서 안정성 및 성능에서 인정 받은 iRTOS™를 연구대상으로 하였고 이를 위한 USB Mass Storage를 구현하였다.

iRTOS™는 USB Mass Storage 구현에 필요한 우선순위(Priority) 기반의 선점형(Preemption) 멀티태스킹(Multitasking)을 지원하고 세마포어(Semaphore), 메시지 큐(Message Queue), 메시지 메일박스(Message Mailbox)를 지원하여 태스크간 동기 및 통신(Synchronization & Communication)을 지원한다.

본 논문의 구성은 2장에서 iRTOS™의 전체적인 구성을, 3장에서는 USB Mass Storage 설계 및 구현을, 4장에서는 테스트 환경 및 결과를 기술한다. 마지막 5장에서는 결론 및 향후 연구과제를 기술한다.

2. iRTOS™

iRTOS™는 실시간 운영체제의 핵심이라 할 수 있는 우선순위 기반의 멀티태스킹 환경 및 태스크간 통신할 수 있는 ITC(InterTask Communication) 환경을 지원한다. 태스크(Task)란 독립적으로 실행되는 프로그램 각각을 지칭하며, 멀티태스킹 환경은 여러 개의 태스크가 동시에 수행될 수 있도록 하는 스케줄링 정책을 의미한다. iRTOS™는 효율적인 자원관리 및 태스크간 통신을 위해 세마포어, 메시지 큐, 메시지 메일박스를 제공한다[1].

2.1 태스크 스케줄링 정책

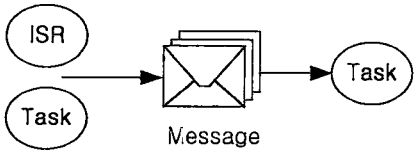
태스크는 0부터 63까지 64단계의 우선순위를 갖고 별도의 테이블로 관리되며, 정해진 시간 안에 가장 높은 우선순위를 갖는 태스크를 찾을 수 있다. 태스크가 서로 다른 우선순위를 갖는 경우 가장 높은 우선순위의 태스크가 CPU를 선점하며, 동일한 우선순위의 태스크들은 타임 슬라이스(Time Slice) 동안 차례로 수행되는 Round-Robin 정책을 채택함으로써 실시간 운영체제의 시간 결정성(determinism)을 보장해 준다.

2.2 ITC(InterTask Communication)

ITC는 세마포어, 메시지 큐, 메시지 메일박스를 가리킨다. 이들은 효율적인 자원관리 및 메시지 전달을 위해서 자원 할당 및 반납, 메시지 송·수신을 한다. 하지만 태스크가 자원이나 메시지를 획득하지 못할 경우 자원이나 메시지를 기다리고(wait)있는데, 기다리는 시간을 제어하여 적절한 time out값을 설정하면 태스크가 ready 상태로 변경되어 적절한 처리를 수행할 수 있도록 한다.

■ 세마포어(Semaphore)
세마포어는 한 자원에 대하여 배타적인 접근으로 공유자원에 대한 효과적인 관리뿐만 아니라 다른 태스크와의 동기화 및 이벤트 발생을 알려줄 때 사용한다.

■ 메시지 큐(Message Queue)
특정 태스크나 ISR(Interrupt Service Routine)이 다른 태스크에 여러 개의 메시지를 전달할 때 사용한다.



[그림 1] Message Queue

■ 메시지 메일박스(Message Mailbox)
메시지 큐의 특수한 경우로서 전달 가능한 메시지가 한 개 이어서 빨리 수행될 수 있다.

3. USB Mass Storage 설계 및 구현 내용

USB는 HID(Human Interface Device)를 제공하여 사용자에게 편리함을 제공하고, 주변장치의 연결이 편리하며, Hot Plug and Play기능을 지원하며, 부족한 IRQ를 해결하고, 연결 선을 단순화 하여 컴퓨터의 세트를 소형화 하며, 고속 직렬 통신을 지원한다. 이러한 USB 인터페이스를 기반으로 저장 매체(FD, HD등)와의 데이터를 송·수신하는 프로토콜이 USB Mass Storage이다.

3.1 USB Data Transfer

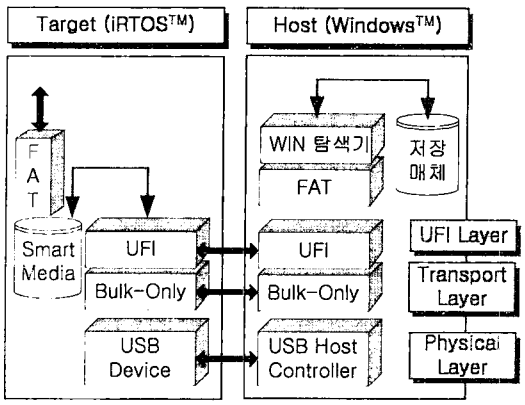
USB는 transaction format에 따라 4가지 종류가 있다. 먼저 Control Transfer는 작은 데이터나 control command를 전송하기 위해서 사용되며, Bulk Transfer는 데이터를 패킷(packet) 형태로 전송하기위해 사용되며, Interrupt Transfer는 주기적인 Request를 실행하기 위해서 사용된다. Isochronous Transfer는 일정한 속도로 일정량의 data를 전송하기를 원할 때 사용된다. 보통 오디오나 비디오 정보를 전송하는 경우 유용하다[2][3].

3.2 USB Mass Storage

USB Mass Storage를 위한 전송 프로토콜에는 데이터 전송에 사용되는 transaction format에 따라 구분할 수 있는데, Control·Bulk·Interrupt Transfer를 사용하는 CBI Transport와 Bulk Transfer만을 사용하는 Bulk-Only Transport가 있다. 본 논문은 데이터 전송 시 데이터 정보 외에는 제어할 필요가 없어 간단히 구현할 수 있는 Bulk-Only Transport를 사용하고 Command Block은 플로피 디스크 드라이버를 위한 Command Block인 UFI를 사용하여 타켓 보드인 CPAD3의 Smart Media를 제어하고 데이터를 송·수신 한다[4].

3.3 USB Mass Storage 구조

호스트의 파일관리 및 FAT는 Windows 탐색기가 전적으로 책임을 지고 UFI Command와 Bulk-Only transport 프로토콜을 사용하여 데이터를 전송한다. USB와 상관없이 Smart Media에 저장되는 파일은 타겟(Target) 시스템의 FAT 제어 모듈(Module)로서 제어되고 그 외에 USB를 통한 데이터의 Read/Write는 USB Mass Storage가 담당한다.



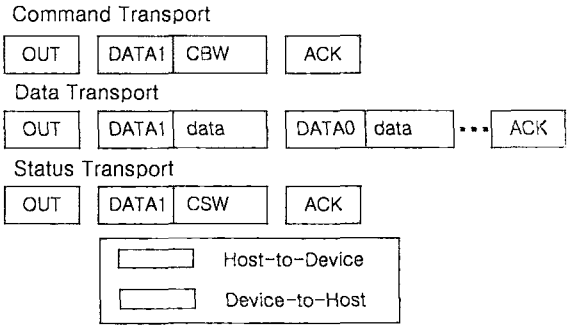
[그림 2] USB Mass Storage의 구조

3.4 FAT(File Allocation Table)

디지털 카메라나 MP3 Player등에 사용되는 대표적인 저장 장치로서 플래시 기억장치(Flash Memory)가 대표적이다. 본 연구에서는 MS-DOS™ 파일 시스템과의 호환을 위해 FAT16을 대상으로 한다.

3.4 데이터의 전달 순서 및 태스크 상태 변이

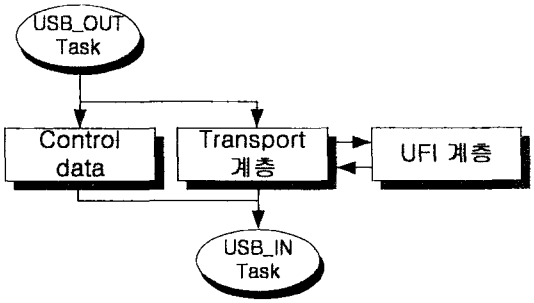
데이터는 호스트에서 발송하는 CBW(Command Block Wrapper), Command에 해당하는 data와 Command 의 성공·실패 여부를 회신 하는 CSW(Command Status Wrapper)의 3종류가 있다. 본 논문에서 구현한 태스크는 호스트로부터 데이터를 받는 USB_OUT 태스크와 호스트로 데이터를 전송하는 USB_IN 태스크가 있다. 태스크의 상태는 SETUP 및 RESET 후와 하나의 Command Block을 처리한 후의 **IDLE** 상태, CBW를 받아 해당 Command를 처리하는 **COMMAND** 상태와 Command를 처리하여 필요하다면 데이터를 전송하는 **DATA** 상태가 있다. 데이터의 전달 후 회신 되는 CSW를 받거나 보내게 되면 하나의 Command 처리가 완료되며 다시 IDLE 상태가 된다.
[그림 3]은 DATA 상태에서 호스트로 데이터를 전달하는 방향을 갖는 CBW를 보였다[5][6].



[그림 3] Command Block의 처리 과정

3.5 데이터의 송·수신

호스트로부터 데이터를 받는 USB_OUT 태스크는 메시지의 크기를 한번에 보낼 수 있는 데이터의 크기인 64Byte로 하는 메시지 메일박스를 생성한 후 메시지를 기다린다(pend), 메시지가 도착하면 ISR이 메시지를 USB_OUT 태스크의 메시지 메일박스에 발송하여(post) USB_OUT 태스크가 깨어나게 한다. 깨어난 태스크는 메시지가 도착한 endpoint에 따라서 control endpoint에 도착한 메시지이면 해당 제어 정보를 호스트로 데이터를 전송하는 USB_IN 태스크의 메시지 메일박스에 발송하며(post) USB Device의 초기화 및 리셋(Reset) 시에 발생한다. Bulk-out endpoint에 도착한 메시지는 Transport 계층에서 취합하여 하나의 완벽한 데이터로 조합하여 Command를 처리할 수 있는 UFI 계층으로 전달한다. UFI 계층에서는 Command를 처리 후 호스트로 보낼 데이터가 존재하면 Transport 계층을 통해서 USB_IN 태스크로 발송한다.

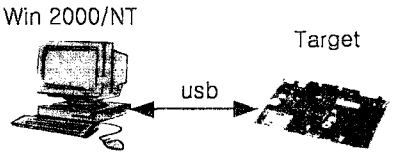


[그림 4] 데이터의 송·수신

호스트로 데이터를 전송하는 USB_IN 태스크는 메시지 메일박스를 생성한 후 메시지가 도착하기를 기다린다(pend). Transport 계층에서 메시지를 USB_IN 태스크로 발송하면 USB_IN 태스크가 깨어나며 bulk-in endpoint를 통해 메시지를 전송한다.

4. 테스트 환경 및 결과

본 논문의 구현은 삼성에서 제작한 CPAD3용 통합개발환경 CalmSHINE16으로 컴파일 하였다. 커널(36Kbyte)을 포함하지 않은 USB Mass Storage 이미지는 약 30Kbyte이다. 테스트 환경은 windows 2000 운영체제를 지원하는 PC를 호스트로 CalmSHINE16에서 컴파일한 이미지를 CPAD3 Emulation Board로 다운로드 하여 실행하였다.



[그림 5] 테스트 환경

5. 결론 및 향후 연구 과제

본 논문에서는 인터넷 정보 가전을 위한 선점형 멀티태스킹을 지원하는 실시간 운영 제제인 iRTOS™ 위에서 동작하는 USB Mass Storage를 설계하고 이를 구현한 내용을 설명하였다. 이로써 iRTOS™가 카메라와 MP3 Player 등에서 활용할 수 있게 되었다. 본 논문에서 구현된 USB Mass Storage는 USB 1.0기반으로 12Mpbs를 지원하고 있다. 최근 더욱 빠른 전송 속도를 요구하는 장치를 충족시키기 위해 전송속도가 480Mbps인 USB 2.0이 등장하게 되었는데 이를 접목하는 부분은 계속 연구되어야 할 것이다.

6. 참고 문헌

[1] http://www.inestech.com
[2] Universal Serial Bus Specification Reversion 1.1, 1998
[3] Jan Axelson, USB Complete Second Edition, 2001
[4] Universal Serial Bus Specification Mass Storage Class Control/Bulk/Interrupt(CBI) Transport, 1998
[5] Universal Serial Bus Specification Mass Storage Class Bulk-Only Transport Reversion 1.0, 1999
[6] Universal Serial Bus Specification Mass Storage Class UFI Command Specification, 1998