임베디드 시스템 장착을 위한 USB 장치의 무선화

정회원 유 진 호*, 조 일 연*, 종신회원 이 상 호**, 정회원 한 동 원*

Making USB Wireless For Attaching to the Embedded System

Jin Ho Yoo*, Il Yeon Cho* ***Regular Members***,
Sang Ho Lee** ***Lifelong Member***, Dong Won Han* ***Regular Member***

요 약

USB는 여타 다른 입출력 장치보다 더 자주 그리고 널리 사용되는 장치이다. PC시스템으로 보면 USB는 매우 성공적인 인터페이스 기술이다. PC시스템에서는 벌써 널리 사용되고 있을 뿐만 아니라 소비가전과 이동기기에서도 또한 널리 장착되고 있다. 게다가 USB는 이미 설치된 킬러응용프로그램도 매우 많다. 그렇지만 이런 입출력장치들은 유선으로 인해 사용자불편의 문제점을 가지고 있다. 많은 기기에 설치된 USB 장치를 무선화 시킬 수 있다면 유선화로 인한 스트레스를 줄임과 동시에 널리 사용되고 있는 USB의 사용이 더욱 더 편리해질 수 있을 것이다. 본 논문에서는 USB의 기존기능, 이식성, 멀티미디어 능력 등을 그대로 수용하면서 무선화할 수 있는 소프트웨어적인 방법을 연구한다. 본 논문은 이러한 USB의 무선화 구현에 관한 것이다. 이는 특히 기존 USB 호스트 기능에 내장된 USB 루트허브기능, 호스트 기능, ETD(Endpoint Transfer Descriptor), 데이터 메모리관리 등을 소프트웨어 모듈로 구성하여 유선일 때와 마찬가지 성능으로 지연없이 요구사항을 만족시켜야 한다. 무선화와 무선을 통해 그 기능을 어떻게 구현할 것인가에 대한 현안을 살펴보기로 한다. 이러한 구현은 기존의 USB의 편리한 사용과 시장력을 더 가속화할 것이다.

Key Words : linux, embedded, wearable, platform, system

ABSTRACT

It's for USB to be used more frequently than another I/O devices. The Universal Serial Bus is the most successful interface in PC history. It's already the de facto interconnect for PCs, and has proliferated into consumer electronics(CE) and mobile devices as well. USB has built on many killer applications, many CE devices, many interfaces. The tangle of wires among the number of devices in your home demands wireless technology. If our devices is unwired, it solves the tangle of wires. In this paper we want to use Legacy USB functionalities, portabilities, multimedia capabilities with wireless interconnection. This paper is related to a study of USB implementation without wires. This paper is related to make the hub function of USB cordless, so it will connect host with devices without wires. In case making USB wireless, it must support the above functionalities. Moreover, It needs the data structure and the resources for host functionalities, e.g. ETD(Endpoint Transfer Descriptor), data memory. This will benefit a convenient use of USB.

Ⅰ. 서 론

많은 사람들이 사용하는 범용컴퓨터에는 USB, IEEE1394, RS232, LAN 그리고 PS2 등의 다양한 입출력 인터페이스들이 있다. 이 중에서도 단연코 USB 입출력 장치는 여타 다른 입출력장치보다 더

* 한국전자통신연구원 디지털홈연구단 차세대PC연구그룹 ({yoojh, iycho, dwhan}@etri.re.kr)
** 충북대학교 전자계산학과 (shlee@chungbuk.ac.kr)
논문번호 : KICS2005-10-420, 접수일자 : 2005년 10월 18일

많이 사용되는 장치 중에 하나일 것이다[1]. 통계에 따르면 지금까지 USB 장치는 일억만개의 장비에 설치되어 사용되고 있다고 발표되었다. 게다가 2006년까지 3억 5천만개가 사용을 위해 선적된다는 발표가 있었다. 게다가 USB는 현재 속도에서 입출력 장치로 더욱 개선되고 사용범위를 넓혀가고 있다. 이미 많은 킬러응용프로그램들이 USB 장치의 성능을 전제로 하여 개발이 진행되고 있다. 사실상 USB는 범용컴퓨터의 가장 성공한 입출력장치라고 할 수 있다. 범용컴퓨터 뿐만 아니라 정보가전, 이동기기 등으로 그 사용을 확대하고 있다. USB는 엔터테인먼트 컴퓨터, MP3 플레이어, 디지털 캠코더, 세트톱박스, 디지털 스틸카메라, PDA, HDTV, DVD-RW, HDD, 개인용 비디오 플레이어, 프린터, 게임콘솔 등으로 그 연결범위를 확장하고 있다.

가정에서 사용하는 정보가전, 컴퓨터 등의 연결을 생각해보면 복잡한 유선을 생각할 수 있다. 사용기기가 많아짐에 따라 이러한 유선의 복잡성은 더 증대될 것이다. 이에 보편적으로 많이 사용되는 USB 장치의 복잡한 유선을 무선화한다면 사용에 훨씬 더 편리함을 느끼게 될 것이다. 무선화는 복잡한 유선을 해결할 수 있는 유일한 길이다. 본 논문에서는 USB 무선화의 중요한 역할을 하는 USB 허브의 무선화에 대해 생각하고, 또한 루트 허브기능을 구현하기 위한 리소스 고려사항들을 살펴본다. 다음으로 디바이스 드라이버 수준 상에서의 IP 패킷과 USB 패킷을 조정하는 컨버전스 계층을 어떻게 구현할 것인지 연구한다. 마지막으로 USB 패킷 전송의 무선화와 그에 따른 오류처리를 어떻게 할 것인지에 관하여 생각해 보기로 한다.

이에 유선 USB 장치를 무선화하여 기기에 사용되는 유선의 복잡함을 해결하고 USB 사용을 더 편리하게 하는 것이 본 연구의 목적이다.

Ⅱ. USB루트허브기능의 무선화

본 논문에서 다루는 USB의 무선화는 USB 루트허브기능의 무선화와 직접적인 연관이 있다. USB 허브는 특별한 USB 디바이스이다. USB 디바이스이면서 USB 호스트기능을 수행하여 USB 디바이스를 연결할 수 있게 한 특별한 형태의 디바이스이다. USB 허브는 호스트 포트로 들어오는 여러가지 정보를 실시간적인 폴링메카니즘으로 탐색한다. 호스트 포트에 연결될 때마다 감지하여 여러가지 시스템 관련 정보를 구성하여 USB 사용에 대한 허용을 관리한다. USB 허브를 무선화하는 작업에서는 USB 허브동작과 관련된 레지스터나 상태검사 이벤트를 포함한 USB의 기능들을 소프트웨어 모듈로 구성해야 한다. USB의 기본적인 토폴로지는 그림 1과 같다. 그림 1은 기존의 단순한 USB의 대략적인 구성이다[3]. 범용컴퓨터에 연결되는 단순한 모형을 표현한 것이다. 그림과 같이 호스트 콘트롤러는 루트허브의 기능을 수행하는데 루트허브에는 사실상 확장 가능한 127개의 USB 디바이스가 연결될 수 있다. 이렇게 연결된 USB 호스트와 디바이스는 주종관계로 서로의 통신관계를 맺는다. 그림에서 모니터 허브라 함은 모니터 기능을 가지는 USB 디바이스에 허브기능이 장착되어 다른 USB 디바이스로 연결될 수 있도록 구성이 된 것이다.

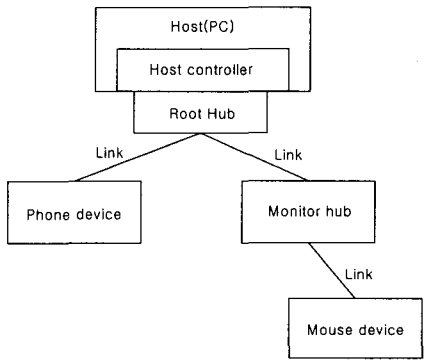

그림 1. USB의 간단한 물리적 구성

허브는 하나의 표준적 USB 디바이스이다. 허브는 단순히 다른 USB 디바이스를 연결하게 해주는 연결점이다. 이렇게 허브에 연결된 USB 디바이스들은 USB 호스트에 의해 주소가 정해지고 서로의 주소관계를 설정한 후 USB 표준에서 정의하는 4가지 모드의 전송패턴을 사용가능하게 만든다. 사용자 관점에서 보면 허브는 USB 디바이스를 다른 USB 디바이스에 연결할 수 있는 소켓을 제공한다. 그림 2는 허브가 그림 1에 기술된 디바이스를 어떻게 연결을 지원할 수 있는지의 예를 보여준다.

허브는 USB 디바이스의 동적인 연결과 끊어짐을 탐지하는 메커니즘을 제공하고, USB 장치의 핫플러깅에 대한 지원을 책임진다. 우선 허브를 무선화하기 위해서 USB 허브의 구조적인 요구사항을 이해해야 한다. USB 허브는 연결관련 동작, 전력관리, 디바이스의 연결과 끊김, 버스실패탐지와 회복, 3가지 속도에 대한 디바이스 요구사항을 지원해야 한다. USB를 무선화하기 위해서는 위에 열거한 큰

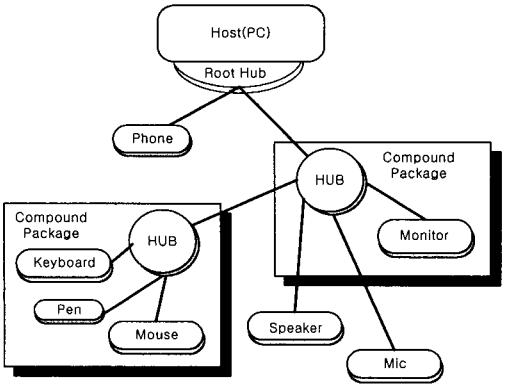

그림 2. HUB Fanout의 예

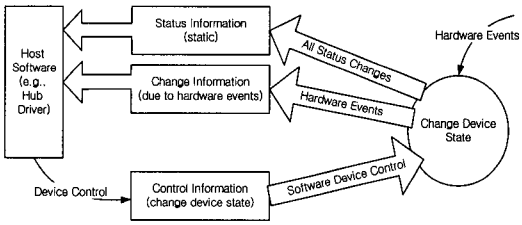

그림 3. 디바이스에 대한 상태, 상태변화, 제어 정보 관계

기능들을 구현해야 할 뿐만 아니라 자료구조와 호스트 기능을 위한 ETD, 데이터메모리 등의 자원을 관리, 제공해야 한다.

그림 3은 디바이스 이벤트관리와 상태정보를 어떻게 얻고 제어할 것인지를 보여준다. 하드웨어나 소프트웨어적인 이벤트가 발생할 때 허브 또는 포트 상태정보 비트가 세트된다[4]. 이와 같이 하드웨어/소프트웨어 상태정보는 해당 레지스터에 그 실시간적 상황이 나타나고 허브는 이것을 실시간적으로 감지해서 처리해 주어야 한다. 여기서는 이러한 일련의 상태와 제어를 소프트웨어 모듈을 통해 정보를 얻거나 제어하게 된다. 소프트웨어는 주기적으로 허브 레지스터나 이벤트 상태를 폴링하여 상태나 이벤트를 검사하고 그에 관한 조치를 취하게 된다.

Ⅲ. 루트허브를 위한 호스트영역 리소스 할당

USB 호스트 기능을 이뮬레이션하기 위해서는 USB 호스트영역 하드웨어 상에 마련된 리소스 구성을 시스템의 주프로세서에서 구현하여야 한다. USB 호스트 영역의 호스트 코어 레지스터세트, ETD메모리와 아비터, 데이터 메모리와 아비터 그리고 호스트콘트롤러 등의 리소스가 이뮬레이션을 위해 소프트웨어적인 자료구조형태로 지원이 되어야

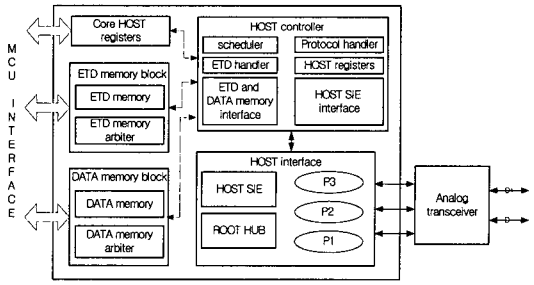

그림 4. USB 호스트 모듈블럭 다이어그램

한다. 그림 4는 우리가 소프트웨어적으로 이뮬레이션해야하는 하드웨어 블록다이어그램이다.

그림 4에서 볼 때 USB 전송모듈에 해당하는 호스트 인터페이스가 있다. 그리고 USB 호스트 콘트롤러는 USB 디바이스 연결할당과 같은 스케줄링 관리를 하며 그에 관한 상태제어 관련 정보를 레지스터로 관리한다[6]. USB 호스트 콘트롤러는 USB 호스트 인터페이스와 직렬 인터페이스 엔진을 통해 연결되며 이때 USB 호스트 콘트롤러는 ETD, 데이터 메모리를 사용하여 저장공간을 할당한다. 이러한 모듈적 구성을 주프로세서와 연결시키기 위해 USB 호스트 레지스터들이 구성된다. 본 논문에서는 그림 4와 같은 구성을 소프트웨어 모듈로 구성한다. 아날로그 트랜시버는 무선을 통해 전달된 정보로 구현되고 받아들여진 데이터는 호스트 콘트롤러 역할의 커널 프로세스에 의해서 USB 프로토콜 스택 형태로 관리된다. 호스트 콘트롤러 프로세스는 호스트 콘트롤러 하드웨어 구성에 해당하는 호스트 레지스터, 메모리 할당, 큐스케줄링 등의 처리를 위한 소프트웨어 자료구조를 제공한다. 이때 필요한 메모리 영역은 주메모리를 사용하여 ETD메모리와 데이터 메모리 두 종류로 할당된다. 호스트 드라이버는 실제 레지스터 세트 대신 소프트웨어로 구성된 자료구조에 접근하며 메모리 접근도 주메모리 상에서 이루어진다. 이와 같은 구현은 그림 5에서 보이는 조정계층(Adaption Layer)을 포함한 하부계층 스택 구현과 깊은 연관이 있다.

Ⅳ. 디바이스 드라이버 수준에서의 USB 영역과 IP 영역의 컨버전스

그림 5는 USB를 무선화하기 위해 필요한 스택 구성이다. USB의 전송량과 이벤트 처리가 가능한 무선 물리계층과 미디움접근계층이 정의가 되면 기존의 USB 디바이스드라이버 수준에서 조정계층을

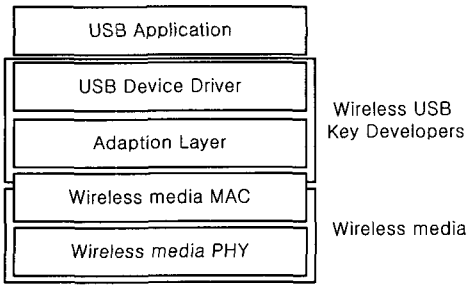

그림 5. 무선 플랫폼 상에 구성될 USB 영역

두어 이뮬레이션을 한다. 그림 5에서 USB 영역의 역할은 완전히 소프트웨어적으로 진행되며 USB 호스트와 USB 디바이스의 상호간의 요구와 통신량을 무선 구간에서 완벽하게 처리할 수 있어야 한다. USB의 전송량과 이벤트 처리에 알맞은 무선디바이스를 선택해야 한다. 이러한 무선디바이스를 살펴보면 USB의 휴먼 인터페이스 디바이스(HID) 등을 위해서는 블루투스나 FM, 전송용량을 요구하는 크기에 따라 802.11a/b/g, UWB 등을 선택할 수 있다. 그리고 이러한 무선디바이스는 USB의 기존의 4가지 전송방식 중 요구된 전송방식을 처리할 수 있는 것이면 된다. 보통은 무선디바이스가 IP 네트워크를 위해 마련된 것과 동시사용을 요구하므로 그림 6처럼 스택이 마련되고 네트워크 스택이나 USB 스택의 요구를 무선미디어영역 상위단계에서 모두 처리하게 된다. 기존의 유선 USB의 동작은 USB 호스트와 USB 디바이스의 소켓이 상호연결 되면, USB 호스트 상에서 이벤트가 발생하여 USB 디바이스의 존재를 인식하여 USB 디바이스에 주소를 할당하고 제어를 시작하게 되며 아이소크러너스, 벌크, 인터럽트, 콘트롤 모드 등의 4가지 모드 전송방식을 시작할 준비를 하게 된다.

무선구성에서도 이와 마찬가지로 USB 디바이스 기능모듈과 연결된 무선장치가 기존 무선채널로 이벤트를 보내어 USB 무선 호스트가 수신하게 되면 USB 호스트에 USB 디바이스가 붙은 것처럼 인식하여 4가지 전송모드를 시작할 준비를 하게 된다. 그림 6에 보여지는 바와 같이 무선채널로 입수된

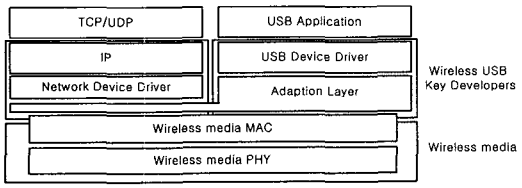

그림 6. IP와 동시 구성되는 USB 영역처리

이벤트 정보는 USB 디바이스의 출현으로 받아들여 전송모드를 준비하게 되며 무선채널로 입력되는 데이터는 IP 패킷인지 USB 패킷인지를 구분하여 서비스 접근을 위해 스택 위로 올려주게 된다. 그래서 IP 패킷은 IP 스택으로 USB 패킷은 USB 스택으로 서비스 접근이 일어난다. 이를 위해서 미디움접근 제어계층에서 패킷을 받아들여 분석하는 조정계층이 필요하게 된다. 이러한 일련의 동작들은 유선으로 구성할 때와 동일하게 이루어져 무선의 잇점을 경험할 수 있게 구성된다. 미디움접근 제어계층에서 받아들여진 패킷은 조정계층(adaption layer)에서 USB 패킷으로 인식되면 USB 패킷은 USB 스택에서 4가지 전송모드와 이벤트 등의 종류를 구분하여 처리된다. 이러한 USB의 무선 시스템 구성은 기존 장비의 사용 편의성을 유지하면서 선을 없앰으로서 편리와 효율을 제공할 수 있다.

V. 패킷전송의 무선화와 오류처리

USB는 구성과정에서 USB의 끝점(endpoint)를 결정하기 위해서 디바이스 기술자를 읽는다. 이때 대역이 허락된다면 USB는 클라이언트 구동자와 디바이스 간의 통신파이프를 설정하게 된다. 이렇게 생성된 통신파이프를 통해 클라이언트 구동자가 끝점(endpoint)에 대해 전송을 수행하려고 할 때 클라이언트 구동자는 전송하기 위해 USB 드라이버에 전송초기화를 수행하고 전송을 요청하게 된다. 클라이언트 구동자가 전송요청을 하면 USB 드라이버는 I/O요청패킷을 구성하고 이러한 요청패킷은 큰 블록의 데이터의 경우 여러 개의 트랜잭션으로 구성된다. I/O 요청패킷은 몇 개의 트랜잭션으로 구성된 후 이들 트랜잭션은 전송순서를 고려하여 호스트 드라이버의 프레임에 할당된다. 큰 블록의 데이터를 여러 개의 트랜잭션이라는 세그먼트로 나누어 구성하므로써 큰 블록의 데이터전송을 위해 전송될 다른데이타가 지연되는 것을 막을 수 있다. 같은 버스에 붙어있는 다른 USB 장비의 전송을 고려해 대역을 정할 수 있게 되는 것이다. 호스트 드라이버 단계에서 여러 클라이언트로부터 적당한 크기로 만들어진 트랜잭션을 시간대 별로 모아 전송 프레임을 구성한다. 이 프레임은 규정된 시간간격(1ms)으로 데이터를 전송하고 USB 드라이버는 자신의 트랜잭션이 어떤 프레임에 할당될 것인가를 대역을 고려하여 할당받게 된다. 이러한 버스 대역 용량에 대한 요구는 USB 디바이스가 버스에 연결될 때 구성과

정을 통해 디바이스 기술자를 참조하여 대역을 정하게 된다. 모든 USB 디바이스들이 매 프레임마다 전송요구를 갖는 것은 아니며 그것은 USB 디바이스의 특징에 의해 결정된다. 예를 들어 키보드 같은 경우 프레임 적재요구가 적을 수 있고 대용량 벌크 전송(bulk transfer)요구 같은 경우는 매 프레임마다 할당될 것을 요구한다. 이러한 일련의 작업은 그림 7에 나타나는 바와 같다.

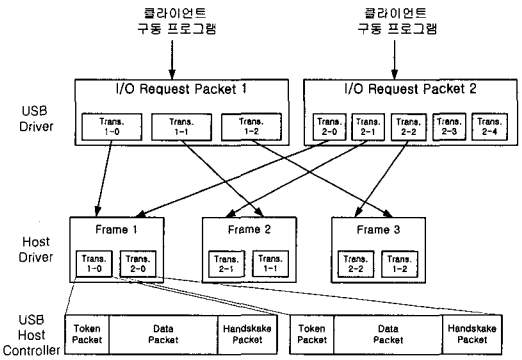

그림 7. 전송요구에 대한 패킷전송연결

우리는 그림 7에서 요구하는 것과 같이 호스트 드라이버 아랫단의 USB 호스트 제어기 부분을 소프트웨어 드라이버로 이뮬레이션한다. 그림 7에 구성된 패킷들은 형태에 따라 인터럽트전송, 벌크 전송, 아이소크러너스 전송, 제어전송 등의 4가지로 구분되게 된다. 각각의 전송특성에 맞게 낮은 속도 지원이나 오류보정, 전송률 보장, 최대 적재용량 등을 지원한다. 아이소크러너스를 제외한 전송은 패킷 오류검사, 잘못된 EOP, 버스 타임아웃(대답없음), 데이터 토글에러검사, Babble(버스 교착상태 등의 원인으로 디바이스가 트랜잭션을 마치지 못한 경우), 버스상의 수행부재(LOA, Loss Of Activity on bus) 등의 6가지 오류검사 메커니즘을 지원해야 한다. 결국 상위계층에서는 전송의 형태를 갖더라도 이는 트랜잭션으로 나누어지고 이 트랜잭션은 패킷의 형태로 통신을 하게 된다. 하나의 트랜잭션은 그림 8에서 보는 것처럼 3개의 단계로 구성되고 패킷 구성도 그림 8과 같다. 아래와 같은 패킷형태가 USB의 모든 패킷의 형태이므로 이러한 패킷에 대한 디바이스 처리과정을 구현하여 이뮬레이션을 수행하도록 한다. 또한 위와 같은 패킷의 형태로 4가지 전송형태를 지원하게 되며 실제 패킷의 형태를 무선 상의 MAC프레임에 삽입하여 USB 전송기능을 무선화한다. 무선 미디어의 형태에 따라 미디어

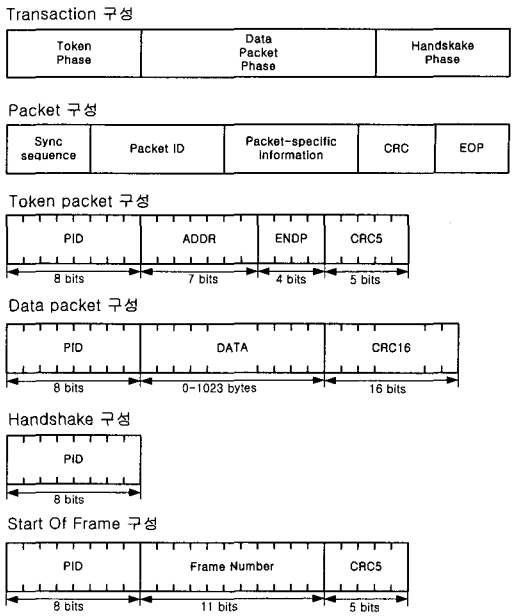

그림 8. Transaction과 패킷의 구성

접근제어(MAC)는 다른 형태를 가지며, 이때 정해지는 미디어접근제어(MAC)의 페이로드 부분에 해당 USB 패킷이 적재되고, 이는 수신 단에서 USB 패킷으로 구성된 후, USB 상위 인터페이스로 전송이 된다.

이러한 전송기능 이외에 위에서 언급한 오류처리 기능이나 대역요구에 따른 시간제약사항 등을 고려하여 구현하고 시험단계를 거쳐 그 처리에 대한 완벽함을 보장해야 한다.

Ⅵ. 실험을 위한 시스템 구성

무선 USB를 실험하기 위해 주프로세서로서 DragonBall MX인 MC9328MX21(i.MX21)를 채택한 자체개발 보드를 사용하였다. 이는 DragonBall시리즈로 ARM 9을 코어로 하며 이동기기 전용 프로세서이다. 개발보드의 운영체제로 리눅스 운영체제를 선택했으며, 버전은 2.4.20으로 mx2bsp를 패치했다. 개발환경은 호스트환경에서 리눅스를 설치하고 크로스컴파일을 위해 툴체인을 설치했다. 범용컴퓨터 환경에서 ARM 명령어 세트 구조에 맞는 코드를 생성하기 위함이다. 우리는 이러한 작업을 위해 인터넷에서 알맞은 세트를 구성하여 다운받아 설치 사용하였으며 기본적인 디바이스 드라이버도 또한 다운로드 받아 사용하였다. 새로운 하드웨어 구성에 따라 메모리맵을 정의하고, 무선모듈로는 블

루투스와 802.11a/b/g모듈을 장착하였다. 시스템의 시작과 기본적인 명령어세트를 위해 busybox를 사용하여 루트화일시스템을 구성했다. MTD(Memory Technology Device) 위에 JFFS2(Journaling Flash File System version 2) 파일시스템을 탑재했고 이를 위해 MTD 칩드라이버를 새로 프로그램했다. 우리는 512바이트단위 삭제블럭으로 정의된 NOR타입 플래쉬 메모리를 사용하였고, USB 스택 상에서 무선디바이스를 사용하여 실험한다. 본 논문의 무선 USB 기술은 원래 이동기기의 무선 입출력 장치사용을 위해 고안되었다. 이동기기와의 연결에 유선을 사용하는 것은 이동시에 무척이나 불편한 것이다. 이러한 무선 USB의 휴먼 인터페이스 장치로의 사용은 3차원 공간마우스 등의 새로운 입력장치의 연결을 자동으로 설정하고 인식즉시 사용하기 위한 방법이다. 시스템에 블루투스나 802.11a/b/g 등의 무선 디바이스를 장착한다면 입출력장치의 인터페이스에 무선을 사용하므로 해서 유선의 번거로움을 해결할 수 있다. 그리고 USB의 사용에 의해 특별한 연결설정 없이 인식 후 바로 사용할 수 있게 하자는 것이다. 또한 이러한 입력장치 뿐만 아니라 이동기기에 대용량 디스크를 무선 USB를 통해 연결하여 무선으로 디스크 싱크기능을 제공할 수 있다. 이동시 채집된 자료가 저장된 이동기기를 대용량 서버에 싱크하는 편리한 방법을 제공할 수 있다는 것이다. 실험을 위한 시스템은 그림 9에 보는 바와 같다. 그림 9에 보이는 개발보드는 USB 호스트 기능을 구현하기 위해 linux-2.4.20이 올라가 있으며 블루투스가 탑재되어 있고 USB를 통해서 무선랜이 설치 가능하다.

인터페이스 보드는 개발보드의 인터페이스 포트 연결을 위해 마련되었고 디버그 인터페이스 보드는 시스템 디버그를 위해 마련된 보드이다. USB 호스트 기능이 디바이스 드라이버 수준에서 수행되며 하부단의 인터페이스는 블루투스나 무선랜 등의 무

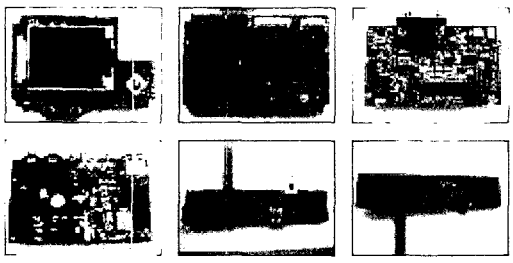

그림 9. 개발보드 앞,뒷면, 인터페이스 보드 앞,뒷면, 디버그 인터페이스보드 앞,뒷면

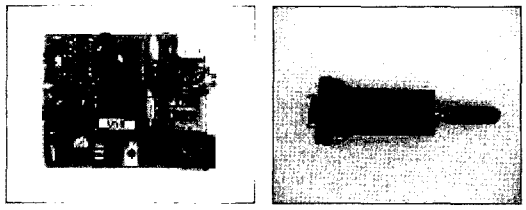

그림 10. USB 디바이스와 Serial-to-BT

선미디어로 연결된다. 저용량 휴먼인터페이스 장치 같은 경우에는 블루투스를 사용하고 더 많은 용량이 요구되는 경우에 무선랜을 사용할 수 있을 것이다. 이에 해당하는 상대 USB 디바이스의 예는 그림 10에서 보는 것과 같다. 그림 10의 USB 디바이스 장비는 무선 USB 소프트웨어 이뮬레이션 모듈 시험을 위한 것으로 USB 디바이스칩, 제어프로세서, 버튼이나 게이지 같은 입력장치 등으로 이루어져 있으며 USB포트와 직렬인터페이스를 갖는다. 버튼이나 게이지와 같은 입력장치를 통해 받아 들여진 데이터를 제어프로세서가 가공하여 USB 디바이스칩으로 보내고 USB 디바이스칩은 USB 포트를 통해 자신의 데이터를 유선으로 USB 디바이스 입장에서 전송하게 된다. 그런데 우리는 무선 구성을 위해 버튼이나 게이지 같은 입력장치로 입력된 데이터를 제어프로세서에서 USB 패킷형태로 가공하여 USB 포트대신 직렬인터페이스로 전송한다.

직렬인터페이스에 연결된 블루투스 모듈은 직렬인터페이스로 입력되는 정보를 블루투스를 통해서 전송하게 되며 USB 호스트단의 블루투스모듈은 이를 수신하여 USB 패킷데이터로 사용하게 된다. 이렇게 하여 USB 패킷형태로 된 정보는 무선미디어 간의 통신 형식에 맞게 구성되어 USB 호스트와 디바이스가 유선으로 연결된 것과 같이 동작하게 된다. 또한 실험장비를 통해 유선연결과 무선연결을 비교검증 할 수 있다.

Ⅶ. 결 론

지금까지 USB 기능을 무선화하기 위해 어느 부분에 접근해야 하는지 연구했다. 이는 분명하게 기존에 유선을 사용하는 USB 사용에 큰편의를 제공할 것이다. USB 유선영역을 무선화하기 위한 무선매체는 다양하게 선택할 수 있다. 예를 들어 UWB, 802.11a/b/g, 블루투스, FM 등이 무선매체가 될 수 있다. 대용량의 전송이 일어나는 모드를 위해서는 UWB등을 고려할 수 있고 기존에 소용량 입출력

장치 정도로 만들어진 것을 이뮬레이션 할 때는 블루투스, FM 등을 사용할 수 있다. 즉 USB의 대역 요구량에 의해 무선매체를 선택할 수 있다. 이러한 무선 구성은 USB의 더 많은 사용과 기존에 선으로 연결할 수 없는 환경에서의 사용까지로 확대될 것이다. 대용량 전송을 위한 UWB 무선 디바이스는 이미 테스트와 증명의 단계에 와 있으며 이러한 면에서 무선디바이스와 USB의 연결은 큰 설득력을 가진다. 결국은 이러한 기능들이 하드웨어 칩으로 구성되고 완벽하게 USB의 기능을 무선으로 구현될 것이다. 추후 이런 기능들은 대용량의 전송이 가능해지는 만큼 보안적인 잇슈로 식별, 인증, 권위 등의 단계를 가져야 할 것이다[5]. 이러한 확인기능을 밴드 내에서 소화할 것인지 밴드 밖에서 소화할 것인지도 결정해야 한다. USB 2.0의 부가기능인 On-The-Go의 기능도 내재시켜 무선 상에서 주종관계를 정의할 수 있게 하여 사용에 큰 편의를 제공하게 될 것이다[2].

참 고 문 헌

[1] USB Working Group, *Universal Serial Bus Specification Revision 2.0*, 2000.

[2] USB Implementers Forum, inc., *On-The-Go Supplement to the USB2.0 Specification Revision 1.0*, 2000.

[3] Don Anderson, *Universal Serial Bus System Architecture*, Mindshare, inc.

[4] John Garney, Ed Solari, Shelagh Callahan, Kosar Jaff, Brad Hosler, *USB Hardware and Software*, Annabooks, 1998.

[5] Agere, Hewlett-Packard, Intel, Microsoft, NEC, Philips, Samsung, *Wireless Universal Serial Bus Specification Revision 1.0*, 2005.

[6] Freescale, *i.MX21 application processor specification Revision 0.6 Chap. 44*, 2004.

유 진 호(Jin Ho Yoo) 정회원

1994년 2월 광운대학교 전자 계산학과 졸업

1996년 2월 서강대학교 전자 계산학과 공학석사

2005년 2월 충북대학교 전자 계산학과 박사과정 수료

1996년~1998년 (구)LG정보통신 주임연구원

1999년~현재 한국전자통신연구원 선임연구원

<관심분야> 임베디드 시스템, 운영체제

조 일 연(Il Yeon Cho) 정회원

1991년 2월 성균관대학교 산업공학과 졸업

1993년 2월 성균관대학교 산업공학과 석사

2004년 충남대학교 컴퓨터공학과 박사과정 수료

1995년~1996년 미국 OSF Research Institute 공동연구 파견근무

1993년~현재 한국전자통신연구원 선임연구원, 웨어러블 컴퓨팅 연구팀장

<관심분야> 임베디드 시스템, 웨어러블 컴퓨터

이 상 호(Sang Ho Lee) 종신회원

1976년 2월 숭실대학교 전자계산학과 졸업

1981년 2월 숭실대학교 시뮬레이션 공학석사

1989년 숭실대학교 공학박사

1981년~현재 충북대학교 교수

<관심분야> 프로토콜 엔지니어링, 네트워크 보안

한 동 원(Dong Won Han) 정회원

1982년 2월 숭실대학교 전자 공학과 졸업

1992년 2월 한남대학교 전자공학과 석사

1997년 2월 충남대학교 컴퓨터과학과 박사과정 수료

1982년 3월~현재 한국전자통신연구원 책임연구원, 차세대PC연구그룹장

<관심분야> 웨어러블 컴퓨팅, HCI, U-Healthcare