근거리 무선 제어를 위한 Bluetooth LAN Access Point 구현

이병웅* · 오원근* · 여 현*

Implementation of bluetooth LAN Access Point for the wireless Local access

Byeong-woong Lee* · Wongeun Oh* · Hyun Yoe*

요 약

본 연구는 블루투스 모듈이 장착된 임베디드 시스템에 리눅스를 포팅하고 리눅스 커널상에서 블루투스 프로토콜 스택을 작성해 LAN Access Point를 구현하는 것을 목적으로 한다.

임베디드 시스템을 위한 하드웨어를 구성하고 블루투스 모듈을 장착해서 리눅스 커널을 포팅한 후 Ethernet 디바이스 드라이버와 블루투스 프로토콜 스택을 작성해서 LAN Access Point를 구현한다. 블루투스 모듈이 장착된 PC를 클라이언트로 LAN Access Point를 통한 TCP/IP 통신의 가능성을 검증한 결과 블루투스 LAN Access Point를 통한 클라이언트의 Telnet, FTP, 웹브라우저 프로그램 등의 실행이 가능함을 확인하였다.

ABSTRACT

In this paper, LAN Access Point is realized by making out bluetooth protocol stack in Linux Kernel, porting Linux in an embeded system employed bluetooth module.

Hardware for a general-purpose embeded system is designed and Linux Kernel is ported after deploying Bluetooth Module. Also, Boot loader for system booting and management is composed, and Kernel is modified to meet with the system. Finally, Ethernet devise driver is made out, and bluetooth protocol Stack and Lan Access Point are realized. The possibility of TCP/IP communication through LAN Access Point to PC employed Bluetooth module, which is considered as client, is evaluated. From the evaluation results, we can see that the performance in Telnet, FTP, Web browsing through Bluetooth LAN Access Point is satisfactory.

I. 서 론

인터넷으로 통칭되는 통신 혁명이 낳은 결과물인 정보가전 분야에서 임베디드 리눅스의 적용분야는 다양하다. 본 연구에서는 범용 임베디드 시스템에 Linux 커널을 포팅하고, 근거리 무선통신 규격인 블루투스 프로토콜 스택을 임베디드 리눅스에 구현, 이를 이용해 bluetooth-LAN Access Point를 구현해 블루투스 장비들의 LAN Access Point를 통한 TCP/IP 통신의 가능성을 검증한다.

임베디드 시스템을 위한 기존의 여러 상용 운영체

*순천대학교 정보통신공학과

접수일자 : 2002. 5. 2

제가 존재하지만 비싼 로열티를 지불해야 하고, 새로운 기능 요구시 핵심 커널을 수정하기가 용이하지 않다. 그러나 임베디드 시스템에 리눅스를 사용할 경우 지금까지 입증된 리눅스 운영체제의 우수성을 그대로 사용할 수 있고, 필요시 소스를 수정해 사용할 수 있어 여러 이점이 있다. 또한 마찬가지로 블루투스 응용프로그램 작성시 필수적인 프로토콜 스택을 외국의 회사에서 비싼 로열티를 지불하고 구입해서 블루투스 응용프로그램을 개발하고 있는 실정이다. 따라서 본 논문에서는 공개된 운영체제인 리눅스를 임베디드 시스템에 포팅하고 블루투스 프로토콜 스택을 직접 작성해 상용 임베디드 운영체제와 블루투스 프로토콜 스택을 대체할 수 있는 가능성을 제시할 것이다.

본 논문의 구성은 다음과 같다. 2장에서는 근거리 무선통신 규격인 블루투스를 프로토콜 중심으로 간략히 기술하였다. 3장에서는 본 논문에서 구현한 임베디드 시스템에 리눅스를 포팅하는 기술과 블루투스 프로토콜 스택의 작동원리를 설명하였다. 그리고 이러한 기술을 바탕으로 LAN Access Point 구현과 성능을 검증한다. 마지막으로 4장에서는 결론 및 향후 과제를 제시한다.

II. Bluetooth 기술 분석

2.1 블루투스 기술 개요

블루투스는 근거리 무선 인터페이스를 통하여 음성 및 데이터의 전송 서비스를 지원하는 통신 프로토콜이다. 블루투스 시스템은 전세계적으로 동작하기 위해서 ISM(Industrial scientific medical)인 2.4GHz대의 주파수 대역을 사용하고, 블루투스 채널은 FH/TDD(frequency-hop /time-division-duplex)방법을 사용한다[5]. 블루투스 시스템은 음성과 데이터를 혼합하는 멀티미디어 응용을 지원하기 위하여 SCO(synchronous connection-oriented)와 ACL(asynchronous connectionless)의 2가지 형태의 링크를 정의한다.

2.2 블루투스 무선통신의 기술적 특징

블루투스는 ISM(Industrial Scientific Medical)밴드로 일컬어지는 2.4GHz대역을 사용하여 운용한다. 주파수는 1MHz 간격으로 79채널을 사용한다.

송신전력에는 세 가지의 클래스가 있으며 출력이 큰 것부터 순서대로 클래스1, 클래스2, 클래스 3으로 나누어져 있다.

표 2.1 블루투스 전력 클래스

클래스	최대출력(Pmax)	통신 가능거리
1	100mW(20dBm)	약 100m
2	2.5mW(4dBm)	약 10m
3	1mW(0dBm)	

블루투스는 다른 기기와의 간섭을 없애기 위해 주파수 호핑 방식을 채용하고 있다. 1슬롯마다 무작위로 주파수를 전환하여 고정 송신주파수에 의한 간섭을 방지하는 동작을 한다. 블루투스의 경우는 1슬롯이 625㎲이므로 1600회/초의 주파수 전환을 하게 된다. 그림 3.1과 같이 송신과 수신을 교대로 행하는 TDD(Time Division Duplex)라는 방식으로 양방향 통신을 한다. 또한, 하나의 패킷은 복수의 슬롯에 걸치는 것도 가능하다.

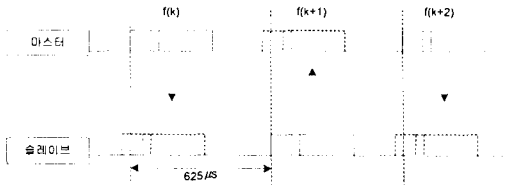

그림 2.1 TTD와 타이밍

2.3 블루투스 프로토콜 스택

블루투스의 프로토콜은 그림 2.2와 같은 구조로 되어 있다.

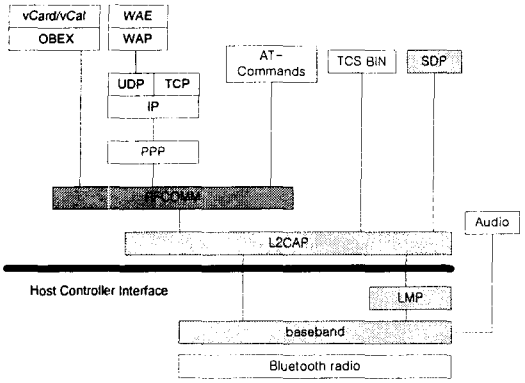

그림 2.2 블루투스 프로토콜 스택

블루투스 프로토콜 스택은 사용용도에 따라 4가지의 계층으로 다시 나눌 수 있다. 각각의 계층에 포함되는 프로토콜은 표 2.2와 같이 나눌 수 있다.

표 2.2 계층별 프로토콜

프로토콜 계층	프로토콜
Bluetooth 코어 프로토콜	Baseband, LMP, L2CAP, SDP
Cable Replacement 프로토콜	RFCOMM
Telephony Contorl 프로토콜	TCS BIN, AT-Commands
Adopted 프로토콜	PPP, UDP/TCP/IP, OBEX, WAP, vCard, vCal

III. LAN Access Point 구현

3.1 하드웨어 구성

임베디드 시스템을 설계하기 위해서는 먼저 용도에 적합한 H/W를 설계하고, 시스템을 어떠한 용도로 사용할 것인가를 결정한 후에 적합한 CPU를 선정하고, 적합한 메모리의 용량을 결정한 후 필요한 주변장치를 디자인하여야 한다.[4] 그리고, H/W 설계의 전체적인 윤곽이 잡히면 S/W 설계작업에 들어간다. 먼저 적합한 OS를 선정하여야 하는 데 본 연구에서는 리눅스를 사용하였다. 그리고 목적하는 응용프로그램을 개발하였다.

본 연구에서 사용한 하드웨어의 구성은 그림 3.1과 같다.

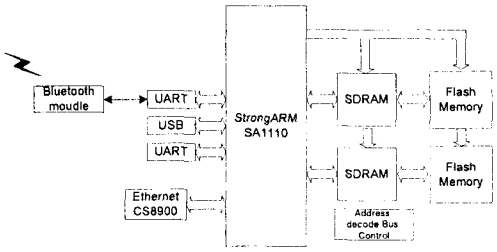

그림 3.1 하드웨어 구성

기본적인 보드의 구성은 CPU, RAM, Flash Memory로 구성되어 있고 UART를 통해서 블루투스 하드웨어 모듈과 연결되어있다. 그리고, LAN Access를 위해 CS8900 Ethernet Chip을 사용하였다. 개발시 모니터링과 시리얼 다운로딩이 가능한 Serial Port가 있고 네트워크에 연동될 수 있도록 CS8900 Ethernet Controller를 사용하였다.

3.2 부트로더 작성

일반적으로 부트로더라 하면 x86계열 리눅스에서는 LILO를 사용한다. LILO란 Linux Loader로써 Windows나 리눅스등 다른 OS를 선택적으로 부팅할 수 있도록 하는 기능을 제공한다. LILO는 하드디스크의 MBR(Master Boot Record)에서 동작되는 프로그램으로 OS가 실행할 수 있도록 점프하는 기능을 수행한다. 그러나 임베디드 시스템에는 하드디스크가 없기 때문에 JTAG를 이용해서 플래쉬 0블록에 부트로더를 써넣어 여러 가지 다양한 기능들을 수행한다[3].

본 연구에서 구현한 부트로더의 흐름도는 그림 3.2와 같다. 시스템에 전원을 공급하면 제일 먼저 실행되는 파일이 start.S이다. 이 파일은 ARM 어셈블리로 작성되었으며 부트로더에서 사용할 각종 장치들의 초기화를 수행한다. 보드 구동시 메모리의 0번지로 점프하고 0번지에는 vector table이 존재한다. hardware reset시 이 vector table의 setup이 가리키

는 주소를 찾아서 실행되는데, 이 vector table을 만들어 준다. 그 다음 CPU의 clock을 설정하고 SDRAM, Flash를 설정하고 C언어로 작성된 부트 로더의 나머지 루틴에서 사용하기 위한 stack point를 초기화 하고 부트로더의 메인 루틴으로 분기한다. 실제적인 메인루틴으로 점프하기 전에 부트로더가 있는 영역을 sdram영역으로 복사하고 그곳으로 점프하여 sdram에서 부트로더가 돌 수 있도록 하는데, 부트로더가 flash에서 동작하면 CPU가 계속 flash를 참조하여, flash에 쓰기가 불가능하므로 sdram으로 자신을 복사한 후 그곳으로 점프한다.

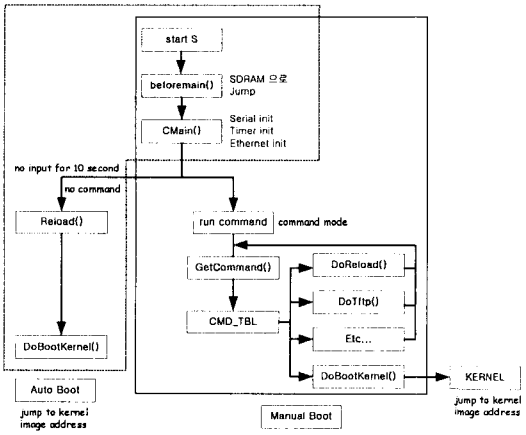

그림 3.2 부트로더의 흐름도

부트로더의 메인함수는 시리얼과 타이머를 초기화하고 사용자의 명령을 기다려 명령에 맞는 함수를 실행한다. 명령을 기다리다 10초동안 아무런 입력이 없으면 플래쉬에 있는 커널을 SDRAM으로 로딩해 커널을 실행시키고 입력된 명령이 있으면 명령을 분석하여 해당 함수가 호출된다. 명령에따른 함수 실행은 CMD_TBL에 등록된 함수를 호출하여 실행한다.

3.3 리눅스 커널 포팅

임베디드 시스템에 리눅스 커널을 포팅하기 위해 CPU와 보드에 맞게 커널을 수정해야 한다. 먼저 임베디드 시스템을 위해 커널을 재구성하기 위해서는 기본적인 CPU, 메모리, 인터럽트에 관련된 부분을 수정하고 그 외에 필요한 장치의 디바이스 드라이버를 작성해야한다.

리눅스 커널을 시스템에 적합하게 수정하고 작성하려면 리눅스 커널의 기본 디렉토리 구조를 알고 있어야 한다. 그림 3.3은 리눅스 커널 디렉토리의 기본적인 구조이다[1]. 많은 디렉토리가 있는데 하드웨어 독립적인 부분들, 즉 특정 하드웨어와 관계없이 작동하는 프로세서 관리부분, IPC 관련부분, 네트워킹 부분들은 수정하지 않아도 되고, 해당 하드웨어마다 다르게 커널을 작성해야 하는 부분들은 하드웨어에 맞게 다시 작성해야 한다. 이러한 부분들이 arch 디렉토리 아래에 모두 모여있다. 따라서, arch 디렉토리 아래의 CPU, 인터럽트, 타이머, 메모리 초기화 등의 부분을 작성하면 된다. 그리고 drivers 디렉토리 아래에는 여러 하드웨어 디바이스 드라이버 들의 소스가 들어있는데, 해당 보드의 디바이스들의 드라이버를 작성하여 이곳에 넣으면 된다.

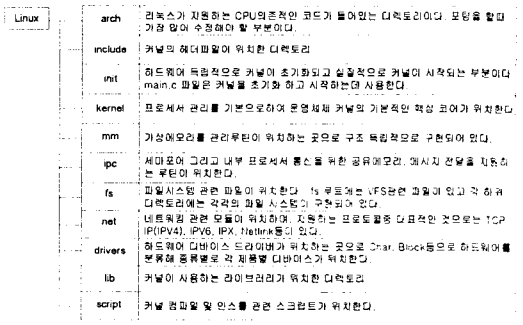

그림 3.3 리눅스 커널 디렉토리 구조

3.4 블루투스 프로토콜 스택 구현

응용프로그램을 통해서 블루투스를 실제로 사용하기 위해서는 RF(Radio Frequency)와 baseband 신호를 처리하는 하드웨어를 제어하고, 이를 응용프로그램과 연결시켜주는 소프트웨어를 필요로한다. 본 연구에서는 임베디드 리눅스의 커널상에 블루투스 프로토콜 스택을 구현해 일반적인 디바이스와 같이 블루투스 하드웨어를 제어하게 하였다.

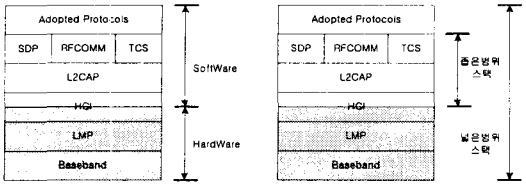

그림 3.4 블루투스 프로토콜 스택

그림 3.4의 프로토콜 스택중 Baseband와 LMP 그리고 HCI의 일부는 블루투스 하드웨어 모듈에 펌웨어로 구현이 된다. 본 논문에서는 임베디드 시스템에 블루투스 하드웨어 모듈을 장착하고 리눅스를 포팅한 후 LAN Access Point를 구현하는 것이 목적이므로 하드웨어 모듈에 포함된 하위 계층의 프로토콜은 논외로 한다.

그림 3.4를 운영체제가 포함된 Host와 블루투스 하드웨어로 구성된 Host Controller로 구분해 프로토콜 스택을 세분화 하면 그림 3.5와 같다.

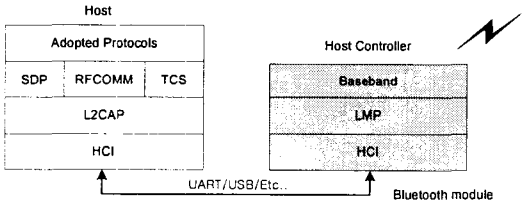

그림 3.5 Host와 Host Controller의 프로토콜 스택

그림에서 Host는 리눅스 운영체제가 포팅될 임베디드 시스템이고 Host Controller는 블루투스 하드웨어 모듈이다.

HCI를 통해 Host와 Host Controller가 연결이 되는데, HCI는 엄밀히 말하면 프로토콜이 아니고, 상위 계층과 하드웨어간에 일관된 연결 방법을 제공하는 인터페이스라고 볼 수 있다. HCI는 UART, USB, PCM 등 다양한 연결 방법을 통해 패킷을 전송하고 Host Controller를 제어하는 역할을 한다. 본 연구에서 리눅스 커널상에 구현한 프로토콜 스택의 구조는 그림 3.6과 같다.

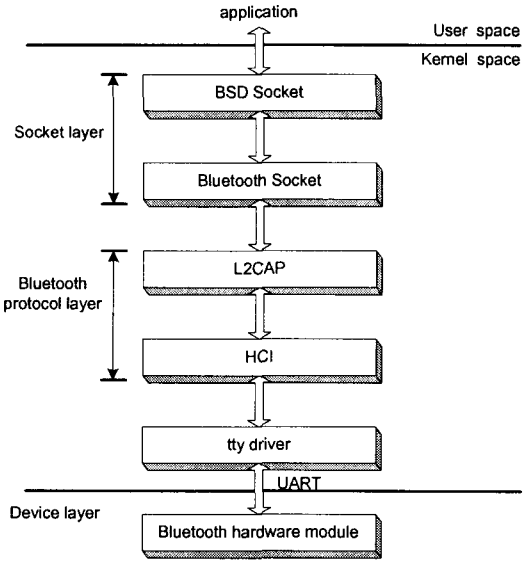

그림 3.6 블루투스 프로토콜 스택 구현의 전체구조

Host의 HCI 드라이버는 데이터와 명령어들을 블루투스 하드웨어의 HCI 펌웨어와 주고 받으면서 통신할 수 있는 루틴을 구현해야 한다. Host와 Host Controller사이의 인터페이스 방법에는 UART, USB 등 여러 가지 방법이 있으나, 본 논문에서는 구현에 간단한 UART를 사용하기 때문에 tty driver를 이용해서 패킷을 전송한다. tty ldisc(line discipline)를 이용해 UART와 블루투스 프로토콜 스택 사이의 데이터 전송을 한다. tty line discipline은 tty line의 동작 모드라고 할 수 있다. 하나의 serial line은 일반적인 터미널로도 쓸 수 있고, PPP나 SLIP 목적으로 사용할 수도 있고, 여기서처럼 tty driver지만 기반은 블루투스일 수도 있다. 시스템에는 여러 가지 discipline이 존재하고, 해당 터미널이 사용되는 용도에 따라서 동적으로 다른 discipline을 적용하게 할 수 있다. 그림 3.7의 HCI 수신 흐름도를 보면 먼저 serial line에 데이터가 도착하면 tty line discipline은 HCI의 수신 함수인 hci_tty_receive()함수를 호출한다.

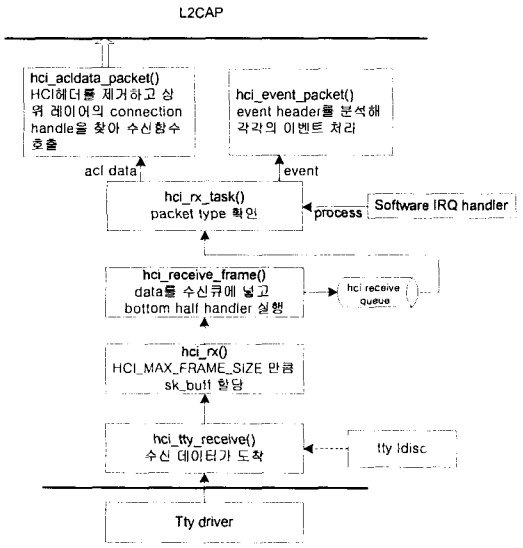

그림 3.7 HCI 드라이버의 수신 흐름도

hci_rx()함수에서는 소켓 버퍼인 sk_buff를 할당한 다음 수신된 데이터를 복사한다. 그리고, 수신된 데이터의 첫 8bit를 검사해서 패킷의 타입을 알아낸다. 패킷의 타입에 따라 헤더를 읽어서 헤더에 담긴 데이터 길이만큼 데이터를 수신해 소켓버퍼의 끝에 데이터를 추가한다.

3.5 LAN Access Point 구현

지금까지 작성한 블루투스 프로토콜 스택을 이용해 LAN Access Point를 구현한다.

3.5.1 LAN Access Profile

RFCOMM위에서 PPP를 사용한 LAN 액세스를 정의하는 프로파일이다.

프로토콜 스택을 보면 그림 3.8과 같다.

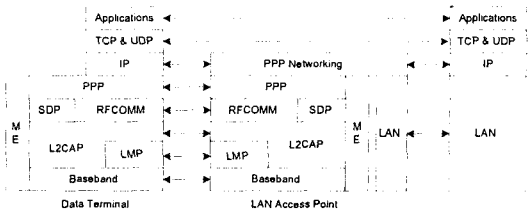

그림 3.8 Lan Access Point Protocol stack

PPP는 IETF Point-to-Point Protocol이다[11].

PPP-Networking은 IP 패킷을 PPP계층과 주고받아서 LAN에 위치시키고, Baseband, LMP, L2CAP은 블루투스 프로토콜이며, RFCOMM은 GSM TS 07.10을 블루투스에 적용한 것이다.

그림 3.9에 본 논문에서 구현한 LAN Access Point의 전체적인 구조를 보여주고 있다. LAN Access Point로 사용될 리눅스가 포팅된 임베디드 시스템은 인터넷과 연결이 되어있고 클라이언트로 사용될 PC는 인터넷과 연결되어 있지 않다.

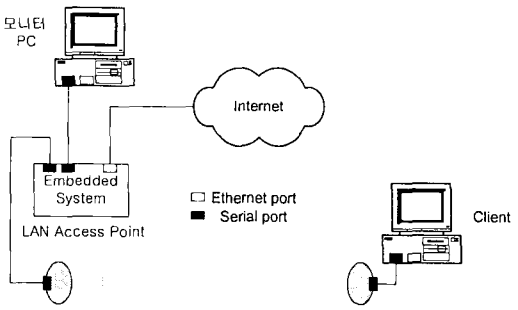

그림 3.9 LAN Access Point의 전체적인 시스템 구성도

LAN Access Point와 클라이언트는 PPP를 통해 IP 패킷을 주고 받고, PPP는 RFCOMM을 통해 HCI, L2CAP 등의 프로토콜 스택과 데이터를 주고 받는다.

3.6 구현 모듈의 수행 실험

실험에 사용한 응용프로그램은 대표적인 TCP/IP 프로그램인 Telnet, FTP, 웹 브라우저를 사용하였다. LAN Access Point의 IP주소는 210.110.81.188을 사

용하였고 클라이언트가 할당받을 IP주소는 210.110.81.234이다.

클라이언트와 LAP가 RFCOMM을 통해 PPP접속이 이루어진 화면을 그림 3.10에서 보여준다. ifconfig 명령을 실행시켜보면 ppp0라는 디바이스가 추가되어 있고 클라이언트(210.110.81.234)와 LAP(210.110.81.188)가 Point-to-Point Protocol로 연결되어 있는 것을 볼 수 있다.

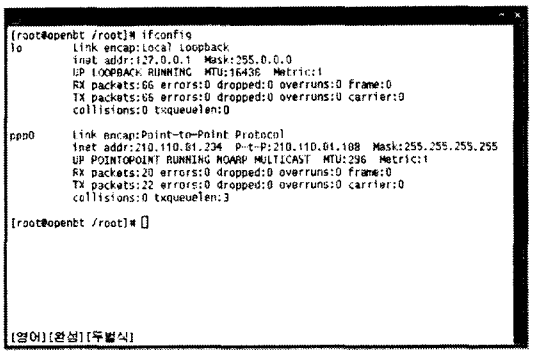

그림 3.10 클라이언트의 PPP접속

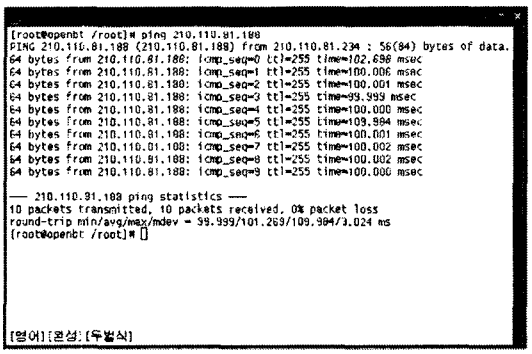

그림 3.11 클라이언트에서 LAP로 ping 테스트

그림 3.11에서 클라이언트에서 LAN Access Point로 ping을 실행시켜보면 이상 없이 접속되어있음을 알 수 있다. 또한 그림 3.12와 3.13에서와 같이 telnet과 FTP도 정상적으로 동작되었다.

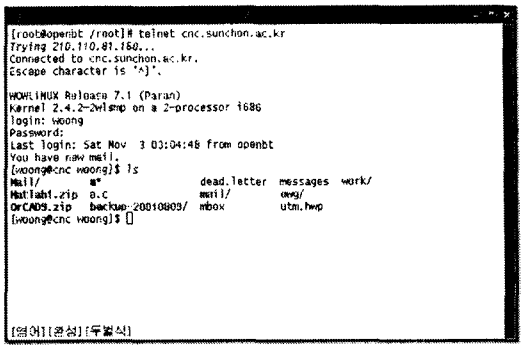

그림 3.12 LAP를 통한 telnet 실행

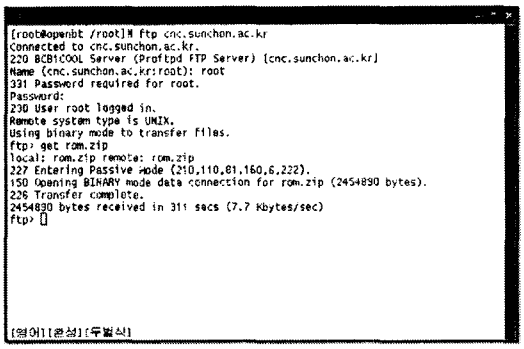

그림 3.13 LAP를 통한 FTP 실행

IV. 결 론

인터넷으로 통칭되는 통신 혁명이 낳은 결과물인 정보가전 분야에서 임베디드 리눅스의 적용분야는 다양하다. 휴대용 단말기에서 PDA, HandHeld PC등 임베디드 리눅스의 발전 가능성은 무한하다. 임베디드 리눅스와 블루투스 모두 새로운 기술이기 때문에 앞으로 개선의 여지가 많은 기술이지만 가능성은 충분하다고 볼 수 있다. 또한, 이같은 기술을 바탕으로 하여 가정용 혹은 사무용 전자 장치의 제어를 쉽게 할 수 있을 것으로 사료된다.

본 연구에서는 임베디드 리눅스를 이용해서 블루투스 LAN Access Point를 구현해 그 가능성을 실험해 보았다. 일차적인 목표는 LAN Access Point를 통한 블루투스 장치의 TCP/IP 응용프로그램의 실행 가능성을 시험해 보는 것이었는데, 실험결과 충분히

가능함을 볼 수 있었다. 하지만 Host와 블루투스 모듈사이의 인터페이스로 UART를 사용한 관계로 전송 속도는 크게 고려하지 않았다. 실험에 클라이언트로 일반 PC를 사용하였는데 일반 PC의 UART 최고 속도가 115200bps 밖에 되지 않기 때문에 전송속도는 무시하고 실행 가능 여부에만 중점을 두었다.

향후 멀티포인트 접속과 UART외의 다른 인터페이스를 통한 접속에 관한 연구가 계속되어야 할 것이다.

"본 연구는 2001년도 순천대학교 산·학·연 컨소시엄사업에 의해 수행되었으며 이에 감사드립니다."

참 고 문 헌

[1] Michael Beck, Addison-Wesley, Linux Kernel Internals, 1997

[2] Alessandro Rubini, Linux Device Drivers, O'reilly, 2000

[3] 한성호, "임베디드 시스템에서 리눅스 커널을 위한 부트 로더에 관한 연구", 성균관대 정보통신대학원, 2001, 2.

[4] 차현준, "네트워킹 디바이스를 위한 임베디드 리눅스 시스템의 설계 및 구현", 포항공과대 대학원 전자컴퓨터공학과, 2001, 2.

[5] http://www.bluetooth.com

[6] Bluetooth Special Interest Group, Core, Specification of the Bluetooth system ver. 1.1, November 2000

[7] Bluetooth Special Interest Group, Profiles, Specification of the Bluetooth System ver. 1.1, November 2000

[8] ETSI, TS 07.10, Versiion 6.3.0

[9] International Telecommunication Union, "ITU-T Recommendation Q.931"

[10] Simpson, W., Editor, "The Point-to-Point Protocol (PPP)", STD 50, RFC 1661, Daydreamer, July 1994

저 자 소 개

이병웅(Byeong-woong Lee)

순천대학교 정보통신공학과 공학사(2000.2)
순천대학교 정보통신공학과 공학석사(2002.2)
현재 (주)터보테크 연구원

오원근(Won-Geun Oh)

1989년 2월 한양대학교 전자통신공학 학사
1991년 2월 한양대학교 대학원 전자통신공학 석사
1997년 2월 한양대학교 대학원 전자통신공학 박사
1997년 3월~현재 (국립)순천대학교 정보통신공학부 조교수
※관심분야 : 신호처리, 능동소음제어, 신경망, 비선형시스템, 마이크로프로세서 응용

여 현(Hyun Yoe)

1984년 항공대학교 전자공학과(공학사)
1987년 숭실대학교 전자공학과(공학석사)
1992년 숭실대학교 전자공학과(공학박사)
1987년 2월~1993년 2월 한국통신 통신망 연구소
1993년 3월~현재 순천대학교 정보통신공학과 교수
1997년 8월~1998년 8월 미국 죠지아 공과대학(Georgia Tech) 방문연구원
※관심분야 : LAN, Wireless Network, Wireless ATM, Internet routing, xDSL Netwok