2001년도 대한전기학회 하계학술대회 논문집 2001.7.18-20

TCP/IP와 무선랜을 이용한 분산제어시스템 구현

박길성 남부희
강원대학교 BK21 전기전자정보통신공학부

Implementation of Distributed Control System using TCP/IP and Wireless LAN

Gil Sung Park and Boo Hee Nam
BK21 Dept. of Electrical and Computer Engineering, KANGWON NATIONAL UNIVERSITY

Abstract - TCP/IP 프로토콜을 사용하여 서버와 클라이언트를 연결하고 서버는 무선랜을 이용하여 여러 개의 디바이스를 제어할 수 있는 분산제어시스템을 구현했다. 디바이스의 제어 명령과 모니터링을 하는 클라이언트는 LAN을 통하여 서버에 접속한 후 디바이스 제어 명령을 서버에 전달한다. 서버는 디바이스 제어 명령을 수신한 후 실제 데이터만을 골라내어 디바이스가 받아들일 수 있는 데이터 형식으로 변환하여 RF 모듈을 통해 디바이스에 전달한다. 서버로부터 데이터를 수신한 디바이스는 제어명령을 실행한 후 센서는 디바이스의 상태를 체크하여 RF모듈을 통하여 서버에 전달한다. 다시 서버는 디바이스의 상태를 클라이언트가 받아들일 수 있는 데이터 형식으로 변환하여 전달하고 클라이언트는 서버로부터 데이터를 수신하여 디바이스의 상태를 모니터링 한다. 각각의 디바이스는 자신만의 마이크로 프로세서를 가짐으로서 서버로부터 간단한 명령을 받아 각 프로세서가 스스로 판단하고 동작을 하게 된다. 서버와의 연결이 끊어질 경우 마이크로프로세서에 마지막 들어온 제어명령의 상태를 유지한다. 이 논문은 인터넷을 이용한 무선장치의 제어할 수 있는 안정된 시스템의 구현방법을 제시하며 서로 다른 프로토콜을 연결시키기 위하여 데이터의 구조를 변환시키고 여러 개의 디바이스가 안정된 동작을 하도록 하는 방법을 제안했다.

1. 서 론

정보 통신 기술의 발달, 초고속 통신망의 확대 그리고 국가적인 정책에 힘입어 인터넷은 빠른 속도로 보급되었고 사용자가 계속 늘어나고 있다. 인터넷은 생활의 일부가 되었으며 회사, 가정, 학교, 은행, 관공소 심지어는 PC방이라는 곳까지 생겨 장소에 상관없이 인터넷을 사용할 수 있게 되었다. 성별, 연령층이 다양화 되면서 사용자들은 다양한 인터넷서비스를 요구하게 되었고 인터넷 기술은 급속도로 발전하여 채팅, 정보의 공유, 광고, 메일서비스등 많은 분야에서 빛을 발하여 우리의 생활 편리에 많은 도움을 주었다. 인터넷 서비스의 확대는 실제적인 하드웨어 장치를 제어할 수 있는 서비스까지 요청하기에 이르렀다. 즉, 인터넷을 통한 가스누출, 화재경보, 무인경비시스템, 웹카메라, 홈네트워킹 시스템등 많은 분야에 적용시키기 위해 많은 기업들이 노력하고 있다. 본 논문에서는 위에서 언급한 기술들의 기반이 될 TCP/IP 프로토콜과 무선랜을 이용한 분산제어 시스템을 구현하였다.

2. 본 론

2.1 시스템의 구성

전체 시스템은 그림1과 같은 설계했다. 인터넷의 통하여 클라이언트는 서버에 접속한 후 서버를 통하여 여러 개의 디바이스를 제어하고 모니터링 한다.

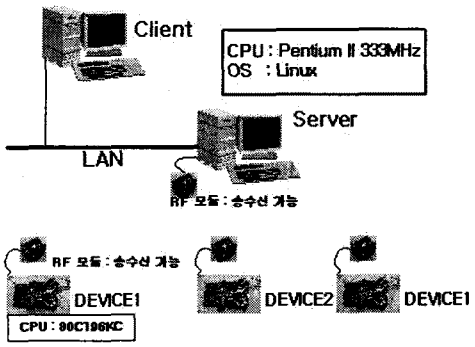

그림 1 시스템 구성도

그림1에서 서버는 OS로 리눅스를 사용하고 있다. 리눅스는 오픈소스 OS이기 때문에 내용을 사용자 임의로 내용을 추가, 삭제할 수 있다. 즉, 윈도우의 경우 모든 작업을 어플리케이션 계층과 같은 상위 레벨에서만 작업이 가능하기 때문에 하드웨어의 직접적인 접근이 어렵다. 리눅스는 사용자가 자신에게 필요한 모든 레벨에 접근 가능하여 프로그래머는 자유롭게 모든 장치에 접근할 수 있다. 서버는 네트워크가 잘 구축되어 있어야하며 멈춤 없이 동작하여야함으로 매우 안정된 시스템이어야 한다. 리눅스는 서버용으로 개발된 OS이기 때문에 네트워크 구축이 가능하며 매우 안정적이다. 또한 MS사의 Window와 달리 비교적 적은 양의 메모리를 필요로 함으로 저사양의 환경에서도 구현이 가능하다. 그래서 서버는 이런 여러 가지 장점을 가지는 리눅스를 OS로 사용했다.

클라이언트는 TCP/IP 프로토콜을 사용하여 통신을 함으로써 이기종 컴퓨터간의 통신이 가능하다. 즉, OS와 상관없이 서로 데이터를 주고받을 수 있다. 그래서 클라이언트는 Visual C++를 사용한 Window용과 GCC를 이용한 리눅스용 두 가지 모두 만들었다.

디바이스의 경우 80C196KC을 이용하여 여러 개의 보드를 구현했다. 디바이스는 서버로부터 데이터를 받아들여 헤더를 읽어 자기에게 들어온 명령인지를 확인한 후 제어명령에 맞는 동작을 하고 자신의 상태를 서버에 응답해 준다. 이 시스템에서는 간단한 동작을 확인하기 위해 DC모터의 회전을 동작시키고 센서는 DC모터의 회전속도를 감지하여 서버에 응답하도록 설계했다.

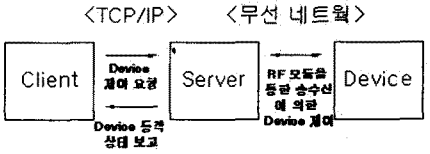

그림 2 시스템의 동작

그림2는 시스템에서의 데이터의 흐름을 간단하게 표현했다. 클라이언트는 서버에 디바이스 제어를 요청하고 서버는 명령을 디바이스에 전달하고 디바이스는 제어명령에 따라 동작한 후 변화된 상황을 다시 서버에 전달하여주고 서버는 다시 클라이언트에 디바이스의 상태를 알려준다. 클라이언트는 수시로 디바이스의 변화를 모니터링 할 수 있다.

2.2 TCP/IP의 기능 및 특징

TCP/IP는 국제 표준 프로토콜인 OSI의 Transport 계층/Network 계층에 해당되는 프로토콜이지만, 일반적으로 TCP/IP에 관계된 프로토콜 전체를 가르킨다. TCP/IP는 정확히 말한다면 'TCP/IP suite'이며, 컴퓨터 네트워크 통신에서 필요로 하는 일련의 프로토콜을 일컫는 말이다. 이 프로토콜은 이기종 컴퓨터 사이의 통신을 위해 개발된 통신 프로토콜로 좁은 의미의 ARPANET(Advanced Research Projects Agency Network)에서 제안한 프로토콜의 집합 중 인터네트워킹에 대한 핵심적 기능을 제공하는 TCP와 IP만을 지칭하기도 하지만, 넓은 의미로는 OSI(Open Systems Interconnection)의 3계층에서 7계층까지 해당하는 소프트웨어나 서비스의 관련 프로토콜의 집합을 지칭한다. TCP/IP suite 프로토콜은 계층 구조이므로, 어플리케이션을 네트워크용 하드웨어로부터 분리할 수 있다. TCP/IP는 계층적 모델을 기본으로 하지만, 기능적 계층을 엄격하게 따르는 것보다는 실제로 정확하게 연결되는 것을 중요시하고 있다. 이러한 이유에 의해, 네트워크의 상호 접속 프로토콜은 TCP/IP가 사실상의 표준으로 되어졌다.

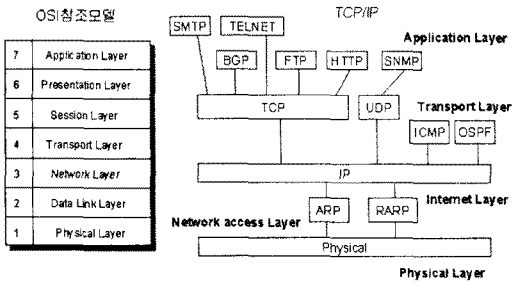

그림 3 OSI와 TCP/IP와의 계층비교

2.2.1 TCP/IP의 각 계층

(a) 네트워크 인터페이스 계층

이 계층은 하드웨어에 종속적인 기능과 인터넷 계층에 표준화된 인터페이스를 제공하고 MAC(Media Access Control)주소를 IP 주소와 mapping 시킨다.

(b) 인터넷 계층

주소 지정, packaging, routing 기능을 한다. 즉, 데이터 전송시 이용되는 경로배정과 중계 기능 및 호스트 구별 및 호스트간의 경로 상에 있는 라우터에서 동작한다. 프로토콜로는 IP, ICMP, ARP, RARP, IGMP등이 있다.

(c) 전달 계층

전달 계층 프로토콜은 TCP와 UDP로 구성되며 응용 프로토콜들은 TCP 혹은 UDP를 사용하여 데이터 전달 서비스를 구현한다. 본 논문의 경우 장치간의 연결 신뢰성을 중요시한다. UDP(User Diagram Protocol)의 경우 비연결형 통신이기 때문에 연결 신뢰성이 부족함으로 본 논문에서는 연결형 서비스를 하는 TCP를 사용하여 구현했다.

(d) 응용 계층

응용 계층 프로토콜을 구현하거나 응용 프로그래밍, 예를 들면 ping, telnet, ftp, NFS, SMTP등을 말한다.

2.3 세부 구현 내용

각 기기의 특징 및 동작을 자세히 알아보자.

2.3.1 시스템의 세부적인 동작

그림4는 위에서 보여준 시스템의 동작을 상세하게 보여주고 있다.

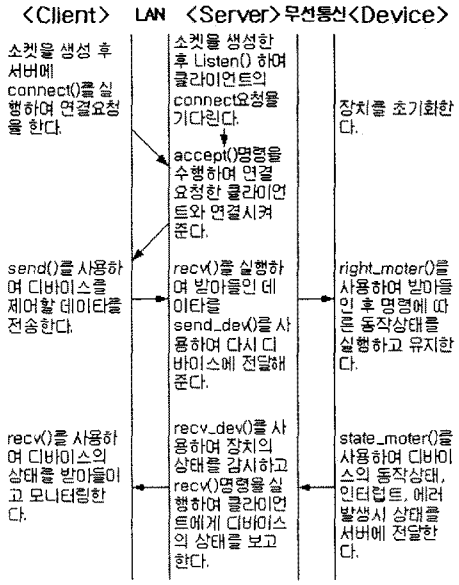

그림 4 시스템의 동작의 상세화

2.3.1 서버

TCP/IP 프로토콜을 사용하여 서버를 구현하였다. 이것은 클라이언트와의 신뢰성 있는 연결을 유지하며 데이터의 정확한 송수신을 가능하게 해준다. 서버는 클라이언트로부터 디바이스 제어명령을 받아들여 디바이스에 전달해 주며 디바이스를 모니터링 하여 클라이언트에 다시 전달해주는 역할을 한다.

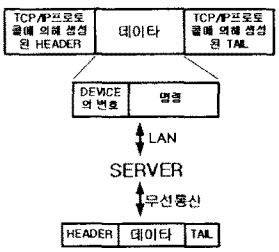

그림 5 서버의 역할

그림5는 서버의 역할을 보여준다. 서버는 TCP/IP 프로토콜을 사용하는 클라이언트로부터 받아들인 데이터를 분석하여 디바이스의 번호와 명령을 가지고 있는 실제 데이터를 잘라내어 디바이스에서 받아들일 수 있는 무선 데이터의 형식으로 변환해주는 역할을 한다. 또한 반대로 디바이스로 받아들인 데이터를 다시 클라이언트가 읽을 수 있는 형식으로 변환해 준다.

2.3.2 클라이언트

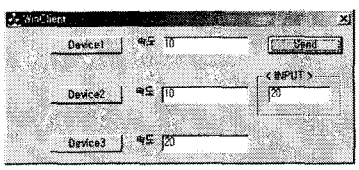

그림 6 Window에서 구현한 클라이언트

그림6은 윈도우즈용 클라이언트이다. 디바이스를 선택하고 INPUT에 속도를 넣고 send 버튼을 누르면 TCP/IP를 통해 구현된 서버에 데이터를 전달하고 서버는 디바이스에서 전달하고 디바이스는 다시 자신의 상태를 클라이언트에 알려주게 된다. 인터넷이 되는 장소 어디서든 프로그램을 다운받아 실행하여 디바이스를 제어하고 모니터링 하는 것이 가능하다.

2.3.3 디바이스

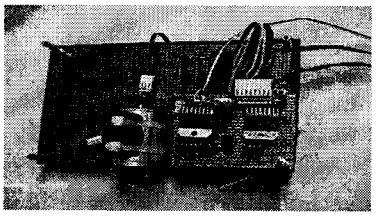

그림 7 디바이스

디바이스는 80C196KC보드를 사용하여 DC 모터 제어한다. 데이터의 송수신은 장치로 RF모듈을 사용하였다. 디바이스의 경우 무선통신을 사용하였는데 이것은 서버 하나가 날려주는 데이터를 여러 개의 장치가 수신하기 위해서이다. 디바이스는 데이터를 읽어 디바이스의 번호를 확인하고 PWM_CONTROL을 이용해 가속과 감속을 하고 포토 트랜지스터를 이용한 엔코더와 4개의 슬롯을 이용해서 1회전의 시간을 이용해 속도를 체크하여 서버에 알려준다. 디바이스는 클라이언트부터 받아들인 속도를 벗어날 때마다 자신의 상태를 디바이스 번호와 상태를 패킷으로 만들어 서버로 리턴한다. 서버가 에러 상태인 경우는 마지막 들어온 명령의 상태를 유지한다.

3. 결 론

본 논문에서 클라이언트와 서버의 통신은 TCP/IP 프로토콜을 사용했다. 이는 클라이언트와 서버의 안정적인 데이터 송수신을 가능하게 했다. 서버와 디바이스는 무선통신을 사용한 시스템을 구현했고 이렇게 함으로써 선연결의 불편함을 없애고 디바이스의 이동을 가능하게 했다. 하지만 무선 통신 중 장애물 및 다른 여러 방해요소에 의해 생기는 노이즈에 의하여 정확한 데이터의 전달이 쉽지 않았으며 RF모듈의 느린 송수신 속도로 인하여 실시간으로 디바이스 제어하는 것이 어려웠다. 속도 문제의 해결을 위해선 빠른 속도의 RF모듈의 개발과 서버와 송수신 장치의 인터페이스로써 USB포트를 사용 한다면 가능할 것이라 생각한다. 본 논문에서 구현된 서버는 고성능의 PC를 사용하였기에 구현은 쉬웠으나 실제 생활에 적용할 경우 부피가 크고, 값이 비싸며, 전력소비량이 크다는 등의 많은 단점을 가진다. 이런 서버가 가진 여러 문제를 해결을 할 수 있는 한가지 방안으로 TCP/IP 프로토콜과 무선통신을 연결시켜줄 수 있는 리눅스 임베디드 서버 구현에 관한 연구가 필요하다.

[참 고 문 헌]

[1] Richard & Neil, "Linux Programming 2nd Edition", wrox press, 2000. 9. 15.
[2] 이상엽, "Visual C++ Programming Bible 6.0", 영진출판사, p.1609~1634, 1998. 11. 20.
[3] 이영진, "TCP/IP를 이용한 필드버스 monitoring 시스템 구현",한국자동제어학술대회(KACC), p.684~687, 1998.
[4] W. Richard Stevens, TCP/IP Illustrated, volume 1, Massachusetts, Addison Wesley Publishing Company, 1994.
[5] 차영배, "MICRO CONTROLLER 80196", 다다미디어, 1999. 7. 15.
[6] 박귀태 & 이상락, "C언어로 쉽게 쓰는 80C196KC", 대영사, 1994. 5. 1.
[7] Igor R.Belousov, "Virtual Reality Tools for Internet Robotics", ICRA, p.1878~1883, 2001