임베디드 네트웍용 프로세서 개발

유문종 · 최종운

호남대학교 정보통신공학부

Development of Embedded Network Processor

Moon-jong Yoo[1], Jong-woon Choi[2]

Department of Information and Communication, Honam University

E-mail : caobr@chollian.net[1]

woon@honam.ac.kr[2]

요 약

8비트급 마이크로프로세서를 사용하여 HTTP 서버를 구현하였다. 사용한 프로세서는 Z80 코어를 채용한 TMP84C015 이고, 이더넷의 물리층은 RTL8019AS를 사용하여 구현하였다. 8비트 프로세서라는 제약과 사용 가능한 메모리의 제한을 극복하기 위하여 프로토콜을 최대한 단순화하였고, 시간당 보낼 수 있는 패킷의 수를 최적화하기 위해서 어셈블리언어를 사용하여 TCP, UDP, IP, ICMP, ARP 프로토콜을 구현하였다. 클라이언트 측에서는 LabVIEW를 이용하여 설계 제작한 임베디드 서버의 동작을 확인하였다.

ABSTRACT

We made a HTTP server using 8 bit microprocessor. It was TMP84C015 which applied a Z80 core and RTL8019AS was installed for an ethernet physical layer. Assembly language was used to optimize a performance of the MPU, to overcome an restriction of memory size and to maximize the throughput of packet using TCP, UDP, IP, ICMP and ARP protocol. We used LabVIEW to verified the each protocol on the client side.

키워드

Embedded processor, TCP/IP, UDP/IP, ICMP, ARP, HTTP, 웹서버

Ⅰ. 서 론

생활의 일부가 된 인터넷의 응용 범위는 광범위하게 확산되고 있다. 특히 초고속통신으로 불리우는 ADSL망이 각 가정 및 사무실에 연결되면서부터는 유선 인터넷의 전송속도와 품질은 과거 모뎀을 사용하여 데이터를 전송하던 시대에 비하여 급격히 상승하고 있어 대량의 데이터를 원격지에서도 손쉽게 주고받을 수 있게되었다. 이와 같은 통신망의 발달의 원동력인 WWW는 이제 모든 인터넷에서 가장 기본적인 프로토콜로 인식되고 있다. 따라서 이와 같은 프로톨콜을 사용하여 원격지에 데이터를 송수신하거나 혹은 각 가정의 가전제품의 원격 조종이 한층 쉽게 구현될 수 있게되었다. 한편으로는 지금까지의 고성능 컴퓨터에서 구현되던 인터넷 프로토콜을 저가의 8비트급 CPU를 사용하여 구현하려는 연구의 필요성이 대두되고 있다. 특히 소형기기에 적합한 경제적인 간이 웹서버는 원격 측정 및 제어분야에

1) 호남대학교 대학원 정보통신공학과 석사과정

2) 호남대학교 정보통신공학과 교수

서 광범위하게 활용될 수 있기 때문에 많은 연구가 진행되고있다.

II. 프로토콜 개관

현재 사용되는 가장 일반적인 인터넷 통신망의 구성은 물리층을 UTP케이블[3]과 Manchester Encoding[4]을 사용하고, 데이터링크계층은 802.3[3]을 사용하며 우리는 Ethernet Adapter를 보통 '랜카드'라고 부른다. 그리고 네트웍계층을 IP로 전송계층은 TCP로 사용하고 그 위에 HTTP 프로토콜을 사용하는 웹브라우저를 사용하고 있다. (그림 1)

Application/ Presentation/ Session Layer	HTTP (웹브라우저)
Transport Layer	TCP
Network Layer	IP
Datalink Layer	Ethernet Adapter
Physical Layer	

그림 1. 프로토콜 스택

물리층과 데이터링크층의 일부 기능은 하나의 LSI로 판매되고 있으며, 여기에서는 사용이 간편하고 비교적 자료가 풍부한 RTL8019AS[10]를 사용하였다. 다음은 IP와 TCP인데 프로그램 작성에 큰 무리가 없도록 비교적 여유 있게 하드웨어를 구성하였다. 이는 완성된 이후에 쉽게 최소한의 사양으로 재구성이 가능하다.(그림 2)

그림 2의 왼쪽은 우선 8비트 Parallel 포트를 이용하여 하위 Nibble은 LED에 상위 Nibble은 스위치에 연결해 두었다.

III. 프로그램 개관

프로그램은 운영체제를 이용하지 않은 직접프로그래밍으로 하였다. 프로그램의 전체적인 흐름을 그림 5의 순서도에 표시하였다. 간단하게 설명하면 전원이 개시되거나 RESET이 눌려지면 하드웨어 초기화가 진행된다. 초기화 과정에서는 IP 및 MAC주소의 일련번호 부분의 변경이 가능하다. 또한 EEPROM 전체의 내용을 초기값으로 새로이 기록하는 것이 가능하도록 하였다. 이는 윈도에서

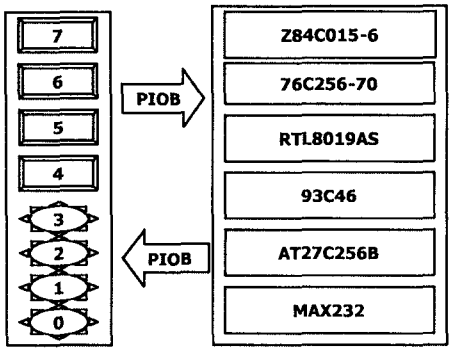

그림 2. 하드웨어 구성도

하이퍼터미널을 이용하였으며, 물론 프로그램 내부에 CheckSum등의 계산을 새로이 하여 기록이 되도록 하였고, 128바이트 용량 중 일부만을 사용하고 있으므로 여기에 상당한 내용을 추가하여 기억시키는 것이 가능하다. 초기화 과정에서 특별한 사항이 없으면 시스템을 다시 초기화한 후 Ethernet 청취상태로 들어간다. 청취상태에서 패킷이 검출되면 ARP, IP만을 사용한다. ARP 패킷이면 ARP 응답 및 처리를 하게 되고, IP이면 ICMP/UDP/TCP 이외의 패킷은 버리게 되며, ICMP는 ICMP echo에만 응답하도록 되어있으나 필요하면 확장이 가능하다. TCP는 HTTP에만 응답하도록 하였으며, 가장 간단한 서버가 되도록, 패킷의 크기를 헤더 포함하여 512바이트를 넘지 않도록 하였다. 물론 윈도우 크기도 1이다. 윈도우의 크기가 1이면 순서조정 및 패킷의 재전송 부분이 아주 단순해진다는 장점이 있다. TCB의 최대 숫자는 8로 하였으나 각각 다른 Client에서 서로 다른 정보를 요구하는 경우가 아니라면 1개면 충분할 것으로 보며, 웬만한 응용에서는 HTTP를 상시 연결된 상태로 사용하지 않기 때문이다. 클라이언트로는 여러 가지 편리한 점 때문에 LabVIEW[5] 프로그램을 이용하였다.

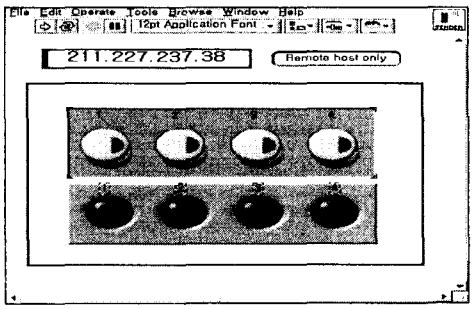

그림 3 Labview 화면(일부 색상 지움)

3) Unshielded Twisted Pair : 차폐층 없는 꼬인선

4) Biphase 방식중 biphase-L방식의 별명. 동기화가 가능하고 변조율이 2배, 오류복원이 쉬운 장점과, 대역폭이 2배로 필요한 단점이 있다.[3]

5) © Copyright 1999, 2000 Part Number 370110A-015) Lapview National Instruments Corporation. All rights reserved. July 2000

LabVIEW는 데이터 수집이나 기초적인 분석에서 아주 뛰어난 편리한 인터페이스와 기능을 가진 프로그램이다. 그림 3에 LabVIEW와 보드가 연결된 상태를 보여준다. 참고로 LabVIEW화면에서 마우스로 Button을 Click하면 보드의 LED를 On/Off할 수 있고, 보드에서 Button 스위치를 누르면 LabVIEW상에 그 상태가 나타난다. 웹브라우저에서 이를 구현할 때는 CGI[6]의 GET나 POST를 이용하여야 하고, 이를 웹서버에서 처리해 주는 부분이 있어야 한다. 물론 문자데이터에서 필요한 명령어를 검색하여 처리하는 방식이 된다. 이 후에 쿠키(Cookie)[7]를 이용하는 방안도 고려 중이나 프로그램이 복잡해지므로 반드시 좋다고 볼 수는 없다.

IV. 웹서버와 TCP

프로그램 작성 중 가장 복잡한 것은 역시 TCP 부분이다. 1개의 패킷 전송은 UDP와 별반 다를 것이 없으나 연결을 설정해놓고 모든 처리를 상대편과 협상하는 형식으로 진행되기 때문이다. TCP 프로그래밍을 하기 위해서 반드시 지나가야만 하는 과정으로는 첫째 하위의 IP 프로토콜과 상위의 HTTP프로토콜 및 Socket 개념이 필요하다. 둘째 TCP의 유한상태기계(Finite State Machine)에 대한 이해가 필요하다는 것이다. 이는 두 방향에서 진행되어야 하는데, 서버와 클라이언트 양쪽을 모두 고려하여야 하고, 상태 전이를 일으키는 요소와, 요소에 따른 TCB[1][2]의 변화 및 그 처리에 관하여 확실한 정리를 해둘 필요가 있다. 셋째로 TCP의 모든 기능을 필요로 하지 않기 때문에 시스템의 능력에 따라 적절한 삭감이 필요하다는 점이다. 대부분의 Embedded Processor는 필요한 성능을 만족하는 범위에서 소형, 저전력, 저가격이 필수조건이기 때문에 다량생산성을 만족하는 수준 또는 생산의 편의성 측면에서 최소 사양을 선택하여야만 한다. 이 시스템은 아직 최적화 단계를 지나지 않았기 때문에 미비한 점이 많이 있다. 이 논문에서는 TCP서버 작성에 필요한 부분을 집중적으로 논의하고자 한다.

가. State Change와 명령 전달(Flags)[1][2]

그림 4는 TCP의 상태변화 중 프로그램과 직접적인 관련이 있는 부분만을 나타냈다. 중간의 REQUEST는 TCP의 것이 아니고 HTTP의 홈페이지 요청 데이터를 나타내고 있다. 여기에 나타낸 것은 정상적인 연결의 설정과 해제를 나타낼 뿐이고, 그 외에도 ACK를 기다리다 설정된 시간이 되면 패킷을 재전송하는 등의 예외적 처리가 필요하다.

6) Common Gateway Interface

7) 쿠키 : 웹서버가 클라이언트의 환경 또는 접속통계 등을 쉽게 알 수 있도록 Client측에 저장되는 문자열.

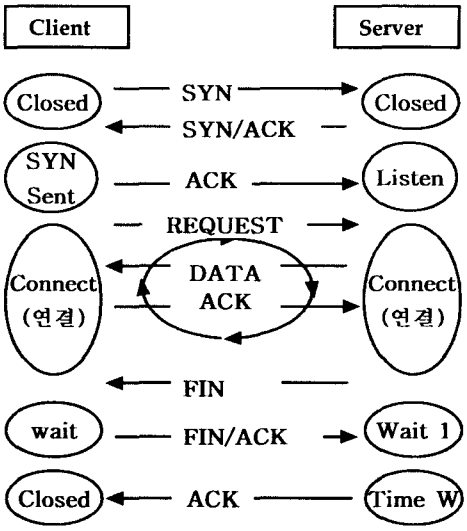

그림 4. 상태변화

나. 개략적인 순서도

TCP 모듈로 패킷이 전달되면, 먼저 HTTP인가를 검사하여 만약 HTTP이면 체크섬을 검사하게 된다. 체크섬에 이상이 없으면, FLAG를 조사하는데, URG, PUSH등은 검사하지 않는다.

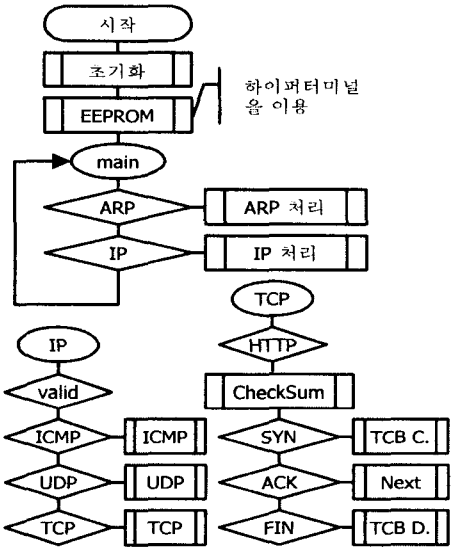

그림 5. 개략적인 순서도

이를 위해서 패킷의 최대 길이를 제한하는 Option을 SYN/ACK에 덧붙여 전송한다. FLAG 검사에서는 TCB 생성과 삭제에 관련된 내용과 데이터의 주고받음에 관한 내용만 검사한다. 한가지 유념해야할 사항은 웹클라이언트로부터 데이터를 받아들여야한다(Establish=Connect)는 것이

다. 서버에서 전송되는 홈페이지에 CGI의 GET를 이용한 폼(FORM)을 전송하고, 사용자가 EVENT를 일으키도록 하여 서버가 응답하도록 구성하였다. 따라서 HTTP 모듈 내부에는 CGI TEXT를 처리하는 부분이 필요하고, 여기에서 데이터의 수집 및 반영도 처리하고 있다.

또 한가지의 방법은 연속적인 화면의 갱신이 필요할 경우이다. 이 때는 HTTP 모듈이 타이밍 신호에 의하여 연속적으로 데이터를 전송하도록 하여야 하는데, 이 경우의 대처 방법은 시간제한성에 따라 프로그램이 다르거나 두 가지 경우 모두를 생각해야 한다. 예를 들어 CPU의 처리능력의 한계점에 가까워질 때는 TCP/HTTP가 계속 연결된 상태로 데이터 요청 패킷이 없어도 데이터를 계속 보내야 하는데 이러한 경우 CPU에 부담을 덜 주게되는 UDP를 시간 인터럽트 형태로 이용하는 편이 더 낳다. CPU의 처리능력이 충분할 때는 JavaScript의 Set/ClearTimeOut()등을 이용하여 필요한 시간 간격으로 클라이언트가 데이터를 요청하게 하면 된다.

V. 하드웨어

하드웨어(그림 2)는 1개의 병렬 포트를 사용하여, 스위치와 LED에 할당하였다. 외부 연결단자는 8비트 연결이 가능하도록 9핀 Connector를 배치하였다. 전원공급은 안정화된 5V를 공급하는 단자와 비안정화 6~12V 단자 중 선택하여 사용할 수 있도록 하였다. 시리얼 EEPROM에 MAC 주소의 시리얼 부분과 LAN에 맞는 IP 주소 등을 입력하여 사용하도록 구성하였으며, EEPROM의 프로그래밍은 RTL8019AS의 기능을 최대한 사용하였다. 프로그램 작성과 디버깅은 NLT TromICE-II+[8] 에뮬레이터를 사용하였으며, 작성이 완료된 프로그램은 최종적으로 EPROM에 저장하였다. 사용자 IP 주소 변경 및 디버깅을 위해서 RS-232C 시리얼 포트를 설치하였다.

VI. 결 론

8비트급 마이크로프로세서를 사용하여 HTTP 서버를 구현하였다. 사용한 프로세서는 Z80 코어를 채용한 TMP84C015 이고, 이더넷의 물리층은 RTL8019AS를 사용하여 구현하였다. 8비트 프로세서라는 제약과 사용 가능한 메모리의 제한을 극복하기 위하여 프로토콜을 최대한 단순화하였고, 시간당 보낼 수 있는 패킷의 수를 최적화하기 위해서 어셈블리언어를 사용하여 TCP, UDP, IP, ICMP, ARP 프로토콜을 구현하였다. 클라이언트 측에서는 LabVIEW를 이용하여 설계 제작한 프로그램으로 임베디드 서버의 동작을 확인하였다.

본 연구에서는 ASSEMBLY 언어[8][9]를 사용하여 프로그램을 하였으나, 앞으로 다른 프로세서에 프로토콜 이식을 고려하여 C나 C++ 언어사용의 필요성이 요구되고 있다. 또한 사용자의 보안성을 높이기 위해서 보안과 관련된 부분을 추가하여 범용으로 이용할 수 있는 소형 저가격의 Embedded Network Adapter로 발전시킬 계획이다.

[감사의 글]

본 연구는 2001년 산학연 공동기술개발 컨소시엄사업의 지원을 받아 수행되었습니다.

참고문헌

[1] Jeremy Bentham, TCP/IP Lean, CMP Books, 2000, chapters 6~7.
[2] Douglas E. Comer & David L. Stevens, Internetworking with TCP/IP Volume II 3rd Ed., Prentice-Hall Inc., 1999, chapters 11-14.
[3] 강창언, 정보통신이론, 복두출판사, 1995.7. p136.
[4] RFC 793 Transmission Control Protocol. J. Postel. IETF, Sep-01-1981.
[5] RFC 791 Internet Protocol. J. Postel, IETF, Sep-01-1981.
[6] 아세쓰 아끼히로외 공저, 인터페이스 시리즈 8, Sep-25-1997, p42.
[7] 윤종호, TCP/IP 및 윈도우 네트워킹 프로토콜, (주)교학사, 1999.3.10., 14장-18장.
[8] 양홍석 외, 마이크로컴퓨터(Z80중심) 설계 및 응용, 동일출판사, 1998.
[9] James W. Coffron, Z80 Applications, Sybex, 1979.
[10] RTL80xx DATA BOOK, REALTEK SEMICONDUCTOR CO., LTD.

8) NLT TromICE-II+ : NL Telecom Co.,Ltd. 의 롬 애뮬레이터, 홈페이지: http://nltelecom.com/