StrongARM을 이용한 원격 감시시스템

(Remote Monitoring Systems Using StrongARM)

임홍식* · 남현도* · 강철구**
* 단국대학교 전기공학과, **건국대학교 기계공학과
(Hong-Sik Lim · Hyun-Do Nam · Chul-Gu Kang)
(* Department Of Electrical Engineering. Dankook University, ** ME, Konkuk University)

Abstract

In this paper, web based monitoring systems are implemented using embedded Linux. The external equipment is controlled via HTTP protocol and web browser program. HTTP protocol is ported into Linux. A micro web server program and external equipment control program are installed on-board memory using CGI to be accessed by web browser. Experimental result of the proposed web based monitoring systems can be used in automation systems and remote distributed control via internet using web browser.

1. 서 론

요즘은 정보화 시대 또는 네트워크 시대라고 불리면서 다수의 장비들과 시스템들이 이를 수용하기를 원하고 있다. 즉, 영상, 음성, 이미지 등과 같은 대용량의 데이터가 갈수록 늘어나면서 전송이나 저장에 대한 안정성, 신속성, 그리고 편리성 때문에 네트워크 기능은 여러 분야에 응용화가 요구되어지고 있다. 네트워크 기능을 사용하기 위해서는 PC의 역할이 중요하지만 대부분의 장비들은 네트워크 기능을 요구하는 점 외에는 각각의 쓰임과 특성이 달라서 PC만을 이용하는 방식은 여러 가지 요구를 수용하기가 쉽지 않다. 성능에서는 일반 PC와 차이가 없고 다양한 요구를 수용할 수 있는 장비의 개발이 필요해지면서 내장형(Embedded) 시스템에 대한 연구와 개발이 활발히 이루어지고 있다. 건축물이 점차 대형화하여 감에 따라 건축물에서 사용하는 전기용량도 증가하고 있으며, 생산성을 고려한 조명환경의 개선에 따른 조명용 전력이 크게 증가하는 추세에 있다. 특히 업무용 빌딩의 경우는 조명용 전력의 비중이 대단히 높게 나타나고 있으며, 형광등이 대부분 사용되고 있다[1].

내장형 시스템이란 특정 기기나 시스템에 하드웨어, 소프트웨어가 내장되어 정해진 기능을 실행하는 시스템으로서 다양한 용도로 활용되는 일반적인 컴퓨터 시스템과 구별하여 명명한 것으로 다양한 시스템을 포함하게 된다.

본 논문에서는 리눅스를 기반으로 한 ARM 프로세서에 웹 서버를 구현하고 ARM 프로세서에 연결된 외부 기기를 인터넷을 통한 범용 웹 브라우저에 의하여 감시하는 시스템의 구현을 다루었다. 이를 위하여 StrongARM 프로세서용 기동 시스템과 내장형 리눅스 운영 시스템을 이용하여 웹 서버를 구현하고, 웹 브라우저에서만 이루어지는 단방향 통신이 아니라 HTML(Hyper Text Markup Language)을 이용해 서버와 사용자간의 양방향 통신을 할 수 있는 CGI(Common Gateway Interface)를 이용하여 프로그래밍 하였다. 리눅스 운영 시스템에서 외부 기기를 제어하기 위하여 디바이스 드라이버를 구현하고 범용 웹 브라우저를 통하여 외부 기기를 감시하는 웹 기반 감시 시스템을 구현하였다.

2. 본 론

2.1. 내장형 리눅스 포팅

2.1.1. 리눅스

리눅스라는 운영 체제는 유닉스와 유사한 운영 체제로서 유닉스의 장점을 이용하여 필요로 하는 기능을 추가한 운영 체제이다. 내장형 시스템 구현에 사용되어지는 내장형 리눅스는 일반적인 PC상에서의 리눅스와는 기본적으로 특정 내장형 애플

리케이션에 맞도록 리눅스 커널의 크기와 성능을 최소, 최적화 시켜 만들어낸 커널을 의미하며 여기에 C 라이브러리나 유틸리티 등이 필요에 따라 추가되거나 교체되어 하나의 시스템을 만드는 형태가 된다. 커널의 최소, 최적화시 커널 내부 구조의 변화도 가능하며 이를 극단화시키면 일반적인 상용 운영 체제의 성능을 뛰어 넘는 것도 가능하다. 내장형 리눅스를 사용하는 이유는 누구나 제약 없이 자유롭게 사용할 수 있는 운영 체제이며 소스가 오픈 되어 있어 소스를 변형, 개발, 재배포 할 수 있는 특성을 가지고 있기 때문이다. 장점으로는 한 대의 시스템에서 여러 사용자들이 사용할 수 있는 다중 사용자 환경 시스템이나, 여러 가상 작업 공간을 사용자에게 제공하여 한대의 컴퓨터 내에서 여러 개의 화면을 통하여 다수 작업들을 동시에 수행할 수 있는 다중 작업 및 가상 터미널 환경을 제공하는 점이다. 또, 네크워크 기능을 갖춘 서버로 사용할 수 있도록 인터넷에서 쓰이는 프로토콜을 대부분 지원하므로 인터넷을 위한 각종 데몬 프로그램을 동작시킬 수 있다. 웹 서버, 이메일 서버, DHCP 서버, DNS 서버, FTP 서버와 같이 인터넷을 운영하는데 있어 필수 불가결한 각종 소프트웨어를 내장형 리눅스에서 운영할 수 있다. 또한, NFS나 AFS와 같은 유닉스를 위한 네트워크 파일 시스템은 물론이고, 삼바를 사용해 윈도우 시스템과도 손쉽게 데이터를 주고 받을 수 있다. MS-Window 운영 체제에서는 하드웨어를 제조한 업체에서 제공한 드라이버를 사용하여야 하지만, 리눅스에서는 하드웨어를 제공한 업체의 드라이버가 아니더라도 하드웨어가 사용하고 있는 칩만 동일하다면 하나의 드라이버로 모든 제조업체의 하드웨어를 사용할 수 있다[3]. 이러한 리눅스의 특성을 바탕으로 StrongARM보드(Hyper104B)에 리눅스(Wowlinux 7.1)를 이용하여 시스템을 구현하였다.

2.1.2. StrongARM 프로세서

StrongARM SA1110 프로세서는 400mW의 낮은 소비전력과 작은 패키지로 최근 내장형 시스템에 많이 쓰이고 있다. 32비트 RISC(Reduced Instruction Set Computer)방식의 프로세서로 Big/Little Endian 모드를 지원한다. 또 클럭발생기 내장, 전원 제어 모드, MMU(Memory Management Unit), 명령 캐쉬(16KByte), 데이터 캐쉬(8KByte), 6 채널의 DMA(Direct Memory Access) 제어기, 28개의 인터럽트 처리 가능한 GPIO(General Port Input Output) 등의 기능을 갖추고 있다[5]. 그림 1에서는 StrongARM 프로세서 SA1110의 구조를 보인다.

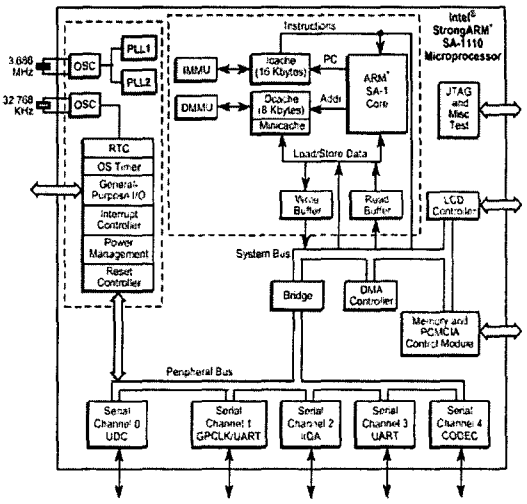

그림 1. StrongARM SA1110 블록 다이어그램

2.1.3. Cross 개발 환경

리눅스를 호스트 컴퓨터에서 타겟 보드로 이식하기 위해서는 작업 환경 구축이 필요하다. Cross 개발 환경이란 호스트 시스템에서 타겟 디바이스용 리눅스를 개발하기 위한 모든 환경을 말한다. 우선 소프트웨어 개발을 진행하기 위해 필요한 각종 소스들을 컴파일하고 바이너리 파일을 생성하는데 필요한 각종 유틸리티 및 라이브러리 모음인 ToolChain을 설치한다. 본 논문에서 사용한 컴파일러는 GNU Tool를 사용하였다. 일반 GNU Tool(x86계열)과는 다른 컴파일러(ARM Cross 컴파일러)를 사용하기 때문에 컴파일하게 되면 ARM에서 실행 가능한 바이너리가 생성된다.

2.1.4. 부트 로더(Boot Loader)

부트 로더는 인텔 관련 보드(X86)에서 말하는 BIOS(Basic Input Output System)와 LILO(Linux Loader)를 결합한 것이라고 하면 이해하기가 쉽다. 내장형 장비에서의 부트 로더 기능은 하드웨어에 대한 초기화(MCU, SDRAM, 플래시 메모리, 인터럽트, UART)를 해주게 되며, 리눅스를 부팅할 수 있게 한다. 또한, 커널(Kernel) 이미지나 램디스크(Ram disk) 이미지를 다운로딩 할 수 있다. ARM에 쓰이는 부트 로더는 BLOB(Boot Loader

OBject)이 있으며 본 논문에서는 BLOB-2.0.5-pre2를 보드에 맞게 포팅하여 사용하였다. BLOB을 컴파일 하기 위해서는 ToolChain이 설치되어 있어야 하며 이를 플래쉬 메모리에 직접 쓰기 위하여 JTAG(Joint Test Action Group) 프로그램인 Jflash를 사용하였다.

2.1.5. 커널 구성

커널은 운영 체제의 가장 핵심적인 부분으로서 프로세서와 시스템 메모리를 관리하여 시스템을 효율적이고 원활히 동작시키는 기능을 수행한다. 내장용 커널은 일반 리눅스에서의 커널을 타겟보드의 특성에 맞는 설정을 하고 의존성을 검사한 후에 이미지 파일을 만들 수 있다. 만들어진 이미지 파일은 BLOB을 이용하여 SDRAM에 다운로드 한다. SDRAM은 전원이 차단되면 저장된 파일을 보존하지 못하므로 전원을 인가할 때 마다 커널 이미지 파일을 다시 다운로드 하는 것을 방지하기 위해 Jflash를 이용하여 플래쉬 메모리에 다운로드 한다.

2.1.6. 파일 시스템

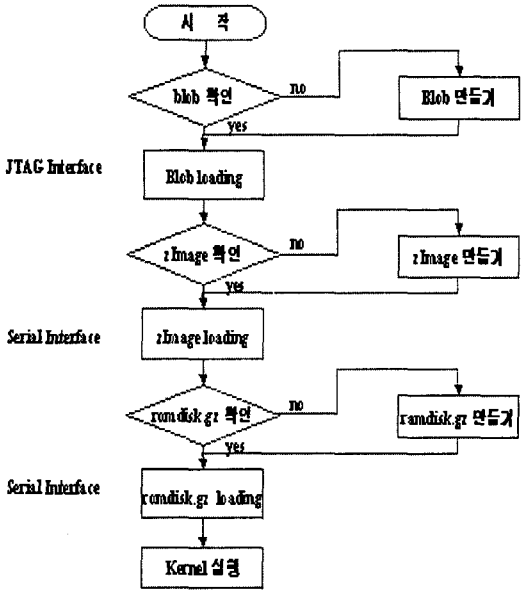

그림 2. 리눅스 운영 시스템 실행 과정

파일 시스템은 램 디스크, Jffs, Jffs2, Cramfs, Ramfs중에서 타겟 보드 사양에 맞게 여러 가지를 선택적으로 사용 할 수 있다. 램 디스크란 별다른 물리적 장치를 지칭하는 것이 아니라, 메모리의 일부를 디스크로 인식시킨 것이다. 램 디스크를 Root 파일 시스템으로 사용하는 것이 내장형 리눅스에서 가장 일반적인 방법이다. 램 디스크는 램에서 동작하기 때문에, 속도가 빠를 뿐만 아니라 gzip 알고리즘으로 압축을 하기 때문에 용량을 줄일 수 있다는 장점 때문이다. 하지만 램은 휘발성이기 때문에 전원을 차단하면 데이터가 없어진다는 단점이 있다. 이를 보완하기 위해 플래쉬 파일시스템(Jffs, Jffs2)을 같이 사용한다.

2.2. 원격 제어를 위한 웹 기반 감시 시스템

2.2.1. 내장형 웹 서버

웹으로 제어를 하기 위해서는 타겟 보드 상에서 웹 서버를 구현해야 한다. 보아 웹 서버는 아파치와 같은 강력한 성능을 제공하지는 않지만 내장형 기기에서 웹 서비스를 제공하기 위한 용도로서 작은 크기의 서버 프로그램이다. 웹 서버에 추가로 들어오는 서버의 요청에 대해 하위 프로세스를 생성하지 않고 내부적으로 모든 연결을 다중화 한다. 크기가 작기 때문에 기능의 제한은 있으나 기본적인 HTML 문서의 전달을 하는 HTTP 프로토콜과 CGI를 지원한다. 그림 3은 내장형 웹 서버의 구성을 보여준다.

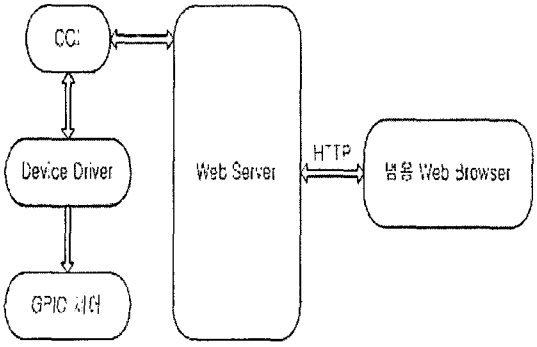

그림 3. 내장형 웹 서버 구성

내장형 웹 서버는 정보를 교환할 수 있는 HTTP와 CGI 지원 가능한 웹 서버 그리고 외부 기기를 제어하기 위한 디바이스 드라이버와 디바이스 드라이버를 제어하여 웹 서버에게 정보를 전달하는 CGI 프로그램 등으로 구성된다.

2.2.2. 디바이스 드라이버

리눅스 운영 시스템에서 하드웨어 주변장치를 제어하기 위해서는 그림 4와 같이 디바이스 드라이버가 필요하다.

하드디스크, 플로피 디스크, 프린터, 단말기, 스캐너와 같이 컴퓨터 시스템 이외의 다른 주변 장치를 디바이스라 하며 리눅스에서 디바이스는 파일을 통해서 접근이 가능하다. 디바이스 드라이버는 서브루틴과 데이터의 집합체로 디바이스와 시스템 사이에 데이터를 주고받기 위한 인터페이스이다. 표준적으로 동일한 서비스 제공을 목적으로 하고 디바이스의 고유한 특성을 감추며 커널의 일부분으로 내장된다. 디바이스 드라이버의 종류로는 크게 3가지가 있다.

첫 번째는 캐릭터(Character) 디바이스 드라이버로 자료의 순차성을 지닌 장치이다. 캐릭터 디바이스 드라이버는 버퍼 캐쉬를 사용하지 않으며 장치의 Raw 데이터를 사용자에게 제공한다.

두 번째는 블록(Block) 디바이스 드라이버다. 이 장치는 랜덤 엑세스(Ramdom Access)가 가능하고 파일 시스템에 의해 마운트(Mount)되어 관리되며 블록 단위의 입·출력이 가능한 장치들이다. 하드디스크나 시디롬, 플로피 디스크등이 있다.

마지막으로 네트웍(Network) 디바이스 드라이버다. 이 장치는 통신과 관련된 장치들로 TCP/IP, 서버 등을 관리할 수 있는 것으로 소켓으로 관리된다.

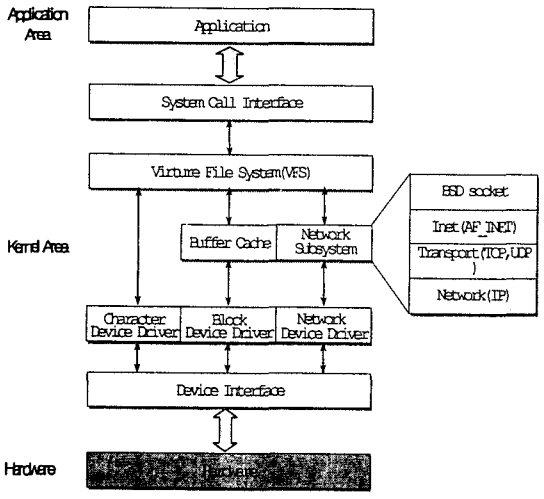

그림 4. 리눅스 시스템 구조

디바이스 드라이버는 커널 모드에서 실행되며 메모리에 상주하고 이것을 통해 하드웨어 장치를 파일처럼 다룰 수 있게 된다.

본 논문에서는 외부 기기(신호발생기)를 웹 서버에 의하여 감시하는 것을 다루므로 외부 기기를 제어하기 위한 디바이스 드라이버를 작성하였다.

ARM 프로세서의 외부 기기는 버퍼를 통하지 않고 직접 읽고 쓸 수 있는 장치로서 캐릭터 디바이스라 하며 커널 모듈(Kernel Module: 커널의 일부분을 동적으로 로드 또는 언로드할 수 있는 커널의 구성요소) 프로그램에 의하여 모듈로 사용할 수 있게 한다[3]. 커널 모듈을 만들 때는 반드시 초기화 함수와 삭제 함수가 정의되어 있어야하며 커널 버전이 정의되어 있어야 한다. 그림 5는 캐릭터 디바이스 드라이버의 구조를 보여준다.

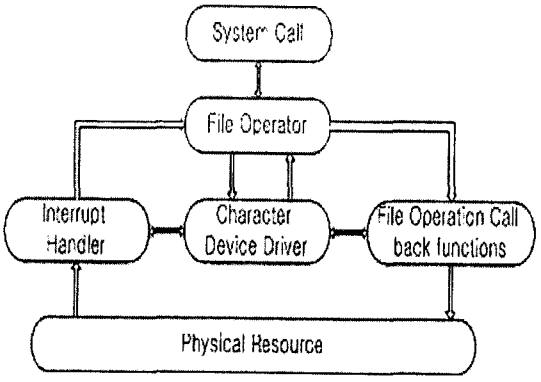

그림 5. 캐릭터 디바이스 드라이버의 구조

2.2.3. CGI (Common Gateway Interface)

웹 브라우저에서 HTML로 여러 가지 정보를 처리하지만 그 기능만으로 모든 정보 처리를 할 수 없다. 이것을 보충하기 위한 외부 프로그램과 웹 서버간의 연결 역할을 하기 위한 규약이 CGI로서, 웹 브라우저와 웹 서버간의 정보 전달을 위한 중간 역할을 제공한다. 사용자는 웹 특정 서비스를 요청하고 필요에 따라 웹 브라우저를 이용하여 데이터를 작성한다. 웹 클라이언트는 사용자에게 데이터를 입력할 수 있도록 하고 입력된 데이터를 인코딩하여 웹 서버에게 전달하며, 웹 서버로부터 넘어 온 서비스 결과를 사용자에게 보여준다. 사용자의 요청에 따라 해당 CGI 프로그램을 구동한다. CGI 프로그램은 웹 서버에 의해 실행되고 웹 서버로부터 전달받은 데이터는 해당 데이터를 해석하여 요청에 맞는 작업을 수행한다[3].

2.3. 실험 및 결과

2.3.1. 실험 환경

호스트 PC는 와우 리눅스 7.1 Paran을 사용하였고 타겟 보드는 StrongARM SA1110을 CPU로 사용하며 32MByte SDRAM, 16MByte 플래시 메모리

를 갖고 있으며 외부 기기는 신호발생기를 사용하였다. 그림 5.1과 5.2는 전체 시스템 구성도와 실제 실험 시스템이다.

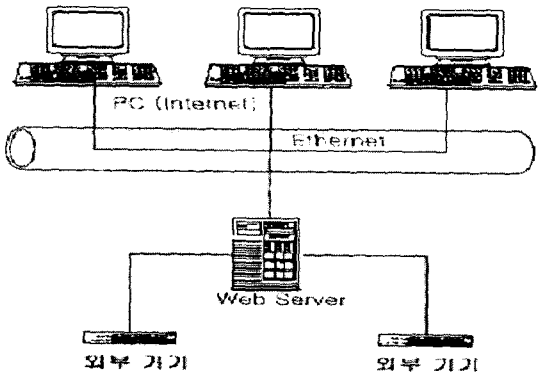

그림 6. 전체 시스템 구성도

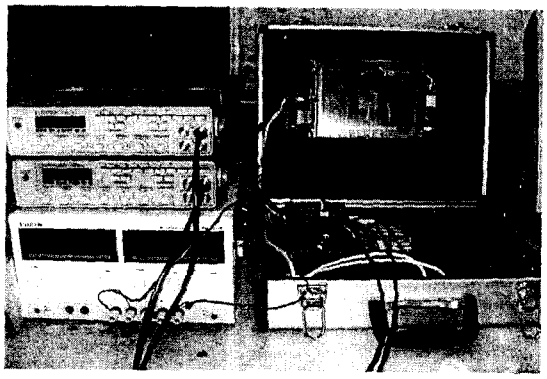

그림 7. 실험 시스템

2.3.2. AD 컨버터

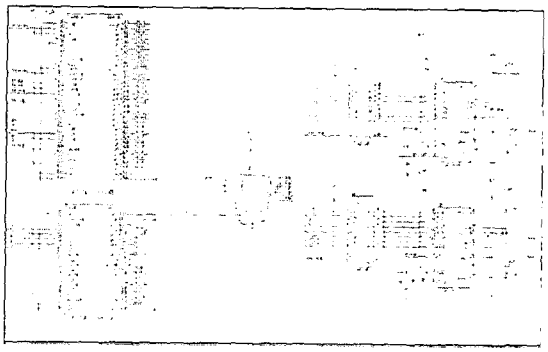

그림 8. AD 컨버터 회로도

외부 기기의 데이터를 디지털로 변환하는 AD 컨버터 회로를 그림 8에 나타내었다. ADC0804의 클럭은 193Hz이다.

2.3.3. 웹 서버

AD 컨버터에서 받은 데이터를 웹 브라우저를 통해 보여진다. 이때 데이터량이 너무 많으면 웹 브라우저에서 불러오는 시간이 오래 걸려 샘플링 시간을 1/193 초로 작게 하였다. Start 버튼을 누르면 데이터를 가져오고 Stop 버튼은 데이터 가져오는 것을 중지시키고 현재까지의 데이터를 웹 브라우저에 나타내는 기능을 한다. 그림 9는 CGI가 실행되고 있는 웹 페이지이다.

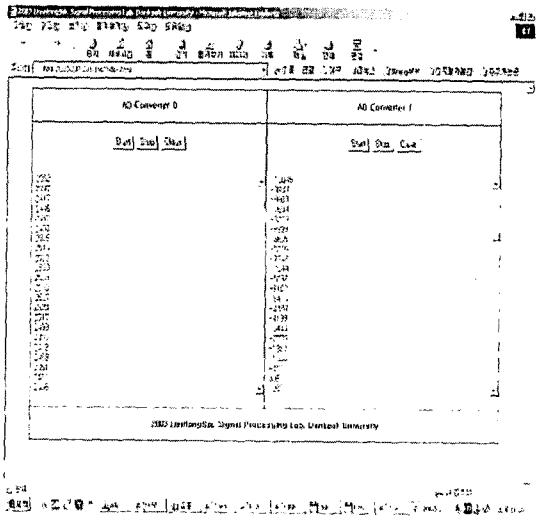

그림 9. CGI 실행 HTML 페이지

3. 결 론

전국적 국가 기간망으로까지 성장한 안정적인 인터넷을 이용해, 매달 고정비용이 발생되지 않은 유·무선 인터넷 기반 공장 및 설비의 실시간 감시 시스템의 구축이 가능해졌다. 이를 이용하여 사무실이나 원격지에서 이상 유무를 자동으로 통보받을 수 있고 상황 발생시 원격에서 통제가 가능하다. 웹 기반 감시 제어 시스템은 현장의 실시간 자료를 언제 어디서든지 손쉽게 웹 브라우저를 이용하여 감시 및 제어가 가능한 반면에 정보 시스템에서 보유하고 있는 풍부한 처리 기능을 구현하기가 어렵다는 단점을 가지고 있다. 즉, 웹 브라우저가 내장된 PC, PDA, WebPad, WebPhone, 휴대폰 등 다양한 단말기를 활용하여 쉽게 이용할 수 있는 모바일 기능에서 최대의 장점을 얻을 수 있는 것이다. 반면에 인터넷에 기반 한 응용 프로그램은 데이터베이스 및 각종 자료 수집 서버 등 인

터넷을 통해 얻은 자료를 처리하고 분석할 수 있는 강력한 자료 처리 능력을 보유하고 있다. 따라서 인터넷 기반의 응용 프로그램 환경과 웹 기반의 인터페이스 기능을 통합하여 제공하는 것이 요즘의 일반적인 시스템 구축 추세이다. 변화를 고려하는 것이 바람직한 것으로 사료된다.

본 연구는, 내장형 시스템의 운영 시스템 중 여러 면에서 장점을 가지고 있어 최근 대두되고 있는 리눅스 운영 시스템을 기반으로 한 ARM 프로세서에 웹 서버를 구현하고 ARM 프로세서에 연결된 외부 기기의 데이터를 받기 위하여 디바이스 드라이버를 구현하고, 웹 브라우저에서만 이루어지는 단방향 통신이 아니라 HTML을 이용해 서버와 사용자간의 양방향 통신을 할 수 있는 CGI를 이용하여 응용 프로그램을 작성하였으며 범용 웹 브라우저를 통하여 외부 기기의 상태를 감시하는 웹 기반 감시 시스템을 구현하였다.

본 연구는 한국과학재단 목적기초연구 (R01-2002- 000-00124-0(2002))지원으로 수행되었음.

참 고 문 헌

[1] 박영환, 임베디드 시스템 임베디드 리눅스, 사이텍 미디어, 2002

[2] 이동훈, "ARM 프로세서와 Linux를 이용한 마이크로 웹 서버 구현", 단국대학교 석사학위논문, 2003

[3] 이석원, 이진우, "내장형 리눅스를 이용한 웹 기반 원격 제어 시스템 구현", 대한전기학회 하계학술대회 논문집 D, pp2609-2611, 2003,

[3] Alessandro Rubini, *Linux Device Driver*, O'Reilly, 2000.

[4] Ed Tittel, Mark Gaither, Sebastian Hassinger and Mike Erwin, *CGI Bible*, IDG Books Press, 1997.

[5] Intel StrongARM SA-1110 Microprocessor, *Developer's Manual*, Intel, 2000.

[6] Michael Barr, *Programming Embedded System : In C And C++*, O'Reilly, 1999.

[7] Neil Matthew and Richard Stones, *Professional Linux Programming*, Wrox Press, 2001.

[8] Richard Stones and Neil Matthew, *Beginning Linux Programming*, Wrox Press, pp. 955-1156, 2000.

[9] Stepben Asbury, Jason Mathews, Selena Sol and Kevin Greer, *CGI How-To : The Definitive CGI Scripting Problem-Solver*, Waite Group Press, 1996.