임베디드 리눅스 기반의 영상전송서버 시스템 구현

Implementation of Image Transmission Server System using Embedded Linux

鄭 然 晟* · 南 副 熙**
(Yeon Sung Jung · Boo Hee Nam)

Abstract - In this paper, we performed the implementation of image transmission server system using embedded system that is for the specified object and easy to install at any places and move to wherever. Since the embedded system has lower capability than PC, we have to reduce the quantity of calculation and transmission. The image compression like JPEG, needs that the server calculates for making compressed image, makes the server carry the load. So we compresses the image at the server and transmit the codes to the clients connected, then the received codes from server are decoded and displayed at the clients. In this process to make the image compression and transmission effectively, we decrease the procedure as simple as possible to transmit the data in almost real-time.
We used the Redhat linux 9.0 OS at the host PC and the target board based on embedded linux. The image sequences are obtained from the camera attached to the FPGA board with ALTERA chip. For effectiveness and avoiding some constraints, we made the device driver. Generally the image transmission server is PC, but using the embedded system as a server makes the server portable and cheaper than the system based on PC.

Key Words : 임베디드 시스템, TCP/IP, 영상압축, 임베디드 리눅스

1. 서 론

임베디드 시스템이란 일반적으로 특정 업무를 수행하기 위해 하드웨어와 소프트웨어를 포함한 시스템이라고 할 수 있다. 그중에서 임베디드 리눅스는 리눅스 커널 자체가 태생적으로 모듈형식으로 이루어져 있어 상용 운영체제와는 달리 쉽게 최소, 최적화가 가능하다. 또한 오픈소스이기 때문에 소스 코드 자체가 공개되어있고 그 적용에 따른 비용 부담이 없어 세계적으로 많은 사용자가 사용하고 있다. 그리고 애플리케이션에 필수적인 TCP/IP, PPP 등의 네트워크 코드를 가지고 있다. 임베디드 장비를 위한 운영체제로 포팅하기 위해서 개발환경을 구축할 때는 크게 두 단계를 거친다. 첫 번째는 개발 툴 구축이고, 두 번째는 디버깅 환경 구축이다.

본 논문에서는 FPGA (Field Programmable Gate Array)에 부착된 카메라로부터 영상 정보를 얻은 후 Intel PXA255 CPU를 장착한 임베디드 타겟 보드로 만든 서버 시스템을 통해 클라이언트로 이미지를 전송하였다. 전송을 하기 위해서 TCP/IP 방식을 통해 정확하고 신뢰성있는 정보를 전송하였다.

* 準 會 員 : 江原 工大 通信multimedia工學 碩士課程
** 正 會 員 : 江原 工大 電氣電子情報通信工學付 敎授

임베디드 리눅스는 상용 운영체제가 아니기 때문에 장치 사용에 필요한 디바이스 드라이버는 자체적으로 개발하여 커널에 삽입하였다. 따라서 효율적인 자원을 관리를 하였다. 임베디드 시스템 자체의 크기가 작고 가격이 저렴하므로 사용자가 여러 분야에서 편리하게 사용할 수 있을 것이다.

2. 전체 시스템 구조

전체적인 시스템의 구조는 그림 1에 보인 것과 같다.

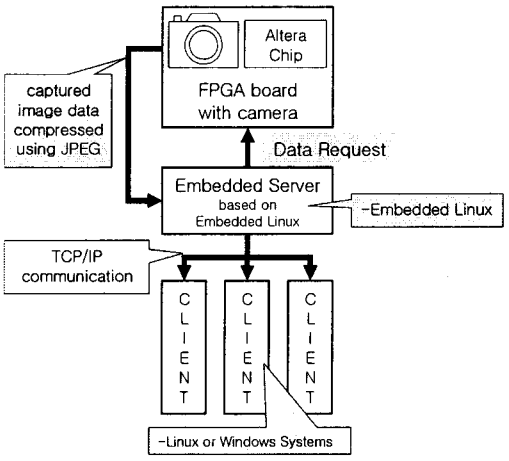

그림 1 전체 시스템 구조
Fig 1. The overall diagram

FPGA (Field Programmable Gate Array)에 부착된 카메라로부터 영상 정보를 얻은 후 PXA255 CPU를 장착한 임베디드 타겟 보드로 만든 서버 시스템을 통해 클라이언트로 이미지를 전송하였다.

3. 영상 전송

3.1 TCP/UDP

TCP/IP (Transmission Control Protocol / Internet Protocol)는 한 노드에서 다른 노드로의 신뢰성 있는 전송을 보장한다. 이는 연결 지향형 프로토콜이며 데이터가 전송되기 전에 장비 간 연결을 해 놓는다. 비록 UDP가 빠르고 TCP보다 적은 양의 오버헤드를 갖지만 확실한 데이터 전송을 보장받기 위해 우리는 TCP 방식을 사용하였다.

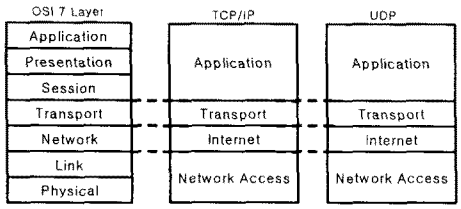

그림 2 TCP/UDP 계층
Fig 2. TCP/UDP Layers

3.2 리눅스 환경 소켓프로그래밍

소켓은 PC를 거대 네트워크에 접속할 수 있도록 해주는 역할을 담당한다. 소켓은 일종의 네트워크 API (Application Programming Interface)로 다른 TCP/IP 프로그램과 프로토콜 간의 통신을 쉽게 할 수 있도록 설계되었다. 이를 통해 TCP/IP를 이용한 통신 애플리케이션은 표준 인터페이스를 사용하게 되는 것이다.

4. 개발 환경 구축

일반적으로 리눅스를 임베디드 장비를 위한 운영체제로 포팅하기 위해서 개발 환경을 구축할 때는 크게 두 단계를 거친다.

4.1 개발 툴 구축

그 첫 번째는 개발 툴 구축이다. 운영체제를 포팅할 때는 개발을 위한 개발 플랫폼과 타깃 호스트가 구별된 경우가 대부분이다. 그 이유는 우선 포팅의 목표 아키텍처를 가진 타깃 호스트는 임베디드 장비를 위한 프로세서를 사용하므로 개발용 장비로 쓰기에는 그 프로세싱 능력이 부족하고 그 프로세서에 운영체제가 올라가 있지 않은 경우가 많기 때문이다. 따라서 개발 플랫폼은 강력한 프로세싱 능력과 안정적인 운영체제를 가진 PC나 유닉스 워크스테이션에 구축하고 타깃을 위한 바이너리를 만들어내는 크로스 컴파일러 툴 세트를 설치한다. 이 구조는 그림 3에 나타내었다.

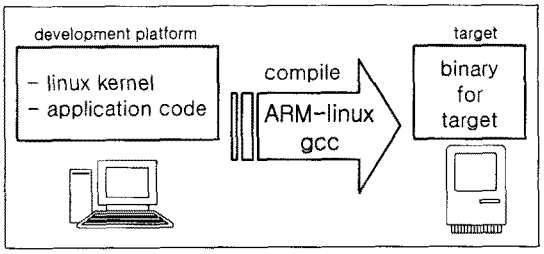

그림 3 크로스 개발 환경
Fig 3. Cross Compile Environment

4.2 디버깅 환경 구축

개발 툴 구축의 두 번째 단계는 디버깅 환경구축이다. 임베디드 운영체제를 위한 디버깅은 소프트웨어 디버깅과 하드웨어를 이용한 디버깅 두 가지로 구분되는데 소프트웨어 디버깅은 소스 레벨에서 브레이크 포인트나 특정 변수 값들을 출력해 보면서 디버깅하는 일반적인 방법이고 하드웨어 디버깅은 프로세서에 정의된 물리적 인터페이스를 통해 프로세서를 외부에서 완전히 컨트롤하여 내부 상태를 관찰해 가면서 디버깅하는 방법이다. 리눅스 커널의 경우 일단 리눅스가 올라 간 다음부터는 커널과 애플리케이션 디버깅을 RS-232나 이더넷 인터페이스로 할 수 있다는 특징이 있다. 즉, 하드웨어 없이도 디버깅 플랫폼을 다로 구성할 수 있다는 의미이며 이는 임베디드 장비 개발에 큰 도움이 된다. 리눅스 상에서 디버깅 환경 구축은 그림 4와 같다.

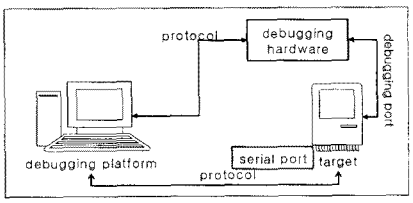

그림 4 디버깅 환경
Fig 4. Debugging Environment

4.3 디바이스 드라이버 개발

디바이스 (Device)란 컴퓨터 이외의 주변 장치를 말하고, 드라이버 (Driver)란 이러한 하드웨어 장치를 제어하고 관리하는 방법을 운영체제에 알려주는 응용 프로그램을 뜻한다. 즉, 디바이스를 구동하기 위해서는 디바이스 드라이버가 필수이다. 따라서, 리눅스 커널에서 제공해주는 드라이버가 없으면 만들어주어야 한다. 본 논문에서는 리눅스 커널에서 제공해주는 드라이버가 실질적으로 특수 목적을 위해 사용되는 시스템에 대한 필요 없는 기능을 포함하기 때문에 자원을 낭비하지 않게끔 직접 개발하였다. 리눅스는 상용 운영체제와는 달리 공개 소프트웨어이고, 모듈 개념을 통해 직접 사용자가 커널을 수정할 수 있다. 따라서 본 논문에서는 디바이스 드라이버 자체 제작에 일단 초점을 맞추었다. 그림 5은 드라이버와 커널 간의 관계를 나타낸다.

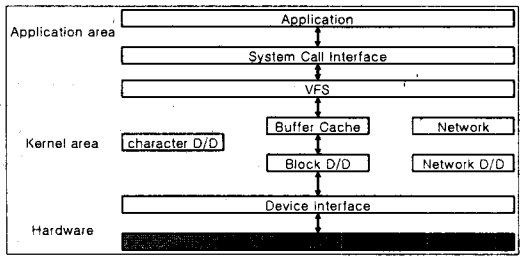

그림 5 드라이버와 커널의 관계
Fig 5. Relationship between Driver and Kernel

4.4 디바이스 드라이버 개발 방법

디바이스 드라이버는 일반적인 파일을 다루는 것과 흡사하다. 드라이버 장치 열기, 쓰기/읽기, 닫기의 과정을 거친다. 파일과 다른 점은 개발하는 과정에서는 장치를 등록하거나 해제하는 과정이 추가되는 데 이 과정을 모듈 등록 및 해제라고 부른다.

4.5 GPIO (General Purpose Input/Output)

PXA255 CPU의 GPIO핀 즉, 범용 입출력 핀은 SA1110과 유사하지만 확장되었다. 이 경우 81개를 갖으며 부팅 초기 시 전원 절약을 위해 모두 입력 상태가 된다. 일반적인 프로세서에 달린 인터럽트 핀과는 다르게 여러 가지 용도로 쓰일 수 있다. 따라서, 입출력뿐 아니라 인터럽트의 기능을 포함한다. 우리는 이 핀을 이용하여 이미지 정보 습득의 시작과 끝을 나타내는 인터럽트를 사용하였다. GPIO핀은 레지스터 설정을 통해 쉽게 제어할 수가 있다.

4.6. 카메라 디바이스 드라이버 개발

연결된 ccd 카메라에 전원이 들어가면 이미지를 보드로 전송해준다. 우리는 이미지 데이터를 담을 버퍼를 설정하고 버퍼가 꽉 차게 될 경우와 버퍼 사용을 중지시키는 경우의 인터럽트를 위해 2개의 GPIO핀을 사용하였다. 애플리케이션에서 장치를 열고 데이터 전송요청신호를 주면 드라이버는 인터럽트를 걸어 데이터를 보드에 모두 전송할 때까지 카메라로부터의 입력을 받지 않는다. 전송이 끝나면 인터럽트 루틴에서 빠져나와 원래 하던 작업을 하게끔 하였다.

4.7 서버 프로그램

일반적인 소켓 프로그램을 통해 카메라로부터 받은 이미지 데이터를 접속 중인 클라이언트에 전달해준다. TCP 방식을 이용한 소켓을 생성하였고 여러 클라이언트가 동시에 접속할 수 있도록 하였다. 전송 전에 리눅스 공개 패키지 중 JPEG 변환 패키지를 이용하여 JPEG 변환을 하여 전송 데이터의 양을 줄였다.

다중으로 클라이언트가 접속한 경우 정확한 데이터의 전송을 위해 폴링 방식을 이용하였다. 같은 데이터를 쓰레드를 이용하여 전달할 경우 오류가 생길 가능성이 높기 때문이다. 이미지 데이터의 전송을 마치면 개발보드에 설치되어 있는 TFT-LCD 화면에 프레임 버퍼를 이용하여 보여준다.

4.8 클라이언트 프로그램

클라이언트는 MFC를 이용하여 전송받은 이미지를 윈도우즈 화면에 보여주는 방식으로 구성하였다. 서버와 마찬가지로 TCP를 이용한 소켓을 통해 데이터를 전송받고 JPEG 이미지를 복호화하여 화면에 보여준다.

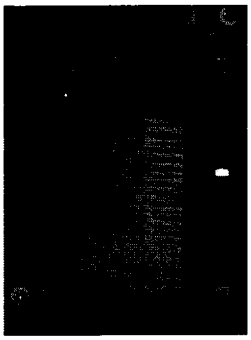

그림 6 서버 : TFT-LCD
Fig 6. Server

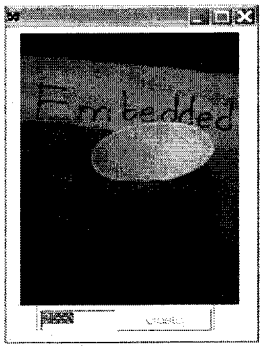

그림 7 클라이언트
Fig 7. Client

5. 결론

본 논문에서 임베디드 시스템을 이용하여 영상을 전송하기 위한 방법을 제시하였다. 카메라로부터 입력 받은 영상을 접속된 모든 클라이언트에게 인터넷을 통해 전해주도록 구현하였다.

WinCE가 아닌 임베디드 리눅스를 사용함으로써 디바이스를 마이크로소프트에 피동적이지 않게 사용할 수 있었고 종전에 서버로 주로 사용되던 PC에 비해 싼 임베디드 시스템을 이용하기에 많은 개발비용을 줄일 수 있었다.

두 달이라는 짧은 연구 기간이었기에 아직 적용하지 못했지만, 추가로 여러 가지 디바이스를 이용하여 움직임 감지나 얼굴인식 등의 알고리듬을 적용, 무선 인터넷의 활용을 통해 더욱 훌륭한 저가의 원격 감시 장비를 만들 계획이다.

감사의 글

본 연구는 2004년도 강원대학교 BK21 (Brain Korea) 지원에 의하여 이루어진 연구로서, 관계부처에 감사드립니다.

참 고 문 헌

[1] Huins Company Development Department, "Intel PXA225 and Embedded Linux Application", Huins Company, 2004

[2] Hwa-jong Kim, "Computer Network Programming - Unix Version", Hong-reung, 2004

[3] Tae-geun Oh, Boo-hee Nam, "Remote monitoring of the moving target using fuzzy-controlled camera", Computational Intelligence for Modelling, Control and Automation (CIMCA'03) Proceedings pp.857-865, Feb. 12-14, 2003, Vienna, Austria.