2001년도 대한전기학회 하계학술대회 논문집 2001.7.18-20

리눅스 임베디드 시스템을 이용한 Caller ID 구현

김형배 이석원 남부희
강원대학교 BK21 전기전자정보통신공학부

Caller ID Implementation Using Linux Embedded System

Hyoung Bae Kim and Seok Won Lee and Boo Hee Nam
BK21 Dept. of Electrical and Computer Engineering, KANGWON NATIONAL UNIVERSITY

Abstract - 본 연구에서는 리눅스를 임베디드 시스템에 포팅하기 위한 기술을 조사하고 하드웨어를 선정하여 리눅스를 포팅하였다. 하드웨어는 Intel 사의 StrongARM SA-1110를 탑재한 Evaluation board를 사용하였다. 개발 환경은 Host computer와 Assabet board 상에 RS-232C를 이용하여 프로그램 다운로드할 수 있는 환경을 구축하였다. StrongARM를 위한 Cross Compiler를 만들고, StrongARM의 하드웨어 메모리 맵을 이용하여 Linux의 메모리 맵을 설정한 후 리눅스를 컴파일하였다. 이를 RS-232C를 통해 Assabet board에 다운로드하여 리눅스를 탑재한 임베디스 시스템을 구현하였다. UCB1300 telecom codec의 device driver를 개발하여 커널에 적재하고 이를 사용하여 전화기로부터 들어오는 신호중 CID 신호를 해석하여 발신자 번호를 출력하는 시스템을 구현하였다.

1. 서 론

Caller ID(발신자 정보 표시 장치)는 전화를 받을 사람이 전화를 거는 사람의 신원을 미리 확인하여 수신여부를 결정할 수 있도록 하는 서비스로, 일방적으로 통화를 강요 받는 불평등을 개선할 수 있는 서비스이다. 근간에 사회적 문제가 되었던 스토킹(stocking)을 해결하는데 좋은 방법 중에 하나라고 할 수 있다. 뿐만 아니라 Caller ID 서비스(발신자식별서비스) 도입에 발신자 위주의 통신환경이 수신자 위주로 전환되면서 수신자의 권리 보호, 텔레마케팅 · 데이타베이스마케팅 산업 활성화라는 장점이 크게 부각되기 때문에 텔레마케팅 업계의 시선이 집중되고 있다.

지난 87년 미국에서 CLASS(Custom Local Area Signalling Services)를 제공하면서부터 Caller ID 서비스 시대를 열었다. 현재 미국내 대부분의 전화사업자들이 Caller ID 서비스를 기본으로 전화번호 소유주의 주소까지 안내하는 각종 패키지 서비스를 개발, 제공하고 있고 유럽에서도 91년 BT에서 「Caller Display」서비스를 도입했으며 DT, FT 등으로 서비스가 확산됐다. 일본에서는 98년 NTT에서 「넘버디스플레이」를 전국에 상용서비스 했고, 서비스 초기 단말장비의 부족현상을 초래할 정도로 인기를 끌었으나 우리 나라 에서는 개인의 사생활 침해를 우려해서 시행이 미뤄지다가 올 4월부터 시행하는 것을 정보통신부에서 발표를 하였고 현재는 서비즈 중이다. 그러나 사생활 침해라고 생각하던 것과는 달리 한국통신과 동아일보에서 공동 여론조사 한 결과 84.52%가 CID서비스에 찬성하면서 많은 관심을 보이고 있다.

실제로 많은 Telecommunication 업계에서도 Caller ID 기능을 갖는 전화기를 선보이고 있으며 Caller ID 기능 뿐만 아니라 통화중 걸려온 전화도 발신자 정보를 표시하는 통화중대기-Caller ID 서비스, 발신자가 자신의 정보를 수신측에서 표시하지 않는 서비스, 익명 전화 수신거부 서비스등을 제공하고 있다. 이런 기능을 구현하기 위해 모뎀을 사용하지 않고 소프트모뎀의 기능을 구현하였다. 기존의 하드웨어 모뎀은 일련의 과정을 모뎀 속의 메인 칩셋에서 처리했다. 그러나 소프트모뎀은 이 과정을 모두 CPU에서 처리한다. 그러므로 가격이 모뎀보다 저렴하다는 장점이 있다. 본 연구에서는 Philips 사의 UCB1300을 이용하여 소프트모뎀의 기능을 구현하고, 리눅스를 탑재한 임베디드 시스템에서 Caller ID 표시 서비스를 구현하였다.

2. 본 론

2.1 개발환경 구축

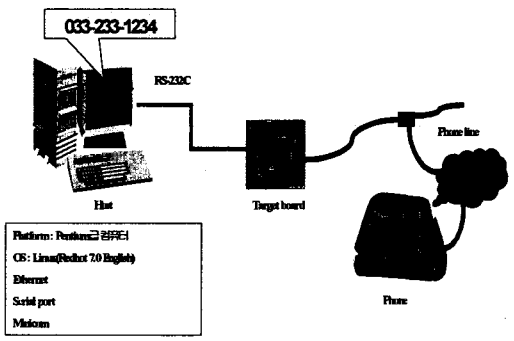

그림 1 시스템 구성도

호스트에서 먼저 크로스 개발(Cross-Development) 환경 구성을 하고 부트 로더 구성, 커널 수정, 실행 커널 이미지와 간단한 애플리케이션을 작성한 후, RS-232C를 이용하여 Target board로 다운로드하는 환경을 구축하였다.

2.1.1 하드웨어 선정

임베디드 프로그래밍에서 가장 중요한 것이 바로 타겟 CPU를 선정하는 일이다. 기존 업체들의 포팅의 사례가 많고 주변에서 구하기 쉬워야하기 때문이다. 삼성 KS32C50100은 ARM7 계열의 CPU로 이더넷 컨트롤러와 UART를 내장한 32비트 CPU이다. 현재 네트워크 프

린터나 웹폰 등에 널리 활용되고 있는 실정이나 MMU(Memory Management Unit)이 없으므로 software로 구현해야한다는 단점이 있다. 그러므로 MMU를 포함하고 Arm 계열보다 강력한 스피드를 내는 StrongArm 계열을 선택하고 evaulation board로는 Intel Assabet (SA-1110 evaluation) board를 사용하였다.

국내 뿐만 아니라 국외에서도 StrongArm에 리눅스를 포팅한 사례가 많으므로 개발기간을 단축할 수가 있기 때문이다. Empeg라는 mp3 플레이어도 있으며, Itsy라는 Compag에서 만든 PDA역시 StrongARM에 리눅스를 포팅하여 만든 제품이고, YOPY(GMATE 사에서 만든 PDA)역시 StrongARM 시스템에 리눅스를 포팅하였으며, MP3 및 MPEG player, 웹 브라우저, 이메일 기능 등을 가진다. 다음 그림은 SA1110 Development Board의 block diagrm이다.

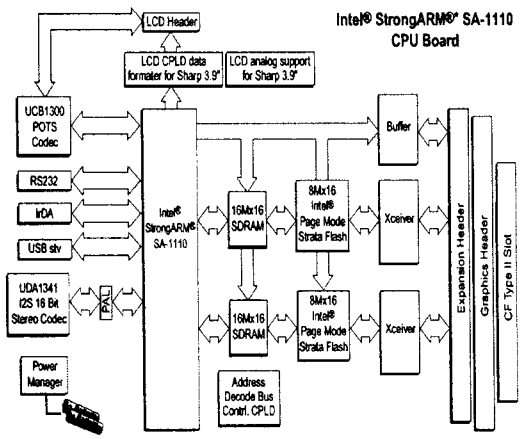

그림 2 SA-1110 Development Board Block Diagram

2.1.2 운영체제 선정

과거의 임베디드 시스템들은 주로 간단하고 단순한 순차적인 작업에 관련되었기 때문에 굳이 OS를 사용할 필요가 없었다. 하지만 최근의 임베디드 시스템 분야에서는 그 시스템 자체가 커지게 되고, 네트워크나 멀티미디어가 시스템에 기본으로 자리잡으면서 임베디드 시스템이 해야 할 일들도 많아지고 복잡해 졌기 때문에 순차적인 프로그램 작성이 매우 어렵게 되었다. 따라서 임베디드 시스템에서도 운영체제의 개념이 필요하게 되었다. 본 연구에서는 산업 현장에서처럼 실시간적인 요소가 필요없으므로, 멀티태스킹을 지원하며 안정적이고 오픈 소스에 라이센스 비용이 필요없는 임베디드 리눅스를 OS로 사용하였다.

2.1.3 크로스 컴파일러 구축

호스트 시스템에 Target Board용 개발을 가능하게 하는 크로스 개발 환경을 구성하여야 하였다. 어떤 경우는 완벽한 크로스 개발환경을 설정하는데 어려움이 있을 수도 있는데 여기서 필요한 일은 적절한 배포본의 선택, 적합한 설정 및 재컴파일을 검토해 보는 것이다. 본 시스템에서는 ARM 용 크로스 컴파일러를 만들기 위하여 호스트 컴퓨터에 다음과 같이 구성하였다.

<호스트 컴퓨터 구성>

- OS : RedHat Linux 6.1
- Platform : X86 Processor 계열
- Tools : Binutils 2.9.5.0.29, Gcc 2.95.2,
  Linux Kernel 2.4.0-test5 Glibc 2.1.3
- Utility : vi, bash, dump, Monitor Program 등

2.1.4 커널 수정

커널 수정은 포팅의 가장 핵심이며 리눅스 포팅 과정에서 가장 중요하다. 리눅스를 새로운 보드에 이식할 때 커널 소스를 수정하고 새로운 코드를 수정하는 일이 반드시 필요하다. 그러나 리눅스 소스 트리구조 내에서 소스가 조직화가 잘 되어 있어 이식성이 높게 디자인되어 있으므로 수월하다.

기본적으로 모든 보드에 의존적인 코드는 리눅스가 지원되는 프로세서라 할지라도 수정되거나 조정되어야 한다. 새로운 요구사항이 나타나고 버그 수정을 해야 할 때면 이러한 변경이 필수적이다. 소스 변경의 핵심으로 보드 의존적인 부분으로 제한되어야 한다는 점이 요구된다. 그러나 /usr/src/linux/arch/ppc 디렉토리 소스 내용 수정이 대부분이다. 그리고 새로운 하드웨어에 적용할 코드와 메모리 맵핑 같은 새로운 기능 추가와 수정이 필요하다. 그러나 대부분의 CPU나 Board를 위한 커널 수정(패치)파일들이 있으므로 커널은 최신 커널의 소스에 패치 파일을 적용하면 되었다.

본 연구에서는 커널 소스는 2.4.0-test5 버전, Arm patch로는 patch-2.4.0-test5-rmk1을 Assabet board patch로는 diff-2.4.0-test5-rmk1-np1을 사용하였다.

2.1.5 리눅스 포팅 및 Target Board로의 다운로드

Host상에 있는 Kernel 소스를 Target Board에 맞는 Arm용 크로스 컴파일러로 컴파일한 후 나온 커널이미지를 target board에 다운로딩한 후 다운로딩한 커널이미지로 리눅스를 실행하였다.

먼저 Host PC에서 Arm용 cross comiler를 이용하여 커널을 컴파일하면 zImage가 생성되면 minicom을 통해 target 보드의 bootloader를 실행한 후 zImage와 ramdisk를 angelboot를 이용하여 다운로딩하였다.

2.2 UCB1200 디바이스 드라이버 개발 및 테스트

전화선으로부터 들어오는 신호를 받기위해 사용하는 UCB1300 telecom codec을 위한 디바이스 드라이버를 개발하였다. UCB1300의 audio driver를 수정하여 telecom 드라이버를 개발하였다. 이미 생성한 커널에다가 디바이스 모듈을 포함하여 Target Board에서 리눅스를 실행하였다. DTMF tone 으로 전화를 걸 번호의 데이터를 만들고 UCB1300 telecom codec에다가 써줌으로써 전화가 걸리는 것을 확인하였고, 전화 벨을 기다리다가 전화 벨이 울리면 off-hook을 하여 통화내용을 웨이브 파일로 저장하는 애플리케이션으로 장치를 테스트하였다. 다음은 UCB1300을 이용하여 전화 거는 과정을 그린 그림이다.

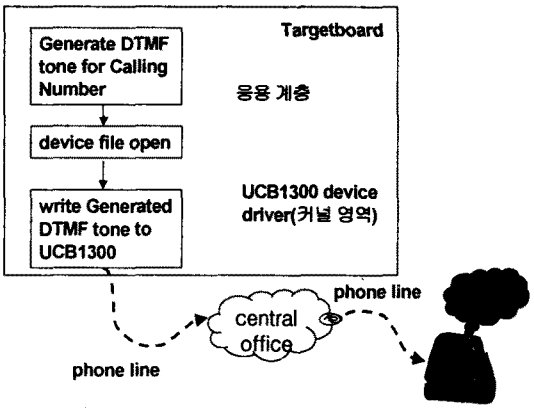

그림 3 Procedure of Calling Up

2.3 Caller ID 구현

2.3.1 Caller ID 란?

Caller ID는 caller 의 인증이외에 많은 정보를 포함한다. CID의 두 종류로 SDMF(Single Data Message Format) 와 MDMF(Multiple Data Message Format)방식이 있다. 두 방식 모두 calling number, 날짜 그리고 시간 정보를 포함하고 있으나 MDMF 방식에는 caller의 이름까지 포함되어 있다.

CID 정보는 binary FSK 방식을 이용하여 1200 bps 의 전송률을 가진다.

두 가지의 주파수를 바이너리의 상태를 표시하는데 1200 Hz일 경우는 logic 1을 2200 Hz일경우는 logic 0를 나타낸다. 그리고 데이터는 첫 번째 Ring과 두 번째 Ring사이에 보내게된다. 첫 벨이 울린후 최소 500ms의 시간이 지연되야 한다. 그리고 프리앰블로서 250ms 동안 0과1을 반복하는 문자열 300개를 보내어 동기를 이룬다. 그리고 데이터를 보내게 된다. 메시지의 각 문자들을 8bit로 구성된다. start bit 로서 전송이 시작되며 각 문자마다 10bit의 stop bit를 가진다. CID의 정보에 이어 에러를 감지하기위한 checksum 데이터를 보낸다. 다음은 SDMF의 메시지를 보여주는 그림이다.

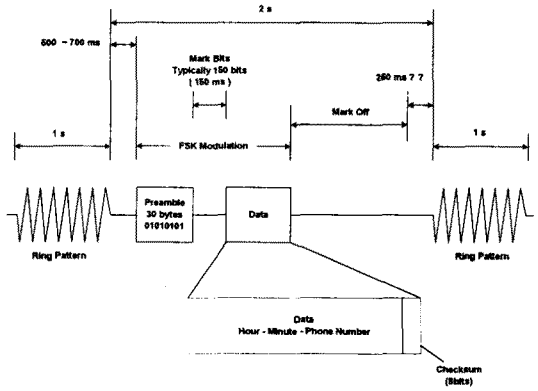

그림 4 Transmission CID Information Between the First Ring and Secod Rings

2.3.2 Caller ID Decoder 구현

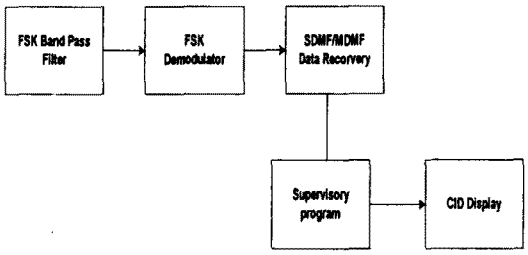

그림 5 Decoder Software Module

전화기로 들어오는 신호를 위의 과정을 거쳐서 CID 정보를 얻어낼 수 있다. FSK Band Pass Filter는 대역외 신호 방식 Filter를 포함하고 FSK Demodulator는 아날로그 데이터를 바이너리 데이터로 변환한 후에 SDMF/MDMF Data Recorvery에서 start bit와 stop bit를 제거하여 메시지를 복원하여 CID 정보를 얻을 수 있었다.

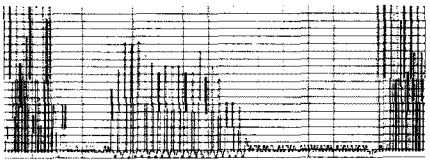

그림 6 FSK Delivery Between the First Ring and Second Ring

다음은 첫 번째 벨과 두 번째 벨 사이의 수신한 FSK를 출력한 그림이다.

첫 번째 벨이 울리면 전화기로 오는 데이타를 버퍼에 저장하여 Decoder Software Module을 거쳐서 CID 의 정보를 얻을 수 있다.

3. 결 론

본 연구에서는 리눅스를 탑재한 Intel Assabet (SA-1110 evaluation) board와 UCB1300 telecom codec을 이용한 CID 표시기를 구현하였다. 연구에서는 Host PC상의 모니터에 CID의 정보를 표시하였으나 추후에는 Assabet 보드상의 LCD 화면에 CID 정보를 출력하게 할 것 이다. 그리고 통화중대기-발신자 정보 표시서비스, 발신자정보표시방지 서비스, 익명전화 수신거부 서비스의 기능을 구현할 것이다. 또한 UCB1300 칩을 이용한 soft modem의 구현도 가능하다.

[참 고 문 헌]

[1] The ARM Linux Project, URL : http://www.arm.linux.org.uk/
[2] The Linux Kernel , URL :http://www.linuxhq.com/guides/TLK/tlk.html
[3] SA-1100 Developers Manual URL:http://sospl6.cs.washington.edu/homes/awong/sa1100dev.pdf
[4] UCB1200Datasheet URL:http://www.semiconductors.philips.com/acrobat/datasheets/UCB1200_2.pdf
[5] 리눅스 디바이스 드라이버, 김인성, 류태중 역, 한빛 미디어