제 18 회 한국정보처리학회 추계학술발표대회 논문집 제 9 권 제 2 호 (2002.11)

리눅스 기반의 차량용 동영상 재생기의 구현

김민호[0], 양효춘, 장성균, 고정국
동명정보대학교 컴퓨터공학과
e-mail : kho51@hanmail.net, kamio@korea.com, mrjang26@hanmail.net, jgkoh@tit.ac.kr

Implementation of Linux Based Movie Player for Vehicles

Min-Ho Kim[0], Hyo-Chyun Yang, Sung-Gyun Jang, Jeong-Gook Koh
Dept. of Computer Engineering, Tongmyong University of Information Technology

요 약

최근 라우터나 Set-Top Box 등 리눅스를 기반으로 하는 시스템의 구현 사례가 많이 등장하고 있다. 또한, 차량에 A/V 기기를 장착하여 주행 중에 비디오를 감상하는 추세도 증가하고 있다. 본 논문에서는 이러한 추세에 부합되도록 리눅스를 이용하여 차량용 동영상 재생기를 개발하고 그래픽 사용자 인터페이스를 추가하여 사용자들이 편리하게 사용할 수 있도록 하였다. 구현된 동영상 재생기를 차량에 탑재하여 기능을 시험한 결과 동영상 재생기의 효용성을 확인할 수 있었다.

1. 서론

리눅스는 90 년대 중반부터 인터넷과 함께 널리 보급된 유닉스 계열의 운영체제이다. 리눅스는 소스 코드가 공개되어 있기 때문에 용도에 맞게 커널이나 유틸리티들을 수정하고 시험해 볼 수도 있다. 최근에는 임베디드 시스템 시장에서도 리눅스를 활용한 다양한 제품들이 출시되고 있다. 임베디드 시스템은 범용 시스템들과 달리 특정 용도에 적합하게 설계되고 최적화시켜 구현된다. 따라서, 리눅스를 임베디드 시스템에 적용하려면 적용 분야에 적합하도록 불필요한 부분은 제거하고 필수적인 기능만 포함하도록 구성해야 한다. 한편, 최근에는 차량에 다양한 A/V 기기를 장착하여 운행 중에도 오디오나 비디오를 감상하는 추세가 점차 일반화되고 있다.

본 논문에서는 이러한 추세에 부합되도록 차량에 장착할 수 있는 동영상 재생기를 리눅스를 기반으로 구현하였다. 또한, 동영상 재생기의 편리한 활용이 가능하도록 Frame Buffer 와 QT/Embedded 라이브러리를 이용하여 그래픽 사용자 인터페이스도 개발하였다.

본 논문의 구성은 다음과 같다. 2 장에서는 차량용 동영상 재생기 구현에 관련된 기술들을 살펴보고, 3 장에서는 차량용 동영상 재생기의 기능 설계에 대해 기술한다. 4 장에서는 차량용 동영상 재생기의 구현 내역을 서술하고, 마지막으로 5 장에서는 결론과 향후 연구과제를 제시한다.

2. 관련기술

본 장에서는 차량용 동영상 재생기 구현에 관련된 다양한 기술들을 살펴본다.

2.1 리눅스 커널

리눅스 커널은 PDA 에서 메인 프레임에 이르기까지 다양한 플랫폼에서 실행된다. 리눅스는 이러한 확장성을 지원할 수 있도록 커널을 다양하게 설정할 수 있다. 커널을 설정하려면 커널 소스의 루트에서 make config 나 make menuconfig, 또는 make xconfig 라는 명령을 입력하여 표준 커널 설정 루틴을 실행한다. 다양한 옵션들을 선택할 수 있으며, 필요에 따라서는 실행시간에 적재될 수 있는 동적 모듈로 컴파일할 수도 있다[1].

2.2 Frame Buffer

Frame Buffer 는 video display 장치를 메모리처럼 접근하도록 하여 저수준의 구동기(driver)와 무관하게 동일한 API 로 video control 이 가능하도록 하는 장치이다. 이러한 장치 구동기를 사용하는 이유는 복잡하고 덩치가 큰 X 윈도우를 가용 자원이 적은 임베디드 시

스템에 적용하기가 곤란하기 때문이다. Frame Buffer driver 는 초기화 루틴, read/write/ioctl 함수, Frame Buffer 정보 구성 및 관련 함수 등으로 구성되어 있다[6].

2.3 initial RAM 디스크(initrd)

리눅스의 부트 이미지를 작게 만들기 위해서는 initrd 파일 시스템을 활용한다. 리눅스의 설치 절차를 간편하게 하기 위해 고안된 initrd 파일 시스템은 커널이 저장 장치에서 메모리로 적재되듯이 소형 파일 시스템을 메모리에 적재할 수 있도록 한다. initrd 파일 시스템은 gzip 으로 압축되어 있기 때문에, 커널은 실행시간에 initrd 파일 시스템의 압축을 풀어서 RAM 디스크에 적재한다. initrd 파일 시스템을 사용하면 필수적인 드라이버와 부트 스크립트만을 포함한 작고 압축된 파일 시스템이 커널과 함께 부팅 장치로부터 적재된다. 그 다음에 커널은 initrd 파일 시스템의 압축을 풀고 " /linuxrc" 파일을 실행시킨다. 이 스크립트는 부트 과정을 완료하기 위해 적합한 드라이버를 적재한 뒤 종료된다. 일단 스크립트가 종료되면 initrd 파일 시스템은 언마운트(unmount)되고 메모리는 반환된다. 그리고 이후에 실제 초기화 과정이 계속 수행된다. 그러나 임베디드 시스템에서는 initrd 파일 시스템을 종료시킬 이유가 없다. 임베디드 어플리케이션이 필요로 하는 모든 소프트웨어들을 initrd 파일 시스템에 상주 시킬 수 있다[1].

2.4 QT/Embedded

KDE 프로젝트의 QT 라이브러리를 기본으로 하여 개발된 임베디드 시스템용 GUI 라이브러리이다. X 윈도우 없이 리눅스 커널에서 제공하는 Frame Buffer 를 이용하여 그래픽 장치에 직접 접근하는 방식을 사용하고 있다. 멀티 플랫폼을 지원하기 때문에 플랫폼이 변경되더라도 소스를 수정해야 하는 번거로움이 적다. 또한, 250 여 개의 C++ 클래스를 지원하며 각 클래스에는 GUI 를 위한 함수와 템플릿 기반의 collection, serialization, 파일, I/O 장치, 디렉토리 관리 및 다양한 종류의 API 를 지원한다[2].

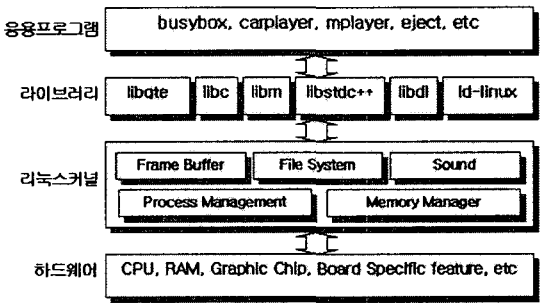
Signal 과 Slot 은 QT 에서 제공하는 이벤트 처리 모델로 이것을 사용하면 객체간의 통신이 가능하고, 복잡한 Callback 함수 작성 부담도 줄일 수 있다[7,8].

3. 차량용 동영상 재생기의 기능 설계

본 장에서는 차량용 동영상 재생기의 기능 설계 내역과 동영상 재생기의 구동 방식을 설명한다.

3.1 시스템 구성

차량용 동영상 재생기는 [그림 1]과 같이 하드웨어, 리눅스 커널, 라이브러리, 응용 프로그램의 4 계층으로 구성되어 있다. 하드웨어는 CPU 나 Graphic Chip 과 같은 실제적인 장치들로서 상위 계층인 리눅스 커널이 이들 장치를 구동시킨다. 응용프로그램이 필요로 하는 기능들을 미리 정의해 놓은 라이브러리는 리눅스 커널과 응용프로그램의 중간에 위치하며 응용프로그램이 활용할 수 있도록 미리 컴파일해 놓은 binary 파일이다. 예를 들어, libqte 는 QT/Embedded 의 라이브러리 파일로서 다양한 GUI Widget 들이 미리 정의되어 응용프로그램의 GUI 를 쉽게 구현할 수 있다. libqte 라이브러리는 리눅스 커널의 Frame Buffer 를 이용하여 콘솔에 그래픽을 표현한다. 응용프로그램은 libqte, libc, libm, ld-linux 등의 라이브러리를 이용하여 필요한 작업을 수행한다.

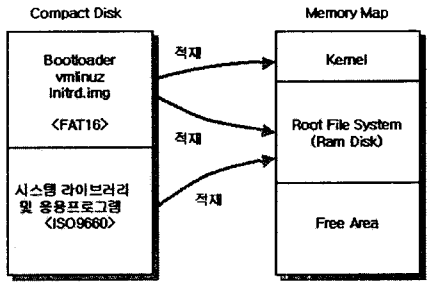


[그림 1] 차량용 동영상 재생기의 구성도

3.2 차량용 동영상 재생기의 구동 방식

본 논문에서는 리눅스 커널과 라이브러리, 재생기 프로그램을 부팅용 CD-ROM 에 담아서 차량용 동영상 재생기를 구현하였다. CD-ROM 은 쓰기가 불가능하여 루트 파일 시스템으로 사용할 수가 없기 때문에, RAM 을 가상 디스크로 사용하여 루트 파일 시스템을 구성하는 initial RAM 디스크(initrd)를 활용하였다. 부팅용 CD-ROM 은 부팅 영역과 데이터 영역이 구분되어 있으며, 부팅 영역에는 부트로더와 커널 이미지, initrd 이미지를 수록하고 있다.

부팅과정은 [그림 2]와 같다. 처음에는 부트로더가 실행되어 커널과 initrd 이미지를 메모리에 적재한다. 커널과 initrd 이미지가 적재되면 커널 이미지의 압축을 푼다. 커널이 실행되면서 하드웨어를 인식하고 램디스크 공간을 만든 후 initrd 이미지의 압축을 해제하여 루트 파일 시스템으로 마운트한다. 그 다음에는 응용프로그램을 수행하는 " /linuxrc" 스크립트를 실행한다. CD-ROM 의 ISO9660 파일 시스템으로부터 응용프로그램과 QT 라이브러리, 시스템 라이브러리, 폰트 파일을 루트 파일 시스템에 복사한다. 복사가 끝난 후에는 동영상 재생기 프로그램이 자동적으로 실행된다.

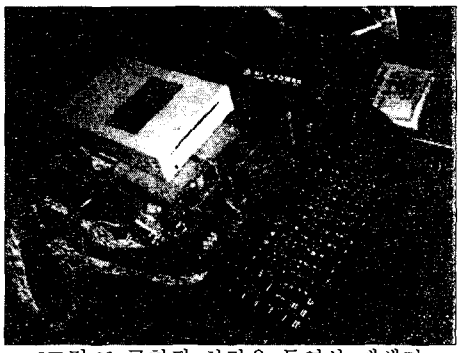


[그림 2] 동영상 재생기의 구동 과정

4. 차량용 동영상 재생기의 구현

4.1 구현 환경

호스트 컴퓨터는 Pentium-3 933Mhz CPU, 256MB 메모리, 40GB HDD, CD-ROM Writer를 장착하였다. 운영체제는 RedHat 7.3 이며 GUI 개발용으로 QT/Embedded 3.0.5 를 사용하였다. 또한, 작업 내용을 신속히 테스트하기 위해 디스크 파티션 하나를 개발용으로 사용하였다. 본 논문에서는 구현된 동영상 재생기를 차량에 장착할 수 있도록 셀러론 500MHz CPU, 256MB 메모리, DVD-ROM, 무선마우스, 무선키보드 등을 [그림 3]과 같이 소형 케이스에 담았으며, 동영상 재생기의 전원은 차량용 파워 인버터를 사용하여 공급하였다.



[그림 3] 구현된 차량용 동영상 재생기

4.2 커널 생성

차량용 동영상 재생기 구현에 Redhat 7.3 의 커널 버전 2.4.18 을 사용하였고 Frame Buffer 와 initial RAM 디스크, 그리고 Ext2, ISO9660 파일시스템을 사용하도록 설정하였다. 커널 컴파일 후 생성된 커널 이미지의 크기는 669KB 였다.

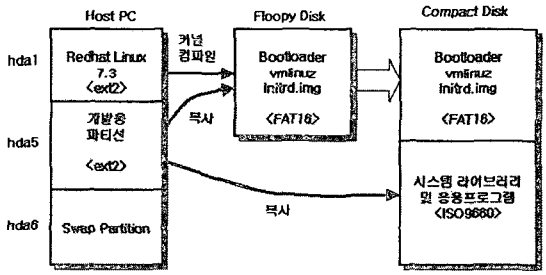
4.3 initrd 이미지 파일 생성

구현된 차량용 동영상 재생기에서는 initial RAM 디스크를 활용하여 32MB 크기의 루트 파일 시스템을 구성하였다. 루트 파일 시스템에는 라이브러리와 프로그램을 저장하였으며, 용량을 줄이기 위해 Multi-Call Binary 파일인 Busybox 를 사용하였다. RAM 디스크 공간은 loopback 장치를 사용하여 만들었고 ext2 파일 시스템 형식으로 포맷한 후 루트 파일 시스템을 마운트(mount)하였다. 그리고, /bin, /dev, /etc, /lib 등의 디렉토리를 생성하고 필요한 파일들을 복사하였다. " /linuxrc" 스크립트를 수정하여 동영상 재생기 프로그램이 필요로 하는 라이브러리와 폰트 파일을 복사한 후 재생기 프로그램이 자동 실행되도록 작성하였다. 끝으로, 라이브러리의 루트 위치를 지정해 주고 파일 시스템을 언마운트(unmount) 하였다. 파일 시스템을 압축해서 이미지 파일을 만들었으며, 최종적으로 생성된 이미지 파일(initrd.img)의 크기는 696KB 였다.

```
dd if=/dev/zero of=/dev/loop bs=1k count=32768
mke2fs -m 0 -I 2000 /dev/loop
mount -o loop -t ext2 /dev/loop /mnt/ramdisk
...        ←필수 디렉토리와 라이브러리의 복사
ldconfig -r /mnt/ramdisk
umount /mnt/ramdisk
dd if=/dev/loop bs=1k count=32768 | gzip -v9 > initrd.img
```

4.4 차량용 동영상 재생기의 개발 과정

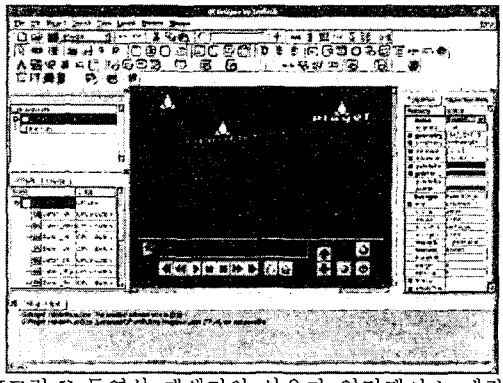
호스트 컴퓨터에서 커널 이미지와 INITRD 이미지 파일을 작성한 후 디스크의 테스트 파티션에서 기능을 시험하였다. 부팅용 CD-ROM 은 SYSLINUX 부트로더를 이용하여 부팅가능한 플로디 디스크를 작성하였다. 플로피 디스크에는 부팅에 필요한 리눅스 커널과 initrd 이미지 파일을 담는다. 부팅용 CD-ROM 을 제작할 때 플로피 디스크를 이용해서 부팅 영역을 만들고 데이터 영역인 ISO9660 파일 시스템에는 응용프로그램과 라이브러리, 기타 필요한 파일들을 복사한다.



[그림 4] 부팅용 CD-ROM 제작 과정

4.5 동영상 재생기의 그래픽 사용자 인터페이스 구현

QT 의 Designer Tool 을 사용해서 Widget 을 디자인하고 시그널과 슬롯을 연결하여 이벤트 처리 기능을 구현하였다.



[그림 5] 동영상 재생기의 사용자 인터페이스 제작

동영상 재생기에서 중요한 기능은 동영상 프로그램인 Mplayer 를 제어하여 동영상 파일을 원활하게 재생시키는 것이다. 대개 임의의 프로그램에서 외부 프로

그램을 실행시킬 때는 표준 C 라이브러리의 system() 함수를 사용한다. 그러나, system() 함수를 이용하여 외부 프로그램을 실행하게 되면 제어가 외부 프로그램으로 완전히 넘어가므로 원래 프로그램의 제어가 불가능하게 된다. 따라서, 본 논문에서는 QT 에서 제공하는 클래스인 QProcess 를 사용하였다. QProcess 클래스는 system() 함수와 달리 자식 프로세스가 종료될 때까지 기다리지 않는 Non-blocking 체제이다. 또한, system() 함수는 수행 결과만 반환하지만 QProcess 클래스는 수행 중에 다수의 인자를 프로세스에 제공하거나 결과를 반환할 수도 있다.

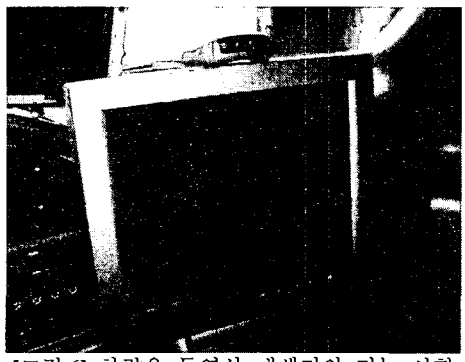
따라서, QProcess 클래스를 이용한 프로세스 생성을 위해서 QProcess 객체를 생성하고 인자를 추가한 후 start() 함수를 호출하게 된다. 그리고, 생성된 프로세스의 종료를 감지하도록 connect() 함수를 사용하여 QT 의 시그널 슬롯을 연결한다. proc 객체의 processExited 이벤트가 발생하면 사용자 정의 함수인 playExited() 가 호출된다. 한편, 실행중인 프로세스를 제어하려면 프로세스에게 표준 입력을 전달하는데, proc 객체의 writeToStdin() 함수를 통해 인자를 넘겨주면 실행중인 프로세스에게 제공된다. 실행중인 프로세스는 입력으로 주어진 인자를 처리한다.

```
proc = new QProcess( this );

proc->addArgument( "mplayer" );
proc->addArgument( "-vo" );
proc->addArgument( "fbdev" );
proc->addArgument( "-fs" );
proc->addArgument( file );

connect( proc, SIGNAL(processExited()), this,
SLOT(playExited()) );

if ( !proc->start() ) {
        // error handling
        exit( -1 );
}

...

void Form1::pause()
{
    proc->writeToStdin("p");
}

...
```

동영상 재생기 프로그램이 완성되면 동영상 재생기 프로그램과 함께 QT/Embedded 라이브러리 파일과 폰트 파일을 루트 파일 시스템인 initrd.img 이미지 파일에 담아서 차량용 동영상 재생기 시스템을 구현한다.

5. 결론

본 논문에서는 리눅스를 기반으로 차량용 동영상 재생기를 구현하였다. 리눅스를 동영상 재생기에 적합하게 구성하여 CD-ROM 에 담았기 때문에 동영상 재생용으로 효율적인 활용이 가능하였다. 그러나, 구현된 차량용 동영상 재생기는 CD-ROM 을 부팅 매체로 사용하고 있어서 약 30 초 정도의 부팅 시간이 소요되는 단점이 있다. 향후에는 플래시 메모리를 부팅 매체로 활용할 예정이며, 이렇게 되면 기존의 부팅 소요 시간을 단축시킬 수 있을 것이다. 또한, 부팅 후 부팅용 CD-ROM 과 동영상 파일을 수록한 CD-ROM 을 교체할 필요가 없어지므로 사용자가 보다 편리하게 동영상 재생기를 사용할 수 있을 것이다.



[그림 6] 차량용 동영상 재생기의 기능 시험

참고문헌

[1] John Lombardo, 임베디드 리눅스, 인포북, 2002

[2] 커널소스 /Documentation/initrd.txt, /Documention/ramdisk.txt

[3] 커널소스, /Documentation /framebuffer.txt

[4] 커널컴파일, http://kldp.org/KoreanDoc/Kernel_Compile-KLDP

[5] 이연조, 임베디드 리눅스 프로그래밍, PCBOOK

[6] 미지리서치㈜ 부설연구소, Linuette Education

[7] Qt/Embedded, http://www.trolltech.com/products/embedded/index.html

[8] 송호중, 따라해보세요 Qt 리눅스 프로그래밍, 한컴프레스, 2000