PDA 용 임베디드 리눅스 파일 시스템 설계

장승주*, 황정현*, 류진영*
*동의대학교 컴퓨터공학과
E-mail : sjjang@deu.ac.kr,
hwang7000@hotmail.com, ros207@naver.com

Design of the Embedded Linux File System for the PDA System

Seung-Ju Jang*, Jung-Hyen Hwang*, Jin-Young Ryu*
*Dept. of Computer Engineering, Dong-Eui University

요 약

본 논문에서는 PDA 용 임베디드 리눅스의 파일시스템을 설계 한다. 개발을 위한 Host PC 구축 [크로스 컴파일러(Cross compiler)] 및 커널 소스의 패치와 디버그를 통한 개발을 한다. 본 논문에서 PDA 시스템에 리눅스를 포팅 하여 PDA 임베디드 리눅스 시스템 및 파일 시스템을 설계 한다.

1. 서론

오늘날 Embedded 시스템에서 Embedded Linux 의 성공적인 사용은 Embedded 업계 및 사용자들은 물론 기사, 웹사이트등에 대폭적인 열의와 관심사를 이끌어 내고 있다. 임베디드 시스템(Embedded system)이란 미리 정해진 특정 기능을 수행하기 위해 컴퓨터의 하드웨어와 소프트웨어가 조합된 전자 제어 시스템을 말한다[1]. 이들 중 Embedded System 을 탑재한 PDA(Personal Digital Assistant) 즉, 휴대형 정보 단말기는 현재 많은 분야에서 널리 사용되고 있으며, 크기는 노트북보다 훨씬 작은 소형 컴퓨터이면서 기능은 전자수첩보다 강력한 컴퓨팅 파워를 갖고 있다.

임베디드 시스템의 OS 로는 팜, WinCE, Embedded Linux 등이 있다. WinCE 는 개발환경이 쉽고 안정된 반면 비싼 라이센스를 지불해야 하며 Embedded Linux 는 오픈 소스이기 때문에 현재 활발한 연구가 이루어지고 있다. Embedded Linux 에서 주로 사용하는 파일시스템으로는 JFFS2 파일시스템이 있는데, 이는 linux kernel 2.4 버전에서 동작하고, 원활한 garbage collection 을 위한 thread 를 가지고 있으며, 갑작스러운 전원공급 차단에 대비한 저널링 기능 및 zlib 를 이용한 데이터 압축 기법을 사용하여 메모리 사용의 극대화를 가능케 하고 있다. 단 저널링을 하기위한 약간의 공간 낭비와 압축으로 인해 성능이 약간 느리다는 단점이 있다.

본 논문 내용은 Embedded Linux 기반의 PDA 에 대한 설계와 메모리 사용 극대화를 위한 File System 의 설계이다. 본 논문은 JFFS2 파일시스템에 새로운 압축 알고리즘을 적용하여 더 나은 파일시스템을 구현함으로써 향상된 메모리 공간의 사용을 목표로 한다.

본 논문의 구성은 2 장에서 관련연구를 살펴보고, 3 장에서 파일시스템 설계 내용에 대해 살펴보고, 4 장 결론 순으로 되어있다.

2. 관련연구

JFFS2(Journalling Flash Filesystem version 2)는 JFFS 의 다음 버전으로 스웨덴의 Axis Communications 에서 개발된 JFFS 파일 시스템을 Redhat 에서 Embedded 시스템을 위해 만든 새로운 저널링 파일 시스템이다. JFFS2 의 장점은 zlib 를 이용한 데이터를 고도의 압축하여 메모리 사용을 극대화 시키는 것이다. 현재 메모리 사용의 극대화 및 안정화를 위한 압축 알고리즘 및 파일 시스템 구축에 대한 연구가 활발히 이루어지고 있다.

대표적인 무손실 압축 알고리즘으로는 RLC(Run Length Coding), VLC(Variable Length Coding), Huffman Coding, DBC(Dictionary Based Coding), LZ-style(Lemple-Ziv-style)등이 있다[8].

3. 설계

3.1 컴파일 환경 수정

커널소스의 root 에 있는 Makefile 을 [그림 1]과 같이 아키텍처 및 크로스 컴파일러 변수를 수정한다.

```
.....
ARCH := arm
.....
CROSS_COMPILE = /usr/local/arm/2.95.3/bin./arm-linux-
.....
```

[그림 1] 수정된 Makefile

또한, include 의 아키텍처도 make 할 아키텍처와 동일하도록 [그림 2]와 같이 수정해준다.

```
ln –s include/asm-arm iclude/asm
ln –s include/asm/arch-sa1100 include/asm/arch
ln –s include/asm/proc-armv include/asm/proc
```

[그림 2] include 의 아키텍처의 수정(심볼링크)

3.2 mtd 패치 및 커널 구조체 정보 수정

3.2.1 mtd-snapshot 의 patch 단계

MTD 는 임베디드 디바이스에서 고형체 파일시스템을 구성하는데 사용하는 플래시 메모리, RAM, 그리고 비슷한 다른 칩셋 등 메모리 장치이다. mtd 패치를 하는 방법은 크게 두 가지로 나뉘는데 커널 소스 트리 안에 패치하는 방법과 커널 트리 밖에서 패치하는 방법이 있다. 본 논문에서는 커널 트리 안에 패치를 하였다.

```
tar vfxz mtd-snopshot.tar.gz
cd mtd/patches
sh patches.sh /커널이 위치한 절대경로
```

[그림 3] mtd 패치 단계

[그림 3]와 같이 패치를 해주면 mtd 안에 각각의 파일들이 커널 소스에 심볼링크 된다.
mtd 패치를 하고 drivers/mtd/maps/ 디렉토리 아래에서 Target board 인 PDA 의 커널 구조체 부분을 결정하는 sa1100-flash.c 파일의 MTD 구조체 부분[그림 4]과 같이 수정한다.

```
static struct mtd_partition h3600_partitions[] =
{
  {name: "H3600 boot firmware",size: 0x00040000, offset: 0,
  mask_flags: MTD_WRITEABLE},
  {name: "H3600 kernel",size: 0x000c0000, offset: 0x0008000},
  {name: "H3600 params", size: 0x00040000, offset: 0x00040000},
  {name: "H3600 root cramfs", size: 0x140000, offset: 0x140000},
  {name: "H3600 usr jffs2",offset: 0x280000, size: 0xd80000}
};
```

[그림 4] 커널 구조체를 수정한 부분

3.3 jffs2 파일 시스템의 변경 및 구현

3.3.1 jffs2 파일 시스템에 입출력 되는 데이터의 변경

3.3.1.1 파일 시스템에서 읽어오는 부분의 변경

[그림 5]은 read.c 파일의 변경 부분으로써 Int JFFS2 _read_dnode()의 부분의 deconprbuf 는 page 로 읽어 들이는 것이고 buf 는 현재 램 디스크에 저장되어 있는 값을 나타낸다.

```
  Int JFFs2_read_dnode()
.....
  Out_deconprbuf
  If(decomprbuf != buf && decomprbuf != readbuf)
                Kfree(decomprbuf);
.....
  for(I=0 ; I <len ; I++)
  {
        decomprbuf[I]=buf[I]-1;
  }
.....
```

[그림 5] read.c 파일의 변경 부분

3.3.1.2 테스트

아래의 [그림 6]은 read 하는 부분만 수정한 경우의 출력 값을 테스트한 것으로 테스트 과정 중 reboot 을 시킨 이유는 한번이라도 파일시스템으로 읽기 및 쓰기를 실행하였다면 사용된 데이터의 내용이 page 로 저장되어 page fault 가 되기 전까지는 재사용 시 파일 시스템으로 저장된 내용을 사용하는 대신 page 로 올라간 내용이 사용되기 때문이다.

```
echo 11 > 1
reboot
cat 1
00
```

[그림 6] read 하는 부분만 수정된 경우

read.c 파일만 변경한 경우 위의 [그림 6]과 같이 cat 을 실행한 결과 값이 00 으로 출력이 되었는데 그 이유는 파일 시스템에 저장된 데이터의 값은 11 이지만 read 될 때 데이터가 □ 1 되어서 쓰이기 때문이다

3.3.2.1 파일 시스템으로 쓰여지는 부분 변경

데이터가 파일 시스템으로 쓰여질 때는 다음[표 3]과 같은 3 가지 방법으로 각각의 알고리즘을 이용하여 압축된 후 쓰여진다.

[표 1] 압축 알고리즘에 사용되는 파일들

압축 알고리즘	압축이 실행되는 파일
JFFS2_COMP_NONE	write.c
JFFS2_COMPR_RTIME	compr_rtime.c
JFFS2_COMPR_ZLIB	compr_zlib.c

입력부분은 하나이지만 압축되는 부분은 NONE, RTIME, ZLIB 3 부분 이므로 수정해야할 출력부분도 [그림 7]과 같은 3 부분이다.

```
write.c 파일의 수정 부분

int Jffs2_write_inode_range()
.....
if(comprtype = JFFS2_COMPR_NONE)
        if(conprbuf)
                kfree(comprbuf);
                comprbuf=buf;
                datalen=cdatalen;
for(I=0; I<datalen; I++)
{
        comprbuf[I]=buf[I]+1;
}
.....


compr_rtime.c 파일의 수정 부분

void jffs2_rtime_decompress
.....
while(outpos < destlen)
{
cpage_out[outpos]=value     -> cpage_out[outpos]=value+1;
}
.....


compr_zlib.c 파일의 수정 부분

void jffs2_zlib_decompress
.....
{

        up(&inflate_sem);
        for(I=0;I<desten;I++)
        {
                cpage_out[I]=cpage_out[I]+1;
        }
}
```

[그림 7] 출력 파일의 수정 부분

3.3.2.2 테스트

read 및 write 부분을 모두 수정한 경우의 출력 값은 다음과 같다.

```
echo 11 > 1
reboot
cat 1
11
```

[그림 8] read 및 write 부분을 모두 수정한 경우

파일을 read 하는 부분(-1)과 write 하는 부분(+1)을 모두 수정하였을 때 다음과 같이 정상적인 데이타가 출력된다. 즉, 실제 파일시스템에는 22 라는 데이터가 저장되며 read 할 경우 각 데이터가 -1 되어 정상적인 데이터인 11 이 출력된다.

3. 결론

본 논문에서는 PDA 용 mtd 패치를한 임베디드 리눅스의 파일시스템(JFFS2)의 입출력부분을 +1,-1 시켜 설계하여 PDA 쉘 상에서 파일을 입출력 시켰을 때 정상적인 출력을 얻었다. 향후 입출력 부분에 새로운 함수를 추가하여 2 중 압축도 실현할 수 있다. 그리고 메모리 사용의 극대화 및 안정화를 위한 새로운 압축 알고리즘과 파일시스템을 구현하여 Embeddle Linux 에 적용 하기 위한 연구가 필요할 것이다.

참고문헌

[1] 임베디드 시스템 + 임베디드 리눅스, 박영환, 사이텍미디어
[2] Embedded Linux Hardware, Software, and Interfacing, Crag Hollabaugh, Ph.D. Addison-Wesley
[3] Understanding the LINUX KERNEL, DANIEL P. BOVE T & MARCO CESATI, O'REILLY
[4] Building Embedded LINUX SYSTEMS, KARIM TAGH MOUR, O'REILLY
[5] Linux hacker 들을 위한 UNIX KERNEL 완전분석으로 가는 길, 박장수 pp110-140
[6] Linux File Systems, MOSHE BAR, OSBORNE pp23-74 pp219-232
[7] RUNNING LINUX 3rd Ed, Matt Welsh, Lar Kaufman, Kalle Dalheimer, O'REILLY,1999
[8] Introduction to Data Compression Second Edition, Khalid Sayood,, Moragn Kaufmann, 2000
[9] http://www.wowlinux.com/nwes/newslist.html