

Strong Flash File System

(Team SFFS)

팀 원 : 정준섭
김재홍
오정환
임종훈

Contents

1. 프로젝트 개요	3
2. 개발환경	4
3. 기능 설명 및 테스트 방법	4
3.1. SFFS 및 YAFFS의 모듈 올리기	4
3.2. 트랜잭션 처리의 샘플 프로그램 설명	6
3.2.1. 샘플 프로그램 1	6
3.2.2. 샘플 프로그램 2	7
3.3. 트랜잭션 처리 테스트(YAFFS) – 새 파일 쓰기	8
3.4. 트랜잭션 처리 테스트(SFFS) – 새 파일 쓰기	9
3.5. 트랜잭션 처리 테스트(YAFFS) – 파일 수정하기	10
3.6. 트랜잭션 처리 테스트(SFFS) – 파일 수정하기	12
3.7. 마모도 평준화(Wear Leveling : YAFFS) 테스트	14
3.7.1. 테스트 방법	14
3.7.2. 마모도 평준화 테스트	15
3.8. 마모도 평준화(Wear Leveling : SFFS) 테스트	16
3.9. 지속적인 마모도 평준화(Wear Leveling) 테스트	17
3.10. LTP(Linux Test Project)의 fsstress 테스트	18
3.10.1. Linux Test Project란?	18
3.10.2. fsstress테스트 결과	18

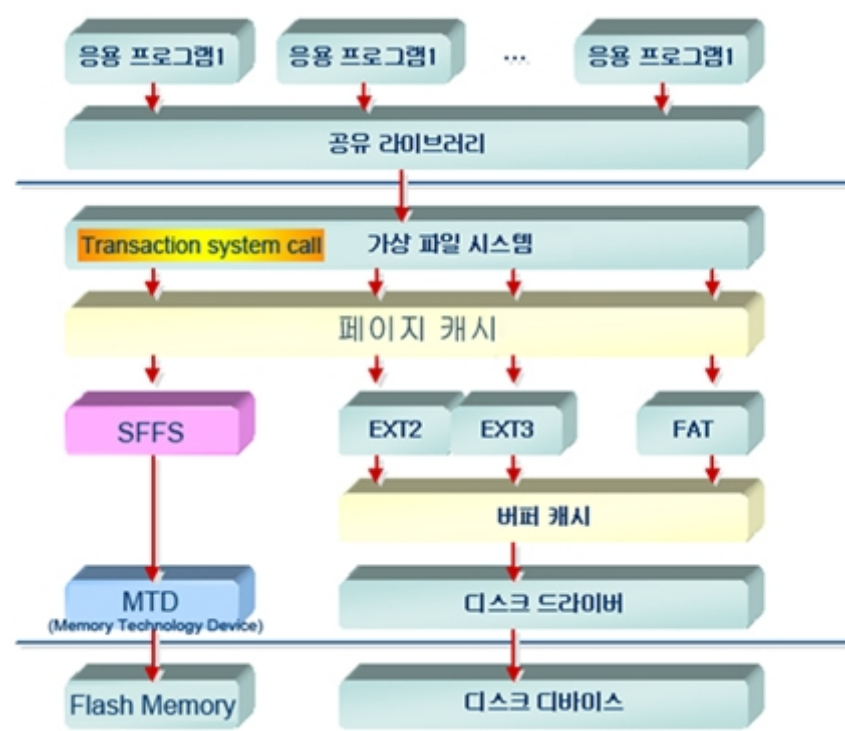
1. 프로젝트 개요

낸드 플래시 메모리는 하드디스크와 틀리게 읽기, 쓰기는 페이지 단위로 지우기는 블록단위로 되어있다. 또한 블록의 수명은 10만번이기 때문에 수명을 늘리기 위해서는 모든 블록을 골고루 써야 한다는 특징이 있다.

낸드 플래시메모리는 같은 주소에 수정된 내용을 바로 다시 쓰는 것이 불가능하다. 만약 같은 자리에 다시 쓰고자 한다면 그 블록이 포함된 세그먼트에 소거작업을 수행한 후 다시 쓸 수 있다. 따라서 수정된 블록은 쓰기 가능한 블록이 있는 세그먼트에 써져야 한다. 여기서 세그먼트는 여러 개의 블록들이 모여 하나의 세그먼트를 이루게 되는데 플래시 메모리의 종류에 따라 그 크기가 다양하다.

이러한 플래시 메모리의 단점을 극복하고 효과적으로 사용하기 위해 다양한 지움 정책(cleaning policy)과 마모도 평준화(wear-leveling) 기법을 적용하기로 하였다.

기존의 플래시 파일 시스템인 YAFFS는 파일을 쓰다가 갑자기 전원이 나가거나 물리적인 에러가 발생을 하면 파일에 쓰여지고 있던 부분들까지 쓰여지고 그 이후의 것들은 그대로 남아 있어서 파일이 깨지게 되는 상황이 발생된다. 이러한 부분을 보완하기 위해 트랜잭션 처리를 생각하게 되었다.



<SFFS의 시스템 구성도>

2. 개발환경

개발보드 : EZ-X5(PXA-255기반) <http://www.falinux.co.kr>

타겟보드의 커널 : 2.4.19

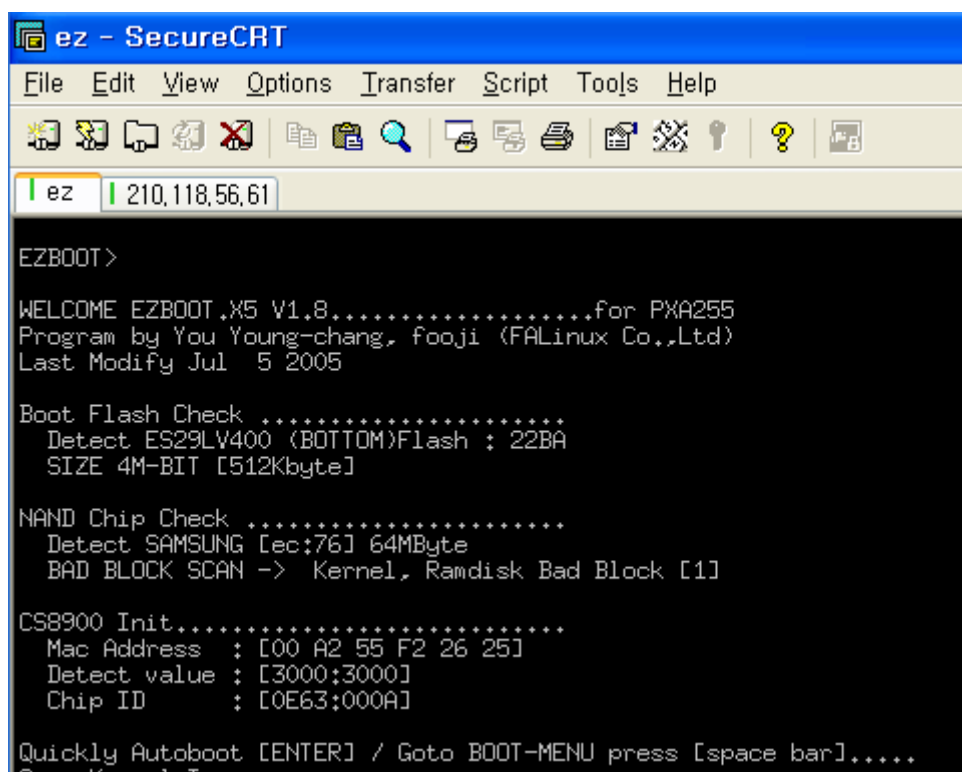
크로스 컴파일러 : arm-gcc

소프트웨어 : vi, Source Insight

3. 기능 설명 및 테스트 방법

3.1. SFFS 및 YAFFS의 모듈 올리기

1. EZ-X5보드에서 커널이미지와 램디스크 이미지를 TFTP를 통해서 올린다.



```
ez - SecureCRT
File Edit View Options Transfer Script Tools Help
ez 210.118.56.61
EZBOOT>
WELCOME EZBOOT.X5 V1.8.....for PXA255
Program by You Young-chang, fooji (FALinux Co.,Ltd)
Last Modify Jul 5 2005

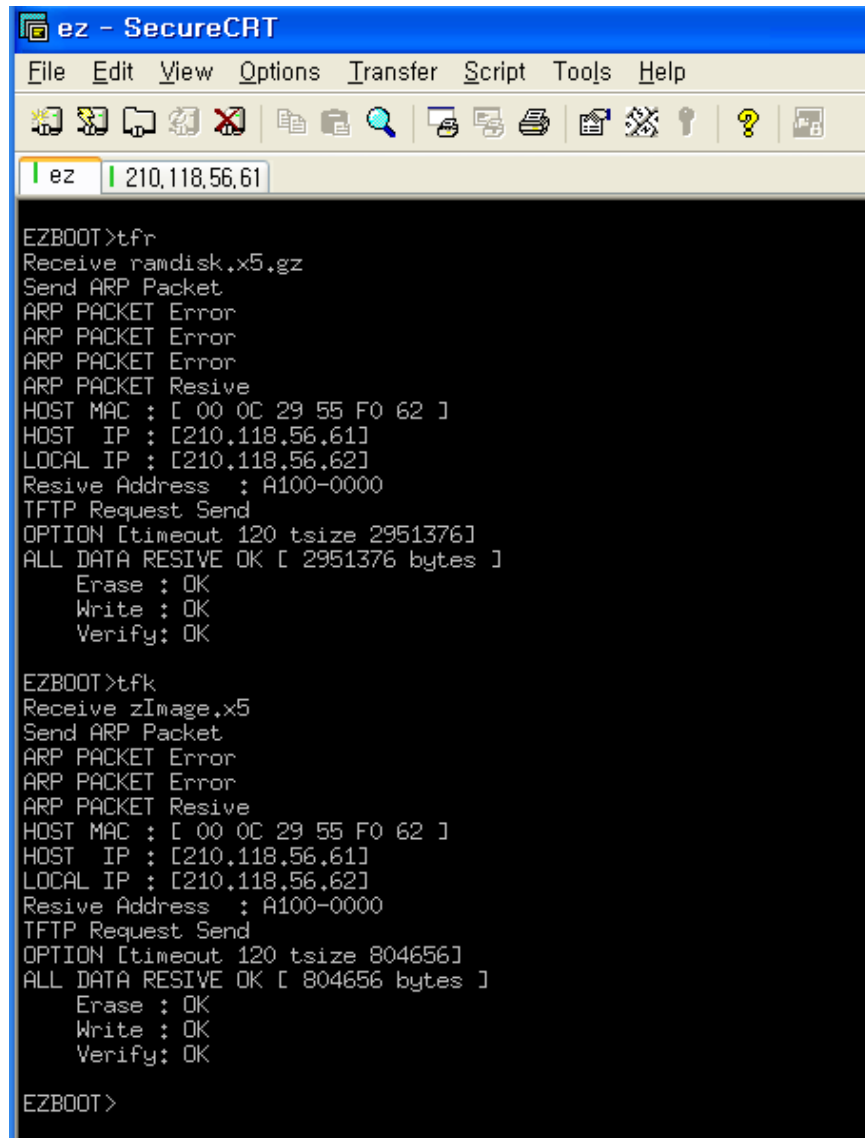
Boot Flash Check .....
Detect ES29LV400 (BOTTOM)Flash : 22BA
SIZE 4M-BIT [512Kbyte]

NAND Chip Check .....
Detect SAMSUNG [ec:76] 64MByte
BAD BLOCK SCAN -> Kernel, Ramdisk Bad Block [1]

CS8900 Init.....
Mac Address : [00 A2 55 F2 26 25]
Detect value : [3000:3000]
Chip ID : [0E63:000A]

Quickly Autoboot [ENTER] / Goto BOOT-MENU press [space bar].....
```

<EZ-X5의 부트로더 화면>



```
ez - SecureCRT
File Edit View Options Transfer Script Tools Help

ez 210.118.56.61

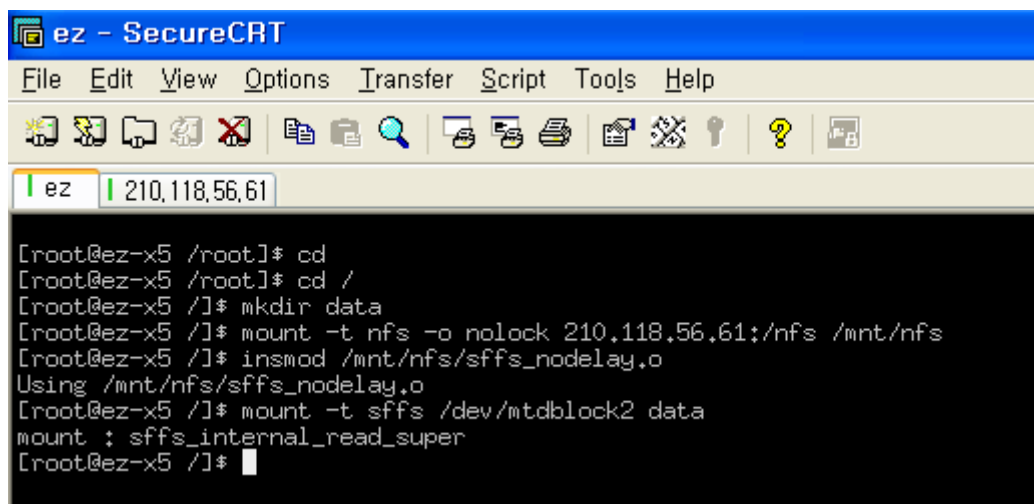
EZBOOT>tfr
Receive ramdisk.x5.gz
Send ARP Packet
ARP PACKET Error
ARP PACKET Error
ARP PACKET Error
ARP PACKET Resive
HOST MAC : [ 00 0C 29 55 F0 62 ]
HOST IP : [210.118.56.61]
LOCAL IP : [210.118.56.62]
Resive Address : A100-0000
TFTP Request Send
OPTION [timeout 120 tsize 2951376]
ALL DATA RESIVE OK [ 2951376 bytes ]
Erase : OK
Write : OK
Verify: OK

EZBOOT>tfk
Receive zImage.x5
Send ARP Packet
ARP PACKET Error
ARP PACKET Error
ARP PACKET Resive
HOST MAC : [ 00 0C 29 55 F0 62 ]
HOST IP : [210.118.56.61]
LOCAL IP : [210.118.56.62]
Resive Address : A100-0000
TFTP Request Send
OPTION [timeout 120 tsize 804656]
ALL DATA RESIVE OK [ 804656 bytes ]
Erase : OK
Write : OK
Verify: OK

EZBOOT>
```

<커널이미지와 램디스크 이미지를 보드에 올린다>

2. 타겟 보드를 리눅스로 부팅한 후 NFS를 마운트하고 파일시스템(SFFS 혹은 YAFFS)를 마운트 한다.



```
ez - SecureCRT
File Edit View Options Transfer Script Tools Help

ez 210.118.56.61

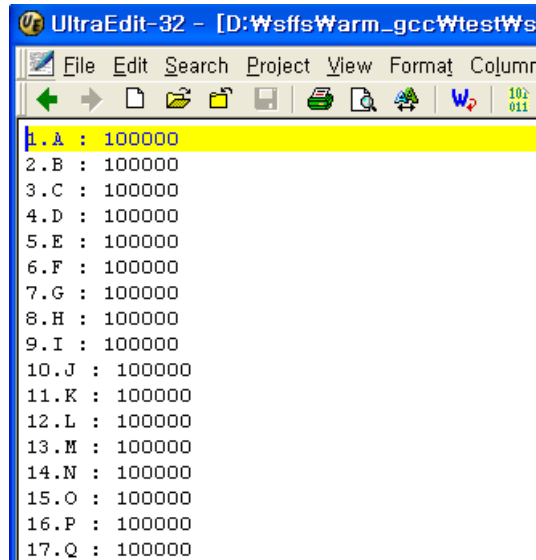
[root@ez-x5 /root]$ cd
[root@ez-x5 /root]$ cd /
[root@ez-x5 /]$ mkdir data
[root@ez-x5 /]$ mount -t nfs -o nolock 210.118.56.61:/nfs /mnt/nfs
[root@ez-x5 /]$ insmod /mnt/nfs/sffs/sffs_nodelay.o
Using /mnt/nfs/sffs/sffs_nodelay.o
[root@ez-x5 /]$ mount -t sffs /dev/mtdblock2 data
mount : sffs_internal_read_super
[root@ez-x5 /]$
```

<NFS와 타겟 파일시스템 마운트>

3.2. 트랜잭션 처리의 샘플 프로그램 설명

3.2.1. 샘플 프로그램 1

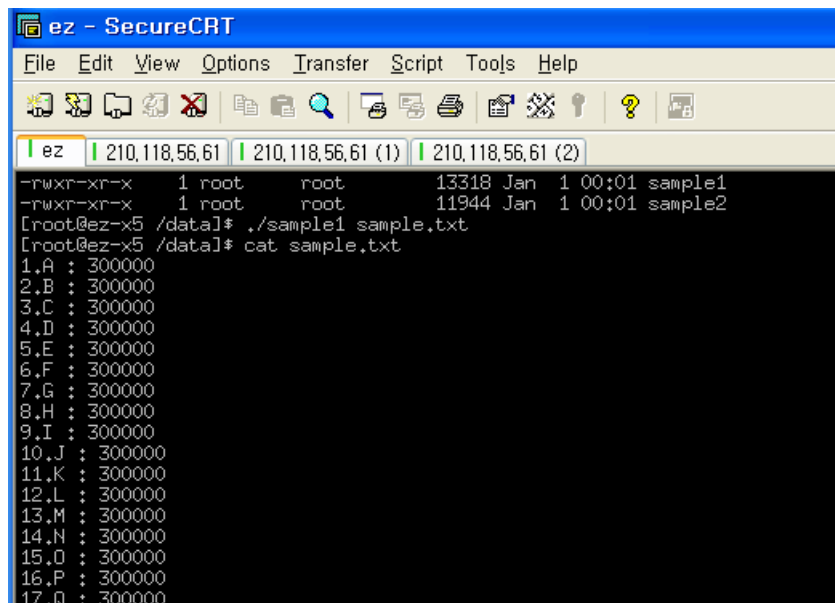
- 실행방법 : [root@ez-x5 /data]\$./sample1 sample.txt
- Input : 샘플 txt파일
- Sample1은 sample1.txt의 데이터를 일부 수정하는 기능을 한다.



```
UltraEdit-32 - [D:\Wsffs\Warm_gcc\Wtest\Wsa
File Edit Search Project View Format Column
1.A : 100000
2.B : 100000
3.C : 100000
4.D : 100000
5.E : 100000
6.F : 100000
7.G : 100000
8.H : 100000
9.I : 100000
10.J : 100000
11.K : 100000
12.L : 100000
13.M : 100000
14.N : 100000
15.O : 100000
16.P : 100000
17.Q : 100000
```

<sample1.txt 파일 - 위의 인덱스가 100번까지 반복된다.>

- Sample1 실행결과

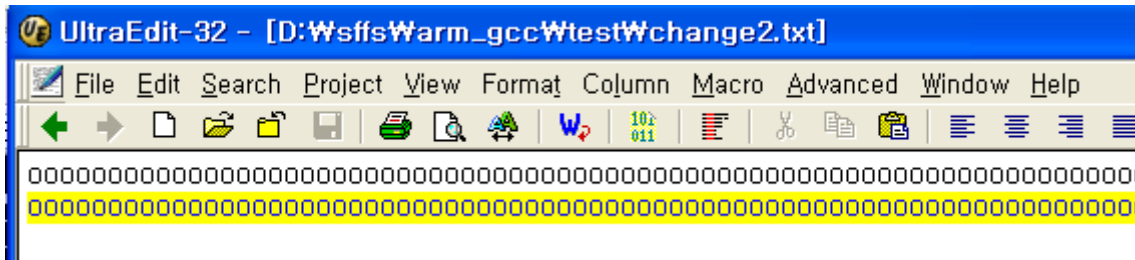


```
ez - SecureCRT
File Edit View Options Transfer Script Tools Help
ez | 210,118,56,61 | 210,118,56,61 (1) | 210,118,56,61 (2)
-rwxr-xr-x 1 root root 13318 Jan 1 00:01 sample1
-rwxr-xr-x 1 root root 11944 Jan 1 00:01 sample2
[root@ez-x5 /data]$ ./sample1 sample.txt
[root@ez-x5 /data]$ cat sample.txt
1.A : 300000
2.B : 300000
3.C : 300000
4.D : 300000
5.E : 300000
6.F : 300000
7.G : 300000
8.H : 300000
9.I : 300000
10.J : 300000
11.K : 300000
12.L : 300000
13.M : 300000
14.N : 300000
15.O : 300000
16.P : 300000
17.Q : 300000
```

<sample1의 실행 결과 - 숫자를 모두 300000로 바꾼다.>

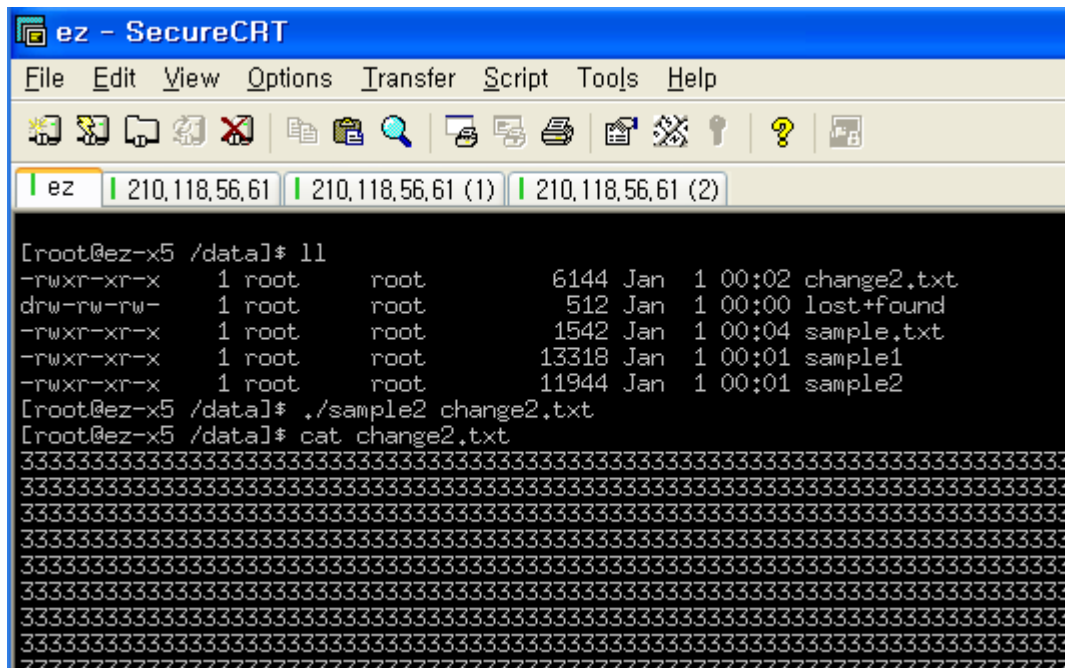
3.2.2. 샘플 프로그램 2

- 실행방법 : [root@ez-x5 /data]\$./sample2 change2.txt
- Input : 샘플 txt파일
- Sample2는 change2.txt의 모든 데이터를 수정하는 기능을 가진다.



<change2.txt 파일 - 6K의 크기이며 모두 0이다>

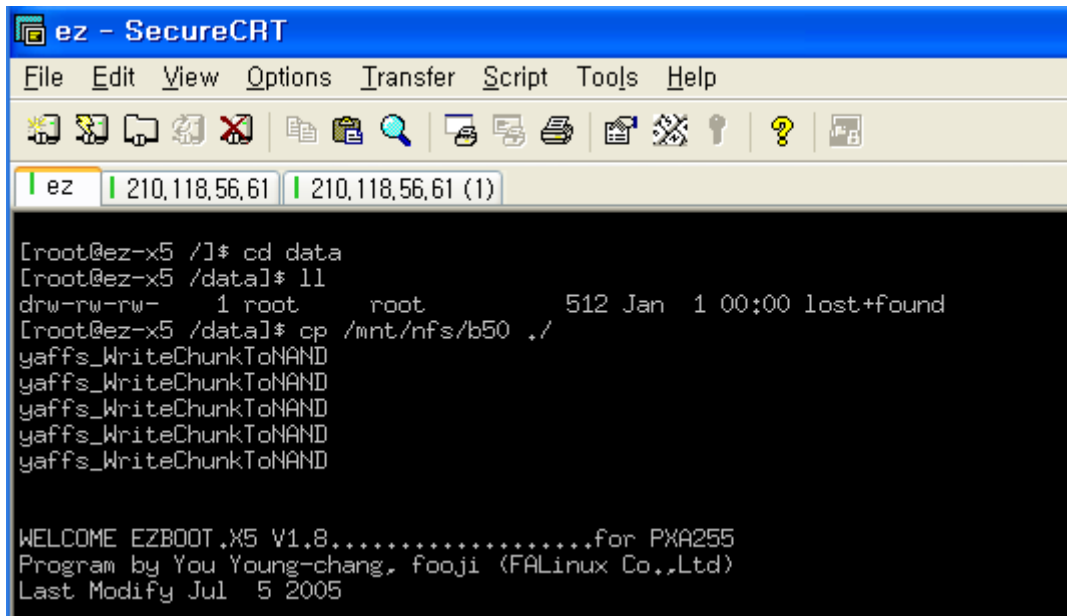
- Sample2 실행결과



<sample2의 실행 결과 - 숫자를 모두 3으로 바꾼다.>

3.3. 트랜잭션 처리 테스트(YAFFS) – 새 파일 쓰기

1. YAFFS로 마운트를 한 후 NFS에서 새 파일을 카피한다.



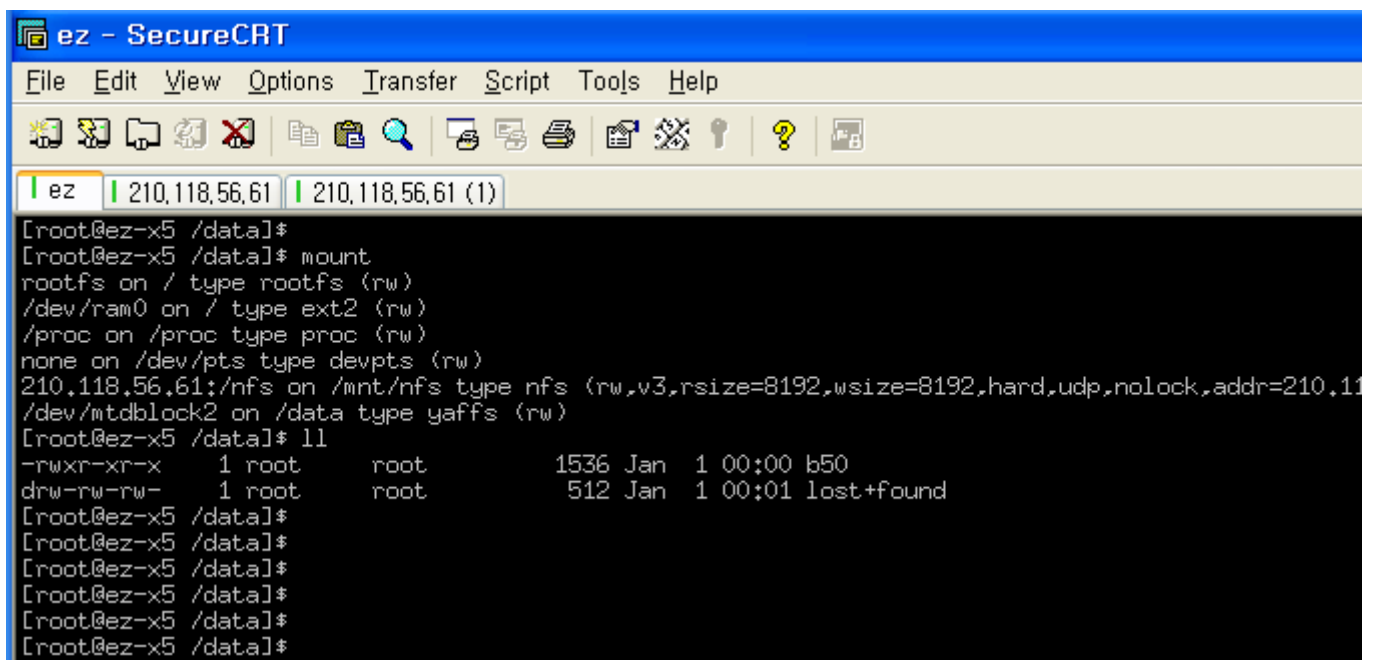
```
ez - SecureCRT
File Edit View Options Transfer Script Tools Help
ez | 210,118,56,61 | 210,118,56,61 (1)

[root@ez-x5 /]$ cd data
[root@ez-x5 /data]$ ll
drw-rw-rw- 1 root root 512 Jan 1 00:00 lost+found
[root@ez-x5 /data]$ cp /mnt/nfs/b50 ./
yaffs_WriteChunkToNAND
yaffs_WriteChunkToNAND
yaffs_WriteChunkToNAND
yaffs_WriteChunkToNAND
yaffs_WriteChunkToNAND

WELCOME EZBOOT,X5 V1.8.....for PXA255
Program by You Young-chang, fooji (FALinux Co.,Ltd)
Last Modify Jul 5 2005
```

<NFS에서 b50이라는 파일을 카피 하는 도중 리셋을 누른다.>

2. 리셋 후 다시 YAFFS로 마운트 하면 데이터가 트랜잭션 단위로 처리되지 않아서 완전하지 않은 파일이 남아 있다. 이것은 데이터의 무결성을 YAFFS가 보장하지 않는 것이다.



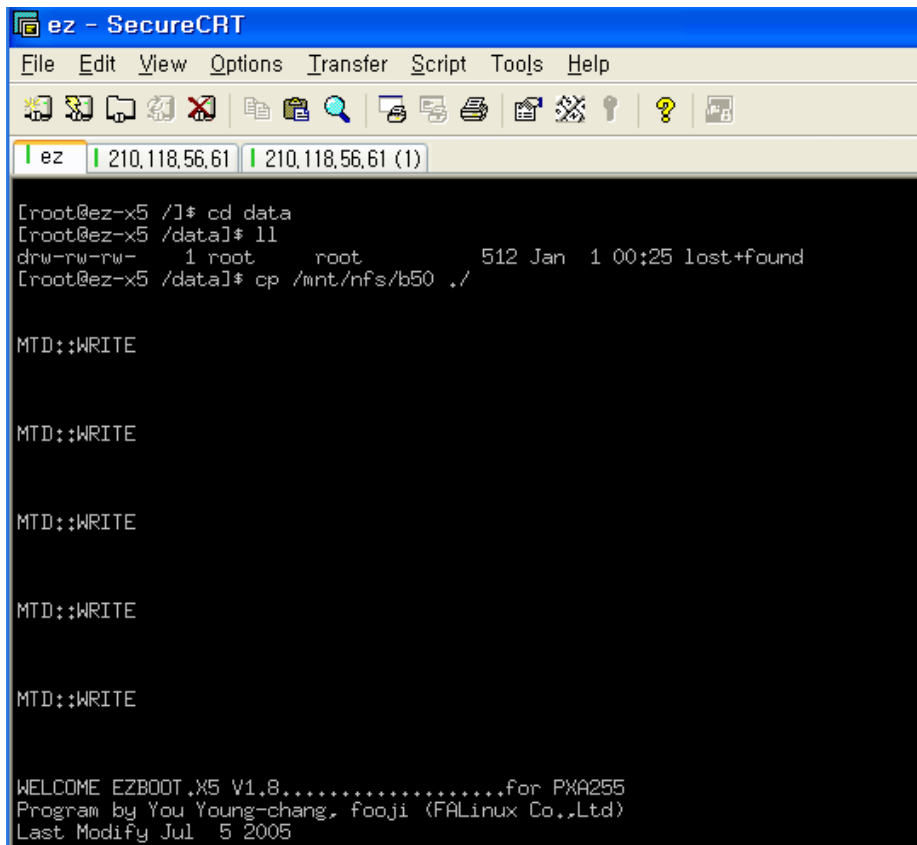
```
ez - SecureCRT
File Edit View Options Transfer Script Tools Help
ez | 210,118,56,61 | 210,118,56,61 (1)

[root@ez-x5 /data]$
[root@ez-x5 /data]$ mount
rootfs on / type rootfs (rw)
/dev/ram0 on / type ext2 (rw)
/proc on /proc type proc (rw)
none on /dev/pts type devpts (rw)
210.118.56.61:/nfs on /mnt/nfs type nfs (rw,v3,rsize=8192,wsz=8192,hard,udp,nolock,addr=210.118.56.61)
/dev/mtdblock2 on /data type yaffs (rw)
[root@ez-x5 /data]$ ll
-rwxr-xr-x 1 root root 1536 Jan 1 00:00 b50
drw-rw-rw- 1 root root 512 Jan 1 00:01 lost+found
[root@ez-x5 /data]$
[root@ez-x5 /data]$
[root@ez-x5 /data]$
[root@ez-x5 /data]$
[root@ez-x5 /data]$
[root@ez-x5 /data]$
```

<리셋 후 YAFFS로 다시 마운트를 한 상태.>

3.4. 트랜잭션 처리 테스트(SFFS) – 새 파일 쓰기

1. SFFS로 마운트를 한 후 NFS에서 새 파일을 카피한다.



```
ez - SecureCRT
File Edit View Options Transfer Script Tools Help
ez 210.118.56.61 210.118.56.61 (1)

[root@ez-x5 /]$ cd data
[root@ez-x5 /data]$ ll
drw-rw-rw- 1 root root 512 Jan 1 00:25 lost+found
[root@ez-x5 /data]$ cp /mnt/nfs/b50 ./

MTD::WRITE

MTD::WRITE

MTD::WRITE

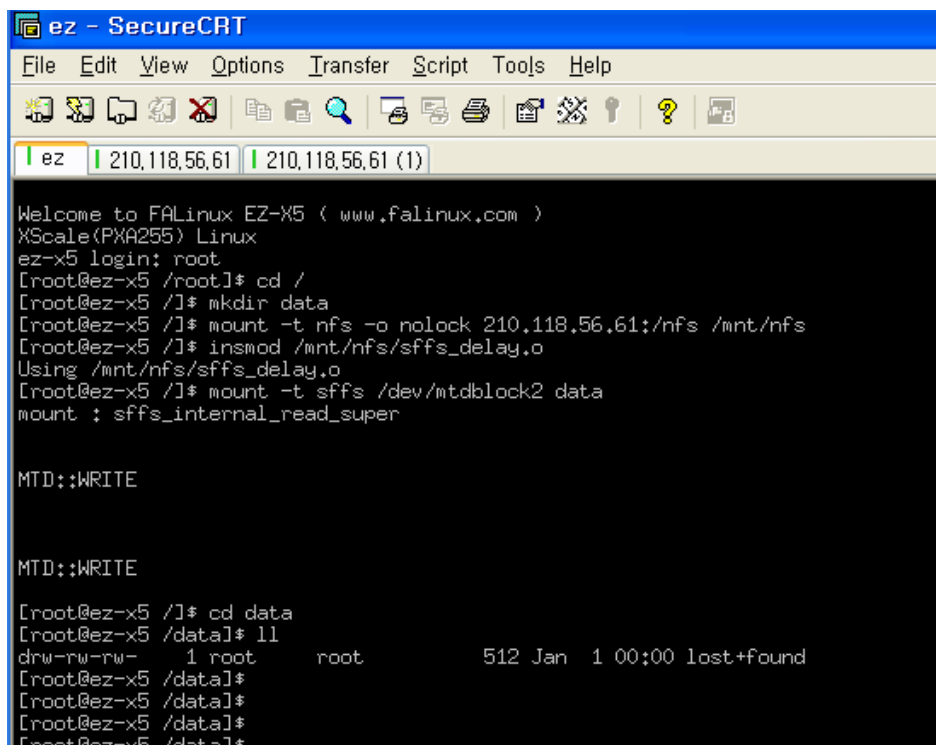
MTD::WRITE

MTD::WRITE

WELCOME EZBOOT.X5 V1.8.....for PXA255
Program by You Young-chang, fooji (FALinux Co.,Ltd)
Last Modify Jul 5 2005
```

<NFS에서 b50이라는 파일을 카피 하는 도중 리셋을 누른다.>

2. 리셋 후 다시 SFFS로 마운트 하면 데이터가 트랜잭션 단위로 처리되어서 쓰여지지 않은 것을 알 수 있다.



```
ez - SecureCRT
File Edit View Options Transfer Script Tools Help
ez 210.118.56.61 210.118.56.61 (1)

Welcome to FALinux EZ-X5 ( www.falinux.com )
XScale(PXA255) Linux
ez-x5 login: root
[root@ez-x5 /root]$ cd /
[root@ez-x5 /]$ mkdir data
[root@ez-x5 /]$ mount -t nfs -o nolock 210.118.56.61:/nfs /mnt/nfs
[root@ez-x5 /]$ insmod /mnt/nfs/sffs_delay.o
Using /mnt/nfs/sffs_delay.o
[root@ez-x5 /]$ mount -t sffs /dev/mtdblock2 data
mount : sffs_internal_read_super

MTD::WRITE

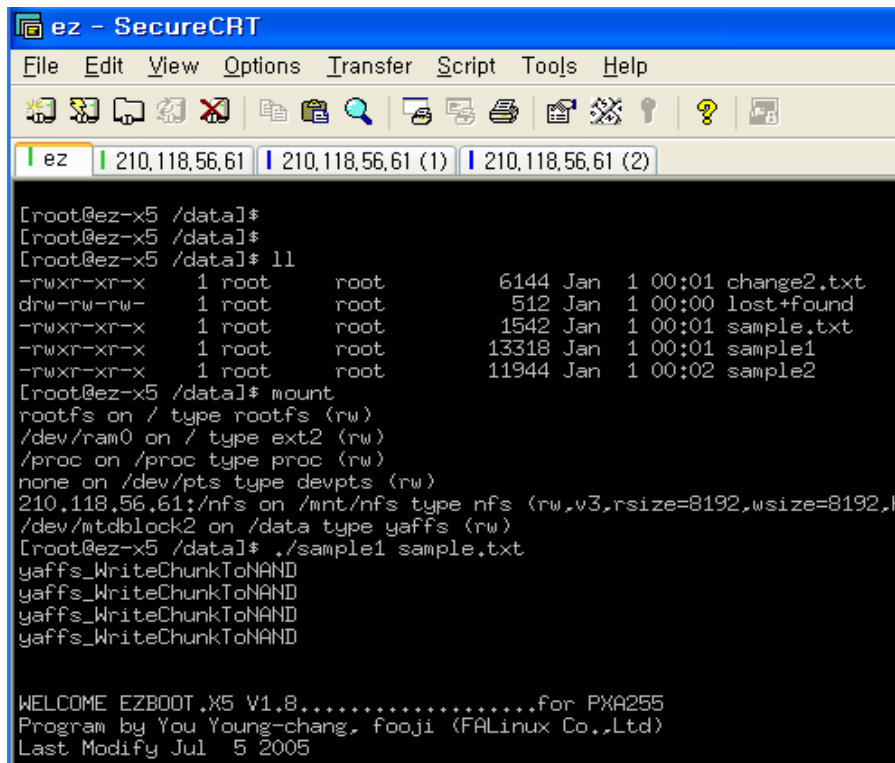
MTD::WRITE

[root@ez-x5 /]$ cd data
[root@ez-x5 /data]$ ll
drw-rw-rw- 1 root root 512 Jan 1 00:00 lost+found
[root@ez-x5 /data]$
[root@ez-x5 /data]$
[root@ez-x5 /data]$
[root@ez-x5 /data]$
```

<리셋 후 SFFS로 다시 마운트를 한 상태.>

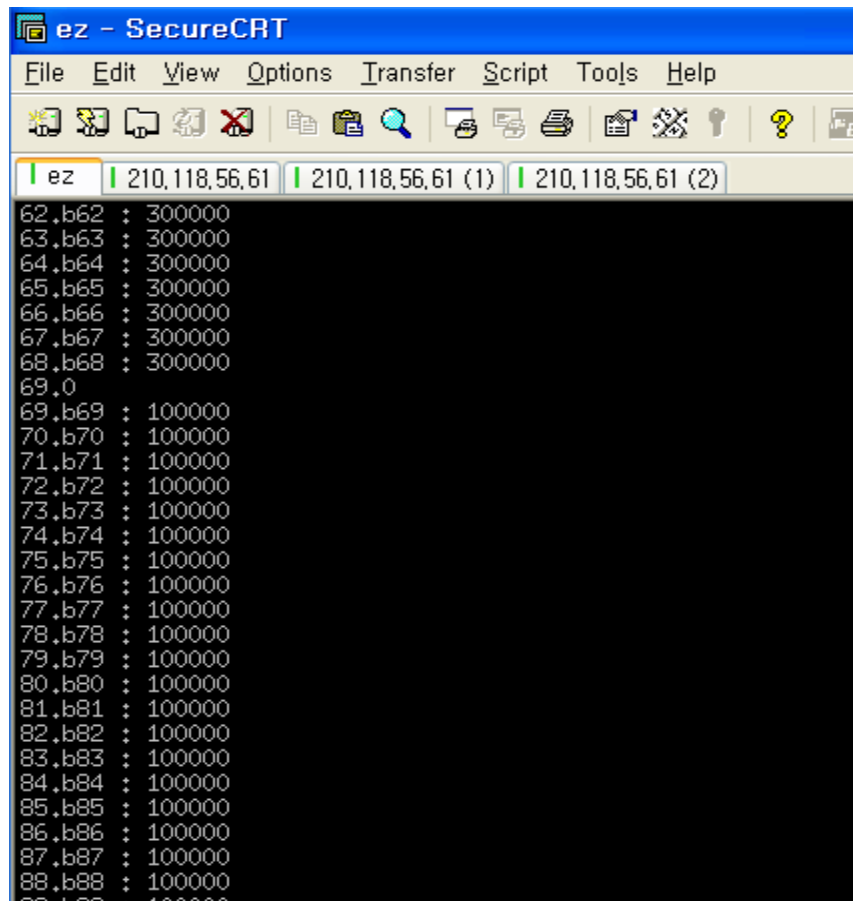
3.5. 트랜잭션 처리 테스트(YAFFS) – 파일 수정하기

1. 샘플 프로그램 1 적용



<샘플1 프로그램을 실행시킨 후 중간에 리셋한다.>

2. 리셋 후 다시 마운트 해서 데이터를 확인한다.



<데이터가 일부만 수정되어있어서 무결성을 보장하지 못한다.>

3. 샘플 프로그램 2 적용

The screenshot shows a terminal window titled "ez - SecureCRT". The menu bar includes File, Edit, View, Options, Transfer, Script, Tools, and Help. The toolbar contains various icons for file operations. The terminal output shows a root user at the prompt [root@ez-x5 /data]#.

```
[root@ez-x5 /data]# ll
-rwxr-xr-x    1 root    root           6144 Jan  1 00:01 change2.txt
drw-rw-rw-    1 root    root           512 Jan  1 00:00 lost+found
-rwxr-xr-x    1 root    root        11944 Jan  1 00:01 sample2
[root@ez-x5 /data]# mount
rootfs on / type rootfs (rw)
/dev/ram0 on / type ext2 (rw)
/proc on /proc type proc (rw)
none on /dev/pts type devpts (rw)
210.118.56.61:/nfs on /mnt/nfs type nfs (rw,v3,rsize=8192,wsiz=8192,h
/dev/mtdblock2 on /data type yaffs (rw)
[root@ez-x5 /data]# ./sample2 change2.txt
yaffs_WriteChunkToNAND
yaffs_WriteChunkToNAND
yaffs_WriteChunkToNAND
yaffs_WriteChunkToNAND
```

At the bottom of the terminal, a welcome message is displayed:

```
WELCOME EZBOOT.X5 V1.8.....for PXA255
Program by You Young-chang, fooji (FALinux Co.,Ltd)
Last Modify Jul  5 2005
```

<샘플2 프로그램을 실행시킨 후 중간에 리셋한다.>

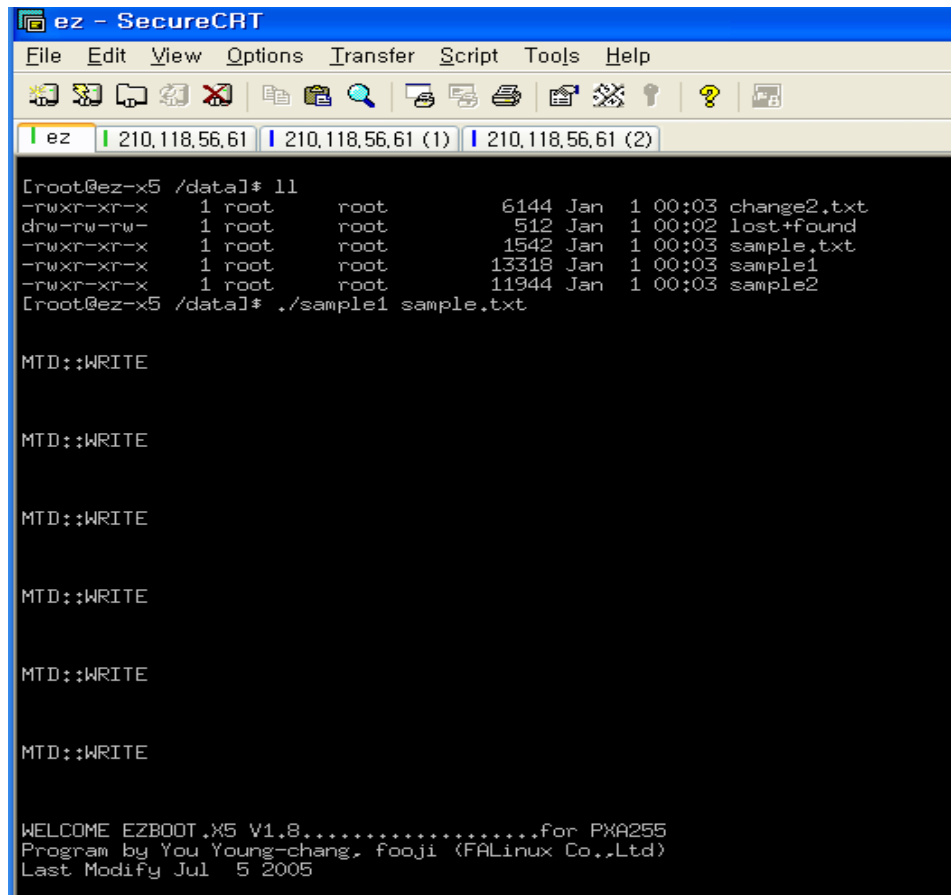
4. 리셋 후 다시 마운트 해서 데이터를 확인한다.

[illegible]

<데이터가 일부만 수정되어있어서 무결성을 보장하지 못한다.>

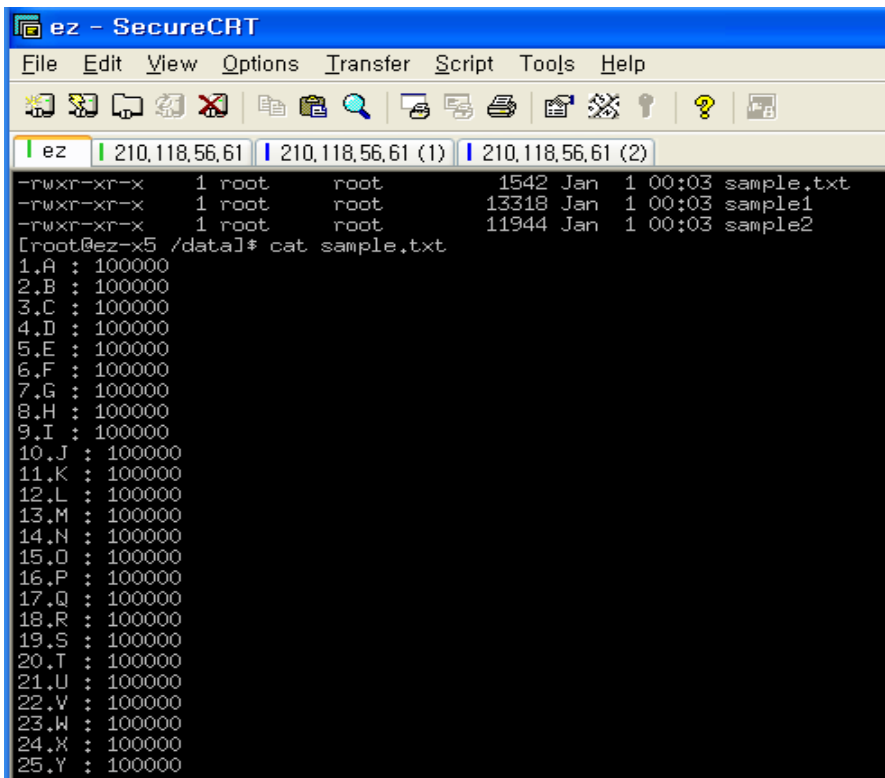
3.6. 트랜잭션 처리 테스트(SFFS) – 파일 수정하기

1. 샘플 프로그램 1 적용



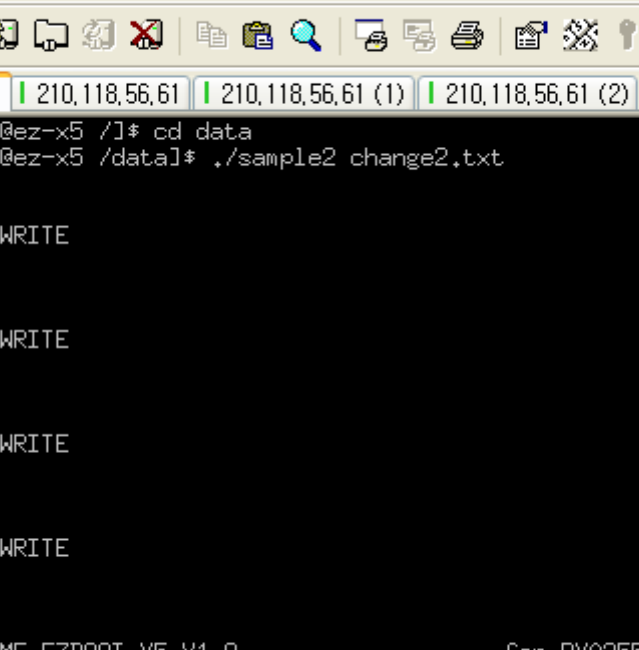
<샘플1 프로그램을 실행시킨 후 중간에 리셋한다.>

2. 리셋 후 다시 마운트 해서 데이터를 확인한다.



<데이터가 수정하기 전의 상태로 롤백되어 있다.>

3. 샘플 프로그램 2 적용



The screenshot shows a SecureCRT terminal window with a blue title bar labeled "ez - SecureCRT". The menu bar includes File, Edit, View, Options, Transfer, Script, Tools, and Help. The toolbar contains various icons for file operations and terminal control. The terminal session is as follows:

```

| ez | 210.118.56.61 | 210.118.56.61 (1) | 210.118.56.61 (2)
[root@ez-x5 /]$ cd data
[root@ez-x5 /data]$ ./sample2 change2.txt

MTD::WRITE

MTD::WRITE

MTD::WRITE

MTD::WRITE

WELCOME EZBOOT.X5 V1.8.....for PXA255
Program by You Young-chang, fooji (FALinux Co.,Ltd)
Last Modify Jul 5 2005

```

<샘플2 프로그램을 실행시킨 후 중간에 리셋한다.>

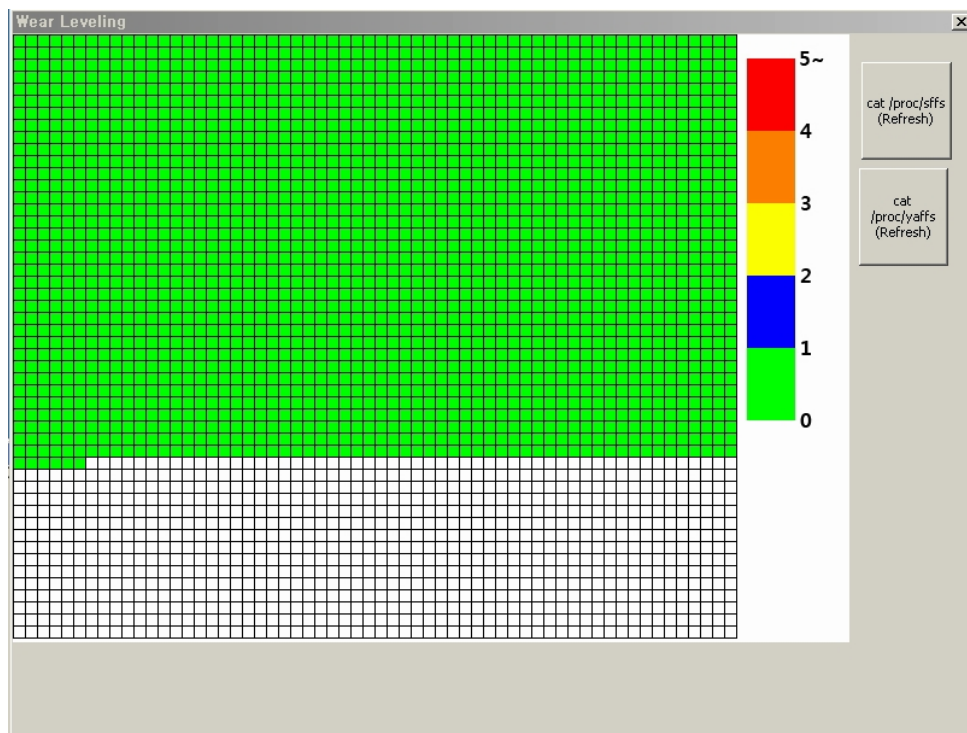
4. 리셋 후 다시 마운트 해서 데이터를 확인한다.

[illegible]

<데이터가 수정하기 전의 상태로 롤백되어 있다.>

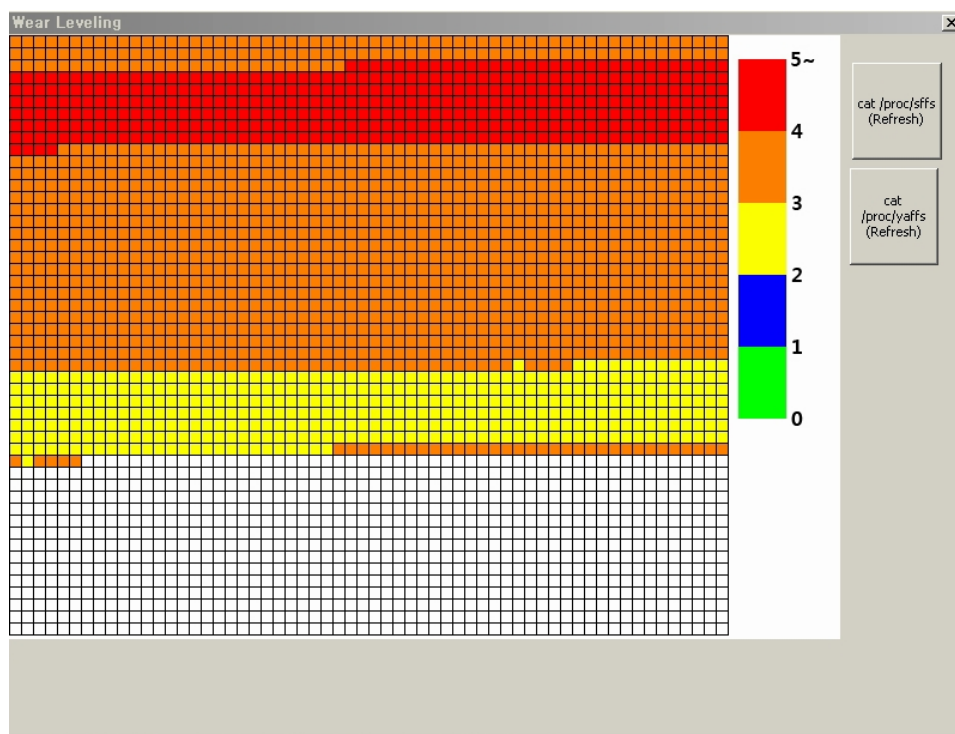
3.7.2. 마모도 평준화 테스트

1. YAFFS를 처음 마운트 한 후의 초기상태.



<proc 파일시스템에서 낸드 플래시 메모리의 사용빈도 값을 읽어서 GUI툴로 나타낸다.>

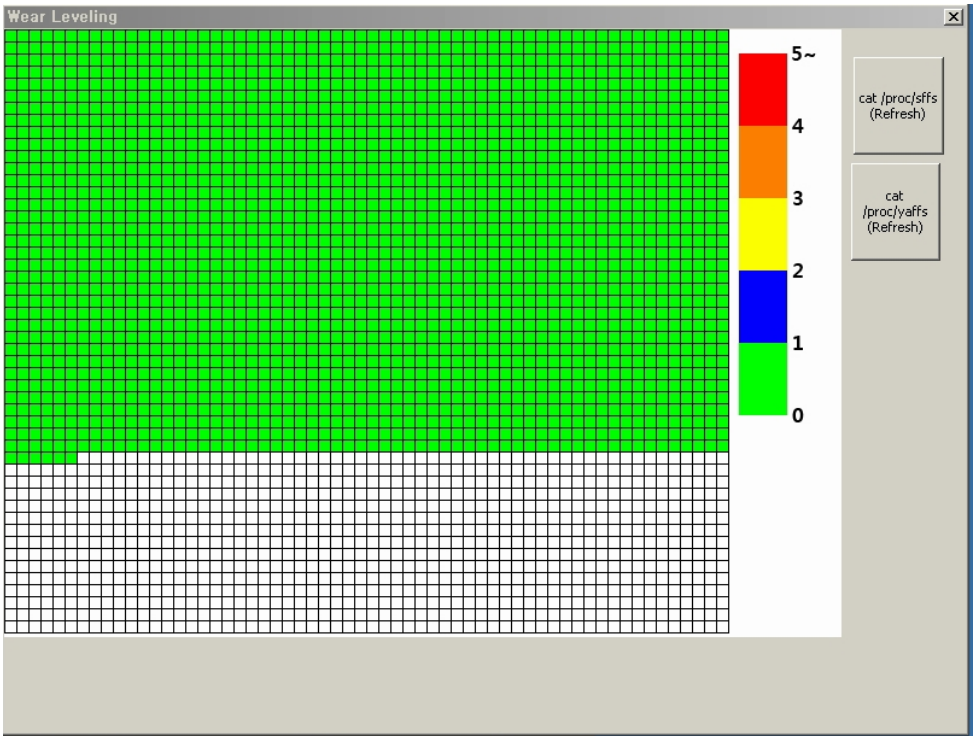
2. 파일의 쓰기와 지우기를 반복한 후의 블록 사용상태.



<사용 레벨이 일정하지 않고 특정 부분만 레벨이 더 높은 것을 확인할 수 있다.>

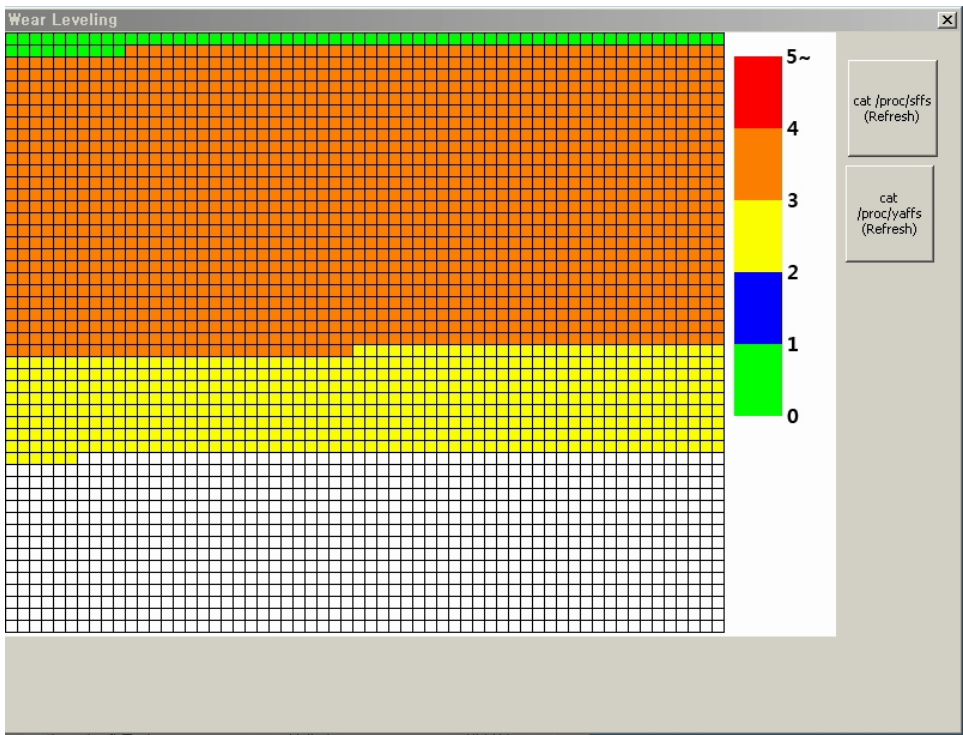
3.8. 마모도 평준화(Wear Leveling : SFFS) 테스트

1. SFFS를 처음 마운트 한 후의 초기상태



<proc 파일시스템에서 낸드 플래시 메모리의 사용빈도 값을 읽어서 GUI툴로 나타낸다.>

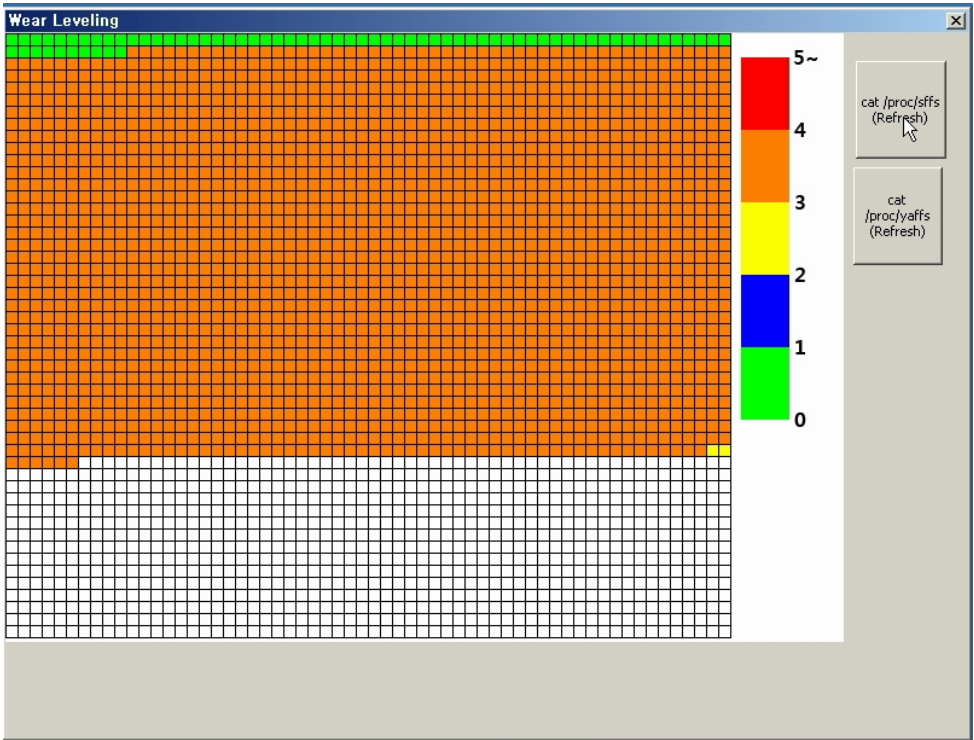
2. 파일의 쓰기와 지우기를 반복한 후의 블록 사용상태



<마모도 평준화가 적용되서 레벨이 일정함을 알 수 있다.>

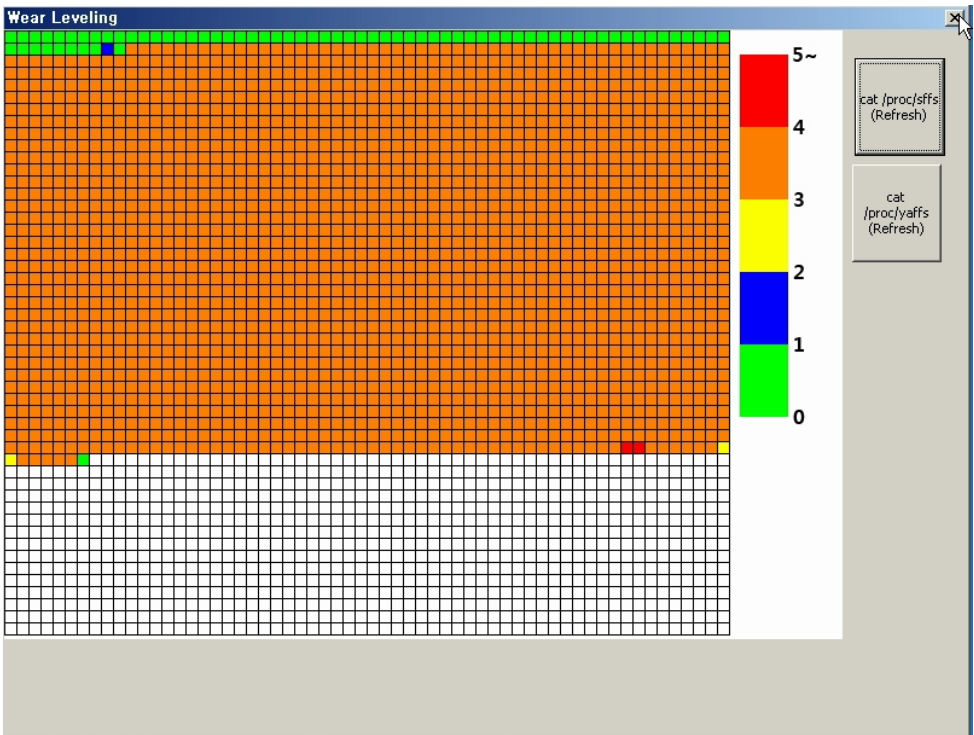
3.9. 지속적인 마모도 평준화(Wear Leveling) 테스트

- 1. SFFS에서 쓰기와 지우기를 반복하다가 umount한다.



<umount하기 전의 마지막 블록 상태를 나타낸다.>

- 2. 파일시스템을 다시 mount 한 후 블록의 사용빈도를 확인해본다. 다시 읽었을 때 umount할때의 블록사용 빈도가 그대로 남아있다. YAFFS에서는 지원을 하지 않는 기능으로 이 방법으로 낸드 플래시 메모리의 수명을 늘릴 수 있다.



<다시 마운트 했을 때의 블록 상태를 나타낸다.>

3.10. LTP(Linux Test Project)의 fsstress 테스트

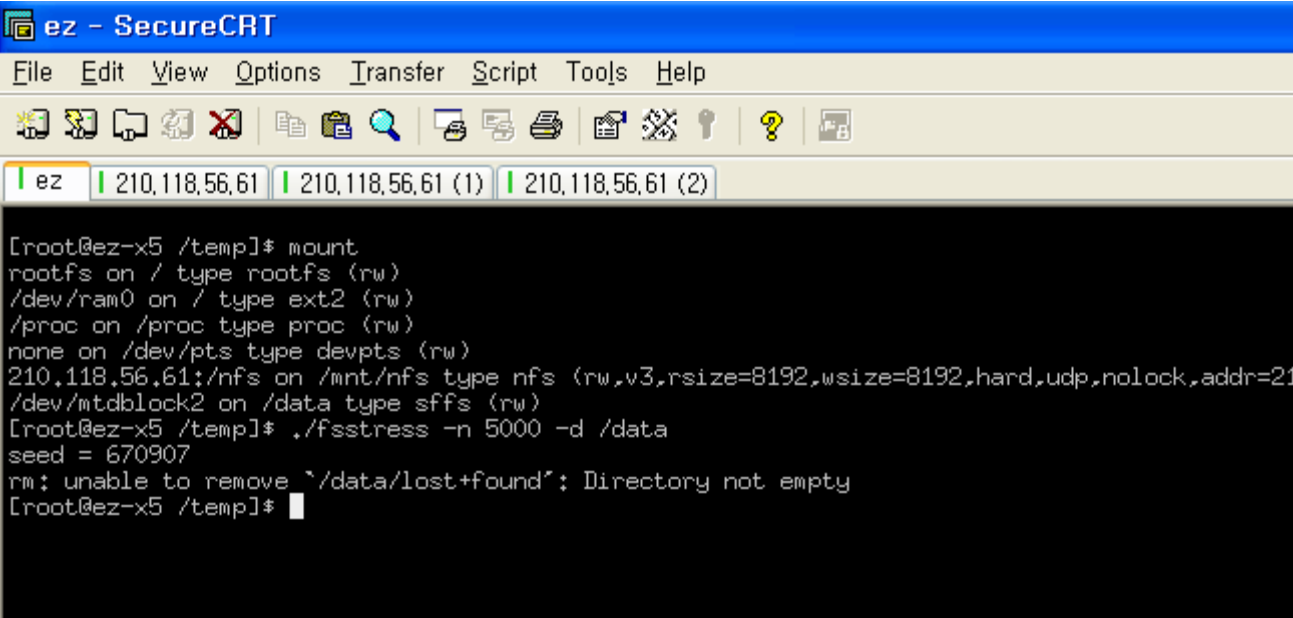
3.10.1. Linux Test Project란?

LTP 는 오픈소스 프로젝트인 리눅스의 안정성을 검사하기 위한 프로젝트로 파일시스템 외에도 네트워크 등의 커널의 전반적인 검사를 수행하여 리눅스 커널의 안정성을 보장해주는 프로젝트이다.

92 년부터 IBM 이나 SGI 등의 회사가 참여하여 만든 오픈 프로젝트로 LTP 를 통과하면 리눅스쪽에서는 안정성이 검증된다고 할 수 있다.(<http://ltp.sourceforge.net>)

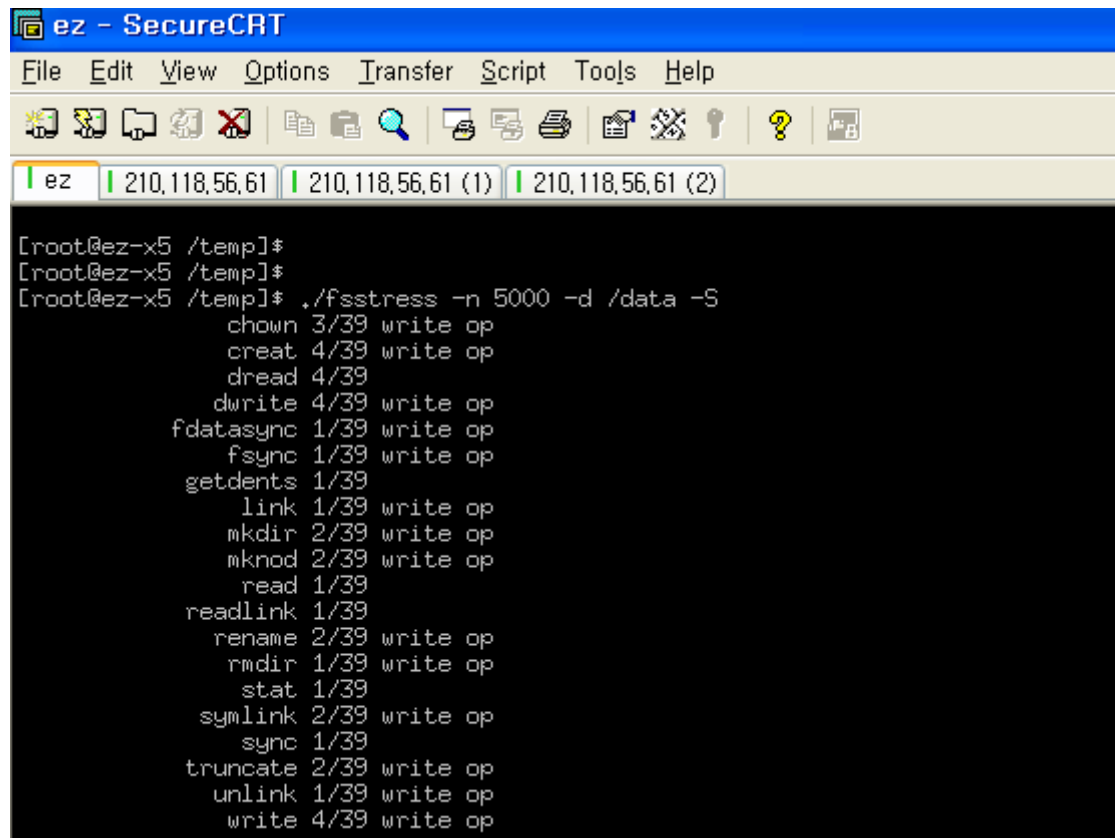
SFFS 는 LTP 내의 파일시스템 테스트 시나리오 중 하나인 fsstress 를 적용했다. fsstress 는 파일시스템의 일반적인 동작(create, read, write, mkdir, etc..)을 랜덤하게 정해진 숫자만큼 반복하여 시스템의 임의적인 부하를 줌으로서 안정성을 검사한다

3.10.2. fsstress테스트 결과



```
ez - SecureCRT
File Edit View Options Transfer Script Tools Help
ez 210.118.56.61 210.118.56.61 (1) 210.118.56.61 (2)
[root@ez-x5 /temp]$ mount
rootfs on / type rootfs (rw)
/dev/ram0 on / type ext2 (rw)
/proc on /proc type proc (rw)
none on /dev/pts type devpts (rw)
210.118.56.61:/nfs on /mnt/nfs type nfs (rw,v3,rsize=8192,wsiz=8192,hard,udp,nolock,addr=210.118.56.61)
/dev/mtdblock2 on /data type sffs (rw)
[root@ez-x5 /temp]$ ./fsstress -n 5000 -d /data
seed = 670907
rm: unable to remove `/data/lost+found': Directory not empty
[root@ez-x5 /temp]$
```

<5000번의 반복되는 테스트를 수행한 결과.>



The screenshot shows a terminal window titled "ez - SecureCRT". The menu bar includes File, Edit, View, Options, Transfer, Script, Tools, and Help. The toolbar contains various icons for file operations and search. The terminal shows a root user at a machine named "ez" in the directory "/temp". The command executed is `./fsstress -n 5000 -d /data -S`. The output lists 39 system calls with their respective counts and operation types.

```
[root@ez-x5 /temp]$  
[root@ez-x5 /temp]$  
[root@ez-x5 /temp]$ ./fsstress -n 5000 -d /data -S  
    chown 3/39 write op  
    creat 4/39 write op  
    dread 4/39  
    dwrite 4/39 write op  
    fdatsync 1/39 write op  
    fsync 1/39 write op  
    getdents 1/39  
    link 1/39 write op  
    mkdir 2/39 write op  
    mknod 2/39 write op  
    read 1/39  
    readlink 1/39  
    rename 2/39 write op  
    rmdir 1/39 write op  
    stat 1/39  
    symlink 2/39 write op  
    sync 1/39  
    truncate 2/39 write op  
    unlink 1/39 write op  
    write 4/39 write op
```

<세부적으로 파일 I/O와 관련된 시스템 콜들로 테스트를 한다.>