

ZNS+: Advanced Zoned Namespace Interface for Supporting In-Storage Zone Compaction

1. How to download Artifact

- 1) FEMU source code
 - ◆ Link: http://nyx.skku.ac.kr/osdi/femu_artifact.tar.gz
- 2) Virtual machine image
 - ◆ It includes O/S kernel source code and evaluation script.
 - ◆ Link: nyx.skku.ac.kr/osdi/u14s.qcow2.tar.gz

2. Getting Started Instructions

- 1) Requirement
 - ◆ Computer with 4 or more CPUs: Allocate 4 cores to guest machine
 - ◆ More than 34GB DRAM: 32GB memory for femu heap that is emulated as SSD capacity and 2 GB memory for guest machine
- 2) Compile FEMU & launch guest machine
 - ◆ FEMU compile
 - Make sure you have installed necessary libraries for building QEMU.
 -
 - Switch to the FEMU building directory

```
# cd femu/build-femu
```
 - only Debian/Ubuntu based distributions supported

```
# sudo ./pkgdep.sh
```
 - Compile & Install FEMU

```
# ./femu-compile.sh
```

FEMU binary will appear as ``x86_64-softmmu/qemu-system-x86_64``
 - ◆ Run FEMU
 - Run the VM using Run-blackbox.sh

```
# cd femu/build-femu
```

```
# ./run-blackbox.sh path-to-qcow2-image path-to-status-file
```

Ex) `./run-blackbox.sh /images /home/artifact/result/test.txt`
 - connect to guest machine
Launch a new terminal

```
# cd femu/build-femu
```

```
# ./conn.sh
```
 - ID/PW of guest machine

- ID: femu
- password: 1

3) Compile O/S & Launch evaluation scripts

◆ Compile guest machine's O/S

- Switch to the O/S building directory
 - # *cd /home/femu/ZNSPlus*
- Configure file system options. (It will be explained later)
 - # *make menuconfig*
- Compile O/S kernel
 - # *make;make modules;make modules_install;make install*
- Restart VM

◆ Launch evaluation script

- Switch to the home directory
 - # *cd /home/femu*
 - Run script.sh
 - # *./script.sh result-file-name*
- Ex) *./fileserver.sh test* (The result file is stored in /home/femu/result/.)

4) How to Configure FEMU (at the host machine)

◆ Hardware configuration

- Select base configuration file in /femu/build-femu/conf
- Copy base configuration file to /femu/build-femu/vssd1.conf
 - # *cp path-to-base-conf-file /femu/build-femu/vssd1.conf*
- Configuration file

```

FILE_NAME_HDA      ../../RAMDISK/rd/ssd_hda.img
FILE_NAME_HDB      ../../RAMDISK/rd/ssd_hdb.img
PAGE_SIZE          16384
PAGE_NB            128
SECTOR_SIZE        512
FLASH_NB           16
BLOCK_NB           8192
PLANES_PER_FLASH   1

LOG_RAND_BLOCK_NB  0
LOG_SEQ_BLOCK_NB   0

REG_WRITE_DELAY    24000
CELL_PROGRAM_DELAY 390000
REG_READ_DELAY     24000
CELL_READ_DELAY    35000
BLOCK_ERASE_DELAY  4000000

CHANNEL_SWITCH_DELAY_R  16
CHANNEL_SWITCH_DELAY_W  33

IO_PARALLELISM      0

WRITE_BUFFER_FRAME_NB 2048
READ_BUFFER_FRAME_NB  2048
CACHE_IDX_SIZE       10

CHANNEL_NB           8
OVP                  0
c MODE 2

```

- We only modify FLASH_NB and CHANNEL_NB in the base configuration file.

◆ Software configuration

- Change header file (/femu/hw/block/ssd/vssim_config_manager.h)

```

#ifndef CONFIG_MANAGER_H
#define CONFIG_MANAGER_H

#include <stdint.h>
#include "common.h"

#define NUM_FLASH (16) //2 4 8 16 32

#define IOUB_DO_COPYBACK
//#define BACK_PD_OFF
//#define PRE_PD_OFF

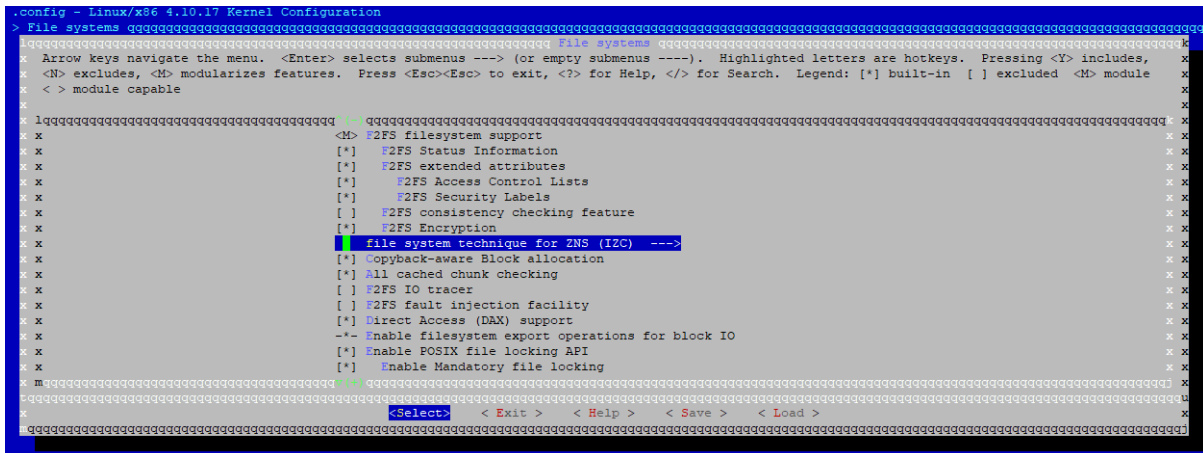
```

- NUM_FLASH: It means the number of flashes in the SSD.
 - ◆ If FLASH_NB in the configuration file is modified, NUM_FLASH in vssim_config_manager.h should be modified to the same value.
- BACK_PD_OFF: This option determines whether the SSD uses back-padding or not.
- PRE_PD_OFF: This option determines whether the SSD uses pre-padding or not.

- Compile FEMU
 - # cd /femu/build_femu
 - # ./femu-compile.sh

5) How to configure F2FS file system (at the guest machine)

- ◆ Configure file system options. (It will be explained later)
 - # make menuconfig
- ◆ Options for ZNS+ (File systems >>)



- file system technique for ZNS: Select technique
 - ZNS: Baseline technique
 - IZC: This technique uses internal zone compaction scheme.
 - HSR: This technique uses hybrid segment recycling scheme.
- Copyback-aware Block allocation: This option determines whether the F2FS uses CAB or not.
- All cached chunk checking: This option determines whether the F2FS uses IZC-H

or not.

6) Result file of Guest machine

- ◆ Benchmark result: it is stored as `"/home/femu/result/benchmark_keyword.txt"`

Ex) `./fileservers.sh test >> fileserver_test.txt`

- ◆ Filesystem result: it is stored as `"/home/femu/result/benchmark_f2fs_keyword.txt"`

Ex) `./fileservers.sh test >> fileserver_f2fs_test.txt`

```
=====[ partition info(nvme0n1p1). #0, RW]====
[SB: 1] [CP: 2] [SIT: 2] [NAT: 36] [SSA: 24] [MAIN: 8112(OverProv:1135 Resv:768)]

Utilization: 90% (3223903 valid blocks)
- Node: 95878 (Inode: 95878, Other: 0)
- Data: 3128025
- Inline_xattr Inode: 0
- Inline_data Inode: 72
- Inline_dentry Inode: 2582
- Orphan Inode: 0

Main area: 8112 segs, 507 secs 507 zones
- COLD data: 5196, 324, 324
- WARM data: 4992, 312, 312
- HOT data: 8048, 503, 503
- Dir dnode: 7803, 487, 487
- File dnode: 7731, 483, 483
- Indir nodes: 8064, 504, 504

- Valid: 2132
- Dirty: 5100
- Prefree: 0
- Free: 880 (51)

CP calls: 1742 (BG: 0)
SC calls: 3000 (BG: 0)
- data segments : 37552 (0)
- node segments : 10440 (0)
Try to move 16986024 blocks (BG: 0)
- data blocks : 13302084 (0)
- node blocks : 3683940 (0)

Extent Cache:
- Hit Count: L1-1:960871 L1-2:1767 L2:12724
- Hit Ratio: 26% (975362 / 3713118)
- Inner Struct Count: tree: 38400(0), node: 8723

Balancing F2FS Async:
- inmem: 0, wb_cp_data: 0, wb_data: 0
- nodes: 5 in 87697
- dents: 0 in dirs: 0 ( 531)
- datas: 23836 in files: 0
- meta: 0 in 8885
- imeta: 2
- NATs: 6/ 93855
- SITs: 11/ 8112
- free_nids: 3629, alloc_nids: 0

Distribution of User Blocks: [ valid | invalid | free ]
[-----|-----|]

IPU: 0 blocks
SSR: 0 blocks in 0 segments
LFS: 28276347 blocks in 55226 segments

BDF: 60, avg. vblocks: 6949
```

Stats of garbage collection

```

Hot data section LFS allocation: 0
Hot data section SSR allocation: 0
Warm data section LFS allocation: 782
Warm data section SSR allocation: 0
Cold data section LFS allocation: 1584
Cold data section SSR allocation: 0
Hot node section LFS allocation: 48
Hot node section SSR allocation: 0
Warm node section LFS allocation: 582
Warm node section SSR allocation: 0
Cold node section LFS allocation: 0
Cold node section SSR allocation: 0
Total LFS allocation: 2996
Total SSR allocation: 0

```

Stats of Segments allocation

```

Hot data LFS block write: 1
Hot data SSR block write: 0
Warm data LFS block write: 6400510
Warm data SSR block write: 0
Cold data LFS block write: 12981289
Cold data SSR block write: 0
Hot node LFS block write: 392867
Hot node SSR block write: 0
Warm node LFS block write: 4768429
Warm node SSR block write: 0
Cold node LFS block write: 0
Cold node SSR block write: 0
Total LFS block write: 24543096
Total SSR block write: 0

```

```

Data write count: 6400511
Node write count: 1576926
Meta write count: 426491

```

Stats of write counts

```

Host I/O: 1265469
Orig_cpbk 168780
IM_cached_pages: 1301196
IM_all_cached_pages: 1192523
IM - data: 37961 - 11715820
DEV R/W: 1580045
DEV CPBK: 10081240

```

Stats of the IZC

```

GC time breakdown
DATA - 401589706528 (tick)
init phase: 142226255890 (tick)
IZC phase: 119933426744 (tick)
checkpoint: 139430023894 (tick)

```

Stats of garbage collection time

7) Result file of Host machine

- ◆ FEMU result file: It is saved in the path specified.

```

NAND_read      5059840 chunks (16KB)
NAND_write     4537269 chunks (16KB)
NAND_cpbk      3498552 chunks (16KB)
IZC_count      825552 LPNs (4KB)
IZC_read       64248 chunks (16KB)
IZC_write      174696 LPNs (4KB)
IZC_copyback   650788 LPNs (4KB)
IZC_avg. time  2868662 ns
SSR_PDcount    14969154 LPNs (4KB)
SSR_PDR&W     1625226 LPNs (4KB)
SSR_PDcpbk    13343928 LPNs (4KB)
SSR_BP         7043066 LPNs (4KB)
SSR_PP         1659300 LPNs (4KB)
SSR_FD         6266788 LPNs (4KB)
CH_util        10.817225 chips/s

```

3. Detailed Instructions

1) Notices

- A. Since FEMU is an emulation environment using VMs, performance variation may occur.
Run the experiment multiple times and use the average value of the results.

2) Average compaction time (Figure 6)

- A. After setting the "file system technique for ZNS" option of the Guest O/S kernel to ZNS, compile and install the kernel. (Baseline technique)
- B. Execute all benchmarks and collect the status result of Guest O/S and Host O/S.
- C. After setting the "file system technique for ZNS" option of the Guest O/S kernel to IZC, compile and install the kernel. (IZC technique)
- D. Execute all benchmarks and collect the status result of Guest O/S and Host O/S.
- E. Make a graph by referring to "GC time breakdown" field of each F2FS status result file.

3) Effect of the threaded logging support (Figure 7)

- A. After setting the "file system technique for ZNS" option of the Guest O/S kernel to HSR, compile and install the kernel. (HSR technique)
- B. Execute all benchmarks and collect the status result of Guest O/S and Host O/S.
- C. Figure (a): Make a graph by referring to "ops/s" of each benchmark result.

- i. Fileserver: `#cat result/fileserver_test.txt | grep IO`

```
325.271: IO Summary: 3155172 ops 10513.689 ops/s 956/1912 rd/wr 251.8mb/s 4.741ms/op
```

- ii. Varmail: `#cat result/varmail_test.txt | grep IO`

```
332.296: IO Summary: 3246908 ops 10821.320 ops/s 1665/1665 rd/wr 66.1mb/s 1.477ms/op
```

- iii. Tpc: `#cat result/tpcc_test.txt | grep TpmC`

```
<TpmC>  
6603.400 TpmC
```

- iv. YCSB-a: `#cat result/yCSBA_text.txt | grep ops`

```
[OVERALL], Throughput (ops/sec), 4084.033064331689
```

- v. YCSB-f: `#cat result/yCSBF_text.txt | grep ops`

```
[OVERALL], Throughput (ops/sec), 3811.476737604601
```

- D. Figure (b): Make a graph by referring to "Stats of write count" field of each F2FS status result file.

- i. Node/Meta write count is normalized to data write count.

- 1. Ex. `#cat result/fileserver_f2fs_test.txt | grep "write count"`

```
Data write count: 0  
Node write count: 0  
Meta write count: 0  
Data write count: 10170623  
Node write count: 1477426  
Meta write count: 213464
```

- 2. Node: $1477426 / 10170623 = 14.5\%$, Meta: $213464 / 10170623 = 2.1\%$

4) File system utilization (Figure 8)

A. Measure the performance of each technique while changing the file system utilization.

- i. How to change file system utilization: Change /home/femu/fileserver.f to /home/femu/fileserver_conf/utilization/*.f. (80.f, 85.f, 90.f, 95.f)

B. Make a graph by referring to "MB/s" of each benchmark result.

- i. Ex. `#cat result/fileserver_test.txt | grep IO`

```
325.271: IO Summary: 3155172 ops 10513.689 ops/s 956/1912 rd/wr 251.8mb/s 4.741ms/op
```

C. How to calculate WAF value

- i. A: Data Write Count field on f2fs stat file (User write)
- ii. B: Node Write Count field on f2fs stat file (Node write)
- iii. C: Meta Write Count field on f2fs stat file (Meta write)

1. Ex. `#cat result/fileserver_f2fs_test.txt | grep "write count"`

```
Data write count: 0
Node write count: 0
Meta write count: 0
Data write count: 10019106
Node write count: 1456757
Meta write count: 212658
```

- iv. D: Try to move X blocks field on f2fs stat file (GC cost)

1. Ex. `#cat result/fileserver_f2fs_test.txt | grep "Try to move"`

```
Try to move 0 blocks (BG: 0)
Try to move 776081 blocks (BG: 0)
```

- v. E: SSR_PDcount field on femu stat file (TL cost)

1. Ex. `#cat test.txt | grep "SSR_PDcount"`

A. `test.txt` is in host PC (second parameter of run-blackbox.sh)

```
SSR_PDcount 508 LPNs (4KB)
SSR_PDcount 508 LPNs (4KB)
SSR_PDcount 16457315 LPNs (4KB)
```

- vi. $WAF = (A+B+C+D+E) / A$

1. Ex. $(10019106 + 1456757 + 212658 + 776081 + 16457315) / 10019106 = 2.88$

5) Flash chip utilization (Figure 9)

A. Compile FEMU after uncommenting "#define BACK_PD_OFF" and "#define PRE_PD_OFF" in the vssim_config_manager.h file. (HSR-FP technique)

```
#ifndef _CONFIG_MANAGER_H_
#define _CONFIG_MANAGER_H_

#include <stdint.h>
#include "common.h"

// #define DEBUG_CHIP_UTIL

#define NUM_FLASH (16) // 2 4 8 16 32

#define IOUB_DO_COPYBACK
#define BACK_PD_OFF
#define PRE_PD_OFF
```

B. Execute all benchmarks and collect the status result of Guest O/S and Host O/S.

- C. Compile FEMU after uncommenting `"#define PRE_PD_OFF"` in the `vssim_config_manager.h` file. (HSR-BP technique)

```
#ifndef CONFIG_MANAGER_H
#define CONFIG_MANAGER_H

#include <stdint.h>
#include "common.h"

//#define DEBUG_CHIP_UTIL

#define NUM_FLASH (16) //2 4 8 16 32

#define IOUB_DO_COPYBACK
//#define BACK_PD_OFF
#define PRE_PD_OFF
```

- D. Execute all benchmarks and collect the status result of Guest O/S and Host O/S.
- E. The HSR technique collected in Figure 7 and the HSR-BP/PP technique in Figure 9 are the same.
- F. Make a graph by referring to "CH_util" field of each femu stat file.

- i. Ex. `#cat test.txt | grep CH_util`

```
CH_util    0.015142 chips/s
CH_util    11.842469 chips/s
CH_util    11.623443 chips/s
```

- ii. Chip utilization: $11.623443 / 16 = 73\%$
1. divide CH_util by 16 (chip count)

6) Flash chip utilization for fileserver workload (Figure 10)

- A. Compile FEMU after uncommenting `"#define DEBUG_CHIP_UTIL"` in the `vssim_config_manager.h` file.
- i. The modified FEMU outputs the chip utilization to the femu stat file every 0.1 seconds.
 - ii. Be careful not to run the VM for too long in this configuration.
- B. Perform the fileserver benchmark in three techniques to create femu stat files.
- C. Using the collected chip utilization results, create graphs of "interval average" and "moving average".

i. Ex. following is a part of femu stat file

```
CH      15.250256      1278046188428909
CH      15.537422      1278046288470302
CH      13.854756      1278046388429788
CH      5.638811       1278046488430766
CH      13.146521      1278046588439252
CH      12.070990      1278046688427242
CH      14.009100      1278046788432890
CH      15.174713      1278046888431576
CH      15.259629      1278046988427470
CH      13.231326      1278047088510051
CH      12.926576      1278047188427986
CH      11.803555      1278047288429184
CH      9.643388       1278047388419054
CH      13.811185      1278047488430210
CH      12.723749      1278047588430281
CH      14.688970      1278047688425864
CH      12.993785      1278047788431745
CH      15.422937      1278047888428795
CH      13.703444      1278047988422533
CH      15.767397      1278048088432022
CH      12.316077      1278048188418582
CH      7.412500       1278048288418764
CH      14.044797      1278048388465019
CH      15.509167      1278048488428727
CH      15.248626      1278048588423320
CH      14.204922      1278048688427810
```

ii. 2nd column is CH_util value per 0.1s interval

1. CH_util / 16 is chip utilization, Ex. Chip utilization: $15.250256 / 16 = 95\%$
2. We plot chip utilization per 0.1s interval as "interval average", and average value during 5.0s as "moving average"

iii. following is a part of same femu stat file

```
NAND_read      737 chunks (16KB)
NAND_write     2564 chunks (16KB)
NAND_cpbk      0 chunks (16KB)
IZC_count      0 LPNs (4KB)
IZC_read       0 chunks (16KB)
IZC_write      0 LPNs (4KB)
IZC_copyback   0 LPNs (4KB)
IZC_avg. time  0
SSR_PDcount    508 LPNs (4KB)
SSR_PDR&W     0 LPNs (4KB)
SSR_PDcpbk    508 LPNs (4KB)
SSR_BP         0 LPNs (4KB)
SSR_PP         0 LPNs (4KB)
SSR_FD        508 LPNs (4KB)
CH_util        0.015142 chips/s
```

iv. Ssd status like upper is also stored in femu stat file

v. To using CH_util value that occurred only at workload runtime, import CH_util value between the last two ssd status prints

vi. Ex. Two ssd status are stored in femu stat file

```

A [ CH 15.250256 1278046188428909
    CH 15.537422 1278046288470302
    .
    .
    .
B [ NAND_read 737 chunks (16KB)
    NAND_write 2564 chunks (16KB)
    .
    .
    .
C [ CH 13.854756 1278046388429788
    CH 5.638811 1278046488430766
    CH 13.146521 1278046588439252
    .
    .
    .
D [ NAND_read 737 chunks (16KB)
    NAND_write 2564 chunks (16KB)
    .
    .
    .
E [ CH 13.146521 1278046588439252
    CH 12.070990 1278046688427242
    CH 14.009100 1278046788432890
    CH 15.174713 1278046888431576
  
```

vii. plot a graph using CH_util of C

7) Number of flash chips (Figure 11)

A. FEMU compile

- i. Modify "NUM_FLASH" in vssim_config_manager.h to 2~32.
- ii. Modify "FLASH_NB" and "CHANNEL_NB" in vssd1.conf. FLASH_NB is the same as NUM_FLASH. CHANNEL_NB is set to half of NUM_FLASH.
- iii. After compiling FEMU, run the guest machine.

B. O/S kernel compile

- i. Modify the value of the "NUM_FTL_BANKS" field in ZNSPlus/fs/f2fs/f2fs.h to the same value as NUM_FLASH.
- ii. Turn on/off "Copyback-aware Block allocation" in menuconfig. When this option is turned on, the CAB scheme is activated. (IZC-CAB / HSR-CAB)
- iii. When using CAB, modify page_addr of PD_aware_struct in ZNSPlus/fs/f2fs/gc.c as follows.
 1. NUM_FTL_BANKS==2 → unsigned long page_addr[NUM_FTL_BANKS][64]
 2. NUM_FTL_BANKS==4 → unsigned long page_addr[NUM_FTL_BANKS][32]
 3. NUM_FTL_BANKS==8 → unsigned long page_addr[NUM_FTL_BANKS][16]
 4. NUM_FTL_BANKS==16 → unsigned long page_addr[NUM_FTL_BANKS][16]
 5. NUM_FTL_BANKS==32 → unsigned long page_addr[NUM_FTL_BANKS][16]

```

struct PD_aware_struct {
#ifdef F2FS_CPBK_AWARE_GC
    unsigned long page_addr[NUM_FTL_BANKS][16];
    unsigned long count[NUM_FTL_BANKS];
#else
    unsigned long page_addr[128];
    unsigned long count;
#endif
    unsigned long single_page_count;
    unsigned long single_page_off[512];
};
#endif

```

- iv. Compile the kernel and install it.
- C. Modify mount.sh.
 - i. Modify the "-s option" of mkfs.f2fs in mount.sh to the same value as NUM_FLASH.
 - ii. If one of the values of A-C does not match, performance may not be properly measured, or the file system may not be mounted and the VM may crash.
- D. Measure the performance of each technique while changing the number of flash chips.
 - i. Change /home/femu/fileserver.f to /home/femu/fileserver_conf/number_of_chips/*.f. (2.f, 4.f, 8.f, 16.f, 32.f)
- E. Figure (a): Make a graph by referring to "MB/s" of each benchmark result.
 - i. same as figure 8
- F. Figure (b): Make a graph by referring to "IZC_write" and "IZC_copyback" field of each femu stat file.
 - i. Ex. `#cat test.txt | grep IZC_write`

```

IZC_write      0 LPNs (4KB)
IZC_write      0 LPNs (4KB)
IZC_write      1713188 LPNs (4KB)

```
 - ii. `#cat test.txt | grep IZC_copyback`

```

IZC_copyback   0 LPNs (4KB)
IZC_copyback   0 LPNs (4KB)
IZC_copyback   10611904 LPNs (4KB)

```
 - iii. Copyback ratio: $IZC_copyback / (IZC_write + IZC_copyback)$
 1. Ex. $10611904 / (1713188 + 10611904) = 86\%$
- G. For HSR mode, make a graph by referring to "SSR_PDcount" and "SSR_PDcpbk" field of each femu stat file.
 - i. Ex. `#cat test.txt | grep SSR_PDcount`

```

SSR_PDcount    508 LPNs (4KB)
SSR_PDcount    508 LPNs (4KB)
SSR_PDcount    5285796 LPNs (4KB)

```
 - ii. `#cat test.txt | grep SSR_PDcpbk`

```

SSR_PDcpbk     508 LPNs (4KB)
SSR_PDcpbk     508 LPNs (4KB)
SSR_PDcpbk     4818468 LPNs (4KB)

```
 - iii. Copyback ratio: $SSR_PDcpbk / SSR_PDcount$
 1. EX. $4818468 / 5285796 = 91\%$

- 8) NAND flash media (Table 3)
 - A. Copy the configuration file in /femu/build_femu/conf/ to /femu/build_femu/vssd1.conf. (TLC / MLC / ZNAND)
 - B. O/S kernel compile
 - i. Turn on/off "All cached chunk checking" in menuconfig. The technique that turned on this option is IZC-H, and the technique that turned off is IZC-D.
 - ii. Compile the kernel and install it.
 - C. Make a table by referring to "MB/s" of each benchmark result.
- 9) Threaded logging and pre-padding ratio (Table 4)
 - A. The rate of threaded logging is calculated using the f2fs result collected in Figure 7.
 - B. Rates of TL
 - i. A: "GC calls" field in f2fs stat file.
 1. Ex. `#cat result/fileserver_f2fs_test.txt | grep "GC calls"`

```
GC calls: 0 (BG: 0)
GC calls: 181 (BG: 0)
```
 - ii. B: "Total SSR allocation" field in f2fs stat file.
 1. Ex. `cat result/fileserver_f2fs_test.txt | grep "Total SSR allocation"`

```
Total SSR allocation: 0
Total SSR allocation: 3345
```
 - iii. Rates of TL: $B / (A + B)$
 1. Ex. $3345 / (181 + 3345) = 95\%$
 - C. Rates of PP
 - i. A: "SSR_PP" field in femu stat file
 1. Ex. `#cat test.txt | grep SSR_PP`

```
SSR_PP      0 LPNs (4KB)
SSR_PP      0 LPNs (4KB)
SSR_PP      1869248 LPNs (4KB)
```
 - ii. B: "SSR_PDcount" field in femu stat file
 1. Ex. `#cat test.txt | grep SSR_PDcount`

```
SSR_PDcount 508 LPNs (4KB)
SSR_PDcount 508 LPNs (4KB)
SSR_PDcount 16457315 LPNs (4KB)
```
 - iii. Rates of PP: A / B
 1. Ex. $1869248 / 16457315 = 11\%$