```c
/*
 * mmap() 함수와 low level function 을 이용한 file copy
 *
 *                lndr4 ("indra.kr"."\x40"."gmail.com")
 *                        http://indra.linuxstudy.pe.kr
 *                                              2003.05.27
 *                        (코멘트 수정일 : 2005. 10. 6.)
 *
 * bash-2.03# ./mmapcp src dst
 * file size:
 *  440825562 bytes.
 *  430493 KB.
 *  420 MB.
 * total 12.837420 seconds.
 * bash-2.03#
 */
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
#include <fcntl.h>
#include <sys/errno.h>
#include <sys/mman.h>

#define MAXMMAP     102400

#define ERR(funct, line) { \
    printf("[error] %s() function error. (line : %d)\n", \
        funct, line); \
}

// 시간 체크를 위한 루틴
struct timeval chkstart, chkend;
double init = 0.0;

double result()
{
    double start, end, tot;

    start = (double)chkstart.tv_sec + (double)chkstart.tv_usec/(1000*1000);
    end = (double)chkend.tv_sec + (double)chkend.tv_usec/(1000*1000);
    tot = end - start;
    init += tot;
    if(tot == 0.0) tot = .0001;
    return tot;
}

int start()
{
    gettimeofday(&chkstart, NULL);
}

int end()
{
    gettimeofday(&chkend, NULL);
}

int main(int argc, char *argv[])
{
    int     rfd,
            wfd;
    int     total = 0,
            remain = 0,
            size = 0;
    char    *src,
            *dst;

    if(argc != 3) {
        printf("Usage: %s <src-file> <dst-file>\n", argv[0]);
```

```
        exit(0);
    }

    if((rfd = open(argv[1], O_RDONLY)) == 0) {
        ERR("open", __LINE__);
        exit(-1);
    }

    size = lseek(rfd, 0, SEEK_END);
    remain = size;

    if((wfd = open(argv[2], O_RDWR|O_CREAT, 0600)) == 0) {
        ERR("open", __LINE__);
        exit(-1);
    }

    lseek(wfd, size - 1, SEEK_SET);

    if(write(wfd, " ", 1) != 1) {
        ERR("write", __LINE__);
        exit(-1);
    }

    start();

    for(;;) {
        if(remain < MAXMMAP) {
            src = mmap((char*)total, remain, PROT_READ, MAP_SHARED, rfd, 0);
            dst = mmap((char*)total, remain, PROT_READ|PROT_WRITE, \
                    MAP_PRIVATE, wfd, 0);
            if((int)src == ENODEV || (int)dst == ENODEV) {
                printf("This machine is not support to %s\n", argv[0]);
        exit(-1);
            }
            memcpy(dst, src, remain);
            munmap(src, remain);
            munmap(dst, remain);
            break;
        }
    else {
            src = mmap((char*)total, MAXMMAP, PROT_READ, MAP_SHARED, rfd, 0);
            dst = mmap((char*)total, MAXMMAP, PROT_READ|PROT_WRITE, \
                    MAP_PRIVATE, wfd, 0);
            if((int)src == ENODEV || (int)dst == ENODEV) {
                printf("This machine is not support to %s\n", argv[0]);
                exit(-1);
        }
            memcpy(dst, src, MAXMMAP);
            munmap(src, MAXMMAP);
            munmap(dst, MAXMMAP);
        }
        total += MAXMMAP;
        remain -= MAXMMAP;

        if(remain <= 0) remain += MAXMMAP;
    }

    end();

    printf("file size: \n");
    printf("\t%d bytes.\n", size);
    printf("\t%d KB.\n", size/1024);

    if(((size/1024)/1024) > 1) printf("\t%d MB.\n", (size/1024)/1024);

    printf("total %8.6f seconds.\n", result());
    close(rfd);
    close(wfd);
}
```