CSGE602055 Operating Systems
CSF2600505 Sistem Operasi
Week 10: I/O Programming & Network Sockets Programming

Rahmat M. Samik-Ibrahim

University of Indonesia

http://rms46.vlsm.org/2/207.html
Always check for the latest revision!

REV125 12-Mar-2018

Operating Systems 2018-1 (Room 3114 Tue/Thu)
Class: A (10:00-12:00) | B (13:00-15:00) | C (16:00-18:00)

Week	Schedule	Topic	OSC9
Week 00	06 Feb - 12 Feb 2018	Intro & Review1	Ch. 1, 16
Week 01	13 Feb - 19 Feb 2018	Review2 & Scripting	Ch. 1, 2
Week 02	20 Feb - 26 Feb 2018	Protection, Security, Privacy,	Ch. 14, 15
		& C-language	
Week 03	27 Feb - 05 Mar 2018	I/O, BIOS, Loader, & Systemd	Ch. 13
Week 04	06 Mar - 12 Mar 2018	Addressing, Shared Lib, & Pointer	Ch. 8
Week 05	13 Mar - 19 Mar 2018	Virtual Memory	Ch. 9
Reserved	20 Mar - 24 Mar 2018		
Mid-Term	03 Apr 2018	13:00 - 15:30 (UTS)	
Week 06	05 Apr - 11 Apr 2018	Concurrency: Processes & Threads	Ch. 3, 4
Week 07	12 Apr - 18 Apr 2018	Synchronization	Ch. 5, 7
Week 08	19 Apr - 25 Apr 2018	Scheduling	Ch. 6
Week 09	26 Apr - 07 May 2018	File System & Persistent Storage	Ch. 10, 11, 12
Reserved	08 May - 14 May 2018		
Week 10	15 May - 21 May 2018	I/O Programming	
		& Network Sockets Programming	
Reserved	22 May - 22 May 2018		
Final	23 May - 26 May 2018	(UAS)	
Deadline	07 Jun 2018 16:00	Extra assignment **deadline**	

- The Check List (Operating Systems)
  - ☐ **Starting Point**: http://rms46.vlsm.org/2/207.html
  - ☐ **Text Book**: any recent/decent OS book but map it to **OSC9**.
  - ☐ Create **public** project "os181" on your github.com account.
    - ☐ Create file "README.md" and add an extra line every week. For e.g.[1]:

      ZCZC Sistem Operasi 2018 Awal (1)
      ZCZC W01 Have tried demo for week 01.
      ZCZC W02 Week 02 is done.
      ZCZC W03 Week 03 is done.

  - ☐ Encode your **QRC** with image size of approximately 250x250 pixels:
    "OS181 CLASS ID GITHUB-ACCOUNT SSO-ACCOUNT SIAK-Full-Name"
    Special for Week 00: Mail your **embedded** QRC to: os181@vlsm.org
    with Subject: [W00] CLASS ID SIAK-NAME.
  - ☐ Write your Memo (with QRC) **every week**.
  - ☐ Using your **SSO** account, login to badak.cs.ui.ac.id via
    kawung.cs.ui.ac.id.
    - ☐ Check folder badak:///extra/Week00/
    - ☐ Every week, copy the weekly demo files to your own home directory.
      Eg. for Week00:
      cp -r /extra/Week00/W00-demos/ W00-demos/

---

[1]Week 00 line is optional. The following "ZCZC WXX" weekly tags are mandatory.

Agenda

- Reference: (demo-w10)

Body Knowledge

- 18 Knowledge Area (IEEE/ACM 2013)

AL - Algorithms and Complexity	AR - Architecture and Organization
CN - Computational Science	DS - Discrete Structures
GV - Graphics and Visualization	HCI - Human-Computer Interaction
IAS - Information Assurance and Security	IM - Information Management
IS - Intelligent Systems	NC - Networking and Communications
OS - Operating Systems	PBD - Platform-based Development
PD - Parallel and Distributed Computing	PL - Programming Languages
SDF - Software Development Fundamentals	SE - Software Engineering
SF - Systems Fundamentals	SP - Social Issues and Professional Practice

- OS - Operating Systems (IEEE/ACM 2013)
  - OS/Overview of Operating Systems (T1:2)
  - OS/Operating System Principles (T1:2)
  - OS/Concurrency (T2:3)
  - OS/Scheduling and Dispatch (T2:3)
  - OS/Memory Management (T2:3)
  - OS/Security and Protection (T2:2)
  - OS(Electives): Virtual Machines, Device Management, File Systems, Real Time and Embedded Systems, Fault Tolerance, System Performance Evaluation.

Issues

- General View (IEEE/ACM 2013)
- The Amnesia and Tabula Rasa Problem
- MEMOS: QRC
- EXAM: Open/Close/Memo
- LABS?
- The "Monkey Book"
- Weeks 0-9
- Week10?
- Week11 Extra

```
#include <stdio.h>

void main (void) {
    int cc = getchar();
    putchar(cc);
    putchar('\n');
}

>>>>> $ 50-get-put
x
x
>>>>> $ 50-get-put
abcde
a

=====================================
#include <stdio.h>

void main (void) {
    int cc;
    while((cc = getchar()) != EOF) {
        putchar(cc);
    }
}

>>>>> $ 51-get-put-loop
xxxx
xxxx
```

```
* === umask() ===
* int open(const char* pathname, int flags, mode_t mode);
* === FLAGS: ===
* O_RDONLY      Open the file so that it is read only.
* O_WRONLY      Open the file so that it is write only.
* O_RDWR        Open the file so that it can be read from and written to.
* O_APPEND      Append new information to the end of the file.
* O_TRUNC       Initially clear all data from the file.
* O_CREAT       If the file does not exist, create it.
                You must include the third parameter.
* O_EXCL        With O_CREAT: exists, the call will fail.
* === MODE ===
* S_IRWXU  00700 user (file  owner) has read, write and execute permission
* S_IRUSR  00400 user has read permission
* S_IWUSR  00200 user has write permission
* S_IXUSR  00100 user has execute permission
*
* S_IRWXG  00070 group has read, write and execute permission
* S_IRGRP  00040 group has read permission
* S_IWGRP  00020 group has write permission
* S_IXGRP  00010 group has execute permission
*
* S_IRWXO  00007 others have read, write and execute permission
* S_IROTH  00004 others have read permission
* S_IWOTH  00002 others have write permission
* S_IXOTH  00001 others have execute permission
```

```
#define FILE1 "demo-file1.txt"
#define FILE2 "demo-file2.txt"
#define FILE3 "demo-file3.txt"

#include <stdio.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>
void main(void) {
   char*  file1=FILE1;
   char*  file2=FILE2;
   char*  file3=FILE3;

   int fd; /* to hold a file descriptor */
   /* umask(0);    ******************* */
   fd = open (file1, O_CREAT | O_RDWR, S_IRWXU);
   close(fd);
   fd = open (file2, O_CREAT | O_RDWR, S_IRWXU|S_IRGRP|S_IWGRP|S_IROTH);
   close(fd);
   fd = open (file3, O_CREAT | O_RDWR, 0711);
   close(fd);
   fd = open (file3, O_CREAT | O_RDWR, 0700);
   close(fd);
}

>>>>> $ ls -al demo-file[234].txt
-rwxr--r-- 1 demo demo  0 Oct  5 17:49 demo-file2.txt
-rwx--x--x 1 demo demo  0 Oct  5 17:49 demo-file3.txt
-rw-r--r-- 1 demo demo 75 Oct  5 17:49 demo-file4.txt
>>>>> $
```

```
#define FILE4 "demo-file4.txt"
#include <stdio.h>
#include <stdlib.h>

void main(void) {
    FILE* fp;
    int   cc;

    printf ("*** Open and listing file %s ***\n\n", FILE4);
    if ((fp=fopen(FILE4, "r")) == NULL) {
        printf("fopen error...\n");
        exit(1);
    }
    while((cc=fgetc(fp)) != EOF) {
        printf("%c", cc);
    }
    printf("\n");
    fclose(fp);
}

*** Open and listing file demo-file4.txt ***

Line 1: Blah Blah Blah 1
Line 2: Blah Blah Blah 2
Line 3: Blah Blah Blah 3
```

54-write

```c
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>

#define  FILE5  "demo-file5.txt"
static char* str1 = "AAAXBBB\n";
static char* str2 = "CCC\n";

void main(void) {
   int fd1, fd2;
   fd1 = open (FILE5, O_RDWR | O_CREAT, 0644);
   fd2 = open (FILE5, O_RDWR | O_CREAT, 0644);
   printf("File Descriptors --- fd1 = %d, fd2 = %d\n", fd1, fd2);
   write(fd1, str1, strlen(str1));
   write(fd2, str2, strlen(str2));
   close(fd1);
   close(fd2);
   printf("See output file %s\n", FILE5);
}


######################################
File Descriptors --- fd1 = 3, fd2 = 4
See output file demo-file5.txt

######################################
demo-file5.txt:
CCC
BBB
```

55-write

```c
#define  FILE6  "demo-file6.txt"
char     buf1[] = "abcdefgh";
char     buf2[] = "ABCDEFGH";

void main(void) {
   int fd;
   fd = creat(FILE6, 0644);
   if (fd < 0) {
      perror("creat error");
      exit(1);
   }
   if (write(fd, buf1, 8) != 8) {
      perror("buf1 write error");
      exit(1);
   } /* offset now = 8 */
   if (lseek(fd, 32, SEEK_SET) == -1) {
      perror("lseek error");
      exit(1);
   } /* offset now = 32 */
   if (write(fd, buf2, 8) != 8) {
      perror("buf2 write error");
      exit(1);
   } /* offset now = 40 */
   close(fd);
   printf("Run: hexdump -c %s\n", FILE6);
}

>>>>> $ hexdump -c demo-file6.txt
0000000   a   b   c   d   e   f   g   h  \0  \0  \0  \0  \0  \0  \0  \0
0000010  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0
0000020   A   B   C   D   E   F   G   H
```

```c
#include <stdio.h>
#include <errno.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#define  BUF_SIZE 16

void main(int argc, char* argv[])
{
   int          fdread, fdwrite;
   unsigned int total_bytes = 0;
   ssize_t      nbytes_read, nbytes_write;
   char buf[BUF_SIZE];
   if (argc != 3) {
      printf("Usage: %s source destination\n",
      argv[0]);
      exit(1);
   }
   fdread = open(argv[1], O_RDONLY);
   if (fdread < 0) {
      perror("Failed to open source file");
      exit(1);
   }
   fdwrite = creat(argv[2], S_IRWXU);
   if (fdwrite < 0) {
      perror("Failed to open destination file");
      exit(1);
   }
```

```
    do {
        nbytes_read = read(fdread, buf, BUF_SIZE);
        if (nbytes_read < 0) {
            perror("Failed to read from file");
            exit(1);
        }
        nbytes_write = write(fdwrite, buf, nbytes_read);
        if (nbytes_write < 0) {
            perror("Failed to write to file");
            exit(1);
        }
    } while (nbytes_read > 0);
    close(fdread);
    close(fdwrite);
    exit(0);
}

##########################################################

>>>>> $ ./56-copy demo-file4.txt demo-copy.txt
>>>>> $ ls -al demo-file4.txt demo-copy.txt
-rwx------ 1 demo demo 75 Oct  5 18:12 demo-copy.txt
-rw-r--r-- 1 demo demo 75 Oct  5 17:49 demo-file4.txt
>>>>> $
```

```c
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>
#define FILE1 "demo-file7.txt"

static char* str1 = "AAAXBBB\n";
static char* str2 = "CCC\n";

Coming Soon
void
   int fd1, fd2;
   fd1 = open (FILE1, O_RDWR | O_CREAT, 0644);
   fd2 = dup(fd1);
   printf("File Descriptors --- fd1 = %d, fd2 = %d\n", fd1, fd2);
   write(fd1, str1, strlen(str1));
   write(fd2, str2, strlen(str2));
   close(fd1);
   close(fd2);
   printf("**** Please check file %s *****\n", FILE1);
   printf("**** Compare with 54-write\n");
}
#############################################
>>>>> $ 57-dup
File Descriptors --- fd1 = 3, fd2 = 4
**** Please check file demo-file7.txt *****
**** Compare with 54-write
>>>>> $ cat demo-file7.txt
AAAXBBB
CCC
>>>>> $
```

```c
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>
#define FILE1 "demo-file8.txt"

static char* str1 = "AAAXBBB\n";
static char* str2 = "CCC\n";

void main(void) {
    int fd1, fd2;
    fd1 = open (FILE1, O_RDWR | O_CREAT, 0644);
    dup2(fd1, fd2);
    printf("File Descriptors --- fd1 = %d, fd2 = %d\n", fd1, fd2);
    write(fd1, str1, strlen(str1));
    write(fd2, str2, strlen(str2));
    close(fd1);
    close(fd2);
    printf("**** Please check file %s ****\n", FILE1);
    printf("**** Compare with 54-write\n");
}
#############################################
>>>>> $ 58-dup2
File Descriptors --- fd1 = 3, fd2 = 0
**** Please check file demo-file8.txt ****
**** Compare with 54-write
>>>>> $ cat demo-file8.txt
AAAXBBB
CCC
>>>>> $
```

```
#include <stdio.h>
#include .....
#define FILE1 "demo-file9.txt"

void main(void) {
   int fd1, fd2;
   char strvar[100];
   printf ("***** Please check file %s ***** *****\n", FILE1);
/* BLOCK ***************
   close(STDERR_FILENO);
   close(STDOUT_FILENO);
   BLOCK *************** */
   fd1 = open (FILE1, O_RDWR | O_CREAT | O_TRUNC, 0644);
   fd2 = dup(fd1);
   printf(          "AAAAA print to standard output!!\n");
   fprintf(stdout, "BBBBB print to standard output!!\n");
   fprintf(stderr, "CCCCC print to standard error!!!\n");
   sprintf(strvar, "DDDDD print to fd1=%d!!!\n", fd1);
   dprintf(fd1,    "%s", strvar);
   dprintf(fd2,    "EEEEE print to fd2=%d!!!\n", fd2);
   close(fd1);
   close(fd2);
}
>>>>> $ 59-io ; echo "^^^^^";cat demo-file9.txt
***** Please check file demo-file9.txt ***** *****
AAAAA print to standard output!!
BBBBB print to standard output!!
CCCCC print to standard error!!!
^^^^^
DDDDD print to fd1=3!!!
EEEEE print to fd2=4!!!
```

```c
#include <stdio.h>
#include .....
#define FILE1 "demo-file9.txt"

void main(void) {
   int fd1, fd2;
   char strvar[100];
   printf ("***** Please check file %s ***** *****\n", FILE1);
   close(STDERR_FILENO);
/* BLOCK ***************
   close(STDOUT_FILENO);
   BLOCK *************** */
   fd1 = open (FILE1, O_RDWR | O_CREAT | O_TRUNC, 0644);
   fd2 = dup(fd1);
   printf(         "AAAAA print to standard output!!\n");
   fprintf(stdout, "BBBBB print to standard output!!\n");
   fprintf(stderr, "CCCCC print to standard error!!!\n");
   sprintf(strvar, "DDDDD print to fd1=%d!!!\n", fd1);
   dprintf(fd1,    "%s", strvar);
   dprintf(fd2,    "EEEEE print to fd2=%d!!!\n", fd2);
   close(fd1);
   close(fd2);
}
>>>>> $ 59-io ; echo "^^^^^";cat demo-file9.txt
***** Please check file demo-file9.txt ***** *****
AAAAA print to standard output!!
BBBBB print to standard output!!
^^^^^
CCCCC print to standard error!!!
DDDDD print to fd1=2!!!
EEEEE print to fd2=3!!!
```

```
#include <stdio.h>
#include .....
#define FILE1 "demo-file9.txt"

void main(void) {
    int fd1, fd2;
    char strvar[100];
    printf ("***** Please check file %s ***** *****\n", FILE1);
    close(STDERR_FILENO);
    close(STDOUT_FILENO);
/* BLOCK ***************
    BLOCK *************** */
    fd1 = open (FILE1, O_RDWR | O_CREAT | O_TRUNC, 0644);
    fd2 = dup(fd1);
    printf(         "AAAAA print to standard output!!\n");
    fprintf(stdout, "BBBBB print to standard output!!\n");
    fprintf(stderr, "CCCCC print to standard error!!!\n");
    sprintf(strvar, "DDDDD print to fd1=%d!!!\n", fd1);
    dprintf(fd1,    "%s", strvar);
    dprintf(fd2,    "EEEEE print to fd2=%d!!!\n", fd2);
    close(fd1);
    close(fd2);
}
>>>>> $ 59-io ; echo "^^^^^";cat demo-file9.txt
***** Please check file demo-file9.txt ***** *****
^^^^^
AAAAA print to standard output!!
BBBBB print to standard output!!
CCCCC print to standard error!!!
DDDDD print to fd1=1!!!
EEEEE print to fd2=2!!!
```

```
#define FILE1 "demo-fileA.txt"
#define  OLOOP 10
#define  ILOOP 3650
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <time.h>
#include <fcntl.h>
#include <dirent.h>
void rwfile (char* fname);
void dirfile(char* dname);
void error  (char* msg);
/* MAIN =========================== */
void main(void) {
    printf("Listing current directory...\n");
    dirfile(".");
    printf("Testing read-write speed...\n");
    rwfile(FILE1);
}
/* DIRFILE ====================== */
void dirfile(char* dname) {
    DIR*             ddir;
    struct dirent*  dp;
    printf("       ");
    ddir = opendir(dname);
    if (ddir != NULL) {
        while ((dp=readdir(ddir))!= NULL)
            printf("%s  ", dp->d_name);
        closedir(ddir); }
    printf("\n\n"); }
```

```
/* ERROR ========================= */
void error(char* msg){
    perror(msg);
    exit(0); }
/* RWFILE ========================= */
void rwfile(char* fname) {
    time_t tt;
    int    fd, ii, jj;
    char   buf[] = "Achtung... Achtung... AAAA BBBB CCCC DDDD\n";
    time(&tt);
    for (ii=0;ii<OLOOP;ii++) {
        if ((fd=creat(fname,00644)) < 0 )
            error("RWFILE: can not create file\n");
        for (jj=0;jj<ILOOP;jj++) {
            write(fd,buf,sizeof(buf)-1);
            fsync(fd); }
        close(fd);
        putchar('.');
        fflush(NULL); }
    tt=time(NULL)-tt;
    putchar('\n');
    printf("Total time: %d seconds\n", (int) tt);
}
#####################
>>>>> $ time 60-readwrite
Listing current directory...
    .shsh  52-open-close.c  demo-file4.txt  02-pointers.c ...
Testing read-write speed...
..........
Total time: 10 seconds
real    0m9.998s ----- user  0m0.024s ----- sys    0m0.576s
```

The End

- ☐ This is the end of the presentation.
- ☒ This is the end of the presentation.
- This is the end of the presentation.