4 Linux

4.1

/ “ ” “ ” “ ” “ ” “ ” “ ”

linux linux( ) Linux

4.1.1

4.1.1.1

/ C/C++ C/C++ C/C++

```
01 #include <stdio.h>
02
03 int main(int argc, char *argv[])
04{
05    printf("hello world!\n");
06    return 0;
07}
```

4.1.1.2

Pascal · = +

4.1.1.3

1.
2.
3. ----
4.

4.1.1.4

ProcessCommunication_Image001

Image not found or type unknown

1.

= =CPU = =

2.

= =

= =

4.1.2

4.1.2.1

fork

: #include <unistd.h>

: pid_t fork(void);

: (>0) 0; fork

```
:   fork
:      fork
```

---

1. linux
2. pcb PC PC fork fork
3.
4.
5. INIT INIT

Tips

linux API ubuntu man

ProcessCommunication_Image002

Image not found or type unknown

ProcessCommunication_Image003

Image not found or type unknown

jz2440\process\1th_create_process\create_process.c

```
01 /*********************************************************************
02  *
03  *
04  *
05  *
06  *
07  * ----------------------------------------------
08  * 2020/05/16       V1.0      zh(ryan)
09  *********************************************************************/
10
11 #include <stdio.h>
12 #include <stdlib.h>
13 #include <unistd.h>
14 #include <sys/types.h>
15
```

```
16 int main(int argc, char *argv[])
17 {
18     pid_t pid;
19
20     pid = fork();  //
21
22     if (pid == 0) {  //
23         int i = 0;
24         for (i = 0; i < 5; i++) {
25             usleep(100);
26             printf("this is child process i=%d\n", i);
27         }
28     }
29
30     if (pid > 0) {  //
31         int i = 0;
32         for (i = 0; i < 5; i++) {
33             usleep(100);
34             printf("this is parent process i=%d\n", i);
35         }
36     }
37
38     while(1);   //
39     return 0;
40 }
```

JZ2440

jz2440 NFS jz2440 ubuntu NFS u-

-

```
arm-linux-gcc create_process.c -o create_process
```

- test NFS

```
cp create_process /work/nfs_root/first_fs
```

- jz2440

ProcessCommunication_Image004

Image not found or type unknown

- ”&” ”&”

```
./create_process &
```

ProcessCommunication_Image005

Image not found or type unknown

- top

```
top
```

ProcessCommunication_Image006

Image not found or type unknown

create_process PID 777(PID 776) PID 776(PID 770)

4.1.2.2

```
exit
: #include <stdlib.h>
: void exit (int status)
```

```
_exit
: #include <unistd.h>
: void _exit(int status);
```

exit _exit

return exit _ exit return main return

exit _exit

1 exit jz2440\process\2th_exit_process\exit_process.c

```
01 /*********************************************************************
02  *        exit
03  *
```

```
04  *
05  *
06  *
07  * ------------------------------------------
08  * 2020/05/16       V1.0      zh(ryan)
09  ******************************************************************/
10 #include <stdio.h>
11 #include <stdlib.h>
12
13 int main(int argc, char *argv[])
14 {
15     printf("hello world\n");
16     printf("will exit");
17     exit(0);    //  _exit
18 }
```

2 _exit jz2440\process\3th_exit_process\exit_process.c

```
01 /******************************************************************
02  *        _exit
03  *
04  *
05  *
06  *
07  * ------------------------------------------
08  * 2020/05/16       V1.0      zh(ryan)
09  ******************************************************************/
10 #include <stdio.h>
11 #include <stdlib.h>
12
13 int main(int argc, char *argv[])
14 {
15     printf("hello world\n");
16     printf("will exit");
17     _exit(0);    //  _exit
18 }
```

15 16 “\n” exit _exit 16 “\n”

JZ2440

1

-

```
arm-linux-gcc exit_process.c -o exit_process
```

- NFS

```
cp exit_process /work/nfs_root/first_fs
```

-

```
./exit_process
```

“hello world” “will exit”

ProcessCommunication_Image008

Image not found or type unknown

4.1.2.3

```
  wait
   : #include <sys/types.h>
    #include <sys/wait.h>
   : pid_t wait(int *status);
   :               -1
```

```
  waitpid
   : #include <sys/types.h>
    #include <sys/wait.h>
   : pid_t waitpid(pid_t pid, int *status, int options);
   :               -1
```

jz2440\process\4th_exit_wait\exit_wait.c

```
1 /*********************************************************************
02  *       exit         waitpid
03  *
04  *
05  *
06  *
```

```
 * ------------------------------------------
 * 2020/05/16       V1.0      zh(ryan)
 ************************************************************************/
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/wait.h>

int main(int argc, char *argv[])
{
    int status = -1;
    pid_t pid;

    pid = fork();
    if (pid == 0){ //
        printf("fork\n");
        exit(1);
    } else if (pid > 0) { //
        pid = waitpid(pid, &status, 0);
        printf("status=0x%x\n", status);
    } else {
        perror("fork\n");
    }

    return 0;
}
```

JZ2440

-

```
arm-linux-gcc exit_wait.c -o exit_wait
```

- NFS

```
cp exit_wait /work/nfs_root/first_fs
```

-

```
./exit_wait
```

ProcessCommunication_Image009

Image not found or type unknown

4.2

“global” **Ubuntu**

1

```
01 #include <stdio.h>
02 int global = 1;
03
04 void delay(void)
05 {
06 	unsigned int a = 1000000;
07 	while(a--);
08 }
09
10 int main(int argc, char *argv[])
11 {
12 	while (1) {
13 		printf("global=%d\n", global);
14 		delay();
15 	}
16 	return 0;
17 }
```

2

```
01 #include <stdio.h>
02 int global = 2;
03
04 void delay(void)
05 {
06 	unsigned int a = 1000000;
07 	while(a--);
08 }
```

```
09
10 int main(int argc, char *argv[])
11 {
12 □while (1) {
13 □□printf("global=%d\n", global);
14 □□delay();
15 □}
16 □return 0;
17 }
```

ProcessCommunication_Image010

Image not found or type unknown

-

```
gcc test1.c -o test1
gcc test2.c -o test2
```

-

```
./test1
./test2
```

ProcessCommunication_Image011

Image not found or type unknown

1

ProcessCommunication_Image012

Image not found or type unknown

2

global ** (

A B B

ProcessCommunication_Image013

Image not found or type unknown

linux

linux

IPC	
socket	socket socket

linux ” ----“ soc

ProcessCommunication_Image014

Image not found or type unknown

4.3

	( )

4.3.1

4.3.1.1

fd[0] fd[1] open pipe

ProcessCommunication_Image015

Image not found or type unknown

4.3.1.2

```
1.   #include <unistd.h>

2.    : int pipe(int fd[2])

3.  :                  fd[0] fd[1]       fd[0]     fd[1]

4.     0    1
```

4.3.1.3

```
1.    read           fd[0]

2.    write           fd[1]

3.     close                              fd[0]        fd[1]
```

4.3.1.4

1

jz2440\process_pipe\1th_write_pipe\my_pipe_write.c

```
01 /*********************************************************************
02  *
03
04  *
05  *
06  *
07  *
08  * ------------------------------------------
09  * 2020/05/16       V1.0      zh(ryan)
10  *********************************************************************/
11 #include <stdio.h>
12 #include <stdlib.h>
13 #include <unistd.h>
14
15 int main(int argc, char *argv[])
16 {
```

```
17     int fd[2];
18     int ret = 0;
19     char write_buf[] = "Hello linux";
20     char read_buf[128] = {0};
21
22     ret = pipe(fd);
23     if (ret < 0) {
24         printf("create pipe fail\n");
25         return -1;
26     }
27     printf("create pipe sucess fd[0]=%d fd[1]=%d\n", fd[0], fd[1]);
28
29     //     fd[1]
30     write(fd[1], write_buf, sizeof(write_buf));
31
32     //     fd[0]
33     read(fd[0], read_buf, sizeof(read_buf));
34     printf("read_buf=%s\n", read_buf);
35
36     close(fd[0]);
37     close(fd[1]);
38     return 0;
39 }
```

JZ2440

-

```
arm-linux-gcc my_pipe_write.c -o my_pipe_write
```

- NFS

```
cp my_pipe_write /work/nfs_root/first_fs
```

-

```
./my_pipe_write
```

” Hello linux”

2

“proce

jz2440\process_pipe\2th_comm\test.c

```
/*****************************************************************
 *      1.
            2.          process_inter   1
            3.     process_inter 1
 *
 *
 *
 *
 * ---------------------------------------------
 * 2020/05/16       V1.0      zh(ryan)
 *****************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>

int main(int argc, char *argv[])
{
    pid_t pid;
    int process_inter = 0;

    pid = fork();  //

    if (pid == 0) {  //
        int i = 0;
        while (process_inter == 0); //
        for (i = 0; i < 5; i++) {
            usleep(100);
            printf("this is child process i=%d\n", i);
        }
    }

    if (pid > 0) {  //
        int i = 0;
        for (i = 0; i < 5; i++) {
            usleep(100);
```

```
37             printf("this is parent process i=%d\n", i);
38         }
39         process_inter == 1;
40     }
41
42     while(1);
43     return 0;
44 }
```

JZ2440

-

```
arm-linux-gcc test.c -o test
```

- NFS

```
cp test /work/nfs_root/first_fs
```

-

```
./test
```

29 process_inter 0

ProcessCommunication_Image017

Image not found or type unknown

3

jz2440\process_pipe\3th_pipe_comm\comm_fork.c

```
01 /*************************************************************************
02  *      1.
03             2.
04             3.
05  *
06  *
07  *
08  *
09  * ---------------------------------------------
10  * 2020/05/16       V1.0      zh(ryan)
```

```
11  ************************************************************************/
12
13 #include <stdio.h>
14 #include <stdlib.h>
15 #include <unistd.h>
16 #include <sys/types.h>
17
18 int main(int argc, char *argv[])
19 {
20     pid_t pid;
21     char process_inter = 0;
22     int fd[2], ret = 0;
23
24     ret = pipe(fd);   //
25     if (ret < 0) {
26         printf("create pipe fail\n");
27         return -1;
28     }
29     printf("create pipe sucess\n");
30
31     pid = fork();  //
32
33     if (pid == 0) {  //
34         int i = 0;
35         read(fd[0], &process_inter, sizeof(process_inter));  //
36         while (process_inter == 0);
37         for (i = 0; i < 5; i++) {
38             usleep(100);
39             printf("this is child process i=%d\n", i);
40         }
41     } else if (pid > 0) {  //
42         int i = 0;
43         for (i = 0; i < 5; i++) {
44             usleep(100);
45             printf("this is parent process i=%d\n", i);
46         }
47         process_inter = 1;
48         sleep(2);
49         write(fd[1], &process_inter, sizeof(process_inter));
50     }
```

```
51
52     while(1);
53     return 0;
54 }
```

JZ2440

-

```
arm-linux-gcc comm_fork.c -o comm_fork
```

- NFS

```
cp comm_fork /work/nfs_root/first_fs
```

-

```
./comm_fork
```

38 2s 2s

ProcessCommunication_Image018

Image not found or type unknown

4.3.2

4.3.2.1

Linux 7

	‘-’ open
	‘d’ mkdir
	‘l’, la -s
( )	‘p’ mkfifo
socket	‘s’ socket
	‘c’
	‘b’

4.3.2.2

```
      int mkfifo(const char * filename, mode_t mode)

                    umask

        0    -1
```

mkfifo

4.3.2.3

1

(jz2440\process_pipe\4th_create_myfifo\create_myfifo.c)

```
01 /*********************************************************************
02  *      1.
03  *
04  *
05  *
06  *
07  * ----------------------------------------------
08  * 2020/05/16       V1.0      zh(ryan)
09  *********************************************************************/
10
11 #include <stdio.h>
12 #include <stdlib.h>
13 #include <unistd.h>
14 #include <sys/types.h>
15
16 int main(int argc, char *argv[])
17 {
18     int ret;
19
20     ret = mkfifo("./myfifo", 0777);   //          777
21     if (ret < 0) {
22         printf("create myfifo fail\n");
23         return -1;
```

```
24     }
25     printf("create myfifo sucess\n");
26
27     return 0;
28 }
```

JZ2440

-

```
arm-linux-gcc create_myfifo.c -o create_myfifo
```

- NFS

```
cp create_myfifo /work/nfs_root/first_fs
```

-

```
./create_myfifo
```

myfifo “-p”

ProcessCommunication_Image019

Image not found or type unknown

2

1 (jz2440\process_pipe\5th_myfifo_comm\5nd_named_pipe.c)

```
01 /*********************************************************************
02  *      1.  1        3rd_fifo    0777
03              2.
04  *
05  *
06  *
07  *
08  * ------------------------------------------
09  * 2020/05/16       V1.0      zh(ryan)
10  ********************************************************************/
11
12 #include <stdio.h>
13 #include <stdlib.h>
```

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

int main(int argc, char *argv[])
{
    int i, ret, fd;
    char p_flag = 0;

    /*       */
    if (access("./3rd_fifo", 0) < 0) {   //           ,
        ret = mkfifo("./3rd_fifo", 0777);
        if (ret < 0) {
            printf("create named pipe fail\n");
            return -1;
        }
        printf("create named pipe sucess\n");
    }

    /*             */
    fd=open("./3rd_fifo", O_WRONLY);
    if (fd < 0) {
        printf("open 3rd_fifo fail\n");
        return -1;
    }
    printf("open 3rd_fifo sucess\n");

    for (i = 0; i < 5; i++) {
        printf("this is first process i=%d\n", i);
        usleep(100);
    }
    p_flag = 1;
    sleep(5);
    write(fd, &p_flag, sizeof(p_flag));

    while(1);
    return 0;
}
```

2 （ jz2440\process_pipe\5th_myfifo_comm\5nd_named_pipe_2.c)

```
/*********************************************************************
 *      1.
            2.
 *
 *
 *
 *
 * ------------------------------------------
 * 2020/05/16       V1.0      zh(ryan)
 *********************************************************************/

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <fcntl.h>

int main(int argc, char *argv[])
{
    int i;
    int fd=open("./3rd_fifo", O_RDONLY);
    char p_flag = 0;

    if (fd < 0) {
        printf("open 3rd_fifo fail\n");
        return -1;
    }

    printf("open 3rd_fifo sucess\n");
    read(fd, &p_flag, sizeof(p_flag));
    while(!p_flag);
    for (i = 0; i < 5; i++) {
        printf("this is second process i=%d\n", i);
        usleep(100);
    }

    while(1);
    return 0;
}
```

JZ2440

-

```
arm-linux-gcc 5nd_named_pipe.c -o 5nd_named_pipe

arm-linux-gcc 5nd_named_pipe_2.c -o 5nd_named_pipe_2
```

- NFS

```
cp 5nd_named_pipe /work/nfs_root/first_fs

cp 5nd_named_pipe_2 /work/nfs_root/first_fs
```

-

```
./5nd_named_pipe &

./5nd_named_pipe_2 &
```

ProcessCommunication_Image020

Image not found or type unknown

4.4 IPC

IPC IPC

IPC

- key key IPC_PRIVATE key ftok
- key IPC IPC IPC IPC ID IPC_id shm_id msg_id
- IPC_id IPC IPC shmctrl shmat shmdt msgctrl msgsnd msgrecv

ProcessCommunication_Image021

Image not found or type unknown

key IPC_id shm_id/msg_id/sem_id

IPC

IPC ID IPC_id IPC_id ID IPC IPC

ProcessCommunication_Image022

Image not found or type unknown

ftok

```
: char ftok(const char *path, char key)

path

   key

      key     -1
```

ftok key IPC

IPC_PRIVATE

ftok key key key key

ftok key

4.4.1

4.4.1.1

Linux

ProcessCommunication_Image023

Image not found or type unknown

-
- key ID
-

4.4.1.2

```
    : int shmget(key_t key, int size, int shmflg)

  : #include <sys/shm.h>

    : key: IPC_PRIVATE   ftok

□□ IPC_PRIVATE   key     ,  0

□□ size :

□□ shmflg :  open

               ID  -1
```

1 jz2440\process_ipc\1st_shm\1st_shm.c

```
01 /*********************************************************************
02  *      1.  IPC_PRIVATE
03  *
04  *
05  *
06  *
07  * ------------------------------------------
08  * 2020/05/16       V1.0      zh(ryan)
09  *********************************************************************/
10
11 #include <stdio.h>
12 #include <stdlib.h>
13 #include <unistd.h>
14 #include <sys/types.h>
15 #include <sys/shm.h>
16 #include <signal.h>
17
18 int main(int argc, char *argv[])
19 {
20     int shmid;
21
22     shmid = shmget(IPC_PRIVATE, 128, 0777);
```

```
23     if (shmid < 0) {
24         printf("create shared memory fail\n");
25         return -1;
26     }
27     printf("create shared memory sucess, shmid = %d\n", shmid);
28     system("ipcs -m");
29     return 0;
30 }
```

JZ2440

-

```
arm-linux-gcc 1st_shm.c -o 1st_shm
```

- NFS

```
cp 1st_shm /work/nfs_root/first_fs
```

-

18 console “ipcs -m” key 0

```
./1st_shm
```

ProcessCommunication_Image024

Image not found or type unknown

2 JZ2440\process_ipc\1st_shm\2nd_shm.c

fotk key

```
01 /*************************************************************************
02  *      1.  ftok    key
03  *
04  *
05  *
06  *
07  * ----------------------------------------------
08  * 2020/05/16       V1.0      zh(ryan)
09  ************************************************************************/
10
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/shm.h>
#include <signal.h>

int main(int argc, char *argv[])
{
    int shmid;
    int key;

    key = ftok("./a.c", 'a');  //    key
    if (key < 0) {
        printf("create key fail\n");
        return -1;
    }
    printf("create key sucess key = 0x%X\n",key);

    shmid = shmget(key, 128, IPC_CREAT | 0777);
    if (shmid < 0) {
        printf("create shared memory fail\n");
        return -1;
    }
    printf("create shared memory sucess, shmid = %d\n", shmid);
    system("ipcs -m");
    return 0;
}
```

JZ2440

-

```
arm-linux-gcc 2nd_shm.c -o 2nd_shm
```

- NFS

```
cp 2nd_shm /work/nfs_root/first_fs
```

-

2nd_shm a.c jz2440

```
touch a.c
```

key 0x610d0169.

```
./2nd_shm
```

ProcessCommunication_Image025

Image not found or type unknown

4.4.1.3

shmat

```
  void *shmat(int shmid, const void *shmaddr, int shmflg)

shmid ID

 shmaddr NULL

  shmflg SHM_RDONLY

    0

          NULL
```

jz2440\process_ipc\1st_shm\3nd_shm.c

```
/*****************************************************************
 *      1.              shmat
 *          2.
 *          3.
 *
 *
 *
 *
```

```
 * ------------------------------------------
 * 2020/05/16       V1.0      zh(ryan)
 ******************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/shm.h>
#include <signal.h>

int main(int argc, char *argv[])
{
    int shmid;
    int key;
    char *p;

    key = ftok("./a.c", 'b');
    if (key < 0) {
        printf("create key fail\n");
        return -1;
    }
    printf("create key sucess key = 0x%X\n",key);

    shmid = shmget(key, 128, IPC_CREAT | 0777);
    if (shmid < 0) {
        printf("create shared memory fail\n");
        return -1;
    }
    printf("create shared memory sucess, shmid = %d\n", shmid);
    system("ipcs -m");

    p = (char *)shmat(shmid, NULL, 0);
    if (p == NULL) {
        printf("shmat fail\n");
        return -1;
    }
    printf("shmat sucess\n");

    //  console
    fgets(p, 128, stdin);
```

```
49
50     //
51     printf("share memory data: %s\n", p);
52
53     //
54     printf("share memory data: %s\n", p);
55     return 0;
56 }
```

JZ2440

-

```
arm-linux-gcc 3nd_shm.c -o 3nd_shm
```

- NFS

```
cp 3nd_shm /work/nfs_root/first_fs
```

-

3nd_shm a.c jz2440

```
touch a.c
```

```
./3nd_shm
```

ProcessCommunication_Image026

Image not found or type unknown

console “hello linux”

ProcessCommunication_Image027

Image not found or type unknown

51 54

shmdt

```
    int shmdt(const void *shmaddr)

    shmat

   :   0   -1
```

jz2440\process_ipc\1st_shm\4th_shm.c

```
01 /*********************************************************************
02  *      1.              shmat
03             2.
04             3.
05             4.  shmdt
06  *
07  *
08  *
09  *
10  * ----------------------------------------------
11  * 2020/05/16       V1.0      zh(ryan)
12  *********************************************************************/
13 #include <stdio.h>
14 #include <stdlib.h>
15 #include <unistd.h>
16 #include <sys/types.h>
17 #include <sys/shm.h>
18 #include <signal.h>
19 #include <string.h>
20
21 int main(int argc, char *argv[])
22 {
23     int shmid;
24     int key;
25     char *p;
26
27     key = ftok("./a.c", 'b');
28     if (key < 0) {
29         printf("create key fail\n");
30         return -1;
31     }
32     printf("create key sucess key = 0x%X\n",key);
```

```
33
34      shmid = shmget(key, 128, IPC_CREAT | 0777);
35      if (shmid < 0) {
36          printf("create shared memory fail\n");
37          return -1;
38      }
39      printf("create shared memory sucess, shmid = %d\n", shmid);
40      system("ipcs -m");
41
42      p = (char *)shmat(shmid, NULL, 0);
43      if (p == NULL) {
44          printf("shmat fail\n");
45          return -1;
46      }
47      printf("shmat sucess\n");
48
49      //write share memory
50      fgets(p, 128, stdin);
51
52      //start read share memory
53      printf("share memory data:%s\n", p);
54
55      //start read share memory again
56      printf("share memory data:%s\n", p);
57
58      //
59      shmdt(p);
60
61      memcpy(p, "abcd", 4);  //       segment fault
62      return 0;
63  }
```

JZ2440

-

```
arm-linux-gcc 4th_shm.c -o 4th_shm
```

- NFS

```
cp 4th_shm /work/nfs_root/first_fs
```

-

  4th_shm.c a.c jz2440

```
touch a.c
```

  , 61 Segmentation fault

```
./4th_shm
```

ProcessCommunication_Image028

Image not found or type unknown

shmctl

```
    int shmctl(int shmid, int cmd, struct shmid_ds *buf)

 : shmid :

 cmd : IPC_START (    )   ---      ipcs -m

    IPC_SET(     )

 IPC_RMID (     )    ---      ipcrm -m

 buf :   IPC_START/IPC_SET    /

   :   0   -1
```

  jz2440\process_ipc\1st_shm\5th_shm.c

```
01 /*************************************************************************
02  *      1.              shmat
03              2.
04              3.
05              4.  shmdt
06              5.    shmctl
07  *
08  *
09  *
10  *
```

```
 * ------------------------------------------
 * 2020/05/16       V1.0      zh(ryan)
 ******************************************************************/

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/shm.h>
#include <signal.h>
#include <string.h>

int main(int argc, char *argv[])
{
    int shmid;
    int key;
    char *p;

    key = ftok("./a.c", 'b');
    if (key < 0) {
        printf("create key fail\n");
        return -1;
    }
    printf("create key sucess key = 0x%X\n",key);

    shmid = shmget(key, 128, IPC_CREAT | 0777);
    if (shmid < 0) {
        printf("create shared memory fail\n");
        return -1;
    }
    printf("create shared memory sucess, shmid = %d\n", shmid);
    system("ipcs -m");

    p = (char *)shmat(shmid, NULL, 0);
    if (p == NULL) {
        printf("shmat fail\n");
        return -1;
    }
    printf("shmat sucess\n");

```

```
51     //write share memory
52     fgets(p, 128, stdin);
53
54     //start read share memory
55     printf("share memory data: %s\n", p);
56
57     //start read share memory again
58     printf("share memory data: %s\n", p);
59
60     //
61     shmdt(p);
62
63     //memcpy(p, "abcd", 4);  //       segment fault
64
65     shmctl(shmid, IPC_RMID, NULL);
66     system("ipcs -m");
67     return 0;
68 }
```

JZ2440

-

```
arm-linux-gcc 5th_shm.c -o 5th_shm
```

- NFS

```
cp 5th_shm /work/nfs_root/first_fs
```

-

```
touch a.c
```

42 66

```
./4th_shm
```

ProcessCommunication_Image029

Image not found or type unknown

4.4.1.4

```
1.   /

2.

3.

4.

5.
```

```
1.              ipcs -l                    cat /proc/sys/kernel/shmmax

2.          shmctl                    (      shmdt  )
```

jz2440\process_ipc\1st_shm\6th_shm.c

```
01 /*********************************************************************
02  *      1.        key  IPC_PRIVATE
03             2.
04             3.
05             4.
06             5.
07             6.
08  *
09  *
10  *
11  *
12  * ----------------------------------------------
13  * 2020/05/16       V1.0      zh(ryan)
14  *********************************************************************/
15
16 #include <stdio.h>
17 #include <stdlib.h>
18 #include <unistd.h>
19 #include <sys/types.h>
20 #include <sys/shm.h>
```

```c
#include <signal.h>
#include <string.h>

void myfun(int signum)
{
    return;
}

int main(int argc, char *argv[])
{
    int shmid;
    int key;
    char *p;
    int pid;


    shmid = shmget(IPC_PRIVATE, 128, IPC_CREAT | 0777);
    if (shmid < 0) {
        printf("create shared memory fail\n");
        return -1;
    }
    printf("create shared memory sucess, shmid = %d\n", shmid);

    pid = fork();
    if (pid > 0) {    //
        signal(SIGUSR2, myfun);
        p = (char *)shmat(shmid, NULL, 0);
        if (p == NULL) {
            printf("shmat fail\n");
            return -1;
        }
        printf("parent process shmat sucess\n");
        while (1) {
            //
            printf("parent process begin to write memory data:");
            fgets(p, 128, stdin);
            kill(pid, SIGUSR1);   //
            pause();              //
        }
    }
```

```
61     if (pid == 0) { //
62         signal(SIGUSR1, myfun);
63         p = (char *)shmat(shmid, NULL, 0);
64         if (p == NULL) {
65             printf("shmat fail\n");
66             return -1;
67         }
68         printf("child process shmat sucess\n");
69         while (1) {
70             pause(); //
71             //
72             printf("child process read share memory data: %s\n", p);
73             kill(getppid(), SIGUSR2);
74         }
75     }
76
77     //
78     shmdt(p);
79
80     //memcpy(p, "abcd", 4);  //       segment fault
81
82     shmctl(shmid, IPC_RMID, NULL);
83     system("ipcs -m");
84     return 0;
85 }
```

JZ2440

-

```
arm-linux-gcc 6th_shm.c -o 6th_shm
```

- NFS

```
cp 6th_shm /work/nfs_root/first_fs
```

-

```
./6th_shm
```

ProcessCommunication_Image030

Image not found or type unknown

server　　jz2440\process_ipc\1st_shm\7th_shm_1.c

```
01 /*************************************************************************
02  *      1.server    ftok    key     key
03              2.
04              3.server          client
05  *
06  *
07  *
08  *
09  * ------------------------------------------
10  * 2020/05/16       V1.0      zh(ryan)
11  ************************************************************************/
12
13 #include <stdio.h>
14 #include <stdlib.h>
15 #include <unistd.h>
16 #include <sys/types.h>
17 #include <sys/shm.h>
18 #include <signal.h>
19 #include <string.h>
20
21 struct mybuf
22 {
23     int pid;
24     char buf[124];
25 };
26
27 void myfun(int signum)
28 {
29     return;
30 }
31
32 int main(int argc, char *argv[])
33 {
34     int shmid;
35     int key;
```

```
36      struct mybuf *p;
37      int pid;
38
39      key = ftok("./a.c", 'a');
40      if (key < 0) {
41          printf("create key fail\n");
42          return -1;
43      }
44      printf("create key sucess\n");
45
46      shmid = shmget(key, 128, IPC_CREAT | 0777);
47      if (shmid < 0) {
48          printf("create shared memory fail\n");
49          return -1;
50      }
51      printf("create shared memory sucess, shmid = %d\n", shmid);
52
53      signal(SIGUSR2, myfun);
54      p = (struct mybuf *)shmat(shmid, NULL, 0);
55      if (p == NULL) {
56          printf("shmat fail\n");
57          return -1;
58      }
59      printf("parent process shmat sucess\n");
60
61      p->pid = getpid(); //  server   pid
62      pause();           //   client   server pid
63      pid=p->pid;        //   client
64
65      while (1) {
66          //write share memory
67          printf("parent process begin to write memory data\n");
68          fgets(p->buf, 124, stdin);
69          kill(pid, SIGUSR1);   //  client     client
70          pause();              //   client
71      }
72
73      //
74      shmdt(p);
75
```

```
76     shmctl(shmid, IPC_RMID, NULL);
77     system("ipcs -m");
78     return 0;
79 }
```

client　　jz2440\process_ipc\1st_shm\7th_shm_2.c

```
01 /*************************************************************************
02  *      1.client    ftok    key     key
03             2.client    server
04             3.
05  *
06  *
07  *
08  *
09  * ------------------------------------------
10  * 2020/05/16       V1.0      zh(ryan)
11  ************************************************************************/
12
13 #include <stdio.h>
14 #include <stdlib.h>
15 #include <unistd.h>
16 #include <sys/types.h>
17 #include <sys/shm.h>
18 #include <signal.h>
19 #include <string.h>
20
21 struct mybuf
22 {
23     int pid;
24     char buf[124];
25 };
26
27 void myfun(int signum)
28 {
29     return;
30 }
31
32 int main(int argc, char *argv[])
33 {
```

```
34      int shmid;
35      int key;
36      struct mybuf *p;
37      int pid;
38
39      key = ftok("./a.c", 'a');
40      if (key < 0) {
41          printf("create key fail\n");
42          return -1;
43      }
44      printf("create key sucess\n");
45
46      shmid = shmget(key, 128, IPC_CREAT | 0777);
47      if (shmid < 0) {
48          printf("create shared memory fail\n");
49          return -1;
50      }
51      printf("create shared memory sucess, shmid = %d\n", shmid);
52
53      signal(SIGUSR1, myfun);
54      p = (struct mybuf *)shmat(shmid, NULL, 0);
55      if (p == NULL) {
56          printf("shmat fail\n");
57          return -1;
58      }
59      printf("client process shmat sucess\n");
60
61      // get server pid
62      //read share memory
63      pid = p->pid;
64      // write client pid to share memory
65      p->pid = getpid();
66      kill(pid, SIGUSR2);   // tell server process to read data
67
68      //client start to read share memory
69
70      while (1) {
71          pause();              // wait server process write share memory
72          printf("client process read data:%s\n", p->buf); // read data
73          kill(pid, SIGUSR2);   // server can  write share memory
```

```
74     }
75
76     //
77     shmdt(p);
78
79     shmctl(shmid, IPC_RMID, NULL);
80     system("ipcs -m");
81     return 0;
82 }
```

console server client console telnet console

4.4.2

4.4.2.1

cat/proc/sys/kernel/msgmax

msgqid_ds msqid_ds msqid_ds.msg

ProcessCommunication_Image031

Image not found or type unknown

4.4.2.2

1.
2.
3.

4.4.2.3

msgget

```
#include <sys/types.h>
     #include <sys/ipc.h>
    #include <sys/msg.h>
int msgget(key_t key, int flag)
key        key
    flag
       ID   -1
```

msgctl

```
int msgctl(int msgqid, int cmd, struct msqid_ds *buf)
msgqid    ID
        cmd IPC_STAT              buf
            IPC_SET              buf
            IPC_RMID
        buf
    0   -1
```

msgsnd

```
#include <sys/types.h>
   #include <sys/ipc.h>
   #include <sys/msg.h>
int msgsnd(int msgqid, const void *msgp, size_t size, int flag)
msgqid    ID
        msgp             msgbuf
        struct msgbuf {
            long mtype;         //
            char mtext[N];      //
        };
        size
        flag IPC_NOWAIT
             0
        0   -1
```

msgrcv

```
int msgrcv(int msgqid, void *msgp, size_t size, long msgtype, int flag)
msgqid    ID
    msgp
    size
    msgtype  0
               0            msgtype
               0            msgtype
        flag IPC_NOWAIT
```

```
                0

                -1
```

4.4.2.4

server　　jz2440\process_ipc\2nd_shm\write_msg.c

```
01 /*************************************************************************
02  *      1.server
03  *
04  *
05  *
06  *
07  * ----------------------------------------------
08  * 2020/05/16       V1.0      zh(ryan)
09  ************************************************************************/
10
11 #include <stdio.h>
12 #include <stdlib.h>
13 #include <unistd.h>
14 #include <sys/types.h>
15 #include <sys/msg.h>
16 #include <signal.h>
17 #include <string.h>
18
19 struct msgbuf {
20     long type;         //
21     char voltage[124];     //
22     char ID[4];
23 };
24
25 int main(int argc, char *argv[])
26 {
27     int msgid, readret, key;
28     struct msgbuf sendbuf;
29
30     key = ftok("./a.c", 'a');
31     if (key < 0){
32         printf("create key fail\n");
```

```
33          return -1;
34      }
35      msgid = msgget(key, IPC_CREAT|0777);
36      if (msgid < 0) {
37          printf("create msg queue fail\n");
38          return -1;
39      }
40      printf("create msg queue sucess, msgid = %d\n", msgid);
41      system("ipcs -q");
42
43      // write message queue
44      sendbuf.type = 100;
45      while(1) {
46          memset(sendbuf.voltage, 0, 124);  //clear send buffer
47          printf("please input message:");
48          fgets(sendbuf.voltage, 124, stdin);
49          //start write msg to msg queue
50          msgsnd(msgid, (void *)&sendbuf, strlen(sendbuf.voltage), 0);
51      }
52
53      return 0;
54  }
```

client　　jz2440\process_ipc\2nd_shm\read_msg.c

```
01  /*************************************************************************
02   *      1.client
03   *
04   *
05   *
06   *
07   * ----------------------------------------------
08   * 2020/05/16       V1.0      zh(ryan)
09   ************************************************************************/
10
11  #include <stdio.h>
12  #include <stdlib.h>
13  #include <unistd.h>
14  #include <sys/types.h>
15  #include <sys/msg.h>
```

```c
#include <signal.h>
#include <string.h>

struct msgbuf {
    long type;         //
    char voltage[124];     //
    char ID[4];
};

int main(int argc, char *argv[])
{
    int msgid, key;
    struct msgbuf readbuf;

    key = ftok("./a.c", 'a');
    if (key < 0){
        printf("create key fail\n");
        return -1;
    }
    msgid = msgget(key, IPC_CREAT|0777);
    if (msgid < 0) {
        printf("create msg queue fail\n");
        return -1;
    }
    printf("create msg queue sucess, msgid = %d\n", msgid);
    system("ipcs -q");

    // read message queue
    while(1) {
        memset(readbuf.voltage, 0, 124);  //clear recv buffer
        //start read msg to msg queue
        msgrcv(msgid, (void *)&readbuf, 124, 100, 0);
        printf("recv data from message queue:%s", readbuf.voltage);
    }

    return 0;
}
```

JZ2440

-

```
arm-linux-gcc write_msg.c -o write_msg
arm-linux-gcc read_msg.c -o read_msg
```

- NFS

```
cp write_msg /work/nfs_root/first_fs
cp read_msg /work/nfs_root/first_fs
```

-

read_msg write_msg console client

```
./read_msg &
./ write_msg
```

ProcessCommunication_Image032

Image not found or type unknown

4.4.3

4.4.3.1 P V

P V

4.4.3.2

P/V / semid_ds

ProcessCommunication_Image033

Image not found or type unknown

POSIX POSIX IPC

	(POSIX)	(IPC)
	sem_t sem1	semget
	sem_init	semctl

	(POSIX)	(IPC)
P	sem_wait	semop
V	sem_post	semop

IPC POSIX

1. A B 1 2 A 1 2
2. B 2 1
3. A 1 2 B 2 1
4. B 2 A 1 B 1 2 A 2 1

4.4.3.3

semget

```
  #includde <sys/types.h>
#includde <sys/ipc.h>
#includde <sys/sem.h>
 : int semget(key_t key, int nsems, int semflag)
   key       key
     nsems
     semflag
        ID   -1
```

semctl

```
  #includde <sys/types.h>
#includde <sys/ipc.h>
#includde <sys/sem.h>
 : int semctl(int semid, int semnum, int cmd, ...union semun arg)

   semid    id
     semnum        ,
     cmd GETVAL
          SETVAL
          IPC_RMID
     union semun arg: union semun {
```

```
                    int              val;    /* Value for SETVAL */
                    struct semid_ds *buf;    /* Buffer for IPC_STAT, IPC_SET */
                    unsigned short  *array;  /* Array for GETALL, SETALL */
                    struct seminfo  *__buf;  /* Buffer for IPC_INFO (Linux-specific) */
                };
          ID    -1
```

semop

```
p/v
   #includde <sys/types.h>
#includde <sys/ipc.h>
#includde <sys/sem.h>
 : int semop(int semid, struct sembuf *opsptr, size_t nops)
   semid    id
      opsptr struct sembuf{
                  short sem_num;   //
                  short sem_op;    //0:            0 1:     V   -1:     P
                  short sem_flg;   //0: IPC_NOWAIT, SEM_UNDO
              }
       nops
          ID    -1
```

4.4.3.4 /

jz2440\process_ipc\3rd_shm\share_sysv.c

```
01 /*********************************************************************
02  *      1.             .
03             2.             .
04             3.     quit              .
05  *
06  *
07  *
08  *
09  * ------------------------------------------
10  * 2020/05/16       V1.0      zh(ryan)
11  *********************************************************************/
12
13 #include <stdio.h>
```

```
14 #include <stdlib.h>
15 #include <string.h>
16 #include <sys/ipc.h>
17 #include <sys/sem.h>
18 #include <sys/types.h>
19 #include <sys/shm.h>
20 #include <signal.h>
21 #include <unistd.h>
22
23 #define N 64
24 #define READ 0
25 #define WRITE 1
26
27 union semun {
28 	int val;
29         struct semid_ds *buf;
30         unsigned short *array;
31         struct seminfo *__buf;
32 };
33
34 void init_sem(int semid, int s[], int n)
35 {
36 	int i;
37 	union semun myun;
38
39 	for (i = 0; i < n; i++){
40 		myun.val = s[i];
41 		semctl(semid, i, SETVAL, myun);
42 	}
43 }
44
45 void pv(int semid, int num, int op)
46 {
47 	struct sembuf buf;
48
49 	buf.sem_num = num;
50 	buf.sem_op = op;
51 	buf.sem_flg = 0;
52 	semop(semid, &buf, 1);
53 }
```

```
54
55 int main(int argc, char *argv[])
56 {
57 	int shmid, semid, s[] = {0, 1};
58 	pid_t pid;
59 	key_t key;
60 	char *shmaddr;
61
62 	key = ftok(".", 's');
63 	if (key == -1){
64 		perror("ftok");
65 		exit(-1);
66 	}
67
68 	shmid = shmget(key, N, IPC_CREAT|0666);
69 	if (shmid < 0) {
70 		perror("shmid");
71 		exit(-1);
72 	}
73
74 	semid = semget(key, 2, IPC_CREAT|0666);
75 	if (semid < 0) {
76 		perror("semget");
77 		goto __ERROR1;
78 	}
79 	init_sem(semid, s, 2);
80
81 	shmaddr = shmat(shmid, NULL, 0);
82 	if (shmaddr == NULL) {
83 		perror("shmaddr");
84 		goto __ERROR2;
85 	}
86
87 	pid = fork();
88 	if(pid < 0) {
89 		perror("fork");
90 		goto __ERROR2;
91 	} else if (pid == 0) {
92 		char *p, *q;
93 		while(1) {
```

```
94 □□□pv(semid, READ, -1);
95 □□□p = q = shmaddr;
96 □□□while (*q) {
97 □□□□if (*q != ' ') {
98 □□□□□*p++ = *q;
99 □□□□}
100 □□□□q++;
101 □□□}
102 □□□*p = '\0';
103 □□□printf("%s", shmaddr);
104 □□□pv(semid, WRITE, 1);
105 □□}
106 □} else {
107 □□while (1) {
108 □□□pv(semid, WRITE, -1);
109 □□□printf("input > ");
110 □□□fgets(shmaddr, N, stdin);
111 □□□if (strcmp(shmaddr, "quit\n") == 0) break;
112 □□□□pv(semid, READ, 1);
113 □□□}
114 □□□kill(pid, SIGUSR1);
115 □□}
116
117 □__ERROR2:
118 □□semctl(semid, 0, IPC_RMID);
119 □__ERROR1:
120 □□shmctl(shmid, IPC_RMID, NULL);
121 □□return 0;
122 }
```

JZ2440

-

```
arm-linux-gcc share_sysv.c -o share_sysv
```

- NFS

```
cp share_sysv /work/nfs_root/first_fs
```

-

console

```
./share_sysv
```

ProcessCommunication_Image034

Image not found or type unknown

4.5

4.5.1

1. GPIO CPU
2.
3. Linux
4. Linux unix

4.5.2

ProcessCommunication_Image035

Image not found or type unknown

4.5.3

kill

```
 #include <unistd.h>

   #include <signal.h>

int kill(pid_t pid, int sig);

pid

        0     -1     INIT

 sig
```

```
   0    EOF
```

raise

```
 #include <unistd.h>

   #include <signal.h>

int raise(int sig);

sig

   0    EOF
```

alarm

```
 #include <unistd.h>

   #include <signal.h>

 int alarm(unsigned int seconds);

seconds

                EOF
```

pause

```
                    sleep

 #include <unistd.h>

   #include <signal.h>

 int pause(void);

    0    EOF
```

signal

```
              kill killall

     #include <unistd.h>

        #include <signal.h>

     void (*signal(int signo, void(*handler)(int)))(int)

    signo

     handler

       0    EOF
```

4.5.4

jz2440\process_single\send_single.c

```
01 /*********************************************************************
02  *      1.
03  *
04  *
05  *
06  *
07  * ----------------------------------------------
08  * 2020/05/16       V1.0      zh(ryan)
09  *********************************************************************/
10
11 #include <stdio.h>
12 #include <stdlib.h>
13 #include <unistd.h>
14 #include <sys/types.h>
15 #include <signal.h>
16
17 void handler(int signo)
18 {
19  □switch (signo) {
20 □□case SIGINT:
21 □□□printf("I have got SIGINT\n");
```

```
22 □□□break;
23 □□□
24 □□case SIGQUIT:
25 □□□printf("I have got SIGQUIT\n");
26 □□□break;
27 □□□
28 □□default:
29 □□□printf("don't respond to this signal[%d]\n", signo);
30 □□□exit(0);
31 □}
32 }
33
34 int main(int argc, char *argv[])
35 {
36 □signal(SIGINT, handler);
37 □signal(SIGQUIT, handler);
38 □while (1)
39 □□pause();
40 □return 0;
41 }
```

JZ2440

-

```
arm-linux-gcc send_single.c -o send_single
```

- NFS

```
cp send_single /work/nfs_root/first_fs
```

-

```
./send_single
```

4.6 socket

4.6.1 socket

QQ QQ QQ socket

Socket UNIX Unix/Linux “ ” “ open –> write/read –> close”

socket server client

ProcessCommunication_Image036

Image not found or type unknown

socket socket

4.6.2

socket

```
socket

 #include <sys/types.h>

   #include <sys/socket.h>

  : int socket(int domain, int type, int protocol)

          ID   -1
```

bind

```
socket      socket

  #include <sys/types.h>

    #include <sys/socket.h>

: int bind(int sockfd, const struct sockaddr *addr, socklen_t addrlen);

  sockfd     socket()

    addr
```

```
    addrlen:

        ID   -1
```

listen

```
    socket                        server

 #include <sys/types.h>

    #include <sys/socket.h>

: int listen(int sockfd, int backlog);

  sockfd      socket()

    backlog server        client

        ID   -1
```

accept

```
client          server

 #include <sys/types.h>

    #include <sys/socket.h>

: int accept(int sockfd,struct sockaddr *addr,socklen_t *addrlen);

  sockfd      socket()

    addr           client

    addrlen:   client

        ID   -1
```

connet

```
client   server      client

  #include <sys/types.h>

     #include <sys/socket.h>

 : int connect(int sockfd, const struct sockaddr *addr, socklen_t addrlen);

   sockfd      socket()

      addr           server

      addrlen:   server

          ID   -1
```

send

```


  #include <sys/types.h>

     #include <sys/socket.h>

 : ssize_t send(int sockfd, const void *buf, size_t len, int flags);

   sockfd        socket

      buf:

      len:

      flags:    0

□□□MSG_DONTROUTE

□□□MSG_DONTWAIT

□□□MSG_OOB
```

□□□MSG_PEEK

□□□MSG_WAITALL

ID -1

recv

#include <sys/types.h>

#include <sys/socket.h>

: ssize_t recv(int sockfd, void *buf, size_t len, int flags);

sockfd

□ buf:

□ len:

□ flags: 0

MSG_DONTROUTE

MSG_DONTWAIT

MSG_OOB

MSG_PEEK

MSG_WAITALL

ID -1

4.6.3 socket

```
Server

1.  socket

2.  socket

3.

4.

5.  /
```

Client

```
1.  socket

2.  socket

3.

4.  /
```

server　　jz2440\process_socket\server.c

```
01 /*************************************************************************
02  *      1.server  client               client
03  *
04  *
05  *
06  *
07  * ------------------------------------------
08  * 2020/05/16       V1.0      zh(ryan)
09  ************************************************************************/
10
11 #include <stdio.h>
12 #include <stdlib.h>
13 #include <unistd.h>
14 #include <sys/types.h>
15 #include <sys/stat.h>
16 #include <string.h>
```

```
17 #include <arpa/inet.h>
18 #include <sys/un.h>
19
20 int main(int argc, char *argv[])
21 {
22     int lfd ,ret, cfd;
23     struct sockaddr_un serv, client;
24     socklen_t len = sizeof(client);
25     char buf[1024] = {0};
26     int recvlen;
27
28     //  socket
29     lfd = socket(AF_LOCAL, SOCK_STREAM, 0);
30     if (lfd == -1) {
31         perror("socket error");
32         return -1;
33     }
34
35     //
36     unlink("server.sock");
37
38     //   server
39     serv.sun_family = AF_LOCAL;
40     strcpy(serv.sun_path, "server.sock");
41
42     //
43     ret = bind(lfd, (struct sockaddr *)&serv, sizeof(serv));
44     if (ret == -1) {
45         perror("bind error");
46         return -1;
47     }
48
49     //
50     ret = listen(lfd, 36);
51     if (ret == -1) {
52         perror("listen error");
53         return -1;
54     }
55
56     //
```

```
57     cfd = accept(lfd, (struct sockaddr *)&client, &len);
58     if (cfd == -1) {
59         perror("accept error");
60         return -1;
61     }
62     printf("=====client bind file:%s\n", client.sun_path);
63 
64     while (1) {
65         recvlen = recv(cfd, buf, sizeof(buf), 0);
66         if (recvlen == -1) {
67             perror("recv error");
68             return -1;
69         } else if (recvlen == 0) {
70             printf("client disconnet...\n");
71             close(cfd);
72             break;
73         } else {
74             printf("server recv buf: %s\n", buf);
75             send(cfd, buf, recvlen, 0);
76         }
77     }
78 
79     close(cfd);
80     close(lfd);
81     return 0;
82 }
```

client　　jz2440\process_socket\client.c

```
01 /*************************************************************************
02  *      1.client                    server
03  *
04  *
05  *
06  *
07  * ----------------------------------------------
08  * 2020/05/16       V1.0      zh(ryan)
09  ************************************************************************/
10 
11 #include <stdio.h>
```

```c
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <string.h>
#include <arpa/inet.h>
#include <sys/un.h>

int main(int argc, char *argv[])
{
    int lfd ,ret;
    struct sockaddr_un serv, client;
    socklen_t len = sizeof(client);
    char buf[1024] = {0};
    int recvlen;

    //  socket
    lfd = socket(AF_LOCAL, SOCK_STREAM, 0);
    if (lfd == -1) {
        perror("socket error");
        return -1;
    }

    //
    unlink("client.sock");

    //
    client.sun_family = AF_LOCAL;
    strcpy(client.sun_path, "client.sock");
    ret = bind(lfd, (struct sockaddr *)&client, sizeof(client));
    if (ret == -1) {
        perror("bind error");
        return -1;
    }

    //   server
    serv.sun_family = AF_LOCAL;
    strcpy(serv.sun_path, "server.sock");
    //
    connect(lfd, (struct sockaddr *)&serv, sizeof(serv));
```

```
52
53     while (1) {
54         fgets(buf, sizeof(buf), stdin);
55         send(lfd, buf, strlen(buf)+1, 0);
56
57         recv(lfd, buf, sizeof(buf), 0);
58         printf("client recv buf: %s\n", buf);
59     }
60
61     close(lfd);
62     return 0;
63 }
```

JZ2440

-

```
arm-linux-gcc server.c -o server


arm-linux-gcc client.c -o client
```

- NFS

```
cp server /work/nfs_root/first_fs


cp client /work/nfs_root/first_fs
```

-

server client client

```
./server &


./client
```

ProcessCommunication_Image037

Image not found or type unknown

4.6.4 server client

server client

1. client client client
2. client client client