MIT World Peace University

Operating Systems
Second Year B. Tech, Semester 3

Interprocess Communication - Shared Memory Concept Implementation using pipe()

Assignment 2
Practical Report

Prepared By

Krishnaraj Thadesar
Cyber Security and Forensics
Batch A2, PA 20

November 3, 2022

1 Code

```
// C program to illustrate
// pipe system call in C
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

int read_write_eg(void)
{
    int pipefds[2];
    char buffer[40];

    if (pipe(pipefds) == -1)
    {
        // perror("pipe");
        exit(EXIT_FAILURE);
    }

    char user_string[40];
    printf("Enter a String that you wanna send via the Pipe command : ");
    scanf("%s", user_string);
    pid_t pid = fork();

    if (pid == -1)
    {
        exit(EXIT_FAILURE);
    }

    if (pid > 0)
    {
        printf("In the parent rn. \n");
        printf("Writing User String to pipe...: ");
        printf("%s\n", user_string);
        write(pipefds[1], user_string, 40);
        printf("Done.\n\n");
    }
    if (pid == 0)
    {
        close(pipefds[1]);
        printf("In the child rn. \n");
        printf("Reading PIN from pipe...\n");
        read(pipefds[0], buffer, 40);
        printf("PIN from pipe: %s\n", buffer);
        printf("Done.\n\n");
    }
    close(pipefds[0]);

    return EXIT_SUCCESS;
}
void main()
{
    read_write_eg();
}
```

Listing 1: Assignment 6.Cpp

2 Input and Output

```
Enter a String that you wanna send via the Pipe command : assignment6
In the parent rn.
Writing User String to pipe...: assignment6
Done.

In the child rn.
Reading PIN from pipe...
PIN from pipe: assignment6
Done.
```

Listing 2: Input and Output.Cpp

9/11/22

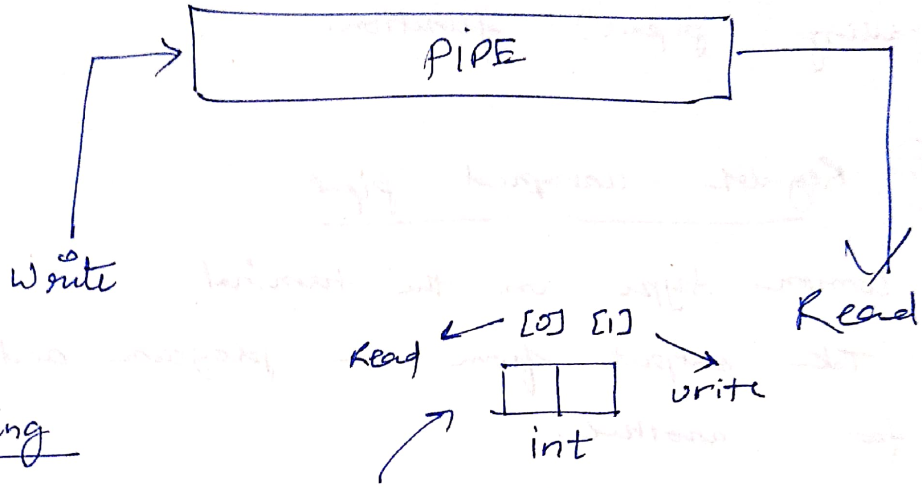
OS- Assignment - 6

FAQs

Krishnaraj PT
1032210888
PA20 ; Batch A1

Q.1: What is pipe and how does it work ?

→ pipe is a communication medium between 2 or more interrelated processes. It can be either within one process or a communication between the child and parent process.



Working

int pipe (int filedes [2]);

→ Process writes to filedes [1] using write function

write (filedes [1], string, MAX);

→ Process then closes filedes [1];

→ Another process reads filedes [0];

→ opens filedes [0]; read (filedes [0], string, MAX);

→ closes filedes [0];

→ This way data is sent and received using pipe between 2 processes

Q.2 Explain types of pipes.

→ There are 2 main types of pipes in UNIX.

① Named pipes :

→ They allow separate processes to communicate by establishing their presence in the file system.

→ Follow FIFO (First in First out)

→ Ones created in C are named pipes upon calling pipe() function.

② Regular - unnamed pipes

→ Common type on the terminal.

→ Take - output from one program and sent it to another.

→ $\# \underbrace{\text{ps -ef}}_{\text{Program 1}} \longrightarrow | \underbrace{\text{grep}}_{\text{Program 2}}$

Q.3 What is a broken pipe? Explain.

→ If a process ~~to~~ tries to ~~a~~ write to a pipe closed by the reader, then that pipe is a broken pipe.

eg. Cat sth.txt | wc -l

If wc -l is dead or over, the pipe breaks from one end.

→ Either pipe creation has failed or ~~dotr~~ one process terminates while other is still reading to its pipe or writing to its pipe.

→ In case of a broken pipe, SIGPIPE signal is given by the kernel.