IPC

Processes in linux do not share address space. Even the child process does not share address space of the parent process. A copy of the parent process' address space is put as the child process' address space at the time of creation. So if a buffer is updated by the parent process before forking the child, the child process can get the updated buffer. But if the buffer is further updated by child/parent, the other process does not see the update. The program fork\_example.c in  <https://shikhan.vbdcss.in/mod/folder/view.php?id=16> shows the behavior.

Unnamed Pipe

If the content of the array updated by the child process has to be made visible to the parent process (or vice-versa), the address of the first element in the array has to be passed through an inter-process communication (IPC) mechanism. The simplest IPC mechanism between a parent process and its child process is called "unnamed pipe". An unnamed pipe is a single directional utility.

Steps of using unnamed pipe between a parent and child process are as follows :

creation of pipe - using pipe() function call. It fills an array of two integers called by reference - element 0 is the file descriptor for read, and element 1 is the file descriptor for write.

Either the parent or child process reads/writes in the pipe. The other descriptor is to be closed in the process where it is not required; i.e if parent process writes to the pipe, it closes the read file descriptor.

read and write to the pipe is done using read() and write() function call. read() ss a blocking function call which returns if there is valid data in the pipe.

C code using unnamed pipe is given in pipe.c in

<https://shikhan.vbdcss.in/mod/folder/view.php?id=16>

Unnamed pipe with non-blocking read

The read() and write() calls are blocking by default. But sometimes it helps to make it non-blocking, especially if some specific task need to be done if there is no data on the pipe. For that, fcntl() is used to make the read file descriptor non-blocking.

For non-blocking read, read() should be called within an infinite loop. In each call, read() can return with three return values : 0 if there is no more data to be sent, > 0 if there is data in the pipe and < 0 if there is an error. errno catches the error, and errno=EAGAIN is actually the case where there is no data on the open pipe.

An example code is given in pipe\_nonblock.c in

Thread Basic

Thread operations include thread creation, termination, synchronization (joins,blocking), scheduling, data management and process interaction.

A thread does not maintain a list of created threads, nor does it know the thread that created it.

All threads within a process share the same address space.

Threads in the same process share:

Process instructions

Most data

open files (descriptors)

signals and signal handlers

current working directory

User and group id

Each thread has a unique:

Thread ID

set of registers, stack pointer

stack for local variables, return addresses

signal mask

priority

Return value: errno

pthread functions return "0" if OK.

Creation of a thread:

A function call pthread\_create() creates a thread. Required header file : <pthread.h>

Function prototype :

int pthread\_create(pthread\_t \* threadId,

const pthread\_attr\_t \* attr,

void \* (\*fptr)(void \*),

void \*arg);

Return value = 0 if successful,else errno contains the error occurred.

Arguments :

threadId : long int typecast as pthread\_t

fptr : Function pointer (typecast to void \*) to the function which is to be executed from the thread context. The argument of fptr is void \* and it returns void \*.

arg : argument pointer of fptr. This void pointer needs to be typecast within the function as necessary.

attr : Attributes of a thread in structure pthread\_attr\_t. If attr = NULL, system defined default attributes are used; otherwise values specified within attr is used. pthread\_attr\_t is defined in bits/pthreadtypes.h .

detached\_state : It says if the thread is joinable or not. It is an enum PTHREAD\_CREAT\_JOINABLE, PTHREAD\_CREATE\_DETACHED.

scheduling policy : PTHREAD\_INHERIT\_SCHED, PTHREAD\_EXPLICIT\_SCHED, SCHED\_OTHER

scheduling parameter

inheritsched attribute (Default: PTHREAD\_EXPLICIT\_SCHED Inherit from parent thread: PTHREAD\_INHERIT\_SCHED)

scope (Kernel threads: PTHREAD\_SCOPE\_SYSTEM User threads: PTHREAD\_SCOPE\_PROCESS Pick one or the other not both.

guard size

stack address (See unistd.h and bits/posix\_opt.h \_POSIX\_THREAD\_ATTR\_STACKADDR)

stack size (default minimum PTHREAD\_STACK\_SIZE set in pthread.h)

Termination of a thread:

A function call pthread\_exit() kills a thread. Required header file : <pthread.h>. Prototype :

void pthread\_exit(void \*retval);

The pthread\_exit function never returns. If the thread is not detached, the thread

id and return value may be examined from another thread by using pthread\_join.

Note: the return pointer \*retval, must not be of local scope otherwise it would

cease to exist once the thread terminates.

Regular expressions

Regular expression is a pattern of characters. For example :

- the string "This word",

- all four letter words starting with 'A' or 'a',

- all 6 letter words starting with 'a' and ending with 'e',

- all lines starting with a number,

- Aadhaar number of a person; i.e 4 numerals followed by a space, 4 numerals followed by a space and 4 numerals again;

- IPv4 address of a host ; i.e three groups of 1,2 or 3 digits followed by '.' and the last group of 1,2,or 3 digits;

- PAN number of a person : 5 alphabets followed by four numerals followed by another alphabet;

Etc

Searching for a regular expression in a text file

grep is a command for searching a regular expression line-by-line in a text file or group of text files, or all files in a directory or all files of different directories recursively. Depending on the given options It can

- highlight the found pattern in the text, or

- show only the lines containing the pattern, or

- show only the lines not containing the pattern, or

- extract the patterns only; which can be collected in an array.

1. Finding the word "pattern" in a file :

Finding lines starting with 5-8 letter words :

^ indicates start of line

\b - Word boundary.

