Fork System Call | Operating System – M01 P08

This is a multipart blog article series, and in this series I am going to explain you the concepts of operating system. This article series is divided into multiple modules and this is the first module which consists of 12 articles.

In this article I am going to explain about “fork” system call. We will see some examples of implementation of fork system call to get a better understanding of the concept.

**Fork:**

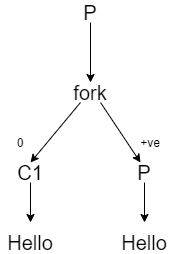
* Fork is a system call.
* It comes under “process control system call”.
* We use fork system call to make child process.
* Suppose, we have a C program, if we write “fork()” in it then a child process will be created. This child process also has its unique ID.
* We use the concept of fork, because let’s assume that the parent process is busy, so a clone will be created which will perform the task which parent process is unable to complete/execute. (Another approach to solve this problem and it is known as “Thread” it is somewhat different from fork.)
* Fork system call generally returns two value 0 and +1 (+ve). 0 denotes child process while +ve denotes parent process. -1 can also be created this happen when due to some reasons child process cannot be created, for example in case operating system/kernel is busy.
* Let us take an example to understand about fork system call.

Main() {

Fork();

Printf(“Hello”);

}



In this case two processes are running parallel, due to the fork statement.

Here “Hello” will be printed two times because, before the execution of the “printf” statement, the fork statement will be executed and a child process is created.

The id of child process is 0, while the id of parent process is +1 (+ve)

* Let’s see one more example; this will help you to understand the concept of fork system call more clearly.

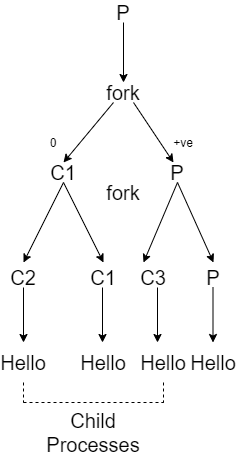
Main() {

Fork();

Fork();

Printf(“Hello”);

}



In this case at the time of 2nd fork C1 and P will behave as a parent. Therefore the following output will be obtained as shown in diagram.

Here “Hello” will be printed 4 times.

* We can calculate the total number of execution directly, it has a formula “total number of execution = 2n”
* While we can also calculate the total number of child process, it also has a formula “Total number of child process = 2n-1”
* N means number of tasks.

This was all about the fork system call, hope you likes it and learn something new from it.

If you have any question, query, doubt or just want to share something with me, then please feel free to contact me.