

## SYSC4001 A1 Report

Repository Link P2: [https://github.com/Xelatrz/SYSC4001\\_A2\\_P2](https://github.com/Xelatrz/SYSC4001_A2_P2)

Repository Link P3: [https://github.com/Xelatrz/SYSC4001\\_A2\\_P3](https://github.com/Xelatrz/SYSC4001_A2_P3)

This simulation models an operating system handling multiple processes using the fork and exec system calls, as well as outputting the execution and a table displaying the system status. This simulation uses trace files to determine what happens during the simulation. These trace files include the following. Fork system call duplicates the parent's memory space and PCB. Exec system call replaces the memory image of the current process with a new one, without creating an entirely new process. The trace files also include a system call that determines whether the child or parent process is running, and a system call to end the I/O. After running these parts of the trace file, the rest of the trace file doesn't really matter because after both the child and parent EXEC calls, they recursively run the rest of the trace in the program files. Each program called using exec has a predetermined trace inside, such as CPU bursts and system calls.

As an interesting example of a trace being run, say we ran the following trace:

```
FORK, 17
IF_CHILD, 0
EXEC program1, 16
IF_PARENT, 0
ENDIF, 0
CPU, 205
```

With the following program1:

```
FORK, 15
IF_CHILD, 0
IF_PARENT, 0
ENDIF, 0
EXEC program2, 33
```

And the following program2:

```
CPU, 53
```

The numbers associated with each step correspond to the duration of the step.

First, FORK runs, creating a child PCB and recursively calling the `simulate_trace()` function on the child's trace (due to `IF_CHILD`). This means the child has control first, while the parent is paused. Child executes program1, which executes FORK again and executes `IF_CHILD`. Since nothing is inside, without processing anything, the simulator continues to `IF_PARENT`, which is the parent of the inner fork, AKA the child of the original trace. This process reaches EXEC for program2, meaning child executes program2: a simple CPU burst of duration 53. When program 2 is over, control returns to the parent inside the original trace file which will execute the CPU burst of duration 205 at the end of the trace.