Report

1.Why break?

We put the break because it prevents the parent process's trace from continuing after the exec call. Once a new program is loaded and its trace begins the execution with recursion, the old process needs to terminate. If we remove the break, the parent trace would keep being executed leading to invalid or duplicated execution steps.

2.Components influence:
PCB (Process Control Block):
Stores all relevant information like its PID, parent PID, program name, size, and memory partition of the process. The PCB data can be seen in the system status output during the simulation.

FORK:
Creates a new child process that is a copy of itself..

EXEC:
Replaces the currently running program in a process with a new program

3.Test analysis

Test 1

Init calls fork that creates a child copy. Then the child loads program1 (10 MB, partition 4) and it executes 100ms CPU burst while the parent waits. Then, Parent loads program2 (15 MB, partition 3) and it executes SYSCALL to device 4.

Test 2:

Init calls fork that creates a child copy. Then the child loads program1 (10 MB, partition 4) while the parent waits. Program1 calls fork creating grandchild. Then grandchild loads program2 (15 MB, partition 3) and executes 53ms CPU burst. Then the parent of program1 loads program2 (15 MB, partition 3, reusing the freed partition) and executes 53ms CPU burst. Finally, the original parent executes a 205ms CPU burst. in program1 is executed by both child and parent of program1 because it's outside conditionals.

Test 3:

Init calls fork that creates a child copy. Child executes 10ms CPU burst and completes. Then parent loads program1 (10 MB, partition 4). Program1 executes 50ms CPU burst, then SYSCALL to device 6 (265ms ISR), then 15ms CPU burst, then END_IO for device 6 (265ms ISR).

Test 4: Multiple Forks with Different Programs (≈2446 ms total)

Init calls FORK, creating a child process. The child executes EXEC program1 (10 MB, partition 4) and performs a 50 ms CPU burst, a SYSCALL to device 6 (265 ms ISR), another 15 ms CPU burst, and an END_IO for device 6 (265 ms ISR). The child then completes and frees partition 4. The parent then calls FORK again, creating a grandchild. The grandchild executes EXEC program2 (15 MB, partition 3) and performs a 53 ms CPU burst, a SYSCALL to device 12 (145 ms ISR), and an END_IO for device 6 (265 ms ISR) before completing. Afterward, the parent resumes, executes EXEC program1 (10 MB, partition 2), and performs a 50 ms CPU burst, a SYSCALL to device 6 (265 ms ISR), another 15 ms CPU burst, and an END_IO for device 6 (265 ms ISR)

Test 5:

Init begins by executing a 30 ms CPU burst before calling FORK, which creates a child process. The child performs a 25 ms CPU burst, then executes EXEC program1 (12 MB, partition 3) while the parent waits. Program1 executes a 60 ms CPU burst, issues a SYSCALL to device 8 (1000 ms ISR), and performs another 35 ms CPU burst before completing. After the child finishes, the parent resumes and performs a 40 ms CPU burst and a 20 ms CPU burst