In UNIX systems, “fork()” is a system call used to create a new process. When a process calls fork(), it makes an exact copy of itself called the child process. Both the parent and child processes run independently from the same starting point, and changes in one don’t affect the other. This allows the system to support multitasking and better process control.

After calling “fork()”:

- The parent gets the child's PID (Process IDentifier) returned.
- The child gets 0 returned.

Based on this return, the parent and child can execute different code.

Activities:

- Memory management: A new virtual memory is created for the child. Uses copy-on-write, meaning pages are only copied when changed.
- PCB creation: A new PCB is created for the child, which includes child’s PID, process state, CPU scheduling information, and resource usage data.
- Resource duplication: File descriptors, environment variables, and other process attributes are duplicated so the child has access to the same resources as the parent.

States:

At first, both parent and child are in the “Ready” state, then take turns running based on CPU scheduling. If a process waits for input or another event, it moves to the “Blocked” state. When done, the child enters the “Exit” state, and stays as a zombie until the parent cleans up.