SYSC 4001 Assignment 1 Group Submission – Part 3 Report

Abdullah Al-Rashid (101310113)

Amin Eldery (101308951)

This part of the assignment implements an interrupt based process simulator that models forking, program replacement using EXEC, and the execution of separate trace files while maintaining correct process control blocks and memory allocation. The core simulate_trace function tracks one running process and a wait queue, advances a global time, and records each event in execution.txt with consistent timing. On a FORK, the simulator clones the current process into a new child with a unique process identifier, allocates memory for the child, moves the parent into the wait queue, and treats the child as the running process. It then records a system_status.txt snapshot that shows the child running and the parent waiting, which matches the required behavior.

The simulator reads the conditional structure of the main trace to separate the child path from the parent path. It executes the child path defined between IF_CHILD and IF_PARENT as the child process, runs it to completion including any EXEC, input output, and CPU events, frees the child memory, and then resumes the parent at the IF_PARENT region. On EXEC, the simulator raises the appropriate interrupt, records the reported program size duration, computes the load time based on the program size, frees the previous address space, updates the process control block with the new program, allocates memory, applies small fixed internal delays, returns to user mode, and writes a status snapshot. It then loads and simulates the trace file matching the new program name from the same directory, so EXEC is modeled as a real change of program followed by correct execution of that program. For the given trace1, program1, and program2 files, the resulting execution.txt shows the child process created by FORK running program1 while the parent waits, and after the child completes, the parent executing program2. The snapshots in system_status.txt occur after FORK, after EXEC of program1, and after EXEC of program2, and they display the correct process identifiers, program names, sizes, partitions, and states. This behavior satisfies the functional and formatting requirements of Part 3 in a clear and consistent way.