Arnab Kumar (21BLC1387)
Sneha Baby Mathew(21BLC1236)
Rohan Raj(21BLC1073)

BCSE303L Operating System

Priority Inversion and Priority Inheritance: A Case Study of the Mars Pathfinder Mission

Priority inversion is a phenomenon in task scheduling where a high-priority task is indirectly preempted by a lower-priority task, contradicting the principle that high-priority tasks should only be preempted by tasks of higher priority. This issue was notably encountered during the Mars Pathfinder mission in 1997.

The Mars Pathfinder's VxWorks real-time operating system (RTOS) uses preemptive fixed-priority scheduling, assigning each task a unique priority level and ensuring the highest-priority task ready to run is the one executing. The priority inversion problem arose in relation to the 1553 bus, which handled data transmission from the lander to the radio and routed command signals from the CPU to the cruise and lander. The critical tasks involved were "bc_sched," responsible for transmission scheduling, and "bc_dist," which determined data recipients.

The issue occurred when the watchdog timer checked if the "bc_dist" task had completed execution. If "bc_dist" was blocked by a lower-priority ASI/MET task, high-priority tasks, such as data transmission, were delayed, leading to resets during weather data collection, a critical operation for the mission.

To address this, the Mars Pathfinder mission employed priority inheritance. This mechanism temporarily elevates the priority of a low-priority process holding a resource needed by a high-priority process, preventing medium-priority processes from preempting the low-priority process and ensuring the high-priority task is not indefinitely blocked.

In the context of concurrent systems, this case study involves the creation of three threads representing different tasks in the system. The shared_resource is a mutex used for synchronization, ensuring only one thread can access the shared resource at a time. The user_input_flag is a volatile flag simulating user input. When set, the high-priority task locks the mutex, performs its task, and unlocks the mutex. The medium-priority task simulates its task by sleeping, and the low-priority task locks the mutex, simulates its task, and unlocks the mutex.

The priority inheritance mechanism is crucial in preventing priority inversion. By temporarily inheriting the high priority, the low-priority task can complete its work and release the shared resource, allowing the high-priority task to proceed without unnecessary delays. This mechanism was key to the success of the Mars Pathfinder mission and serves as a valuable lesson for real-time systems design.