Thread / Core synchronization

A necessary number of parallel execution threads can be independently created and activated from any currently running threads.

Operating system automatically assigns one of a processor core for each thread at:

1. Its creation and first activation;
2. Resuming from inactive state caused by non-cooperative multitasking, software interrupt, or resource waiting.

This indicates that a thread usually “jumps” from its current core to another randomly chosen core.

On the contrary, thread number assigned at the moment of its creation remains unchanged during the entire thread existence time.

The thread number is the only attribute used for thread synchronization by the OS.

Therefore, no special synchronization mechanisms are necessary and exist to address thread synchronization for threads residing on different processor cores since the standard OS object lock and unlock uniformly provides synchronization for threads residing in either same or different cores.

An example of the behavior is shown in the listing below:

