* Conditional variables and Semaphores for Threads Synchronization

(1) **Conditional variables** :- A Synchronization primitive that allows one thread to wait for a signal from another thread When

Page No.:
Date:

Some condition becomes true, instead of constantly checking in a loop. It always work together with a lock (mutex) and solves an issue called "busy waiting" (an issue where CPU cycles are waste just to keep checking whether the resource is free or not) that we see in mutex. Also solves the issue of contention.

② **Semaphores** :-

- Earlier we have only one resource that is being shared by multiple threads/ processes, now in critical section we have 1, 2, 3, ... etc many resources, so now if we have S=3, we can scheduled 3 processes at a time parallely, this is called Semaphore.

- A Synchronization tool that controls access to shared resources by multiple threads. It uses counter to keep track of how many threads can access the resource simultaneously.

- Types of Semaphores :-

(a) Counting Semaphore :- Has a counter that can be greater than 1, allows multiple threads.

(b) Binary Semaphore (Mutex) :- Counter 0 or 1, only one thread can access the resource at a time.