4.3 Synchronization Mechanisms in FreeRTOS

FreeRTOS provides several synchronization primitives, the most common of which are Mutexes and Semaphores. Both are built upon a basic data structure called a queue.

1. **Mutex (Mutual Exclusion)**
   A mutex can be thought of as a "key" to a resource. A task that wants to access the resource must first "take" the key. As long as that task holds the key, any other task that also needs the resource must wait. Once finished, the first task must "release" the key so another task can use it.

   Note: In FreeRTOS, the task that takes a mutex must be the same task that releases it.

2. **Semaphore**
   A semaphore is a more general synchronization mechanism than a mutex. It acts like a counter that controls access to a number of resources.

   Types of Semaphores:
   - Binary Semaphore : Has a maximum count of 1 (it can be either 0 or 1). It is often used for signaling between tasks (event synchronization) rather than for pure mutual exclusion, as it lacks a priority inheritance mechanism.
   - Counting Semaphore : Has a count value greater than 1. It is very useful for managing access to a pool of identical resources, such as connections to a server, memory buffers, or slots in a pool.

Revision #1
Created 2025-08-28 16:07:57 UTC by NS
Updated 2025-08-28 16:12:43 UTC by NS