Real Time Operating systems (RTOS) concepts

Abu Bakr Mohamed Ramadan
Eng.abubakr88@gmail.com

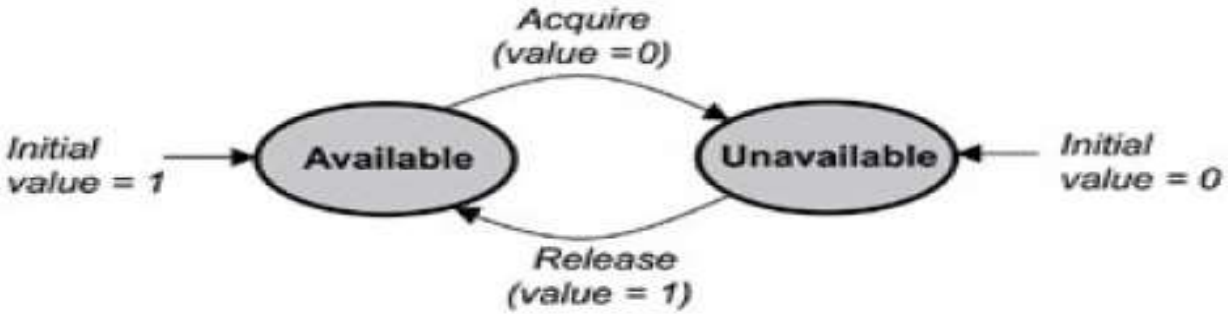
Content:

- **Semaphore Types**
- **Binary Semaphore.**
- **Counting Semaphore.**
- **Mutual Exclusion with Binary Semaphore.**
- **Semaphore in µC/OS-II.**
- **Dead Lock.**
- **Avoid Dead Lock.**
- **References and Read more**

Semaphore Types

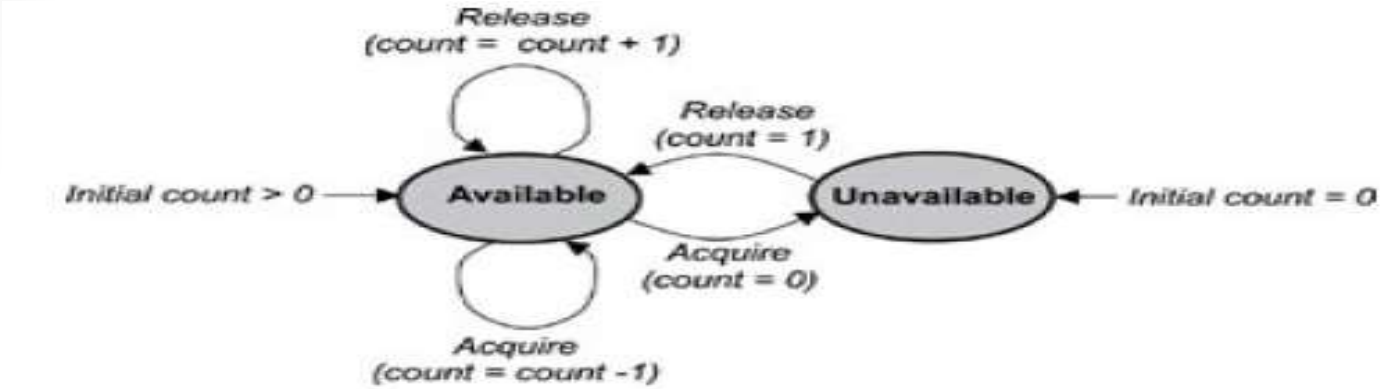
- **Binary Semaphore.**
- **Counting Semaphore.**

Binary Semaphore



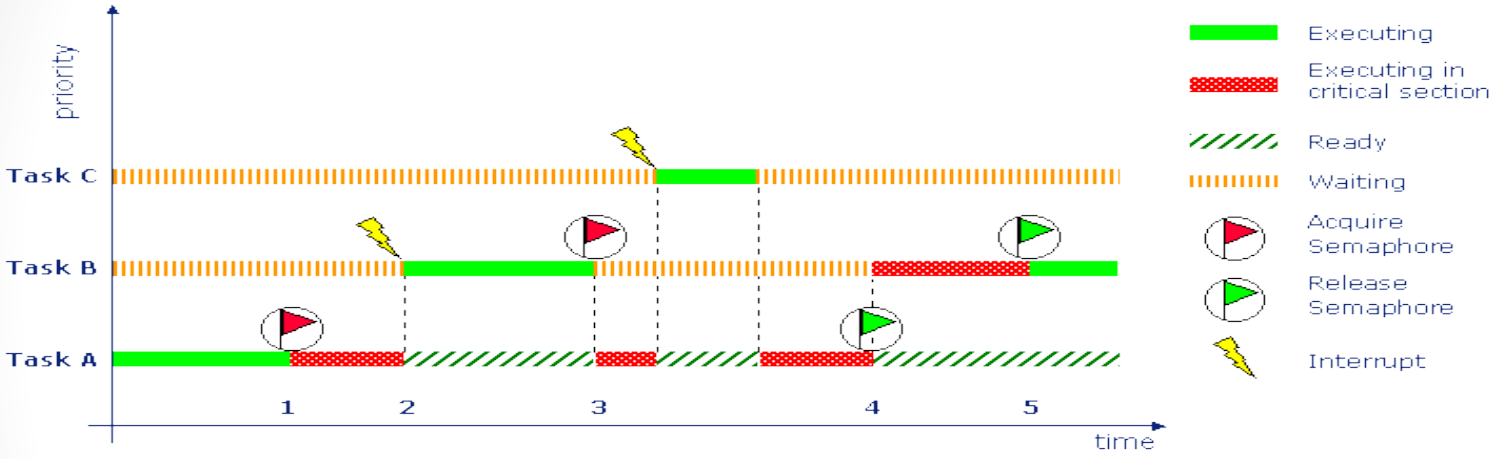
- It’s value = 0, if it’s not available.
- It’s value = 1, if it’s available.
- when a binary semaphore is first created, it can be initialized to either available or unavailable.
- Binary semaphores are *global resources* shared between tasks.
- Any task can release it, even if the task did not initially acquire it.

Counting Semaphore



- When creating a counting semaphore, assign the semaphore a count that denotes the number of semaphore tokens it has initially.
- It's Count = 0, if it's not available.
- It's Count > 0, if it's available.
- counting semaphores are global resources shared between tasks.
- Any task can release a counting semaphore token, even if the task making this call did not acquire a token in the first place.

Mutual Exclusion with Binary Semaphore.



- Using semaphore to access shared data doesn't affect the interrupt latency.
- If ISR or the current running task makes a higher priority task to run it will run immediately.

Semaphore in µC/OS-II.

- An example of accessing shared data using semaphore in **µC/OS-II** :

```
OS_EVENT *SharedDataSem;
void Function (void)
{
    INT8U err;
    OSSemPend(SharedDataSem, 0, &err);
    .
    .    /* You can access shared data in here (interrupts are recognized) */
    .
    OSSemPost(SharedDataSem);
}
```

Dead Lock

- Also called Deadly Embrace.
- When two tasks are waiting the resource held by the other.

- Example
  - Task1 has an exclusive access to Resource1,
  - And Task2 has an exclusive access to Resource2.
  - If Task1 needs an exclusive access to Resource2,
  - And Task2 needs an exclusive access to Resource1.
  - Both the tasks will be blocked, and a DeadLock happen.

Avoid Dead Lock

- Through a timeout, If the resource is not available for a certain time, the task will resume executing.
- Good Design ☺.

References and Read more:

- **Real-Time Concepts for Embedded Systems book** by Qing Li and Carolyn.
  - http://www.e-reading.club/book.php?book=102147
- **An Embedded Software Primer** by David E. Simon.
  - http://www.amazon.com/Embedded-Software-Primer-David-Simon/dp/020161569X
- **Embedded Systems Building Blocks 2e** by Jean J. Labrosse.
  - http://www.amazon.com/Embedded-Systems-Building-Blocks-Ready/dp/0879306041
- **FreeRTOS website.**
  - http://www.freertos.org