Carrera de Especialización en Sistemas Embebidos

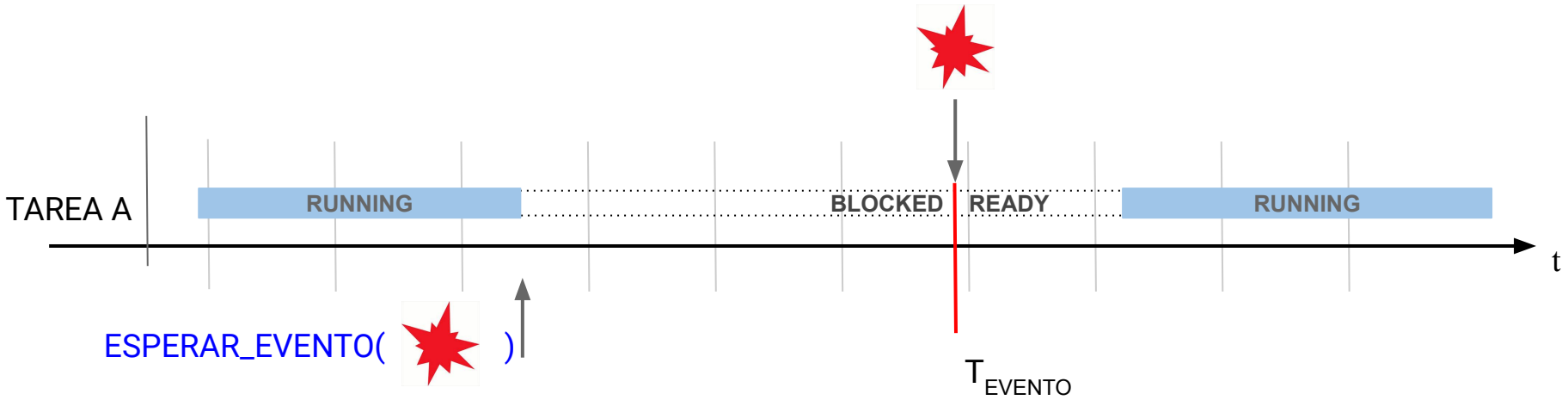
Sistemas Operativos en Tiempo Real

Clase 3: Sincronización de tareas (1ra parte)

ACSE
Asociación Civil para la Investigación, Promoción y Desarrollo de los Sistemas Electrónicos Embebidos
FACULTAD DE INGENIERIA
Universidad de Buenos Aires
CIM
CADIEEL

Problema 1: Sincronización

- ¿Como se puede sincronizar el evento que ocurre en un contexto con una acción que requiere ser disparada en otro contexto a causa de dicho evento ?



Problema 1: Sincronización

Prioridad B < Prioridad A

SEMÁFORO

- Es un tipo de dato abstracto que permite sincronizar tareas mediante un mecanismo sencillo.

- Restringe el acceso a una sección del código hasta que alguien lo señalice (lo ponga "en verde")

- Tipos:
  - BINARIOS
  - CONTADORES
  - MUTEX

SEMÁFOROS BINARIOS EN FREERTOS

- El semáforo binario sólo puede estar en dos estados: 0 o 1

- Una tarea puede tomar (mediante la acción "take") el semáforo. Si el semáforo ya está tomado, entonces la tarea esperará a que alguien lo libere (mediante la acción "give")

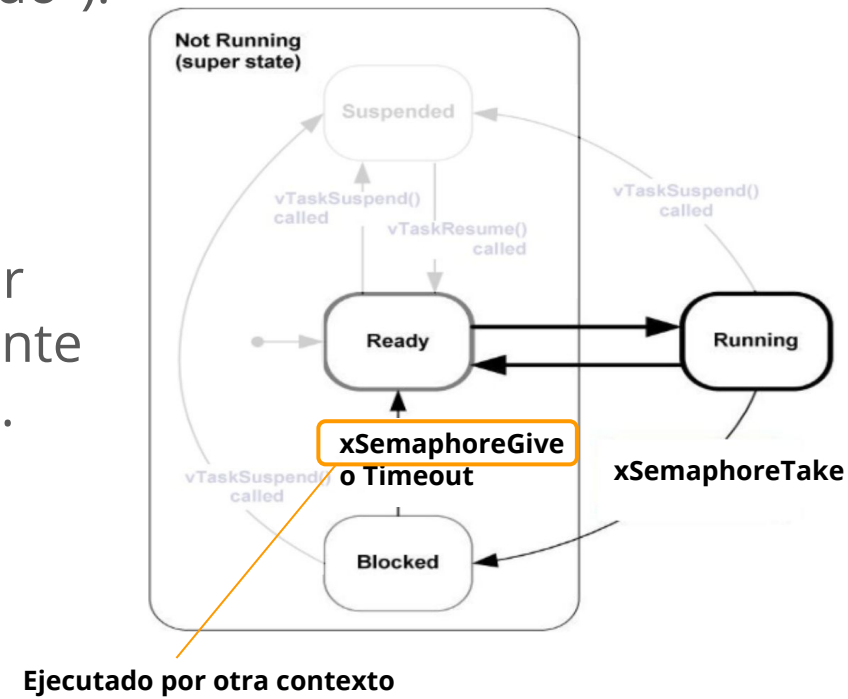
- El semáforo no tiene "dueño". Es decir, lo puede tomar una tarea y lo puede liberar otra.

API: **Semáforos Binarios**

- SemaphoreHandle_t mi_semaforo = **xSemaphoreCreateBinary**();
  - Crea el semáforo (arranca "tomado").

- **xSemaphoreTake**(mi_semaforo , TickType_t xTicksToWait);
  - Toma el semáforo.
  - Retorna pdTRUE si se pudo tomar correctamente y pdFALSE si durante xTicksToWait ticks nadie lo liberó.

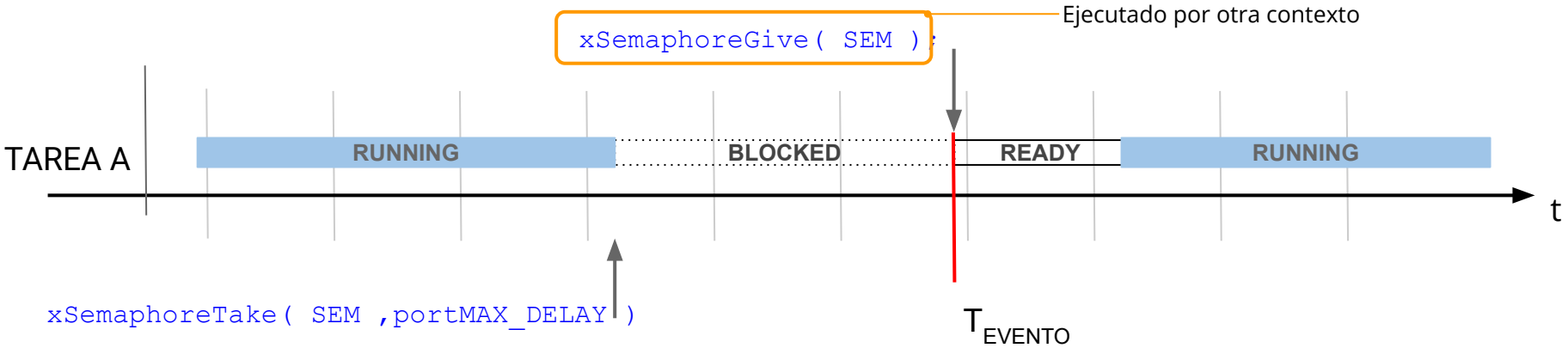
- **xSemaphoreGive**(mi_semaforo)
  - Libera el semáforo.



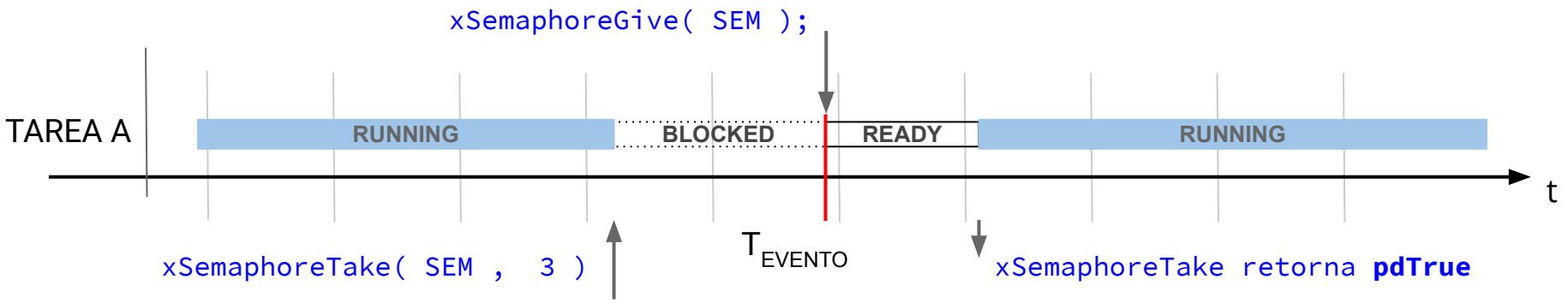
SEMÁFOROS BINARIOS EN FREERTOS

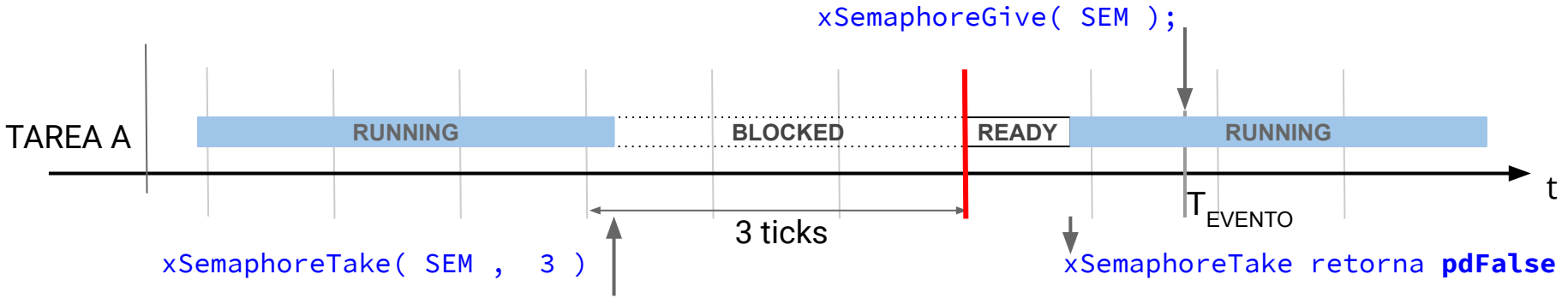
```
SEM = xSemaphoreCreateBinary();          /* se instancia un semáforo
                                            se inicializa "tomado"  */

TAREA_A()
{
     while(1)
     {
          xSemaphoreTake( SEM, portMAX_DELAY );
          /* código que se ejecuta cuando otro contexto llama a xSemaphoreGive */
     }
}
```

Esta macro obliga esperar "por siempre"



SEMÁFOROS BINARIOS EN FREERTOS



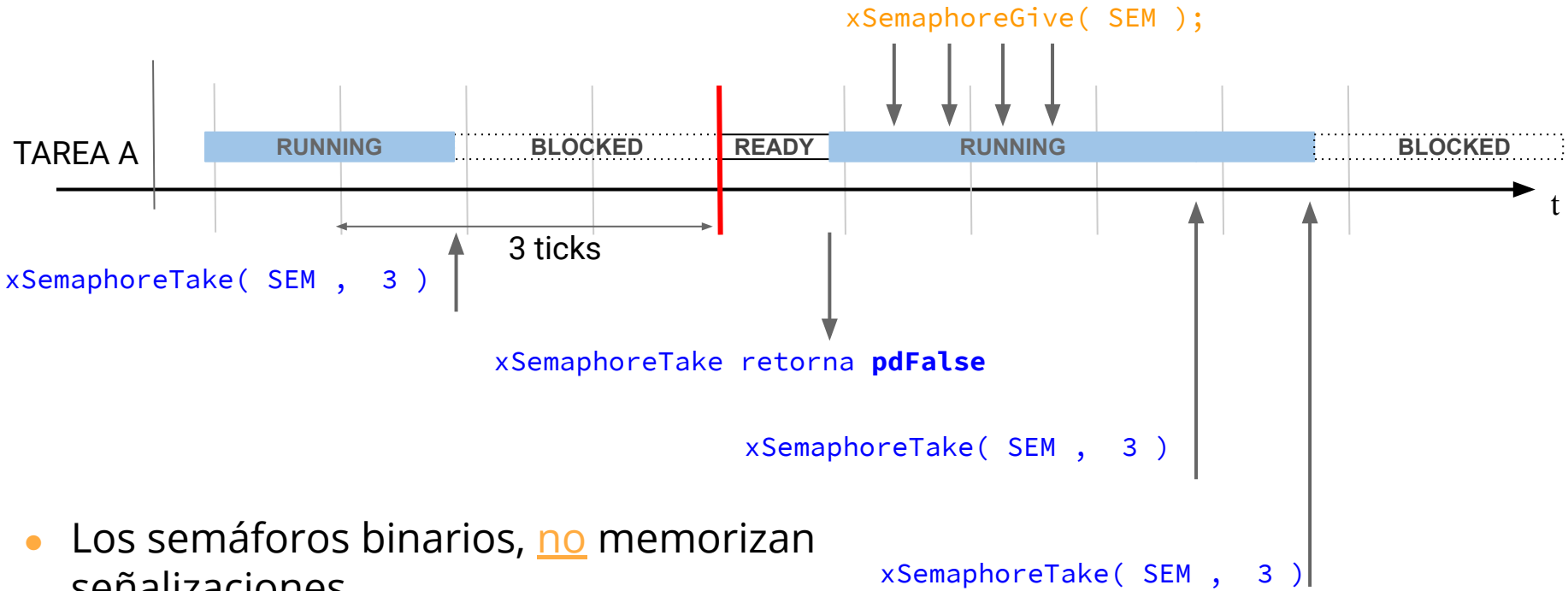


SEMÁFOROS BINARIOS EN FREERTOS

```
SEM = xSemaphoreCreateBinary();        /* se instancia un semáforo
                                          se inicializa "tomado"  */


TAREA_A()
{
     while(1)
     {
         if( xSemaphoreTake( SEM , 3 ) == pdTRUE )
         {
            /* otro contexto liberó el semáforo */
         }
         else
         {
            /* pasaron más de 3 ticks, sin que otro contexto libere el semáforo */
         }
    }
}
```

SEMÁFOROS BINARIOS EN FREERTOS

- ¿Que sucede en este caso ?



- Los semáforos binarios, no memorizan señalizaciones.

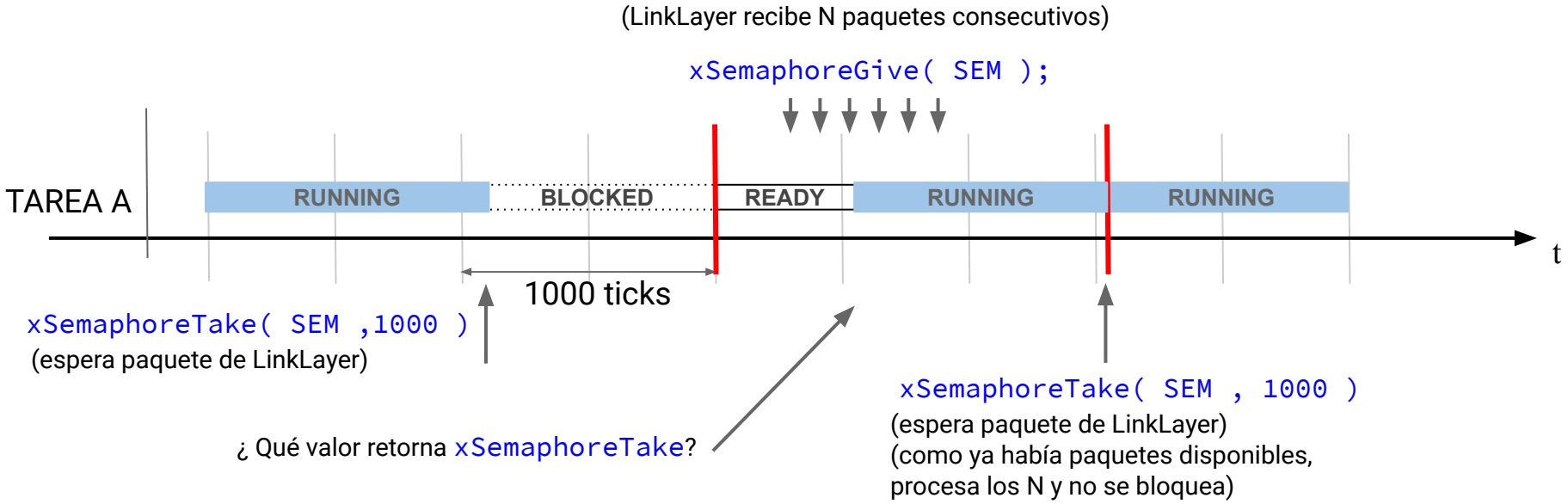
SEMÁFOROS CONTADORES

- El semáforo contador puede estar en N estados: 0 a N-1
- Más de una tarea puede tomar (mediante la acción "take") el semáforo. Si el semáforo ya está tomado, entonces la tarea esperará a que alguien lo libere (mediante la acción "give")
  - "Tomar" un semáforo significa decrementar el contador (y si ya estaba en cero, bloquearse)
  - "Dar" un semáforo significa incrementar el contador.
- El semáforo no tiene "dueño". Es decir, lo puede tomar una tarea y lo puede liberar otra.
- SemaphoreHandle_t mi_semaforo = **xSemaphoreCreateCounting**(MAX , INICIAL);
  - Crea el semáforo contador con un valor máximo MAX y un valor INICIAL preestablecido (recordar que 0 es tomado, y >0 liberado N veces)

SEMÁFOROS CONTADORES: EJ1

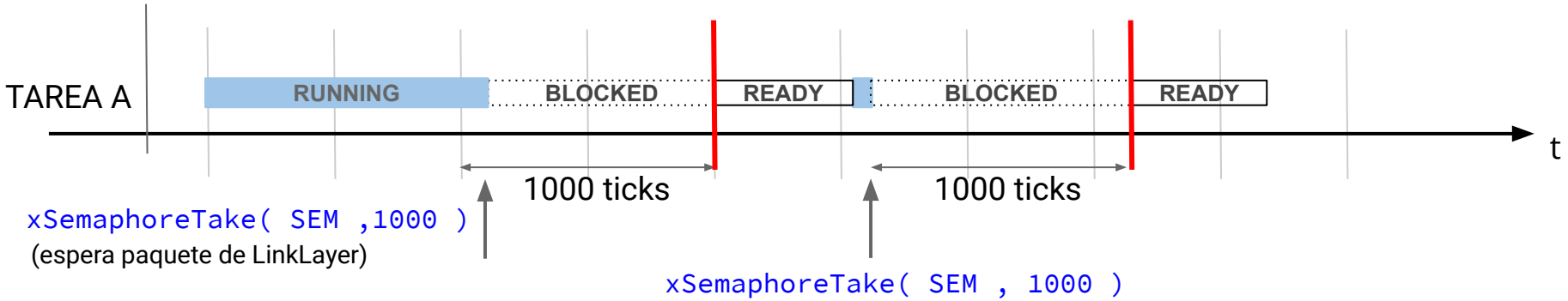
APLICACIÓN	Tarea_A()
LINK LAYER	procesa paquetes y los envía a capa superior
PHY = UART	procesa bytes y los envía a capa superior

(LinkLayer recibe N paquetes consecutivos)



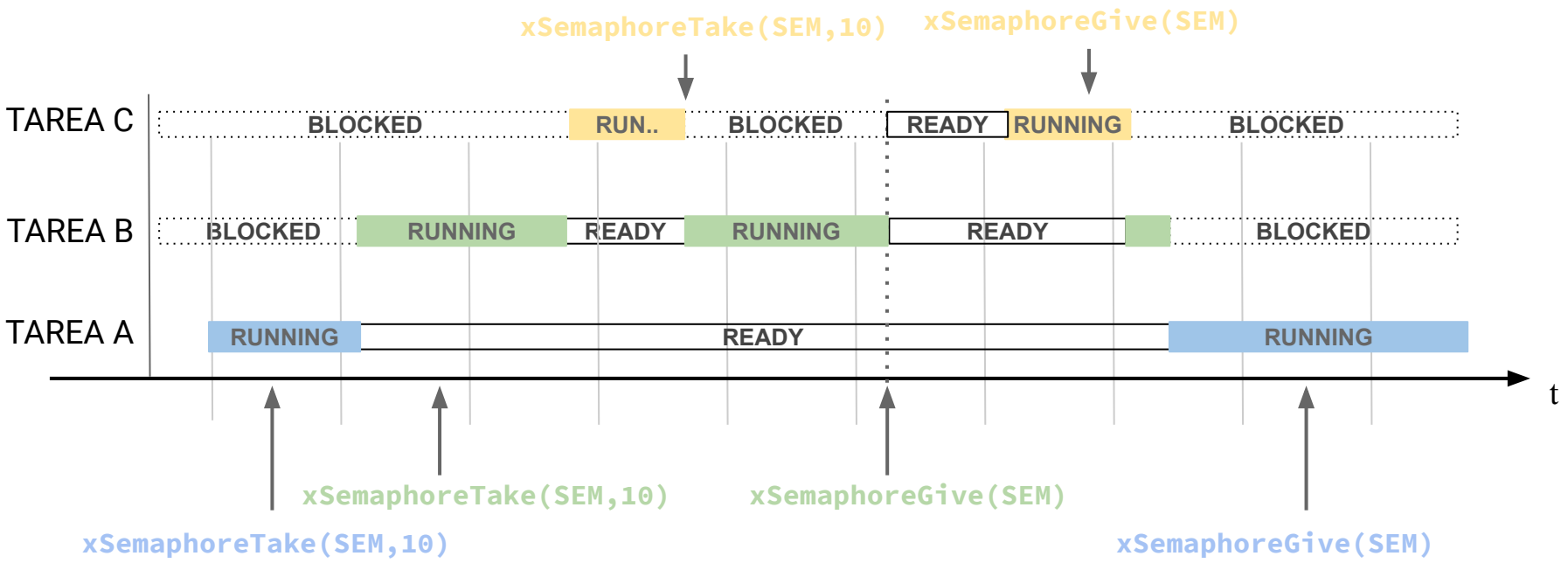
SEMÁFOROS CONTADORES: EJ1 (caso2)

APLICACIÓN	Tarea_A()
LINK LAYER	procesa paquetes y los envía a capa superior
PHY = UART	procesa bytes y los envía a capa superior



SEMÁFOROS CONTADORES: EJ2

Tarea_A(), Tarea_B() y Tarea_C() eventualmente solicitan un bloque de memoria a un gestor de bloques de memoria con solo 2 bloques disponibles.



Bibliografia

- https://www.freertos.org
- Introducción a los Sistemas operativos de Tiempo Real, Alejandro Celery - 2014
- FreeRTOS - Semáforos, Cursos INET, Franco Bucafusco, 2017

Licencia

"Sincronización de Tareas en FreeRTOS (parte 1)"