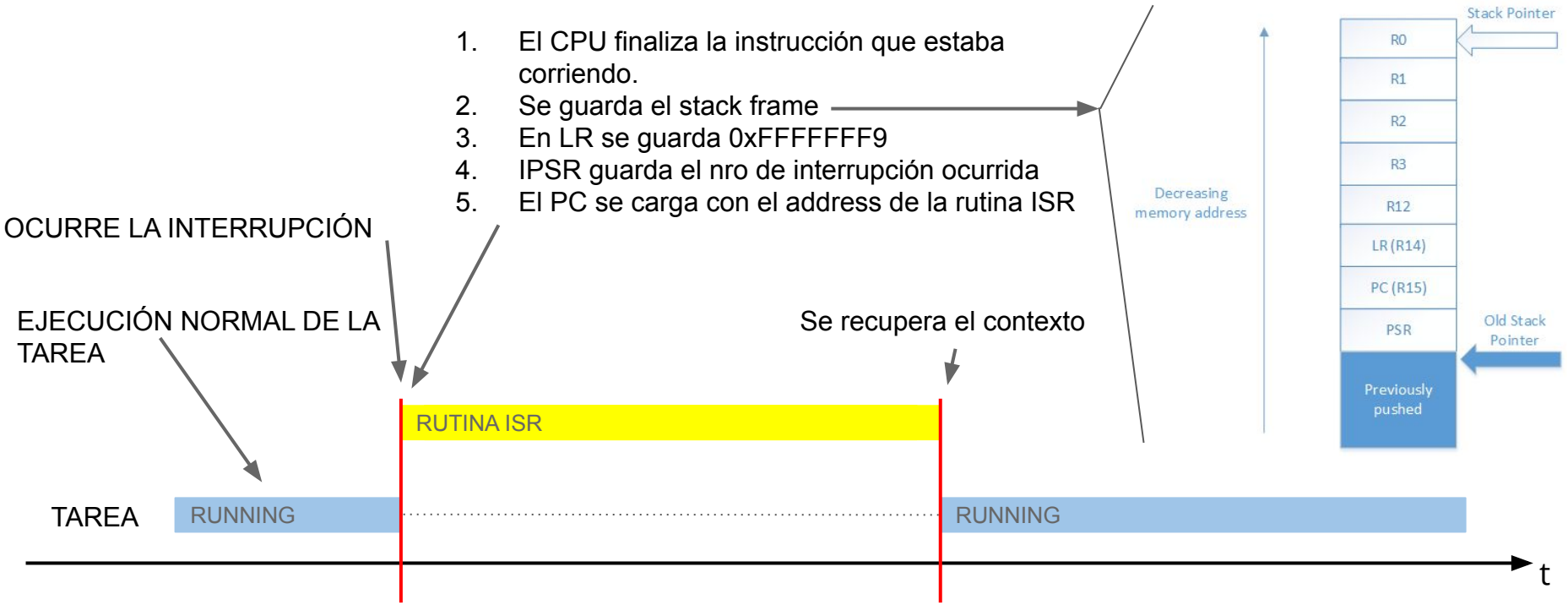
Carrera de Especialización en Sistemas Embebidos

Sistemas Operativos en Tiempo Real

Gestión de Interrupciones.

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

¿QUÉ OCURRE DURANTE LAS ISRs?

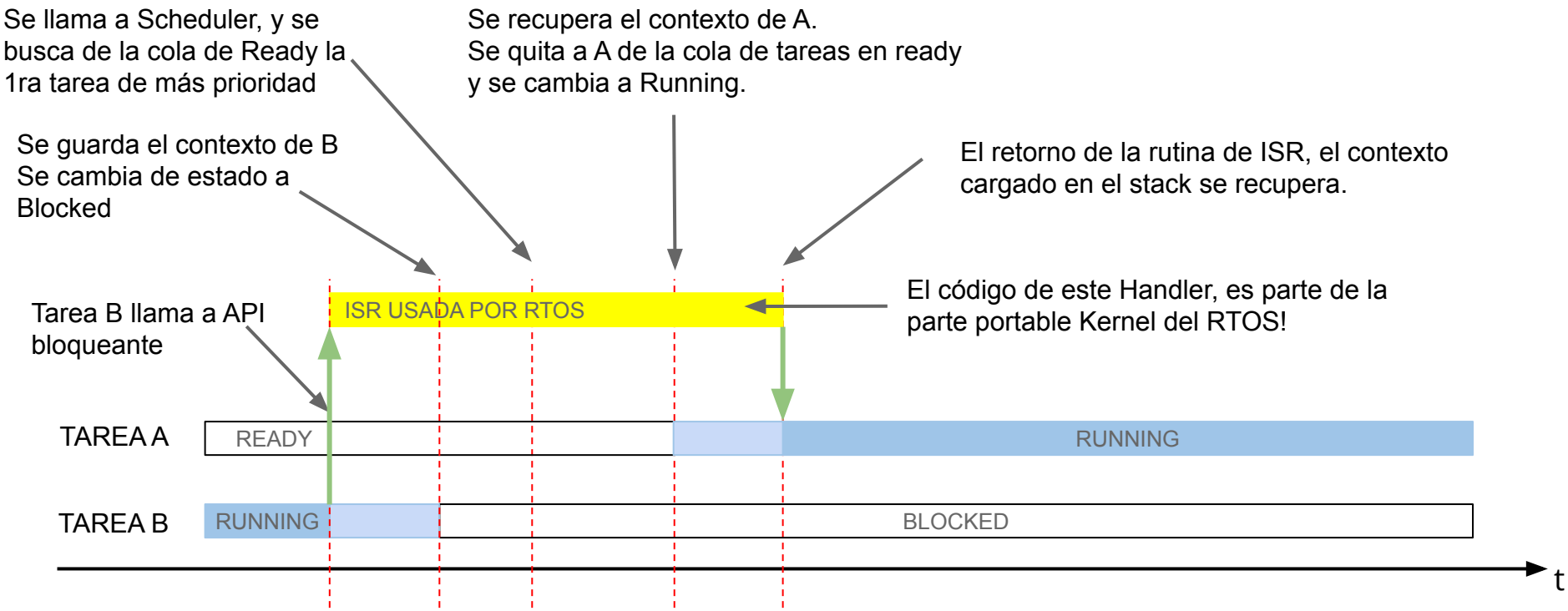


Todo este proceso lo gestiona el hardware: El RTOS no se entera!

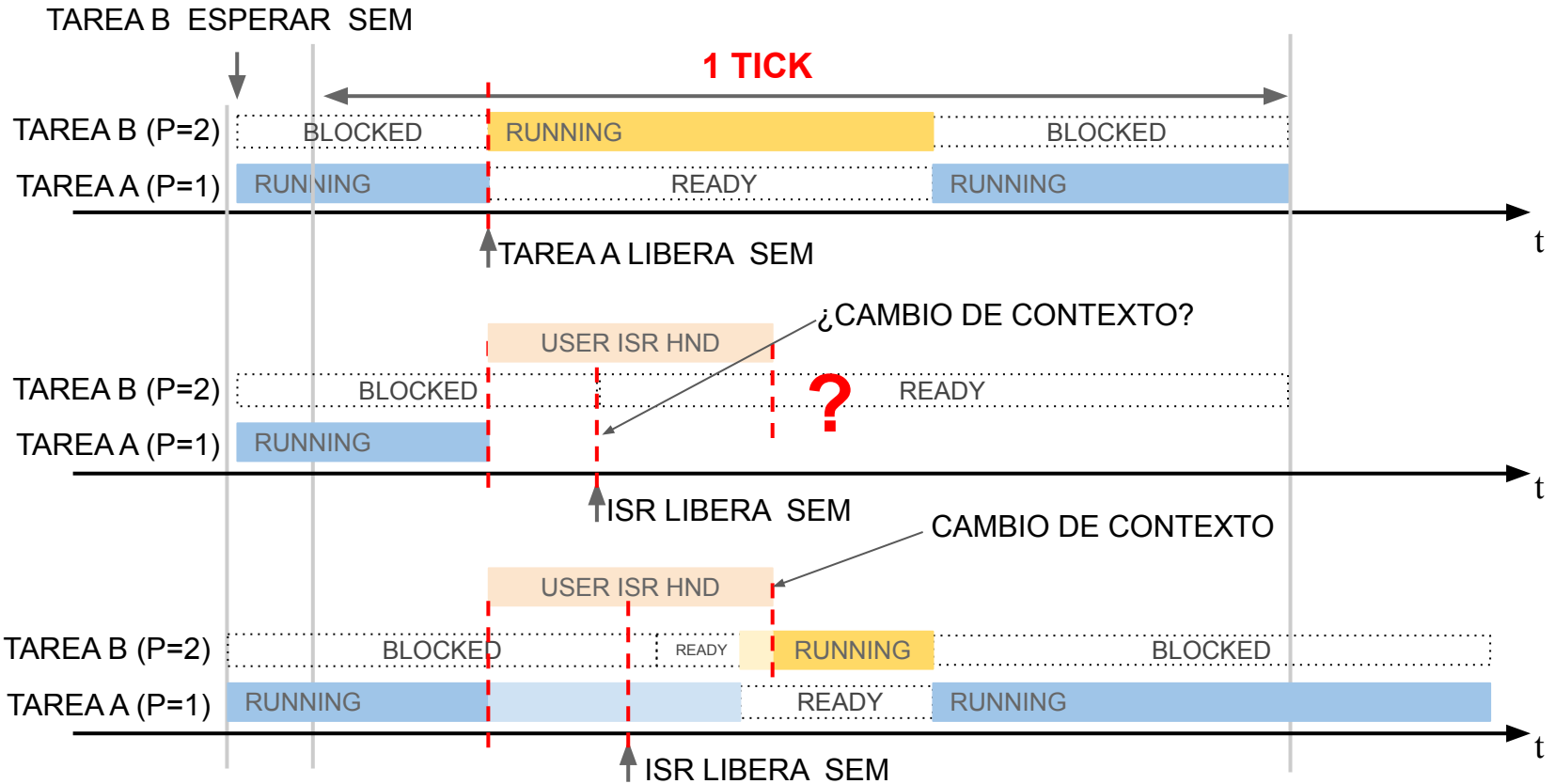
Tareas e Interrupciones

- Las tareas son elementos de software controladas por el RTOS.
- Los handler de interrupción, son controlados por el hardware. Su prioridad no tiene nada que ver con la prioridad asignada a las tareas.
- Los handler de interrupción pueden anidarse.
- Las tareas se ejecutan siempre y cuando no haya ningún handler de interrupción pendiente.
  - El handler de interrupción de menor prioridad (por hardware) será el techo de prioridad de todas las tareas implementadas.
  - O sea, cualquier handler interrumpirá a cualquier tarea, cualquiera sea su prioridad (salvo que las ISR estén enmascaradas)

¿ Cómo es el cambio de contexto ?



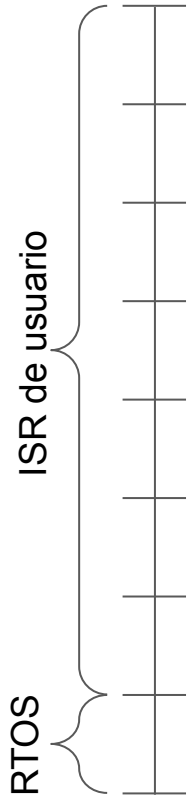
¿Cómo se integran las ISR de usuario al RTOS ?



¿ Que ISRs Utiliza el RTOS ?

- Los cambios de contexto ocurren en la ISR de la menor prioridad de hardware.
  - En ARM las prioridades se pueden cambiar.
  - En otras plataformas, que no poseen recursos de hardware para correr un RTOS, se las arreglan con instrucciones de ASM.

- Las interrupciones utilizadas por un OS, dependen de la plataforma.
  - Por ej: Algunos RTOS sobre ARM utilizan solo en PendSV para generar el contexto. Otros, ademas de PendSV, también utilizan al SVC.



Interrupciones en FreeRTOS

- Todas las funciones de la API que se pueden usar en un handler de interrupción finalizan con **FromISR**()

- Todas las funciones **...FromISR**() llaman al scheduler, para conocer cuál es la siguiente tarea a ejecutar.

- A todas las funciones **...FromISR()** habrá que pasarle un nuevo parámetro que indicará si hay que realizar un cambio de contexto o no.

- Como NO SE PUEDE utilizar bloqueos
  --> NINGUNA de las funciones **...FromISR()** poseen el timeout.

Ejemplo de Handler

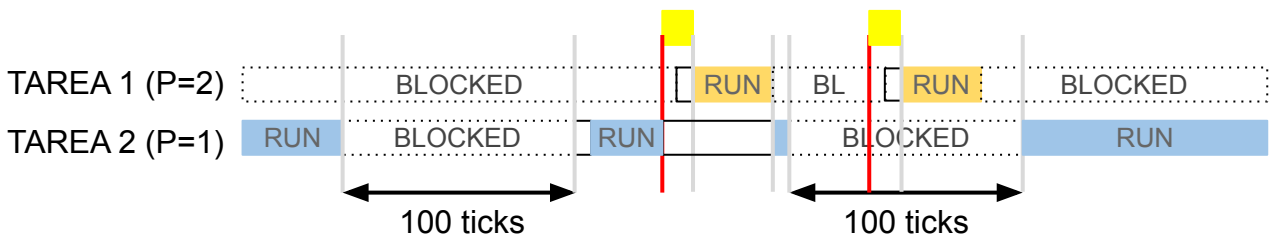
```
void vTimerISR( )
{
    static unsigned char contador= 0;

    BaseType_t desperto_tarea_mas_prioritaria = pdFALSE;

    /* decremento contador */
    contador++;


    if( contador>= MAX )
    {
        /* libera un semaforo */

        xSemaphoreGiveFromISR( xSemaphore, &desperto_tarea_mas_prioritaria );

        /* resetea contador */
        Contador = 0;
    }

    /* si una tarea de mayor prioridad se despertó, se realiza el cambio de contexto */

    portYIELD_FROM_ISR( desperto_tarea_mas_prioritaria );
}
```

Ejemplo:

```
void tarea1(void *p)
{
   while(1)
   {
      xSemaphoreTake( SEM , portMAX_DELAY );
      /* código crítico */
   }
}
```

```
void tarea2(void *p) /* p2 < p1 */
{
   while(1)
   {
      vTaskDelay( 100 );
      /* código no tan importante */
   }
}
```

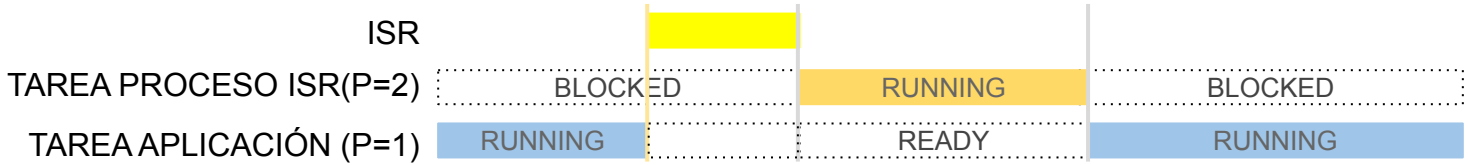
```
void GPIO0_Hanlder( )
{
    BaseType_t desperto = pdFALSE;

    xSemaphoreGiveFromISR( SEM , &desperto );

    portYIELD_FROM_ISR( desperto );
}
```



Buenas prácticas

- Los handlers deben ser **lo más corto posible** porque:
  - Cualquier tarea va a ser retrasada por la ejecución del mismo.
  - El tiempo de ejecución de un handler puede agregarle mucha latencia variable a una tarea crítica.
  - Interrupciones múltiples, podrían empeorar el escenario.

Deferred Interrupt Handling



- El Handler de ISR registra el evento de HW y limpia registros. Señaliza via cola o semáforo de la ocurrencia del evento.
- Una tarea de prioridad mayor al resto, procesa el evento.
- Se utiliza cuando:
  - Se necesitan procesamiento de ISR con tiempo de ejecución largos
  - Existe algún beneficio en usar API normal en vez de usar ...FromISR.
  - Se aceptan algunas latencias extras en el procesamiento de la ISR (menor determinismo)

Bibliografia

- Amazon FreeRTOS – Interrupt management
- Introducción a los Sistemas operativos de Tiempo Real, Alejandro Celery - 2014
- Interrupciones, CAPSE, Franco Bucafusco, 2017
- Interrupciones - Ejemplos con FreeRTOS, CESE, Franco Bucafusco, 2017
- Deferred Interrupt Handling

Licencia

"Gestión de Interrupciones en FreeRTOS"