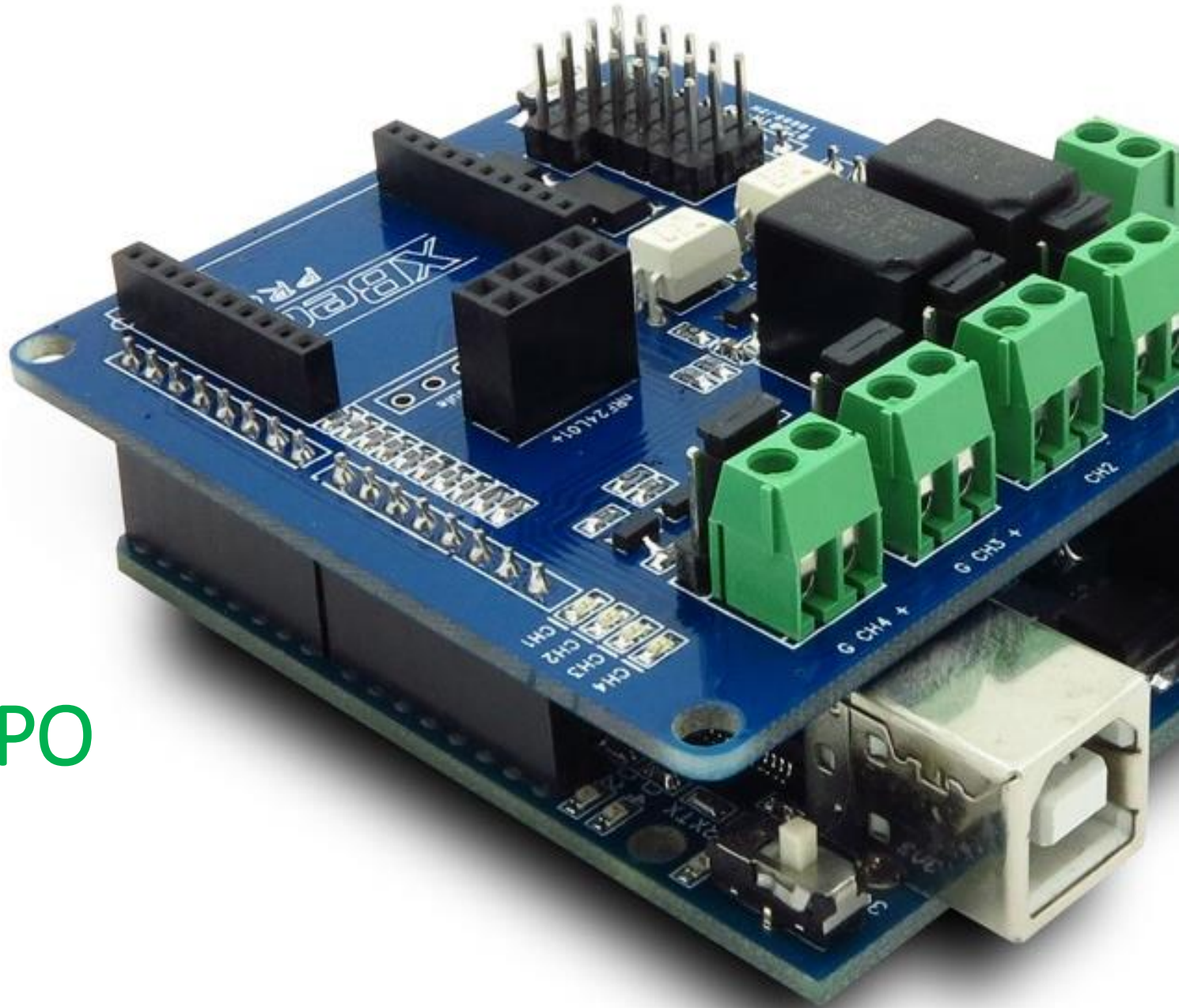


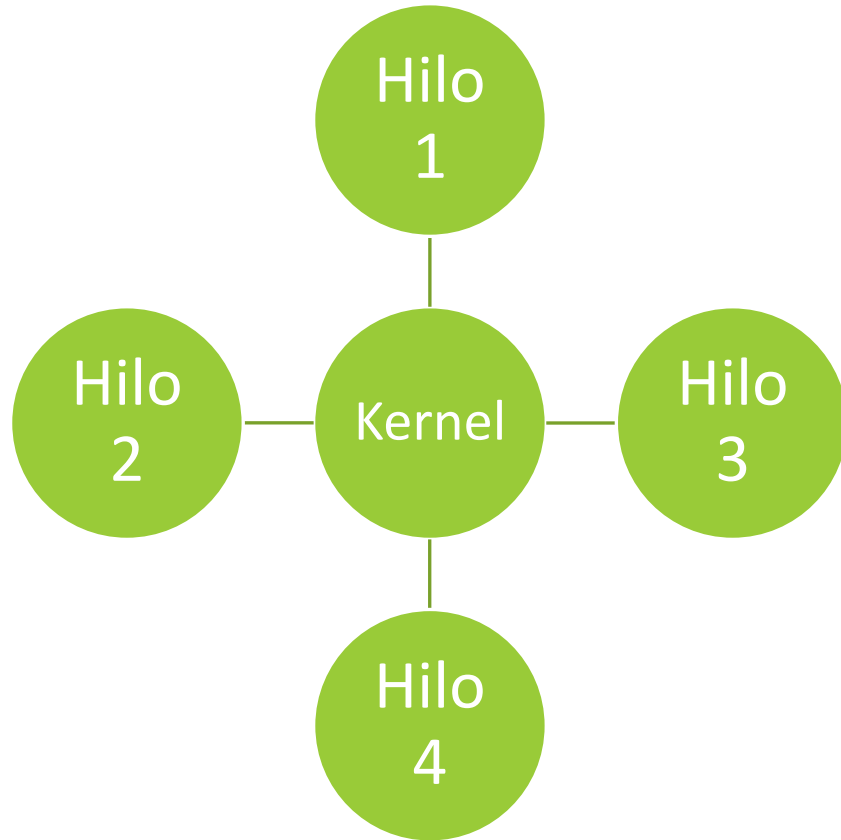


SISTEMAS EN TIEMPO REAL

SISTEMAS EMBEBIDOS



RTOS



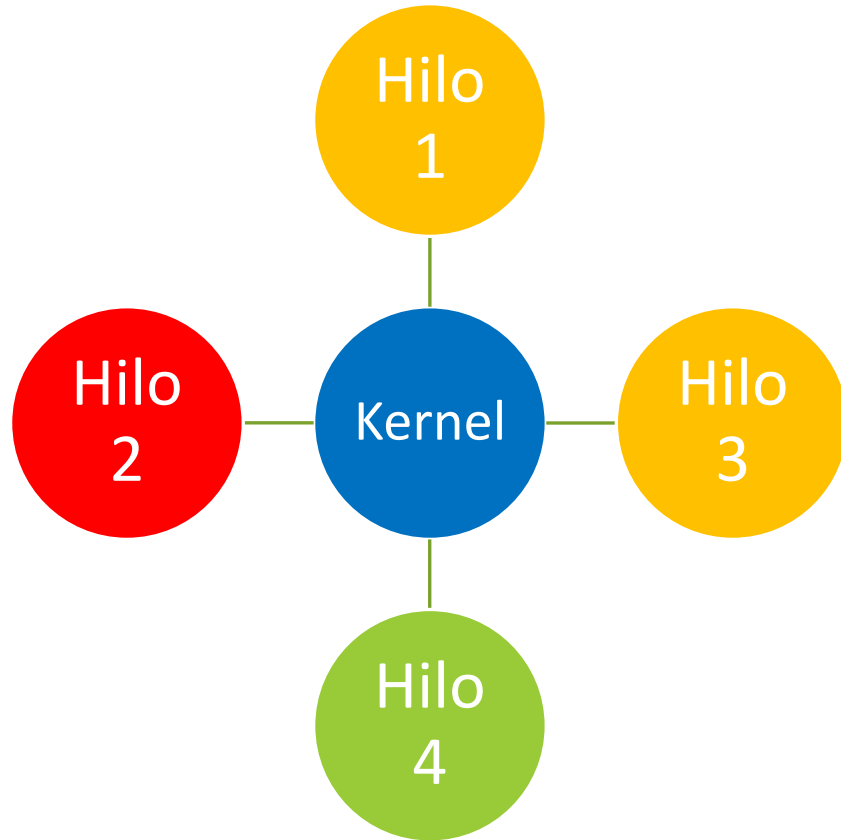
En una computación tradicional cada proceso (hilo) es realizado por la CPU con estructuras establecidas como ejemplo:

FIFO: First In First Out

FILO: First In Last Out

Por esta razón, cada tarea tiene una misma prioridad y se asigna carga computacional según sus necesidades. Esto puede influenciar al rendimiento del sistema.

RTOS



Los sistemas en tiempo real se busca dar prioridad a los procesos que se consideren más importantes. Esto se puede considerar como “semaforización de tareas”, donde a cada una de ellas se les reserva la capacidad computacional necesaria para no producir retardos. En este sentido, los sistemas RTC no se enfocan en reducir el tiempo de respuesta. Su objetivo es ejecutar una tarea en el momento adecuado.

Tipos de Variables

Prefix	Significado
'c'	Int8_t (char)
's'	Int16_t (short)
'l'	Int32_t (long)
'b'	BaseType_t

Funciones

vTaskPrioritySet(): No tiene retorno y es definido dentro de task.c

xQueueReceive(): Retorna un dato de cualquier tipo es definido dentro de queue.c

pvTimerGetTimerID(): retorna un puntero de ubicación y es definido por timers.c

Administrador de tareas

```
void ATaskFunction( void *pvParameters );
```

return : void

argument : pointer to void

```
BaseType_t xTaskCreate( TaskFunction_t pvTaskCode,  
                        const char * const pcName,  
                        uint16_t usStackDepth,  
                        void *pvParameters,  
                        UBaseType_t uxPriority,  
                        TaskHandle_t *pxCreatedTask  
                        );
```

PROGRAMACIÓN

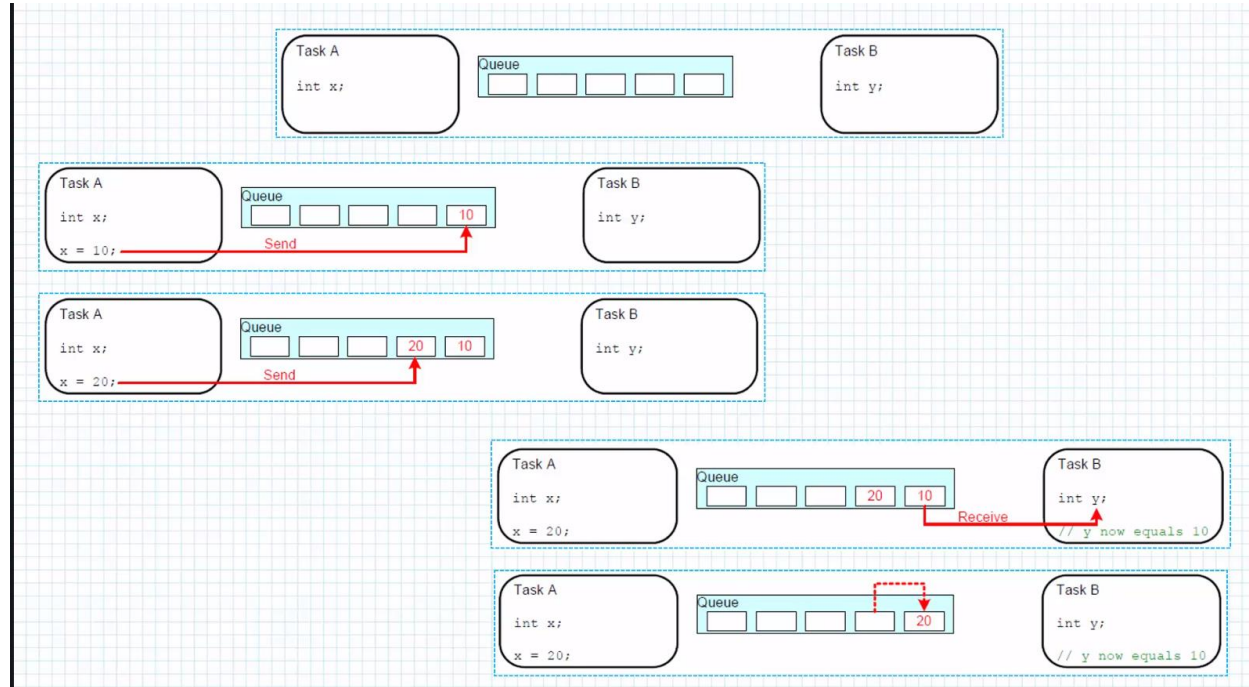


CÓDIGO 25: Creación de funciones en RTOS



CÓDIGO 26: Desactivar funciones en RTOS

COMUNICACIÓN ENTRE TAREAS



COMUNICACIÓN ENTRE TAREAS

```
QueueHandle_t xQueueCreate(  UBaseType_t uxQueueLength,  
                             UBaseType_t uxItemSize );
```

```
BaseType_t xQueueSend(  QueueHandle_t xQueue,  
                        const void * pvItemToQueue,  
                        TickType_t xTickToWait );
```

```
BaseType_t xQueueReceive( QueueHandle_t xQueue,  
                           void * pvBuffer,  
                           TickType_t xTickToWait );
```

PROGRAMACIÓN



CÓDIGO 27: Envío de datos entre tareas

