Comunicación entre tareas

Existen diferentes mecanismos de comunicación entre tareas, los cuales se muestran en la siguiente figura



Estaremos tocando cada uno de estos temas pero prácticamente son facilidades que posee el sistema operativo para realizar acciones de paso de información o espera de eventos entre tareas.

*Señales*

Realizan el trigger de ejecución de estados entre tareas. Practicamente es un segmento de código esperando algo de otra tarea, en teoría un número que bloquea o hace seguir al siguiente paso de las tareas.

Se puede tener de una (1) a treinta y dos (32) señales



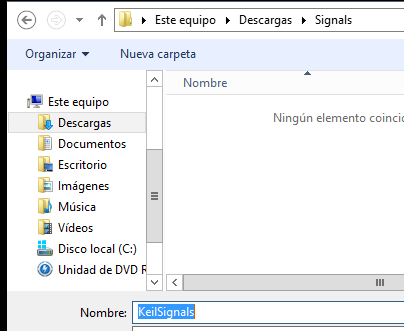
La función listada superior se encarga de hacer señalización de tareas y tiene dos argumentos que son el ID de la tarea y la señal



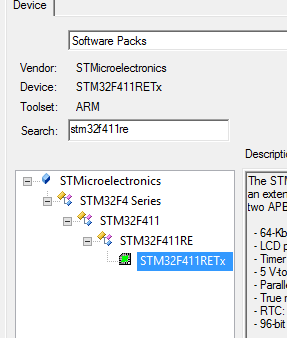
Esta es una función que espera un tiempo por la señal, si la señal existe y esta sometida a bloquear a la tarea esta espera un tiempo y luego cesa.

Implementando señales

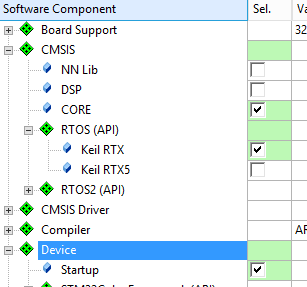
1 – Crear un nuevo proyecto



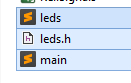
2 – Seleccionar su tarjeta



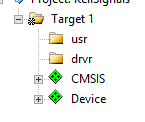
3 – Activar las funcionalidades básicas



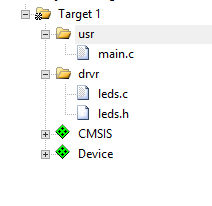
4 – Copiar del proyecto anterior estos archivos dentro de su carpeta de proyecto



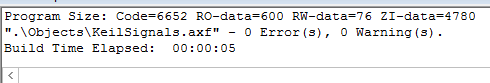
5 – Crear estas carpetas dentro del proyecto de keil



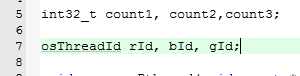
6 – Importe los archivos de la siguiente manera



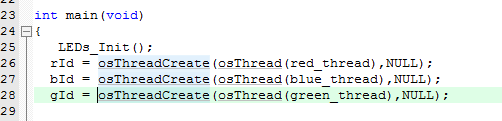
7 – Compile y observe que tendrá 0 errores y 0 advertencias



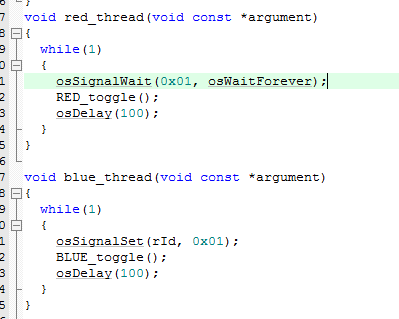
8 – Crear 3 identificadores de tareas



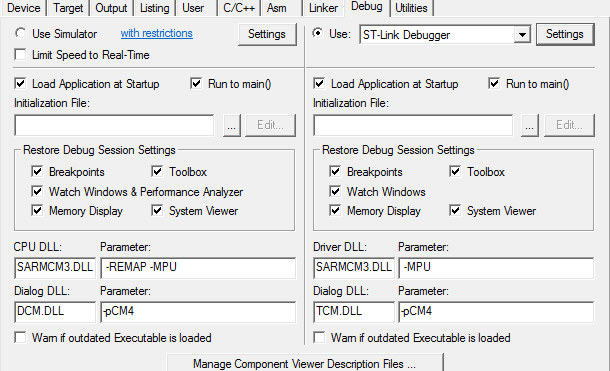
9 – Asigne los identificadores en las tareas creadas



10 – Ahora haremos secuencialidad en las tareas utilizando señales, esto depende del programador, pero realizaremos que primero sea la tarea del LED azul, el LED azul controlara la tarea de la roja y el LED de la roja controlara la tarea del otro led.

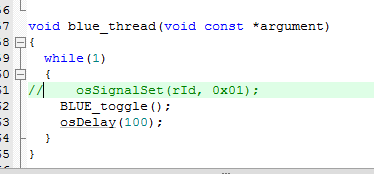


11 – Ahora ajustar las opciones del depurador

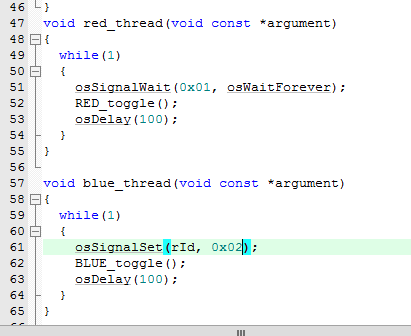


12 – Descargue el programa en el microcontrolador y observe que pasa, deben encender los LEDs como debe ser porque las señalizaciones están llegando.

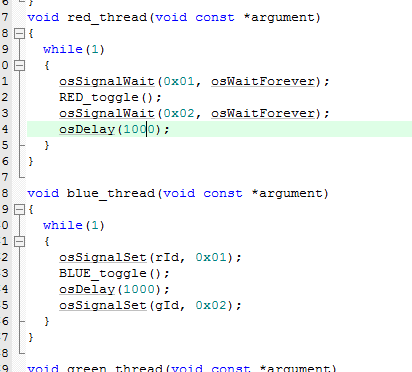
13 – Ahora deshabilite la siguiente línea, compile, descargue y observe que pasa.



14 – De igual manera modifique el código como se observa seguidamente y observe que pasa



15 – Ahora realizaremos sincronización entre tareas para que solamente una se ejecute a la vez, es decir, un led al mismo tiempo.

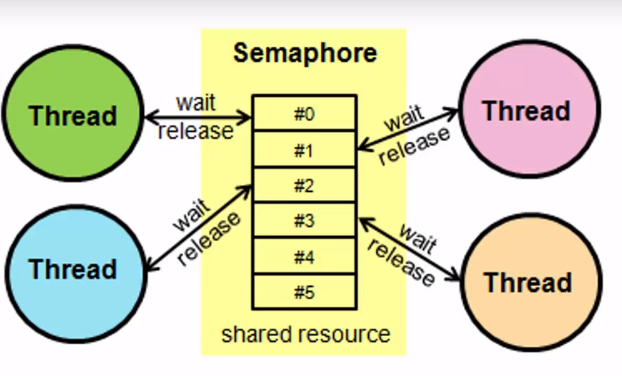


Si ejecuta este código observará que un solo led se habilita a la vez

*Semáforos*

Administran el acceso a recursos compartidos

Permiten a un numero fijo de tareas acceder a un solo recurso



Un semáforo en fin es un contenedor de un número de tokens. Cuando una función se ejecuta esta puede acceder un semáforo que está asociado a un recurso. Dependiendo del número de tokens tomados del semáforo este, para cada función será decrementado en un valor de uno. Si no existe ninguno entonces se deberá esperar a que la función libere el recurso para que pueda ser tomado el semáforo, es decir, el token del semáforo se incrementó en 1.

Trabajando con semáforos

1 – Declarar el contenedor del semáforo e inicializar el semáforo





2 – Inicializar el semáforo con el número de tokes



3 – Tomar el contenedor del semáforo



Realizar para este laboratorio dos rutinas

* Para utilizar señales y controlar los LEDs diferente a lo explicado en clase
* Utilizar semaformos para el control del mismo recurso, p.e. el LED rojo