Práctico I

Implementación de cambio de contexto

Implementación de Sistemas Operativos I

Maestría en Sistemas Embebidos

Año 2023

Docentes

Mg. Ing. Gonzalo Sanchez

Esp. Ing. Hannes Sciarrone

Tabla de contenido

Registro de cambios

Revisión	Cambios realizados	Fecha
1.0	Creación del documento	30/08/2023

Actividades a realizar

Stack Frame

Según lo visto en la teoría, para lograr un cambio de contexto exitoso es necesario que cada tarea creada posea un stack propio y un stack frame válido el cual permita que toda la información en el stack sea persistente entre llamadas a la tarea y la misma siga corriendo en el punto exacto en donde fue interrumpida.

Es necesario que para cada tarea usted implemente:

- Una región de memoria que actuará como el stack de esa tarea.
- Inicialización del stack frame para la primera vez que esa tarea es llamada.

Cambio de contexto

Cuando una tarea es expropiada por el scheduler para ejecutar otra, es necesario generar un cambio de contexto para que la próxima tarea a ejecutar pueda iniciar (o proseguir su ejecución) de forma correcta. El cambio de contexto se encarga de almacenar el contexto actual y obtener el próximo contexto a ejecutar. Aquí es fácil confundir **contexto** con **tarea**, pero son dos conceptos diferentes. Quien selecciona la siguiente tarea a ejecutar es el scheduler (y siempre lo es). Una vez el Scheduler determina cual es la siguiente tarea a ejecutar, el cambio de contexto debe almacenar el valor del stack pointer del sistema (valor del registro MSP) y cargar el stack pointer de la próxima tarea en el registro MSP.

Es necesario que usted implemente:

- Mecanismo de almacenamiento del valor actual del MSP en RAM y carga del stack pointer de la siguiente tarea a ser ejecutada en el MSP.
- Recuerde que el cambio de contexto ocurre dentro de una interrupción y por lo tanto debe ser lo más sucinto posible.

Se recomienda la creación de una estructura de control para cada tarea según se especifica en el documento de requerimientos.

Scheduler

Todo sistema operativo necesita de una base de tiempos para poder funcionar, y usualmente esa base de tiempos tiene una granularidad la cual llamamos ticks del sistema. Por cada tick de sistema, es usual llamar al scheduler con el fin de determinar si es necesario expropiar los recursos (CPU) a la tarea corriendo actualmente y entregarle estos recursos a otra tarea.

Es necesario que usted implemente:

- Base de tiempos del sistema, basado en la interrupción del Systick.
- Llamada del scheduler en cada interrupción (tick).
- Lógica para determinar la próxima tarea a ejecutar (el scheduler en sí).
- Una vez seleccionada la próxima tarea, setear la interrupción PendSV para efectuar el cambio de contexto necesario.

Para la realización de este práctico, el scheduler solo comprenderá una lista de tareas que se ejecuta de forma “circular” (round robin). Esto quiere decir que si existen tres tareas en el sistema, el orden de ejecución será:

Tarea 1 -> Tarea 2 -> Tarea 3 -> Tarea 1 -> Tarea 2 -> …….

Para lograr esto, se recomienda implementar una estructura de control de sistema operativo donde se tenga cuenta de la cantidad de tareas existentes en el sistema operativo. Dado que la máxima cantidad de tareas es finita y está dada por el documento de requerimientos, usted podrá implementar un array estático el cual represente las tareas y acceder a sus elementos para lograr el scheduling desead.