

SEMINARIO: “Diseño y construcción de microrrobots”

Sistemas empotrados en tiempo real

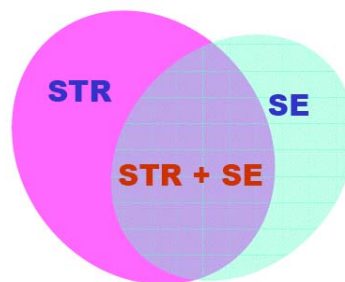
Javier Cid Ortega

Sistemas empotrados en tiempo real

Los sistemas empotrados son los habitualmente utilizados entre otras muchas cosas en la construcción de robots. Debido a la complejidad que están alcanzando los robots y a su dificultad de manejo y programación, se hace necesario el uso de sistemas operativos. Otra importante característica que aporta el uso de sistemas operativos es la portabilidad así como la seguridad que proporcionan en el funcionamiento del sistema y el soporte a posibles fallos. Por definición, un sistema operativo es un conjunto de programas destinados a abstraer el HW de un dispositivo al usuario, gestionando sus recursos de forma eficiente.

Como elementos básicos a la hora de plantearnos el uso de un sistema operativo, necesitaremos, además de un microprocesador o microcontrolador: un dispositivo de almacenamiento no volátil (Memoria NVRAM, Flash...) y memoria RAM.

Los sistemas operativos en tiempo real (STR) y los sistemas empotrados (SE) son dos conceptos que habitualmente van unidos. Estos sistemas son de una gran importancia y nos ayudan a controlar elementos muy dispares, desde centrales nucleares a una lavadora. Se trata de sistemas pequeños, baratos y que deben operar dentro de unos tiempos perfectamente acotados y conocidos. Empezaremos centrando el tema de este trabajo con la definición de estos dos conceptos por separado.



Un Sistema operativo en tiempo real se define como: un sistema que debe responder ante estímulos generados por el entorno dentro de un periodo de tiempo finito especificado.

Un sistema empotrado es un sistema informático de uso específico construido dentro de un dispositivo mayor. Los sistemas empotrados se utilizan para usos muy diferentes de los usos generales para los que se emplea un PC. En un sistema empotrado la mayoría de los componentes se encuentran incluidos en la placa base (la tarjeta de vídeo, audio, módem, etc.). Se enfrentan, sobre todo, al problema de que un fallo en un elemento implica la necesidad de reparar la placa íntegra.

Los sistemas empotrados o embebidos se enfrentan normalmente a problemas de tiempo real. Ejemplos de esto son los: teléfonos móviles, radios, televisiones, sistemas de control de automóviles o aviónica, electrodomésticos, etc.

Un sistema operativo en tiempo real interacciona de forma repetida con el entorno físico que le rodea realizando acciones de control sobre él y reaccionando a sus

cambios. Asimismo dicho sistema posee la habilidad de proporcionar el nivel de servicio requerido con un tiempo de respuesta acotado. Reseñar que no basta con que las acciones sean correctas sino que deben también producirse dentro de un intervalo de tiempo determinado.

Los requisitos temporales que nos podemos encontrar en los sistemas operativos en tiempo real son distintos. Por un lado podemos tener que ejecutar actividades periódicamente (en este caso sería un sistema determinista) y por otro responder a eventos esporádicos.

Según la corrección temporal de dichos sistemas nos encontramos con:

- **Estrictos o críticos:** la corrección temporal es crítica. El tiempo de respuesta es muy importante y no puede ser sacrificado por una mejora en otros aspectos. En ciertos sistemas (por ejemplo sistemas de seguridad de centrales nucleares o centrales hidroeléctricas) la corrección temporal es tan importante que el criterio de corrección lógica puede ser relajado en aras de alcanzar un tiempo de respuesta determinado.
- **No estrictos o acrícos:** la corrección temporal no es crítica. Fallos ocasionales en generar un resultado dentro del tiempo fijado no producen consecuencias serias en el funcionamiento general del sistema. Las tareas de tiempo real no estrictas son ejecutadas tan rápido como es posible, pero no están forzadas por tiempos límite absolutos, pudiendo ser sacrificada la corrección temporal bajo ciertas circunstancias.

Por otro lado y dado que los fallos son inevitables en este tipo de sistemas, éstos deben ser robustos. Según la robustez que posea el sistema en tiempo real tenemos:

- **Sistemas con parada segura** (“fail-safe”): En caso de fallo colocan al sistema en un estado seguro. Esta es una de las directrices que se toman a la hora de elaborar sistemas críticos.
- **Sistemas con degradación aceptable** (“fail-soft”): Presentan una pérdida parcial de funcionalidad o prestaciones en caso de fallo, pudiendo seguir funcionando a pesar de dicha merma.

Estableciendo una última clasificación, esta vez según la activación de las tareas, tenemos los siguientes tipos de sistemas operativos en tiempo real:

- **Activados por eventos o interrupciones.** Cualquier actividad es iniciada en respuesta a la ocurrencia de un evento particular causado por el entorno del sistema.
- **Activados por tiempo.** Las actividades del sistema se inician como instantes predefinidos de un tiempo globalmente sincronizado.

Otro punto importante, quizás en el que haya que hacer más hincapié dentro de un sistema en tiempo real es el control de los procesos. En un sistema en tiempo real el ordenador tiene que interactuar con los diferentes elementos del entorno (sensores, actuadores, etc.), los cuales funcionan de manera concurrente. Para ello es de vital importancia implementar algoritmos de planificación en este tipo de sistemas. Con ello se consigue, como ya hemos dicho, determinar un orden de ejecución de las tareas que sea factible y que satisfaga los requisitos de tiempo y recursos para esas tareas. El planificador en sí está dividido en dos partes:

- **Planificador o scheduler,** que determina quién es el siguiente proceso que tomará la CPU.

- **Dispatcher o repartidor:** conmuta el procesador de un trabajo a otro.

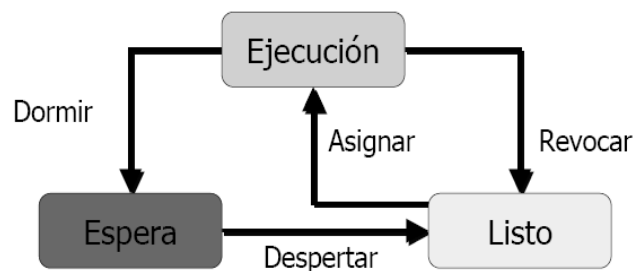
Según como sea el método de planificación tenemos:

- **Métodos estáticos:** el análisis se puede hacer antes de la ejecución.
- **Métodos dinámicos:** el análisis se hace durante la ejecución.

Existen para ello varias políticas de planificación:

- Planificador **round robin** que en rasgos muy generales lo que hace es asignar la CPU a un proceso durante una rodaja o fracción previamente establecida de tiempo.
- Variaciones del SJF (Primero el Más Corto) como el **EDF** (Earliest-deadline First), que divide los trabajos por plazos y selecciona primero el trabajo con el plazo más próximo.
- **Asignación por prioridades monótonas en frecuencia:** este método consiste en asignar mayor prioridad (estática) a las tareas de menor período.
- ...

Estos algoritmos siguen el diagrama de estados básico que posee todo sistema operativo, en el que hay que destacar los estados principales de: listo, espera y ejecución.



Un ejemplo muy claro de la utilidad de los sistemas operativos en tiempo real empotrados es el que se da en el mundo de la automoción. Aquí tenemos diversos sistemas de este tipo, como pueden ser el sistema ABS, el control de tracción o ASR, el Airbag...

En conclusión, lo que conseguimos utilizando los sistemas operativos empotrados es una mayor facilidad de programación del sistema así como una mayor eficiencia. Dado que la programación es más sencilla, nos despreocupamos de muchas cosas que son competencia del propio sistema operativo y que de otra forma tendríamos que implementar. Consecuencia de esto es que tenemos que invertir menos tiempo en la programación. Asimismo los resultados finales son de mayor calidad y eficiencia. Por tanto, y visto lo expuesto en estas páginas parece razonable utilizar un sistema empotrado con sistema operativo siempre que el sistema a implementar tenga una cierta complejidad. Si el sistema es muy simple no merece la pena puesto que el hardware necesario para implementar estos sistemas es caro y ocupa un cierto y preciado espacio, característica muy valorada en algunos proyectos. Por último decir que existen sistemas operativos de libre distribución (gratuitos) a disposición del usuario cuyo código, por puede ser leído por cualquier persona.

Bibliografía:

Documentos descargados de Internet:

- Marga Marcos (2001), Sistemas de Tiempo Real.
- Real, Oscar Sapena Vercher (2004), Programación de Sistemas en Tiempo Real

Transparencias del seminario: Diseño y construcción de microrrobots.