Sistemas de Tiempo Real

Prof. David A. Pérez A.

david.perez@ciens.ucv.ve

Facultad de Ciencias - UCV

Sistemas de Tiempo Real

- Objetivos:
  - Explicar los requisitos de temporización de los sistemas de tiempo real.
  - Distinguir entre los sistemas de tiempo real estrictos y no estrictos.
  - Explicar las características que definen a los sistemas de tiempo real.
  - Describir los algoritmos de planificación para los sistemas de real estrictos.

Sistemas de Tiempo Real

- Definición:
  - “Un sistema de tiempo real es un sistema informático que no sólo requiere que los resultados calculados sean correctos, sino que también esos resultados se produzcan dentro de un período de tiempo especificado”

Sistemas de Tiempo Real

- Ejemplos:
  - Robot autónomo.
  - Sistema informático típico.
  - Sistema de frenado de un automóvil.
    - ABS.
  - Sistemas de seguridad crítica.
    - Marcapasos, etc.

Sistemas de Tiempo Real

- Sistema de tiempo real estricto.
  - Garantizan que las tareas de tiempo real críticas se completan dentro del período especificado.
- Sistema de tiempo real no estricto.
  - Garantizan que las tareas de tiempo real críticas tienen prioridad sobre otras tareas.

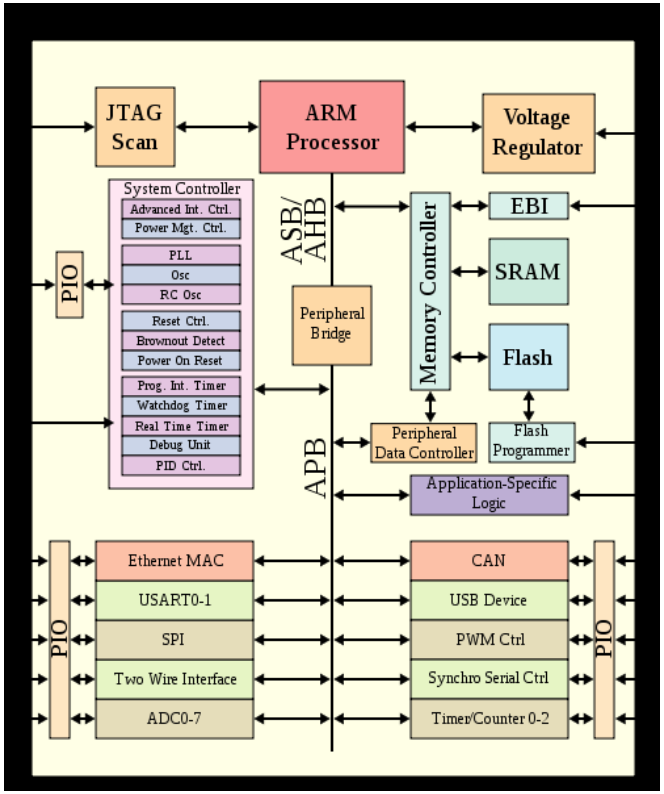
Sistemas de Tiempo Real

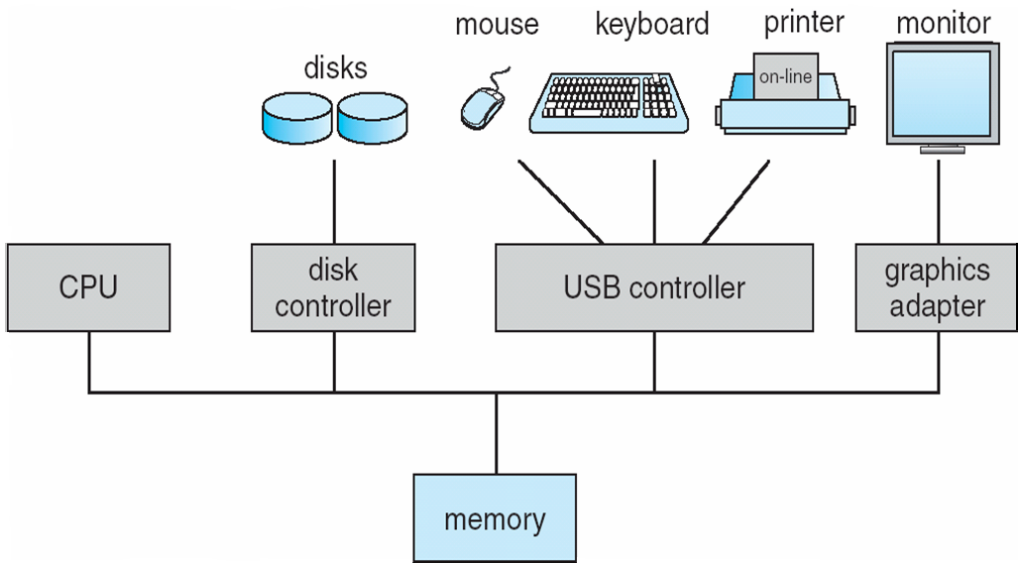
- El sistema operativo y, en particular, el planificador, es quizás el componente más importante de un sistema de tiempo real.
  - ¿Por qué?

Características

- Tienen un único propósito.
  - MP3 y DVD.
- Son de tamaño pequeño.
  - Footprint.
- Son de bajo costo y se producen en masa.
  - Uso (electrodomésticos, etc.).
  - SOC (System On Chip).
- Tienen requisitos de temporización específicos.

SOC vs. Organización tradicional





Características

- Los procesos pueden ser:
  - Aperiódicos.
    - Pueden existir restricciones de inicio o finalización.
  - Periódicos.
    - Exactamente cada T unidades.

Características adicionales

- Determinismo.
  - Controlar el retardo.
- Sensibilidad.
  - Reconocimiento de eventos.
- Control de usuario.
  - Control sobre la prioridad de los procesos.
- Fiabilidad.
- Tolerancia a fallos.

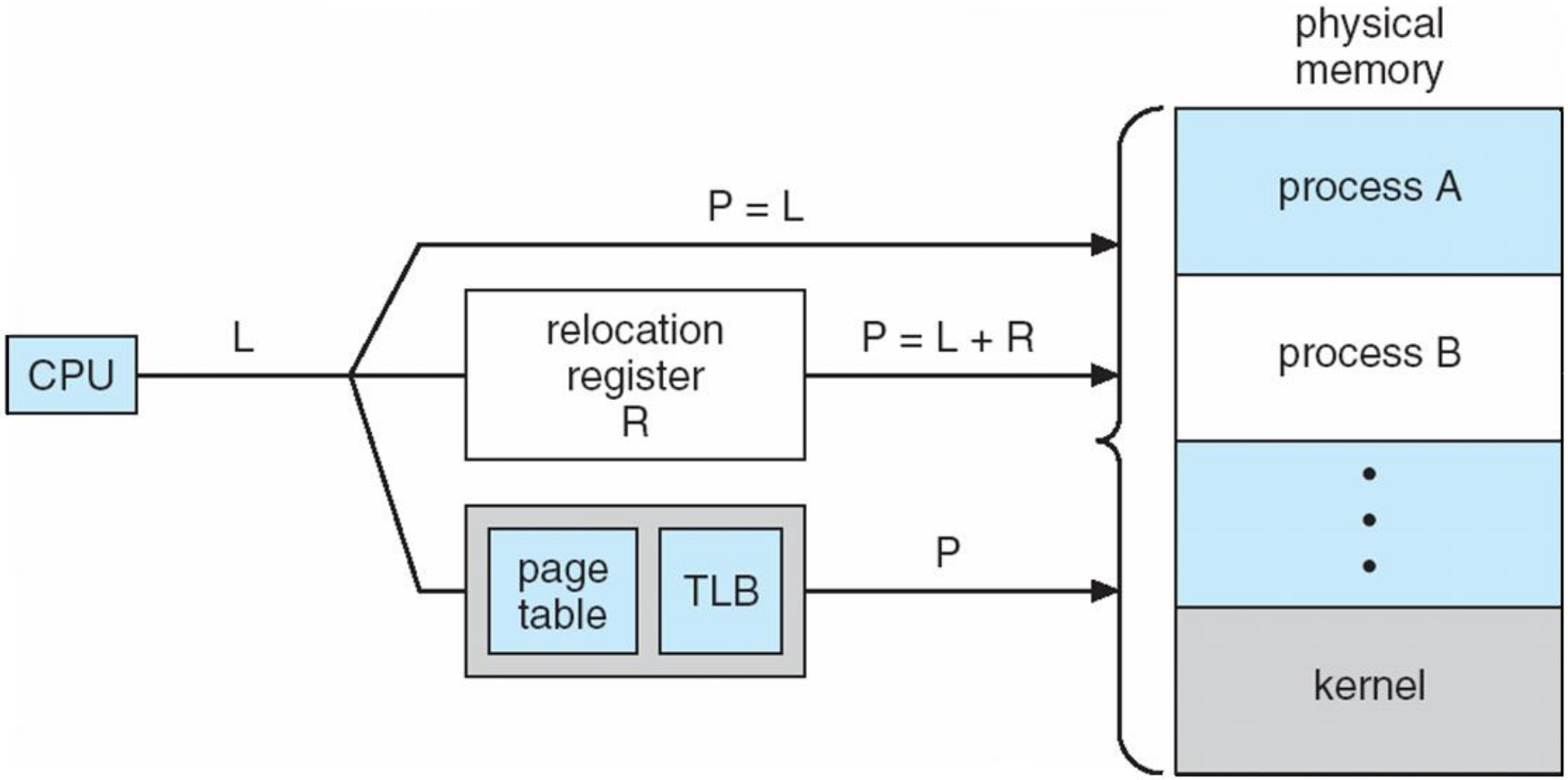
Kernel de tiempo real

- La mayoría de los sistemas de tiempo real no proveen las características encontradas en un sistema de computo estándar.
- Razones:
  - Típicamente son de propósito específico.
  - No requieren de interacción con el usuario.
  - Capacidades limitadas en hardware.
    - MMU.

Kernel de tiempo real

- ¿Cómo realizar la traducción de direcciones?
  - Modo de direccionamiento real.
  - Registro de relocalización.
  - Implementación completa de memoria virtual.

Kernel de tiempo real



Implementación

- Características necesarias en un sistema de tiempo real:
  - Planificación apropiativa basada en prioridades.
  - Kernel apropiativo.
  - Latencia minimizada.

Planificación basada en prioridades

- Responder de manera inmediata a un proceso de tiempo real.
  - ¿Qué debe permitir el planificador?
- ¿Es esto suficiente para las tareas de tiempo real estricto?
  - Ideas.

Kernels apropiativos

- Un kernel no apropiativo no permite desalojar un proceso que se esté ejecutando kernel.
  - ¿Y entonces?
- Un kernel apropiativo permite desalojar un proceso que se este ejecutando en modo kernel.
- ¿Por qué es necesario considerar esta cualidad?

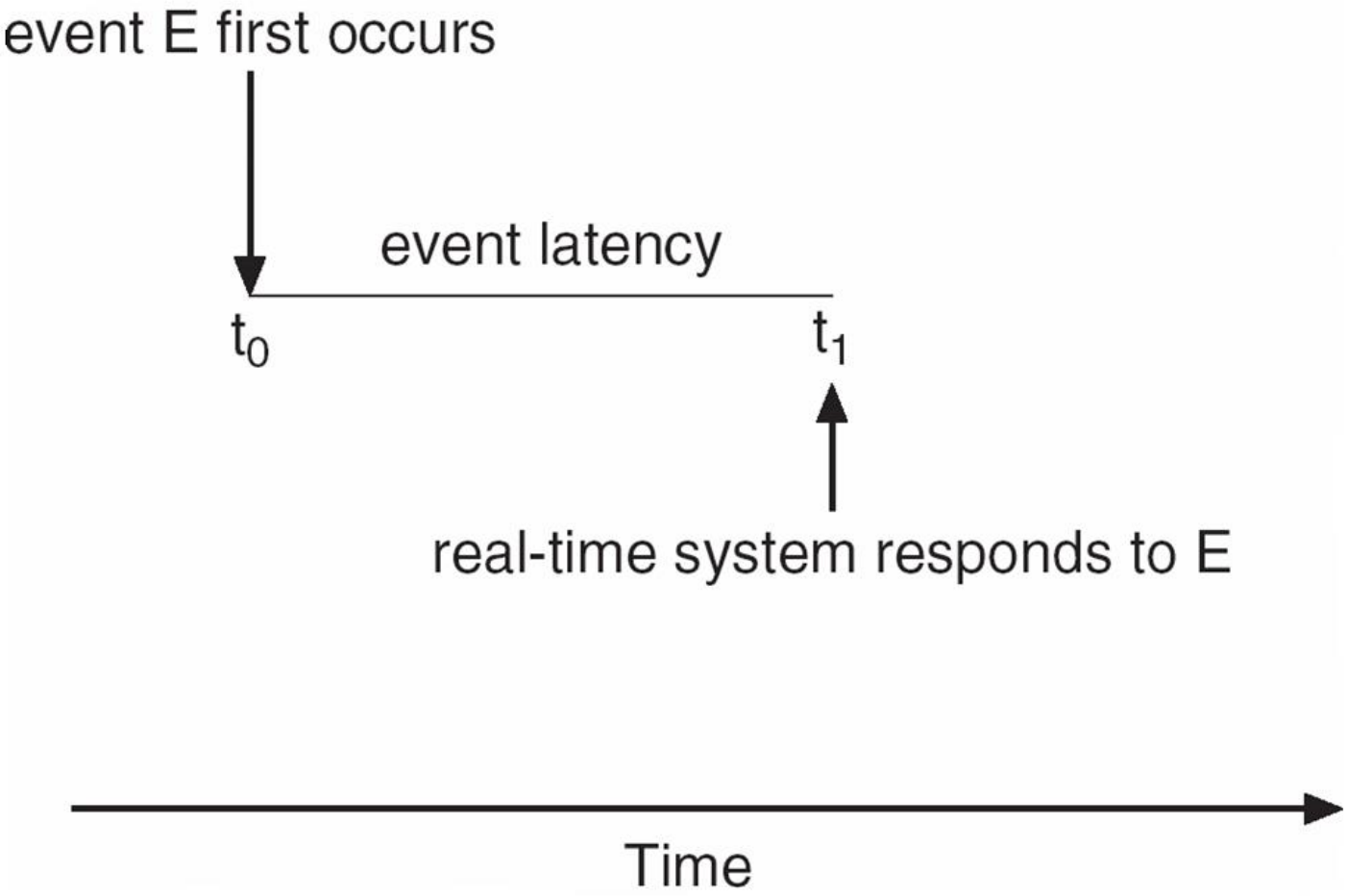
Kernels apropiativos

- Estrategias:
  - Puntos de desalojo.
    - ¿En que consisten?
  - Mecanismo de sincronización.
    - Ideas.

Minimización de la latencia

- Naturaleza dirigida por sucesos.
  - Software.
  - Hardware.
- Denominamos latencia del suceso a la cantidad del tiempo que transcurre desde el momento que tiene lugar el suceso hasta el momento en el que se le da servicio.

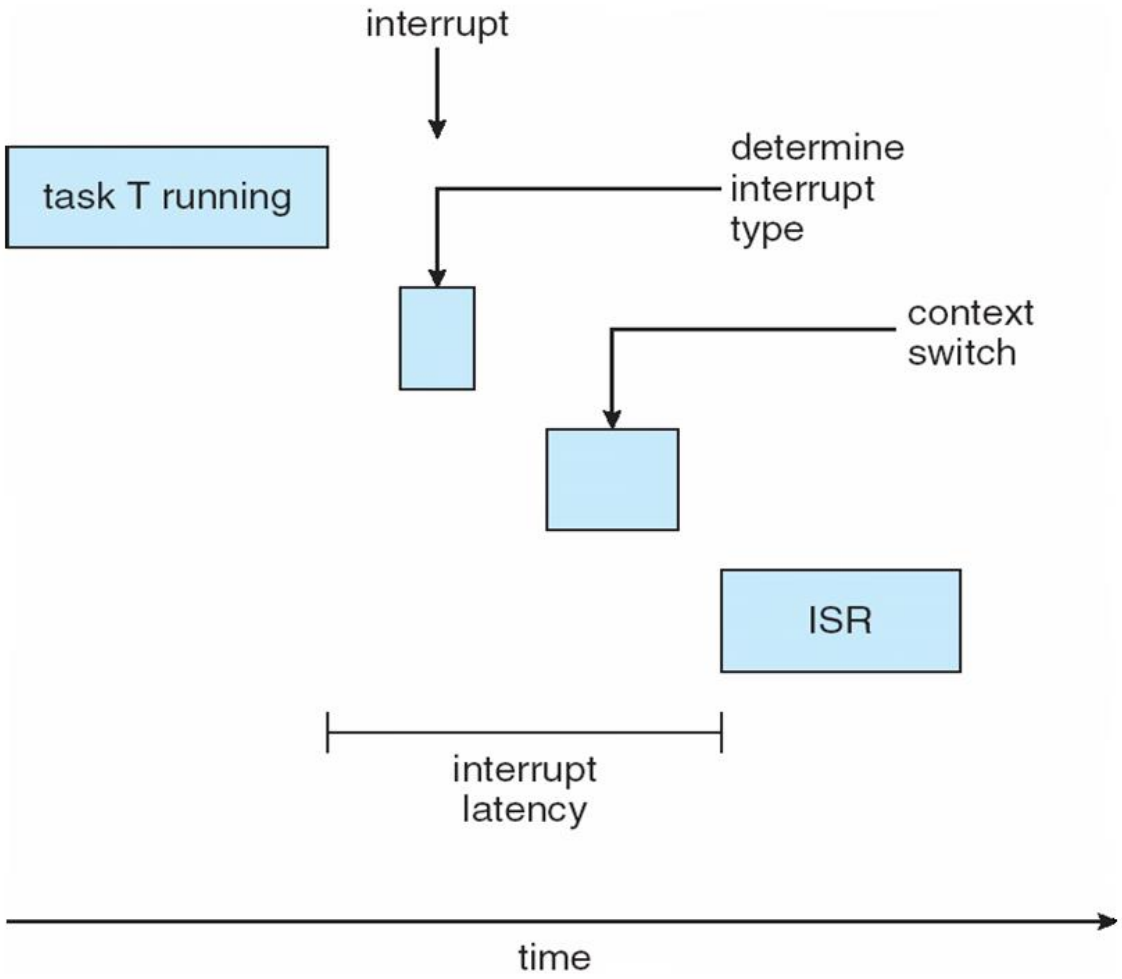
Minimización de la latencia



Minimización de la latencia

- Dos tipos de latencia que afectan a los sistemas de tiempo real:
  - Latencia de interrupción.
  - Latencia de despacho.

Minimización de la latencia



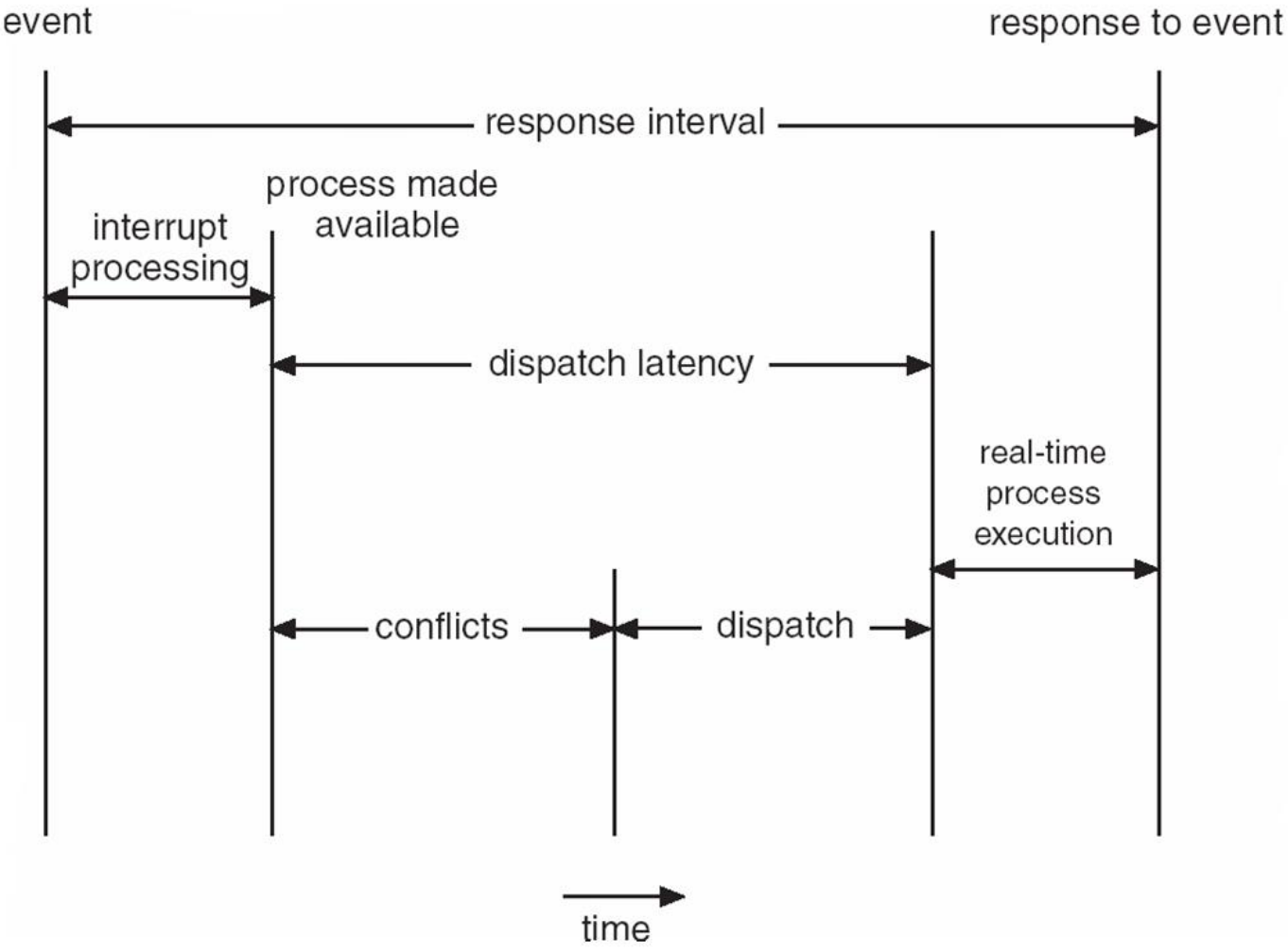
Minimización de la latencia

- ¿Qué ocurren con las interrupciones?
  - ¿Se deshabilitan?
  - Ideas.

Minimización de la latencia

- ¿Cuál es la técnica más efectiva para evitar la latencia de despacho?
  - Ideas.
- Fases de conflicto:
  - Desalojo de un proceso que se ejecuta en modo kernel.
  - Liberación de los recursos retenidos por procesos de baja prioridad, para satisfacer demandas de los de alta prioridad.

Minimización de la latencia



Minimización de la latencia

- Supongamos tres procesos L, M y H; con prioridades:
  - L < M < H
- H requiere un recurso R que posee L.
  - ¿Qué ocurre?
  - M pasa a ejecución y se desaloja al proceso L.
- M afecta el tiempo que el proceso H debe esperar.
  - Inversión de prioridades.

Minimización de la latencia

- ¿Cómo resolver el problema de inversión de prioridades?
  - Protocolo de herencia de prioridades.
  - Ideas.
  - Explicación gráfica.

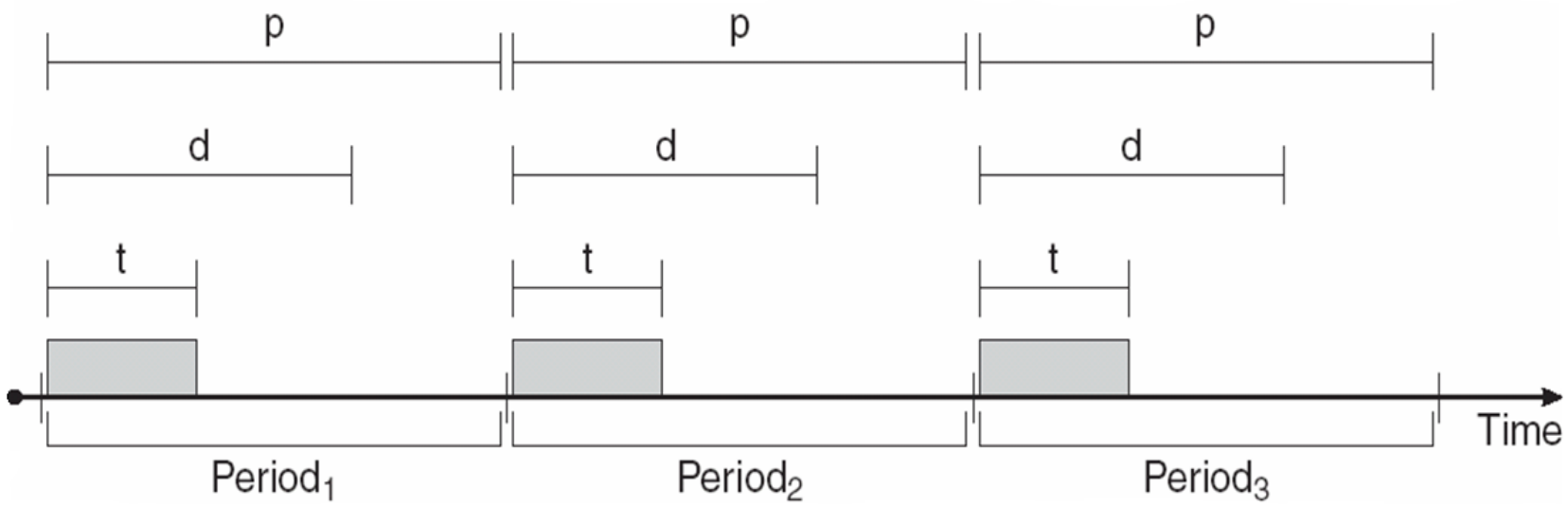
Planificación en tiempo real

- Los sistemas de tiempo real fuerte.
  - Poseen requisitos muy fuertes.
  - Es necesario dar servicio a una tarea en un tiempo prefijado.
    - ¿Qué ocurre si esto no se cumple?

Planificación en tiempo real

- Consideraciones:
  - Los procesos se consideran periódicos.
  - Tiempos:
    - t de procesamiento fijo.
    - d de plazo antes de ser servido.
    - p de período.
  - Se cumple la siguiente relación:
    - $0 \leq t \leq d \leq p$.
- Esto permite implementar un algoritmo de control de admisión.

Planificación en tiempo real



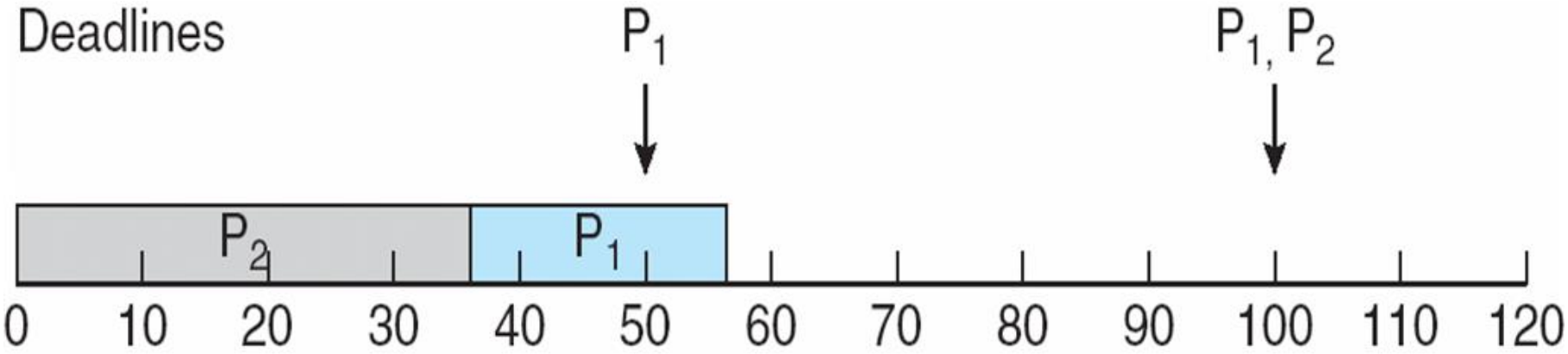
Rate-monotonic scheduling

- Prioridad estática con apropiación.
- Prioridad se asigna en función inversa al período.
  - ¿Quién tiene mayor prioridad?
- Se supone que el tiempo de procesamiento es el mismo en cada ráfaga.

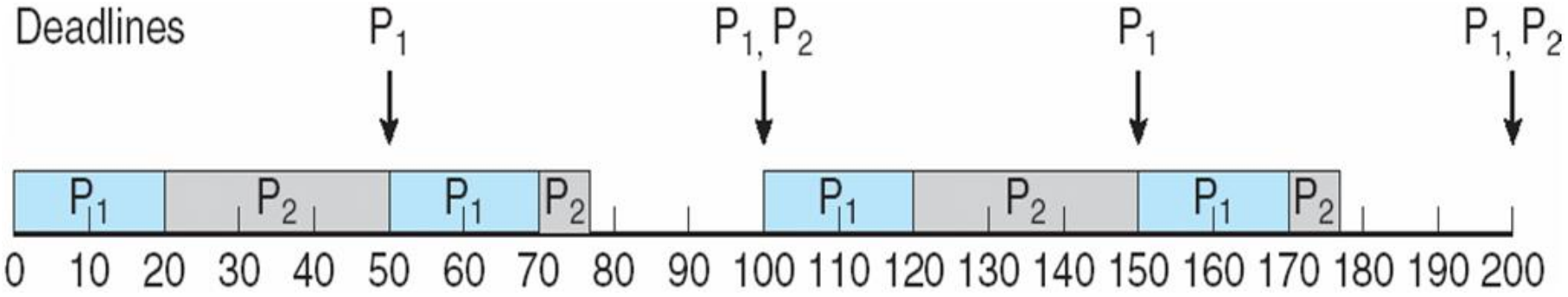
Rate-monotonic scheduling

- Ejemplo:
  - Dos procesos p1 y p2.
  - Los períodos para p1 y p2 son: 50 y 100.
  - Los tiempos de procesamiento son: 20 y 35.
  - El plazo para cada proceso requiere que el proceso complete su ráfaga de CPU antes de que comience el siguiente período.

Rate-monotonic scheduling



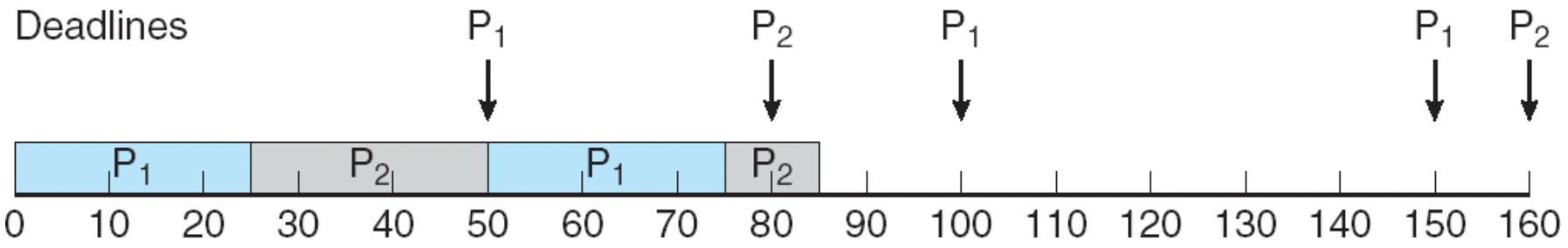
Rate-monotonic scheduling



Rate-monotonic scheduling

- Ejemplo:
  - Dos procesos p1 y p2.
  - Los períodos para p1 y p2 son: 50 y 80.
  - Los tiempos de procesamiento son: 25 y 35.
  - El plazo para cada proceso requiere que el proceso complete su ráfaga de CPU antes de que comience el siguiente período.

Rate-monotonic scheduling



Rate-monotonic scheduling

- Limitación.
  - La utilización del CPU esta acotada.
    - $2(2^{1/n}-1)$
  - Consideraciones.

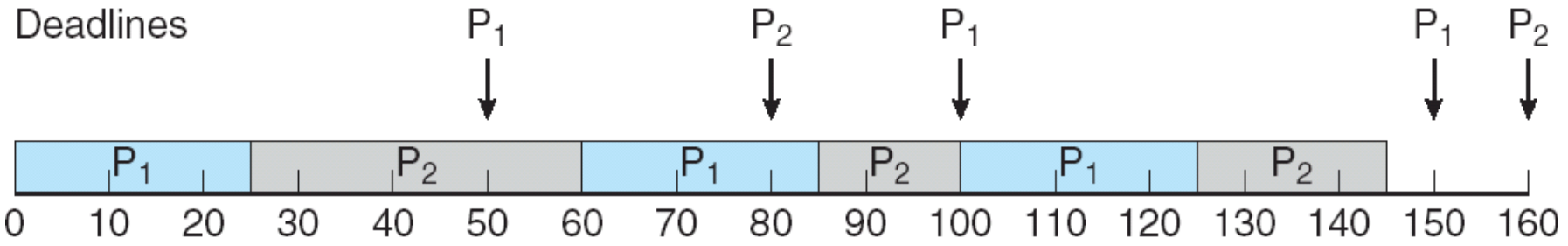
Earliest Deadline First

- Asignación dinámica de prioridad.
  - Finalización de plazos.
- Diferencia con el algoritmo anterior.

Earliest Deadline First

- Ejemplo:
  - Dos procesos p1 y p2.
  - Los períodos para p1 y p2 son: 50 y 80.
  - Los tiempos de procesamiento son: 25 y 35.
  - El plazo para cada proceso requiere que el proceso complete su ráfaga de CPU antes de que comience el siguiente período.

Earliest Deadline First



Earliest Deadline First

- No requiere que los procesos sean periódicos.
- No deben ejecutarse una cantidad constante de tiempo.
- ¿Cuál es su único requisito?
  - Información al planificador.

Planificación con cuota proporcional

- Funciona asignando T cuotas entre todas las aplicaciones.
- Cada aplicación recibe N cuotas de tiempo.
  - Garantizando que la aplicación posee N/T.
- Supongamos tres procesos.
  - A → 50 cuotas.
  - B → 15 cuotas.
  - C → 20 cuotas.

Planificación con cuota proporcional

- Este algoritmo debe trabajar en conjunto con la política de control de admisión.
  - ¿Por qué?
  - Ideas.
  - Ejemplo.