Tema 6

Integración, coste y prestaciones

Indice

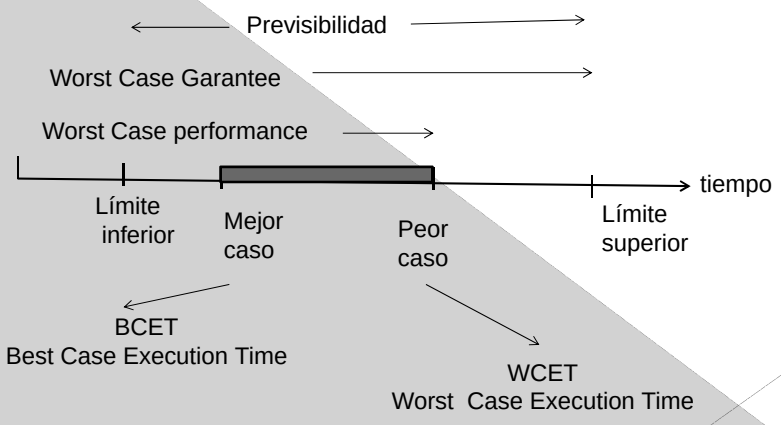
BIBLIOGRAFÍA

Embedded Systems Handbook
Editado por Richard Zurawski
Editorial Taylor & Francis

6.1 Introducción: límites en los tiempos de ejecución

- Importancia de la ejecución en tiempo real
- Restricciones temporales exigentes: Ligaduras hard y soft
- Análisis de planificabilidad: límite superior e inferior de texec de todas las tareas del sistema lo más ajustados posible
- Problema: caches, segmentación y especulación
  - El tiempo de ejecución depende de la historia

6.1 Introducción: límites en los tiempos de ejecución



6.1 Introducción:límites en los tiempos de ejecución

- El tiempo de ejecución de una instrucción está limitado por los siguientes 2 casos:
  - 1.- Límite inferior, la instrucción va sin problemas:
    - Hay acierto de cache,
    - Los operandos están preparados,
    - No hay conflicto de recursos con otras instrucciones
  - 2.- Límite superior, todo va mal:
    - No hay acierto de cache,
    - Los operandos no están preparados,
    - Hay conflicto de recursos con otras instrucciones

Indice

6.2 Herramienta de análisis de tiempos de ejecución

- Podemos distinguir dos partes:
  - **Predicción del comportamiento del procesador**
    - a.- Predicción del comportamiento de cache:
      - capacidad,
      - tamaño de línea,
      - grado de asociatividad,
      - técnica de remplazo
      - Información sobre aciertos/fallos
    - b.- Predicción del comportamiento del pipeline ¿Cuánto tiempo pasa una instrucción en el pipeline?
  - **Análisis de caminos**: computar un límite superior de todos los tiempos de ejecución de todos los posibles caminos del programa. Se suele utilizar programación lineal entera

6.2 Herramienta de análisis de tiempos de ejecución

- El tiempo de ejecución de una instrucción depende de su historia
  - Ej. lazos: primera iteración diferente a las demás
    - ¿Datos e instrucciones en cache?
    - Predicción de saltos
- La precisión mejora si las instrucciones se consideran en su **contexto** de **flujo de control**
- Se usan **bloques básicos**: secuencias de instrucciones en las que el flujo de control entra al principio y sale al final, sin saltos
  - Ej lazos, condiciones, funciones
- Para cada bloque se estudian los accesos a memoria y su efecto en cache, y un análisis de su ejecución en un pipeline determinado. Esto da lugar a una traza

6.3 Análisis de rendimiento de sistemas distribuidos

- Un sistema empotrado distribuido suele estar formado por componentes hw que se comunican a través de una **red de comunicación**
- El rendimiento depende tb de la interacción de las distintas cadenas de datos en el medio de comunicación
- Normalmente los nodos tienen un alto grado de independencia y se comunican a través de paso de mensajes
- Existe una conexión con el entorno físico a través de sensores y actuadores que determinan la velocidad a la que el sistema debe funcionar
  - Evento de llegada: marca el comienzo de una ejecución
  - Evento de finalización: marca final de la ejecución

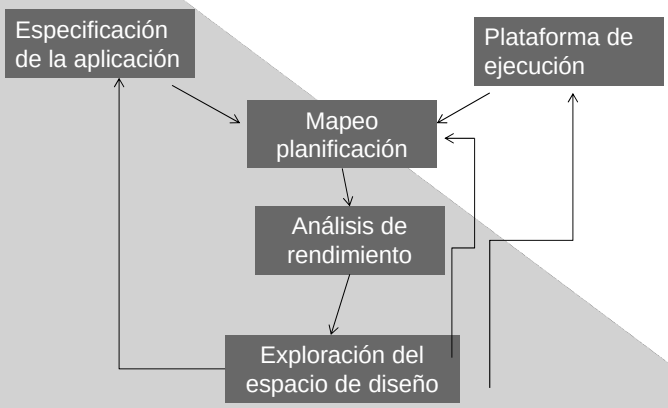
6.3 Análisis de rendimiento de sistemas distribuidos

- El WCET y el BCET son el máximo y mínimo intervalo de tiempo entre los eventos de llegada y finalización para todos los sistemas admisibles y estados del entorno.
- Sólo aquellos que cumplan las ligaduras de límite superior e inferior pueden considerarse
- Medidas estadísticas: se puede tener una caracterización estadística del comportamiento del sistema

6.3 Análisis de rendimiento de sistemas distribuidos

- Reglas durante el proceso de diseño: lo que se pretende es estimar las características esenciales de la implementación final lo antes posible
  - ¿Qué funciones deben implementarse en hw y cuáles en sw?
  - ¿Qué componentes hw deben elegirse?
  - ¿Se cumplen los requerimientos temporales?
  - ¿Qué bus o procesador actúa como cuello de botella?

6.3 Análisis de rendimiento de sistemas distribuidos



6.3 Análisis de rendimiento de sistemas distribuidos

- Requerimientos del análisis de rendimiento:
  - Fiabilidad
  - Precisión
  - Dentro del proceso de diseño
  - Tiempo de análisis corto

6.3 Análisis de rendimiento de sistemas distribuidos

- Métodos
  - a.- Basados en simulación
  - b.- Planificación integral
  - c.- Composición

6.3 Análisis de rendimiento de sistemas distribuidos

- **A.- Métodos basados en simulación**
  - Hay que considerar muchas interacciones dinámicas simultáneas
  - Se debe poder ajustar el nivel de abstracción según el nivel de precisión requerido
  - Importante:
    - Concepto de tiempo
    - Plataforma de ejecución
    - Procesos de comunicación
    - Políticas de compartición de recursos
    - Conjunto apropiado de estímulos que deben cubrir todos los casos posibles
  - Suelen ser útiles para estimar el caso de rendimiento medio, pero no para el caso peor

6.3 Análisis de rendimiento de sistemas distribuidos

- **B.- Métodos basados en planificación integral**:
- Se llama integral porque planifica las comunicaciones como una computación más.
- Existen herramientas comerciales
  - El sistema de comunicaciones se trata como los nodos de computación,
  - Permite mezclar sistemas disparados por eventos con otros por tiempo.
  - El procesamiento y las comunicaciones se dirigen por la ocurrencia de eventos y el paso del tiempo

6.3 Análisis de rendimiento de sistemas distribuidos

- Métodos basados en composición
  - Hay 3 problemas asociados a los sistemas empotrados distribuidos complejos:
    - La arquitectura es muy heterogénea
    - Las aplicaciones suelen tener un alto grado de concurrencia
    - Se producen eventos de diferentes tipos
  - Para un conjunto de políticas de planificación y arbitraje, y para un conjunto de patrones de llegada (periódicos, esporádicos, por ráfaga, etc.) se estiman los WCET y BCET

6.3 Análisis de rendimiento de sistemas distribuidos

- La abstracción de una tarea consiste en
  - Un conjunto de patrones de llegada (o eventos de disparo)
  - Estimación de WCET y BCET
- Una aplicación es una concatenación de varias tareas
- Fundamental: los patrones de llegada deben encajar en unos modelos básicos que permitan calcular los tiempos de respuesta

Indice

6.4 Consumo en Sistemas Empotrados

- Influencia de la tecnología en:
  - Consumo
  - Empaquetamiento
  - Ventilación
  - Coste
  - Seguridad
  - Tiempo de vida de la batería
- Disipación de potencia
  - **Dinámica**: por cambios de estado
    $P \sim C * V_{DD}^2 * f * r$ r: fracción de transistores que conmutan
  - **Estática**: entre conmutaciones del circuito debida a las fugas sub-umbrales. Aumenta con la tecnología
    $P \sim V_{DD} * N_{tran} * K_{design} * I_{leak}$

6.4 Consumo en Sistemas Empotrados

- Se necesita un diseño y ubicación adecuado de los distintos recursos del sistema para conseguir la máxima eficiencia
- Para ello son necesarias políticas de gestión del consumo dinámico y estático (p.e. apagar parte de los recursos cuando no se utilicen)
- Distintos niveles de optimización dependiendo de las condiciones del entorno (con/sin batería)
- Decisiones importantes.
  - Particionamiento hw/sw
  - Subsistema de memoria
  - Estructura de las comunicaciones

6.4 Consumo en Sistemas Empotrados

6.4.1 Modelo de consumo y energía

- Para poder realizar estimaciones a nivel de sistema son necesarios modelos
- Gran cantidad de información que dispara el tiempo de exploración
- Importante la fidelidad
  - A.- Modelo a nivel de función e instrucción
  - B.- Modelo de micro arquitectura
  - C.- Modelo de memoria y bus
  - D.- Modelo de batería

6.4 Consumo en Sistemas Empotrados

6.4.1 Modelo de consumo y energía

- A.- Modelo a nivel de función e instrucción
  - ¿Cuánto consume una determinada instrucción o tipo de instrucción?
  - Mejores resultados cuando se tiene datos de bloques de instrucciones o funciones
- B.- Modelo de micro arquitectura
  - Simuladores ciclo-ciclo para cada procesador con parámetros configurables sobre jerarquía de memoria
  - Un 40%-45% de la potencia consumida de un procesador es el reloj global
  - Importante la disipación estática
- C.- Modelo de memoria y bus
  - Memorias regulares-> estimaciones precisas
  - Distintas estimaciones para distintas configuraciones de la jerarquía de memoria
  - Buses: $P=C*V_{DD}^2*f*bus$
    - Fbus=f(palabras/s)*(transistores/palabra) Se obtiene por simulación
- D.- Modelo de batería:
  - La capacidad de una batería es una función no lineal de la corriente que se toma de ella $C=K/I^a$
  - Un métrica importante es la de energía/retardo hay una pérdida de rendimiento por ganancia en consumo

Indice

6.4 Consumo en Sistemas Empotrados

6.4.2 Optimizaciones a nivel de aplicación/sistema

- Durante el diseño hay que explorar el espacio de diseño para obtener una relación consumo/tvida_batería/rendimiento
  - Ej. Sistema con batería+celdas solares Realizar la mayor parte del trabajo durante el día
  - Estados Active-Standby
- Soluciones:
  - A.- Escalado de frecuencia y voltaje
  - B.- Escalado dinámico de los recursos
  - C.- Selección del core del procesador
  - D.- Selección del subsistema de memoria

6.4 Consumo en Sistemas Empotrados

6.4.2 Optimizaciones a nivel de aplicación/sistema

- **A.- Escalado de frecuencia y voltaje**

$P \sim C * V_{DD}^2 * f * r$ $\quad td \sim V_{DD}/(V_{DD}-Vt)^2$

- Si se decrementa mucho el voltaje hay que bajar la frecuencia
- Soluciones:
  - escalado dinámico de voltaje y frecuencia (20us)
  - Predicción, si hay o no batería

- **B.- Escalado dinámico de los recursos**
  - Inhabilitación total o parcial de componentes

6.4 Consumo en Sistemas Empotrados

6.4.2 Optimizaciones a nivel de aplicación/sistema

- **C.- Selección del core del procesador**
- "High performance"=mucho consumo
- Mejor ASIPs y DSPs, VLIW o EPIC
- Usar coprocesadores dedicados

- D.- Selección del subsistema de memoria
  - Grandes caches- consumo >40%
  - Importante explotar al máximo la localidad (exploración del espacio de diseño)
  - Peores soluciones: grandes caches muy asociativas

6.4 Consumo en Sistemas Empotrados

6.4.2 Optimizaciones a nivel de aplicación/sistema

- Soluciones
  - Esquemas de particionamiento vertical y horizontal
    - Bufferes adicionales, *precoded instruction* buffer, *loop buffer*
    - 2 caches de nivel 1
  - Escalado dinámico
    - *Cache declive*
  - Memorias controladas por Sw (*Scratch –Pad memories*)
  - Mejora de los patrones de acceso a memoria off-chip (*prefeching sw*)
  - Compresión de código
  - Optimización del interconexionado
    - *Bus splitting*
    - *Bus invert coding*

Indice

6.5 Seguridad

- Especialmente importante en SE
- Parámetros de seguridad:
  - A.- Confidencialidad
  - B.- Integridad
  - C:.- No repudiación
  - D.- Disponibilidad
  - E.- Autentificación
  - F.- Control de Acceso
- Niveles de implementación de seguridad
  - Físico
  - Hw
  - Sw fallos del sw
  - Red
- Tipos de fallos:
  - Estáticos
  - Programables

6.5 Seguridad

6.5.2 Restricciones de seguridad

- Energía: especialmente en sistemas basados en batería
  - criptografía
- Capacidad de procesamiento:
  - Flexibles, adaptables
    coste

6.5 Seguridad

6.5.3 Diseño de Sistemas Empotrados seguros

- A.- A nivel de diseño
  - Resistencia a manipulación
  - Protección de memoria
  - Protección IP
  - Comunicaciones
- B.- A nivel de aplicación
  - Identificación de usuario y control de acceso
  - Protección de aplicaciones IP
  - Protección frente a virus y código malicioso

6.5 Seguridad

6.5.4 Criptografía en Sistemas Empotrados

- Es la solución a muchos problemas de seguridad
- Problemas
  - Consume mucho
  - Requiere muchos recursos
  - *Side-channel attacs*
- Soluciones:
  - Instrucciones que consuman lo mismo y tarden lo mismo
  - Protección hw
  - Protección sw
  - Protección matemática