TEMA 4:

ARQUITECTURAS CON PARALELISMO A NIVEL DE INSTRUCCIÓN (ILP)

Lección 1: MICROARQUITECTURAS ILP. CAUCES SUPERESCALARES.

Arquitecturas con DLP (Data Level Parallelism): Ejecutan las operaciones de una instrucción concurrentemente o en paralelo. Unidades funcionales, vectoriales o SIMD.

Arquitecturas con ILP (Instruction Level Parallelism): Ejecutan múltiples instrucciones concurrentemente o en paralelo. Cores escalares segmentados, superescalares o VLIW/EPIC.

Introducción: Definición y nota histórica.

Un procesador superescalar es un procesador segmentado que puede finalizar más de una instrucción por ciclo y que posee recursos hardware para extraer el paralelismo.

Los procesadores superescalares y los VLIW disponen de un cauce en el que cada etapa puede procesar más de una instrucción simultáneamente.

La diferencia entre un procesador superescalar y un VLIW radica en si las técnicas que permiten el procesamiento simultáneo de instrucciones en cada etapa se implementan en hardware (superescalares) o descansan en la capacidad del compilador (VLIW).

Una de las formas de aumentar las prestaciones de los procesadores segmentados consiste en utilizar varias unidades funcionales que puedan trabajar simultáneamente con instrucciones distintas. En cualquier caso, añadir varias unidades funcionales, si bien puede mejorar el rendimiento de un procesador segmentado, no permite sobrepasar el límite máximo de una instrucción terminada por cada ciclo, dado que la última etapa del cauce (WB) sólo puede procesar una instrucción por ciclo. La idea que subyace en los procesadores superescalares y VLIW es la de conseguir que puedan procesarse simultáneamente varias instrucciones en todas las etapas. Esta situación da lugar a importantes problemas relacionados con la forma en que se gestionan las dependencias de datos y de control.

Una primera forma de mejorar el número de instrucciones procesadas por ciclo consiste en organizar el procesador en varios cauces que funcionen en paralelo y de forma casi independiente. En algunos casos, cada uno de los cauces está especializado en un tipo de instrucción, por lo que la disposición de las instrucciones en el código es muy importante desde el punto de vista del aprovechamiento de los recursos. Usualmente es el compilador el que debe ser capaz de aplicar estas reglas de forma adecuada para conseguir códigos optimizados en este tipo de procesadores, de forma análoga a lo que ocurre en los procesadores segmentados, y también en los procesadores VLIW.

Aunque, la forma más usual de organizar la microarquitectura es que cada una de las etapas sea capaz de procesar varias instrucciones por ciclo. Entre etapa y etapa existen diferentes estructuras de almacenamiento que guardan la información obtenida para cada instrucción procesada por la etapa anterior, hasta que les toca el turno de ser procesadas en la etapa siguiente.

ILP (Instruction Level Parallelism):

- Procesadores/cores segmentados (1961 (IBM 7030), 1982 (chip Intel i286, Motorola 68020) …
- Procesadores con múltiples unidades funcionales (1967 (IBM 360/91)).
- Procesadores/cores superescalares (1989 (chip Intel 960CA(3)), 1990 (chip IBM Power1 (4)), 1992 (HP PA 7100(2/2)), etc).
- Procesadores/cores VLIW (1990 (chip DSP Intel i860(2)), 1997(chip DSP TMS320C6x(8)), 2001(chip Intel Itanium) …

Paralelismo entre instrucciones (ILP). Orden en Emisión y Finalización.

Ordenaciones en una secuencia de instrucciones.

- En una secuencia de instrucciones se pueden distinguir tres tipos de ordenaciones:
  - El orden en que se captan las instrucciones (el orden de las instrucciones en el código).
  - El orden en que las instrucciones se ejecutan.
  - El orden en que las instrucciones cambian los registros y la memoria.

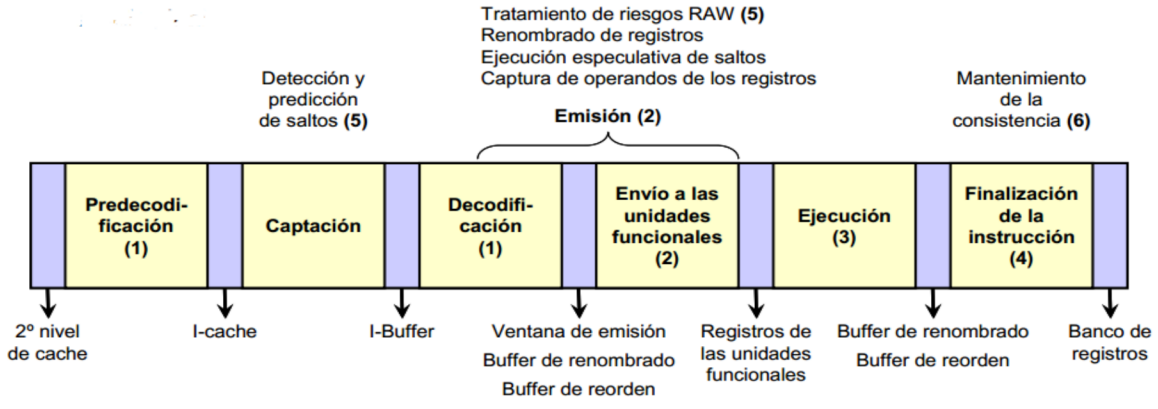
- El procesador superescalar debe ser capaz de identificar el paralelismo entre instrucciones (ILP) que exista en el programa y organizar la captación, decodificación y ejecución de instrucciones en paralelo, utilizando eficazmente los recursos existentes (el paralelismo de la máquina).

- Cuanto más sofisticado sea un procesador superescalar, menos tiene que ajustarse a la ordenación de las instrucciones según se captan, para la ejecución y modificación de los registros, de cara a mejorar los tiempos de ejecución. La única restricción es que el resultado del programa sea correcto.

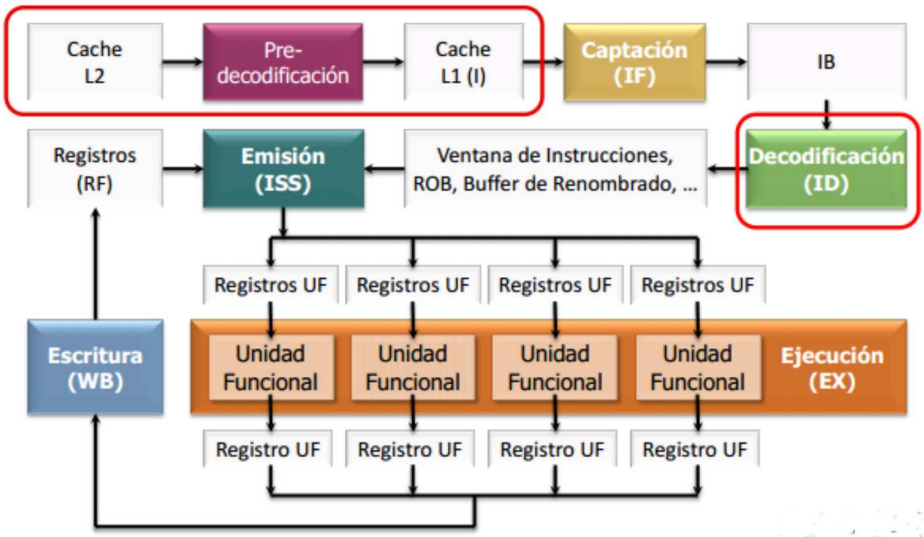
Cauces superescalares.

Decodificación Paralela y predecodificación.

- Aspectos del Procesamiento Superescalar y Etapas del Cauce:
  1. Decodificación Paralela (Decodificación a mayor velocidad -> Predecodificación).
  2. Emisión Paralela de Instrucciones a las Unidades Funcionales (Dependencias).
  3. Ejecución Paralela en las distintas Unidades Funcionales, UF, (segmentadas).
  4. Finalización del Procesamiento de la Instrucción.
  5. Detección y predicción de saltos.
  6. Mantenimiento de la consistencia secuencial (desacoplar la ejecución y la escritura de resultados).



- Etapas de un Procesador Superescalar: Predecodificación.



  - Esta imagen proporciona un esquema con las principales etapas de un procesador superescalar, junto con los recursos más relevantes que se utilizan para gestionar las dependencias entre instrucciones.
  - Todas las etapas de un procesador superescalar pueden procesar varias instrucciones por ciclo.
  - El procesador incluye una serie de elementos como ventanas de instrucciones o estaciones de buffers, buffers de renombramiento, buffers de reordenamiento, etc.

Predecodificación.

- En muchos casos, una de las etapas de decodificación se implementa entre la caché de segundo nivel y la caché de instrucciones de primer nivel. Esto es lo que se conoce como etapa de predecodificación.

- Esta etapa se suele encargar de determinar el tipo de instrucción y, con ello, facilita la identificación posterior de los recursos que se van a necesitar.
  - Instrucciones con salto: si al captar la instrucción se puede determinar de forma inmediata que, efectivamente, se trata de una instrucción de salto, su procesamiento se puede empezar antes incluso de que pase por la etapa de decodificación.
  - Si en la predecodificación se obtiene información de la unidad funcional que va a utilizarse en el procesamiento de cada instrucción, se favorece una emisión más rápida de la instrucción a los cauces de enteros o de unidades de coma flotante.
  - Si se determina si una instrucción va a hacer una referencia a memoria, también se podría avanzar su procesamiento una vez captada.

- Esta etapa añade algunos bits a las instrucciones cuando pasan desde la memoria principal o desde niveles de caché inferiores a la memoria caché de instrucciones en el primer nivel. Con ello consigue acelerar el proceso de decodificación completo de las instrucciones en la etapa posterior de decodificación, o incluso que algunas instrucciones complejas (como las de salto) puedan empezar a procesarse ya desde su captación.

Captación (IF):

- Toma varias instrucciones de la memoria caché de instrucciones y las pasa a la cola de instrucciones. Las instrucciones se captan según el orden en que se encuentran en la memoria (que es el orden en el que están en el código), y se introducen en ese mismo orden en la cola de instrucciones, desde donde se transfieren a la unidad de decodificación (ID) para decodificarse en el mismo orden.

Decodificación (ID):

- Procesador superescalar: Esta etapa debe ser capaz de decodificar varias instrucciones por ciclo.
- Procesador segmentado: Decodificar la instrucción correspondiente y acceder a los operandos de la instrucción.

- En un procesador superescalar es imposible comprobar al mismo tiempo las dependencias entre los operandos de las instrucciones decodificadas y las instrucciones que ya se están ejecutando.

- Por otra parte, habrá que determinar a qué unidades funcionales se envía cada instrucción en el momento en que esa unidad esté libre y tenga sus operandos disponibles.

- Una vez captadas las instrucciones, se almacenan en una cola (cola de instrucciones) en el mismo orden en el que se han captado. La etapa de decodificación toma varias instrucciones por ciclo según el orden en el que se encuentran en la cola de instrucciones y, tras ser decodificadas se escriben en una serie de estructuras. Una de esas estructuras es la ventana de instrucciones y la otra las estaciones de reserva.

- Desde la ventana de instrucciones es donde se emiten a las unidades funcionales que las ejecutan, cuando tienen sus operandos disponibles y la unidad funcional está libre.

- En el caso de disponer varias estaciones de reserva, esta etapa también debe encargarse de seleccionar la estación de reserva que almacenará la instrucción hasta que pueda comenzar su ejecución.

Emisión Paralela de Instrucciones. Estaciones de Reserva & Ventana de instrucciones.

- Etapas de un Procesador Superescalar: Emisión Paralela de Instrucciones.
- Ventana de instrucciones:
    - Esta estructura se implementa a través de una cola de registros donde se almacenan las instrucciones que han sido decodificadas, y que están a la espera de ser emitidas a las unidades funcionales donde se ejecuta la operación que codifican. La etapa de emisión se encarga de determinar que instrucciones pueden emitirse al disponer de sus operandos y existir unidades funcionales libres para su ejecución.
    - Las instrucciones se cargan en la ventana una vez decodificadas.
        - Se utiliza un bit para indicar si un operando está disponible (se almacena el valor o se indica el registro desde donde se lee) o no (se almacena la unidad funcional desde donde llegará el operando).
    - Puede ser centralizada (que se conoce como ventana de instrucciones) o distribuida (conocido como estaciones de reserva).

- Existen dos tipos de emisión según el alineamiento:
    - Emisión alineada: La emisión es alineada si no pueden introducirse nuevas instrucciones en la ventana de instrucciones hasta que ésta no esté totalmente vacía, es decir, hasta que no se hayan emitido todas las instrucciones.
    - Emisión no alineada: En la emisión no alineada mientras que exista espacio en la ventana, se pueden ir introduciendo instrucciones para ser emitidas.

- Existen dos tipos de emisión según el orden:
    - Emisión ordenada: Se respeta el orden en que las instrucciones se han ido introduciendo en la ventana de instrucciones. Este orden es el mismo en que las instrucciones se han ido decodificando, y coincide con el orden de las instrucciones en el programa. Básicamente, primero que llega primero que se ejecuta.
        - En la emisión ordenada existe bloqueo entre instrucciones: si las instrucciones que aparecen primero en el orden del programa no pueden emitirse bloquean la emisión de las instrucciones que las siguen.
        - Inconveniente: Dependencias RAW.

    - Emisión desordenada: En este caso, no hay bloqueo, ya que pueden emitirse todas las instrucciones que dispongan sus operandos (si no disponen de los operandos, se bloquea) y de la correspondiente unidad funcional donde se ejecutarán. Básicamente, si una instrucción se bloquea por una dependencia, se busca otra para emitirla mientras que en la ordenada todas las que están después de la bloqueada se esperan.
        - Inconveniente: Aparecen WAW y WAR.

- Estaciones de Reserva.
    - Las microarquitecturas de los procesadores superescalares han tendido a distribuir la ventana de instrucciones a través de las denominadas estaciones de reserva.
    - De esta forma, se dispone de espacio suficiente para almacenar las instrucciones mientras esperan a que se pueda iniciar su ejecución, pero distribuyendo este almacenamiento en estructuras más pequeñas, con menos complejidad hardware y con un acceso más rápido.
    - Así, en lugar de disponer de una única ventana de instrucciones, donde, tal y como se ha visto, se introducen las instrucciones tras ser decodificadas y desde donde se emiten a las distintas unidades funcionales, existe una estructura similar a la ventana, pero específica para cada unidad funcional o conjunto de unidades funcionales.
    - En cada una de ellas solo se introducen las instrucciones que se ejecutarían en algunas de las unidades funcionales del procesador.
    - En una microarquitectura con estaciones de reserva, las funciones de la etapa de emisión se desdoblan.
        - En primer lugar, las instrucciones decodificadas deben pasarse a la estación de reserva adecuada (desde la que se pueda acceder a la unidad funcional correspondiente).
        - En esa estación de reserva es donde cada instrucción debe esperar a que le toque el turno de pasar a la unidad funcional para ejecutarse, una vez que dispone de sus operandos.
        - La primera parte se suele seguir denominando emisión (ISS), y se implementa en la misma etapa de decodificación, que pasa a denominarse etapa de decodificación/emisión (ID/ISS).
        - La determinación de qué instrucciones de la estación de reserva tiene sus operandos disponibles y pasa a ejecutarse, se denomina envío.

- Las estaciones de reserva como hemos dicho antes pueden ser para una unidad funcional o para un conjunto (estación de reserva compartida).
  - La estación de reserva compartida afecta a la complejidad de la arquitectura y a la eficiencia del paralelismo entre instrucciones. Si se dispone de más estaciones, éstas pueden llevar menos líneas ya que habría que almacenar instrucciones para un número menor de unidades funcionales, y su gestión necesita un hardware menos costoso. Puede ocurrir que la estructura de buses sea más compleja.
  - Si cada unidad funcional tiene su estación de reservas, no hace falta guardar información del tipo de operación a realizar en cada una de las líneas de las estaciones y se ahorran algunos bits en cada línea de la estación de reservas. No obstante, incrementa la complejidad de las estructuras de buses que comunican las distintas estaciones con la salida de las restantes unidades funcionales, el banco de registros u otras estructuras de almacenamiento.
  - En cualquier caso, para encontrar la solución óptima hay que determinar el número de líneas más adecuado para cada estación de reserva y esa distribución determina el ritmo al que pasan instrucciones a cada una de las estaciones de reserva.

- Alternativas para el Envío a las Unidades Funcionales.
  - Reglas de Selección: Se determina las instrucciones que pueden enviarse (las instrucciones ejecutables). Son ejecutables todas aquellas instrucciones que tengan sus operandos disponibles.

  - Reglas de Arbitraje: Se selecciona la instrucción a enviar, o las que se envían, en el caso de que haya varias unidades funcionales que comparten la estación de reserva y pueden aceptar instrucciones en un momento determinado. El orden de envío se realiza teniendo en cuenta la antigüedad de las instrucciones en la estación de reservas. Esto es, se respeta el orden de las instrucciones en el caso de colisión.

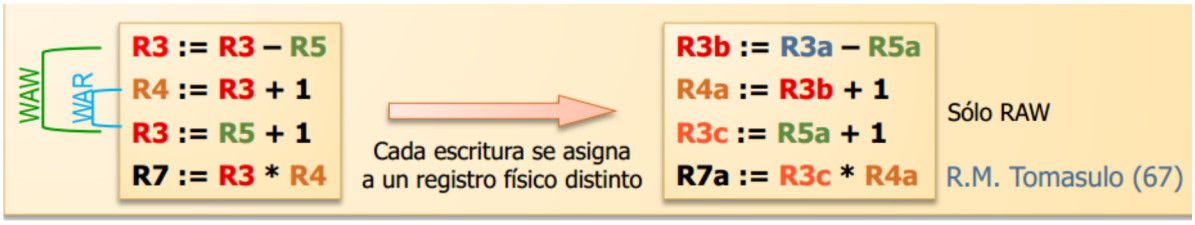
  - Orden de Envío: Ordenadas, Desordenadas, o Parcialmente ordenadas (ciertas instrucciones no ejecutables bloquean instrucciones de un tipo, pero no de otros).
    - Ordenado: las instrucciones no ejecutables bloquean a las ejecutables que están después de ellas en la estación de reservas.
    - Parcialmente ordenado: Se produce bloqueo solo entre determinados tipos de instrucciones.
    - Desordenado: no existe bloqueo entre instrucciones.

  - Velocidad de Envío: Número de instrucciones que se envían por ciclo.
    - Una por ciclo: Si una estación de reserva solo envía instrucciones a una única unidad funcional, puede enviar una instrucción por ciclo como máximo.
    - Varias por ciclo: Si la estación de reserva es compartida por varias unidades funcionales podría diseñarse para enviar varias instrucciones por ciclo.

Renombramiento de registros.

- Técnica para evitar el efecto de las dependencias WAR, o antidependencias (en la emisión desordenada) y WAW, o dependencias de salida (en la ejecución desordenada).
    - Riesgo WAR y WAW: Se pueden evitar si se dispone de un espacio de almacenamiento alternativo en el que se realiza la escritura final.

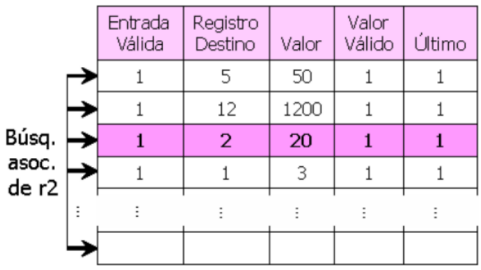
    - Riesgo RAW: La lectura debe esperar a que termine la escritura sobre el elemento de almacenamiento. Se trata de un riesgo asociado a una dependencia real entre las operaciones de un programa: para que empiece la operación que necesita el operando que se lee, debe terminar la operación que genera dicho operando.



- Implementación Estática: Durante la compilación.

- Implementación Dinámica: Durante la ejecución del programa utilizando registros y hardware de control extra.
    - En el caso del renombramiento dinámico, las prestaciones del procesador se verán afectadas por la velocidad del renombrado.
    - Velocidad del Renombrado: máximo número de asignaciones de registros por ciclo que admite el procesador.
    - La velocidad de renombramiento y el coste del mismo vendrán determinadas, a su vez, por características como el número y tipo de buffers que se utilizan como registros temporales y por los mecanismos para acceder a los mismos.

- Buffers de Renombramiento: Permite varias escrituras pendientes a un mismo registro.
  El bit de 'último' se utiliza para marcar la escritura más reciente.



Entrada Válida	Registro Destino	Valor	Valor Válido	Último
1	5	50	1	1
1	12	1200	1	1
1	**2**	**20**	**1**	**1**
1	1	3	1	1
⋮	⋮	⋮	⋮	⋮

- Características de los Buffers de Renombrado:
    - Tipos de Buffers: separados o mezclados con los registros de la arquitectura.
    - Número de Buffers de Renombrado.
    - Mecanismos para acceder a los Buffers (asociativos o indexados).

- Algoritmo de Tomasulo.