Arquitectura de Computadoras

PROCESADORES SÚPER-ESCALARES

PROF. JORGE RUNCO – CURSO 2021

1

Mejora en las prestaciones

En clases anteriores vimos las distintas técnicas para resolver los posibles atascos y llegar al límite de tener una instrucción por ciclo.

¿Podemos ejecutar más de una instrucción por unidad de tiempo?

Hay dos estrategias:

1) Procesadores súpersegmentados: aprovechando las mejoras tecnológicas en cuanto a la velocidad de los circuitos, se diseña un cauce con un mayor número de etapas de menor duración, con lo que, al reducirse el tiempo de la etapa, el cauce puede funcionar a frecuencias mayores.

2) Procesadores súperescalares: el cauce de estos procesadores puede procesar simultáneamente más de una instrucción por etapa. Esto significa que puede "arrancar" ó emitir varias instrucciones en paralelo.

2

Emisión de instrucciones

Es importante recordar lo que vimos en clases anteriores: los riesgos por dependencia de datos y de control eran quienes limitaban el poder procesar una instrucción por ciclo.

Ahora con procesadores súperescalares, donde están replicadas las unidades funcionales, la "habilidad" de ejecutar varias instrucciones en paralelo puede estar limitada por los riesgos estructurales en el uso de las unidades de ejecución (alu, unidad de pf)

Conclusión: para ejecutar varias instrucciones en paralelo hay que formar grupos de instrucciones sin dependencias ***paquete de emisión.***

Emitir una instrucción quiere decir "pasar" la instrucción a la etapa de ejecución, con procesadores súperescalares son varias instrucciones a la vez.

3

Emisión de instrucciones

Pensando en el momento en el cual se elije el grupo de instrucciones a emitir, existen dos alternativas:

Estáticas: la selección del grupo de instrucciones la realiza el compilador. El compilador selecciona los grupos de instrucciones (paquete de emisión) evitando riesgos de datos y control, así el paquete está formado por instrucciones independientes que pueden ejecutarse a la vez.

Dinámica: aquí el que es responsable de seleccionar el grupo de instrucciones es el procesador. Esta estrategia es la que utilizan las arquitecturas súperescalares.

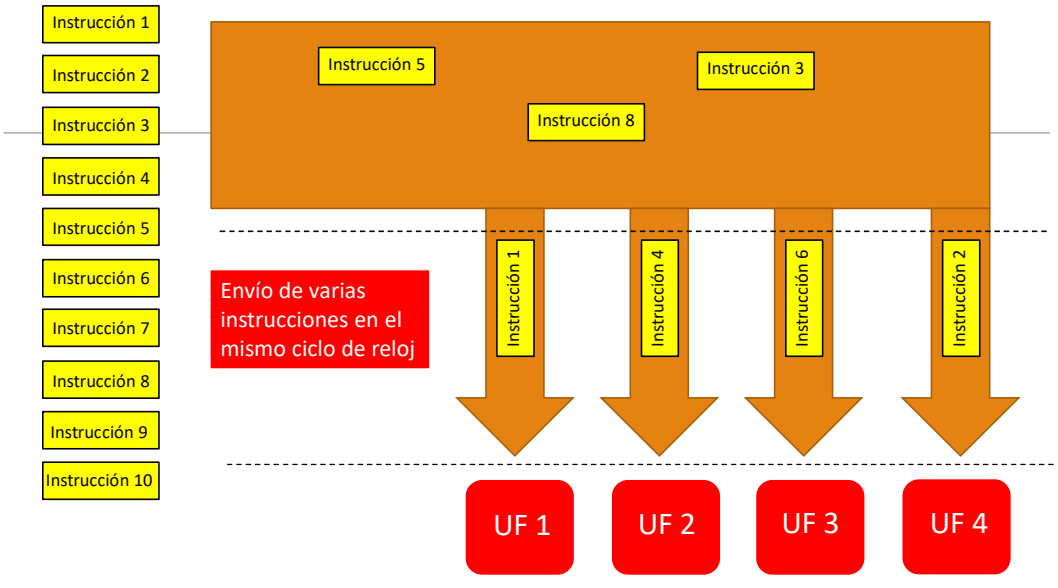
4

Hardware en súper-escalares

El procesador súperescalar no sólo tiene que tener replicadas las unidades de ejecución, también debe ser capaz de captar (etapa fetch) varias instrucciones a la vez en un mismo ciclo. Recordemos que el procesador segmentado convencional (el que vimos en clases anteriores), sólo captaba de a una instrucción por ciclo.

En el procesador segmentado convencional, también puede haber varias unidades funcionales en las que se ejecutan simultáneamente varias instrucciones, pero en este último las instrucciones se envían a ejecución (a su unidad funcional correspondiente) a razón de una por ciclo. Sin embargo, en el procesador súperescalar, en un único ciclo se envían múltiples instrucciones a las unidades funcionales correspondientes.

5

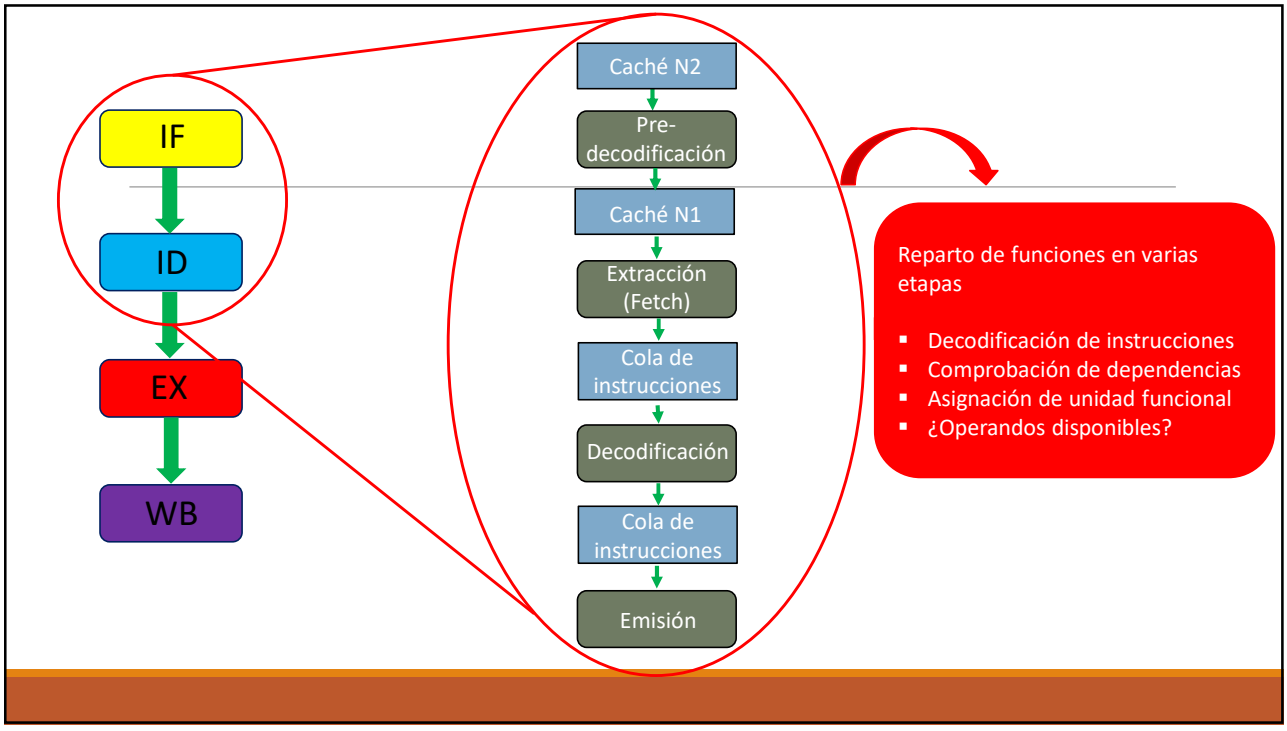


6

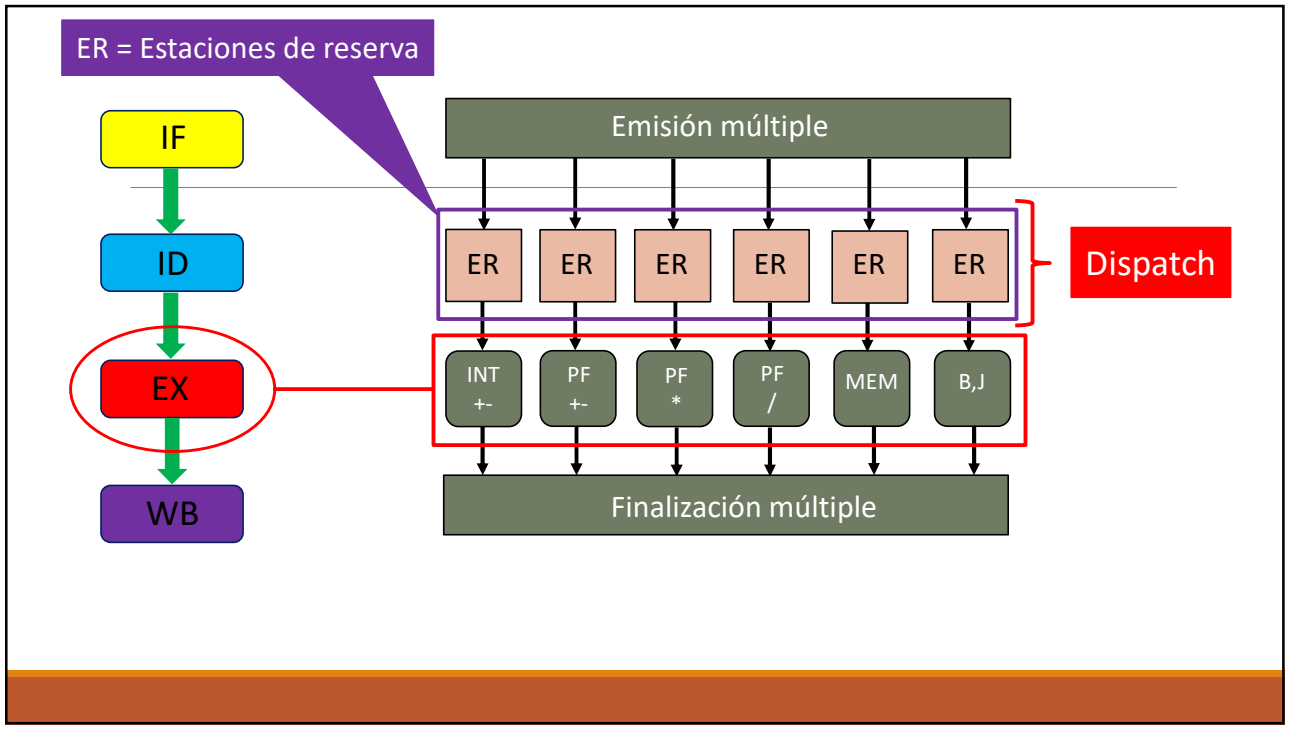
Ciclo de instrucción: súper-escalares

- **CAPTACION (fetch):** múltiples instrucciones son captadas simultáneamente, utilizando técnicas de predicción de saltos y ejecución especulativa.
- **DECODIFICACION (decode):** en dos pasos, i) Predecodificación entre la memoria y el cache para identificación de saltos, y ii) Determinación de la operación, localización de operandos y localización del resultado.
- **VENTANA DE EJECUCION = ENCOLADO (dispatch) y EMISION (issue):** identificación de las instrucciones de la cola que están listas para comenzar su ejecución, o sea que tienen sus dependencias satisfechas.
- **EJECUCION (execute):** en paralelo, en diferentes unidades funcionales.
- **FINALIZACION (commit):** El resultado es confirmado en su destino.

7



8



9

Estados por los que pasa una instrucción

Distribuida (dispatched) cuando ha sido enviada a una estación de reserva asociada a una o varias unidades funcionales del mismo tipo.

Emitida (issued) sale de la estación de reserva hacia una unidad funcional.

Finalizada (finished) cuando abandona la unidad funcional y pasa al buffer de reordenamiento, los registros se encuentran temporalmente en registros no accesibles al programador.

Terminada (completed) o terminada arquitectónicamente, cuando ha realizado la escritura de los resultados desde los registros de renombramiento a los registros arquitectónicos, ya son visibles al programador. Se realiza la actualización del estado del procesador.

Retirada (retired) cuando ha realizado la escritura en memoria, si no se necesita escribir en memoria la finalización de una instrucción coincide con su retirada.

10

Estados por los que pasa una instrucción

Vimos todas las tareas a realizar en la etapa ID de un procesador segmentado.

Para un procesador súperescalar las tareas en esta etapa, en general tareas se reparten entre la ***pre-Decodificación***, la propia ***Decodificación*** y la ***Emisión (Issue).***

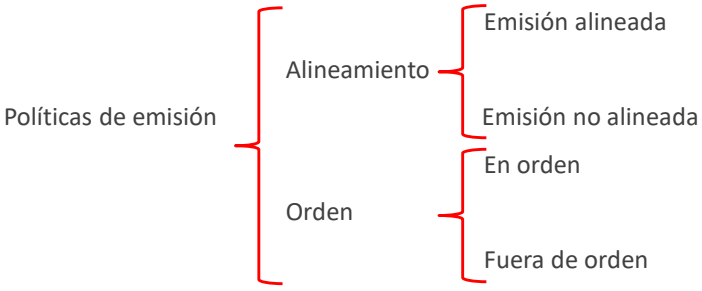
Una vez que las instrucciones son decodificadas, se van dejando en una estructura llamada ***ventana de instrucciones.*** En este punto las instrucciones decodificadas son estructuras más complejas donde se agregó información (bits) que ayudarán a su posterior ejecución.

Para que una instrucción sea emitida, se necesita que estén disponibles tanto sus operandos como la unidad funcional que implementa la operación de la instrucción. La etapa de Emisión se encarga, por tanto, de determinar qué instrucciones pueden emitirse por estar disponibles sus operandos y existir alguna unidad funcional disponible para su ejecución.

11

Políticas de emisión de instrucciones

Puede suceder que en un ciclo determinado, haya más instrucciones disponibles para ser emitidas de las que realmente pueden ser emitidas (porque estén disponibles sus operandos y la correspondiente unidad funcional) ó alguna instrucción no esté "lista" para ser emitida, hay que establecer que ***política de emisión*** se seguirá. Es decir que instrucciones serán seleccionadas para ser emitidas.



12

Políticas de emisión de instrucciones

La ***emisión es alineada*** si no pueden introducirse nuevas instrucciones en la Ventana de Instrucciones hasta que ésta no está totalmente vacía. Es decir, hasta que no se hayan emitido todas las instrucciones que en un ciclo anterior se introdujeron en la Ventana de Instrucciones.

En la ***emisión no alineada***, mientras que exista un espacio libre en la ventana, se pueden ir introduciendo nuevas instrucciones para ser emitidas.

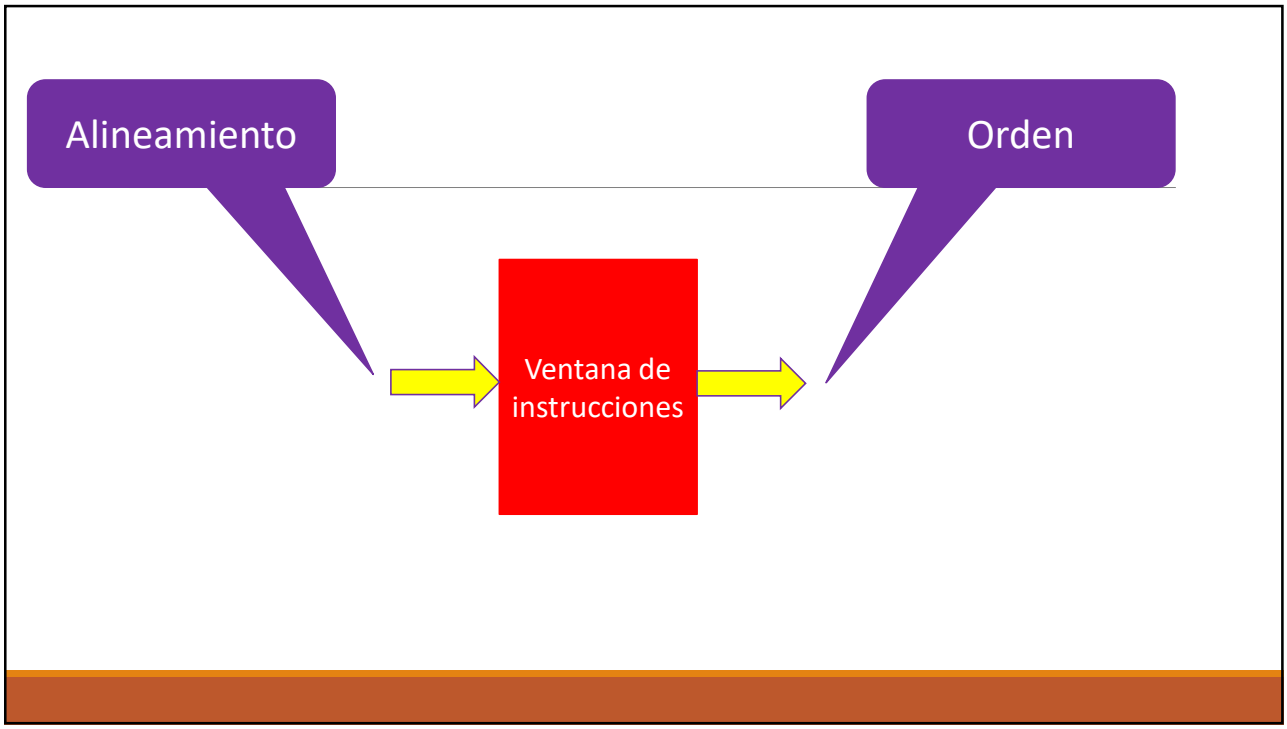
13

Políticas de emisión de instrucciones

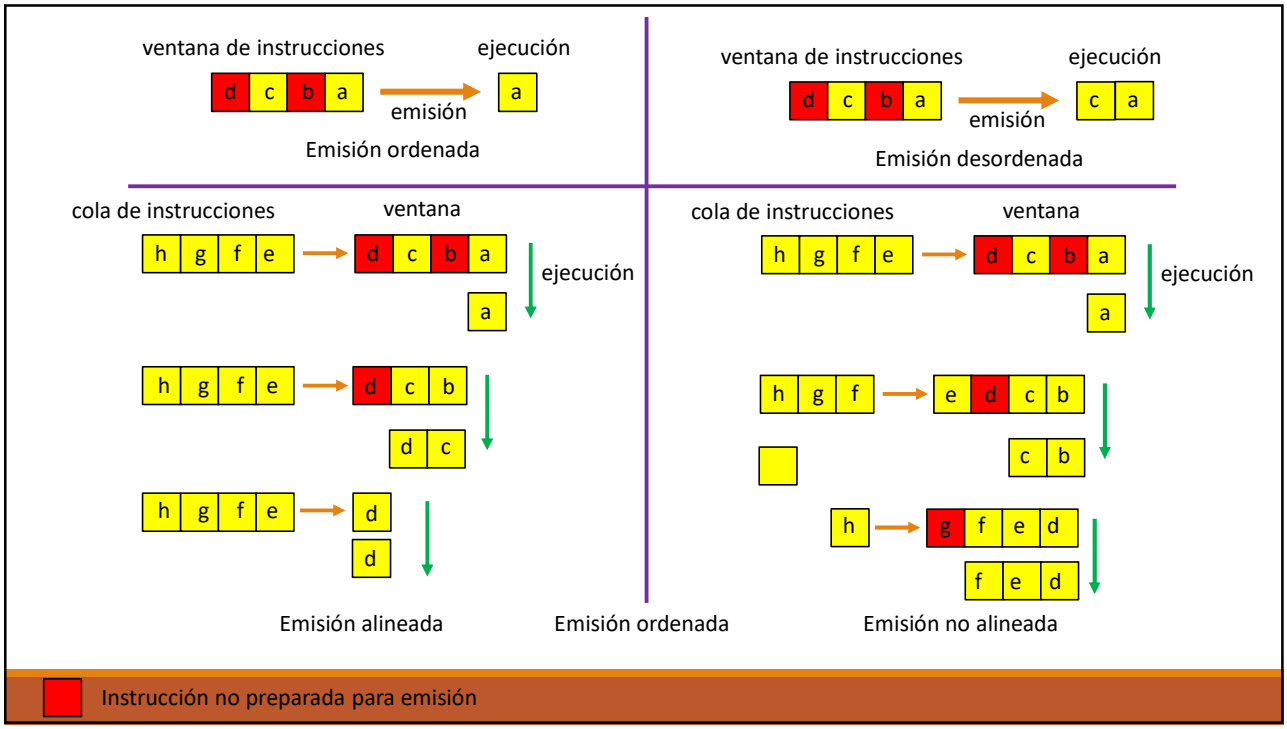
En la **emisión ordenada** se respeta el orden en que las instrucciones se han ido introduciendo en la Ventana de Instrucciones. Este orden es el mismo en que las instrucciones se han ido decodificando, y coincide con el orden de las instrucciones en el programa. De esta forma, si una instrucción de la Ventana de Instrucciones no puede emitirse, las instrucciones que le siguen tampoco podrían emitirse aunque tuvieran sus operandos y la unidad funcional disponibles. Es decir, existe un bloqueo entre instrucciones: si las instrucciones que aparecen primero en el orden de un programa no pueden emitirse, bloquean la emisión de las instrucciones que las siguen.

En el caso de la **emisión desordenada** no existe bloqueo, ya que pueden emitirse todas las instrucciones que dispongan de sus operandos y de una unidad funcional del tipo apropiado.

14


Alineamiento
Orden
Ventana de instrucciones

15


ventana de instrucciones
ejecución
d c b a
emisión
a
Emisión ordenada
ventana de instrucciones
ejecución
d c b a
emisión
c a
Emisión desordenada
cola de instrucciones
ventana
h g f e
d c b a
ejecución
a
h g f e
d c b
d c
h g f e
d
d
Emisión alineada
Emisión ordenada
cola de instrucciones
ventana
h g f e
d c b a
ejecución
a
h g f
e d c b
c b
h
g f e d
f e d
Emisión no alineada
Instrucción no preparada para emisión

16

Técnicas de optimización

Se utilizan básicamente tres técnicas de **hardware** para aumentar las prestaciones:

- Duplicación de recursos
- Política de emisión desordenada de instrucciones con ventana de ejecución
- Renombrado de registros

17

Duplicación de los recursos

No sólo necesito recursos para ejecutar varias instrucciones en paralelo, sino que además necesito captar y decodificar varias instrucciones simultáneamente.

El dispatcher es crucial, ya que hay que mantener ocupadas las unidades funcionales (UF).

Es el único elemento con “inteligencia”. El resto de los componentes “acompaña”.

18

Políticas de ejecución paralela(1)

Se clasifican según dos factores:

- El orden según el cual las instrucciones son enviadas a ejecutar (emitidas, issued).
- El orden en que las instrucciones finalizan su ejecución al escribir en registros o memoria (commit).
- La política más simple es emitir y finalizar en orden secuencial. La ventaja es que el procesador sólo debe preocuparse por las dependencias de datos reales ya que las dependencias de salida y las antidependencias no existen.

19

Políticas de ejecución paralela(2)

- Finalización desordenada: no espera a que terminen las instrucciones anteriores para enviar la actual.
- Aparecen las dependencias de salida (además de las verdaderas), por lo que se necesita una lógica de emisión más complicada. Peligroso ante interrupciones.
- Emisión desordenada (implica la anterior): utilizando la ventana de instrucciones puede irse más allá de un punto de conflicto en la emisión para encontrar instrucciones sin dependencia de datos ni conflictos de recursos (anticipación). Aparecen las antidependencias.

20

Políticas de ejecución paralela(3)

Ventana de ejecución

- Es el conjunto de instrucciones que se consideran para su ejecución en un determinado momento. El hardware determina cuáles son las instrucciones de la ventana que pueden enviarse a ejecutar en paralelo, limitado por las dependencias de datos y disponibilidad de recursos.
- La ventana debe ser lo más grande posible. La limitan la capacidad de captar intrucciones a una gran tasa y el problema de los saltos.
- Se utiliza predicción de saltos y ejecución especulativa. La porción de código predicha se incorpora a la ventana de ejecución y se ejecutan tentativamente. Si la predicción fue correcta el resultado se hace visible y permanente (commit), si no se elimina

21

Renombrar registros

- Cuando se utilizan técnicas de desordenación los valores de los registros no pueden conocerse completamente en cada instante de tiempo. Las instrucciones entran en conflicto por el uso de registros y el procesador debe detener alguna etapa para resolverlo.
- Las técnicas de software de optimización de registros empeoran la situación.
- Los efectos de las dependencias de salida pueden disminuirse por esta técnica, que consiste en disponer de registros adicionales (internos, ocultos al programador) y asignarlos (por hardware) a instrucciones en conflicto. Existen R3a y R3b, por ejemplo.

22