

procede a su inclusión, lo que en caso de que la tabla ya esté llena provocará el reemplazo de una de las entradas existentes.

2.3.4. TÉCNICAS SUPERESCALARES

2.3.4.1. Concepto

En un procesador superescalar las instrucciones usuales (aritmética entera y flotante, captaciones y escrituras de datos, bifurcaciones, etc.) pueden ejecutarse de manera simultánea e independiente unas de otras. Esta posibilidad plantea problemas de difícil solución relacionados con la segmentación de instrucciones.

La aproximación superescalar se ha generalizado con la difusión de las máquinas RISC, aunque actualmente muchos procesadores con caracteres combinados CISC-RISC utilizan técnicas de este tipo. El nombre hace referencia a la mejora de las operaciones efectuadas con magnitudes escalares, que son las utilizadas en la mayoría de las aplicaciones actuales (aunque hay un importante y creciente campo de aplicaciones en donde las magnitudes fundamentalmente usadas son vectoriales).

Tal y como se evidencia en la Tabla 4, las técnicas superescalares utilizan también la segmentación, y además permiten obtener mejores resultados que con las técnicas conocidas como supersegmentadas. Aprovechan estas últimas el hecho de que un cauce se puede dividir en etapas que requieran una fracción de un ciclo de reloj, con lo que se establece un reloj interno que funciona a una velocidad múltiplo del reloj externo (2x en el caso del ejemplo de la Tabla 4).

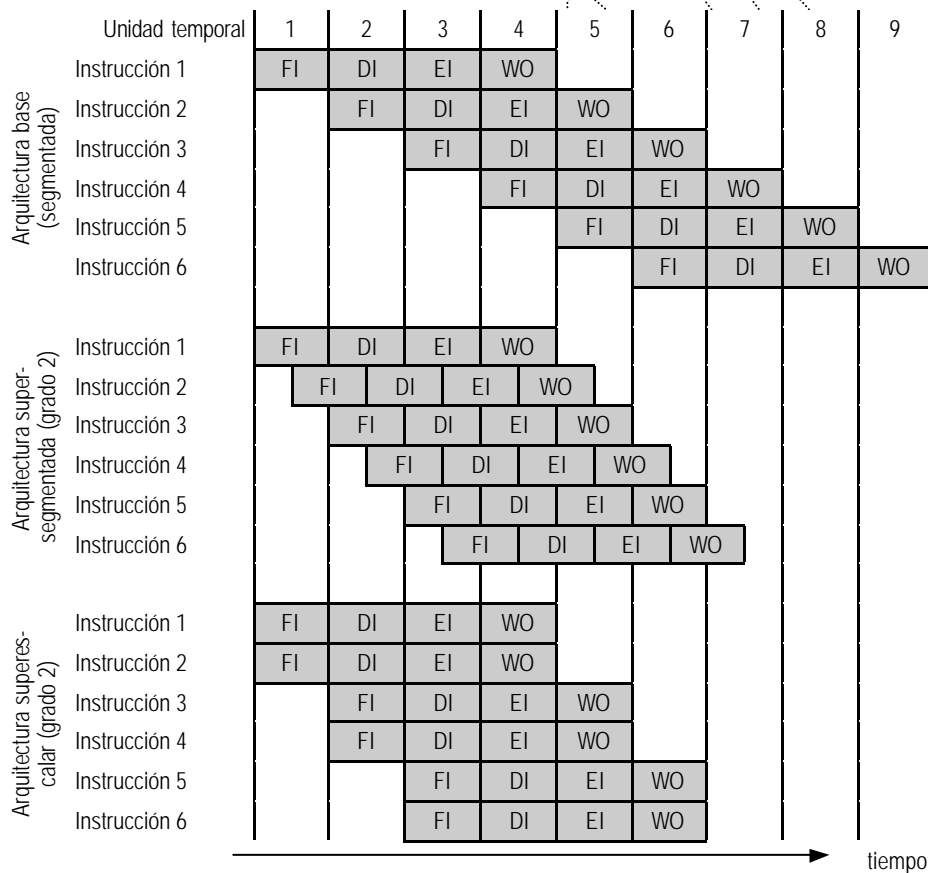


Tabla 4 Comparación de la arquitectura supersegmentada con la superescalar.

2.3.4.2. Restricciones

El éxito, tanto del enfoque segmentado/supersegmentado como del superescalar depende de la habilidad para ejecutar varias instrucciones de forma simultánea. De este modo, se aprovecha el **paralelismo a nivel de instrucciones** (ILP, instruction level parallelism) que a la vez que permite estas arquitecturas actúa como un limitador de su eficacia. Además del ILP se suele hacer referencia al **paralelismo de la máquina** que es una medida de la capacidad

del microprocesador para aprovechar el ILP. Este paralelismo de hardware depende del número de cauces y de la sofisticación de los algoritmos usado para detectar instrucciones mutuamente independientes cuya ejecución puede simultanearse.

Los expertos han aislado una serie de restricciones que limitan el nivel de paralelismo a nivel de instrucciones. Las principales son las siguientes:

- **Dependencia de datos verdadera** (conocida también como dependencia de lectura tras escritura ó RAW -read after write-): se produce este tipo de dependencias cuando una instrucción tiene como entrada el resultado producido por una instrucción anterior. Por ejemplo, si en una instrucción se multiplica el contenido de un registro por otro y en la siguiente se lleva el resultado a un tercer registro (ver código a continuación), es evidente que las dos instrucciones no se podrán ejecutar en paralelo. En general, la ejecución de cualquier instrucción deberá retrasarse hasta que todos los datos de entrada estén disponibles.

```
mul reg1, reg2;   multiplica el registro 1 por el registro 2 y deja el resultado en el registro 1
mov reg3, reg1;   mueve al registro 3 el contenido del registro 1
```
- **Dependencia relativa al procedimiento:** cuando se produce una bifurcación condicional, las instrucciones que siguen al salto presentan una dependencia relativa al procedimiento respecto a las de salto y no se pueden ejecutar hasta que no se conozca el resultado de la bifurcación. Otro caso de este tipo de dependencia se da cuando la longitud de las instrucciones es variable (típico en los procesadores CISC) lo que hace que una instrucción tenga que ser decodificada, en todo o en parte, antes de la captación de la siguiente instrucción. Por este motivo, las técnicas superescalares se aplican más a los procesadores RISC en los que la longitud de la instrucción es fija.
- **Conflictos en los recursos** se da este tipo de dependencias cuando dos o más instrucciones compiten por el acceso a un recurso determinado (por ejemplo una unidad funcional). Es un tipo de dependencia similar a la de datos, pero en este caso se puede subsanar incrementando el número de recursos del mismo tipo.
- **Dependencia de salida y antidependencias:** se estudian en el siguiente apartado.

2.3.4.3. Gestión del orden de atención a las instrucciones

2.3.4.3.1. Concepto

Tanto en el caso de las técnicas segmentadas como en el de las superescalares, los problemas y detenciones causados por dependencias en los datos o por conflictos de control de flujo pueden ser evitados mediante la reordenación de las instrucciones antes de proceder a su ejecución. Evidentemente, existe un límite a estas manipulaciones, límite estipulado por el nivel de ILP. De este modo, el paralelismo no se consigue simplemente multiplicando el número de unidades funcionales (vg. el número de cauces) sino que el microprocesador ha de llevar a cabo una compleja tarea de captación, decodificación y ejecución de instrucciones teniendo en cuenta la identificación precisa del nivel de ILP subyacente con el fin de obtener el máximo beneficio y de explotar al máximo el paralelismo del hardware. A todo este proceso se le suele conocer como **despacho de instrucciones** y al protocolo que se sigue para su optimización, **política de despacho de instrucciones**.

Aunque en principio esto no sea evidente, el microprocesador no tiene por qué respetar el orden inicial en el que le llegan las instrucciones que ha de ejecutar. De hecho, en referencia a este orden existen tres posibles ámbitos de decisión:

- El orden de captación de las instrucciones.
- El orden de ejecución de las instrucciones.
- El orden en que las instrucciones alteran los registros y la memoria.

Para maximizar el paralelismo el procesador puede alterar cualquiera de los órdenes anteriores con respecto al estricto orden secuencial en el que le son suministradas las instrucciones. La única regla a cumplir es que el resultado final ha de ser válido, pero cómo se llegue a él depende completamente de la habilidad del procesador para gestionar el flujo de instrucciones. De este modo, las detenciones provocadas por dependencias de datos, por ejemplo, pueden evitarse reordenando las instrucciones para que no dependan de los resultados de instrucciones anteriores que pueden todavía estar ejecutándose.