

PREDICCIÓN HARDWARE DE BIFURCACIONES

- ❖ Predicción dinámica: más acertada y costosa que la estática
- ❖ **Efectividad** esquema de predicción dinámica depende de
 - Porcentaje de aciertos en las predicciones
 - Tiempo medio adicional consumido por el esquema de predicción dinámica
- ❖ Se precisa más efectividad cuanto mayor sea la frecuencia de las instrucciones de bifurcación
- ❖ **Imprescindible** en procesadores que emiten más de una instrucción por ciclo ($\Rightarrow$ más bifurcaciones por ciclo)
 - Un elevado porcentaje de fallos en la predicción de bifurcaciones limitaría el rendimiento de esos procesadores

TABLA DE PREDICCIÓN DE BIFURCACIONES

❖ *Branch prediction buffer o branch history table (BHT)*

❖ Tabla indexada por la parte menos significativa del PC

❖ Campo de predicción

- Versión más simple: 1 bit indica si la última vez se saltó o no

✓ Se predice lo mismo que ocurrió la última vez

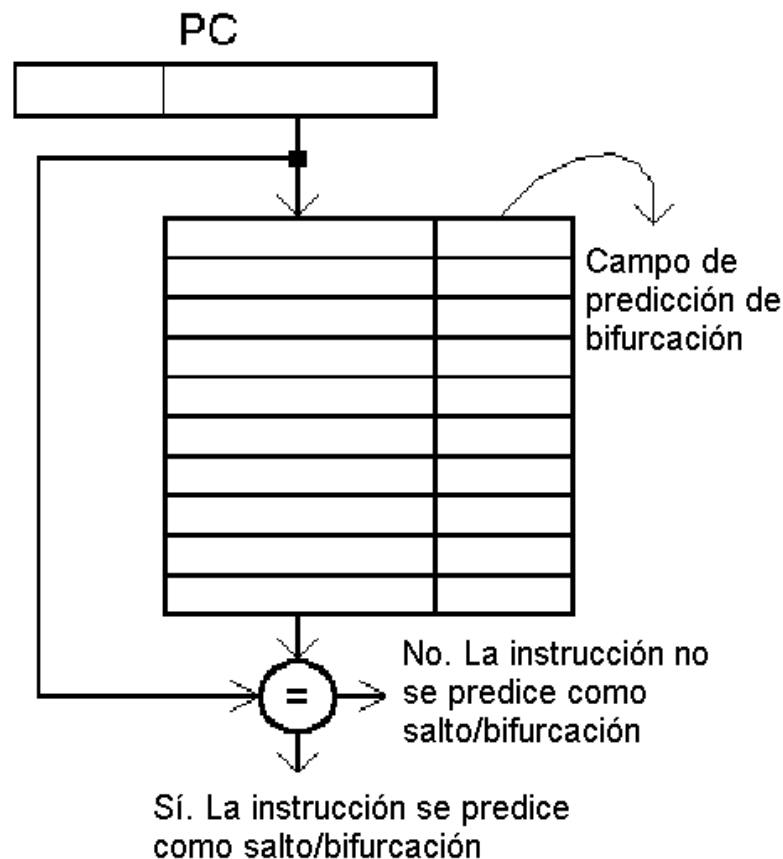
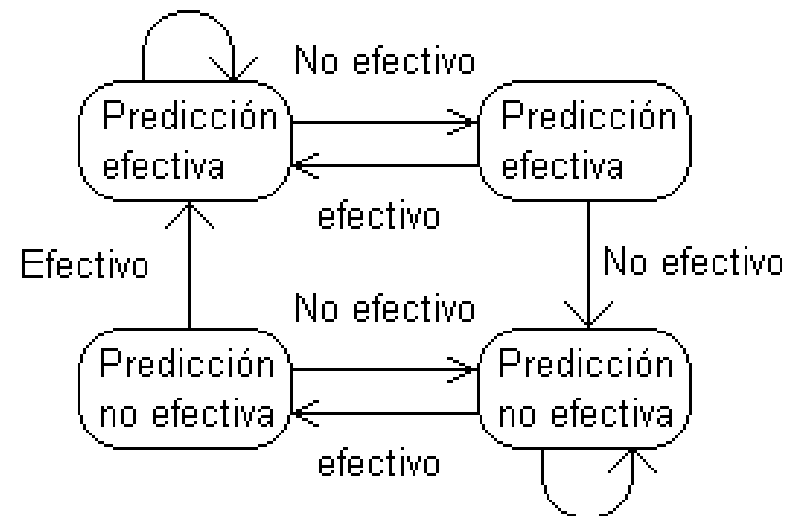


TABLA DE PREDICCIÓN DE BIFURCACIONES (cont.)

- Ejemplo de predicción con dos bits
 - ✓ Primer bit: predicción actual. Segundo bit: lo que ocurrió la última vez =>
 - La predicción se cambia al fallar dos veces seguidas
 - ✓ Más acierto que la predicción con un bit



Predicción del salto con dos bits de estado

TABLA DE PREDICCIÓN DE BIFURCACIONES (cont.)

- ❖ La BHT es accedida en la etapa de búsqueda
 - Si se predice salto efectivo, el PC se actualiza en cuanto se conozca la dirección de salto
 - ✓ Cuanto antes se conozca menor penalización en los saltos efectivos acertados
 - Más adelante, una vez evaluada la condición de salto, puede ser necesario **actualizar la BHT**
- ❖ Claramente, la BHT sólo tiene sentido en cauces en que se conoce la **dirección de salto antes** de evaluar la condición de salto

MEJORAS DEL ESQUEMA DE PREDICCIÓN

- ❖ *Correlating predictor o two-level predictor*
- ❖ El comportamiento de una bifurcación está relacionado con el de las bifurcaciones ejecutadas poco antes
- ❖ Idea: considerar dos niveles de historia para hacer la predicción:
 - Global: comportamiento de las k últimas bifurcaciones ejecutadas
 - Local: comportamiento de la bifurcación teniendo en cuenta como se ha comportado las últimas veces que ocurrió un determinado comportamiento de las k últimas bifurcaciones

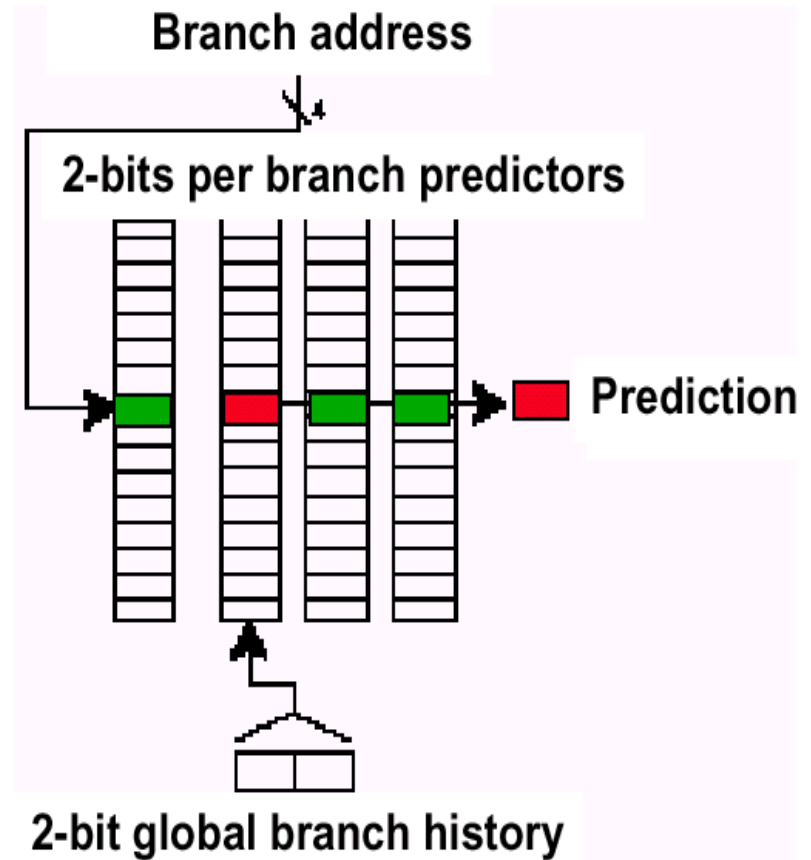
MEJORAS DEL ESQUEMA DE PREDICCIÓN (cont.)

❖ Implementación (*k-m predictor*):

- *Branch History Register* (BHR): registro de desplazamiento de k bits para la historia global
- Hay 2^k **tablas** de predicción de salto, con **entradas de m bits** (para la historia local)
- El BHR determina la tabla a la que hay que acceder
- También puede entenderse que hay una sola tabla que se direcciona concatenando los bits menos significativos del PC con los del BHR

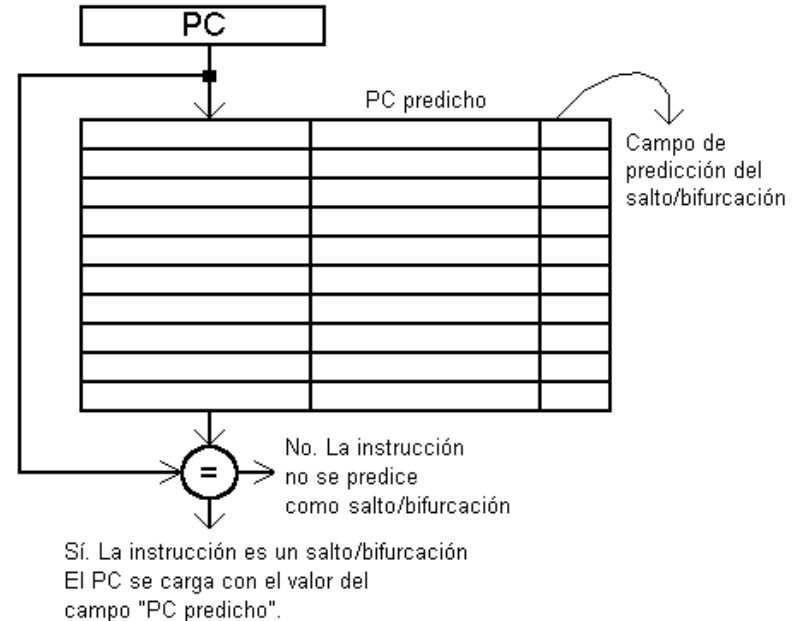
MEJORAS DEL ESQUEMA DE PREDICCIÓN (cont.)

- ❖ En transparencia 2, si el campo de predicción es de m bits tendríamos un $0-m$ predictor
- ❖ En la fig. de la dcha. Se tiene un $2-2$ predictor
- ❖ Se ha comprobado que un $2-2$ predictor con tablas de 1K entradas consigue más aciertos que un $0-2$ predictor con tablas de 4k entradas (igual tamaño total) e incluso mayores



BUFFER DE DESTINOS DE LA BIFURCACIONES

- ❖ *Branch-target buffer* (BTB) o *branch-target cache*
- ❖ El BTB es accedido en la etapa FETCH usando la **dirección completa** del PC
- ❖ Si la **predicción** es **acertada**, la **penalización** de la bifurcación es **nula**



BUFFER DE DESTINOS DE LA BIFURCACIONES (cont.)

- ❖ Bifurcación resuelta → Verificar si la bifurcación fue correcta =>
 - A menudo será preciso actualizar el BTB =>
Detención en la fase FETCH
- ❖ Campo de predicción de la bifurcación: opciones
 - Dos bits
 - ✓ Inconveniente: Puede ser preciso actualizar el BTB incluso aunque la predicción hubiese sido correcta

BUFFER DE DESTINOS DE LA BIFURCACIONES (cont.)

- Un solo bit
 - ✓ Sólo se cambia la predicción cuando no ha sido acertada => En los aciertos la penalización por la predicción dinámica es nula
- Suprimirlo y sólo colocar en el BTB las bifurcaciones que se predicen como efectivas
 - ✓ Modificar el BTB sólo cuando se predice bifurcación efectiva y no lo es o cuando no se predice y lo es

PREDICCIÓN DE SALTOS INDIRECTOS

- ❖ La dirección de destino varía en tiempo de ejecución. Caso típico: retorno de subrutinas
- ❖ Un simple BTB no predecirá bien estos saltos
- ❖ Varios procesadores de alto rendimiento usan una pila de direcciones de retorno de subrutinas
 - Al saltar a la subrutina se coloca la dirección de retorno en esa pila
 - Al retornar, se recoge la dirección de la pila
- ❖ Una alternativa, de menor coste, es usar una BTB normal en la que se elimina cualquier entrada que produzca varias predicciones consecutivas fallidas (sin determinar la causa)