ARQUITECTURA DEL PROCESADOR II

Práctico U2.2 complementario - Predicción de saltos y excepciones

Ejercicio 1: La importancia de tener un buen predictor de saltos depende de cuan frecuentemente se ejecutan saltos condicionales. En conjunto la exactitud del predictor de saltos y la cantidad de saltos ejecutados, determinan cuanto tiempo se gasta en atascos debido a los saltos mal predichos. En este ejercicio se asume que la división por categoría de las instrucciones es la siguiente:

Tipo R	BEQ	JMP	LW	SW
40 %	25 %	5 %	25 %	5 %

También asume las siguientes exactitudes de los predictores de saltos:

Siempre tomado	Siempre no-tomado	2 bits
45 %	55 %	85 %

1. Los ciclos de atasco debido malas predicciones de los saltos incrementan el CPI. ¿Cuál es el CPI extra debido a saltos mal predecidos con el predictor “siempre tomado”? Asuma que la resolución del salto se hace en la etapa EX, que no hay riesgos de datos y que no se usa la ranura de demora.
2. Repita el ejercicio anterior para el predictor “siempre no-tomado”.
3. Repita el ejercicio anterior para el predictor “2 bits”.
4. Con el predictor de 2bits, ¿que ganancia se podría conseguir si de alguna manera se pudieran convertir la mitad de las instrucciones de salto en instrucciones de ALU?. Asuma que las instrucciones que se predicen correctamente y aquellas que se predicen incorrectamente tienen las mismas posibilidades de ser reemplazadas.
5. Con el predictor de 2 bits, ¿qué ganancia se podría conseguir si pudiésemos convertir de alguna manra la mitad de las instrucciones de salto en dos instrucciones ALU? Asuma que las instrucciones que se predicen correctamente y aquellas que se predicen incorrectamente tienen las mismas posibilidades de ser reemplazadas.
6. Algunas instrucciones de salto son mucho más predecibles que otras. Si se sabe que el 80% del total de instrucciones de salto ejecutadas son saltos hacia atrás fáciles de predecir y que siempre son predichas correctamente, ¿cuál es la exactitud del predictor de 2 bits sobre el 20% de instrucciones de salto restante?

Ejercicio 2: Este ejercicio explora cómo el manejo de las excepciones afecta el diseño del procesador segmentado. Los tres primeros problemas de este ejercicio se refieren a las dos instrucciones siguientes:

Instrucción 1	Instrucción 2
BNE R1, R2, Label	LW R1, 0(R1)

1. ¿Qué excepciones pueden disparar cada una de estas instrucciones?. Para cada una de estas excepciones especifique la etapa en la cual es detectada.
2. Si hay una dirección de manejador distinta para cada excepción, mostrar cómo la organización del procesador se debe cambiar para ser capaz de manejar esta excepción. Se puede asumir que la dirección de estos manejadores se conocen cuando se diseña el procesador.
3. Si *Instrucción 2* es recuperada inmediatamente después de *Instrucción 1,* describa que ocurre en el procesador cuando la primer instrucción causa la primer excepción de acuerdo a lo enumerado en el problema 1.
4. En el manejador de excepción vectorizado, la tabla de direcciones de excepción está en memoria en una dirección (fija) conocida. Modifique el procesador para

implementar este mecanismo de manejo de excepción. Repita el problema 3 usando este procesador modificado que maneja las excepciones de manera vectorizada.

5. Se desea emular el manejo vectorizado de excepciones (descripto en el problema 4) sobre un procesador que tiene solamente una dirección fija de manejador. Escriba el código que debiera estar en esa dirección fija. Pista: este código debiera identificar la excepción, obtener la dirección correcta desde la tabla de vectores de excepción y transferir la ejecución a ese manejador. Considere que el registro $cause mantiene el código con la causa de la excepción y el registro $epc mantiene la dirección de la instrucción siguiente a la que causó la excepción.