Arquitectura de Computadoras

Solución del Parcial 2015

- *El parcial consta de 5 preguntas que se deben responder por escrito en hojas aparte. Completar TODAS las hojas con el nombre y el número de cédula. Numerarlas y escribir el total en la primer hoja.*
- *Utilizar una única carilla e iniciar cada respuesta en una hoja nueva.*
- *No se puede utilizar material de ningún tipo.* ***Apagar celulares.***
- *Se dispone de cartillas Intel 8086 y MIC-1.*
- *El parcial dura 1 hora y media.*
- *Se contestarán preguntas hasta 20 minutos antes de la hora de finalización.*

Pregunta 1

Demuestre que la minimización con mapas Karnaugh no es única.

Solución:

a\bc	00	01	11	10
0	1	1	1	
1	1		1	

Dado este mapa, se puede minimizar como $f = \overline{bc} + bc + \overline{ab}$ o como $f = \overline{bc} + bc + \overline{a}c$

Pregunta 2

Indique cuantas veces se ejecuta la instrucción `inc cx` al correr los siguientes dos fragmentos de código 8086. Justifique.

```
Fragmento 1:
  ...
  ...
  mov ax, bx
etiq1:
  inc cx
  xor cx, cx
  jnz etiq1
  mov bx, ax
  ...
```

```
Fragmento 2:
  ...
  ...
  mov ax, bx
  xor cx, cx
etiq1:
  inc cx
  jnz etiq1
  mov bx, ax
  ...
```

Solución:

Fragmento 1: dado que el xor actualiza las banderas, la bandera Z valdrá 1 luego de su ejecución, por lo tanto el salto no se tomará (jnz salta si $Z = 0$), provocando que la instrucción inc cx se ejecute una única vez.

Fragmento 2: la primera vez que se ejecuta el salto *jnz*, se hace con el valor $CX = 1$, por lo tanto, dado que *inc* también actualiza la bandera Z, $Z = 0$ y por lo tanto el salto se toma. La segunda vez se ejecutará con $CX = 2$ y por tanto también Z valdrá 0, con lo cual se tomará nuevamente el salto. Así sucesivamente hasta que $CX = 0$ (y por lo tanto $Z = 1$), lo cual se dará cuando CX pase de 0xFFFF a 0, al dar overflow la instrucción *inc*, lo cual sucederá en su ejecución número 2^{16}.

Pregunta 3

Indique:

- Regla de codificación
- Rango de representación
- Conservación del orden
- Cantidad de representaciones del cero
- Conservación de operaciones

A) Complemento a dos de números enteros.
B) Punto fijo de números fraccionarios.

Solución:

A) Los números positivos (y el 0) se representan en expresión en base 2 (rellenando con 0s a la izquierda) mientras que para representar números negativos se complementa la expresión en base 2 de su valor absoluto y se le suma 1. Ej: Para n=4, la representación del 7 es 0111 y la del -7 es 1001. Para n bits, el rango de la representación es $[-2^{n-1}, 2^{n-1}-1]$. El 0 tiene una única representación y es consistente pues $(0=-0)$.
No conserva el orden.
La representación preserva la suma y la resta (también la multiplicación y la división aunque las implementaciones de las ALUs al extender los bits de los resultados llevan a que no se conserven).

B) Se utiliza una cantidad fija de bits (n-f) para la parte entera del número y una cantidad fija (f) para representar la parte fraccionaria del número. La representación se obtiene multiplicado el número a representar por 2^f y representando el resultado en complemento a 2.
El rango $[-2^{n-f-1}, 2^{n-f-1}-2^{-f}]$
No conserva el orden. El 0 tiene una única representación.
Se preservan la suma y la resta. También la multiplicación corrigiendo el resultado (dividiendo por 2^f)

Pregunta 4

Describa hazard estructural y explique como se resuelve su ocurrencia entre las etapas de Fetch y Read.

Solución:

Un hazard es un problema asociado a los procesadores con pipeline, por el cual no se puede lograr un funcionamiento óptimo del mismo. Un hazard estructural es un tipo particular de hazard, en el cual el problema en concreto es la necesidad del uso de un mismo recurso de hardware por parte de dos o más etapas del pipeline.

En el caso particular de las etapas Fetch y Read, el hazard consiste en que ambas etapas hacen uso del bus principal para acceder a la memoria al mismo tiempo (en la etapa Fetch para obtener una instrucción y en la etapa Read para acceder a los operandos). Para un hazard estructural, la solución puede consistir en que una etapa espere o agregar hardware para que ambas puedan ejecutar sin interferencias. Para este hazard en particular, la solución típica consiste en mantener dos memorias separadas, una de instrucciones y otra de datos y disponer de buses diferentes para acceder a ellas (Arquitectura Harvard).

Pregunta 5

Se tiene un procesador de 32 bits operando con una memoria RAM de 4GB y se direcciona de a bytes. Se cuenta con una memoria caché de 64 KB y un tamaño de bloque de 16 bytes.
Indique y justifique cómo se interpreta una dirección de memoria para indexar la caché para los tres

tipos de correspondencia vistos en el curso. Para el caso de asociativa por conjuntos, utilizar 8 vías.

Solución:
Como la memoria caché es de 64 KB (2^{16} bytes) y el tamaño de bloque es de 16 bytes (2^4 bytes), entonces el caché tiene 4096 líneas (2^{12}). Si la memoria es de 4GB entonces podemos asumir que la dirección es de 32 bits.

Para el caso de función de correspondencia totalmente asociativa, la dirección se divide en TAG y BYTE. El campo byte, dado que la memoria tiene bloques de 16 bytes y la memoria es direccionable por bytes, requiere 4 bits para su direccionamiento, por lo tanto los restantes 28 bits forman el campo TAG.

31 - - TAG - - 4	3 - - BYTE - - 0

Para el caso de función de correspondencia directa, la dirección se divide en TAG, LÍNEA y BYTE. El campo línea, dado que el caché tiene 4096 líneas, ocupa 12 bits, y por tanto el campo tag ocupa los restantes 16.

31 - - TAG - - 16	15 - - LÍNEA - - 4	3 - - BYTE - - 0

Por último, para el caso de correspondencia asociativa por conjuntos de n vías, la dirección se divide en TAG, CONJUNTO y BYTE. Para este caso particular de 8 vías, dado que el caché tiene 4096 líneas, este tiene 4096 / 8 = 512 conjuntos. Por lo tanto el campo conjunto ocupa 9 bits. Los 19 bits restantes forman el TAG.

31 - - TAG - - 13	12 - - CONJUNTO - - 4	3 - - BYTE - - 0