Universidad de Costa Rica
Escuela de Ciencias de la Computación e Informática

CI-0120 Arquitectura de computadores I-2021

Primer Examen Parcial

Prof. Francisco Arroyo

Martes 25 de Mayo del 2021

Observaciones generales

i. La solución debe ser **presentada** como máximo el día **Martes 2021/Jun/01** antes de la media noche
ii. La solución del examen puede ser presentada por grupos de dos personas o de manera individual
iii. Las soluciones deben seguir las buenas prácticas estudiadas
iv. Escriba sus suposiciones
v. Debe entregar en un archivo compactado todos los elementos de su solución
vi. Debe entregar su solución en mediación virtual y guardar una versión en su repositorio de control de versiones

A) (60 pts.) Implantar en "*logisim*" un simulador para una memoria cache asociativa de 2 vías.

Las direcciones de memoria que debe implantar son de 16 bits. La cache deberá tener 16 entradas con cuatro bytes cada una. Tendremos un circuito principal donde colocaremos estas 16 entradas o líneas de la cache, además este circuito principal tendrá una memoria que contrendrá los datos a buscar cuando ocurren los fallos. Habrá una entrada de 16 pines para la dirección solicitada, que se hará de manera interactiva con el usuario; la salida de este circuito es el dato de 8 bits que hemos solicitado con la dirección. Cada entrada de la cache tendrá cuatro bytes.

- Entregable 1 (10 pts.) escriba en un documento, sin la intervención de "*logosim*"
  1. Describa los elementos componentes de las direcciones de 16 bits (bits para el índice de la cache, bits para la etiqueta de la cache (tag), bits para obtener el deplazamiento del byte) y cualquier otro que considere importante
  2. Explique las condiciones para que ocurra un "*miss*"
  3. Liste los componentes que piensa debe tener cada línea de la cache y explique su funcionamiento

- Entregable 2 (30 pts.) construya en "*logosim*" para representar una línea (conjunto o entrada) de la cache
  1. Explique cuáles elementos componen la ruta de datos (datapath) y cuáles son de control en su circuito
  2. Entradas (pines)
     a) *Data-in*: valor de 32 bits, si ocurrió un "miss", este campo contrendrá datos que vienen del siguiente nivel de memoria que debe ser guardado en esta línea
     b) *Etiqueta*: valor de la etiqueta que queremos buscar en la memoria cache
     c) Reloj
     d) Otras que considere necesarias
  3. Salidas (pines)
     a) *Hit*: indicador de que encontramos el datos en esta línea de la cache
     b) *Data-out*: valor de 32 bits con el dato obtenido de la memoria cache

- Entregable 3 (20 pts.) construya en "logisim" el circuito principal que muestre el correcto funcionamiento de las líneas de cache diseñadas en los pasos anteriores
  1. Explique cuáles elementos componen la ruta de datos (datapath) y cuáles son de control en su circuito
  2. Entradas (pines)
     a) *Address*: valor de 16 bits
  3. Salidas (pines)
     a) Byte: solicitado a la memoria
     b) Otras que considere necesarias

B) (10 pts.) Considere la siguiente secuencia de instrucciones de alto nivel:

```
a = b + c
b = a + c
d = a – b
```

Utilizando la técnica de **propagación por copia** para transformar este secuencia hasta el punto en que ningún operando es un valor calculado. Anote las líneas en las que la transformación ha reducido el trabajo computacional y los casos en que aumentó.

C) (20 pts.) Calcule el CPI efectivo para una implantación de una CPU RISC-V empleando la siguiente tabla y los programas indicados en la columna de equipo, por ejemplo, el equipo 4 utiliza un promedio de los datos de los programas *libquantum* y *mcf*:

Equipo	Programs	Loads (%)	Stores (%)	Branches (%)	Jumps (%)	ALU operations (%)
1, 8	*astar*	26	6	18	2	45
1, 9	*bzip*	21	7	11	1	53
2, 8	*gcc*	17	23	20	4	34
2, 9	*gobmk*	21	12	14	2	47
3, 10	*h264ref*	33	14	5	2	41
3, 11	*hmmer*	28	9	17	0	45
4, 10	*libquantum*	16	6	29	0	47
4, 11	*mcf*	35	11	24	1	28
5, 12	*omnetpp*	23	15	17	7	29
5, 13	*perlbench*	25	14	15	7	37
6, 12	*sjeng*	19	7	15	3	53
6, 13	*xalancbmk*	30	8	27	3	29

Table 1: Mezcla de instrucciones dinámicas para un RISC-V con los programas SPECint2006

Suponga que se ha realizado las siguientes medidas promedio de CPI para cada tipo de instrucciones:

Tipo de instrucción	Ciclos de reloj
Todas las operaciones de ALU	1
Loads	5
Stores	3
Branches	
Tomados	5
No tomados	3
Jumps	3
Otros	3

Table 2: Ciclos de reloj por tipo de instrucción

D) (10 pts.) Utilice el siguiente fragmento de código:

```
Loop:   load    x1, 0(x2)       ; load x1 from address x2 + 0
        addi    x1, x1, 1       ; x1++
        sd      x1, 0, (x2)     ; store x1 at address x2 + 0
        addi    x2, x2, 4       ; x2 = x2 + 4
        sub     x4, x3, x2      ; x4 = x3 – x2
        bnez    x4, Loop        ; branch to Loop if x4 != 0
```

Muestre la tabla de tiempos de la secuencia de instrucciones para un pipeline RISC de 5 etapas, similar a la figura C-8; no disponemos de hardware adicional para realizar “forwarding” ni “bypassing”, pero suponga que la lectura y escritura de los registros ocurre en el mismo ciclo y el valor es adelantado entre registros como se muestra en la figura C-5.