Arquitectura de computadoras 2022 - Parcial 2A

Nombre:

El sistema posee una CACHE exclusiva para INSTRUCCIONES de correspondencia ASOCIATIVA POR CONJUNTOS de 2 VIAS de 128 bytes y 2 palabras de 32 bits por línea, sobre un procesador de 64bits, CPI = 1, que resuelve todos los data y control hazard sin necesidad de stalls, tiene una memoria principal de 4G palabras de 1 byte cada una. Considerar que la CACHÉ contiene los datos mostrados a continuación al inicio de la ejecución del segmento:

			Via #0			Via #1
Set	Tag	V	Data	Tag	V	Data
000	Bece528	1	8b1f03ff_8b1f03ff	80ce521	1	941055c_5500604a
001	3ffffff	9	********	3111111	0	ecestere eccesses
010	01F5580	1	8b1f83ff_17fffffa	00ce581	0	00000000_00000000
011	366666	1	301f030f_87f00ffa	366666	1	90160366_5706668
100	344444	0	ffffffff_ffffffff	344444	0	umun'mum
101	1 FEEEE	8	86666666 6666666	3ffffff		ettette tettette
110	344444	10	* *********	01f5580	-3-	961f0300_870f65f8
311	00000f1		1 9418b5bc_b588884a	SFFFFFF	0	66666666 6666666

a) Determinar el estado de la CACHÉ al final de la ejecución del segmento para N = 100.
Para esto completar la siguiente tabla SOLO con el contenido de las lineas de CACHÉ que hayan sufrido modificaciones. El tamaño de la tabla no representa la cantidad de lineas necesarias.

Set Tag V Data		NAME OF TAXABLE PARTY.	
1.1	-		
	u mito		
C . C . C	1830		The Boat

b) Determine la cantidad de ciclos de clk necesarios para su ejecución en las mismas condiciones del punto a), considerando que cada MISS de caché tiene un tiempo de ejecución de 10 ciclos. Suponer que el pipeline ya se encuentra en régimen y que la CACHE de DATOS solo produce aciertos en los accesos, por lo que no se tiene penalidades por acceso a datos.

Expresión:	Respuesta:	ciclos de clk
	A 5151 A 619 P 2010 S 15	

Ejempioio 4

Se cuenta con un predictor por tomeos que está compuesto por dos predictores más simples, un predictor global con un GR de 4 bits y un predictor de dos bits clasico. Considerando el siguiente segmento de código en LEGVA y que los registros implicados estan incializados con los siguientes valores. X0×1, X1×15, X3×93, GR×1001:

1> 1: Subis x0, #1

b.meg El 25

Do add xl, xer, xer

45 El: cmpi X1, 80

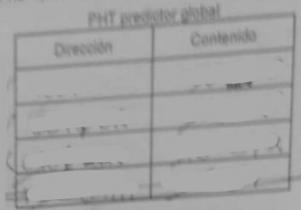
50 5.00 62

6> addi X1, X1, #1

75 E2: cmpi X1, 50

8> b.lt i 9> subis x3, X3, W1

180 cbc L



a) indicar que posiciones de la PHT del predictor global se modifican y considerando que esta completamente inicializada en cero, que valores quedarian almacenados. b) Si este segmento de código se ejecuta muchas veces, indicar cuál de los dos predictores. obtendirá mejores resultados para cada uno de los saltos:

10>

Ejercicio 5

Se muestra a continuación el estado de los registros y las reservation station en un determinado momento de un procesador out-of-order. Deducir el segmento de código que se está ejecutando.

Hardware
ssue = 2 manucciones
pad = 3 RS / 4 ofk
store = 3 RS / 4 c/k
ALU entero > 3 RS / 1 clk
Multiplicación enteros = 3 RS / 2 c/k
a) Li punto flotante = 3 RS / 2 clk
Multiplicación punto florante = 3 RS / 4 elic

	_		Ri	servation static	NTS.	200	A
Same	-	Op	VI	Vk	Q	ÇK	[X2] × 60
100.00	Busy		[X2]		0	0	
ced1	22	load			0	0	[82] + 68
tsed 7		towd	[X2]		mult tp.1	0	OB OB
		ASDEW	-	[82]	Committee of		
state I	(9)					0	
pore 2	52	addi	[X2]	26	0		-
alo int 1	- 5	sub	[X1]		0	alu ini 1	
alu int 2		900	-				
nut mil			_				
nutrini 2					load 1	load 2	
lu (p. 1		bba		-	10016.3	10490.0	-
u to 2					-		-
R61		flum		[03]	alu tp 1	0	
At (p 2	0						17
snch		cbnz			alu int 2	0	37

				Register Statu	5			
	D0	D1	D2	03	D4	D5	D6.	07
0	alufp1	load 1	load 2		mult to 1			
	XO	X1	X2	X3	X4	X5	X0	X7
01	alu int 2		alu int 1					

Considere una caché de 4Mbyte, ASOCIATIVA por conjuntos de 8 vías, dispuesta en un procesador de 64 bits con una capacidad de direccionamiento de 4Gbytes (cada byte es directamente direccionable en memoria). Se sabe que la memoria principal posee 32M

bloques. Se pide:

a) Completar cada casillero con el número de bits de cada campo del formato de dirección de memoria principal:

Tag	Set	Word	Offset byte(*
1			

^{*} Llenar con 0 en el caso que no corresponda

b) Suponga que cada LÍNEA de la caché contiene además un bit de validación (V) y 8 bits de contador de accesos. Cual es el tamaño completo de un CONJUNTO (expresado en bits) de la caché, considerando datos, tags y los bits de status antes mencionados?

Respuesta:

c) Cual es el tamaño total de cada VÍA de la caché, expresada en bits?

Respuesta: ____ bits.

d) Si la memoria caché y la memoria principal tienen tiempos de acceso de 5ns y 400ns respectivamente, ¿qué hit rate (tasa de acierto) se necesitaria para obtener un tiempo promedio de acceso a memoria (AMAT) de 25ns?

Respuesta:

200

Considere la ejecución del siguiente segmento de código LEGv8 para N > 0, donde N→ X3, y X6 contiene la dirección base del arreglo A[] del tipo uint64_t.

ADDRESS	<label></label>	OPCODE	ASSEME	BLY
ex83394834	<count_z>: <loop_for>:</loop_for></count_z>	0x8b1f03e0 0x8b0303e9 0xb40000e9 0xf84000ca 0xb500004a	add add cbz ldur cbnz	x0, xzr, xzr x9, xzr, x3 x9, NEXT x10, [x6] x10, NOT_Z
0x03394848 <	NOT_Z>:	0x9410b5bc 0xd1000529 0x910020c6 0x17fffffa	bl subi addi b	ZERO x9, x9, #0x1 x6, x6, #0x8 LOOP_FOR
KØ3394854 KNE	EXT>:	0x8b1f03ff	add	xzr, xzr, xzr

0x037c1f38 <ZERO>;

x0, x0, #0x1 addi 0x91000400

x30 0xd61f03c0 br

Ejercicia 3

Dado un procesador de arquitectura LEQu8 2-ssue, que precise los saltes perfectamente, de modo que los hazant de control son manejados por hardware, con una modificación que permite que en cada issue packet una instrucción pueda ser qualquier tipo y la oria deba ser una instrucción antimetica logica. Para el siguiente tragmento de cédigo LEGv8, donde todos los registros se enquentran inicialización por 100.

		11年日日日本 11日日日におりた3
	D	ACCI 18,128,432
LOOP		AUDE X1, X10, #8X100
		SUB X2, X10, X0
	45	CB2 X2, LOOP_END
	5>	AUG X11,X1,XZR
	5>	ADDI X10,X10,48
	75	LOUR X12,[X11,#0]
	8	LOUR X13,[X11,#8]
	9>	ADD X14,X12,X12
		ORR X12, X13, X13
		STUR X13,[X11,#0]
		STUR X12,[X11,#8]
	530	R 100P

Instrucción de cualquiel tipo	instrucción artimética/stgica
\$	-6
	10
9	19 N
1 40	0
-13	10
43	N
13	A.

LOOP END: ...

- a) Sin alterar el orden de las instrucciones, mostrar en la tabla de amba cómo organizaria los issue packets para ejecutar el programa en la menor cantidad posible de ciclos de clock (cada instrucción sólo puede agruparse con la inmediata anterior, la inmediata posterior o una nop). Usar los números correspondientes para referirse a las instrucciones del código.
- b) Mostrar el orden de ejecución del código del punto "a" en el procesador 2-issue (sólo hasta completar una iteración del bucle "LOOP"). Indicar los caminos de forwarding utilizados.
- c) El siguiente código resultó de aplicar las técnicas de loop unrolling y register renaming al tragmento de código dado. Sin alterar los resultados obtenidos tras la ejecución del mismo y tragmento de código dado. Sin alterar los resultados obtenidos tras la ejecución del mismo y tragmento que siempre en la instrucción nº1 se asigna el número 32 al registro XO, eliminar asumiendo que siempre en la instrucciones posible y completar los espacios vacios en las tratantes.

```
LOUP: ADDI X1,X19, (X3)

SUB X2,X18,X8

CBZ XZ, LOOP_END

ADDI X10,X10, #16

LDUR X22,[X11,*-150] X

LDUR X23,[X11,*-32-3] X

ADD X24, X11,*-32-3] X

ORR X15, X13, (X11, 200 0)

STUR X23 ([X11, 200 0]

STUR X23 ([X11, 200 0]

STUR X13, [X11, 200 0]
```