CC4301 Arquitectura de Computadores
Solución Examen

Profesor: Pablo Guerrero P.
Prof. Aux.: Gaspar Pizarro V.
Ayudante: Ian Yon Y.

9 de diciembre de 2012

1. Conceptos

a) Es util porque permite independizar la CPU del manejo de los datos entre disco y memoria. La limitante principal es que de todas formas se comparte el bus de datos, de forma que solo un dispositivo (CPU y DMA) puede usar el bus. La linea DREQ es la linea con la que el dispositivo E/S pide el bus de datos al DMA. La linea HOLD es la linea con la que el DMA pide el bus de datos. La linea HOLDA es la linea con la que la CPU avisa que el bus está libre para que el DMA lo use.

b) Una linea sucia es una linea en el caché que tiene información no válida, porque otro nucleo ha escrito en este o en memoria. La principal ventaja del bus snooping es que permite eliminar la redundancia en los datos de los distintos cachés. Una desventaja es que agrega complejidad en el controlador de memoria para implementar el bus snooping. El estado *modified* se le asigna a las lineas que han sido modificadas por el procesador, el estado *exclusive* se le asigna a las lineas que no han sido modificadas y están solo en un caché, el estado *shared* se asigna a las lineas que podrían existir en el caché de otro procesador, y el estado *invalid* se asigna a las líneas que han sido escritas externamente en la memoria, por ejemplo con DMA.

c) Los problemas que aparecen en pipelining de grado 1 son:

- Accesos a memoria: Como el bus de datos solo puede se accedido para obtener solamente un valor, entonces las instrucciones siguientes que requieran acceder a memoria tienen que esperar. Si se necesita el valor obtenido en memoria para ejecutar otra instrucción, se produce un *pipeline stall*, y la siguiente instrucción debe esperar. Para solucionar esto se usan las técnicas de *register bypassing*, *register scoreboarding* y ejecución fuera de orden.
- Bifurcaciones: En las bifurcaciones se carga de todas formas la instrucción siguiente a la bifurcación. Pero en muchos casos, si el salto ocurre, se debe reiniciar el pipeline. Para solucionar esto se usa la ejecución especulativa o *branch prediction*.

d) i) *Register Renaming* consiste en cambiar el registro utilizado en una instrucción para evitar el *pipeline stall* producido por la dependencia aparente de registros. Es útil cuando se produce el caso de *pipeline stall* mencionado anteriormente.

ii) Ejecución fuera de orden es cambiar el orden de ejecución de las instruciones dinámicamente para ejecutar tardíamente las instrucciones posteriores a una instrucción de memoria que dependan de estos datos.

iii) Ejecución especulativa consiste en intentar precedir el resultado de un salto cargando la instrucción que se cree va a ejecutarse (la instrucción siguiente al salto o la instrucción a la que apunta el salto). Se usa en los saltos.

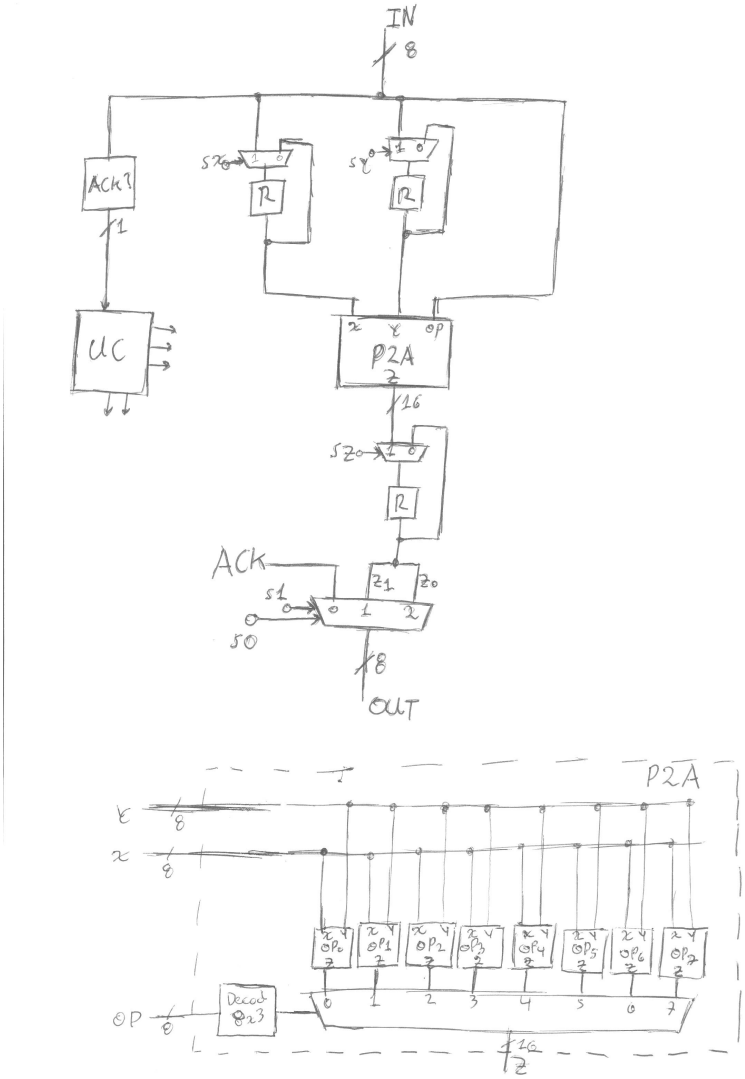
e) La memoria caché es más rápida, más cara y más chica que la memoria que la RAM. La memoria caché se implementa con SRAM porque es más rápida mientras que la memoria RAM se implementa con DRAM porque es más barata. Son complementarias porque la caché puede almacenar elementos de la RAM que sean actualizados con frecuencia.

Asignación de Puntaje

Definida en enunciado.

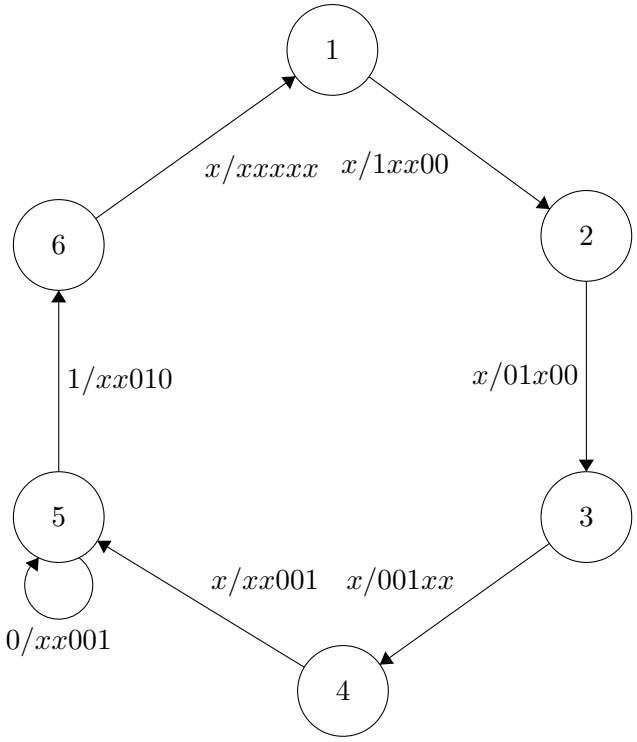
2.

a, b


IN
8
ACK?
1
UC
SZ
R
SY
R
x
y
op
P2A
z
16
SZ
R
ACK
S1
S0
z1
z0
0
1
2
8
OUT

P2A
y
8
x
8
OP0
OP1
OP2
OP3
OP4
OP5
OP6
OP7
OP
8
Decod 3:8
0
1
2
3
4
5
6
7
16

c



Estado 1= 000
Estado 2= 001
Estado 3= 010
Estado 4= 011
Estado 5= 100
Estado 6= 101

La tabla de verdad del circuito combinacional y los mapas de karnaugh asociados están en las tablas 1, 2, 3, 4, 5, 6, 7, 8 y 9

A	Q_2	Q_1	Q_0	S_X	S_Y	S_Z	S_1	S_0	D_2	D_1	D_0
x	0	0	0	1	x	x	0	0	0	0	1
x	0	0	1	0	1	x	0	0	0	1	0
x	0	1	0	0	0	1	0	0	0	1	1
x	0	1	1	x	x	0	0	1	1	0	0
0	1	0	0	x	x	0	0	1	1	0	0
1	1	0	1	x	x	0	1	0	1	0	1
x	x	x	x	x	x	x	x	x	0	0	0

Cuadro 1: Tabla de verdad

$AQ_2 \backslash Q_1Q_0$	00	01	11	10
00	1	0	x	0
01	x	x	x	x
11	x	x	x	x
10	1	0	x	0

$S_X =\sim Q_1 \sim Q_0$

Cuadro 2: Mapa de karnaugh S_X

$AQ_2\backslash Q_1Q_0$	00	01	11	10
00	x	1	x	0
01	x	x	x	x
11	x	x	x	x
10	x	1	x	0

$$S_Y = Q_0$$

Cuadro 3: Mapa de karnaugh S_Y

$AQ_2\backslash Q_1Q_0$	00	01	11	10
00	x	x	0	1
01	0	x	x	x
11	x	0	x	x
10	x	x	0	1

$$S_Z =\sim Q_2 \sim Q_0$$

Cuadro 4: Mapa de karnaugh S_Z

$AQ_2\backslash Q_1Q_0$	00	01	11	10
00	0	0	0	0
01	0	x	x	x
11	x	1	x	x
10	0	0	0	0

$$S_1 = Q_2Q_0$$

Cuadro 5: Mapa de karnaugh S_1

$AQ_2\backslash Q_1Q_0$	00	01	11	10
00	0	0	1	0
01	1	x	x	x
11	x	0	x	x
10	0	0	1	0

$$S_0 =\sim AQ_2 + Q_1Q_0$$

Cuadro 6: Mapa de karnaugh S_0

$AQ_2\backslash Q_1Q_0$	00	01	11	10
00	0	0	1	0
01	1	0	0	0
11	0	1	0	0
10	0	0	1	0

$$D_2 =\sim AQ_2 \sim Q_1 \sim Q_0 + AQ_2 \sim Q_1Q_0 + \sim Q_2Q_1Q_0$$

Cuadro 7: Mapa de karnaugh D_2

3.

$AQ_2\backslash Q_1Q_0$	00	01	11	10
00	0	1	0	1
01	0	0	0	0
11	0	0	0	0
10	0	1	0	1

$$D_2 = \sim Q_2 \sim Q_1Q_0 + \sim Q_2Q_1 \sim Q_0$$

Cuadro 8: Mapa de karnaugh D_1

$AQ_2\backslash Q_1Q_0$	00	01	11	10
00	1	0	0	1
01	0	0	0	0
11	0	1	0	0
10	1	0	0	1

$$D_0 = \sim Q_2 \sim Q_0 + AQ_2 \sim Q_1Q_0$$

Cuadro 9: Mapa de karnaugh D_0

a) P1:

```
    pushl %ebp
    movl %esp, %ebp
    pushl %edi
    pushl %esi
    pushl %ebx
    movl 8(%ebp), %edi #puerto de control
    movl 12(%ebp), %ebx #puerto de datos
L1:
    movl (%edi), %ecx
    andl $1, %ecx
    testl %ecx
    jz L1
    movb (%ebx), %al
    popl %ebx
    popl %esi
    popl %edi
    leave
    ret
```

b) P2:

```
    pushl %ebp
    movl %esp, %ebp
    pushl %edi
    pushl %esi
    pushl %ebx
    movl 8(%ebp), %edi #puerto de control
    movl 12(%ebp), %ebx #puerto de datos
    movb 16(%ebp), %dl #byte a escribir
L1:
    movl (%edi), %ecx
    andl $2, %ecx
    testl %ecx
    jz L1
    movb %dl, (%ebx)
    popl %ebx
    popl %esi
    popl %edi
    leave
    ret
```

c) P2:

```
    pushl %ebp
    movl %esp, %ebp
    pushl %edi
    pushl %esi
    pushl %ebx
    movb 8(%ebp), %bl #x
    movb 12(%ebp), %cl #y
    movb 16(%ebp), %dl #op
    pushb %bl
    pushl $0xfa01
    puhsl $0xfa02
    call P1
    popb %bl
    addl $8, %esp

    pushb %cl
    pushl $0xfa01
    puhsl $0xfa02
    call P1
    popb %bl
    addl $8, %esp
```

```
pushb %dl
pushl $0xfa01
puhsl $0xfa02
call P1
popb %bl
addl $8, %esp

pushl $0xfa01
puhsl $0xfa02
call P2
shll %eax, %4
```

```
pushb $0xFF
pushl $0xfa01
puhsl $0xfa02
call P1
popb %bl
addl $8, %esp

popl %ebx
popl %esi
popl %edi
leave
ret
```

Asignación de Puntaje

Definida en enunciado.

4.

La tabla 10 muestra cuáles direcciones de memoria caen en el caché y cuales no. El estado final del

	b0a4	13b0	10a8	4810	6810	4818	0b0a	080a
etiqueta	b0a	13b	10a	481	681	481	0b0	080
índice	0a	3b	0a	81	81	81	b0	80
exito?	no	no	no	no	no	sí	no	no

Cuadro 10: Aciertos y *misses* en caché

caché se muestra en la tabla 11.

	Banco 1		Banco 2	
línea	etiq	cont	etiq	cont
0a	b0a		10a	
3b	13b			
80	080			
81	481		681	
b0	0b0			

Cuadro 11: Estado final de caché

5.

#	Intruccin	Ciclo													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	shift R1, 2, R2	F	D	A	E	S									
2	add R1, 1, R1	F	D	A	E	S									
3	load [R6+R2], R3		F	D	A	E	M	M	S						
4	add R3, R4, R4		F	D	A				E	S					
5	cmp R1, R5			F	D	A	E	S							
6	blt loop			F	D	A			E						
7	shift R1, 2, R2				~~F~~	F	D	A		E	S				
8	add R1, 1, R1				~~F~~	F	D	A		E	S				
9	load [R6+R2], R3						F	D	A		E	M	M	S	
10	add R3, R4, R4						F	D	A					E	S
11	cmp R1, R5							F	D	A	E	S			
12	blt loop							F	D	A			E		

- De las instrucciones 1 a la 3 se realiza registry bypass para el registro R2.
- Entra las instrucciones 3 y 4 se realiza registry scoreboarding del registro R3.
- De la instrucci´n 6 a la 7 y 8 se realiza una branch prediction. Las primeras fases o fetch de las instrucciones 6 y 7 se eliminan por la prediccin del branch.
- Entre instrucciones 6 y 7, 7 y 8, y 8 y 9 hay structural hazard.
- Entre las instrucciones 9 y 10 se realiza un registry scoreboarding del registro R3.
- Observemos que las instrucciones de branch se ejecutan DESPUES de la fase de store de la instruccin de comparacin, ya que es en esa fase en el cual se escribe o o el registro de flags, necesarios para determinar el branch.
- De las instrucciones 8 a la 11 se realiza registry bypass para el registro R1.