Problema 1 – Gizeh García – 17005256

I. Mapeo directo.

Dirección	Etiqueta	contenido
000		
001	~~0010~~ 1011 1	~~E~~ G
010		
011	1111 1	D
100	0000 0	B
101		
110	~~1011~~ 1111 1	~~A~~ C
111	0010 0	F

110

Dirección original	Cont
1011 1110	A
0000 0100	B
1111 1110	C
1111 1011	D
0010 1001	E
0010 0111	F
1011 1001	G

II. Mapeo 2 vías

Dirección	Etiqueta	Cont.
00	0000 01	B
01	0010 10	E
	1011 10	G
10	1011 11	A
	1111 11	C
11	1111 10	D
	0010 01	F

III. Mapeo 4 vías

Dirección	Etiqueta	cont
0	1010 111	A
	0000 010	B
	1111 111	C
	1111 101	D
1	0010 100	E
	0010 011	F
	1011 100	G

N. Completamente Asociativo.

~~Etiqueta~~ dirección	Etiqueta	cont.
1011 1110	1011 1110	A
0000 0100	0000 0100	B
1111 1110	1111 1110	C
1111 1011	1111 1011	D
0010 1001	0010 1001	E
0010 0111	0010 0111	F
1011 1001	1011 1001	G

Problema 2 - Gizeh Garcia - 170052560

1. CISC
2. RISC
3. RISC
4. CISC
5. RISC
6. CISC
7. RISC
8. CISC
9. CISC
10. RISC

Problema 3

a. foo(r1, r1, r2)

b. hace un branch para salir de la subrutina, es bx porque así puede tomar la dirección actual de PC más un desplazamiento, si solo se usa un branch normal (B) vamos a caer en donde esté el PC y puede haber conflicto.

c. Bx como ya dije es para salir de la subrutina, sobre todo porque no guarda la dirección de memoria en la que estaba, BLX se utiliza más para entrar a una subrutina, ya que guarda la dirección en donde estaba y así es más facil regresar.

d. Cuando se manda a llamar bar, los registros r1 y R2 aún no están listos, esto crea un problema en el looping, y entonces no se puede seguir adelante con el código, además, se queda trabado.

e. foo
```
mov r1, #0
ands r1, r1, r2
bl bar
bx lr
```

Problema 4 - Gizeh García - 17005256

a. Son impares porque de esta manera se puede llevar un mejor control. y se puede accesar a las memorias más fácilmente.

b. las estrategias estáticas se ~~crean~~ utilizan en el compilador y en el ámbito del Software porque son hazards a nivel de control o estructura del código.
las estrategias dinámicas se implementan en el hardware, como por el procesador. Una de ~~de~~ estas estrategias sería ~~tener~~ agregar un corrimiento de branches, para poder tener más memoria

C. Asumiendo que $0x0000\_2210$, Cuanto vale después de POP {r0-r6, lr, Pc}?

r0 → 0x0000_2210
r1 → 0x0000_22F0
r2 → 0x0000_22B0
r3 → 0x0000_2111
r4 → 0x0000_2100
r5 → 0x0000_2000
r6 → 0x0000_1FFF
PC → 0x0000_1FFB

d. ventajas del uso de interrupciones sobre polling.
- Cuando se produce una interrupción se puede ver una bandera en esa dirección que puede ayudar para entrar en las subrutinas o no.
- Con estas interrupciones en el polling también se pueden realizar saltos en la dirección de memoria para poder ejecutar loops.

Es mejor utilizar el direct memory address cuando sabemos bien las instrucciones y el numero de ellas es limitado.

e. Cuál es el speedup máximo que puede obtener un pipeline de 3 etapas.

3		5	8	11	14	17			
2	2	4	7	10	13	16			
1	1	3	6	9	12	15			
	1	2	3	4	5	6			

$S(n) =$

→ es imposible porque el número de tareas va a depender de la etapa en la que se esté trabajando

Problema 5 - Gizeh Garcia. - 17005256

Type	% inst programa	CPI
INT	40%	1
branch	20%	4
load	30%	2
store	10%	3

frecuencia 4.8 GHz
instrucciones 25 mil millones
salto 2 ciclos de instrucción

$$speedup = \frac{1}{(1-F) + \frac{F}{S}}$$

Problema 6

addr EQU 0x2000 0004.
alias EQU 0x22000084.
Var1 DCB 0x04
Var2. DCB 0x01, 0x02, 0x03.

R0	0x 2000 0004
R1	0x04
R2	
R3	0x 2200 0084
R4	0x03

0x2000 0008

0x2000 0000	20	00	00	04								
0x2000 0010												
0x2000 0020	02											
0x2000 0030	03								22	00	00	84