TEMA 3:

1. En un microprocesador SMT (multihebra simultánea) se pueden enviar a ejecutar varias instrucciones de una misma hebra en un instante determinado. V
2. En el protocolo MESI para mantener la coherencia de caché, una línea dada de memoria puede estar, en un momento dado, en el estado E(exclusivo) en una caché y en el estado S(compartido) en otras cachés. F
3. En el protocolo MESI, si en la caché de un nodo N1 hay un bloque B en estado E(exclusivo) y en ese nodo detecta que otro procesador en el nodo N2 intenta leer un dato que está en el bloque B, dicho bloque pasa al estado S(compartido) en las cachés del N1 y N2. V
4. En el protocolo MSI, si en la caché de un nodo N1 hay un bloque B en estado M(modificado) y ese nodo detecta que otro procesador en el nodo N2 intenta escribir un dato que está en el bloque B, dicho bloque pasa al estado I(no válido) en la caché del N1 y a M(Modificado) en N2. V
5. En un multiprocesador NUMA con 8 nodos, 16 GB por nodo y líneas de caché de 64 B. ¿Cuántas entradas tiene el directorio de memoria utilizado en cada nodo para mantener la coherencia de caché en un protocolo MSI sin difusión? 16 GB = 2^34 B. 2^34/2^6 = 2^28 líneas. entradas = B por nodo / líneas de caché(B)

6. En un multiprocesador NUMA descrito en la pregunta anterior. ¿Cuántos bits tienen cada una de las entradas del directorio que se utiliza para mantener la coherencia de caché en un protocolo MSI con directorio? 8+1 = 9 1 bit x nodo + 1 de estado
7. En el mismo multiprocesador NUMA descrito en la pregunta anterior con el mismo protocolo de caché, se puede tener el estado 1 1 0 … 0 V (1:hay copia del bloque en la caché del nodo correspondiente al bit ; 0: no hay copia en la caché del nodo correspondiente al bit; V: bloque válido en MP) en alguna de las entradas de alguno de los directorios. V
8. Si el modelo de consistencia de memoria de un multiprocesador NO respeta el orden R->W (sí respeta todo los demás), e inicialmente X=Y=Z=r1=0 (donde X,Y y Z son variables compartidas y r1 es un registro de P1), al final se podría tener Y=0. F

P1:			P2:		
//R(Z)	while(Z==0){} ;		X=1 ;	//	**W(X)**
//R(X)	r1=X ; //	**RAW EN R1**	Y=2 ;	//	**W(Y)**
//W(Y)	Y =r1 ;		Z=1 ;	//	**W(Z)**

9. Cuando se utilizan instrucciones del tipo LL/SC (lectura enlazada / escritura condicional) para implementar un cerrojo, los recursos hardware asociados a dicha técnica permiten detectar si, entre la ejecución de la lectura enlazada (LL) y la ejecución de la escritura condicional(SC) a la dirección de memoria del cerrojo, algún otro procesador ha accedido a dicha dirección. V
   Precisamente en esto consiste la técnica. Gracias al uso de una marca en un registro auxiliar

10. Si en la secuencia de instrucciones siguiente se tiene que r1=1 , r2=0 , r3=0 , dicha secuencia implementa un cerrojo (lock(k)) en el que k=1 significa que el cerrojo está cerrado y k=0 que está abierto. F, r3 debe ser igual a 1

```
b=r1 ;
do
        compare&swap(r2,b,k) ;      //si r2==k , k y b se intercambian
while(b==r3) ;
```

11. Si en la secuencia de instrucciones siguiente se tiene que r1=1 , r2=0 , r3=1 , dicha secuencia implementa un cerrojo (lock(k)) en el que k=1 significa que el cerrojo está cerrado y k=0 que está abierto. V

```
b=r1 ;
do
        compare&swap(r2,b,k) ;      //si r2==k , k y b se intercambian
while(b==r3) ;
```

12. En un microprocesador SMT(multihebra simultánea), se procesan varias hebras concurrentemente y en un instante determinado solo se pueden enviar a ejecutar instrucciones de una misma hebra. F
13. En el protocolo MESI para mantener la coherencia de caché, una línea puede estar en el estado E(exclusivo) solo en una caché del multiprocesador. V
    De ahí, precisamente el nombre del estado, aunque ocurre lo mismo con una línea en el estado M
14. En el protocolo MESI, si en la caché de un nodo N1 hay un bloque B en estado S(Compartido), y ese nodo detecta que otro procesador en el nodo N2 intenta leer un dato que está en el bloque B , dicho bloque pasa al estado S(compartido) en las cachés del N1 Y N2. V
15. En un multiprocesador NUMA con 8 nodos, 8 GB por nodo y líneas de caché de 128 B, ¿entradas que tiene el directorio de memoria utilizado en cada nodo para mantener la caché en un protocolo MSI sin difusión? 2^33 / 2^7 = 2^26 entradas
16. En el multiprocesador NUMA descrito en la pregunta anterior , ¿cuántos bits tiene cada una de las entradas del directorio que se utiliza para mantener la coherencia de caché en un protocolo MSI con directorio? 8+1(**bit de validez**) = 9
17. En el mismo multiprocesador NUMA descrito en la pregunta anterior con el mismo protocolo de caché, se puede tener el estado 1 1 0 … 0 I (1:hay copia del bloque en la caché del nodo correspondiente al bit ; 0: no hay copia en la caché del nodo correspondiente al bit; I: bloque inválido en MP) en alguna de las entradas de alguno de los directorios. F
18. Si el modelo de consistencia de memoria de un multiprocesador NO respeta el orden R->W (sí respeta todo los demás), e inicialmente X=Y=Z=r1=0 (donde X,Y y Z son variables en memorias compartidas y r1 es un registro de P1), al final SOLO se podría tener Y=1. V

```
P1:     while(Z==0){} ;                    P2:     X=1 ;
```

r1=X ; Y=2 ;
Y =r1 ; Z=1 ;

19. Cuando se utilizan instrucciones del tipo LL/SC (lectura enlazada/escritura condicional) para implementar un cerrojo, los recursos hardware asociados a dicha técnica impiden que, entre la ejecución de la lectura (LL) y la ejecución de la escritura (SC) a la dirección de memoria del cerrojo, ningún procesador puede acceder a dicha dirección de memoria del cerrojo. F
    Sí, se permite el acceso a la memoria, pero se modifica un bit de marca para tenerlo en cuenta en la instrucción SC posterior
20. Si en la secuencia de instrucciones siguiente se tiene que r1=1 , r2=0 , r3=1 , dicha secuencia implementa un cerrojo (lock(k)) en el que k=1 significa que el cerrojo está cerrado y k=0 que está abierto. V, es un lock con espera activa

```
b=r1 ;
do
        compare&swap(r2,b,j) ;      //si r2==k , k y b se intercambian
while(b==r3) ;
```

//tb aparece k en el compare en vez de j y misma veracidad

21. Un microprocesador multinúcleo no incluye memoria caché. F
22. En un microprocesador SMT (multihebra simultánea) , en un instante determinado se pueden enviar a ejecutar instrucciones de hebras diferentes en cada ciclo. V
23. en un microprocesador SMT(multihebra simultánea) se procesan varias hebras concurrentemente aunque en un instante determinado solo se pueden enviar a ejecutar instrucciones de la misma hebra. F
24. En el protocolo MESI para mantener coherencia de caché, una línea puede estar en el estado E en varias cachés del multiprocesador. F
25. En el protocolo MSI, si en la caché de un nodo N1 hay un bloque en estado M(modificado) y ese nodo detecta que otro procesador intenta leer un dato que está en ese bloque, el bloque pasa al estado I(no válido) en el nodo N1. F
26. En un multiprocesador NUM con 16 nodos , 4GB por nodo y líneas de 64 B, ¿entradas que tiene el directorio de memoria utilizado en cada nodo para mantener la coherencia de caché en un protocolo MSI sin difusión? 2^32 / 2^6 = 2^26 entradas
27. bits de lo anterior? -> 17
28. ¿qué valores se puede observar en R si el modelo de consistencia de memoria del computador donde están los procesadores que ejecutan estos códigos no respeta el orden W->W (sí respeta los demás) e inicialmente X=Y=0?

    //R es un registro del procesador donde se ejecuta P2 y x e y son direcciones de memoria compartida

```
P1:     X=2 ;                   P2:     R=1;
        Y=1 ;                           if(Y==1)R=X ;
```

    R=0 ; R=1 ; R=2
    Sí, respeta el orden W->W: R=1 , R=2

29. ¿Qué valores se observarían en el registro R del problema anterior si no se garantiza el orden W->R ? lo mismo (no cambia el orden de las escrituras en P1) R =1 o R = 2

30. ¿qué valores deben r1,r2 y r3 tener para que la secuencia de instrucciones siguiente implemente un cerrojo (lock(k)) en el que k=1 significa que el cerrojo está cerrado y 0 que está abierto?

```
b=r1 ;
do
        compare&swap(r2,b,k);        //si r2==k k y b se intercambia
while(b==r3) ;
```

r1=1,r2=0,r3=1

31. El código siguiente permite implementar un cerrojo (lock(k)) en el que k=1 significa que el cerrojo está cerrado y k=0 que está abierto:

```
        b=0,k=1 ;
        while(fetch_and_add(k,b)==1){} ;
```

F -> b debería ser igual a 1 y utilizar fecth_and_or(k,b)

32. En el protocolo MESI, si en la caché de un nodo N1 hay un bloque B en estado M(modificado), y ese nodo detecta que el nodo N2 intenta escribir en un dato del mismo bloque B, dicho bloque pasará al estado M(Modificado) en la caché del nodo N2, y de mantendrá en el estado M en la caché del nodo N1 tras actualizarse en la memoria principal. F: el bloque en N1 se invalida
33. En el protocolo MESI, si en la caché de un nodo N1 hay un bloque B en estado S(Compartido), y ese nodo detecta que otro procesador en el nodo N2 intenta escribir un dato que está en el bloque B(que no está en la caché de N2) , dicho bloque pasa al estado E(exclusivo) en la caché del nodo N2 y se mantendrá en el estado S(compartido) en la caché del N1. F: pasará al estado I en el nodo N1 y a M en el nodo N2
34. En el protocolo MESI, si en la caché de un nodo N1 hay un bloque B en estado S(Compartido), y ese nodo detecta que otro procesador en el nodo N2 intenta leer un dato que está en el bloque B , dicho bloque pasa al estado S(compartido) en la caché del nodo N2, y se mantendrá en el estado S en la caché del nodo N1.
    V: eso es precisamente lo que ocurre según el protocolo MESI
35. En el protocolo MSI de espionaje para la coherencia, si en la caché de un nodo N1 hay un bloque B en estado M(Modificado) y detecta que el nodo N2 intenta escribir en un dato del mismo bloque B, dicho bloque pasará el estado S(compartido) en las cachés de los nodos N1 y N2 tras actualizarse en la memoria principal. F
36. En el protocolo MSI de espionaje para la coherencia, si en la caché de un nodo N1 hay un bloque B en estado M (modificado) y detecta que el nodo N2 intenta leer un dato del mismo bloque B,dicho bloque pasará al estado S(compartido) en las cachés de los nodos N1 y N2 tras actualizarse en la memoria principal. V
37. En un multiprocesador NUMA con protocolo MSI basado en directorios de vector de bits completo NO puede haber una entrada en uno de los directorios con todos sus bits de copias en cachés a cero(no hay ninguna copia del bloque correspondiente en las cachés de la máquina) y el bit de estado del bloque en memoria igual a 0(estado

no válido en memoria) V: si el bloque no se ha llevado a una caché debe estar en estado válido en la MP
cuidado con esta pregunta que sale sin el “no” y sería falsa!!

38. En un multiprocesador NUMA con protocolo MSI basado en directorios de vector de bits completo NO puede haber una entrada en uno de los directorios con un único bit a uno (hay una copia del bloque correspondiente en una caché de la máquina) y el bit de estado del bloque en memoria igual a 0(estado no válido en memoria) F : sí podría haberlo. El bloque estaría en estado modificado en la caché y no sería coherente con la memoria
39. En un multiprocesador NUMA con protocolo MSI basado en directorios de vector de bits completo puede haber una entrada en uno de los directorios con un único bit a uno(hay ninguna copia del bloque correspondiente en las cachés de la máquina) y el bit de estado del bloque en memoria igual a 1(estado válido en memoria)
    V: el bloque estaría en estado compartido en la caché y sería coherente con la memoria
40. En un multiprocesador, el procesador P1 ejecuta las instrucciones
    1. while(Z==0){};
    2. r1=Y ;

    en paralelo con las instrucciones que ejecuta el procesador P2
    a. X=1
    b. Y=2
    c. Z=1

    Si el modelo de consistencia de memoria de un multiprocesador NO respeta el orden W -> R (sí respeta todos los demás), e inicialmente X=Y=Z=r1=0 (donde X,Y,Z son variables en memoria compartida y r1 es un registro de P1) , al final se podría tener r1=2
    V: estaría ejecutándose hasta que (c) se haya ejecutado y r1 se cargaría con el último valor almacenado en Y, que es 2. El orden W->W se respeta

41. En un multiprocesador, el procesador P1 ejecuta las instrucciones
    1. while(Z==0){};
    2. r1=Y ;

    en paralelo con las instrucciones que ejecuta el procesador P2
    a. X=1
    b. Y=2
    c. Z=1

    Si el modelo de consistencia de memoria de un multiprocesador NO respeta el orden W -> W (sí respeta todos los demás), e inicialmente X=Y=Z=r1=0 (donde X,Y,Z son variables en memoria compartida y r1 es un registro de P1) , al final se podría tener r1=0
    V: estaría ejecutándose hasta que (c) se haya ejecutado, pero (c) puede adelantar a (a) y (b) y r1 se cargaría con el último valor almacenado en Y, que podría haberse mantenido a 0 si no se ha ejecutado (b). Hay que tener en cuenta que el orden W->W no se respeta

42. En un multiprocesador, el procesador P1 ejecuta las instrucciones
    1. while(Z==0){};
    2. r1=Y ;

    en paralelo con las instrucciones que ejecuta el procesador P2
    a. X=1
    b. Y=2
    c. Z=1

    Si el modelo de consistencia de memoria de un multiprocesador NO respeta el orden W -> W (sí respeta todos los demás), e inicialmente X=Y=Z=r1=0 (donde X,Y,Z son variables en memoria compartida y r1 es un registro de P1) , al final SOLO se podría tener r1=2

    F:(1) estaría ejecutándose hasta que © se haya ejecutado, pero (c) puede adelantar a (a) y (b) y r1 se cargaría con el último valor almacenado en Y, que, por ejemplo, podría haberse mantenido a 0 si no se ha ejecutado (b). Hay que tener en cuenta que el orden W->W no se respeta
43. Los comercialmente denominados procesadores HT de intel (hyper-threading) son procesadores multihebra simultánea(SMT)

    V: HT es la denominación comercial de intel para sus microprocesadores multihebra simultánea
44. Los microprocesadores SMT(multihebra simultánea) pueden ejecutar varias instrucciones de una misma hebra en paralelo

    V: precisamente, pueden enviar varias instrucciones a ser ejecutadas en un ciclo. Incluso esas instrucciones pueden pertenecer a hebras diferentes
45. Si una línea de la caché del nodo N1 está en el estado M del protocolo MSI para mantener la coherencia de caché, el contenido de esa línea es coherente con su contenido en memoria principal

    F: no es coherente porque ha podido ser modificada
46. un procesador multinúcleo no incluye memoria caché F
47. En un microprocesador SMT(multihebra simultánea), en un instante determinado se pueden enviar a ejecutar instrucciones de hebras diferentes V
48. En un microprocesador SMT(multihebra simultánea) se pueden enviar a ejecutar varias instrucciones de una misma hebra en un instane determinado V
49. En el protocolo MESI para mantener la coherencia de caché una línea puede estar en el estado S solo en una de las cachés del multiprocesador F
50. En el protocolo MESI para mantener la coherencia de caché, una línea puede estar en el estado E en varias cachés del multiprocesador F
51. En el protocolo MESI para mantener la coherencia de caché, una línea puede estar en el estado E solo en una caché del multiprocesador V
52. En el protocolo MESI para mantener la coherencia de caché, una línea puede estar en el estado M solo en una caché del multiprocesador V
53. En el protocolo MSI, si en la caché de un nodo N1 hay un bloque B en estado M(modificado) y ese nodo detecta que otro procesador en el nodo N2 intenta leer un dato que está en el bloque B, dicho bloque pasa al estado I(no válido) en la caché del N1 y a M(Modificado) en N2. F

54. En el protocolo MSI, si en la caché de un nodo N1 hay un bloque B en estado M(modificado) y ese nodo detecta que otro procesador en el nodo N2 intenta leer un dato que está en el bloque B, dicho bloque pasa al estado I(no válido) en el nodo N1
    F
55. En el modelo de consistencia de liberación no se garantizan los órdenes W->W y W->R, pero sí los R->RW F
56. ¿Qué valor pondría en a para que la secuencia de instrucciones siguiente implemente un cerrojo (lock(k)) en el que k=1 significa que el cerrojo está cerrado y 0 que está abierto?

```
b=1;
a=0
do
        compare&swap(a,b,k) ;        //lectura mod-escritura atómica
while(b==1) ;
```

57. ¿Cómo implementaría un cerrojo (lock(k)) en el que k01 significa que el cerrojo está cerrado y 0 que está abierto con una primitiva del tipo test_&_set?
    while(test_&_set(k) == 1) {} ; // k se inicializa a 0

58. En el protocolo MSI de espionaje para la coherencia, si en la caché de un nodo N1 hay un bloque B en estado M (modificado) y detecta que el nodo N2 intenta escribir en un dato del mismo bloque B,dicho bloque pasará al estado M(modificado) en la caché del nodo N2, y al estado I en la caché del nodo N1 tras actualizarse en la memoria principal. V

// el número de entradas en el directorio coincide con el número de bloques o líneas de memoria del nodo

TEMA 4:

1.

Considere que en un mismo ciclo se decodifican las siguientes cuatro instrucciones (el índice i, i+1,... indica el orden en el que están en el código) y pasan a una ventana de instrucciones centralizada desde la que se emiten a las unidades funcionales. El procesador dispone de un buffer de reordenamiento (ROB) que también implementa el renombramiento.
Además, tiene DOS unidades de suma, UN multiplicador, y una unidad de CARGA de memoria, y puede emitir CUATRO instrucciones en el mismo ciclo, con emisión DESORDENADA:

Nº inst	Instrucción	Significado	1	2	3	4	5	6	7	8	9
i	ld r2, 0(r3)	r2←m(r3+0)	EX	EX	ROB	WB					
i+1	mul r4,r2,r1	r4←r2×r1			EX	EX	EX	EX	ROB	WB	
i+2	add r1,r1,r2	r1←r1+r2			EX	ROB				WB	
i+3	add r4,r3,r5	r4←r3+r5	EX	ROB							WB
Pregunta 5											
Pregunta 6											

1. Las instrucciones i e i+3 se podrían emitir en el mismo ciclo -> V

2. Las instrucciones i+1 e i+2 no se podrían emitir en el mismo ciclo porque entre ellas hay riesgo de tipo WAR -> F: los riesgos WAR desaparecen con el renombramiento que se hace en el ROB

3. El compilador puede evitar el riesgo WAW entre las instrucciones i+1 e i+3 utilizando el registro r6 en lugar de r4 en la instrucción i+1 -> V

4. En el procesador que se ha descrito al comienzo, aunque la emisión es desordenada, la finalización de instrucciones es ordenada -> V: hay un ROB que precisamente permite que la finalización sea ordenada

5. Si en el procesador descrito la SUMA tiene un retardo de un ciclo, la CARGA de memoria un retardo de dos ciclos, y la MULTIPLICACIÓN un retardo de cuatro ciclos ¿cuántos ciclos tardaría en EJECUTARSE la secuencia de cuatro instrucciones anterior? (Contando desde el ciclo en que termina de ejecutarse la primera de las cuatro instrucciones) -> 4 ciclos (ver tabla)

6. Suponiendo que el ROB puede retirar dos instrucciones por ciclo. ¿En cuántos ciclos se retirarían las cuatro instrucciones anteriores (contando desde el ciclo en el que se retira la primera de las cuatro instrucciones)? -> 5 ciclos (ver tabla)

7. Las instrucciones de movimiento condicional de datos permiten reducir el número de instrucciones de salto condicional en los códigos -> V

8. La predicción de saltos dinámica implícita para una instrucción de salto condicional, i, utiliza el resultado (salto o no salto) de la ejecución previa de dicha instrucción de salto condicional, i, para hacer la predicción -> V

9. Los procesadores WLIW no tienen buffers de renombramiento porque la planificación de instrucciones la realiza el compilador -> V

10. Considere las dos instrucciones siguientes(la instrucción i precede a la i+1 en el código)

    ```
    (i)     sw    0(r5),r2      //m(r5+0) <- r2
    (i+1)   ld    r4,0(r6)      //r4 <- m(r6+0)
    ```

    Un procesador que NO implemente adelantamiento ESPECULATIVO de LOADs a STORIES podría adelantar la ejecución de i+1 a la de i -> F: debe haber recursos para el adelantamiento especulativo porque podría ocurrir que r5 sea igual a r6 y se violaría el riesgo RAW si se produce el adelantamiento

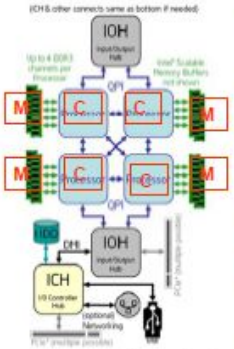
    UN EXAMEN T3 Y 4:

Un microprocesador multinúcleo no incluye memoria cache

(F)

El servidor de la figura tiene una arquitectura de tipo UMA

(F)



Señale en la figura (con una C) dónde está la memoria cache

Señale en la figura (con un M) dónde está la memoria principal

En un microprocesador SMT (Multihebra simultánea), en un instante determinado se pueden enviar a ejecutar instrucciones de hebras diferentes

(V)

En un microprocesador SMT (multihebra simultánea), se procesan varias hebras concurrentemente aunque en un instante determinado solo se pueden enviar a ejecutar instrucciones de la misma hebra.

(F)

En el protocolo MESI para mantener la coherencia de cache una línea puede estar en el estado S solo en una de las caches del multiprocesador

(F)

En el protocolo MESI para mantener la coherencia de cache, una línea puede estar en el estado E en varias caches del multiprocesador

(F)

En el protocolo MSI, si en la cache de un nodo N1 hay un bloque en estado M (Modificado), y ese nodo detecta que otro procesador intenta leer un dato que está en ese bloque, el bloque pasa al estado I (Inválido) en el nodo N1

(F)

Suponga que en un mismo ciclo se decodifican estas cuatro instrucciones (el número entre paréntesis indica en orden en el que están en el código), y pasan a una ventana de instrucciones única desde la que se emiten a las distintas unidades funcionales. El procesador tiene un buffer de reorden (ROB) para la finalización ordenada donde también se implementa el renombrado.

(i)	add r2, r2, r1	// r2 = r2+r1
(i+1)	mul r3, r4, r1	// r3 = r4*r1
(i+2)	add r4,r3,r1	// r4 = r3+r1
(i+3)	add r1, r2, r1	// r1 = r2+r1

1. Si el procesador tiene TRES unidades de suma y UN multiplicador y puede emitir cuatro instrucciones por ciclo con emisión DESORDENADA, se podrían emitir las cuatro instrucciones en el mismo ciclo. -> F: hay dependencias RAW
2. Entre la instrucción (i+1) y la (i+2) SOLO hay un riesgo de tipo WAR -> F: también hay dependencia RAW en r3 aparte de la WAR en r4
3. Si el procesador descrito anteriormente puede emitir (con emisión desordenada) DOS instrucciones por ciclo y tiene DOS unidades de suma (con un retardo de 2 ciclos) y UN multiplicador (con un retardo de 5 ciclos). ¿Cuántos ciclos tardan en emitirse las cuatro instrucciones (cuente a partir del ciclo de la decodificación, sin incluir

este) ?

	1	2	3	4	5	6	7	8	9
(i)	EX	EX	ROB	WB					
(i+1)	EX	EX	EX	EX	EX	ROB	WB		
(i+2)						EX	EX	ROB	WB
(i+3)			EX	EX	ROB				WB

Tardan 5 ciclos en emitirse las instrucciones y 9 en retirarse todas (suponiendo que se puede retirar más de dos instrucciones por ciclo)

4. En la predicción de salto dinámica implícita , cada vez que llega una instrucción de salto condicional se predice que NO se va a producir el salto.
   F : es dinámica, luego debe cambiar según lo que ocurre en la ejecución del código. En este caso, se hace lo que se hizo en la última ejecución.
5. Indique cómo expresaría sin instrucciones de salto utilizando un repertorio típico de instrucciones con predicado en el que TODA instrucción se puede predicar(suponga que los predicados están inicializados a 0):
   if(r2==0) then r2=r2+r3 else r2=r2-r3 ;

   p1,p2 cmp. eq r2,0
   (p1) add r2,r2,r3
   (p2) sub r2,r2,r3
6. NO existen instrucciones de ejecución condicional en los repertorios de instrucciones de procesadores superescalares F
7. La segmentación software NO necesita un hardware especial de apoyo V
8. La responsabilidad del aprovechamiento del paralelismo entre instrucciones (ILP) en un procesador VLIW recae fundamentalmente en el compilador V
9. Los procesadores superescalares no pueden tener un repertorio de instrucciones CISC, por eso todos los núcleos de Intel son VLIW F
10. Considere las dos instrucciones siguientes: (i precede a i+1 en el código)

    (i) sw 0(r5),r2 //M(r5) <- r2
    (i+1) lw r4,32(r5) //r4 <- M(r5+32)

    Para que un procesador adelante la ejecución de i+1 a la de i, DEBE IMPLEMENTAR adelantamiento de loads a stores ESPECULATIVO:
    F la dirección la que se hace el almacenamiento, r5+0, siempre va a ser distinta de la dirección desde donde se lee, r5+32

VÍDEO MANCIA:

1. Suponer que un procesador ideal que ejecuta cada instrucción en T segundos se segmenta en cuatro etapas ideales de duración T/4. Con ello se consigue que:
    a) una instrucción se ejecute en T/4 segundos
    b) 4 instrucciones se ejecuten en 4T segundos
    c) cada T/4 segundos se termine de ejecutar una instrucción
    d) cada 4T segundos se terminen de ejecutar 4 instrucciones
2. Si representamos la fase Decode con una D , Execute con una E , Fetch con una F y Writeback con una W , el orden correcto de las distintas fases de una instrucción máquina es:
    a) F D E W
    b) D F E W
    c) D E F W
    d) F W D E
3. La ganancia en velocidad ideal de un cauce de K etapas de igual duración T ejecutando un programa de N instrucciones es
    a) S=NKT/(N-K+1)T
    b) S=NT/(N+K-1)T
    c) S=KN/(K-N+1)
    d) S=KN/(K+N-1)

    // si n tiende a infinito, quedaría k -> que son las etapas