Enero 2017

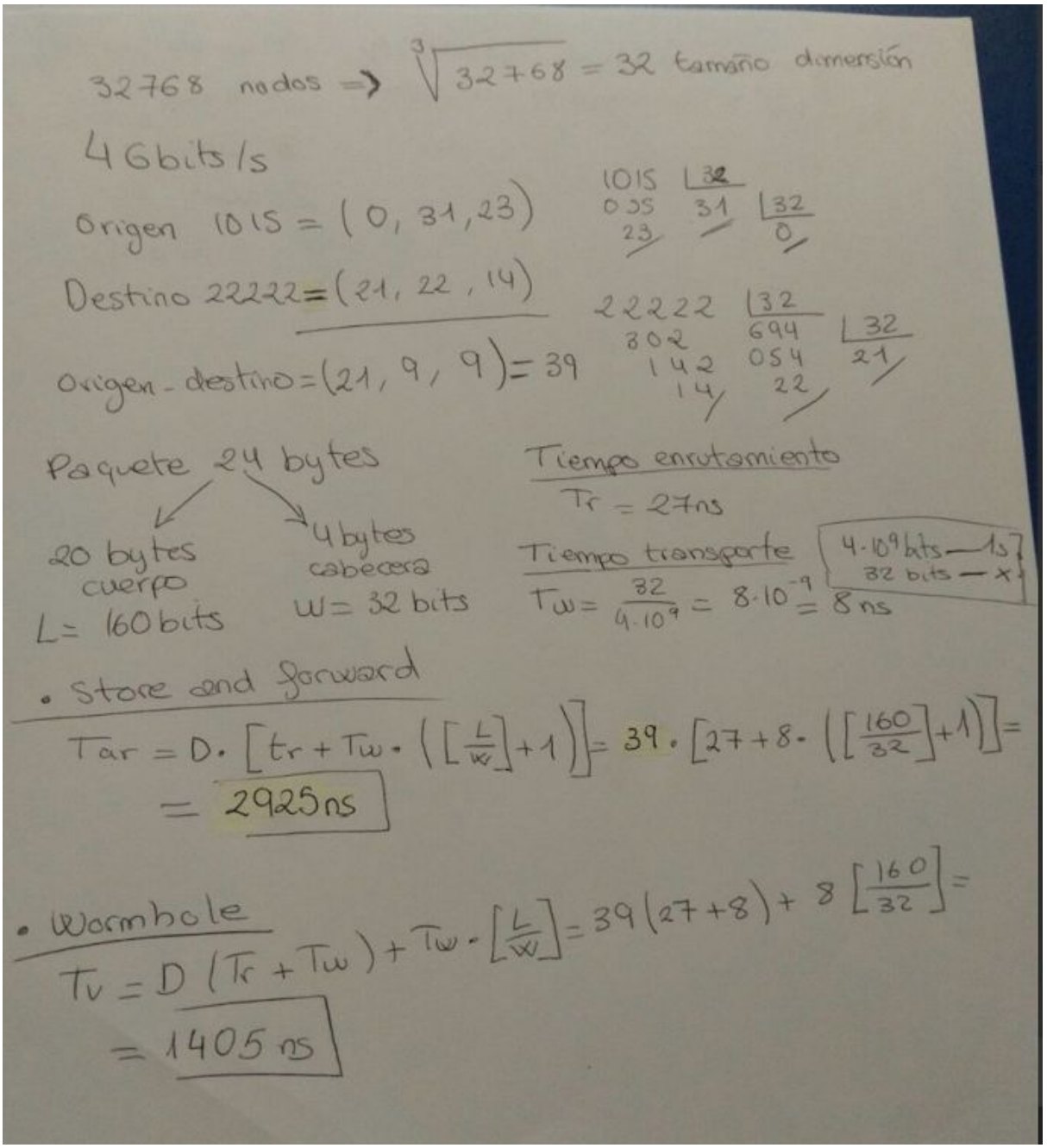
Pregunta 1 (1.5 ptos). Contesta a las siguientes preguntas:

1. (1 pto) Explica qué dos tipos de buffer de renombrado podemos encontrar y cuales son las ventajas de cada tipo.
    a. ***Buffer de renombramiento con acceso asociativo****: cada línea de buffer tiene cinco campos : asignación válida,registro de destino, contenido,contenido válido y bit de asignación última.*
        i. ***Asignación válida:*** *indica si la línea en cuestión se ha utilizado para nombrar algún registro, es decir , si los restantes campos tienen información válida.*
        ii. ***Registro de destino:*** *se indica el número de registro de la arquitectura que se ha renombrado utilizando la línea en cuestión.*
        iii. ***Contenido:*** *almacenará los datos correspondientes al registro que se ha renombrado hasta que esos datos se actualicen en el registro de la arquitectura correspondiente.*
        iv. ***Contenido válido:*** *se utiliza como bit de validez del contenido,indicando, si está a cero, que alguna instrucción que se ha emitido o enviado a escribir su resultado en el campo del contenido de linea.*
        v. ***Bit de asignación última:*** *está a 1 en la línea del buffer de renombramiento en la que se haya hecho la última asignación a un registro dado.*
    b. ***Buffer de renombramiento con acceso indexado:*** *cada registro de la arquitectura existe un índice que apunta a la línea del buffer que se utiliza para renombrar ese registro.Junto con ese índice también existe un campo de asignación válida que indica si se ha hecho o no el renombramiento(y el contenido del campo de índice es válido).El buffer de renombramiento propiamente dicho únicamente tiene el campo de contenido y el de contenido válido.*

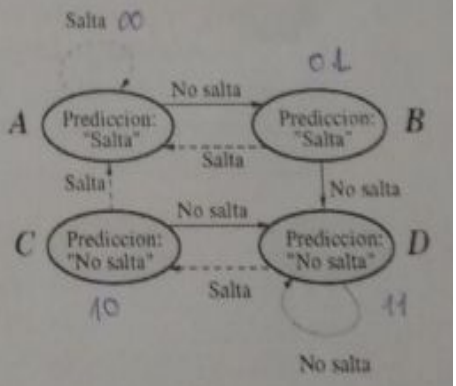
2. (0.5 ptos) Explica cómo se realiza el encaminamiento en una red CCC(no hace falta dibujarla).

Pregunta 2 (2 ptos). Una universidad ha adquirido un supercomputador formado por **32.768 nodos** conectados mediante una red **malla abierta 3D** cuyos enlaces tienen una velocidad de **4Gbit/s**.Para terminar de analizar el rendimiento del supercomputador se desea saber cuánto tardará un paquete formado por **24 bytes (incluyendo la cabecera)** que se envía desde el **nodo 1015** al **nodo 22.222**. El tiempo de enrutamiento es de **27ns**.Calcula los tiempos de envío tanto utilizando “”store and forwarding” como “wormhole”.

Nota: la cabecera del paquete está formada por 4 bytes.



Pregunta 3 (2,5 ptos) Suponer un computador superescalar que dispone un buffer de reorden, que permite resolver los riesgos WAR y WAW, y una ventana de instrucciones con un número de entradas suficiente. El procesador es capaz de decodificar, emitir y completar 3 instrucciones por ciclo. Además, la emisión de las instrucciones puede ser desordenada y dispone de unidad de adelantamiento. Para las tareas de ejecución, se dispone de las siguientes unidades segmentadas: 2 FP mul/div (5c), 2 FP add (2c), 2 ALU int (1) y 2 load/store (3). Finalmente, se dispone de un predictor de saltos dinámico que utiliza BTB de 4 entradas y 2 bits de predicción. Cuando se añade una nueva entrada en el BTB, su primera predicción siempre sería de estado A (salto efectivo).



En el computador se ejecuta el siguiente fragmento de programa:

```
; r1 almacena la dirección de a
; r2 almacena la dirección de b
      addi r3,r1,#80 ; condicion de final
      addi r1,r1,#8 ; inicialización de los indices
      addi r2,r2,#8 ;
      addi r5,r1,#3;
      ld f0,coef ; cargar coeficiente
loop: ld f2,-8(r1) ; cargar a[i-1]
      ld f4,0(r1) ; cargar a[i]
      beqz r5, fin
      muld f8,f2,f0 ; a[i-1]*coef
      divd f9,f2,f0 ;
      addd f4,f8,f4 ; a[i-1]*coef + a[i]
      sd 0(r2),f4 ; almacenar b[i]
      addi r1,r1,#8 ; incrementar indices
      addi r2,r2,#8
      subi r5,r5,#1
      slt r4,r1,r3
      bnez r4,loop
fin:  subd f2,f1, f3
```

r1 = 0
r2 = 100
r5 = 2 → si == 0 fin r5 = 1
r1 == 16
set less than
si 1 = 0 loop

Página 1 de 2

a)(1.5 ptos) Planificar las instrucciones utilizando una tabla como la siguiente hasta la primera iteración del bucle (sin realizar el salto).Suponer que inicialmente r1=0 y r2=100.
Enlace al excel donde está solucionado.
b)(0.5 ptos) Realizar una traza de ejecución del código, mostrando el contenido de la BTB(BTB inicialmente vacía) para todas las iteraciones del bucle.

Iterración 1

dir salto	dir destino	bit de predicción
beqz r5, fin	fin	A
bnez r4, loop	loop	A

Iterración 2

dir salto	dir destino	bit de predicción
beqz r5, fin	fin	B
bnez r4, loop	loop	A

Iterración 3

dir salto	dir destino	bit de predicción
beqz r5, fin	fin	D
bnez r4, loop	loop	A

Iterración 4

dir salto	dir destino	bit de predicción
beqz r5, fin	fin	D
bnez r4, loop	loop	A

Iterración 5

dir salto	dir destino	bit de predicción
beqz r5, fin	fin	D
bnez r4, loop	loop	A

Iterración 6

dir salto	dir destino	bit de predicción
beqz r5, fin	fin	D
bnez r4, loop	loop	A

Iterración 7

dir salto	dir destino	bit de predicción
beqz r5, fin	fin	D
bnez r4, loop	loop	A

Iterración 8

dir salto	dir destino	bit de predicción
beqz r5, fin	fin	D
bnez r4, loop	loop	A

Iterración 9

dir salto	dir destino	bit de predicción
beqz r5, fin	fin	D
bnez r4, loop	loop	B

c) (0,25 ptos) ¿Existe alguna penalización en la ejecución del código? Si es así,indica con qué instrucción y cuando.

*Si, con la instrucción **beqz r5,fin** ya que en la primera iteración la predicción por defecto es que salta pero esta instrucción no salto y en la segunda iteración es aun estando en el estado B la predicción para este estado es saltar pero la instrucción no salta .*

d)(0,25 ptos) Determinar el número de ciclos que tardaría en ejecutarse el código

Interacción 1 : 19 ciclos

Interacción 2 - 8 : 18 ciclos

interacción 9 : 11 ciclos

***total** : x ciclos*

Pregunta 4 (1 pto) Explica la diferencia entre predicción dinámica explícita y predicción dinámica implícita.

***Predicción dinámica implícita:**si no se guarda ninguna información explícita que represente el comportamiento pasado de la instrucción. Se almacena únicamente la dirección de la instrucción que se ejecutó tras la instrucción de salto la última vez que se captó esta.La dirección puede ser la dirección de destino del salto, lo que equivale a predecir que se produce el salto, o bien la dirección de la instrucción siguiente a la de salto, con lo que se predice que el salto no se produce.*

***Predicción dinámica explícita:** para cada instrucción de salto condicional, existe un conjunto de bits que codifican la información relativa al comportamiento pasado de la instrucción en cuestión.Esos bits se denominan bits de historia.El número de bits de historia que se guardan para cada instrucción depende del tipo de esquema de predicción dinámica explícita que se haga.*

Pregunta 5 (1.75 ptos). En la paralelización de una aplicación orientada a una máquina paralela de memoria distribuida un ingeniero ha descompuesto dicha aplicación en 12 tareas, que nombramos como T1,T2...T12, donde el subíndice indica el orden de ejecución de cada tarea en la versión secuencial. Seguidamente nos indica que los grupos de tareas T2 a T6 (ambas incluidas) y T8 a T10(ambas incluidas) son independientes entre sí.

Por último, nos informa del porcentaje de tiempo que toma cada tarea por separado en la versión secuencia(ver tabla adjunta). Suponga que disponemos de 2 multicomputadores con 2 y 4 nodos respectivamente(todos los nodos iguales), conectados entre sí con una red cuya sobrecarga puede modelar com T_overhead(p) = 0.05*p(donde p es el número de procesadores).

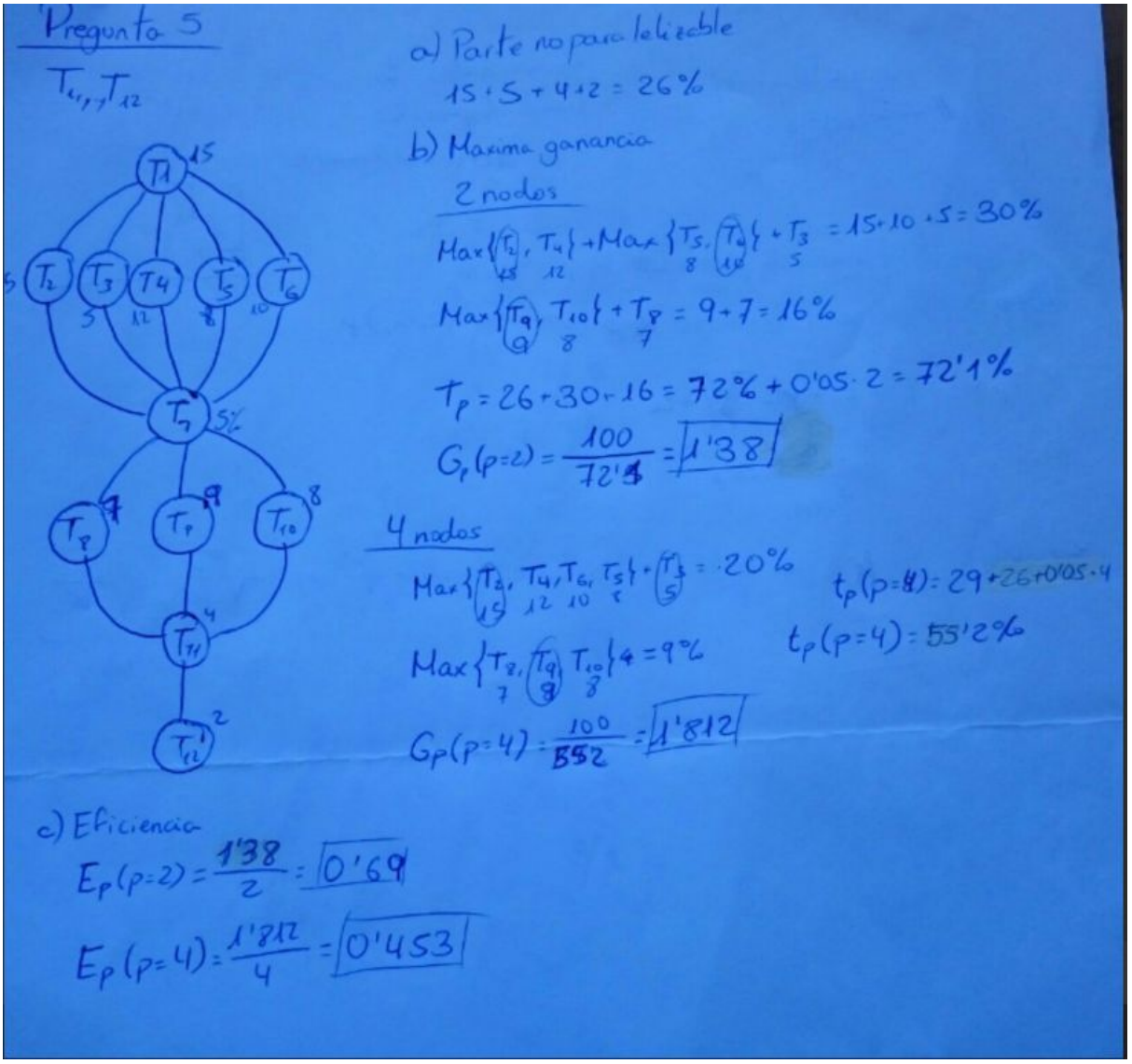
T1=15%,T2,=15%,T3=5%,T4=12%

T5=8%,T6=10%,T7=5%,T8=7%

T9=9%,T10=8%,T11=4%,T12=2%

VERDE: GRUPO 2
ROJO: GRUPO 1

Se pide:
a)Dibuje el grafo de precedencia entre tareas calcule la fracción no paralelizable del problema **(0.25 ptos)**
b)¿Cuál es la máxima ganancia en velocidad que se puede obtener con cada uno de los multicomputadores ? **(1 pto)**
c) ¿En qué cluster es más eficiente la ejecución de nuestra aplicación con la paralelización propuesta? **(0.5 ptos)**

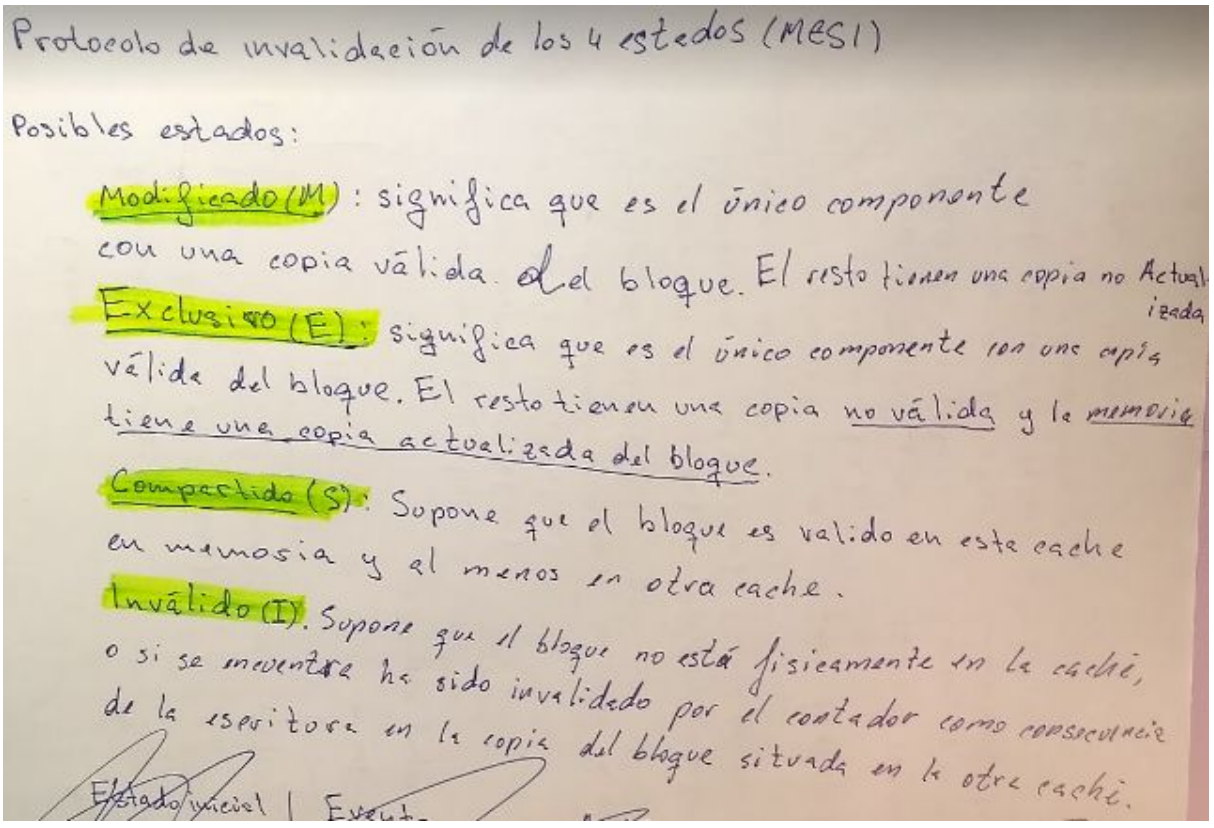


Nota: Indique claramente cuál es la asignación de las tareas a los distintos nodos que maximiza la ganancia en velocidad. Explique clara y pormenorizadamente cada paso que dé en la resolución del problema.

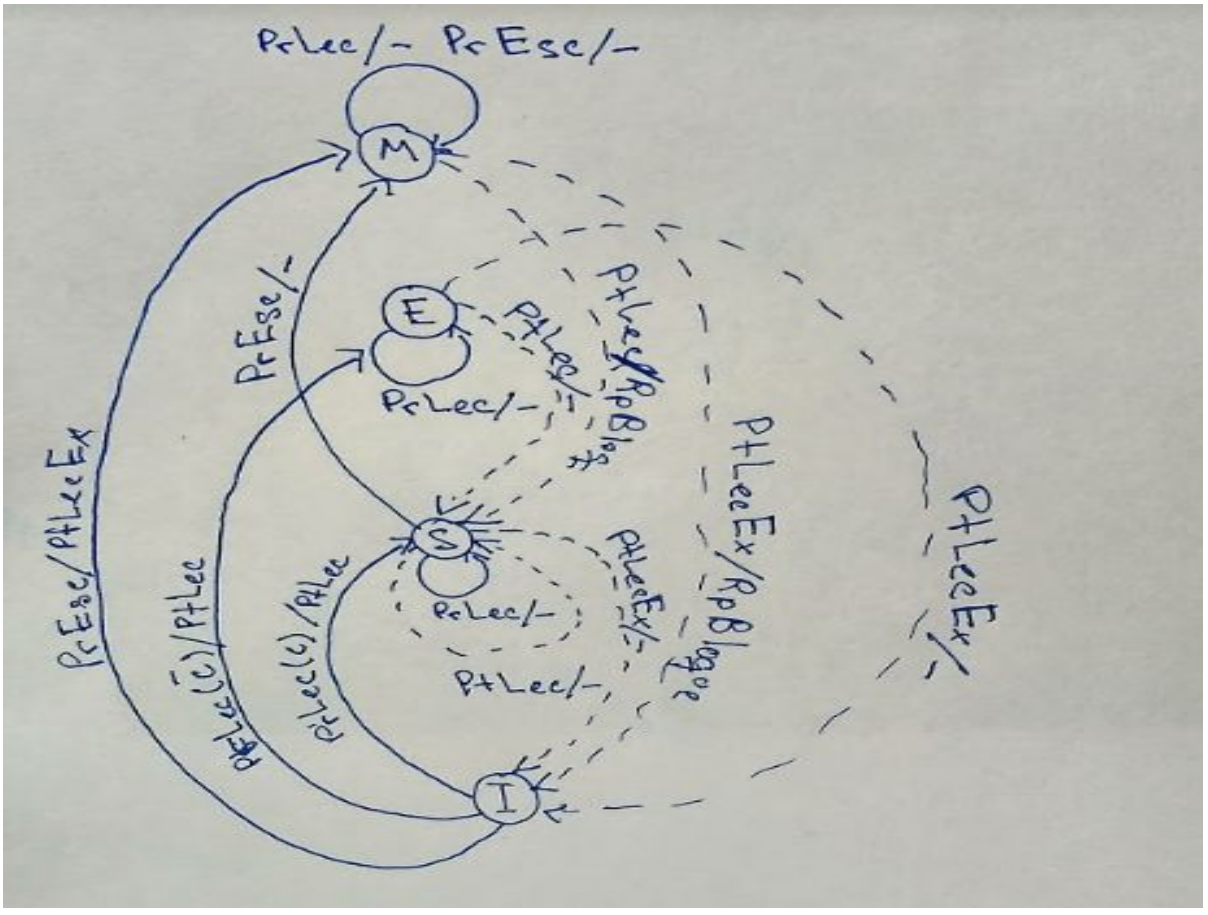
Pregunta 6 (1.25 ptos)
Una cierta línea de caché de uno de los nodos de un multiprocesador equipado con un sistema de caché que implementa el **protocolo MESI** pasa al estado **I.**
Indique todo lo que podemos saber(mecanismo ha llevado a la línea a cambiar de estado, porqué, posible estado o estados previos,...) y qué consecuencias tiene cambiar al estado **I** (¿en qué casos se ha de compartir y con quién el contenido de dicha línea?).

*Lo que podemos saber es lo siguiente: se ha producido un **fallo de escritura al no estar el bloque en caché**,es decir, el procesador escribe(PrEsc) y el bloque no está en caché. El controlador de caché del procesador que escribe donde un paquete de petición de acceso exclusivo al bloque(PtLecEx).El estado del bloque en la caché después de la escritura será de modificado.El paquete PtLecEx provoca los siguientes efectos.*

1. *Si una caché tiene el bloque en estado modificado, bloquea la lectura de memoria y deposita el bloque en el bus. El bloque pasa en esta caché a estado inválido.*
2. *Si una caché tiene el bloque en estado exclusivo o compartido, pasa a estado inválido.*



Estado	Evento	Acción	Siguiente Estado
Modificado (M)	Procesador lee (PrLec)	:	: M
	Procesador escribe (PrEsc)	:	: M
	Paquete de lectura (PtLec)	: Genera paquete respuesta bloque (RpBloque)	: S
	Paquete de acceso exclusivo al bloque (PtLecEx)	: Genera paquete respuesta bloque (RpBloque) Invalida copia local	: I
	Reemplazo	: Genera paquete posescritura bloque (PtPEsc)	: I
Exclusivo (E)	Procesador lee (PrLec)	:	: E
	Procesador escribe (PrEsc)	:	: M
	Paquete de lectura (PtLec)	:	: S
	Paquete de acceso exclusivo al bloque (PtLecEx)	: Invalida la copia local	: I
Compartido (S)	Procesador lee (PrLec)	~~Genera paquete PtLecEx~~	: S
	Procesador escribe (PrEsc)	: Genera paquete (PtLecEx)	: M
	Paquete de lectura (PtLec)	:	: S
	Paque exclusivo de acceso al bloque (PtLecEx)	: Invalida copia local	: I
Inválido (I)	Procesador lee (PrLec)	: (Copia en otras caches C=1) Genera paquete PtLec	: S
	Procesador lee (PtLec)	: (no hay copias en otras caches C=0) Genera paquete PtLec	: E
	Procesador escribe (PrEsc)	: Genera paquete PtLecEx	: M
	Paquete de lectura (PtLec)	:	: I
	Paquete de acceso exclusivo al bloque (PtLecEx)	:	: I



		3 ins/c			3 ins/c				
instrucción	**if**	**id/iss**	**ex**	**rob**	**wb**	**comentario**	**decodifica, completar,emitir 3 inst/c**		
addi r3,r1,#80	1	2	3	4	5	**alu1**	**emisión desordenada**		
addi r1,r1,#8	1	2	3	4	5	**alu2**	**finalización ordenada ya que tiene rob**		
addi r2,r2,#8	1	2-3	4	5	6	**alu1** /*al estar todas las alus ocupadas tengo que esperar*/	**con adelantamiento**		
addi r5,r1,#3	2	3	4	5	6	**alu2**	**2FP mult/div (5c)**		
ld f0,coef	2	3	4-6	7	8	**load/store 1**	**2FP add (2c)**		
loop:ld f2,-8(r1)	2	3	4-6	7	8	**load/store 2**	**2FP ALU int(1c)**		
ld f4,0(r1)	3	4-6	7-9	10	11	**load/store 1** /*todas las load/store estan ocupadas hay que esperar*/	**2FP load/store (3c)**		
beqz r5,fin	3	4	5	6-10	11	**alu1**			
muld f8,f2,f0	3	4-6	7-11	12	13	**mult/div 1 /***tiene que esperar a que f2 se libre y obtenrla con adelantamiento*/			
divd f9,f2,f0	4	5-6	7-11	12	13	**mult/div 2** /*ya que el adelantamiento ya esta en uso en el ciclo 8 tengo que esperar un ciclo mas*/			
addd f4,f8,f4	4	5-11	12-13	14	15	**add 1 /***espera a que se libere f8 y obtiene con adelantamieto*/			
sd 0(r2),f4	4	5-13	14-16	17	18	**load/store 1 /***espera a que se libre f4 de la instrucción anterior*/			
addi r1,r1,#8	5	6	7	8-17	18	**alu1**			
addi r2,r2,#8	5	6	7	8-17	18	**alu2**			
subi r5,r5,#1	5	6-7	8	9-18	19	**alu1** /*todas las alus estan ocupadas hay que esperar*/			
slt r4,r1,r3	6	7	8	9-18	19	**alu2**			
bnez r4,loop	6	7-8	9	10-18	19	**alu1 /*** todas las alus ocupadas hay que esperar*/			
fin: subd f2,f1,f3	x								