P1.- Al ejecutar un programa en un procesador que tiene una frecuencia de 100 MHz, se obtiene un rendimiento en MIPS = 25. Un análisis del código ejecutado revela que el programa tiene la distribución de instrucciones indicada en la siguiente tabla. El CPI de las instrucciones ejecutadas en la unidad INT es 4, se desconoce el CPI de las instrucciones ejecutadas en la unidad CF y el resto de instrucciones tienen un CPI de 5.

	Unidad INT	Unidad CF	Resto instrucciones
%Instrucciones	50	25	25
CPI	4	¿?	5

a) ¿Qué CPI se ha conseguido para las instrucciones de CF?

MIPS= f / (CPI x 10^6)
CPI = f / (MIPS x 10^6) = 100 x 10^6 /(25 x 10^6) = 4

CPI = 0,5 x CPI_{INT} + 0,25 x CPI_{FP} + 0,25 x CPI_{Resto}
4 = 0,5 x 4 + 0,25 x CPI_{FP} + 0,25 x 5

CPI_{FP} = 3

P2.- Suponga que en el problema anterior el CPI de las instrucciones ejecutadas en CF es 10. Explique en términos de los parámetros Fm, Am y Ag, que se describen en la ley de Amdahl, que opción sería mejor:

a) Aplicar una mejora parcial, que consiga un CPI=1 en la unidad INT.
b) Aplicar una mejora parcial, que consiga un CPI=4 en la unidad CF.

Mejora INT, Fm = 0,5x4 /(0,5 x4 + 0,25 x 10 + 0,25 x5) = 2 / 5,75 =0,3478
Am= 4/1 =4
Ag = 1/ ((1-Fm) + Fm/Am) = 1/ ((0,6522 + 0,3478/4) = 1/0,739 =1,35

A= (5,75)/(0,5 x1 + 0,25 x 10 + 0,25 x5) = 5,75/ 4,25 = 1,35

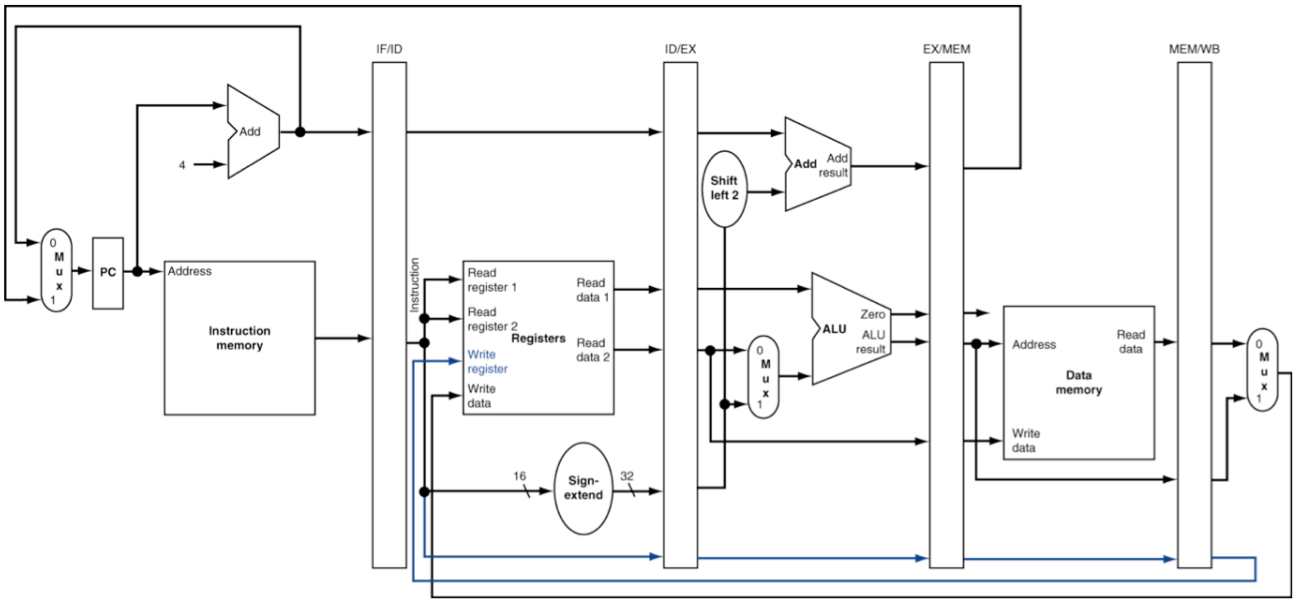
Mejora CF, Fm = 0,25x10 /(0,5 x4 + 0,25 x 10 + 0,25 x5) = 2,5 / 5,75 =0,4348
Am= 10/4 =2,5
Ag = 1/ ((1-Fm) + Fm/Am) = 1/ ((0,5652 + 0,4348/2,5) = 1/0,739 =1,35

A= (5,75)/(0,5 x4 + 0,25 x 4+ 0,25 x5) = 5,75/ 4,25 = 1,35

Ambas mejoras son equivalentes.

P3.- Se ejecuta una instrucción **STORE** en un procesador segmentado de 5 etapas, como el descrito en el libro de Patterson y Hennessy. **Señale sobre la figura** los recursos utilizados durante las etapas 3, 4 y 5.

En la etapa 3 (**EX)** se realizan operaciones en la ALU, se calcula la dirección efectiva en accesos a memoria, y se evalúa la condición en saltos. En la etapa 4 (**MEM)** se accede a memoria de datos y se realiza la actualización del PC en saltos y en la etapa 5 (**WR)** se escribe en el banco de registros**.**



Nota: Remarque sobre la figura tanto los elementos que se activan como las líneas de conexión que estén siendo utilizadas.

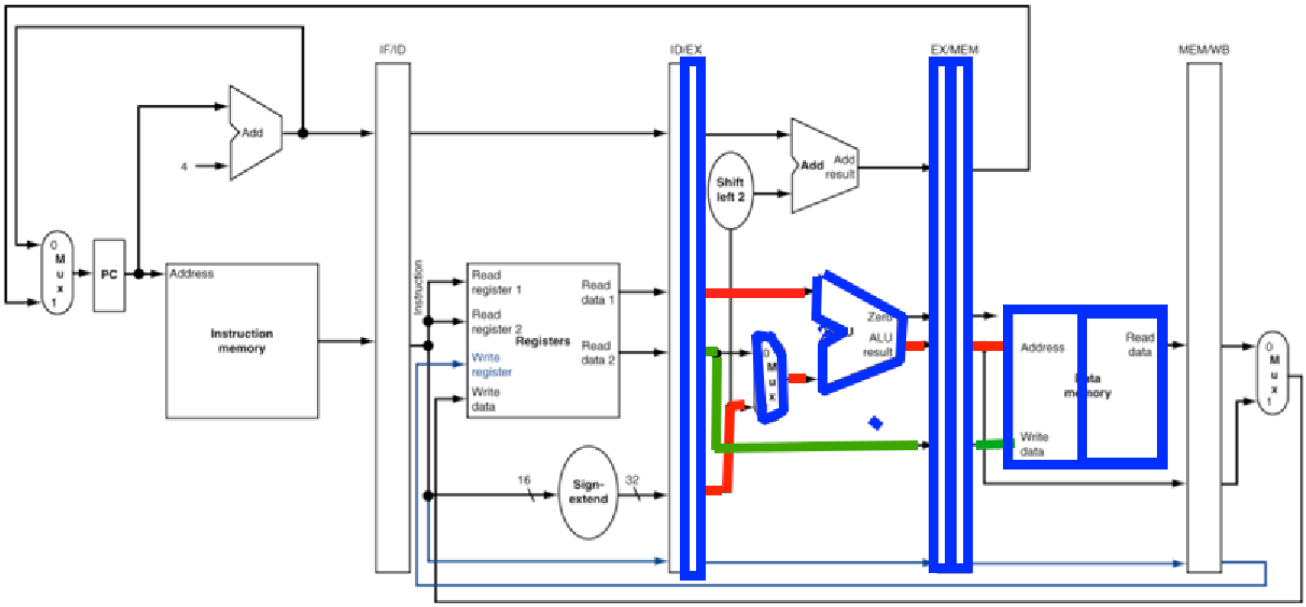
SOLUCION:

ST R1, 64(R0) ; Mem (0x40) <= R1

En verde se marcan las líneas utilizadas para llevar el contenido de R1

En rojo las líneas utilizadas para calcular la dirección de acceso a memoria = 0x40

En azul los recursos utilizados: ALU, memoria de datos, multiplexor para seleccionar el segundo operando de la ALU y registros de segmentación: ID/Ex se lee, Regs Ex/Mem se lee y se escribe.



P4.- Para un BTB de 8 entradas, con el contenido que se muestra en la tabla, y que corresponde a una situación anterior a la ejecución del salto en el siguiente fragmento de código:

```
1000 003C    LOAD R4, 64(R0)        ; 64 en decimal = 0x40 hexadecimal
1000 0040    BEQ  R1, R3, 7         ; 7 en decimal = 0x07 hexadecimal
1000 0044    ADD  R1, R2, R5
1000 0048    SUB  R2, R1, R0
…
1000 0058    MUL  R3, R1, R4
1000 005C    SUB  R2, R12, R1
1000 0060    ADD  R1, R3, R0
1000 0064    ADD  R4, R2, R5
```

Contenido del BTB:

0000 1000	1000 003C	11
0000 1010	1000 0040	01
1000 0040	1000 0060	~~10~~ 11
1000 1010	1000 0040	10
		01
		01
		01
		01

Nota: las entradas en blanco están vacías.

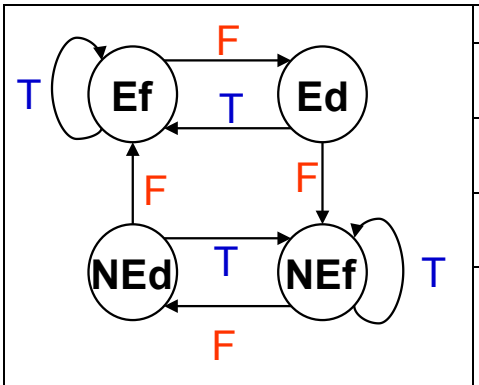
a) Explique qué contenido se guarda en cada columna del BTB.

La primera columna guarda la dirección que corresponde a instrucciones de salto.
La segunda columna guarda la dirección destino de los saltos.
La tercera columna guarda la predicción a realizar (estado de la FSM).

b) ¿Qué predicción realiza el BTB para el salto BEQ R1, R3, 7?
La entrada del BTB correspondiente a este salto es la tercera y realiza predicción efectiva (10)

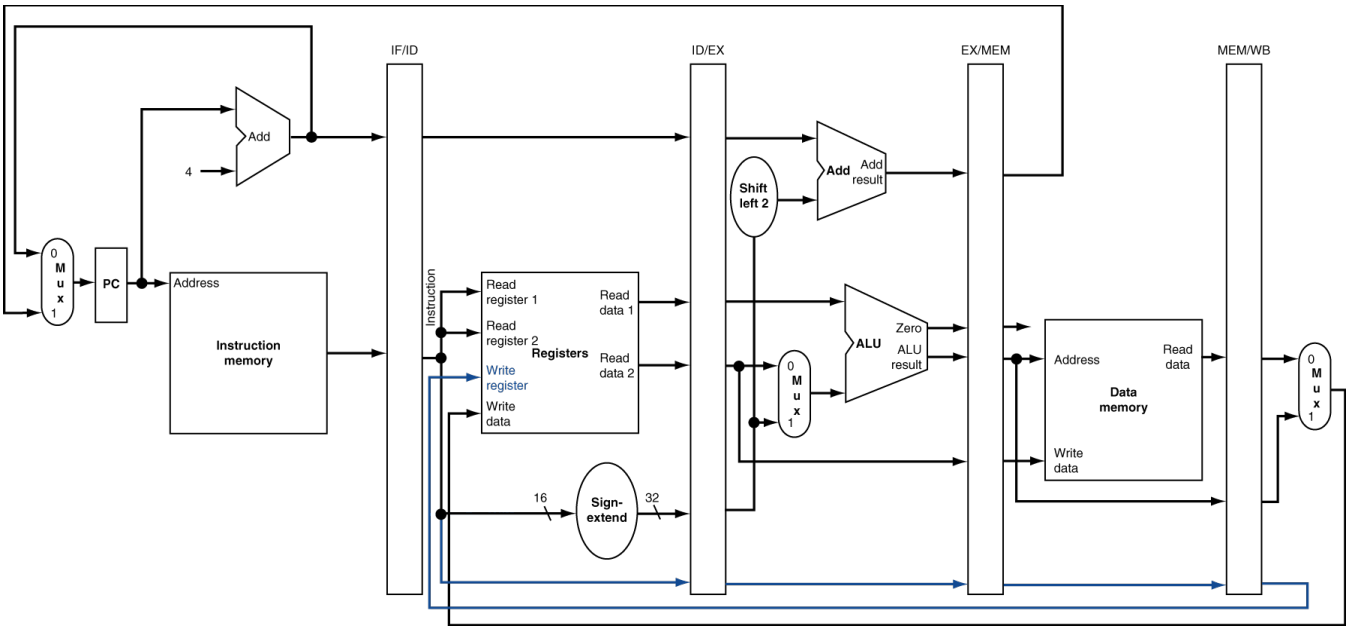
c) Indique (sobre la misma figura) cómo se modifica el contenido del BTB después de ejecutar el salto, suponiendo que el salto ha sido efectivo y que la evolución del sistema de predicción es la indicada en la máquina de estados.

Nota: Si el salto no está en el BTB se realiza predicción estática no efectiva (similar al estado inicial).



Estados	Codificación	Predicción	
Efectivo fuerte (Ef)	11	Efectiva (Salta)	
Efectivo débil (Ed)	10	Efectiva (Salta)	
No Efectivo débil (NEd)	01	No Efectiva (No salta)	Estado inicial
No Efectivo fuerte (NEf)	00	No Efectiva (No salta)	

P5- La siguiente secuencia de código se ejecuta en un sistema con arquitectura Harvard y un procesador segmentado con 5 etapas como el descrito en el libro de Patterson y Hennessy: **F** (captura de instrucciones), **D** (decodificación, detección de riesgos y lectura de registros), **E** (opera en la ALU, calcula la dirección efectiva, evalúa la condición en saltos), **M** (acceso a memoria de datos y actualización en saltos del PC) y **W** (escritura en el banco de registros). El banco de registros permite escritura y lectura en el mismo ciclo.



Analice el funcionamiento del siguiente programa, empezando en la captura de la instrucción I1 y suponiendo que el salto (instrucción I4) es efectivo (salta al destino).

```
I1 ADD R1, R2, R3          ; R1 = R2+R3
I2 LOAD R4, 0(R1)          ; R4 = M[0+R1]
I3 MUL R5, R4, R4          ; R5 = R4·R4
I4 BNE R1, R4, L1          ; Si R1 /= R4, salta a L1 (sí se salta)
I5 SUB R5, R5, R2          ; R5 = R5-R2
I6 ADD R7, R4, R5          ; R7 = R4+R5
I7 / L1: ADD R6, R1, R5    ; R6 = R1+R5
I8: OR R1, R7, R8          ; R1 = R7 or R8
I9: ST R6, 0(R1)           ; M[0+R1] = R6
```

a) Enumere los posibles riesgos que se presentan en la secuencia de código y que hay que detectar al ejecutarlo en este procesador.

Riesgo de control por el salto en I4
Riesgos de datos:
RAW: I2 con I1 por R1
RAW: I3 con I2 por R4
RAW: I4 con I2 por R4
RAW: I5 con I3 por R5
RAW: I6 con I5 por R5
RAW: I7 con I5 por R5
RAW: I8 con I6 por R7
RAW: I9 con I8 por R1

Complete los siguientes cronogramas de ejecución bajo los siguientes supuestos:

b) Sin adelantamientos. Los saltos condicionales se predicen no efectivos.

Instrucción \ tiempo	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
I1 ADD R1, R2, R3	F	D	E	M	W															
I2 LOAD R4, 0(R1)		F	(D)	(D)	D	E	M	W												
I3 MUL R5, R4, R4			(F)	(F)	F	(D)	(D)	D	E	M	W									
I4 BNE R1, R4, L1						(F)	(F)	F	D	**E**	M	W								
I5 SUB R5, R5, R2									F	(D)	~~D~~	~~E~~	~~M~~	~~W~~						
I6 ADD R7, R4, R5										(F)	~~F~~	~~(D)~~	~~(D)~~	~~D~~	~~E~~	~~M~~	~~W~~			
I7 / L1: ADD R6, R1, R5												F	D	E	M	W				
I8: OR R1, R7, R8													F	D	E	M	W			
I9: ST R6, 0(R1)														F	(D)	(D)	D	E	M	W
…																				

c) Con adelantamientos. Los saltos condicionales se predicen no efectivos.

Instrucción \ tiempo	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
I1 ADD R1, R2, R3	F	D	E	M	W															
I2 LOAD R4, 0(R1)		F	D	E	M	W														
I3 MUL R5, R4, R4			F	(D)	D	E	M	W												
I4 BNE R1, R4, L1				(F)	F	D	E	M	W											
I5 SUB R5, R5, R2						F	D	~~E~~	~~M~~	~~W~~										
I6 ADD R7, R4, R5							F	~~D~~	~~E~~	~~M~~	~~W~~									
I7 / L1: ADD R6, R1, R5									F	D	E	M	W							
I8: OR R1, R7, R8										F	D	E	M	W						
I9: ST R6, 0(R1)											F	D	E	M	W					
…																				

d) Con adelantamientos e incorporando un sistema de predicción basado en un BTB y suponiendo que el salto está en el BTB y acierta en la predicción de salto

Instrucción \ tiempo	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
I1 ADD R1, R2, R3	F	D	E	M	W															
I2 LOAD R4, 0(R1)		F	D	E	M	W														
I3 MUL R5, R4, R4			F	(D)	D	E	M	W												
I4 BNE R1, R4, L1				(F)	F	D	E	M	W											
I5 SUB R5, R5, R2																				
I6 ADD R7, R4, R5																				
I7 / L1: ADD R6, R1, R5						F	D	E	M	W										
I8: OR R1, R7, R8							F	D	E	M	W									
I9: ST R6, 0(R1)								F	D	E	M	W								
…																				

P6.- Se dispone de un procesador con sistema de memoria virtual paginado. Dispone de un TLB y de una cache real unificada de dos niveles (L1 y L2). Se pide evaluar el tiempo medio de acceso a memoria (en ciclos) con los siguientes datos. Tiempo de acceso a TLB t_{TLB}=1, tiempo de acceso a tabla de páginas t_{tablas}=10, tiempo de acceso a cache L1 t_{L1}=2, tiempo de acceso a cache L2 t_{L2}=4, y tiempo de acceso si hay fallo en L2 $t_{falloL2}$=15. El tiempo de acceso incluye la transferencia completa de datos entre los diferentes niveles de la jerarquía de memoria. La tasa de aciertos para L1 es del 80%, de L2 del 90% y del TLB es del 85%.

a) Indique como se realiza un acceso a memoria y exprese en una fórmula como debe calcularse el tiempo empleado en cada una de las estructuras de este sistema de memoria.

Tacc=(tTLB+fTLB*ttablas)+(tL1+fL1*(tL2+fL2*t falloL2)

b) Calcule el tiempo medio de un acceso en ciclos

Tacc=(tTLB+fTLB*ttablas)+(tL1+fL1*(tL2+fL2*$t_{falloL2}$) = (1+0,15*10)+(2+0,2*(4+0,1*15))=2,5+3,1=5,6

c) ¿Cuántos ciclos tarda un acceso que acierte en TLB y en la cache L2?

T = (tTLB)+(tL1+fL1*(tL2+fL2*t falloL2) = 1 + 2 + 4 = 7 ciclos tarda cada uno de este tipo de accesos y contribuye al tiempo medio de acceso con 1 + 2 + 0,2x4 = 3,8 ciclos

d) ¿Cuántos ciclos tarda un acceso a memoria en este sistema en el mejor caso posible?

Tmejor caso(acceso TLB + Acceso cache L1) = 3 ciclos

e) Si se cambiara la cache del sistema por una cache virtual manteniendo las mismas características de organización y estructura. ¿cuantos ciclos tarda un acceso a memoria en el nuevo sistema para el mejor caso posible?

Tmejor caso (acceso a cache L1) = 2 ciclos

P7.- Un sistema de memoria paginado utiliza un tamaño de página de 4 kBytes y el tamaño de los descriptores es de 2 bytes. No dispone de TLB y el inicio de la tabla de páginas en memoria principal se indica en el registro CR3 dentro de la MMU. Suponga que el valor de este registro es CR3 =0xA0000 y que el contenido de la memoria es el siguiente (solo se muestran algunos valores):

MEMORIA							
DIRECC.	**DATO**	**DIRECC.**	**DATO**	**DIRECC.**	**DATO**	**DIRECC.**	**DATO**
35EC0	**FF**	**50D80**	**44**	**A04F0**	**24**	**A09E0**	**C4**
35EC1	**44**	**50D81**	**5C**	**A04F1**	**22**	**A09E1**	**2F**
- - - -	**- - - -**	**- - - -**	**- - - -**	**- - - -**	**- - - -**	**- - - -**	**- - - -**
- - - -	**- - - -**	**- - - -**	**- - - -**	**- - - -**	**- - - -**	**- - - -**	**- - - -**
35ECE	**33**	**50D9E**	**32**	**A04F7**	**10**	**A09EE**	**50**
35ECF	**22**	**50D9F**	**F4**	**A04F8**	**F3**	**A09EF**	**3F**

En los descriptores los bits de control están el la parte más significativa, se utiliza representación “little endian” (el byte de mayor peso corresponde con la dirección de memoria mayor) y que la CPU accede a la DV 0x4F7D80 para leer una palabra de 2 bytes.

Responda razonadamente a las siguientes preguntas:

a. ¿Qué DR corresponde a la DV ?

DR= 0x50D80.

b. ¿Qué valor obtendrá la CPU?

Lee la palabra 0x5C44.

c. ¿Cuántos accesos a memoria han sido necesarios para todo el proceso y que direcciones de memoria han sido accedidas?

Inicio de la tabla de páginas en dirección de memoria 0xA0000.
DV= 0x4F7D80 , NPV = 0x4F7 que se usa como índice para acceder a la tabla (teniendo en cuenta el tamaño del descriptor)
Se accede a la dirección= 0xA0000+(0x4F7 x 2) = 0xA09EE
Se lee el descriptor 0x3F50. Marco 0x50 y junto al offset 0xD80 se genera la DR= 0x50D80.

Se accede a la dirección de memoria 0x50D80 para leer la palabra 0x5C44.

En todo el proceso se accede a las direcciones de memoria 0xA09EE y 0x50D80

P8.- A un sistema procesador con bus de direcciones de 16 bits, tamaño de palabra de 2 bytes y formato de instrucciones de 16 bits, se le desea dotar de una pequeña unidad caché para instrucciones, de 16 bloques y 4 bytes por bloque. La estrategia para reemplazamiento es LRU. Se valoran dos posibles configuraciones:

a. Caché de correspondencia directa.
b. Caché completamente asociativa.

Para valorar cuál de las soluciones es más adecuada, se ejecuta un programa con la siguiente secuencia de instrucciones:

- Las instrucciones en las direcciones de memoria de 0 a 62 se ejecutan una vez.
- Las instrucciones en las direcciones de memoria de 64 a 130, pertenecen a un bucle que se ejecuta 10 veces.

Responda razonadamente a las siguientes preguntas:

a) Indicar como se decodifica la dirección para el acceso a ambas caches.

CD:

ETIQUETA	INDICE	B/B
10	4	2

CA:

ETIQUETA	B/B
12	2

b) Encontrar la tasa de fallos para ambas cachés.

Una cache de 16 bloques * 4 bytes/bloque almacena 64 bytes. Por tanto, en ambos casos, las instrucciones de 0 a 62 llenan la caché. Dado que tenemos 4 bytes por bloque, se almacenan 2 instrucciones por bloque, de modo que por cada bloque se produce un fallo y un acierto. Se solicitan 32 instrucciones -> 16 fallos 16 aciertos. Las instrucciones de la 64 a la 126 sobrescriben la cache completamente. Se solicitan 32 instrucciones → 16 fallos 16 aciertos. Las instrucciones 128 y 130 respectivamente sobrescriben el bloque 0. Se solicitan 2 instrucciones → 1 fallo 1 acierto
El comportamiento del bucle varía en función del tipo de caché:
CD:
Como se ejecuta el bucle 10 veces, las nueve veces restantes se produce un fallo al cargar las direcciones 64 y otro al cargar la 128.
Por tanto, el total de fallos es 16 + (16+1) + (9*2) = 51. 51 fallos / 372 accesos = 0,137 13,7%
CA:
En este caso, el bucle provoca un fallo en cascada ya que al rellenar el bloque 0 con las instrucciones 128 y 130 se provoca fallo en todos los bloques (debido a LRU) al acceder a la primera instrucción de cada bloque, y luego acertamos con la segunda, por tanto 9 * 17 fallos = 153 fallos.
Por tanto, el total de fallos es 16 + (16+1) + 153 = 186. 186 fallos / 372 accesos = 0,5 50%

Tras obtener la tasa de fallos, se selecciona la caché de correspondencia directa. Se quiere añadir soporte para memoria virtual, por lo que se propone incluir una MMU con TLB y un sistema paginado de dos niveles, donde el primer nivel se implementa usando una memoria de sustitución directa dentro de la MMU. El tamaño de página es el más adecuado para permitir acceso simultáneo a TLB y caché. El direccionamiento virtual debe permitir trabajar con 1 MB de memoria aunque solo disponga de 64KB de memoria física. El TLB debe ser completamente asociativo con 32 entradas. Indicar justificando la respuesta:

a) El **tamaño mínimo de página** que permita el acceso en paralelo al TLB y la unidad cache.

El offset debe ser igual a la suma de los bits de índice y b/b, por tanto 6 bits → 64 bytes

NOTA: En caso de no responder a este apartado, se permite elegir un tamaño de página para el resto de secciones.

b) Indicar como se decodifica la dirección para acceder al TLB.

La dirección virtual es de 20 bits. Si el offset ocupa 6 bits, el resto de la dirección se usa como etiqueta dentro del TLB al ser completamente asociativo.

ETIQUETA	OFFSET
14	6

c) Indicar como se decodifica la dirección para acceder a la tabla de páginas. El tamaño de descriptor se corresponde con el número de bytes mínimo necesario.

La dirección real tiene 16 bits. Si el offset es de 6, el MPR es de 10. Por tanto, el número de bytes mínimo que debe ocupar el descriptor es 2 bytes (16 bits).
Con esta información la decodificación de la dirección virtual para el acceso a la tabla de páginas es:

N1	N2	OFFSET
9	5	6

d) El tamaño de la MMU.

La MMU contiene al TLB y la memoria de sustitución directa. Por tanto, el tamaño de la MMU es el tamaño del TLB + tamaño de N1.
Tamaño del TLB = 32 entradas * (14 bits etiqueta + 16 bits descriptor) = 960 bits
Tamaño de N1 = 2^9 posiciones * 16 bits descriptor = 8192 bits
Total = 9152 bits = 1144 bytes.

P9.- Se dispone de un procesador con sistema de memoria virtual paginado. Dispone de un TLB y de una cache virtual unificada de dos niveles (L1 y L2). Se pide evaluar el tiempo medio de acceso a memoria (en ciclos) con los siguientes datos. Tiempo de acceso a TLB t_{TLB}=1, tiempo de acceso a tabla de páginas t_{tablas}=10, tiempo de acceso a cache L1 t_{L1}=2, tiempo de acceso a cache L2 t_{L2}=4, y tiempo de acceso si hay fallo en L2 $t_{falloL2}$=15. El tiempo de acceso incluye la transferencia completa de datos entre los diferentes niveles de la jerarquía de memoria. La tasa de aciertos para L1 es del 80%, de L2 del 90% y del TLB es del 85%.

b) Indique el orden de acceso a los diferentes elementos cuando se realiza un acceso a memoria y exprese en una fórmula como debe calcularse el tiempo medio de acceso a memoria en este sistema.

Orden: Acceso a L1 cache virtual, si falla acceso a L2 cache virtual , si falla acceso al L1 TLB para obtener DR, si falla acceso al L2 TLB para obtener DR, si falla acceso a tablas para obtener DR. Acceso a L2 y L1 de cache virtual para actualizar el bloque en cache.

Tacc= (fl1xfl2)(tTLB+fTLB*ttablas)+(tL1+fL1*(tL2+fL2*t falloL2)

b) Calcule el tiempo medio de un acceso en ciclos.

Tacc=(fl1xfl2)((tTLB+fTLB*ttablas)+(tL1+fL1*(tL2+fL2*$t_{falloL2}$) =
0,02x(1+0,15*10)+(2+0,2*(4+0,1*15))=0,05+3,1=3,15 ciclos

c) ¿Cuántos ciclos tarda un acceso a memoria en este sistema en el mejor caso posible?

Tmejor caso = 2 ciclos (acceso a L1 cache)

d) Si se cambiara la cache del sistema por una cache real manteniendo las mismas características de organización y estructura. ¿cuantos ciclos tarda un acceso a memoria en el nuevo sistema para el mejor caso posible?

Tmejor caso = 1 + 2 = 3 ciclos (acceso al TLB + acceso a L1 cache)

P10.- La MMU de un sistema está compuesta por un TLB y un registro de 32 bits que mantiene la dirección de memoria donde se encuentra el primer nivel de la tabla de páginas. El sistema de memoria virtual es multinivel con dos niveles, consiguiendo direccionar hasta 4 Gbytes de memoria aunque sólo dispone de 16Mbytes de memoria real. Se elige un tamaño de página de 4 kbytes y el sistema de jerarquía de memoria se define de tal manera que se tenga el mismo número de entradas en ambos niveles, empleando en todos los casos descriptores de 4 bytes. En todo el sistema de memoria se garantiza el alineamiento de las diferentes estructuras en múltiplos del tamaño de página. Con este alineamiento, se pretende utilizar el mayor número posible de los bits de cada descriptor como bits de control. Estos bits de control ocupan la parte menos significativa del descriptor.

Considere que la CPU direcciona la DV: 0x00801045 y en la MMU se produce un fallo en TLB y en el acceso al sistema multipaginado se leen los siguientes valores para le registro y los descriptores:

Registro de la MMU = 0x00440000, Descriptor de N1: 0xC045FFFF y Descriptor de N2: 0xA0000000

a) Indique los campos en los que se descompone la dirección virtual para acceder a la tabla de páginas.

DV de 32 bits, y DR de 24 bits

Campos para decodificar la DV:
N1: 10 bit
N2: 10 bit
Offset:12bit

b) ¿Qué DR devuelve la MMU?

Descriptor de N2 : Marco = A00 control: 00000

DR= A00045

c) Indique las direcciones de memoria principal a las que ha accedido la MMU para realizar la traducción

DV: 00801045
N1: 0000 0000 10
N2: 0000 0000 01
Offset : 045

Acceso a memoria para obtener descriptor de N1
Dirección Real : 440000 + N1*4 = 440000 + 008 = 440008

Acceso a memoria para obtener el descritpor de N2
Dirección Real : C04000 + N2*4 = C04000 + 004= C04004

P11.- Sea un sistema procesador MIPS con memoria total de 1 Mbyte y una caché unificada de 4 Kbytes con bloques de 256 bytes. Se quiere ejecutar el siguiente programa cuyo bucle se ejecuta 4 veces. A los datos del vector A se acceden a partir de la dirección 0x321F0 y a los del B desde la dirección 0x42200.

```
Dirección   Instrucción                  Comentario
0x021FC     inic: load r1, r2(0x1F0)     ; load A[i] into r1
0x02200           add r6, r6, r1         ; add A[i] to R6
0x02204           add r7, r1, r10        ; A[i] + cte. R10 contains cte
0x02208           store r7, r4(0x200)    ; store B[j]
0x0220C           add r2, r2, 4          ; i++ (data size is 4 bytes)
0x02210           add r4, r4, 4          ; j++ (data size is 4 bytes)
0x02214           bnq r2, r0, inic       ; Jump to inic
```

Se quieren evaluar dos arquitecturas para la caché: a) Correspondencia directa y b) Asociativa de 4 vías.

Asumiendo que la caché está vacía inicialmente, se pide:

- Indicar los recursos necesarios para implementar cada tipo de cache, suponiendo que su esquema de funcionamiento es post-escritura.

  CD Tag: 8 bits Índice: 4 bits B/B: 8 bits

  16 entradas x (8 bits de tag + 1 bit validez + 1 bit dirty) 1 comparador

  CA Tag: 10 bits Índice: 2 bits B/B: 8 bits

  4 entradas x (10 bits de tag + 1 bit validez + 1 bit dirty + 2 bits LRU) 4 comparadores

- Calcular el porcentaje de aciertos en caché. Para ello se debe completar una tabla por cada tipo de caché que represente los campos en los que se decodifican las direcciones y las entradas de la caché que se modifican.

Correspondencia Directa				
Dirección	**Iter. 1**	**Iter. 2**	**Iter. 3**	**Iter. 4**
02 1 FC	F	F	F	F
32 1 F0	F	F	F	F
02 2 00	F	A	A	A
02 2 04	A	A	A	A
02 2 08	A	A	A	A
42 2 00	F	F	F	F
02 2 0C	F	F	F	F
02 2 10	A	A	A	A
02 2 14	A	A	A	A
Asociativa 4 vías				
Dirección	**Iter. 1**	**Iter. 2**	**Iter. 3**	**Iter. 4**
0200 01 FC	F v1	A v1	A v1	A v1
3200 01 F0	F v2	A v2	A v2	A v2
0200 02 00	F v1	A v1	A v1	A v1
0200 02 04	A v1	A v1	A v1	A v1
0200 02 08	A v1	A v1	A v1	A v1
4200 02 00	F v2	A v2	A v2	A v2
0200 02 0C	A v1	A v2	A v2	A v2
0200 02 10	A v1	A v2	A v2	A v2
0200 02 14	A v1	A v2	A v2	A v2

Aciertos: 19/36 → 52%

Aciertos: 32/36 → 88%