Examen Junio 2012 II

Estructura de Computadores (Universitat Oberta de Catalunya)

grapa

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	18/01/2012	15:30

75.573 18 01 12 EX

Espacio para la etiqueta identificativa con el código personal del **estudiante**.
Examen

Ficha técnica del examen

- **Comprueba que el código y el nombre de la asignatura corresponden a la asignatura de la cual estás matriculado.**
- **Debes pegar una sola etiqueta de estudiante en el espacio de esta hoja destinado a ello.**
- **No se puede añadir hojas adicionales.**
- **No se puede realizar las pruebas a lápiz o rotulador.**
- **Tiempo total** 2 **horas**
  - **En el caso de que los estudiantes puedan consultar algún material durante el examen, ¿cuál o cuáles pueden consultar?:** No se puede utilizar calculadora ni consultar material auxiliar.
  - **Valor de cada pregunta:** Pregunta 1 (20%); Pregunta 2 (40%); Pregunta 3 (40%).
- **En el caso de que haya preguntas tipo test: ¿descuentan las respuestas erróneas?** NO **¿Cuánto?**
- **Indicaciones específicas para la realización de este examen**

Enunciados

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	18/01/2012	15:30

Pregunta 1: (50%: 1.1: 10%, 1.2: 10%)

Solamente se deben completar las instrucciones marcadas.
Los puntos suspensivos indican que hay más código pero no se tiene que completar.

NOTA: Si el código propuesto en cada pregunta no se corresponde con la forma en que vosotros plantearíais la respuesta, podéis rescribir el código o parte del código según vuestro planteamiento.

Pregunta 1.1

Completa el código correspondiente a la subrutina m_mul_escalar_mr (producto escalar, escalar * mr).

```
m_mul_escalar_mr:
    push rbp
    mov rbp, rsp

    mov esi, 0
    m_mul_escalar_bucle:
      mov  eax, [mr+esi*4]
      imul eax, [escalar]
      mov  [mr+esi*4], eax
      inc esi
      cmp esi, 8
      jle  m_mul_escalar_bucle

    m_mul_escalar_end:
    mov rsp, rbp
    pop rbp
    ret
```

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	18/01/2012	15:30

Pregunta 1.2

Completa el código correspondiente a la subrutina m_det_mr, subrutina de cálculo del determinante de la matriz mr por la regla de Sarrus. Se pide completar solamente la parte de código correspondiente a extender la matriz 3x3 mr sobre la matriz auxiliar mAux 5x3.

```
m_det_mr:
    push rbp
    mov rbp, rsp
    mov esi, 0      ;Inicializamos el índice de la matriz mAux 5x3
    mov edi, 0      ;Inicializamos el índice de la matriz mr 3x3

    m_det_ini:
      mov eax, [mr+edi*4]
      mov [mAux+esi*4], eax
      inc esi
      cmp esi, 14       ;Hemos acabado la copia sobre mAux
      jg  m_det_next
      inc edi           ;Incrementamos el índice de mr
      cmp edi, 8        ;Finaliza el recorrido de mr
      jle m_det_ini
      mov edi, 0
      jmp m_det_ini
m_det_next:
...
```

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	18/01/2012	15:30

Pregunta 2 (40%: 2.1: 15%, 2.2: 15%, 2.3: 10%)

Apartado 2.1 (15%)

El estado inicial del computador CISCA justo antes de comenzar la ejecución de cada fragmento de código (de cada apartado) es el siguiente:

R0 = 00000000h R1 = 00001000h R2 = 00002000h	M(00001000h) = 00001000h M(00002000h) = 00001000h M(00003000h) = 00000000h	Z = 0 C = 0 S = 0 V = 0

¿Cuál será el estado del computador después de ejecutar cada fragmento de código? (sólo modificaciones, excluyendo el PC).

a)

```
XOR  R1, R2
MOV  R2, 00001000h
SUB  R1, [00001000h+R2]
```

R1 = 00002000h
R2 = 00001000h
Z = 0, S = 0
C = 0, V = 0

b)

```
MOV  R2, 00001000h
SUB  R1, 60h
XOR  R2, R0
MOV  R0, FFFFFFFFh
```

R2 = 00001000h
R1 = 00000FA0h
R0 = FFFFFFFFh
Z = 0, S = 0
C = 0, V = 0

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	18/01/2012	15:30

Apartado 2.2 (15%)

En la memoria de un computador CISCA tenemos almacenada una matriz de 10 x 100 elementos (10 filas por 100 columnas) a partir de la dirección simbólica M. Cada elemento es un número entero codificado en complemento a 2 con 32 bits.

Completad los huecos del fragmento de código CISCA para poner a 0 el elemento M[i][j], lo que en C se especificaría con la sentencia: M[i][j] = 0;

La matriz está almacenada por filas en posiciones consecutivas de memoria, como es habitual cuando se traduce código en C. Por ejemplo, los elementos M[0][0], M[0][1], M[1][0] y M[7][40] se encuentran almacenados en las direcciones de memoria M, M+4, M+400 y M+2960 respectivamente.

Se sabe que en R1 se encuentra almacenado el valor de la variable i, y en R2 el de la j y que después de ejecutarse el fragmento de código todos los registros deben mantener los valores originales. El código es correcto pero no es todo lo eficiente que podría ser.

```
PUSH R1
PUSH R2
MUL  R1, 400
MUL  R2, 4
ADD  R1, R2
MOV  [M+R1], 0
POP  R2
POP  R1
```

Examen 2011/12-1

Asignatura	**Código**	**Fecha**	**Hora inicio**
Estructura de computadores	75.573	18/01/2012	15:30

Apartado 2.3 (10%)

Dado el siguiente fragmento de código de un programa en lenguaje ensamblador del CISCA:

```
            JMP  Label_2
Label_1:    SUB  [B+R1], 4
Label_2:    MOV  [R3+4], R4
```

Traducidlo a lenguaje máquina y expresadlo en la siguiente tabla. Suponed que la primera instrucción del código se ensambla a partir de la dirección 00FF0014h (que es el valor del PC antes de empezar la ejecución del fragmento de código). Suponed también que la dirección simbólica B vale 0000F000h. En la tabla para el resultado usad una fila para codificar cada instrucción. Si suponemos que la instrucción comienza en la dirección @, el valor Bk de cada uno de los bytes de la instrucción con direcciones @+k para k=0, 1,… se debe indicar en la tabla en hexadecimal en la columna correspondiente (recordad que los campos que codifican un desplazamiento en 2 bytes o un inmediato o una dirección en 4 bytes lo hacen en formato little endian, esto hay que tenerlo en cuenta escribiendo los bytes de menor peso, de dirección más pequeña, a la izquierda y los de mayor peso, dirección mayor, a la derecha). Completad también la columna @ que indica para cada fila la dirección de memoria del byte B0 de la instrucción que se codifica en esa fila de la tabla.

A continuación os damos como ayuda las tablas de códigos:

Tabla de códigos de instrucción

Instrucción	B0
JMP	40h
MOV	10h
SUB	21h

Tabla de modos de direccionamiento (Bk<7..4>)

Campo modo Bk<7..4>	Modo
0h	Inmediato
1h	Registro
2h	Memoria
3h	Indirecto
4h	Relativo
5h	Indexado
6h	A PC

Examen 2011/12-1

Asignatura	**Código**	**Fecha**	**Hora inicio**
Estructura de computadores	75.573	18/01/2012	15:30

Tabla de modos de direccionamiento (Bk<3..0>)

Campo modo Bk<3..0>	Significado
Nº registro	Si el modo debe especificar un registro
0	No se especifica registro.

Tabla para contestar a la pregunta:

		Bk para k=0..10										
@	**Ensamblador**	0	1	2	3	4	5	6	7	8	9	10
00FF0014	`JMP Label_2`	40	00	25	00	FF	00					
00FF001A	`SUB [B+R1], 4`	21	51	00	F0	00	00	00	04	00	00	00
00FF0025	`MOV [R3+4], R4`	10	43	04	00	14						
00FF002A												

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	18/01/2012	15:30

Pregunta 3 (40%: 3.1: 15%, 3.2: 15%, 3.3: 10%)

Apartado 3.1 (15%)

Tenemos un sistema de memoria en el que todos los accesos se realizan a palabra (no nos importa de qué tamaño es una palabra). Supondremos que el espacio de direcciones de memoria se descompone en bloques de 4 palabras. Cada bloque comienza en una dirección múltiplo de 4. Así, el bloque 0 contiene las direcciones 0, 1, 2 i 3, el bloque 1, las direcciones 4, 5, 6 i 7, y el bloque N las direcciones 4*N, 4*N+1, 4*N+2 i 4*N+3. Una fórmula para calcular el identificador numérico del bloque es la siguiente:

Bloque = Dirección de memoria (dirección a palabra) DIV 4 (tamaño del bloque en palabras)

Suponemos que el sistema también dispone de una memoria cache de 4 líneas (donde cada línea tiene el tamaño de un bloque, es decir, 4 palabras). Estas líneas se identifican como líneas 0, 1, 2 i 3. Cuando se hace referencia a una dirección de memoria principal, si esta dirección no se encuentra en la memoria cache, se lleva todo el bloque correspondiente desde la memoria principal a una línea de la memoria cache (así si hademos referencia a la dirección 2 de memoria principal traeremos el bloque formado por las palabras 0, 1, 2 i 3).

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	18/01/2012	15:30

Apartado 3.1.1

Suponemos que el sistema utiliza una **política de asignación directa**, de manera que cada bloque de la memoria principal sólo se puede llevar a una línea determinada de la memoria cache. En este caso, el identificador del bloque determina la línea específica donde se puede guardar utilizando la siguiente fórmula (similar a la fórmula para determinar el bloque):

Línea = identificador de bloque MOD 4 (tamaño de la cache en líneas)

La ejecución de un programa genera la siguiente lista de lecturas a memoria:

1, 2, 3, 4, 5, 10, 11, 12, 30, 31, 32, 1, 2, 3, 4, 5, 10, 11, 12, 30

3.1.1.a)
La siguiente tabla muestra el estado inicial de la cache, que contiene las primeras 16 palabras de la memoria (organizadas en 4 bloques). Completar la tabla para mostrar la evolución de la cache durante la ejecución del programa. Para cada fallo en la cache hay que rellenar una nueva columna indicando que referencia a memoria ha provocado el fallo y el cambio que se produce en el estado de la memoria cache (la línea que se modifica).

	Estado Inicial	Fallo:30	Fallo:32	Fallo:1	Fallo:12
Línea 0	0, 1, 2, 3		32, 33, 34, 35	0, 1, 2, 3	
Línea 1	4, 5, 6, 7				
Línea 2	8, 9, 10, 11				
Línea 3	12, 13, 14, 15	28, 29, 30, 31			12, 13, 14, 15

	Fallo: 30	Fallo:	Fallo:	Fallo:	Fallo:
Línea 0					
Línea 1					
Línea 2					
Línea 3	28, 29, 30, 31				

3.1.1.b) ¿Cuál es la tasa de fallos (T_f) ?

T_f = 5 fallos / 20 accesos = 0,25

3.1.1.c) Suponemos que el tiempo de acceso a la memoria cache, o tiempo de acceso en caso de acierto (t_e), es de 4 ns y el tiempo total de acceso en caso de fallo (t_f) es de 24 ns. Considerando la tasa de fallos obtenida en la pregunta anterior, ¿cuál es el tiempo promedio de acceso a memoria (t_m) ?

$t_m = T_f \times t_f + (1\text{-}T_f) \times t_e$ = 0,25 * 24 ns + 0,75 * 4 ns = 6 ns + 3 ns = 9 ns

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	18/01/2012	15:30

Apartado 3.1.2

Ahora suponemos que el mismo sistema utiliza una **política de asignación completamente asociativa,** de manera que cualquier bloque de la memoria principal se puede llevar a cualquier bloque de la memoria cache.

Si encontramos que la cache ya está llena, se utiliza un algoritmo de reemplazo LRU, de manera que sacaremos de la memoria cache aquel bloque que hace más tiempo que no se ha referenciado.
Consideramos la misma lista de lecturas a memoria:

1, 2, 3, 4, 5, 10, 11, 12, 30, 31, 32, 1, 2, 3, 4, 5, 10, 11, 12, 30

3.1.2.a)
La siguiente tabla muestra el estado inicial de la cache, que contiene las primeras 16 palabras de la memoria (organizadas en 4 bloques). Completar la tabla para mostrar la evolución de la cache durante la ejecución del programa. Para cada fallo en la cache hay que rellenar una nueva columna indicando que referencia a memoria ha provocado el fallo y el cambio que se produce en el estado de la memoria cache (la línea que se modifica).

	Estado Inicial	Fallo:30	Fallo:32	Fallo:1	Fallo: 4
Línea 0	0, 1, 2, 3	28, 29, 30, 31			
Línea 1	4, 5, 6, 7		32, 33, 34, 35		
Línea 2	8, 9, 10, 11			0, 1, 2, 3	
Línea 3	12, 13, 14, 15				4, 5, 6, 7

	Fallo: 10	Fallo: 12	Fallo: 30	Fallo:	Fallo:
Línea 0	8, 9, 10, 11				
Línea 1		12, 13, 14, 15			
Línea 2			28, 29, 30, 31		
Línea 3					

3.1.2.b) ¿Cuál es la tasa de fallos (T_f) ?

T_f = 7 fallos / 20 accesos = 0,35

3.1.2.c) Suponemos que el tiempo de acceso a la memoria cache, o tiempo de acceso en caso de acierto (t_e), es de 4 ns y el tiempo total de acceso en caso de fallo (t_f) es de 24 ns. Considerando la tasa de fallos obtenida en la pregunta anterior, ¿cuál es el tiempo promedio de acceso a memoria (t_m) ?

$t_m = T_f \times t_f + (1-T_f) \times t_e$ = 0,35 * 24 ns + 0,65 * 4 ns = 8,4 ns + 2,6 ns =11 ns

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	18/01/2012	15:30

Apartado 3.2 (15%)

3.2.1 (5%)

El siguiente código CISCA transfiere datos entre memoria y disco utilizando la técnica de E/S por interrupciones.

```
1.              CLI
2.              PUSH  R0
3.              PUSH  R1
4.              IN    R0, [10]
5.              IN    R0, [18]
6.              MOV   [R1], R0
7.              POP   R1
8.              POP   R0
9.              STI
```

Indicar si la transferencia es de salida al disco o de entrada desde el disco, y los dos errores que se encuentran en el código

Transferencia de entrada

1. Falta IRET al final del código
2. Falta obtener en R1 un valor válido como dirección destino de los datos leídos del disco, y actualizarla

No es necesariamente un error el hecho de ejecutar dos instrucciones IN seguidas sobre el mismo registro.

3.2.2 (10%)

Una rutina de servicio de interrupciones (RSI) hecha con el repertorio CISCA sirve para transferir un dato de 4 bytes desde el disco hasta memoria. La transferencia con el disco de un bloque de 8.000 datos (cada dato de 4 bytes) tarda 32 milisegundos. El tiempo consumido por la CPU desde que se detecta la interrupción hasta que se acaba de ejecutar la RSI es siempre de 8 ns.

¿Qué porcentaje del tiempo total de transferencia dedica la CPU?

Tiempo consumido por la CPU por interrupción: 8 ns
Número total de interrupciones producidas: 8.000 datos / 1 dato por interrupción= 8.000
Tiempo consumido en total: 8 ns por interrupción x 8.000 interrup. = 64 us

Porcentaje de tiempo consumido por la CPU durante el tiempo de la transferencia:
64 us / 32.000 us = 0,002 ➔ 0,2 %

Examen 2011/12-1

Asignatura	**Código**	**Fecha**	**Hora inicio**
Estructura de computadores	75.573	18/01/2012	15:30

Apartado 3.3 (10%)

El sistema de E/S por DMA de un computador tiene las siguientes características:

- El **tiempo de cesión** y el **tiempo de recuperación** del bus ($t_{cesión}$ + t_{recup}) son 4 ns.
- El **tiempo de la transferencia** por el bus (t_{mem}) es de 2 ns.
- Se envían por el bus 2.000 datos de 32 bits cada uno

Calcular el tiempo total de ocupación del bus por parte del controlador de disco/DMA para llevar a término la transferencia (no se utilizan ráfagas).

Tiempo de ocupación del Bus, t_{transf_dato}: $t_{cesión} + t_{mem} + t_{recup} = 4 + 2 = 6$ ns
Número de peticiones del Bus, N_{pet_bus}: 2.000
Tiempo total de ocupación del Bus t_{transf_bloque}: $t_{transf\_dato} \times N_{pet\_bus} = 6 \text{ ns} \times 2.000 =$ 12 us