grapa

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	21/01/2012	09:00

75.573 21 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	21/01/2012	09:00

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_copy_col (copia el vector b de 3 posiciones en la columna de la matriz mr 3x3 indicada por la variable col).

```
m_copy_col:
   push rbp
   mov rbp, rsp

   mov esi, [col]
   mov edi, 0
   m_copy_col_bucle:
     mov  eax, [b+edi*4]
     mov  [mr+esi*4], eax
     add esi, 3
     inc edi
     cmp edi, 3
     jl  m_copy_col_bucle

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

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	21/01/2012	09:00

Pregunta 1.2

Completa el código correspondiente a la subrutina que hace la transpuesta de la matriz mr.

```
m_trans_mr:
  push rbp
  mov rbp, rsp
  mov edx, 8
  m_trans_mr_row:
    mov esi, edx ;Valor de referencia de la diagonal.
    mov edi, edx ;esi estará por debajo de la diagonal y edi por encima.
    m_trans_mr_col:
      sub  esi, 1 ; Restamos primero para evitar transponer los
      sub  edi, 3 ; valores de la diagonal que no hay que mover
      jl   m_trans_mr_next ;Hemos acabado la columna
      mov  eax, [mr+esi*4]; Hacemos el intercambio de valores.
      xchg eax, [mr+edi*4]
      mov  [mr+esi*4], eax
      jmp  m_trans_mr_col

  m_trans_mr_next:
    sub edx, 4          ;Nos movemos por la diagonal.
    jge m_trans_mr_row
  m_trans_mr_end:
    mov rsp, rbp
    pop rbp
    ret
```

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	21/01/2012	09:00

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(00000000h) = 00001000h M(00001000h) = 00002000h M(00002000h) = 10000000h	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)

```
MOV  R2, FFFFFFFFh
SAR    [R1+00001000h], 4
XOR    R2,[00002000h]
```

[00002000h] = 01000000h
R2 = FEFFFFFFh
Z = 0, S = 1
C = 0, V = 0

b)

```
ADD [R1], R1
MOV R2, 00002000h
SUB [R1], R2
```

R2 = 00002000h
M(00001000h) = 00001000h
Z = 0, S = 0
C = 0, V = 0

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	21/01/2012	09:00

Apartado 2.2 (15%)

Completad los huecos en el fragmento de código CISCA para que sea una traducción eficiente al ensamblador del código en C que pone a 0 la columna 20 de la matriz M de 7 por 100 números enteros (7 filas por 100 columnas):

```
i = 0;
do {
   M[i][20]=0;
   i = i+1;
} while (i<7);
```

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[5][40] se encuentran almacenados en las direcciones de memoria M, M+4, M+400 y M+2160 respectivamente.

Después de ejecutarse el fragmento de código todos los registros deben mantener los valores originales.

```
        PUSH  R1
        MOV   R1, 80
DO:     MOV   [M + R1], 0
        ADD   R1, 400
        CMP   R1, 2880          ; si aquí se pone 2480
        JL    DO                ; aquí se debe poner JLE
        POP   R1
```

Examen 2011/12-1

Asignatura	**Código**	**Fecha**	**Hora inicio**
Estructura de computadores	75.573	21/01/2012	09:00

Apartado 2.3 (10%)

Dado el siguiente código en ensamblador:

```
                MUL   R3, [X+R5]
                SUB   R5, 4
      end_w:    MOV   [M], R3
```

Traducidlo a lenguaje máquina y expresadlo en la siguiente tabla. Suponed que la primera instrucción del código se ensambla a partir de la dirección 00002027h (que es el valor del PC en el estado inicial). Suponed también que las direcciones simbólicas X y M valen 00000004 y 00000008 respectivamente. En la siguiente tabla usad una fila para codificar cada instrucción (como se ha hecho en el ejemplo anterior). Si suponemos que la instrucción comienza en la dirección @, el valor de cada uno de los bytes de la instrucción con direcciones @+i para i=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).

A continuación os damos como ayuda las tablas de códigos:

Tabla de códigos de instrucción

Instrucción	B0
MOV	10h
SUB	21h
MUL	22h

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

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

Examen 2011/12-1

Asignatura	**Código**	**Fecha**	**Hora inicio**
Estructura de computadores	75.573	21/01/2012	09:00

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

		Bk para k=0..10										
@	**Ensamblador**	0	1	2	3	4	5	6	7	8	9	10
00002027h	MUL R3, [X+R5]	22	13	55	04	00	00	00				
0000202Eh	SUB R5, 4	21	15	00	04	00	00	00				
00002035h	MOV [M],R3	10	20	08	00	00	00	13				
0000203Ch												

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	21/01/2012	09:00

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	21/01/2012	09:00

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, 10, 11, 12, 13, 14, 30, 31, 32, 1, 2, 3, 10, 11, 12, 13, 14, 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, 13, 14

	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) ?

$t_m = T_f \times t_f + (1-T_f) \times t_e$ = 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	21/01/2012	09:00

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

	Fallo:12	Fallo:30	Fallo:	Fallo:	Fallo:
Línea 0					
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 = 6 fallos / 20 accesos = 0,3

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) ?

$t_m = T_f \times t_f + (1-T_f) \times t_e$ = 0,3 * 24 ns + 0,7 * 4 ns = 7,2 ns + 2,8 ns = 10 ns

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	21/01/2012	09:00

Apartado 3.2 (15%)

Una RSI de repertorio CISCA transfiere datos desde el disco a la memoria del procesador:

- Tiempo medio de latencia del disco de 3 ms ($t_{latencia}$)
- Velocidad de transferencia del disco de 20 MB/s (ν_{transf})
- La RSI ejecuta 10 instrucciones, cada una en un tiempo de 3 ciclos de reloj
- La CPU requiere 8 ciclos de reloj adicionales desde que se detecta la interrupción del disco hasta que se transfiere el control a la RSI, (t_{rec_int}).
- Tiempo de programación y finalización de la transferencia de 20 ns (t_{prog} + t_{final})
- Transferencia de N_{datos}= 4.000 datos de 4 Bytes

¿Hasta qué frecuencia de reloj mínima del procesador se puede garantizar que no se pierdan datos en la transferencia?

En la fase de transferencia de datos, el disco proporciona 20 MB/s, es decir, 20 Bytes cada us, o 4 bytes cada 200 ns (por tanto, tenemos una interrupción cada 200 ns). Este es el tiempo máximo que debería tardar la gestión de la interrupción, incluyendo el tiempo adicional por transferir el control a la RSI.

El tiempo consumido por la CPU en cada interrupción, t_{transf_dato}, es de 8 + 3×10 = 38 ciclos de reloj (transferir control a RSI + ejecutar RSI).

200 ns / 38 ciclos de reloj = 5,26 ns / ciclo de reloj (es el tiempo máximo)

Por tanto, la frecuencia mínima para no perder datos sería de:

1 ciclo de reloj / 5,26 ns = 190 millones de ciclos / segundo = 190 MHz

Examen 2011/12-1

Asignatura	Código	Fecha	Hora inicio
Estructura de computadores	75.573	21/01/2012	09:00

Apartado 3.3 (10%)

3.3.1 (5%)

El tiempo medio de latencia de un disco, $t_{latencia}$, es de 3 ms y la velocidad de transferencia, ν_{transf}, es de 40 MB/s. Si se quiere leer del disco un bloque consecutivo de 1 MB, N_{bytes}, ¿Cuál es el tiempo de transferencia de todo el bloque, t_{bloque} ?

t_{datos} = 1 MB (N_{bytes}) / 40 MBytes/s (ν_{transf}) = 0,025 segundos = 25 ms.

t_{bloque} = $t_{latencia}$ + t_{datos} = 3 ms + 25 ms = 28 ms

3.3.2 (5%)

Un programa hecho con el repertorio CISCA envía un bloque de 8.000 datos de 4 bytes cada dato, desde la memoria al disco mediante la técnica de E/S programada. Cada una de las instrucciones tarda 4 cicles en ejecutarse, y el microprocesador funciona con un reloj de de 2 GHz. La velocidad de transferencia del disco es de 100.000 bytes per segundo.

¿Cuánto tiempo dura la transferencia y qué porcentaje de este tiempo dedica la CPU a la gestión de la transferencia?

Se escriben un total de 4 × 8.000 = 32.000 Bytes, con un ritmo de transferencia de 100.000 Bytes /s → 32.000 / 100.000 = 320 ms

La CPU está completamente dedicada a la transferencia (100%) y no hace otra cosa durante el tiempo en que se produce la transferencia con el disco.