Espai grapa

Examen 2015/16-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/01/2016	18:30

⊂05.573ℜ13ℜ01ℜ16ℜEΞ*∈
05.573 13 01 16 EX

Enganxeu en aquest espai una etiqueta identificativa amb el vostre codi personal
Examen

Aquest enunciat correspon també a les assignatures següents:

- 05.096 - Ampliació d'estructura i tecnologia de computadors

Fitxa tècnica de l'examen

- **Comprova que el codi i el nom de l'assignatura corresponen a l'assignatura en la qual estàs matriculat.**
- **Només has d'enganxar una etiqueta d'estudiant a l'espai corresponent d'aquest full.**
- **No es poden adjuntar fulls addicionals.**
- **No es pot realitzar la prova en llapis ni en retolador gruixut.**
- **Temps total:** 2 **h.**
- **En cas que els estudiants puguin consultar algun material durant l'examen, quin o quins materials poden consultar?**
  No es pot utilitzar calculadora, ni material auxiliar.
- **Valor de cada pregunta:** Pregunta 1 (20%); Pregunta 2 (35%); Pregunta 3 (35%); Pregunta 4 (10%)
- **En cas que hi hagi preguntes tipus test: Descompten les respostes errònies?** NO **Quant?**

- **Indicacions específiques per a la realització d'aquest examen:**

Enunciats

No es pot utilitzar calculadora. Cal saber interpretar un valor en binari, decimal o hexadecimal per a realitzar l'operació que es demani. I el resultat s'ha d'expressar en el format corresponent.

Examen 2015/16-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/01/2016	18:30

Valoració de les preguntes de l'examen

Pregunta 1 (20%)

Pregunta sobre la pràctica.

Cal completar les instruccions marcades o afegir el codi que es demana.
Els punts suspensius indiquen que hi ha més codi però no l'heu de completar.

NOTA: En cas que el codi proposat en cada pregunta no es correspongui amb la forma que vosaltres plantejaríeu la resposta, podeu reescriure el codi o part del codi segons el vostre plantejament.

1.1: 10%
1.2: 10%

Pregunta 2 (35%)
2.1: 10%
2.2: 15%
2.3: 10%

Pregunta 3 (35%)
3.1: 15%
3.1.1: 10%
3.1.2: 5%

3.2: 20%
3.2.1: 10%
3.2.2: 5%
3.2.3: 5%

Pregunta 4 (10%)
4.1: 5%
4.2: 5%

Examen 2015/16-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/01/2016	18:30

Pregunta 1

1.1 Pràctica – Part obligatòria

Escriure un fragment de codi assemblador de la subrutina addpPairsLP1, quan hem de fer una parella. (No s'ha d'escriure el codi de tota la subrutina).

```
;;;;;
; Aparellar nombres iguals des de l'esquerra de la matriu (m) i acumular els punts al
; marcador sumant el punts de les parelles que s'hagin fet. Recórrer la matriu per files
; d'esquerra a dreta i de dalt a baix. Quan es trobi una parella, dos caselles
; consecutives amb el mateix nombre, ajuntem la parella posant la suma dels nombres de
; la parella a la casella de l'esquerra i ; un 0 a la casella de la dreta i acumularem
; aquesta suma (punts que es guanyen). Si una fila de la matriu és:
; [8,4,4,2] i moved = 0, quedarà [8,8,0,2], p = (4+4)= 8 i moved = 1.
; Si al final s'ha ajuntat alguna parella (punts>0), posarem la variable (moved) a 1
; per indicar que s'ha mogut algun nombre i actualitzarem la variable (score) amb
; els punts obtinguts de fer les parelles.
; Per a accedir a les matrius des d'assemblador cal calcular l'índex a partir de la
; fila i la columna cridant la subrutina calcIndexP1. m[row][col], en C, és equivalent
; a WORD[m+eax], en assemblador, si eax = ((row*DimMatrix)+(col))*2. m[1][2] és [m+12].
; No s'ha de mostrar la matriu.
; Variables utilitzades:
; moved   : Per a indicar si s'han fet canvis a la matriu.
; score   : Punts acumulats fins el moment.
; m       : matriu 4x4 on hi han el números del tauler de joc.
; row     : fila per a accedir a la matriu m.
; col     : columna per a accedir a la matriu m.
; indexMat: índex per a accedir a
; Paràmetres d'entrada : Cap.
; Paràmetres de sortida: Cap.
;;;;;
addPairsLP1:
    ...
    mov r10, 0                      ;p = 0
    mov rbx, 0
    mov r8d, 0                      ;i = r8d
    addPairsLP1_Rows:
    mov r9d , 0                     ;j = r9d
        addPairsLP1_Cols:
        mov DWORD[row], r8d
        mov DWORD[col], r9d
        call calcIndexP1
        mov eax, DWORD[indexMat]
        mov ecx, eax                ;índex de [i][j]
        mov bx, WORD[m+ecx]
        cmp bx, 0                   ;m[i][j]!=0
        je  addPairsLP1_EndIf
            inc DWORD[col]
            call calcIndexP1
            mov eax, DWORD[indexMat]
            cmp bx, WORD[m+eax]     ;m[i][j]==m[i][j+1]
            jne addPairsLP1_EndIf
                shl bx,1                    ;m[i][j]*2
                mov WORD[m+ecx], bx         ;m[i][j]  = m[i][j]*2
                mov WORD[m+eax], 0          ;m[i][j+1]= 0
                add r10w, bx                ;p = p + m[i][j]
                inc r9d                     ;j++
        addPairsLP1_EndIf:
        inc r9d
        cmp r9d, DimMatrix-1        ;j<DimMatrix-1
        jl  addPairsLP1_Cols
    addPairsLP1_Next:
    inc r8d
    cmp r8d, DimMatrix              ;i<DimMatrix
    jl  addPairsLP1_Rows
    ...
    ret
```

Examen 2015/16-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/01/2016	18:30

1.2 Pràctica – Part opcional

Completar el codi de la subrutina showNumberP2. (Només completar els espais marcats, no es poden afegir o modificar altres instruccions)

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Aquesta subrutina es dóna feta. NO LA PODEU MODIFICAR.
; Mostrar un caràcter en la pantalla, rebut com a paràmetre al
; registre (dil), en la posició on està el cursor,
; cridant a la funció printch_C.
; Variables utilitzades: Cap.
; Paràmetres d'entrada : rdi (dil): Caràcter que volem mostrar a la pantalla.
; Paràmetres de sortida: Cap.
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
printchP2:

;;;;;
; Convertir el valor rebut com a paràmetre (edx) de tipus int(DWORD)
; als caràcters ASCII que representen el seu valor
; S'ha de dividir el valor entre 10, de forma iterativa,
; fins que el quocient de la divisió sigui 0.
; A cada iteració, el residu de la divisió que és un valor entre (0-9)
; indica el valor del dígit que s'ha de convertir a ASCII ('0' - '9')
; sumant '0' (48 decimal) per a poder-lo mostrar.
; S'han de mostrar els dígits (caràcters ASCII) des de la posició
; indicada per la fila (edi) i la columna (esi) rebuts com a paràmetre,
; posició de les unitats, cap a l'esquerra.
; Com el primer dígit que obtenim són les unitats, després les desenes, ..., per a
; mostrar el valor s'ha de desplaçar el cursor una posició a l'esquerra a cada iteració.
; Per a posicionar el cursor cridar a la subrutina gotoxyP2 i per a mostrar els caràcters
; a la subrutina printchP2 implementant correctament el pas de paràmetres.
; Variables utilitzades: Cap.
; Paràmetres d'entrada : rdi (edi): Fila on el volem mostrar a la pantalla.
;                        rsi (esi): Columna on el volem mostrar a la pantalla.
;                        rdx (edx): Valor que volem mostrar a la pantalla.
; Paràmetres de sortida: Cap.
;;;;;
showNumberP2:
    push rbp
    mov  rbp, rsp
       ...
    mov rax, rdx            ;Valor que volem mostrar
    showNumberP2_While:
        cmp eax, __0__
        jle showNumberP2_End
        mov edx, 0
        mov ebx, __10__
        div ebx         ;EAX=EDX:EAX/EBX, EDX=EDX:EAX mod EBX
        add dl, __'0'__    ;convertim el valor a caràcter ASCII
        ;Mostrar dígits
        call gotoxyP2
        push rdi
        mov  dil, __dl__
        call printchP2
        pop   __rdi__
        dec  esi     ;desplacem cursor a l'esquerra.
        jmp  showNumberP2_While

    showNumberP2_End:
    ...
    mov rsp, rbp
    pop rbp
    ret
```

Examen 2015/16-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/01/2016	18:30

Pregunta 2

2.1

L'estat inicial del computador CISCA just abans de començar l'execució de cada fragment de codi (a cada apartat) és el següent:

R1 = 00000010h R5 = 00000028h R10 = 00000050h	M(00000078h) = 810077E0h M(00000800h) = 00000810h	Z = 0, C = 0, S = 0, V = 0

Completeu l'estat del computador després d'executar cada codi (indiqueu els valors dels registres en hexadecimal).
Suposeu que l'adreça simbòlica V val 800h.

```
a )

MOV R2, 0000FFFFh
SAL R2, R1
NOT R2


R2= 0000FFFFh
R2=FFFF0000h
R2= 0000FFFFh

R2= 0000FFFFh
C=0, V=0, S=1, Z=0
```

```
b)

ADD R5, R10
MOV R5, [R5]
AND [V], R5


R5= 00000078h
R5= 810077E0h
[V]= 00000000h

R5= 810077E0h
[V]= 00000000h
C=0, V=0, S=0, Z=1
```

Examen 2015/16-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/01/2016	18:30

2.2

Suposem que tenim el vector V de 10 elements de 32 bits. Completar la traducció del programa en assemblador CISCA perquè executi l'algorisme d'alt nivell mostrat. (Hem deixat 6 espais per omplir)

```
i= 9;
while i>=0 do  {
      if (V [i] > 10)  V[i] = V[i]-1;
      else V[i] = 0;
      i= i-1;
};
```

```
      MOV  R1, 36
COND: CMP  R1  , 0
      JL   END
      CMP  [V+R1], 10
      JLE  ELSE
      SUB  [V+R1], 1
      JMP  NEXT
ELSE: MOV  [V+R1], 0
NEXT: SUB  R1, 4
      JMP  COND
END:
```

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/01/2016	18:30

2.3

Donat el següent fragment de codi d'un programa en llenguatge assemblador de CISCA:

```
        CMP R0, [A]
        JE END
        MOV R0, 42
END:
```

Traduïu-ho a llenguatge màquina i expresseu-ho en la següent taula. Suposeu que la primera instrucció del codi s'assembla a partir de l'adreça 00012C00h (que és el valor del PC abans de començar l'execució del fragment de codi). Suposeu que l'adreça simbòlica A val 00004000h. En la següent taula useu una fila per codificar cada instrucció. Si suposem que la instrucció comença en l'adreça @, el valor Bk de cadascun dels bytes de la instrucció amb adreces @+k per a k=0, 1,… s'ha d'indicar en la taula en hexadecimal en la columna corresponent (recordeu que els camps que codifiquen un desplaçament en 2 bytes o un immediat o una adreça en 4 bytes ho fan en format little endian, això cal tenir-ho en compte escrivint els bytes de menor pes, d'adreça més petita, a l'esquerra i els de major pes, adreça major, a la dreta). Completeu també la columna @ que indica per a cada fila l'adreça de memòria del byte B0 de la instrucció que es codifica en aquesta fila de la taula.

A continuació us donem com a ajuda les taules de codis:

Taula de codis d'instrucció

B0	Instrucció
41h	JE
26h	CMP
10h	MOV

Taula de modes d'adreçament (Bk<7..4>)

Camp mode Bk<7..4>	mode
0h	Immediat
1h	Registre
2h	Memòria
3h	Indirecte
4h	Relatiu
5h	Indexat
6h	Relatiu a PC

Taula de modes d'adreçament (Bk<3..0>)

Camp mode Bk<3..0>	Significat
Nº registro	Si el mode ha d'especificar un registre
0	No s'especifica registre.

Examen 2015/16-1

Assignatura	**Codi**	**Data**	**Hora inici**
Estructura de computadors	05.573	13/01/2016	18:30

		Bk per a k=0..10										
@	**Assemblador**	0	1	2	3	4	5	6	7	8	9	10
00012C00h	CMP R0, [A]	26	10	20	00	40	00	00				
00012C07h	JE END	41	60	07	00							
00012C0Bh	MOV R0, 42	10	10	00	2A	00	00	00				
00012C12h												

Examen 2015/16-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/01/2016	18:30

Pregunta 3

3.1. Memòria cau

Memòria cau d'assignació directa
Tenim un sistema de memòria en el que tots els accessos es fan a paraula (no ens importa quina és la mida d'una paraula). Suposarem que l'espai d'adreces de memòria es descompon en blocs de 8 paraules. Cada bloc comença en una adreça múltiple de 8. Així, el bloc 0 conté les adreces 0, 1, 2, 3, 4, 5, 6, 7, el bloc 1, les adreces 8, 9, 10, 11, 12, 13, 14, 15, i el bloc N les adreces 8*N, 8*N+1, 8*N+2, 8*N+3, 8*N+4, 8*N+5, 8*N+6, 8*N+7.

Suposem que el sistema també disposa d'una memòria cau de 4 línies (on cada línia té la mida d'un bloc, es a dir, 8 paraules). Aquestes línies s'identifiquen com a línies 0, 1, 2 i 3. Quan es fa referència a una adreça de memòria principal, si aquesta adreça no es troba a la memòria cau, es porta tot el bloc corresponent des de la memòria principal a una línia de la memòria cau (així si fem referència a l'adreça 2 de memòria principal portarem el bloc format per les paraules 0, 1, 2, 3, 4, 5, 6, 7).

Suposem que el sistema fa servir una **política d'assignació directa**, de manera que cada bloc de la memòria principal només es pot portar a una línia determinada de la memòria cau.

L'execució d'un programa genera la següent llista de lectures a memòria:

0, 1, 2, 12, 54, 55, 56, 64, 17, 18, 19, 32, 4, 6, 65, 66, 20, 56, 42, 50

3.1.1 La següent taula mostra l'estat inicial de la cau, que conté les primeres 32 paraules de la memòria (organitzades en 4 blocs).

Completar la taula per a mostrar l'evolució de la cau durant l'execució del programa. Per a cada accés cal omplir una columna indicant si es tracta d'un encert o una fallada.

Si és un encert escriurem E en la línia corresponent davant de les adreces del bloc, si és una fallada escriurem F i en aquest cas s'indicarà el nou bloc que es porta a la memòria cau en la línia que li correspongui, expressat de la forma b:e (a_0 - a_7) on b:número de bloc, e:etiqueta i (a_0 - a_7) són les adreces del bloc, on a_0 és la primera adreça del bloc i a_7 és la vuitena (darrera) adreça del bloc.

Línia	Estat Inicial	0	1	2	12	54
0	0:0 (0 - 7)	E 0:0 (0 - 7)	E 0:0 (0 - 7)	0:0 (0 - 7)	0:0 (0 - 7)	0:0 (0 - 7)
1	1:0 (8 - 15)	1:0 (8 - 15)	1:0 (8 - 15)	1:0 (8 - 15)	E 1:0 (8 - 15)	1:0 (8 - 15)
2	2:0 (16 - 23)	2:0 (16 - 23)	2:0 (16 - 23)	2:0 (16 - 23)	2:0 (16 - 23)	F 6:1 (48 - 55)
3	3:0 (24 - 31)	3:0 (24 - 31)	3:0 (24 - 31)	3:0 (24 - 31)	3:0 (24 - 31)	3:0 (24 - 31)

Línia	55	56	64	17	18	19
0	0:0 (0 - 7)	0:0 (0 - 7)	F 8:2 (64 - 71)	8:2 (64 - 71)	8:2 (64 - 71)	8:2 (64 - 71)
1	1:0 (8 - 15)	1:0 (8 - 15)	1:0 (8 - 15)	1:0 (8 - 15)	1:0 (8 - 15)	1:0 (8 - 15)
2	E 6:1 (48 - 55)	6:1 (48 - 55)	6:1 (48 - 55)	F 2:0 (16 - 23)	E 2:0 (16 - 23)	E 2:0 (16 - 23)
3	3:0 (24 - 31)	F 7:1 (56 - 63)	7:1 (56 - 63)	7:1 (56 - 63)	7:1 (56 - 63)	7:1 (56 - 63)

Examen 2015/16-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/01/2016	18:30

Línia	32		4		6		65		66		20	
0	F	4:1 (32 - 39)	F	0:0 (0 - 7)	E	0:0 (0 - 7)	F	8:2 (64 - 71)	E	8:2 (64 - 71)		8:2 (64 - 71)
1		1:0 (8 - 15)		1:0 (8 - 15)		1:0 (8 - 15)		1:0 (8 - 15)		1:0 (8 - 15)		1:0 (8 - 15)
2		2:0 (16 - 23)		2:0 (16 - 23)		2:0 (16 - 23)		2:0 (16 - 23)		2:0 (16 - 23)	E	2:0 (16 - 23)
3		7:1 (56 - 63)		7:1 (56 - 63)		7:1 (56 - 63)		7:1 (56 - 63)		7:1 (56 - 63)		7:1 (56 - 63)

Línia	56		42		50							
0		8:2 (64 - 71)		8:2 (64 - 71)		8:2 (64 - 71)						
1		1:0 (8 - 15)	F	5:1 (40 - 47)		5:1 (40 - 47)						
2		2:0 (16 - 23)		2:0 (16 - 23)	F	6:1 (48 - 55)						
3	E	7:1 (56 - 63)		7:1 (56 - 63)		7:1 (56 - 63)						

3.1.2 a) Quina és la taxa d'encerts (T_e) ?

T_e = 11 encerts / 20 accessos = 0,55

3.1.2 b) Suposem que el temps d'accés a la memòria cau, o temps d'accés en cas d'encert (t_e), és de 4 ns i el temps total d'accés en cas de fallada (t_f) és de 20 ns. Considerant la taxa de fallades obtinguda a la pregunta anterior, quin és el temps mitja d'accés a memòria (t_m) ?

$t_m = T_e \times t_e + (1-T_e) \times t_f$ = 0,55 * 4 ns + 0,45 * 20 ns = 2,2 ns + 9 ns = 11,2 ns

Examen 2015/16-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/01/2016	18:30

3.2 Sistema d'E/S

3.2.1 E/S programada

Es vol analitzar el rendiment de la comunicació de dades entre la memòria d'un processador i un port USB, utilitzant E/S programada, amb les següents característiques:

- Velocitat de transferència del dispositiu d'E/S v_{transf} = 2 MBytes/s = 2000 Kbytes/s
- Temps de latència mitjà del dispositiu $t_{latència}$ = 0
- Adreces dels **registres d'estat** i **dades** del controlador d'E/S: 0A00h i 0A04h
- El bit del **registre d'estat** que indica que el controlador del port d'E/S està disponible és el bit 2, o sigui el tercer bit menys significatiu (quan val 1 indica que està disponible)
- Processador amb una freqüència de rellotge de 2 GHz, el temps de cicle t_{cicle} = 0,5 ns.
- El processador pot executar 2 instruccions per cicle de rellotge
- Transferència de **escriptura** des de memòria al port d'E/S
- Transferència de N_{dades}= 160000 dades
- La mida d'una dada és m_{dada} = 4 bytes
- Adreça inicial de memòria on resideixen les dades: C0000000h

a) El següent codi realitzat amb el joc d'instruccions CISCA realitza la transferència descrita abans mitjançant la tècnica d'E/S programada. Completeu el codi.

```
1.        MOV R3, _160000_
2.        MOV R2, C0000000h
3.Bucle:  _IN_     R0, [0A00h]  ; llegir 4 bytes
4.        AND R0, _00000100b_
5.        _JE_Bucle
6.        MOV R0,_[R2]_         ; llegir 4 bytes
7.        ADD R2, 4
8.        OUT _[0A04h]_, R0     ; escriure 4 bytes
9.        SUB R3, 1
10.       JNE Bucle
```

3.2.2) Quant temps dura la transferència del bloc de dades t_{transf_bloc}?

t_{transf_bloc} = $t_{latència}$ + (N_{dades} * t_{transf_dada})
$t_{latència}$ = 0
N_{dades} = 160000
t_{transf_dada} = m_{dada} / v_{transf} = 4 Bytes / 2000 Kbytes/s = 0,002 ms
t_{transf_bloc} = 0 + (160000 * 0,002 ms) = 320 ms = 0,32 s

3.2.3) Si volguéssim fer servir el mateix processador i el mateix programa però amb un dispositiu d'E/S més ràpid, quina és la màxima taxa o velocitat de transferència del nou dispositiu que es podria suportar sense que el dispositiu s'hagués d'esperar?

t_{cicle} = 0,5 ns (nanosegons)
t_{instr} = 0,5 ns / 2 = 0,250 ns

El mínim nombre d'instruccions que ha d'executar el programa per a cada dada transferida són les 8 instruccions: 3, 4, 5, 6, 7, 8, 9 i 10. Executar les 8 instruccions requereix 8 * t_{instr} = 8 * 0,250 ns = 2 ns

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/01/2016	18:30

Per tant, el temps mínim per a transferir una dada és: 2 ns

Es poden transferir 4 bytes cada 2 ns, es a dir: $4 / 2 * 10^{-9} =$ 2000 Mbyte/s = 2 Gbytes/s

Examen 2015/16-1

Assignatura	Codi	Data	Hora inici
Estructura de computadors	05.573	13/01/2016	18:30

Pregunta 4

4.1

Si parlem del format de les instruccions, quins elements componen una instrucció?

• Codi d'operació: especifica l'operació que fa la instrucció.
• Operant font: per fer l'operació poden ser necessaris un o més operands font; un o més d'aquests operands poden ser implícits.
• Operant destinació: emmagatzema el resultat de l'operació realitzada. Pot estar explícit o implícit. Un dels operands font es pot utilitzar també com operant destinació.
• Adreça de la instrucció següent: especifica on està la instrucció següent que s'ha d'executar; sol ser una informació implícita, ja que el processador va a buscar automàticament la instrucció que es troba a continuació de l'última instrucció executada. Solament les instruccions de ruptura de seqüència especifiquen una adreça alternativa.

4.2

a) Un dels factors bàsics que fan que l'esquema de jerarquia de memòries funcioni satisfactòriament és la proximitat referencial. Quins tipus de proximitat referencial podem distingir? Explicar breument en que consisteix cadascun d'ells.

Distingim dos tips de proximitat referencial:

1) **Proximitat temporal**. És quan, en un interval de temps determinat, la probabilitat que un programa accedeixi de manera repetida a les mateixes posicions de memòria és molt gran.
La proximitat temporal és deguda principalment a les estructures iteratives; un bucle executa les mateixes instruccions repetidament, de la mateixa manera que les crides repetitives a subrutines.

2) **Proximitat espacial**. És quan, en un interval de temps determinat, la probabilitat que un programa accedeixi a posicions de memòria properes és molt gran.
La proximitat espacial és deguda principalment al fet que l'execució dels programes és seqüencial – s'executa una instrucció darrere l'altra llevat de les bifurcacions–, i també a la utilització d'estructures de dades que estan emmagatzemades en posicions de memòria contigües.

b) Una manera d'optimitzar les operacions d'E/S per DMA consisteix a reduir el nombre de cessions i recuperacions del bus, mitjançant una modalitat de transferència anomenada mode ràfega. Quin és el funcionament en el cas d'una transferència del dispositiu a la memòria?

Cada cop que el mòdul d'E/S té una dada disponible el controlador de DMA l'emmagatzema en la memòria intermèdia i decrementa el registre comptador. Quan la memòria intermèdia és plena o el comptador ha arribat a zero, sol·licita el bus. Un cop el processador li cedeix el bus, escriu a memòria tot el conjunt de dades emmagatzemades en la memòria intermèdia, i fa tants accessos a memòria com dades tenim i actualitza el registre d'adreces de memòria en cada accés. En acabar la transferència del conjunt de dades allibera el bus.
Un cop acabada una ràfega, si el registre comptador no ha arribat a zero, comença la transferència d'una nova ràfega.