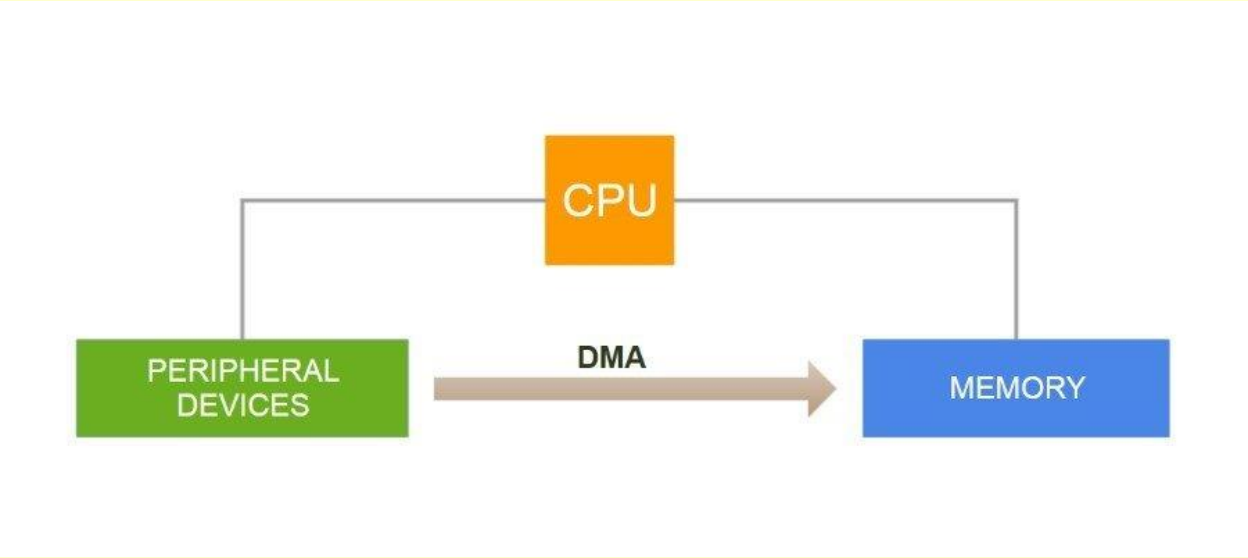
C6
CONTROLLER–ul DMA I8237A

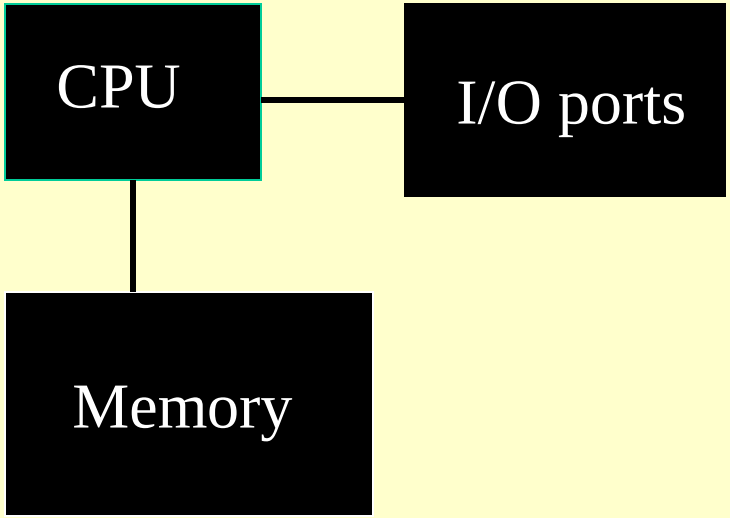
1. Principiul transferului DMA.
2. Pini.
3. Arhitectura.
4. Programare.
5. Utilizare in PC.
6. Teme



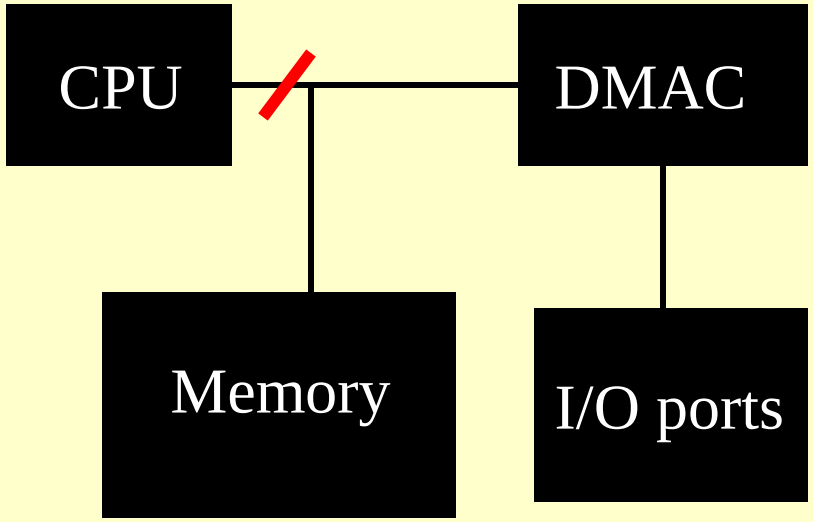
https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/performing-dma-in-64-bit-windows
https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/version-3-of-the-dma-operations-interface
https://www.intel.com/content/www/us/en/docs/programmable/683853/21-1-4-0-0/design-example-overview.html

1. PIO vs. DMA

PIO
(Programmable I/O, Polling)



DMA



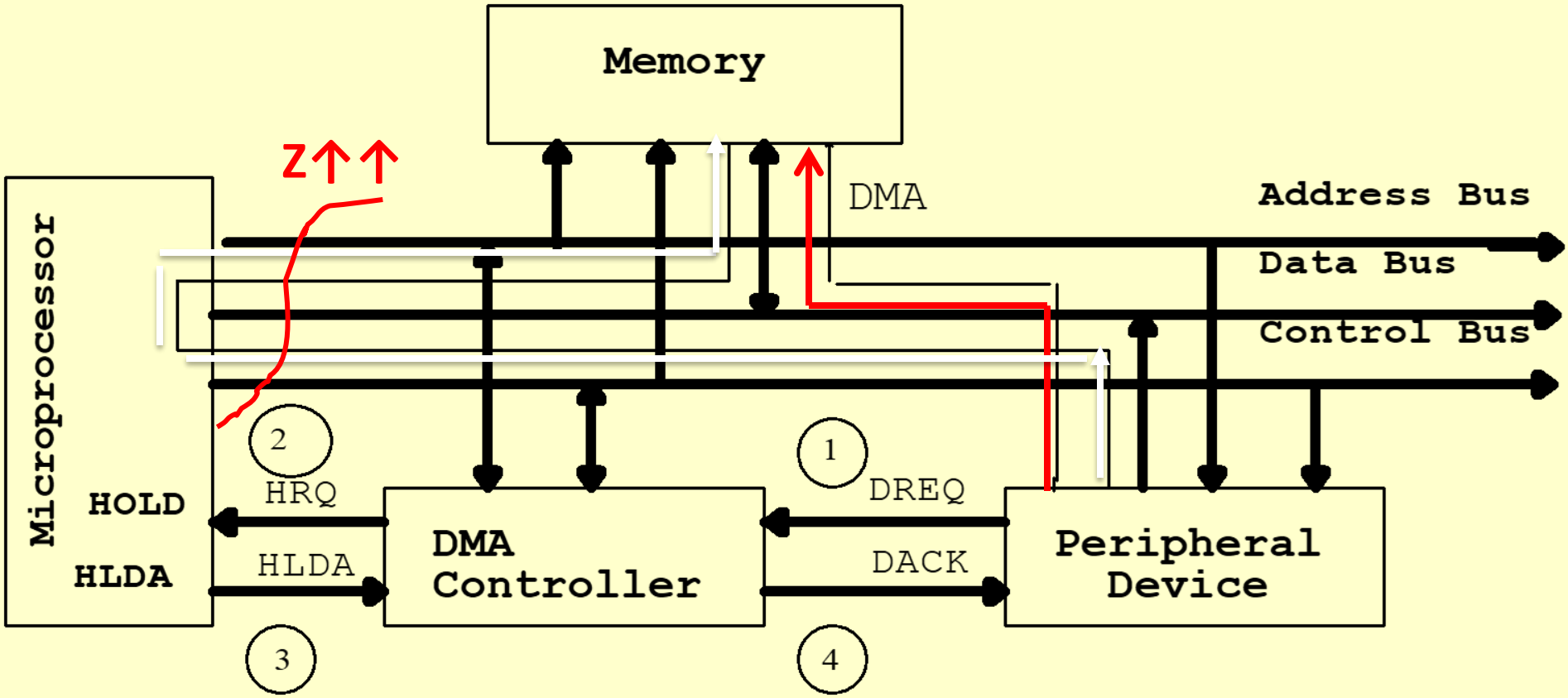
- De exemplu, secventa urmatoare de cod citeste 512 bytes de la un port de intrare 380H si scrie datele intr-un buffer in memorie:

```
        mov bx,offset buf ; destination address
        mov cx,512        ; count
        mov dx,380H       ; source port
loop:
        in al,dx          ; get byte from i/o port
        mov [bx],al       ; store in buffer
        inc bx            ; next memory location
        dec cx            ; decrement bytes left
        jnz loop
```

Tema. Rescrieti secventa de program folosind instructiunea INS .

!!

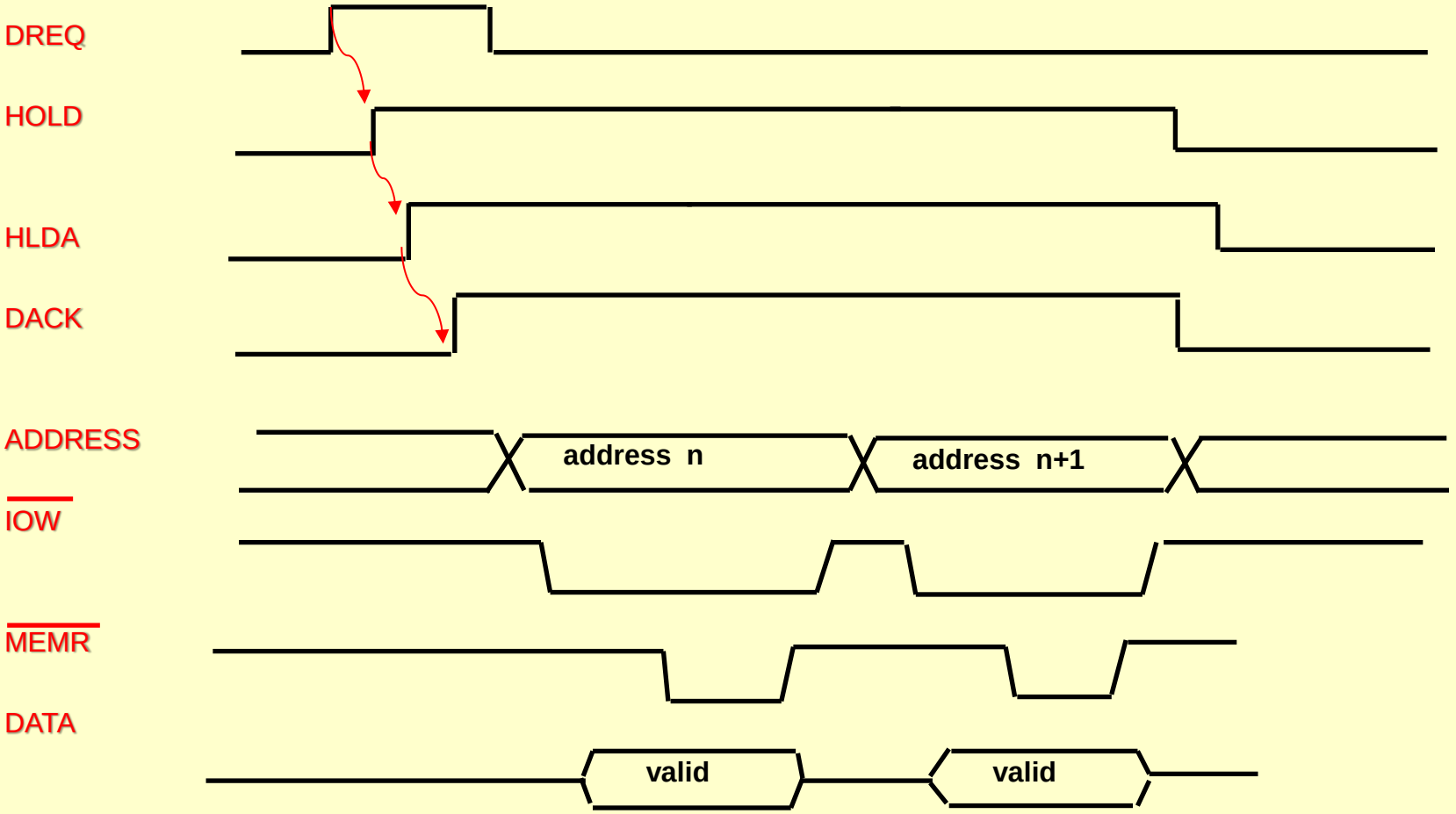
- PIO este mai vechi in comparatie cu DMA
- PIO cere mai multa implicare din partea CPU in comparatie cu DMA
- PIO este mult mai simplu in comparatie cu DMA
- Dispozitivele folosesc PIO cand DMA este problematic



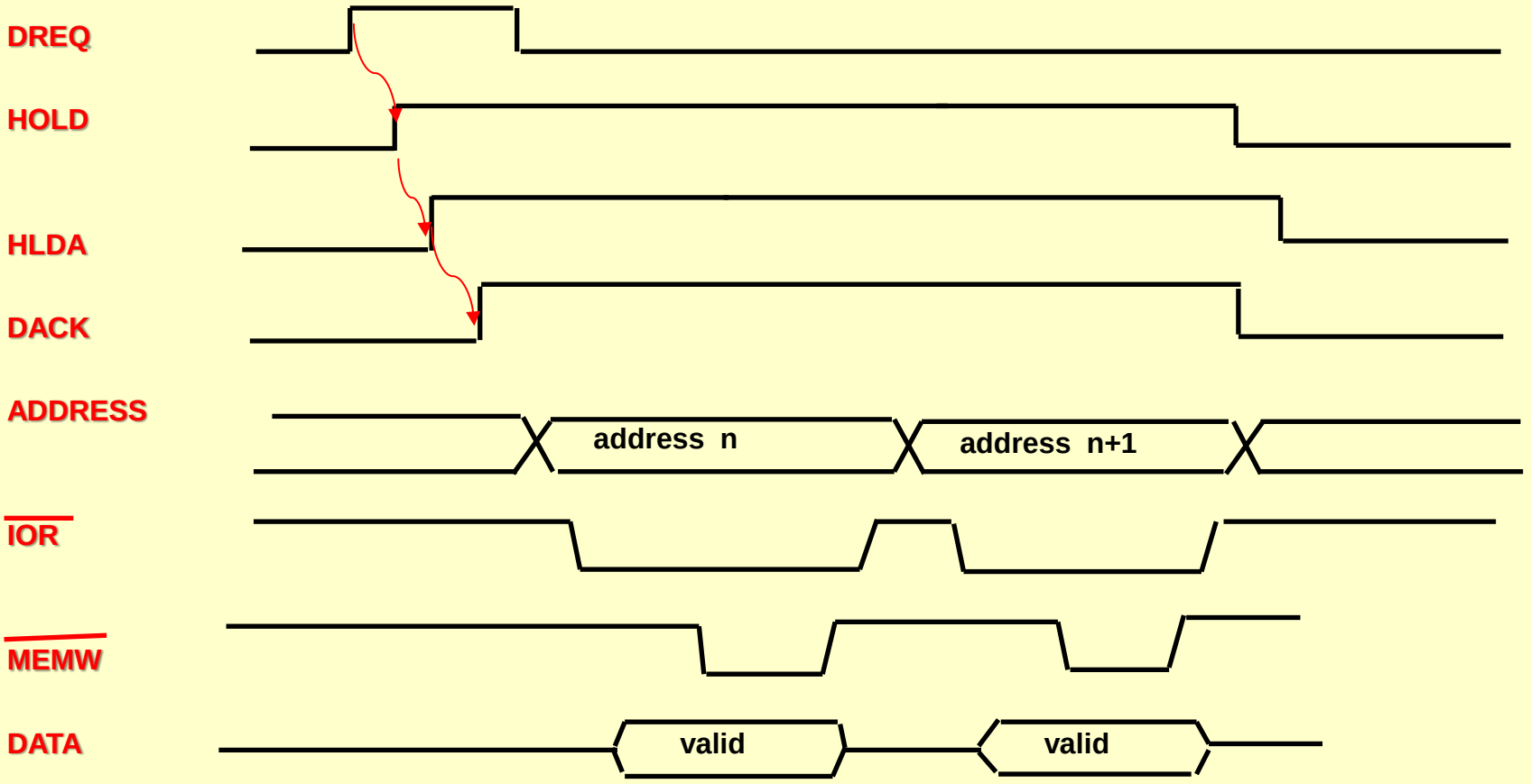
Principiul transferului de date prin acces direct la memorie

DMA - generic term used to refer to a transfer protocol where a peripheral device transfers information directly to/from memory, without the processor involvement

DMA Timing, Transfer Memorie-Periferic

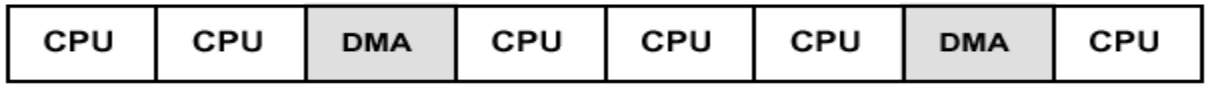


DMA Timing, Transfer Periferic-Memorie



Moduri de operare DMA

Modul cu “furt” de ciclu (Cycle Stealing)



- DMAC preia controlul Bus-urilor ptr. fiecare octet/word de transferat, CPU nu trebuie sa astepte poate executa alte task-uri

Modul avalansa (Burst)

- un bloc de date este transferat inainte de a returna controlul bus-urilor CPU

– Eficienta mai mare in transferul de date

Mod intercalat/transparent (interleaved) - în această tehnică, controlerul DMA preia busul sistem atunci când microprocesorul nu-l folosește. ("*Hidden DMA data transfer mode*“)

- Canalele DMA se folosesc numai pe bus-ul ISA (EISA,VLB)
- Dispozitivele PCI nu folosesc canalele DMA standard
- DMA-ul standard este denumit uneori *"third party"* DMA
- Aceasta se refera la faptul ca DMA controllerul (3) realizeaza transferul (celelalte doua fiind emitatorul (1) si receptorul (2))
- "First party" DMA numit si "*bus mastering*" – perifericul preia controlul bus-ului sistemului si realizeaza transferul cu memoria
- *Daca orice cartela de I/O poate deveni master, atunci nu mai este necesar un DMA controller separat; fiecare cartela are functia de DMAC*

- Viteza de transfer a datelor este determinată de viteza dispozitivului de memorie sau un controller-ul DMA.

- În cazul în care t_{access} la memorie este de 50 ns, transferurile DMA apar la rate de până la 1/50 ns sau 20 MB /s

- În cazul în care controlerului DMA funcționeaza, la o frecventa maximă de 15 MHz, cu o memorie de 50 ns, rata maximă de transfer este de 15 MHz (67ns), deoarece controlerul DMA este mai lent decât memoria

- In multe cazuri, controlerul DMA încetinește viteza sistemului atunci când au loc transferuri !!!

2. Arhitectura DMAC

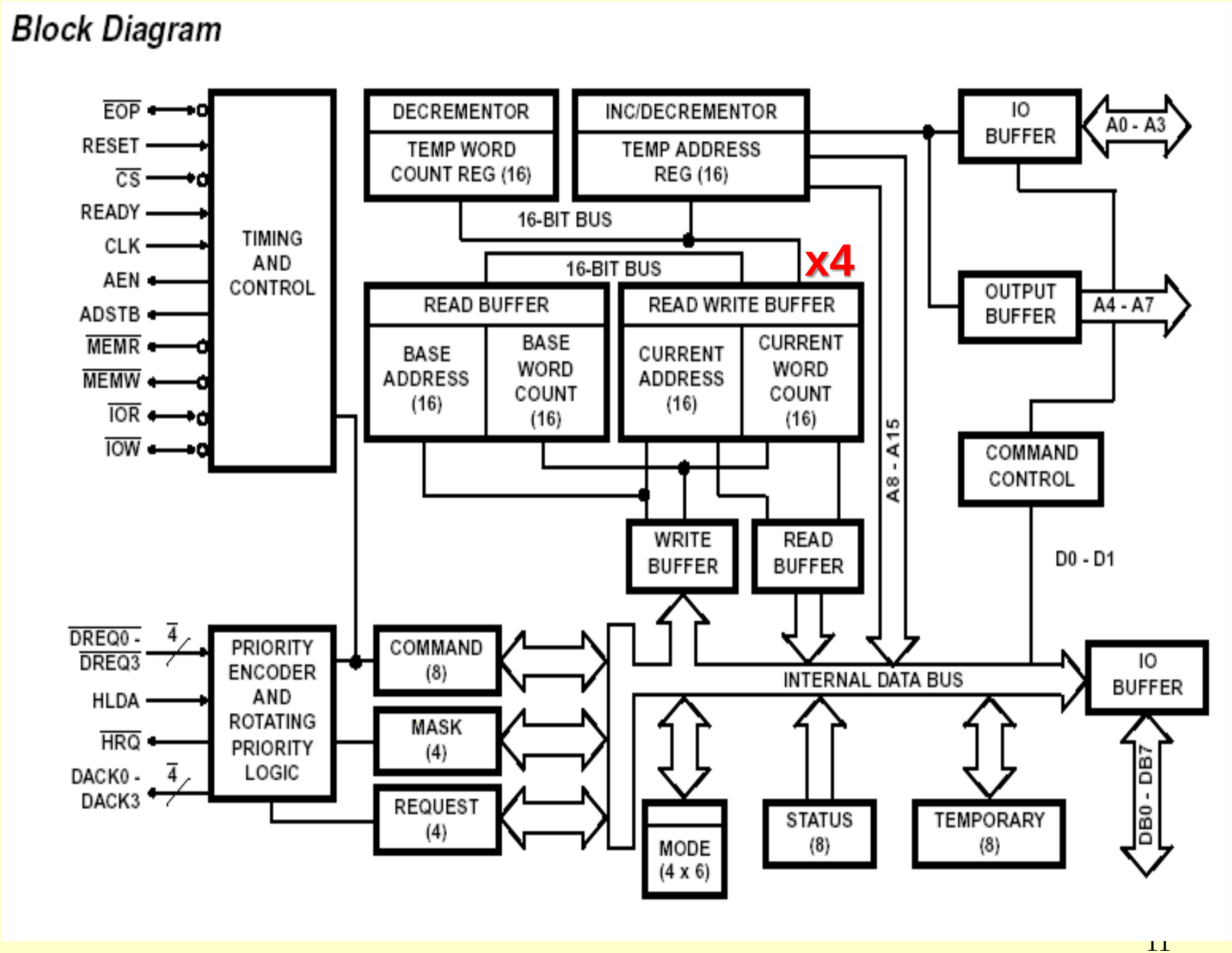
- DMAC a fost proiectat sa lucreze cu µP pe 8biti + un reg. extern 8 biti

- are 4 canale independente, permit max. 64kB transfer date

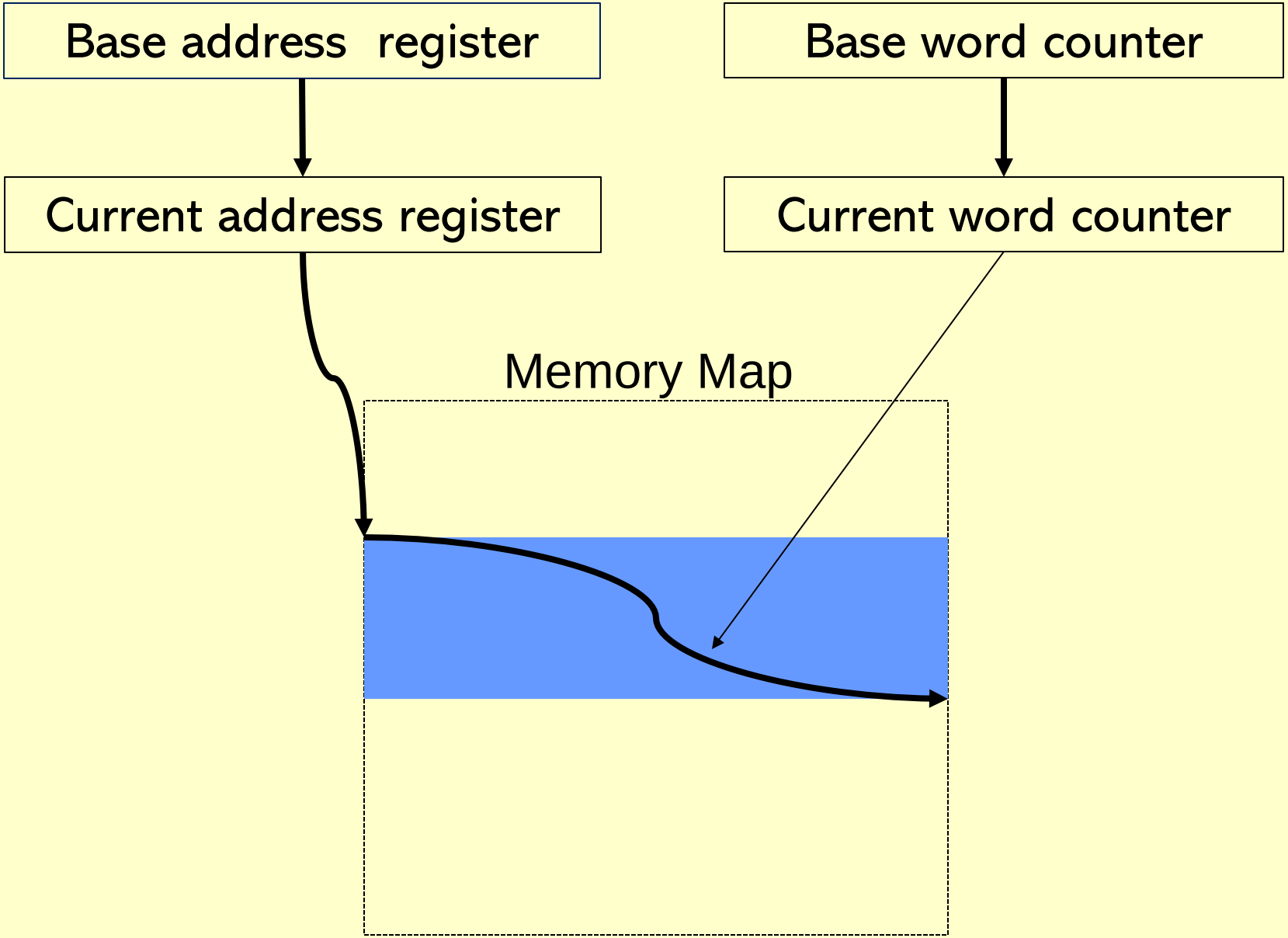
- 2 tipuri de cicluri: activ (master) si inactiv (slave) – se programeaza

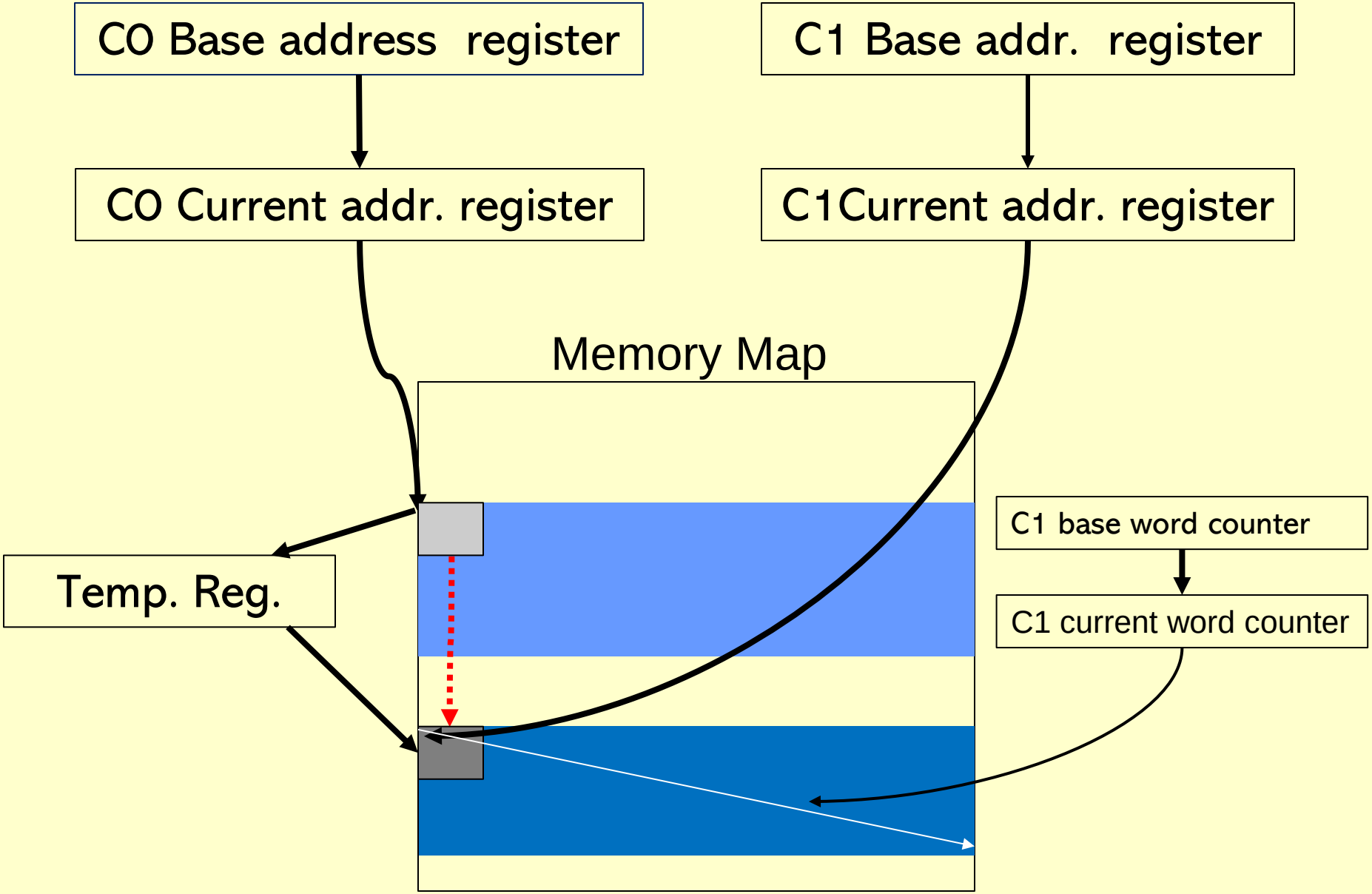
Blocuri ale DMAC:

- Bloc de temporizare si control
- Bloc comanda si control
- Bloc logica control prioritate
- Canale de transferDMA



Schema bloc internă a DMAC - I8237A


Base address register
Base word counter
Current address register
Current word counter
Memory Map



Transfer Memorie-Memorie prin DMA

REGISTRUL	DIMENSIUNE REG. (biţi)	NUMĂR REG.
Adresă initială	16	4
Contor cuvinte initiale	16	4
Adresă curentă	16	4
Contor cuvinte curente	16	4
Adresa temporară	16	1
Contor cuvinte temporar	16	1
Stare	8	1
Comandă	8	1
Temporar	8	1
Mod	6	4
Mască	4	1
Cerere	4	1

Canal	Port*	Port**	(R/W)	Registrul (word)
0	00H	C0h	W	Adresă iniţială canal 0 DMA
0	00H	C0h	R	Adresă curentă canal 0 DMA
0	01H	C2h	W	Contor iniţial canal 0 DMA
0	01H	C2h	R	Contor curent canal 0 DMA
1	02H	C4h	W	Adresă iniţială canal 1 DMA
1	02H	C4h	R	Adresă curentă canal 1 DMA
1	03H	C6h	W	Contor iniţial canal 1 DMA
1	03H	C6h	R	Contor curent canal 1 DMA
2	04H	C8h	W	Adresă iniţială canal 2 DMA
2	04H	C8h	R	Adresă curentă canal 2 DMA
2	05H	CAh	W	Contor iniţial canal 2 DMA
2	05H	CAh	R	Contor curent canal 2 DMA
3	06H	CCh	W	Adresă iniţială canal 3 DMA
3	06H	CCh	R	Adresă curentă canal 3 DMA
3	07H	CEh	W	Contor iniţial canal 3 DMA
3	07H	CEh	R	Contor curent canal 3 DMA

*circuitul “slave” la PC-AT (singurul DMAC la PC-XT)
**circuitul “master” la PC-AT (inexistent la PC-XT)

3. PINII DMAC 8237A

- *semnale de control şi sincronizare*

RESET(13) :CLK(12) :CS/(11): READY(6) : AEN(9) :ADSTB(8) : (MEMW/ (4) MEMR/ (3), /IOR/ (1) IOW/(2) :

- *semnale de cerere /răspuns si control bus–ului*

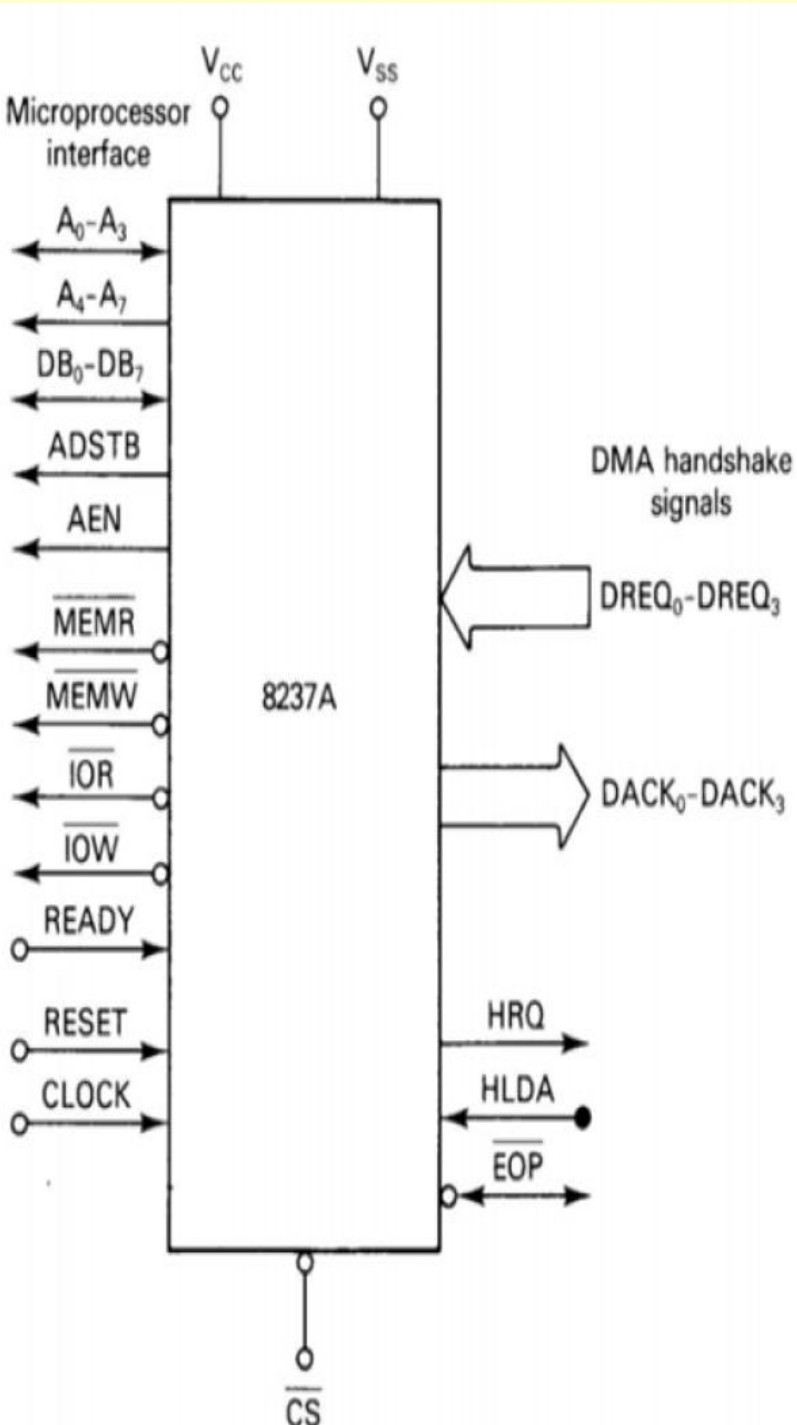
DREQi (19-16)
DACKi (36/37/14/15):
HRQ (10)
HLDA (7):

- *linii de adresă*

A0…A3 (29-26) : A4…A7(24-21) :

- *linii de date*

D0…D7 (31-35,38-40) :



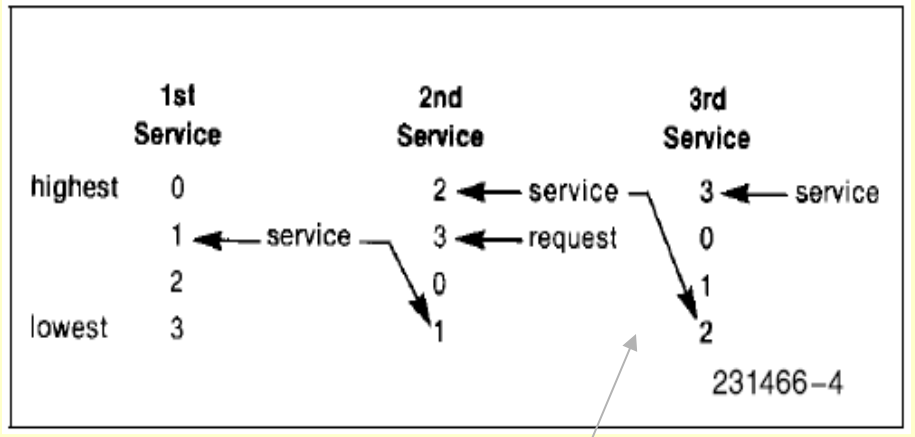
4. PROGRAMAREA DMAC 8237A

Register / Software command (#)	Port*	Port**	R / W Operation
Status register	08H	D0h	R (in al, 08h)
Command register	08H	D0h	W (out 08H, al)
Software requests register	09H	D2h	W (out 09H, al)
Individual masks register	0AH	D4h	W (out 0AH, al)
Mode register	0BH	D6h	W (out 0BH, al)
Internal flip-flop clear #	0CH	D8h	W (out 0CH, al)
Temporary register	0DH	Dah	R (in al, 0DH)
Reset I8237A –master clear #	0DH	Dah	W (out 0DH, al)
Clear mask register #	0EH	DCh	W (out 0EH, al)
Write bits of mask register	0FH	DEh	W (out 0FH, al)

* Slave circuit at PC-AT (the only DMAC at PC-XT)
** Master circuit at PC-AT (non-existing at PC-XT)
# Software commands

Registrul de comandă

7	6	5	4	3	2	1	0
DACKH	**DRQL**	**EWR**	**RPY**	**CTIM**	**DISAB**	**C0AHE**	**M-M**



M – M = 0 dezactivare transfer memorie – memorie
= 1 permite transfer memorie – memorie

C0AHE = 0 dezactivare menţinerea adresei canalului 0
= 1 permite menţinerea adresei canalului 0
= x dacă M – M = 0

DISAB = 0 activează I 8237A
= 1 dezactivează I 237A

CTIM = 0 ciclu normal
= 1 ciclu comprimat
= x dacă M – M = 1

DRQL = 0 DRQ activ pe 1
= 1 DRQ activ pe 0

RPY = 0 priorităţi fixe
= 1 rotire priorităţi

EWR = 0 impuls scriere întârziat
= 1 impuls scriere extins
= x dacă CTIM = 1

DACKH = 0 DACK activ pe 0
= 1 DACK activ pe 1

Registrul de cereri software

7	6	5	4	**3**	**2**	**1**	**0**
-	-	-	-	-	**SRQB**	**SC1**	**SC0**

SRQB = 0 ştergere bit cerere
= 1 setează bit cerere

SC1	SC0	Semnificatie
0	0	Selectez canal 0
0	1	Selectez canal 1
1	0	Selectez canal 2
1	1	Selectez canal 3

Registrul de măşti individuale

7	6	5	4	3	2	1	0
-	-	-	-	-	**SMKB**	**SC1**	**SC0**

SC1	SC0	Semnificatie
0	0	Selectez bit masca canal 0
0	1	Selectez bit masca canal 1
1	0	Selectez bit masca canal 2
1	1	Selectez bit masca canal 3

SMKB = 0 şterge bit mască
= 1 setează bit mască (blochează canal)

Registrul de mod

7	6	5	4	3	2	1	0
SM1	**SM0**	**DECR**	**AUTOI**	**SDT1**	**SDT0**	**SC1**	**SC0**

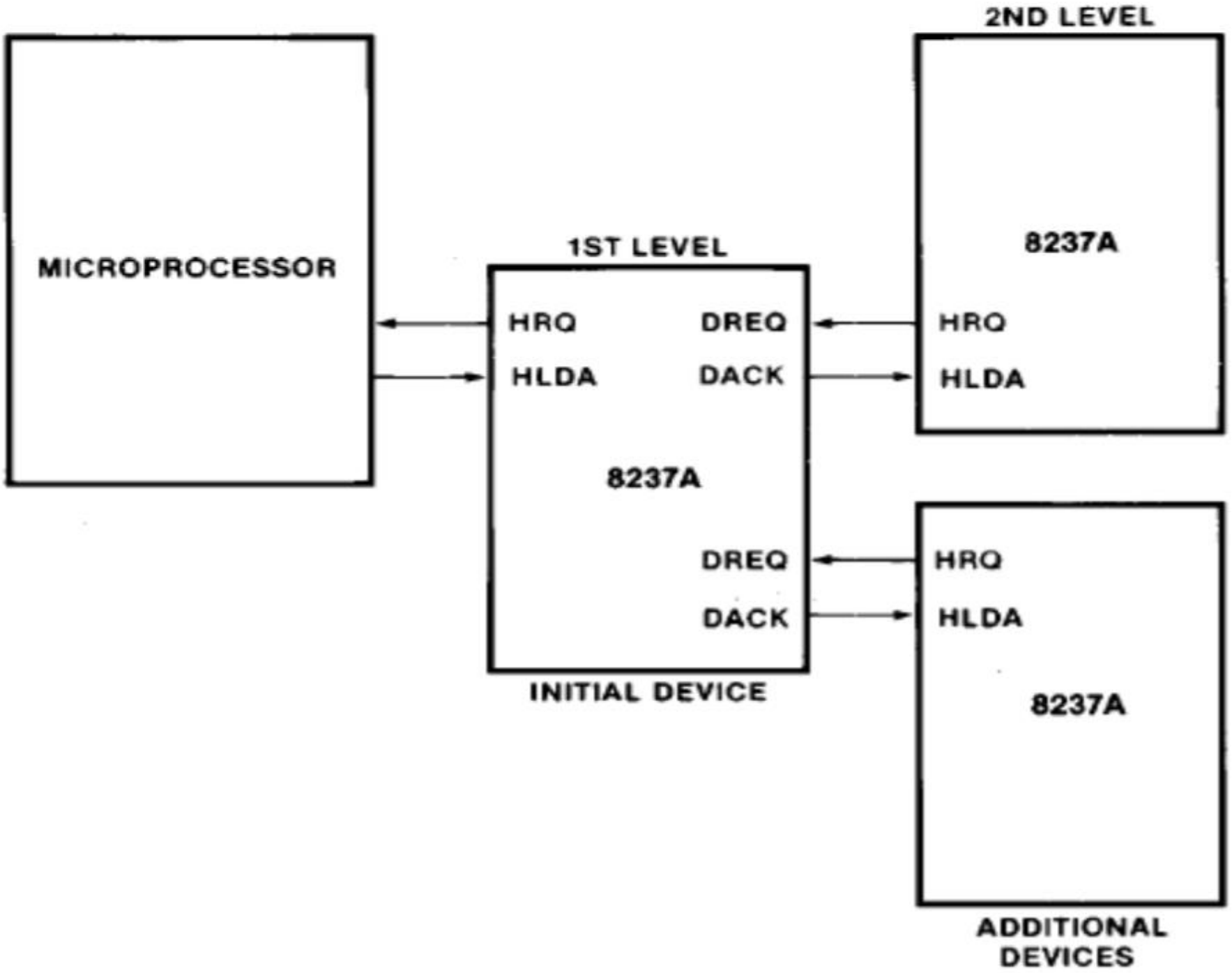
SM1	SM0	SEMNIFICATIE
0	0	Mod cerere
0	1	Mod octet
1	0	Mod bloc
1	1	Mod cascadă

SC1	SC0	SEMNIFICAŢIE
0	0	Selectează canal 0
0	1	Selectează canal 1
1	0	Selectează canal 2
1	1	Selectează canal 3

SDT1	SDT0	SEMNIFICATIE
0	0	Verificare
0	1	Scriere în memorie
1	0	Citire din memorie
1	1	Ilegală
X	X	Dacă SM0 = 0 şi SM1 = 1 (bloc)

AUTOI = 0 dezactivare autoiniţializare
= 1 permite autoiniţializare

DECR = 0 selectează incrementarea adresei
= 1 selectează decrementarea adresei


2ND LEVEL
MICROPROCESSOR
1ST LEVEL
8237A
HRQ
DREQ
HLDA
DACK
HRQ
HLDA
8237A
DREQ
HRQ
DACK
HLDA
INITIAL DEVICE
8237A
ADDITIONAL
DEVICES

Registrul de măşti (pentru toate canalele)

7	6	5	4	3	2	1	0
-	-	-	-	SMK3	SMK2	SMK1	SMK0

SMKi = 0 ştergere bit mască pentru canalul i
= 1setează bitul de mască pentru canalul i , i = 0 , 1 , 2 , 3

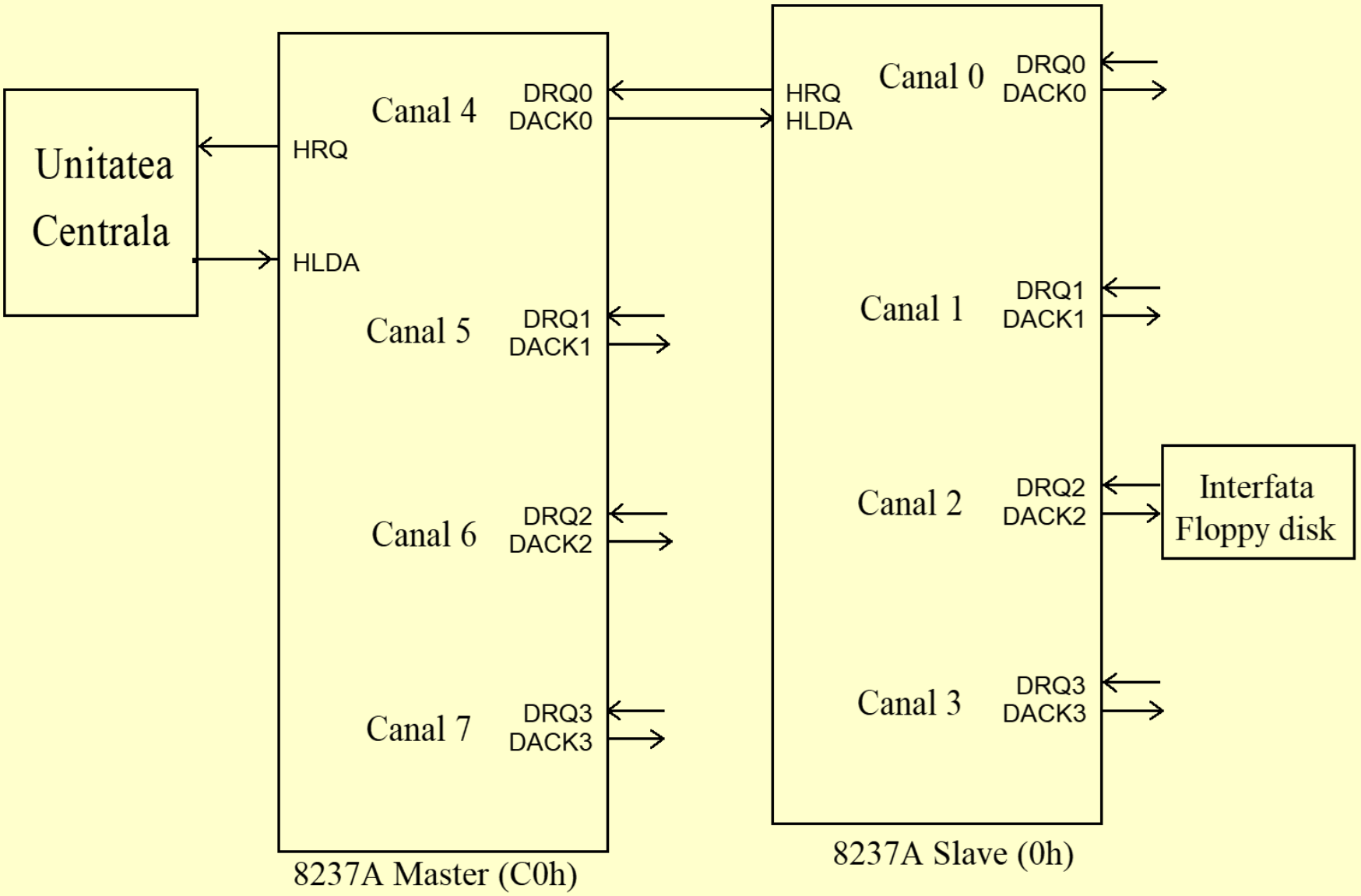
Registrul de stare

7	6	5	4	3	2	1	0
RQ3	RQ2	RQ1	RQ0	TC3	TC2	TC1	TC0

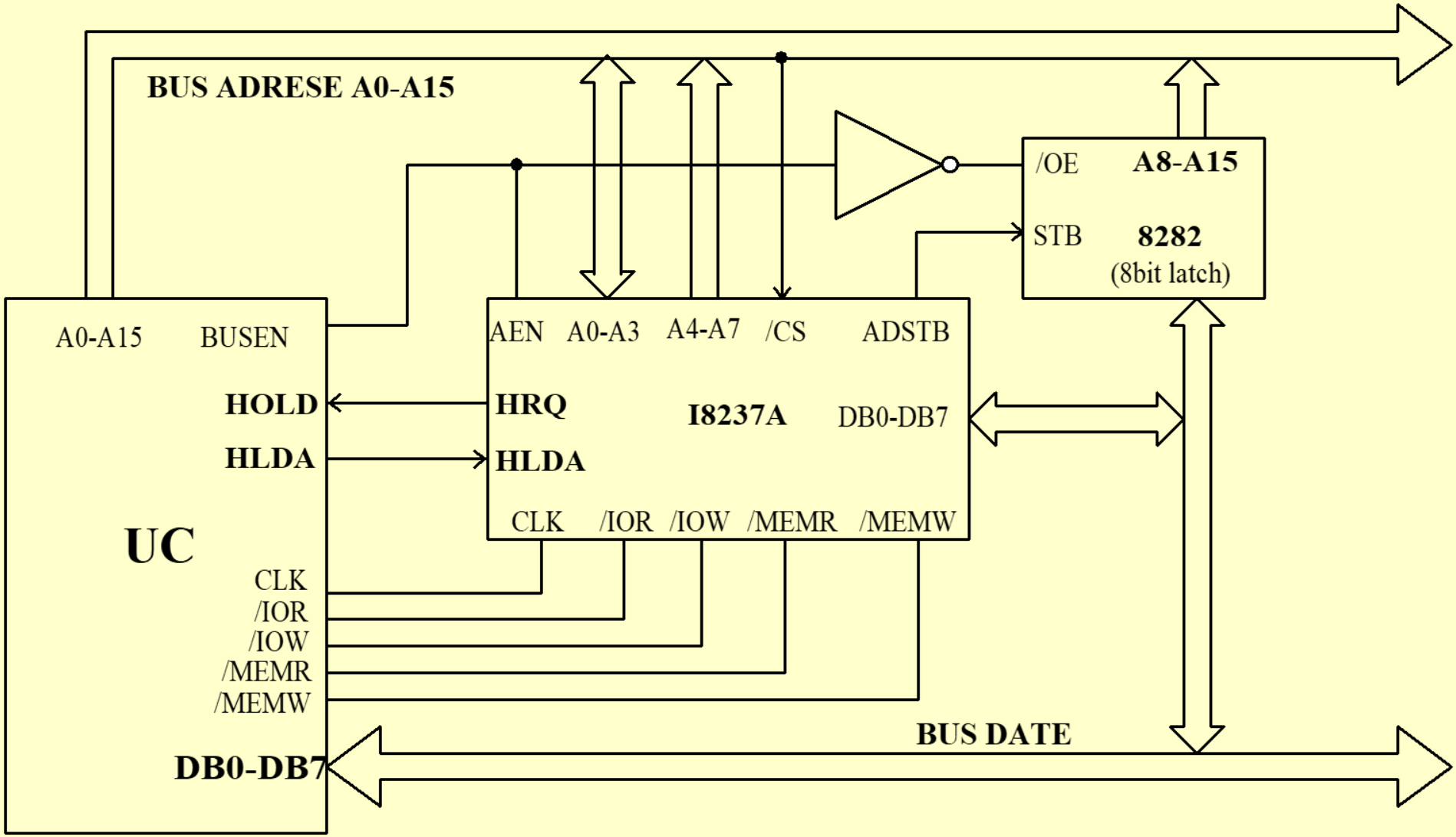
TCi = 1 canalul i are contor nul
= 0 canalul i nu are contor nul i = 0 , 1 , 2 , 3

RQi = 1 cerere transfer DMA pe canalul i
= 0 nu există cerere de transfer pe canalul i, i = 0 , 1 , 2 , 3

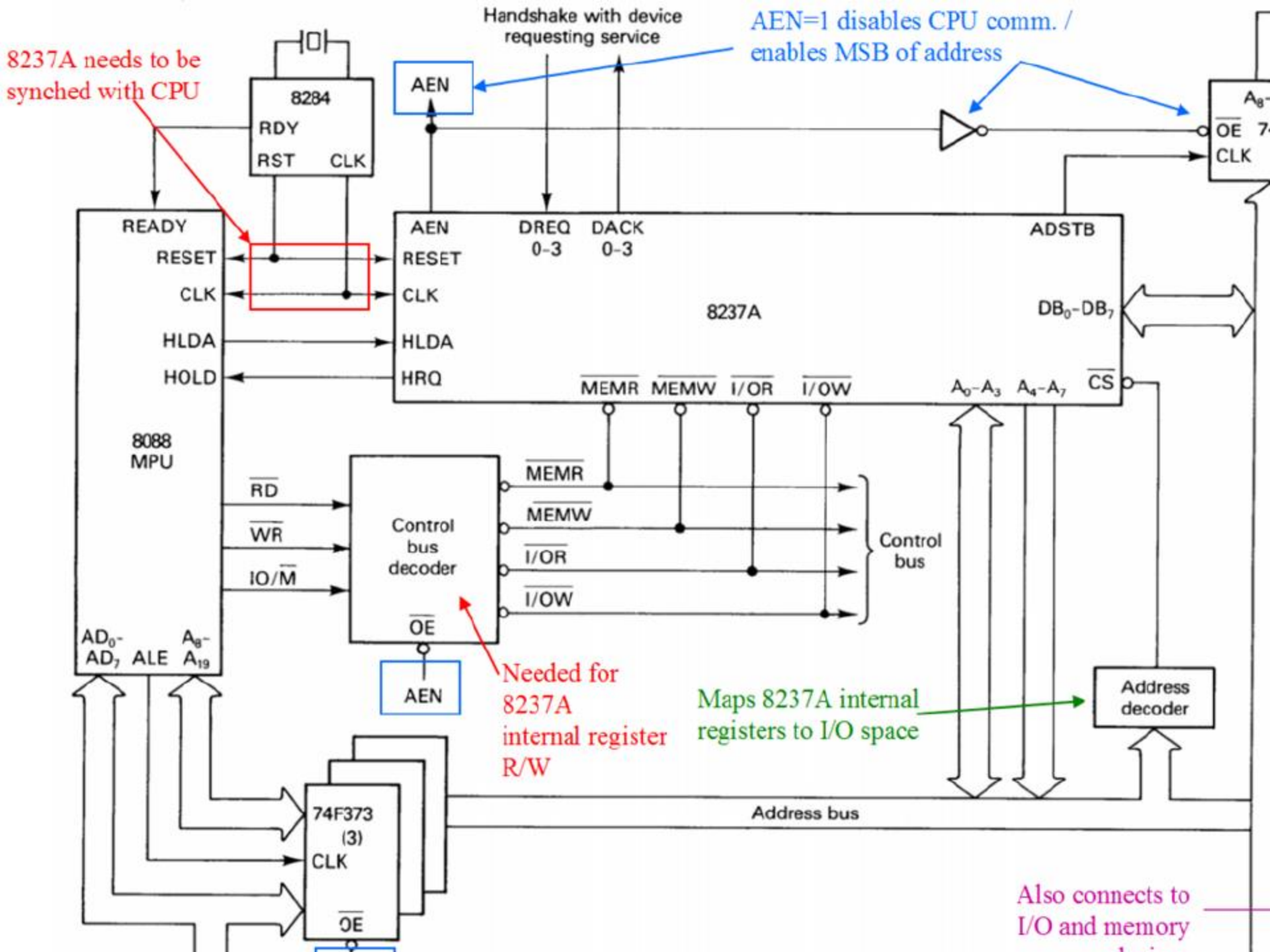
5. Utilizare DMA in PC

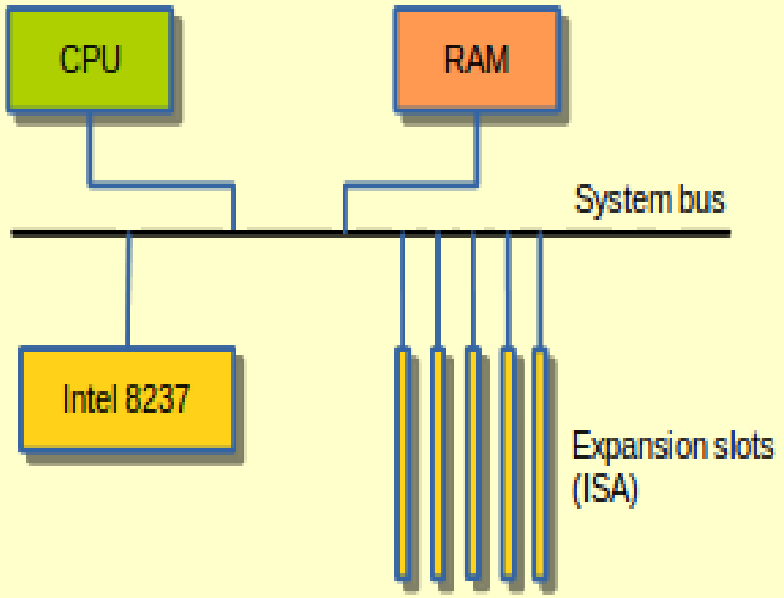


Conectarea DMAC în cascadă la PC-AT



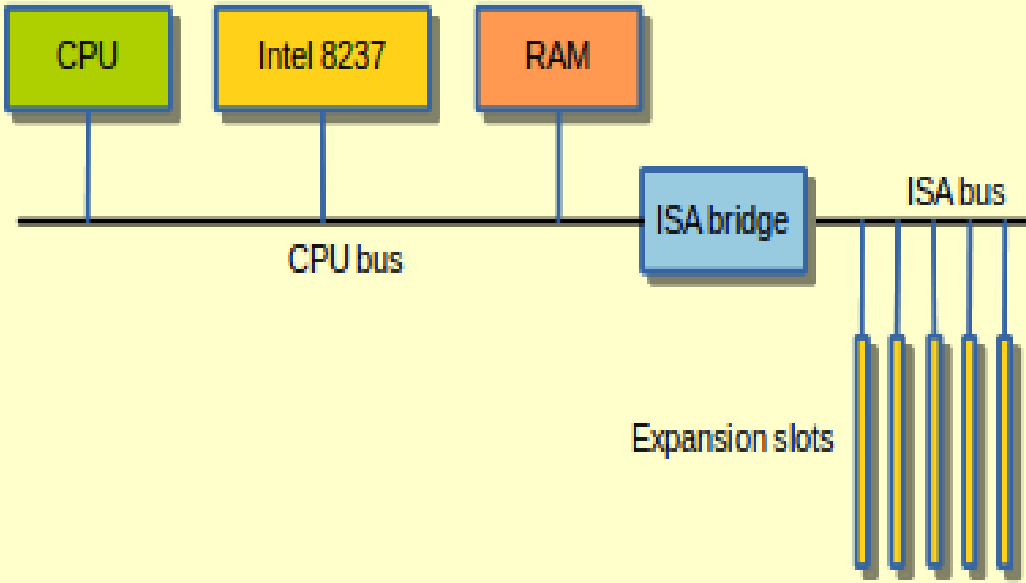
Interfatarea I8237A la sistem


8237A needs to be synched with CPU
Handshake with device requesting service
AEN=1 disables CPU comm. / enables MSB of address
8284
RDY
RST
CLK
AEN
READY
RESET
CLK
HLDA
HOLD
8088 MPU
AD0-AD7
ALE
A8-A19
AEN
DREQ 0-3
DACK 0-3
ADSTB
8237A
DB0-DB7
MEMR
MEMW
I/OR
I/OW
A0-A3
A4-A7
CS
HRQ
RD
WR
IO/M
Control bus decoder
OE
Control bus
Needed for 8237A internal register R/W
Maps 8237A internal registers to I/O space
Address decoder
74F373 (3)
Address bus
Also connects to I/O and memory

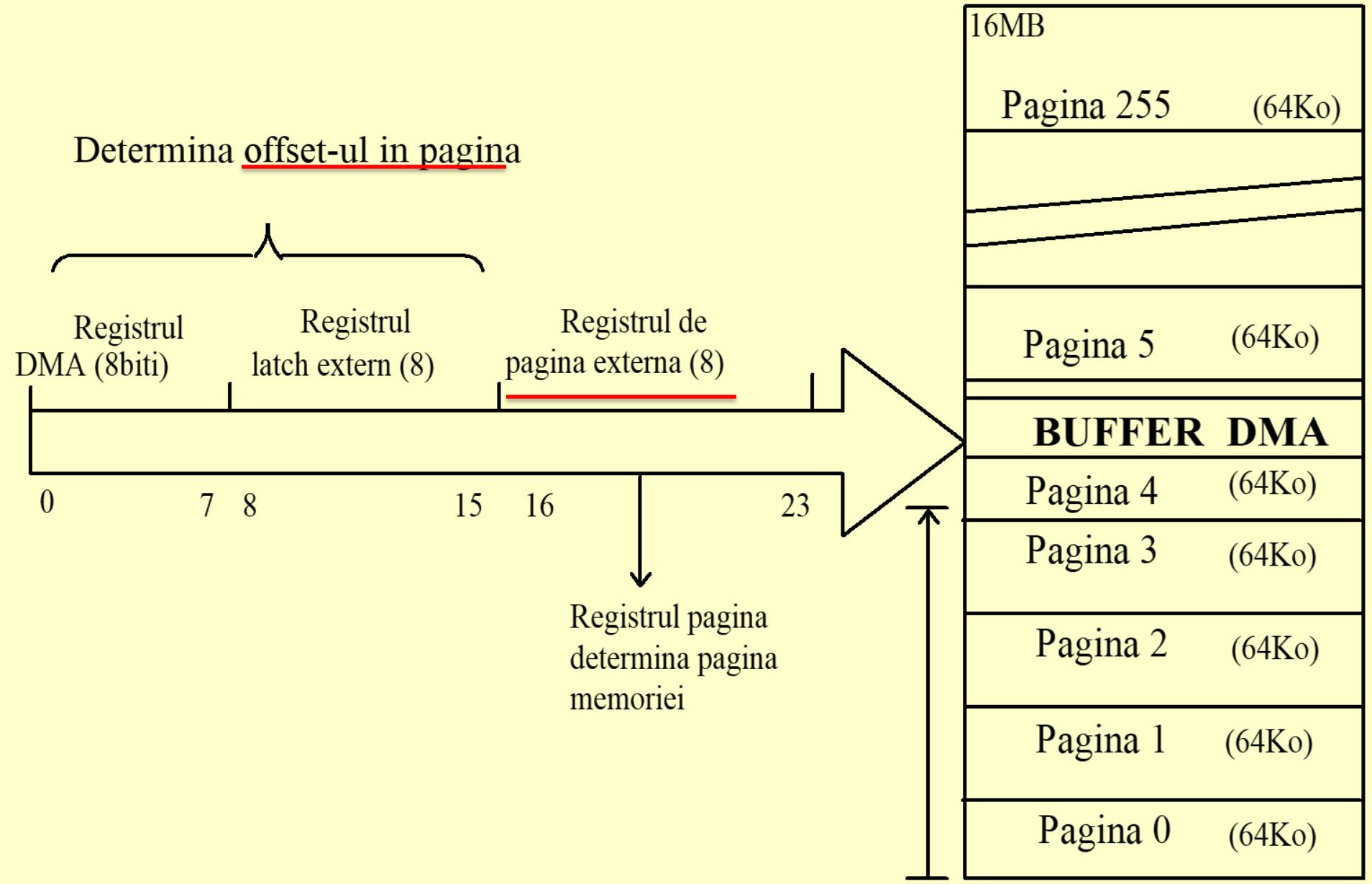


Single system bus architecture

vs



Separated CPU and ISA bus architecture



Generarea adresei de memorie la transferul DMA cu I8237A

PC-XT:

81h pentru canalul 2 DMA (biţi de adresă 16-19)
82h pentru canalul 3 DMA (biţi de adresă 16-19)
83h pentru canalele 0 şi 1 DMA (biţi de adresă 16-19)

PC-AT:

81h pentru canalul 2 DMA (biţi de adresă 16-23)
82h pentru canalul 3 DMA (biţi de adresă 16-23)
83h pentru canalul 1 DMA (biţi de adresă 16-23)
87h pentru canalul 0 DMA (biţi de adresă 16-23)
89h pentru canalul 6 DMA (biţi de adresă 17-23)
8bh pentru canalul 5 DMA (biţi de adresă 17-23)
8ah pentru canalul 7 DMA (biţi de adresă 17-23)
8fh pentru refresh DRAM

Canal DMA	8/16 biţi	Utilizare implicită	Alte utilizări
0	8	Nefolosit	-
1	8	Cartele Sunet (low DMA)	Adaptoare SCSI, port paralel ECP, Cartele reţea, modemuri vocale
2	8	Floppy Disk	Cartele acceleratoare de bandă
3	8	Nefolosit	Port paralel ECP, Adaptoare SCSI, cartele sunet(low DMA), cartele reţea, modemuri vocale
4	16	Cascadare	-
5	16	Cartele sunet (high DMA)	Adaptoare SCSI, cartele reţea
6	16	Nefolosit	cartele sunet(high DMA), cartele reţea
7	16	Nefolosit	cartele sunet(high DMA), cartele reţea

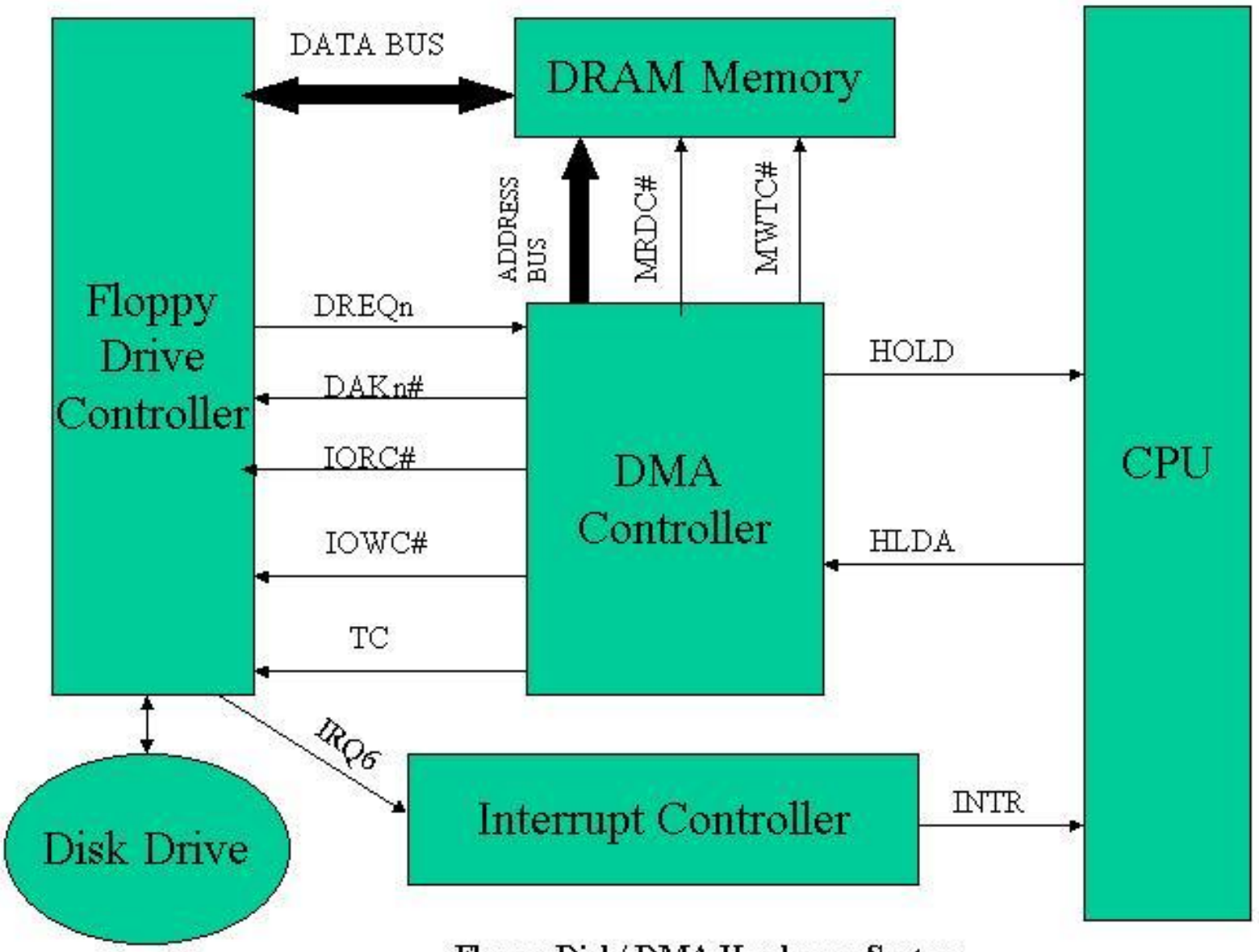
Etapele Programarii DMAC

Programarea DMA controller nu este un proces simplu, dar poate fi redus la urmatorii pasi:

- Gasim Pagina si Offsetul de pagina a blocului de memorie unde facem transferul
- Blocam canalul DMA (astfel nu accepta cereri in timpul reprogramarii)
- Sterg flagul flip/flop
- Scriu modul de transfer in registrul de mod
- Scriu offset-ul de pagina in registrul baza de adresa
- Scriu lungimea (-1 byte/word in registrul contor (0=1 byte, 65535=64Kbytes)
- Scriu pagina # in registrul pagina
- Validam canalul DMA
- Configuram dispozitivul vizat

OBSERVATII

- Trecerea la transferurile de date seriale în sisteme moderne a făcut DMA-ul sa fie mai puțin important
- Transferurile seriale de date pe bus-ul PCI Express, depășesc cu mult vitezele de transfer DMA.
- SATA (serial ATA) interfața pentru unitățile HD utilizează transferuri seriale la rata de 300 Mbps și a înlocuit transferurile DMA pentru HDD
- Transferurile seriale dintre componente, la principalele placi folosind conexiunea PCI Express > 20 Gb/s.
- *Transferurile DMA* sunt utilizate pentru comunicare intracore la procesoare și chiar de a copia datele din memoria unui computer în memoria unui alt computer prin rețea DMA la distanță
- Ex. **I/O Acceleration Technology** (**I/O AT**) DMA engine (an embedded DMAC)

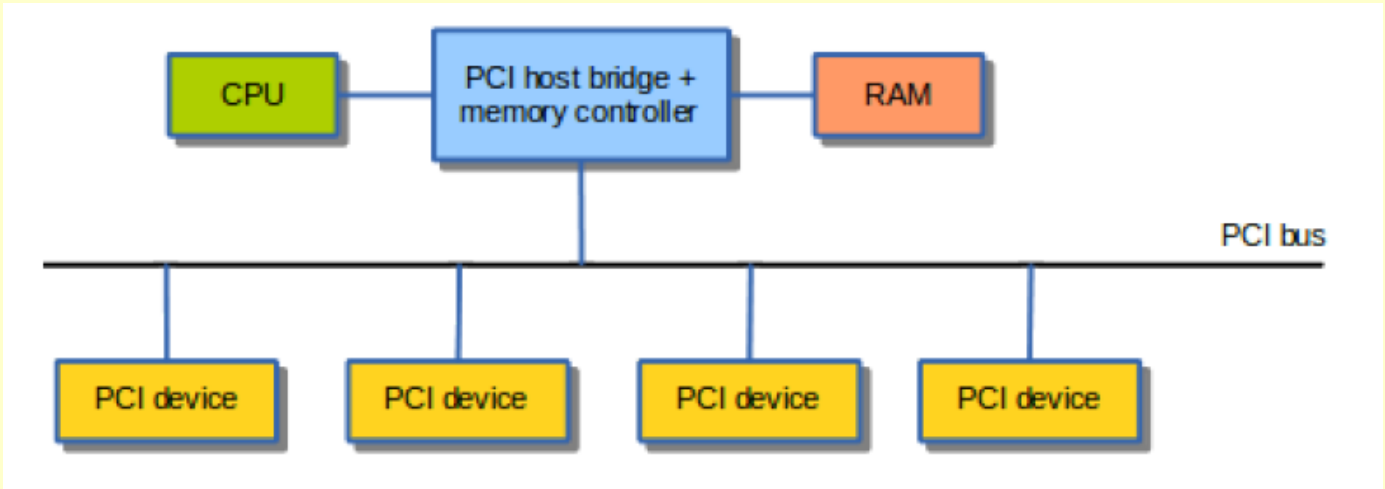


Floppy Disk/ DMA Hardware System

Exemplu de transfer de date de la unitatea de dischetă in memoria sistem

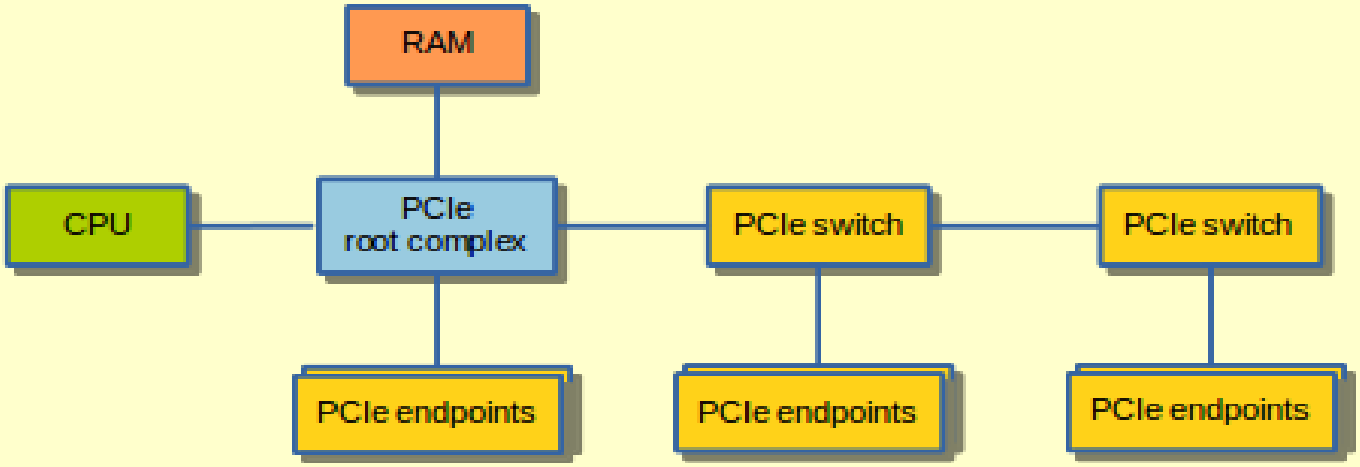
Transfer DMA pe bus-ul PCI

•Spre deosebire de ISA arhitectura ***PCI nu are nici un controler central DMA***.
Un periferic PCI poate solicita controlul bus-ului (devenind master pe bus) și să ***emită comenzi de citire/ scriere din/în memoria sistemului***
•Cererea controlului bus-ului, de catre un periferic PCI, se face de la controller-ul de magistrala PCI ***(de ex. South-bridge într-un PC modern)***, care se va comporta ca ***arbitru,*** în cazul în care mai multe dispozitive solicita dreptul de control al bus-ului, în același timp.
•Odată ce ***un periferic devine master pe bus***, va emite comenzi de citire/scriere pe magistrala PCI, care vor fi ***primite de către controlerul de bus PCI și transmise către controlerul de memorie*** (schema mai detaliată este specifică fiecărui chipset).
•CPU va fi întreruptă numai în cazul în care întreaga secvență a transferurilor DMA este completată de motorul DMA

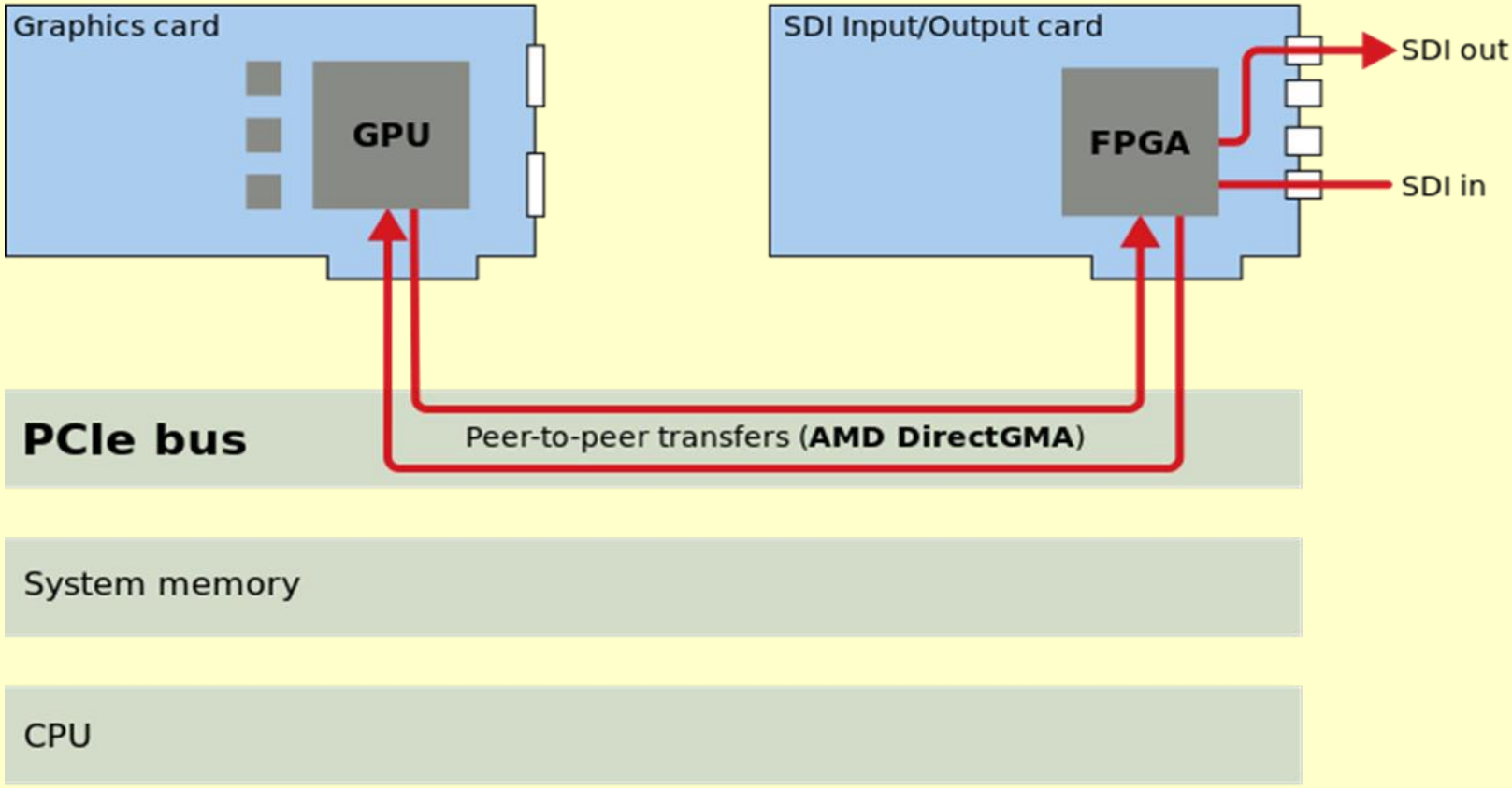


Transfer DMA pe bus-ul PCIe

•PCIe a transformat magistrala PCI convențională de la o arhitectură reală de tip magistrala, cu ***mai multe dispozitive fizice partajând aceeași magistrală***, într-o arhitectură de tip conexiune ***punct-la-punct*** cu ***comutare de pachete***; foarte asemănător cu modul în care funcționează rețelele cu comutare de pachete.



- PCIe conectează fiecare dispozitiv cu ***o legătură dedicată, bidirecțională (lane)***, la un comutator PCIe. Ca rezultat, PCIe suportă transferuri DMA full-duplex ale mai multor dispozitive în același timp.
- ***Logica arbitrajului*** este înlocuită de ***logica de rutare a pachetelor*** implementată în comutatoarele PCIe.
- În timp ce PCIe este complet diferit de PCI la nivel hardware, PCIe păstrează compatibilitatea cu PCI la nivel de driver.
- Dispozitivele PCIe mai noi, pot fi detectate și utilizate de către driverele PCI fără suport explicit pentru standardul PCIe. Cu toate acestea, noile caracteristici PCIe nu pot fi utilizate.

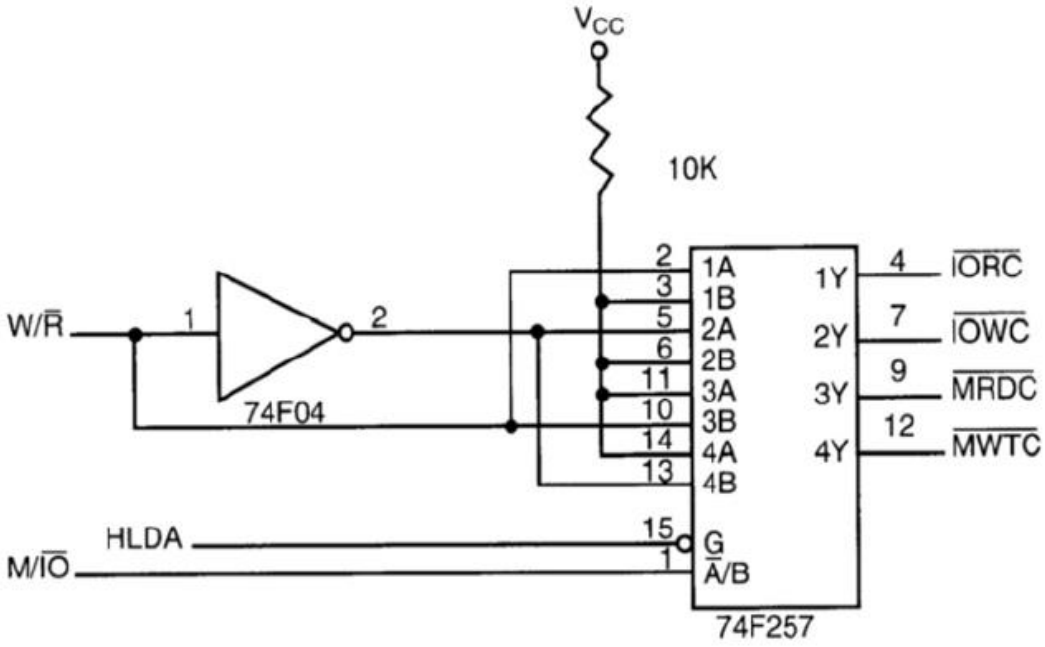


- AMD DirectGMA este o forma de DMA
- Aceasta permite un transfer de date peer-to-peer low-latency intre dispozitive pe bus-ul PCIe si produse AMD FirePro-branded
- Dispozitivele *Serial digital interface (SDI)* care suporta DirectGMA pot scrie direct in memoria grafica a GPU si vice versa GPU poate accesa direct memoria unui device pereche

Tema 1. Studiati schema alaturata.

A Simple DMA Controller Circuit

Uses a 74F257: 2-line input, tri-state multiplexer



```
; This routine programs the DMAC for channels 0-3
;
; IN: [DMA_Channel], [DMAbaseAdd], [DMApageReg]
   ;must be setup
;      [DAMBaseAdd] =  Memory Address port
;
;    dh = mode
;    ax = address
;    cx = length
;    dl = page
_____________________________________________
PROC Prog_DMA03 NEAR

    push   bx
    mov    bx,ax

    mov    al,4
    add    al,[DMA_Channel]
    out    0Ah,al        ; mask reg bit

    sub    al,al
    out    0Ch,al        ; clr byte ptr

    mov    al,dh
    add    al,[DMA_Channel]
    out    0Bh,al        ; set mode reg

    push   dx
    mov    dx,[DMAbaseAdd]
    mov    al,bl
    out    dx,al         ; set base address low
    mov    al,bh
    out    dx,al         ; set base address high

    inc    dx            ;point to length
    mov    al,cl
    out    dx,al          ; set length low
    mov    al,ch
    out    dx,al          ; set length high

    pop    dx

    mov    al,dl
    mov    dx,[DmaPageReg]
    out    dx,al          ; set DMA page reg

    mov    al,[DMA_Channel]
    out    0Ah,al ; unmask (activate) dma channel
    pop    bx
    ret
ENDP
```

Tema 2. Studiati rutinele alaturate.

```
; Code to convert segment:offset to page:page_offset
;
; SI contains offset,
; DX contains segment
   mov ax,dx
   mov cl,4
   shl ax,cl              ; shift segment left 4
   shr dx,cl              ; shift segment right 12
   add si,ax              ; add shifted segment to offset
   adc dh,0
; Now SI contains the offset within the page,
; the low 4-bits of DH contain the page #

; Disable the DMA channel so we can set it
   mov ax,channel
   and ax,3               ; channel mod 4
   or ax,MODE             ; set mode bits
   out MASKREG,al         ; write DMA mode

; Clear byte ptr F/F
   out BYTEREG,al         ; any value to reset

; write mode to mode register
   mov al, mode
   out MODEREG,al
```

```
; write page offset to address reg.
   mov ax,si              ; get offset in ax
   out ADDRREG,al         ; write LSB of DMA offset
   mov al,ah
   out ADDRREG,al         ; write MSB of DMA offset
                          ; write length (-1) to count reg.
   mov ax,length          ; ax=length-1
   dec ax                 ;
   out COUNTREG,al        ; write LSB of size
   mov al,ah
   out COUNTREG,al        ; write MSB of size

                          ; write page# to page register
   mov al,dh
   out PAGEREG,al         ; write page

; re-enable the channel
   mov ax,channel
   and al,3                          ; channel mod 4
   out MASKREG,al         ; enable sound card DMA
                          ; now set up the target device...
```


Tema3. Studiati secventa de program.