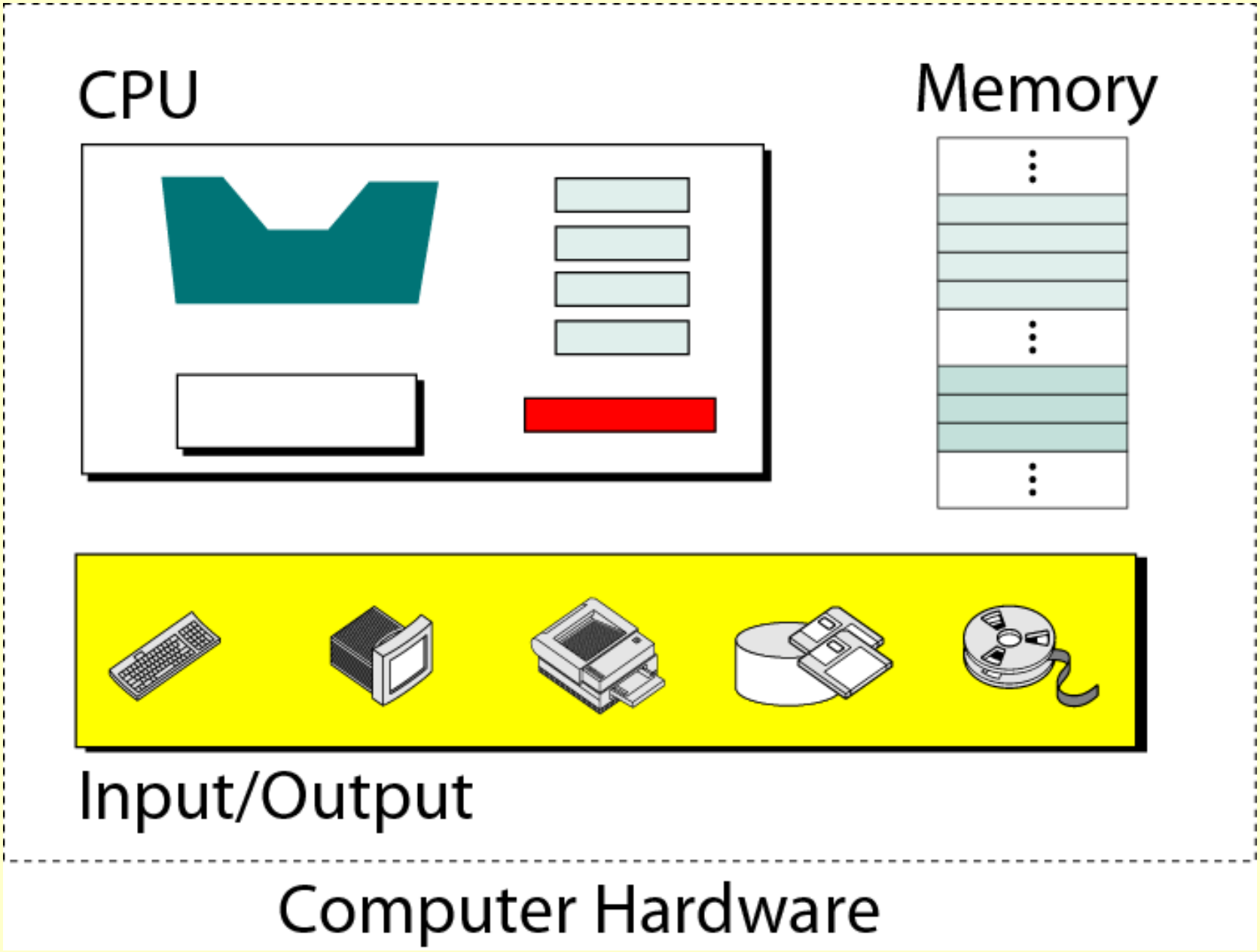
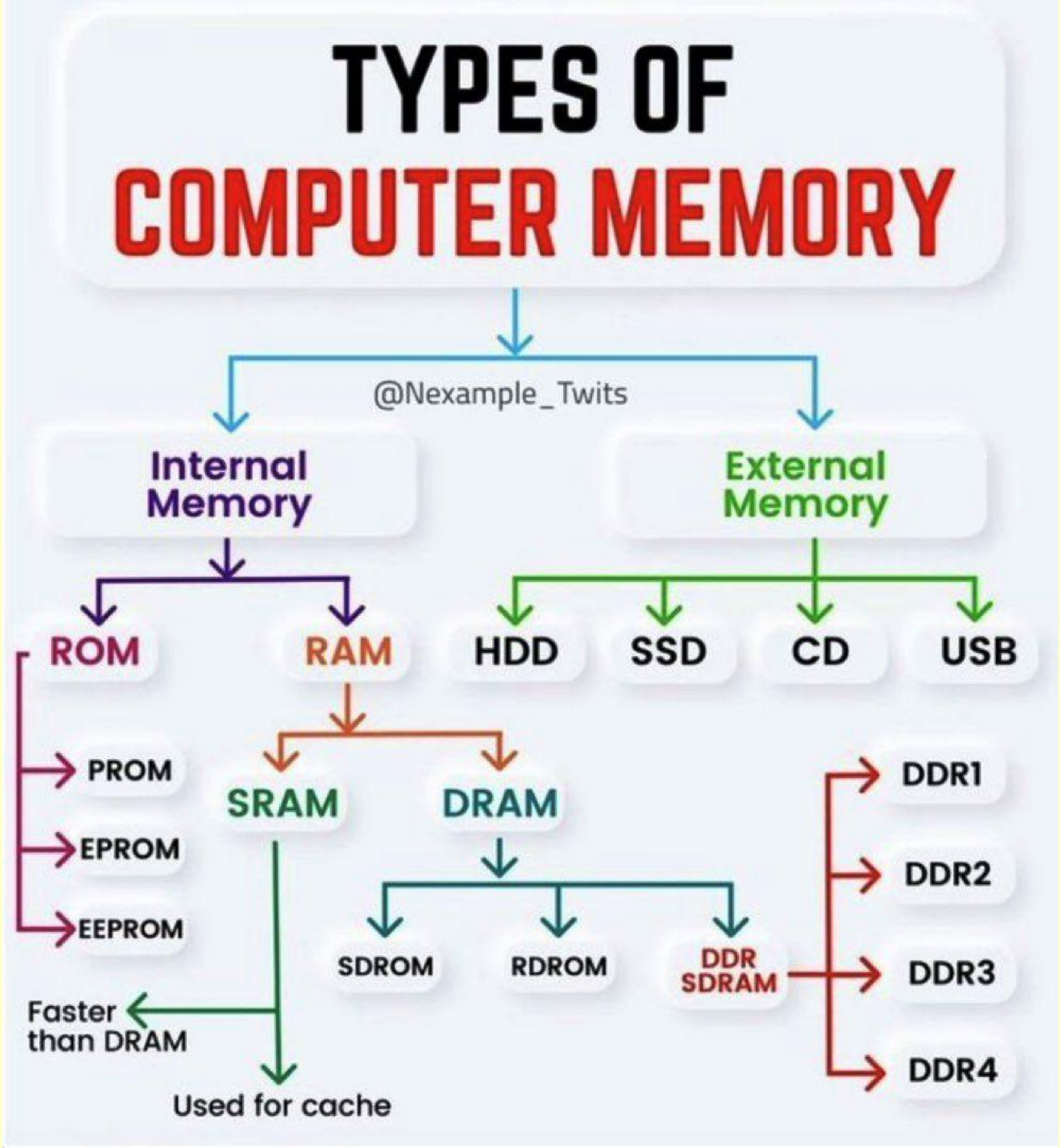
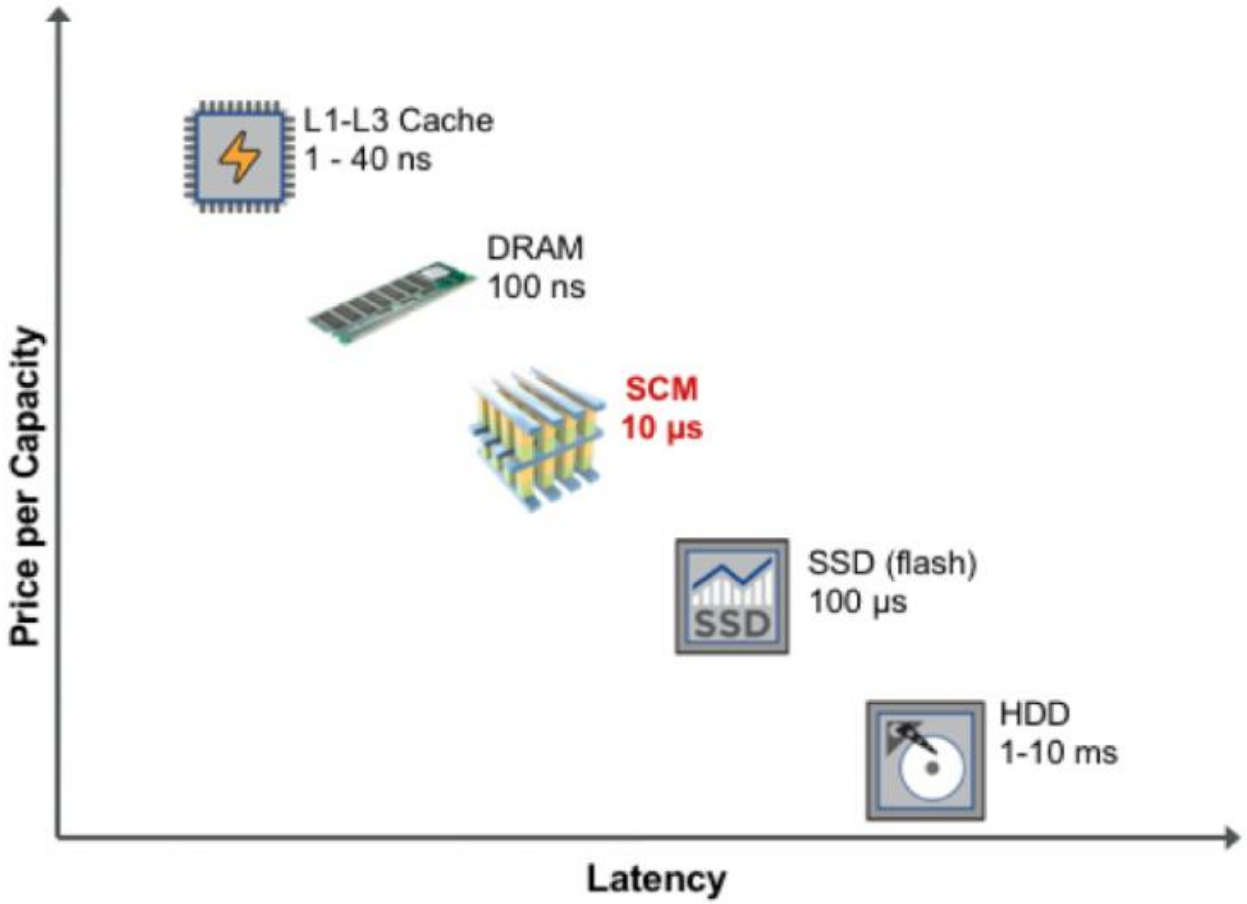
C2- MEMORIA IN PC

1. Clasificarea memoriilor
2. Limitarile memoriei. Ierarhizarea
3. Managementul memoriei
4. Organizarea memoriei PC-ului in mod real
5. Aplicatii
6. Teme + tendinte in dezvoltarea memoriilor

http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-823-computer-system-architecture-fall-2005/lecture-notes/
https://www.eeeguide.com/linear-logical-address-and-physical-address-in-microprocessor/


CPU
Memory
Input/Output
Computer Hardware


TYPES OF
COMPUTER MEMORY
@Nexample_Twits
Internal Memory
External Memory
ROM
RAM
HDD
SSD
CD
USB
PROM
EPROM
EEPROM
SRAM
DRAM
SDROM
RDROM
DDR SDRAM
DDR1
DDR2
DDR3
DDR4
Faster than DRAM
Used for cache



https://www.computerweekly.com/feature/Flash-vs-3D-Xpoint-vs-storage-class-memory-Which-ones-go-where

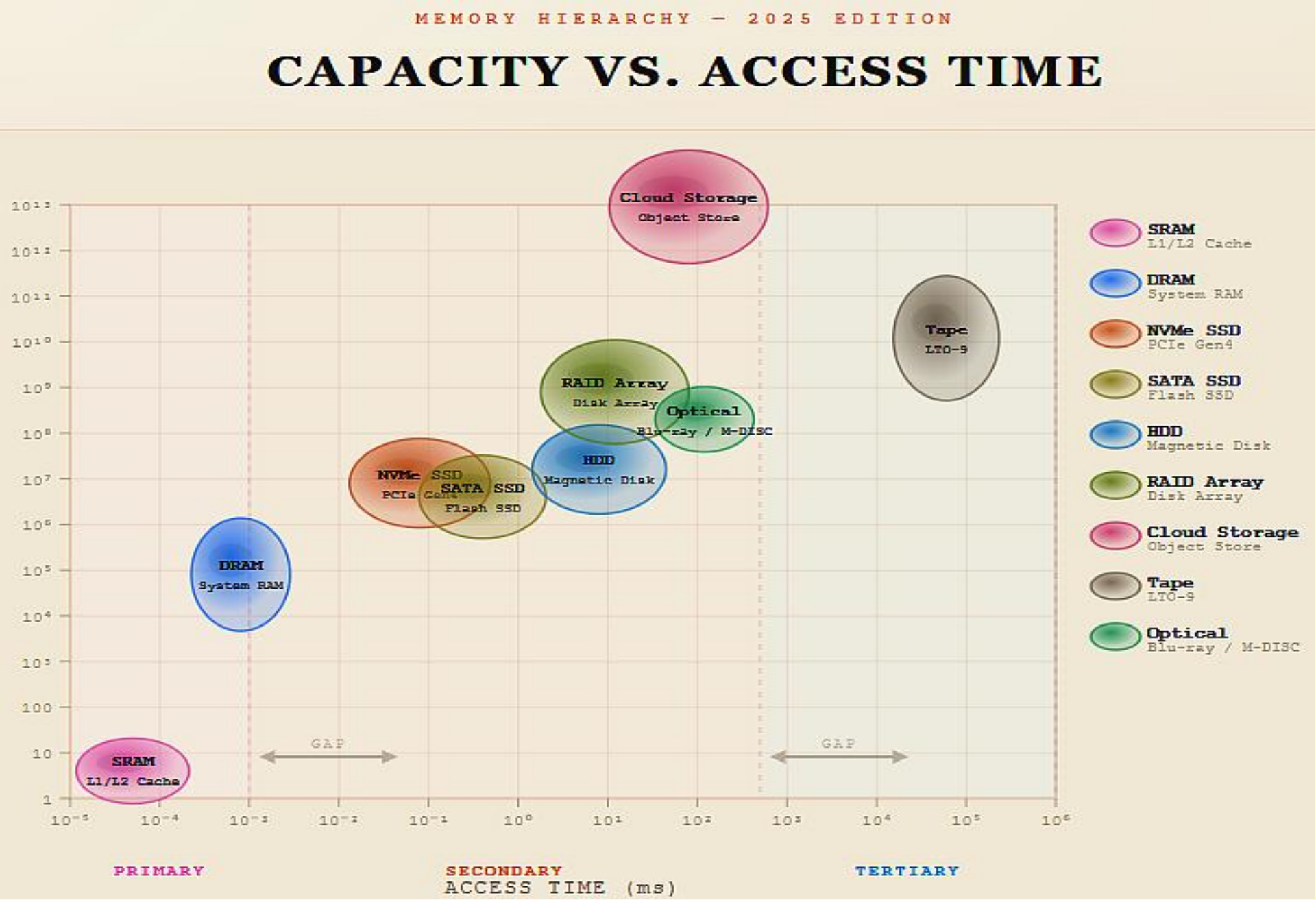
- **SCM -** *Memorie din clasa de stocare* este o formă de stocare creată din flash-NAND. Este un pas intermediar între DRAM de înaltă performanță și HDD-uri ieftine. Poate oferi performanțe de scriere care sunt semnificativ mai rapide decât tehnologia HDD și performanțe de citire similare cu DRAM.

•**SCM -** *Memorie din clasa de stocare* este o formă de stocare creată din flash-NAND. Este un pas intermediar între DRAM de înaltă performanță și HDD-uri ieftine. Poate oferi performanțe de scriere care sunt semnificativ mai rapide decât tehnologia HDD și performanțe de citire similare cu DRAM.

Principalele avantaje ale memoriei din clasa de stocare includ:

- oferă latențe de acces mai mici decât cele ale SSD-urilor flash
- crește randamentul mai mult decât la stocarea flash-urilor
- este mai puțin costisitor decât DRAM
- oferă acces în timp real la date - permite accesul DRAM-like la seturi de date mari prin capacitatea de memorare a sistemului
- păstrează datele stocate chiar și după oprirea alimentării

Memoria din clasa de stocare este perfectă pentru mediile sensibile la perioadele de nefuncționare cauzate de întreruperile de curent sau de blocare a sistemului, precum și pentru mediile care necesită acces frecvent la seturi de date mari și complexe.



https://www.eejournal.com/article/want-to-know-the-future-of-new-memories-mram-fram-pcm-tom-coughlin-and-jim-handy-make-predictions/

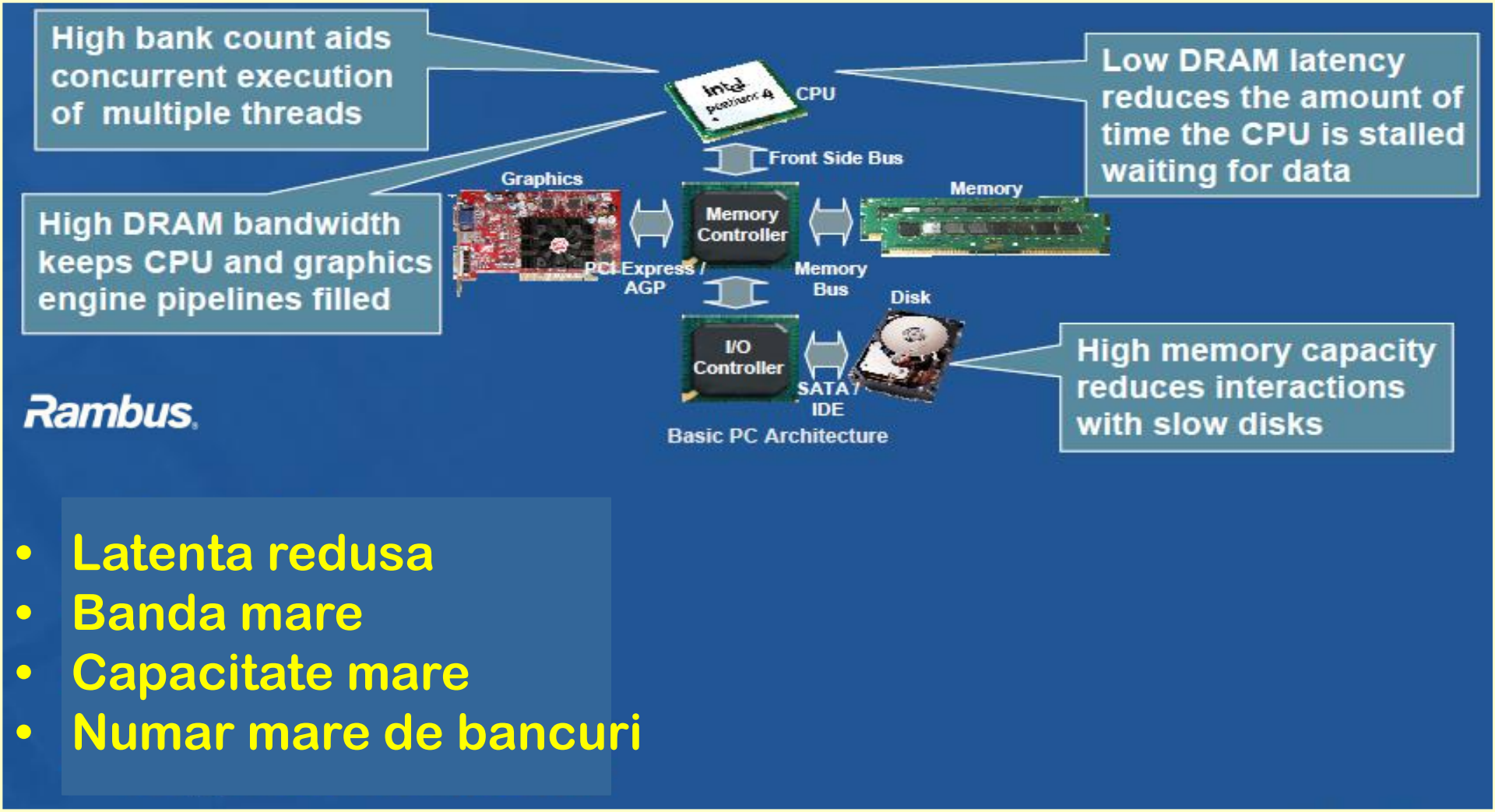
http://www.wtec.org/loyola/opto/c3_s2.htm

Tehnologie	Viteza	Putere consumata	Capacitate	Cost
TTL (bipolar)	+	+	-	+
ECL	++	++	--	++
NMOS	0	-	++	--
CMOS	-	--	+	-

Clasificarea diferitelor tipuri de memorii semiconductoare

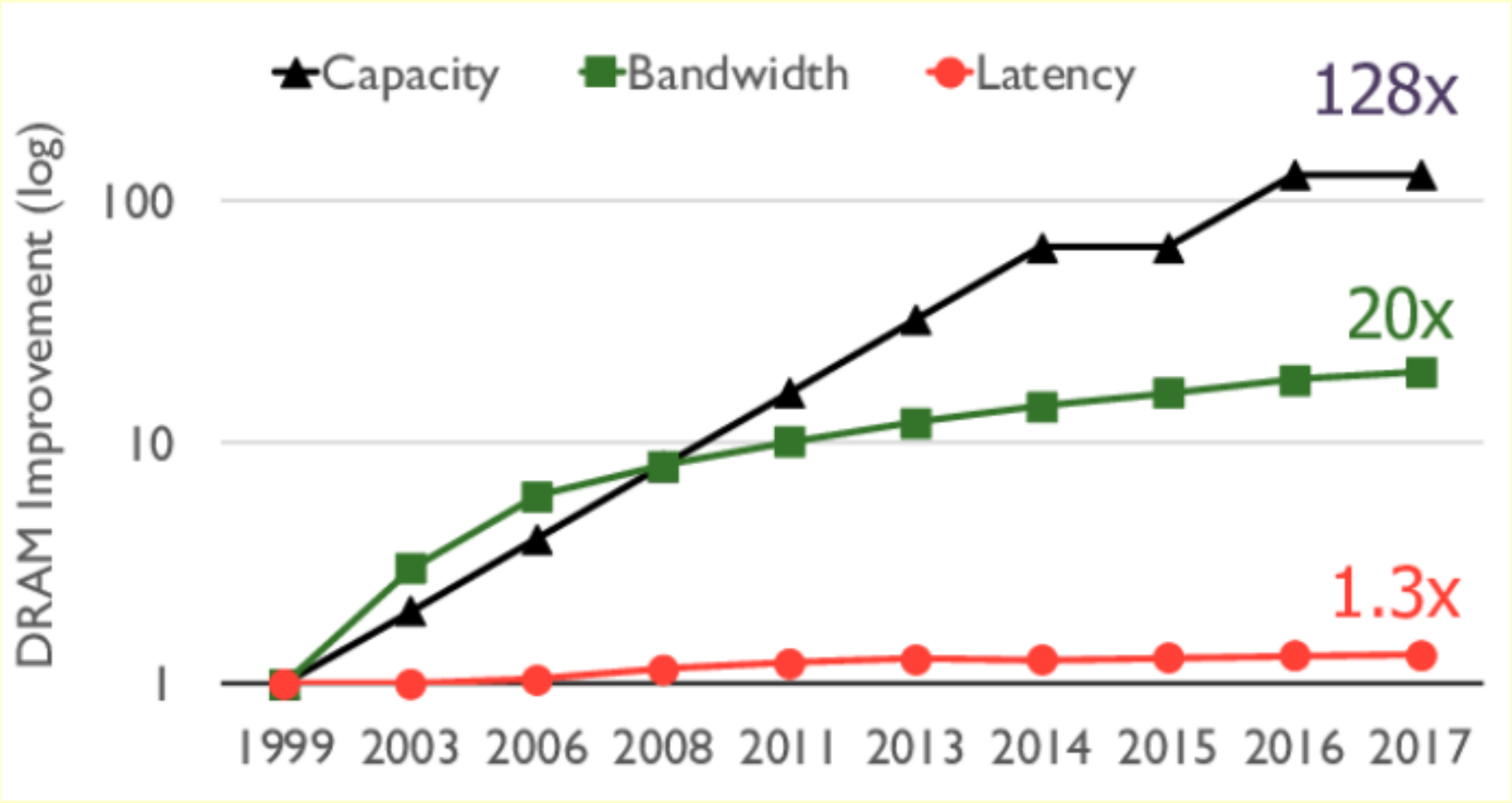
2. Limitarile memoriei. Ierarhizarea

- Ce caracteristici fac ca un sistem de memorie sa fie bun?



- Latenta redusa
- Banda mare
- Capacitate mare
- Numar mare de bancuri

- **Latenta = timpul pentru un acces singular (Memory access time >> Processor cycle time)**
- **Banda = numarul de accese/unitatea de timp** Daca sunt m instructiuni load/store, 1 + m acces/instr. (CPI = 1) sunt cel putin 1 + m accese memorie/ciclu

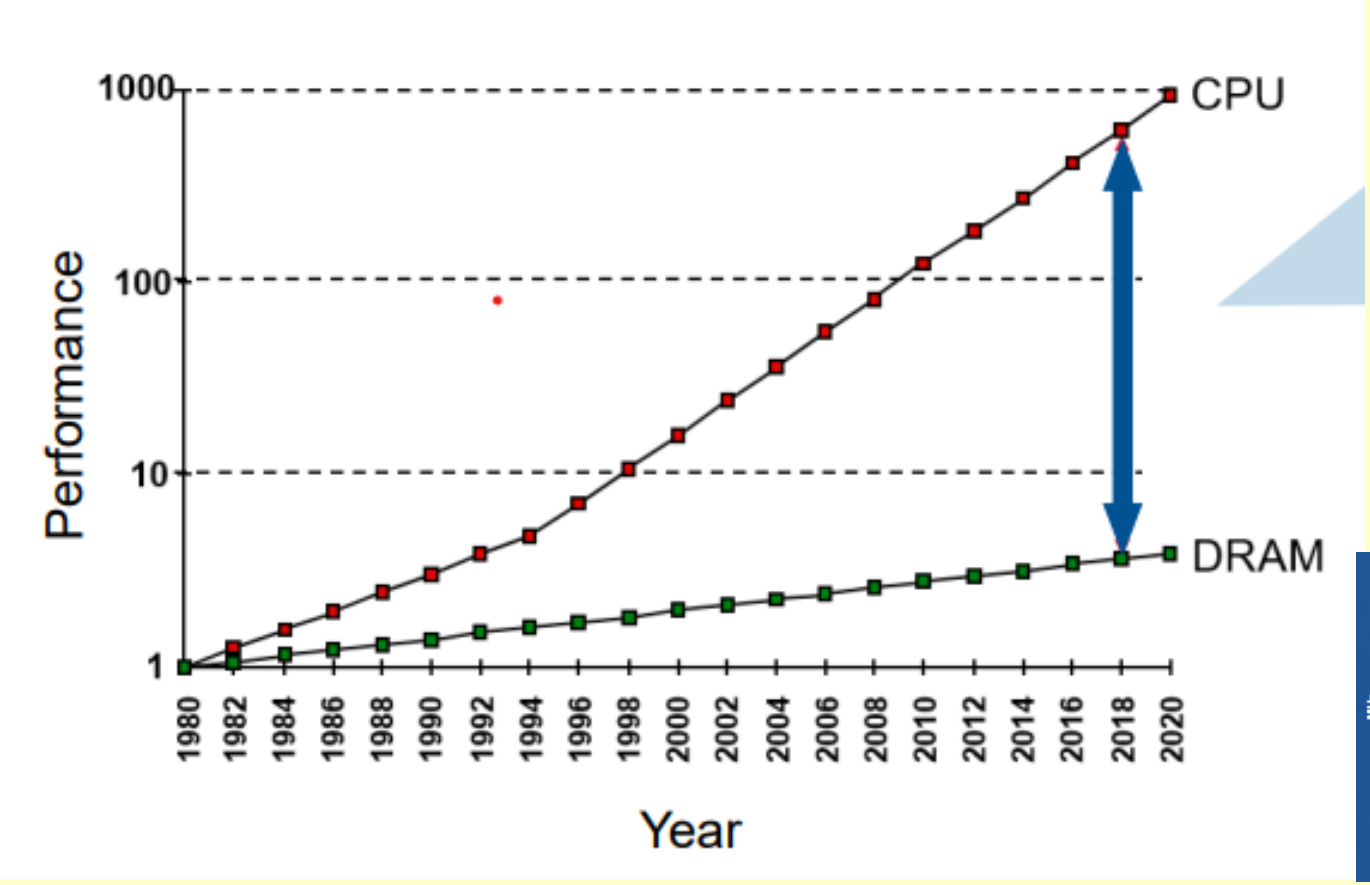


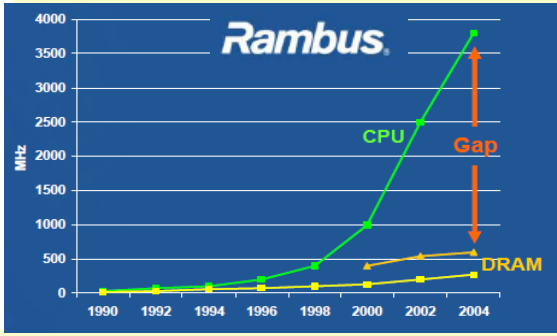
Imbunatatiri ale performantelor DRAM

https://www.slideserve.com/norina/processor

Legea lui Amdahl

- Performanta unui sistem este limitata de componenta cea mai lenta
- Decalajul de performanta dintre procesor-memorie





https://www.quora.com/Why-is-the-gap-between-the-CPU-and-the-main-memory-speed-widening

Limitarile introduse de memorie

PROBLEMA

– Frecventele procesoarelor continua sa creasca

– Este nevoie de memorie care sa tina pasul cu procesoarele

- altfel, sunt prea multe intarzieri in pipeline

– Este nevoie de memorii de mare capacitate ptr. a satisface nevoile crescande ale aplicatiilor SW, paralelism

– *Memoria DRAM*

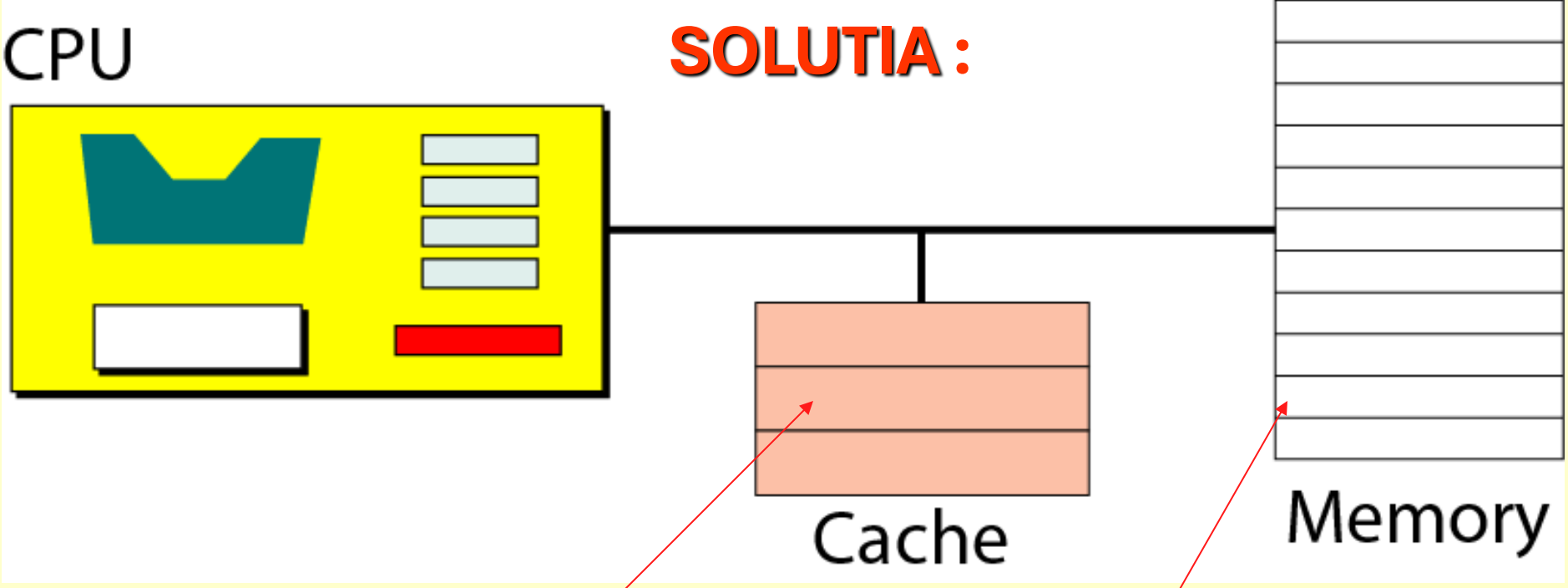
- Capacitate mare la cost redus (GBs) – bine
- Acces lent (10-100x de cicli procesor) – slab

- *Memoria SRAM*

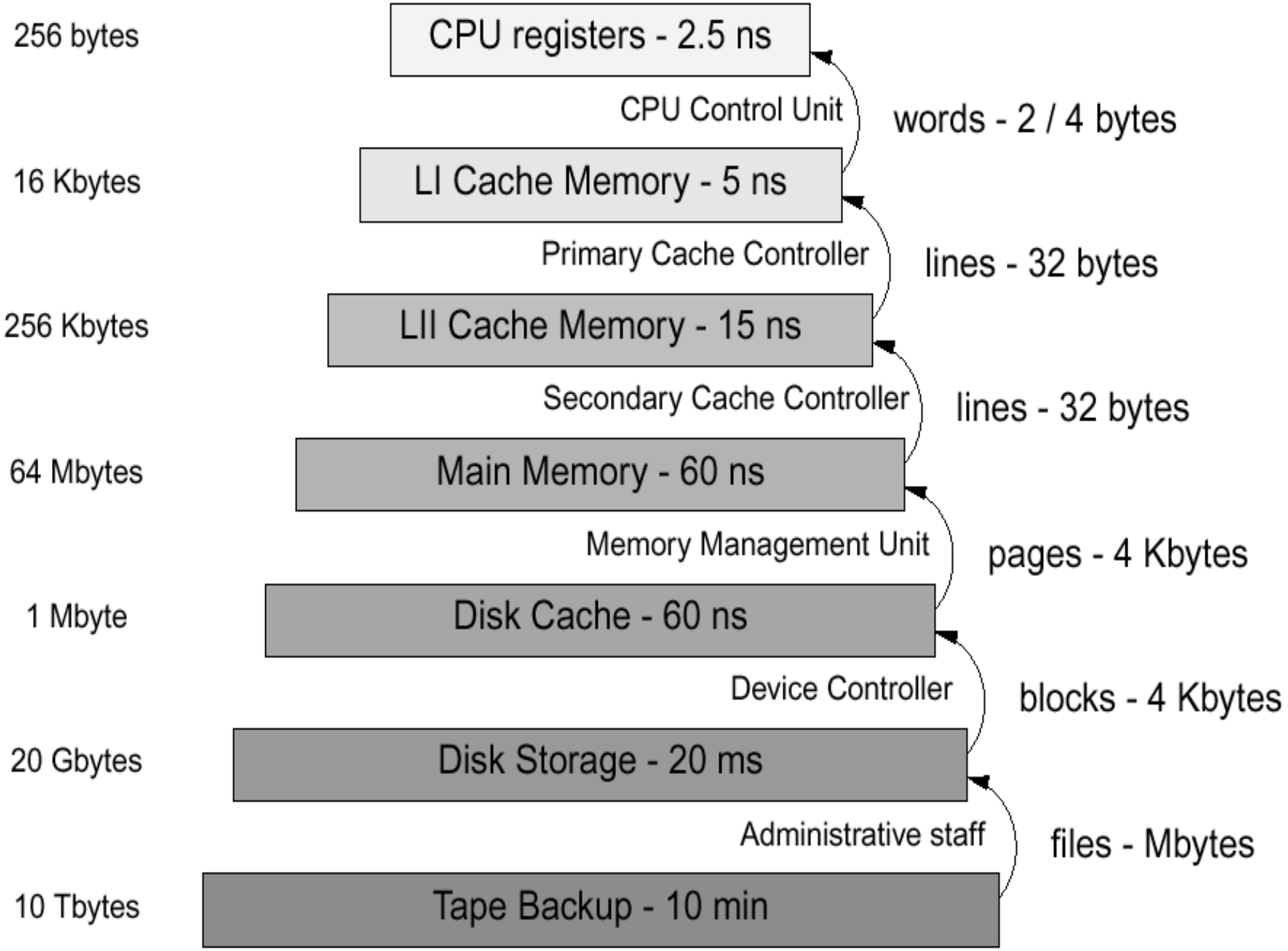
- Capacitate mare este ft. scumpa (MBs) – slab
- Acces rapid (1-10 ciclii procesor) – bine

– Putem obtine capacitatea si costul DRAM la viteza de acces a SRAM?

SOLUTIA :

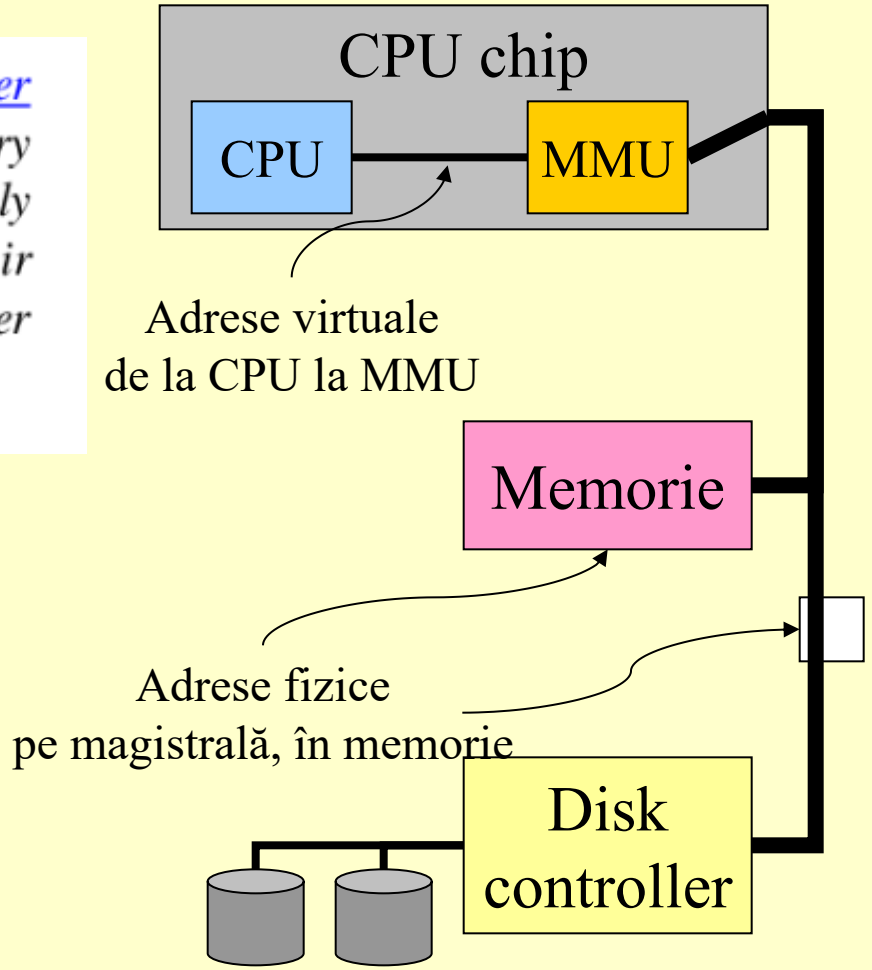


SRAM	DRAM
1. SRAM has lower access time, so it is faster compared to DRAM.	1. DRAM has higher access time, so it is slower than SRAM.
2. SRAM is costlier than DRAM.	2. DRAM costs less compared to SRAM.
3. SRAM requires constant power supply, which means this type of memory consumes more power.	3. DRAM offers reduced power consumption, due to the fact that the information is stored in the capacitor.
4. Due to complex internal circuitry, less storage capacity is available compared to the same physical size of DRAM memory chip.	4. Due to the small internal circuitry in the one-bit memory cell of DRAM, the large storage capacity is available.
5. SRAM has low packaging density.	5. DRAM has high packaging density.


256 bytes
CPU registers - 2.5 ns
CPU Control Unit
words - 2 / 4 bytes
16 Kbytes
LI Cache Memory - 5 ns
Primary Cache Controller
lines - 32 bytes
256 Kbytes
LII Cache Memory - 15 ns
Secondary Cache Controller
lines - 32 bytes
64 Mbytes
Main Memory - 60 ns
Memory Management Unit
pages - 4 Kbytes
1 Mbyte
Disk Cache - 60 ns
Device Controller
blocks - 4 Kbytes
20 Gbytes
Disk Storage - 20 ms
Administrative staff
files - Mbytes
10 Tbytes
Tape Backup - 10 min

3. Gestionarea Memoriei

Memory management is the act of managing computer memory. The essential requirement of memory management is to provide ways to dynamically allocate portions of memory to programs at their request, and freeing it for reuse when no longer needed.



http://www.c-jump.com/CIS77/ASM/Memory/lecture.html#M77_0010_addressing
Adresă Liniară, Logică și Fizică în Microprocesoare (eeeguide.com)

- *Adresa logică* este o adresă virtuală accesibilă utilizatorului.
- *Adresa fizică* nu poate fi accesată direct de utilizator. Adresa logică este folosită ca referință pentru accesul fizic.
- Distincție esențială: *CPU generează adrese logice* la execuție; *MMU le translatează* în locații fizice de RAM.
- Spațiul de adrese logice permite protecția memoriei și multiprogramarea eficientă prin izolarea fiecărui proces.
- Adresele virtuale și fizice sunt adrese liniare, dar nu și invers.

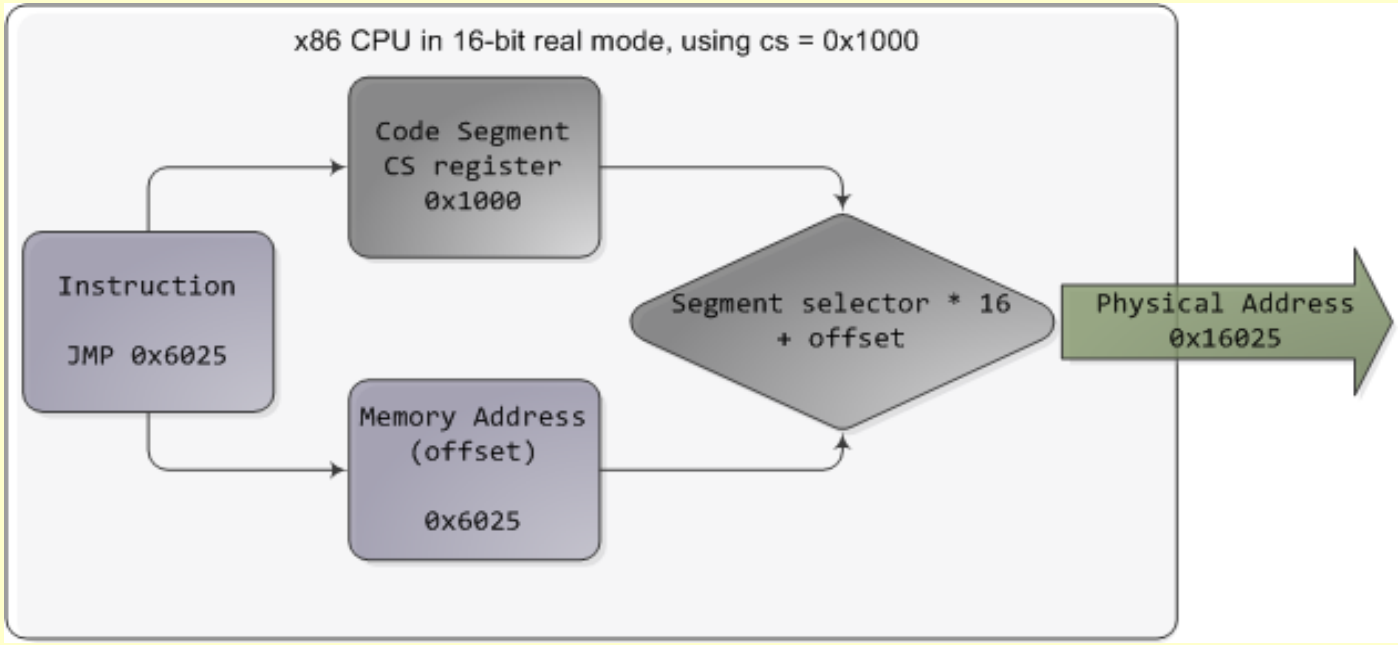
Adrese Logice = Adrese Virtuale -> Adrese Fizice

MODURI DE OPERARE

- **Modul adresă reală**
  - nativ MS-DOS
- **Modul protejat**
  - mod nativ (Windows, Linux)
    - Modul Virtual-8086
      - hibrid al modului protejat
      - fiecare program are propriul calculator 8086
- **Modul de gestionare**
  - gestionarea energiei, securitate sistem, diagnostice

3.1 MODUL ADRESĂ REALĂ

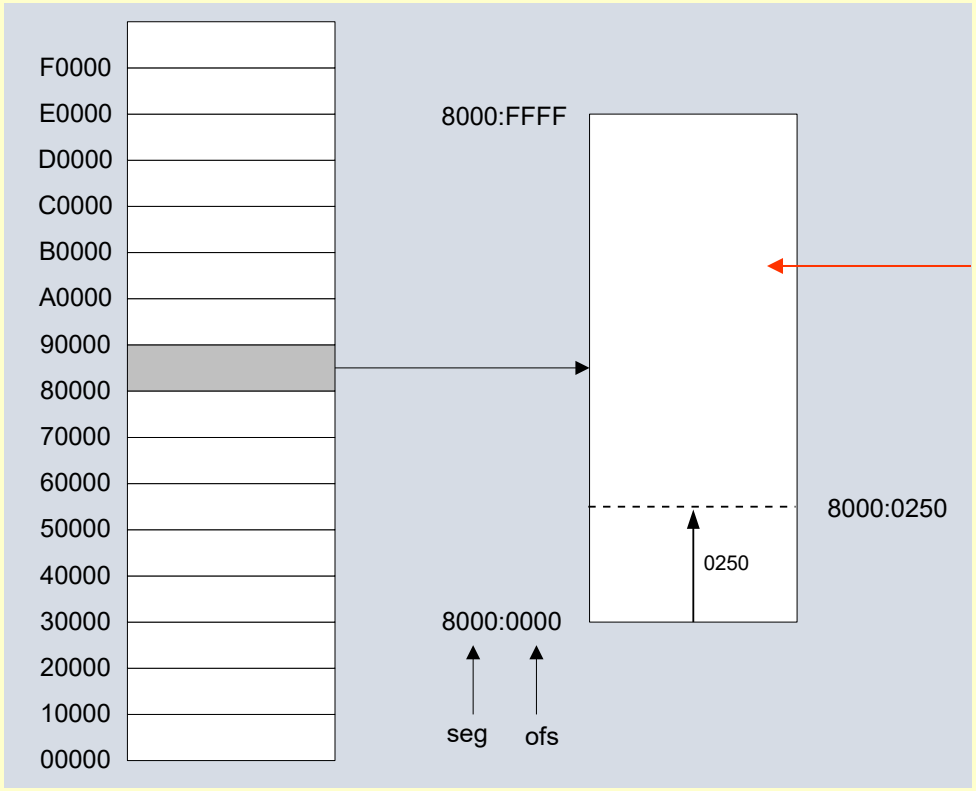
- Maxim 1 MB RAM adresabil (adresă pe 20 de biți)
- Programele de aplicație **pot accesa orice zonă din memorie**
- Monotasking
- Suportat de sistemul de operare MS-DOS
- Orice program **avea acces la orice funcție hardware**



Segmentare în modul real

Memorie segmentată

- Segmented memory addresing : physical (linear) address is a combination of a 16-bit segment value added to a 16-bit offset



Adresă fizică:

80000h + 0250h = 80250h

3.2 MODUL PROTEJAT

- 4 GB DRAM adresabil (adresă pe 32 biți) -(00000000 la 0FFFFFFFFh)
- Fiecărui program i se atribuie o **partiție de memorie** protejată față de alte programe
- Conceput pentru multitasking
- Suportat de Linux & MS-Windows

Adrese logice și liniare

- În modul protejat, unitatea de segmentare convertește o adresă logică într-o adresă liniară. Fiecare octet din memorie este accesat printr-o adresă logică.
- Adresă logică = selector segment 16 biți + offset 32 biți. Selectorul identifică descriptorul de segment care furnizează *adresa de bază a segmentului*. Offset = poziția octetului în acel segment.
- *Descriptorul de segment are un câmp limită.*
- Dacă o adresă liniară depășește limita segmentului, CPU generează o excepție — protejând segmentele de accese invalide.

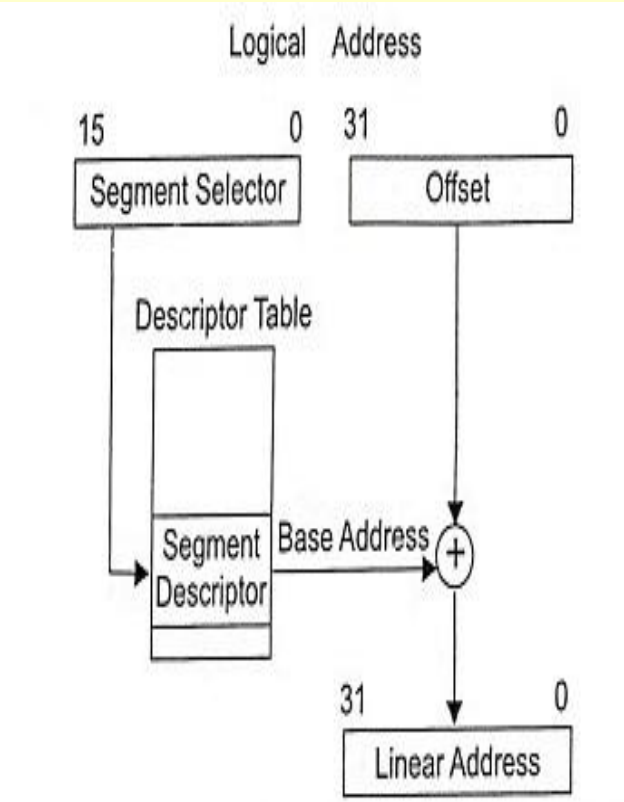


Fig. 12.17 *Logical address and linear address*

- La utilizarea *tehnicii de paginare pentru gestionarea memoriei*, unitatea de paginare *preia ieșirea unității de segmentare (adresa liniară)* și o convertește *adresa liniară* în *adresă fizică*.
- Dacă nu se folosește paginarea, *adresă liniară* este direct mapată la adresa *fizică*.

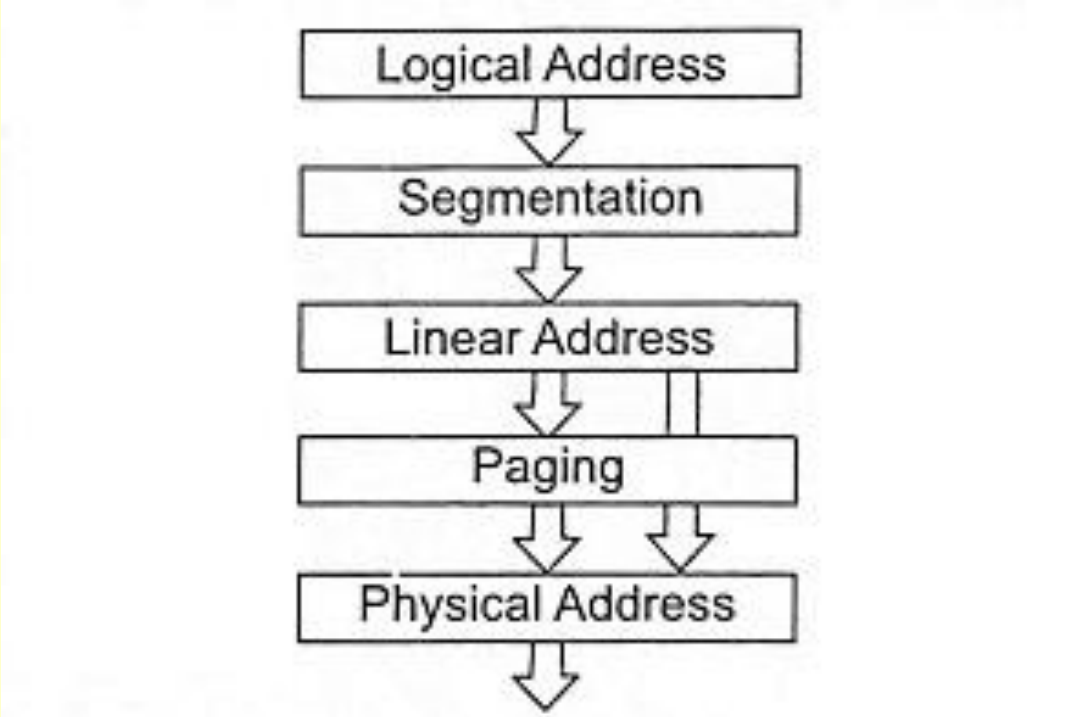
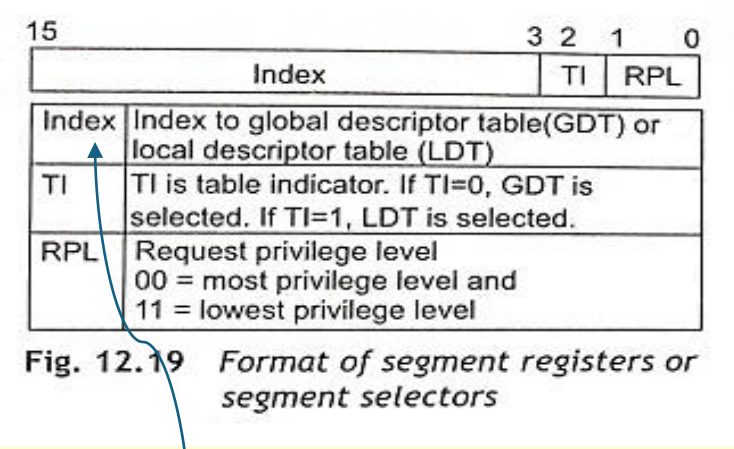


Fig. 12.18 *Physical-address generation*

Segmentele sunt interpretate diferit în modul protejat față de modul real:

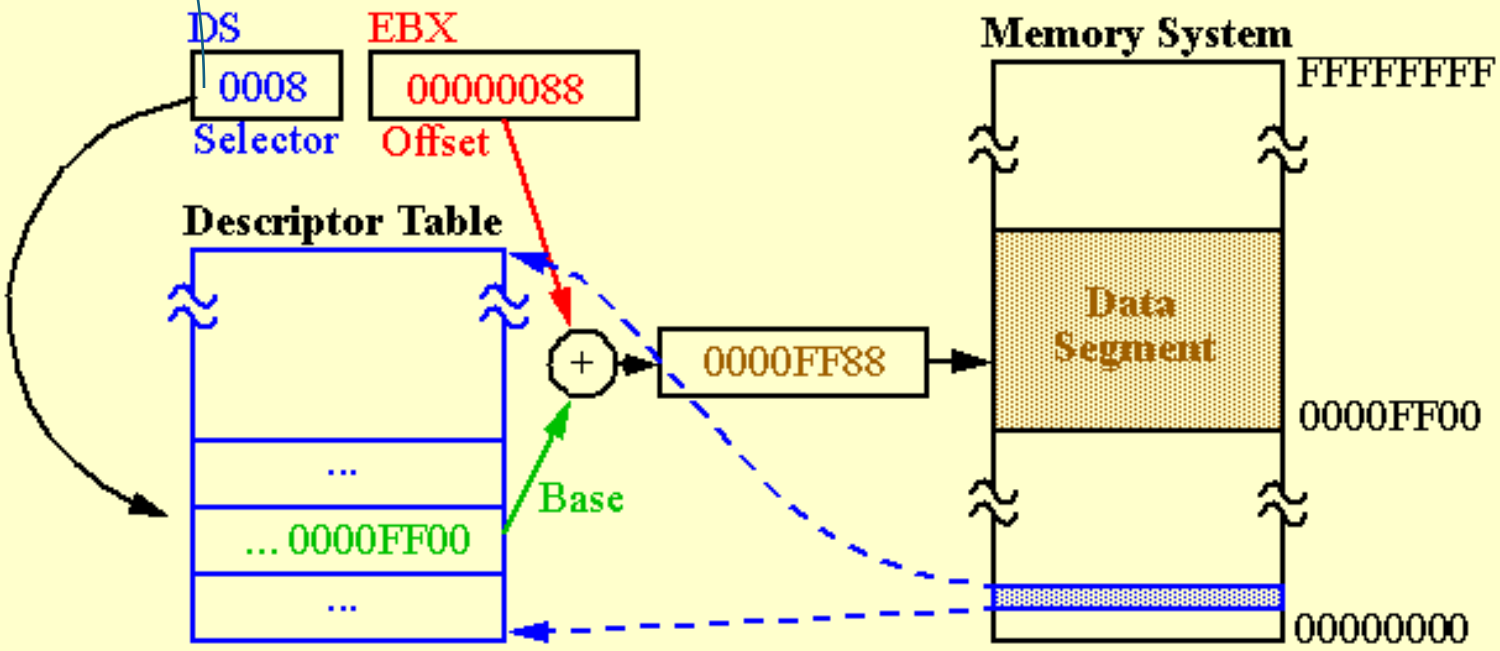
•Segment register contains a selector that selects a descriptor from the descriptor table.

•The descriptor contains information about the segment, e.g., it's base address, length and access rights. The offset can be 32-bits.



Index	Index to global descriptor table(GDT) or local descriptor table (LDT)
TI	TI is table indicator. If TI=0, GDT is selected. If TI=1, LDT is selected.
RPL	Request privilege level 00 = most privilege level and 11 = lowest privilege level

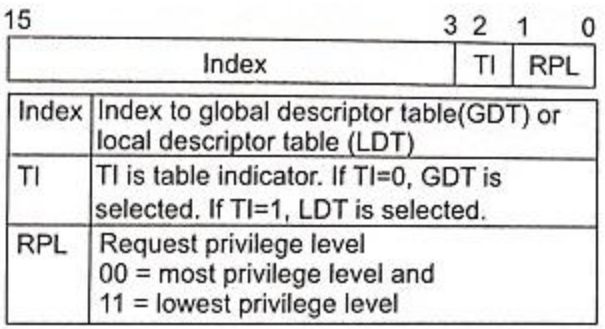
Fig. 12.19 *Format of segment registers or segment selectors*



Registre de segment sau selectori de segment

- În modul protejat, selectorii de segment constau din trei câmpuri: index, Indicator de Tabelă (TI) și RPL (Nivel de Privilegiu Solicitat). Aceștia indică descriptori, nu adresele de bază direct.

- Câmpul index (biții 15–3): selectează un descriptor de segment din GDT (Tabela Globală de Descriptori) sau LDT (Tabela Locală de Descriptori).



Index	Index to global descriptor table(GDT) or local descriptor table (LDT)
TI	TI is table indicator. If TI=0, GDT is selected. If TI=1, LDT is selected.
RPL	Request privilege level 00 = most privilege level and 11 = lowest privilege level

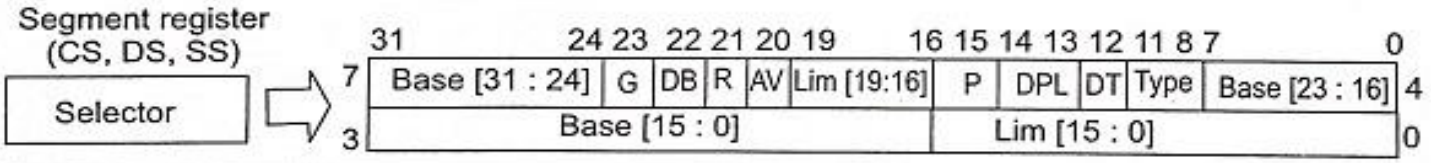
Fig. 12.19 *Format of segment registers or segment selectors*

- Bitul TI (bitul 2): selectează tabela de descriptori — ***TI=0 → GDT*** (partajat, la nivel de sistem); ***TI=1 → LDT*** (per-proces, privat).

- Registrele GDTR / LDTR dețin adresele de bază ale GDT şi LDT în spațiul de adrese liniare al procesorului.
- RPL (biții 1–0): definește nivelul de privilegiu al programului solicitant.
- 4 niveluri de privilegiu (inele): PL0 = cel mai ridicat (nucleul SO), PL3 = cel mai scăzut (aplicații utilizator). CPU aplică controlul accesului pe baza CPL (Nivelul de Privilegiu Curent).

Tabele globale și locale de descriptori și registre cache

- Tabela Globală de Descriptori (GDT) este o listă în memorie care descrie dimensiunile și adresele segmentelor din memorie sub formă de descriptori de segment.
- Fiecare descriptor de segment are 8 octeți.



Base [31:24], Base [23:16], Base [15:0]	Segment Base	G	Granularity
Limit [19:16], Limit [15:0]	Segment limit	DB	Segment size (Default/Big) DB = 0, 16 bit operand/address (Default) DB = 1, 32-bit operand/ address (Big)
P	Segment present	R	Reserved
DPL	Descriptor privilege level 00 = PL 0, 11 = PL3	DT	Descriptor type DT = 0, system segment DT = 0, Application segment
Type	Type of system or application segment		

Fig. 12.20 *Segment descriptors*

- G = 0, granularitate octet: cei 20 de biți reprezintă 1 Moctet.
  G = 1, granularitatea este 4K și reprezintă 20 biți x 4K = 4 Gocteți
- o instrucțiune pe 32 biți este *mov eax, mem32* când DB = 1,
  o instrucțiune pe 16 biți este *mov ax, mem32* dacă DB = 0.

- Pentru a evita accesele repetate la tabelele de descriptori din RAM, CPU stochează automat datele descriptorilor în *registre cache pentru descriptori de segment* (invizibile pentru programator). Fiecare registru de segment (CS, DS, SS, ES, FS, GS) are propriul cache de 64 biți cu adresa de bază, limita și drepturile de acces ale segmentului curent.

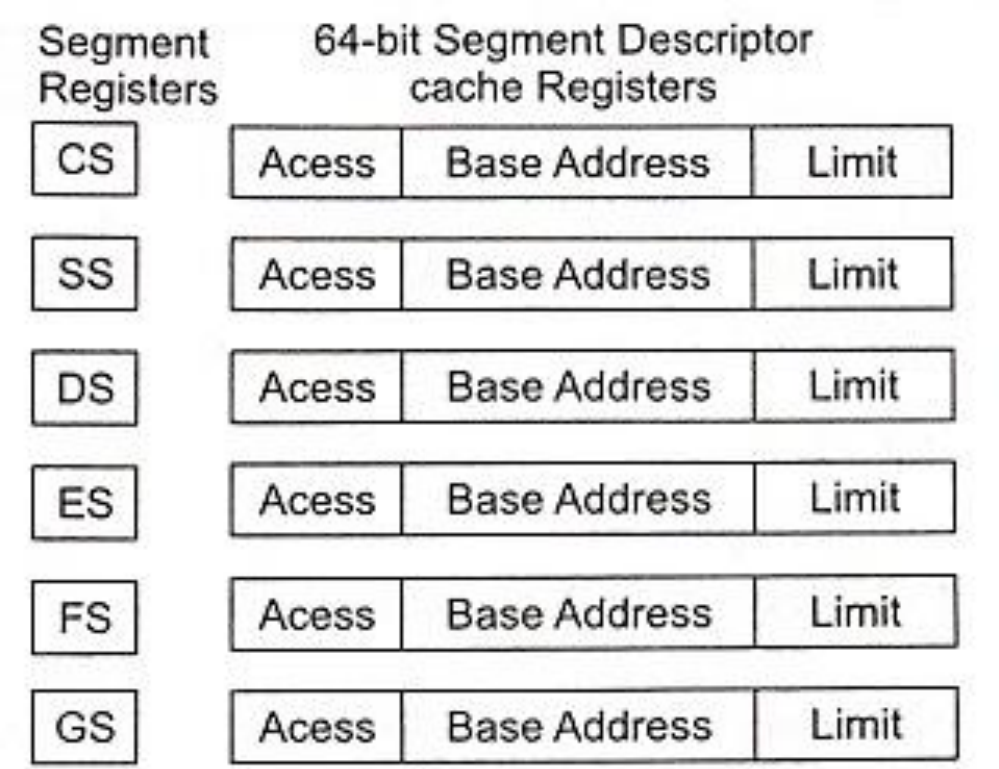
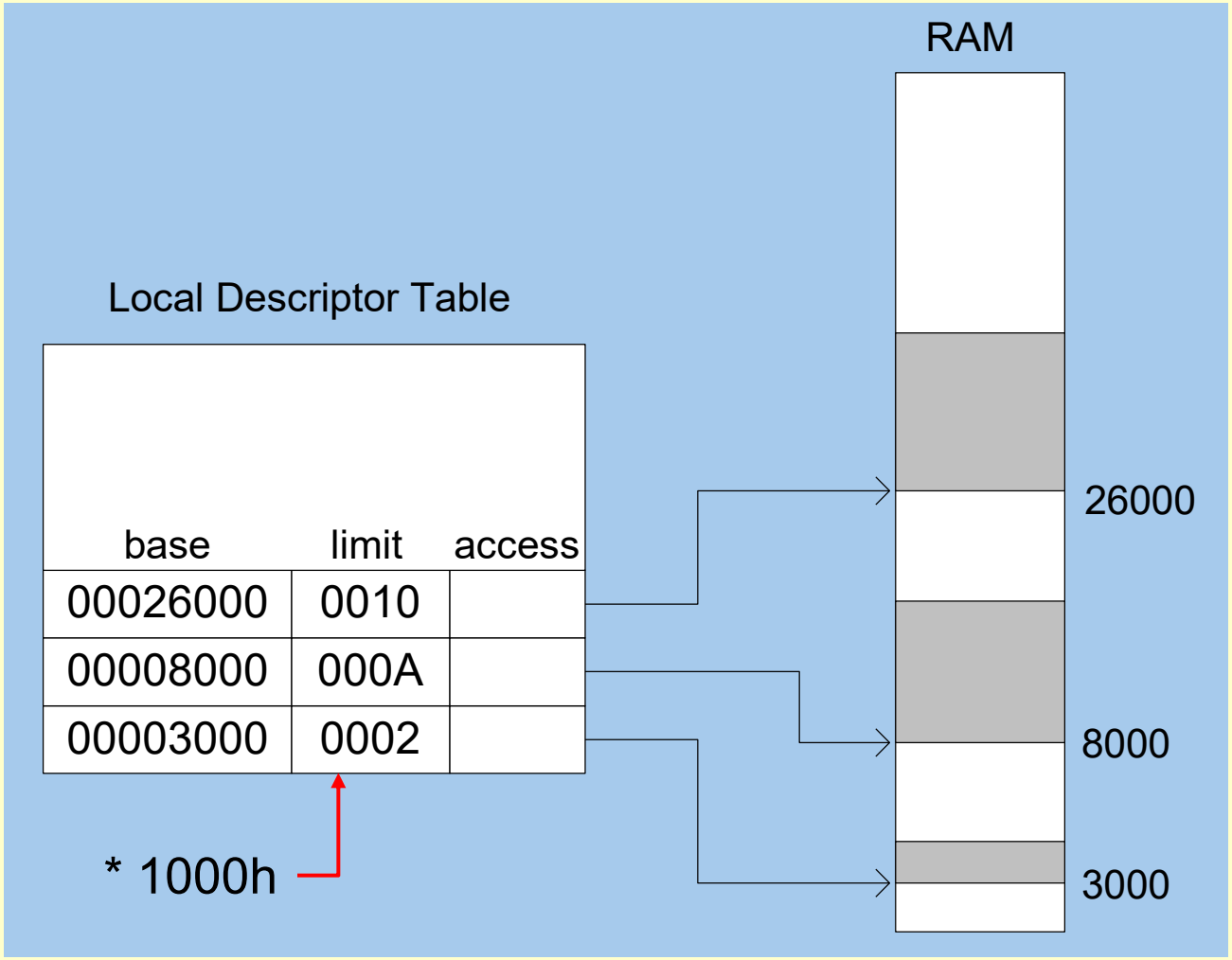


Fig. 12.21 *64-bit segment descriptor cache registers*

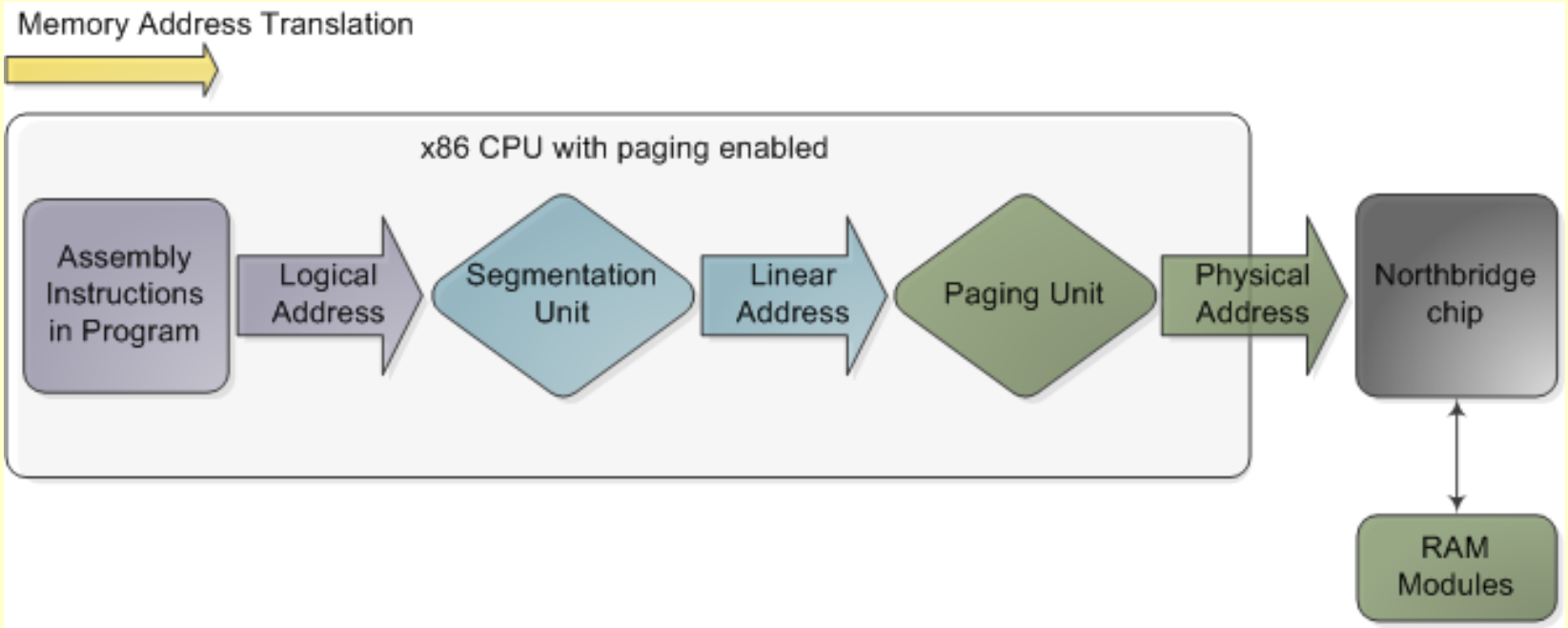
Model multi-segment

- Fiecare program are o tabelă locală de descriptori (LDT)
  - LDT conține un descriptor pentru fiecare segment utilizat de program

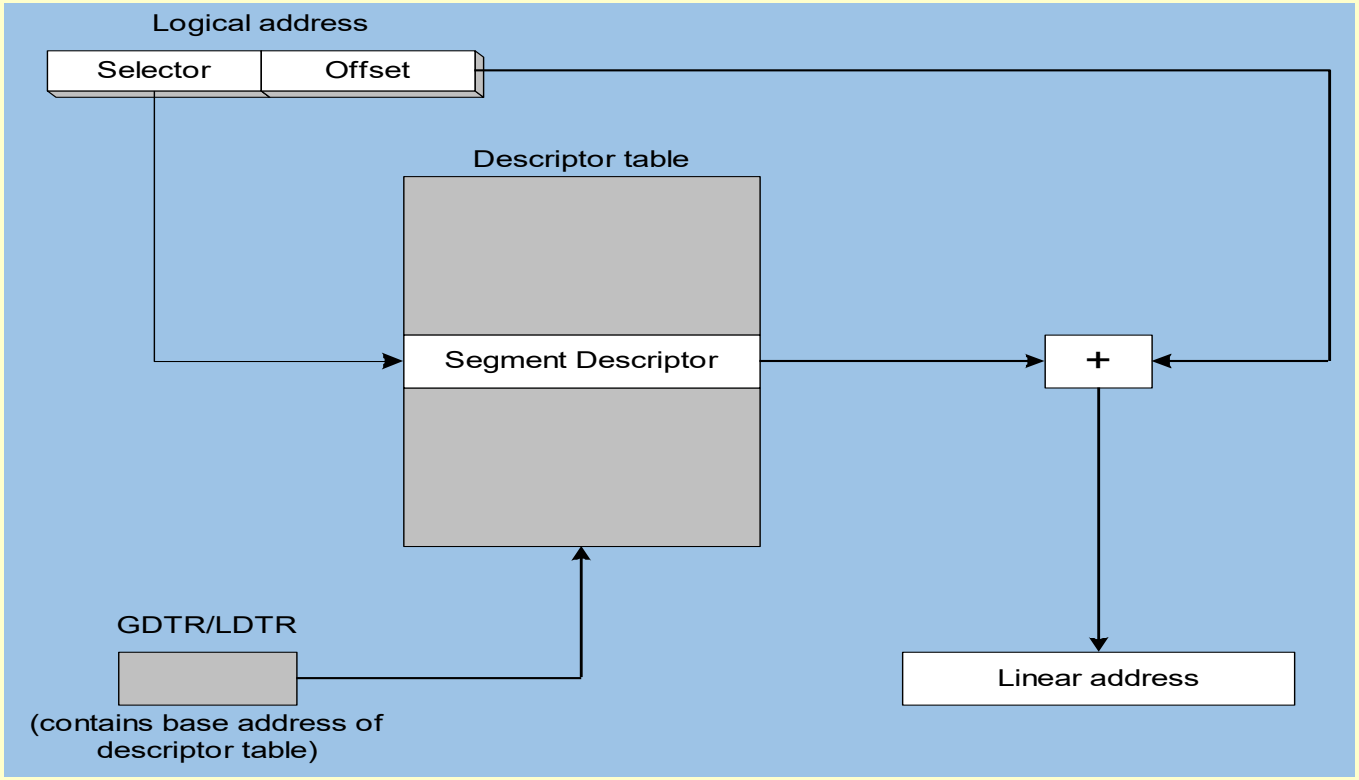


3.3 Translatarea Adreselor

- The IA-32 processor uses a one or two-step process to convert a variable's *adresa logică* a unei variabile într-o *locație unică de memorie*.
- Prima *etapă* combină o *valoare de segment* cu un offset de variabilă *offset* pentru a crea o *adresă liniară*
- Prima *second (optional) step*, called translatare de pagini, converts a linear address to a physical address



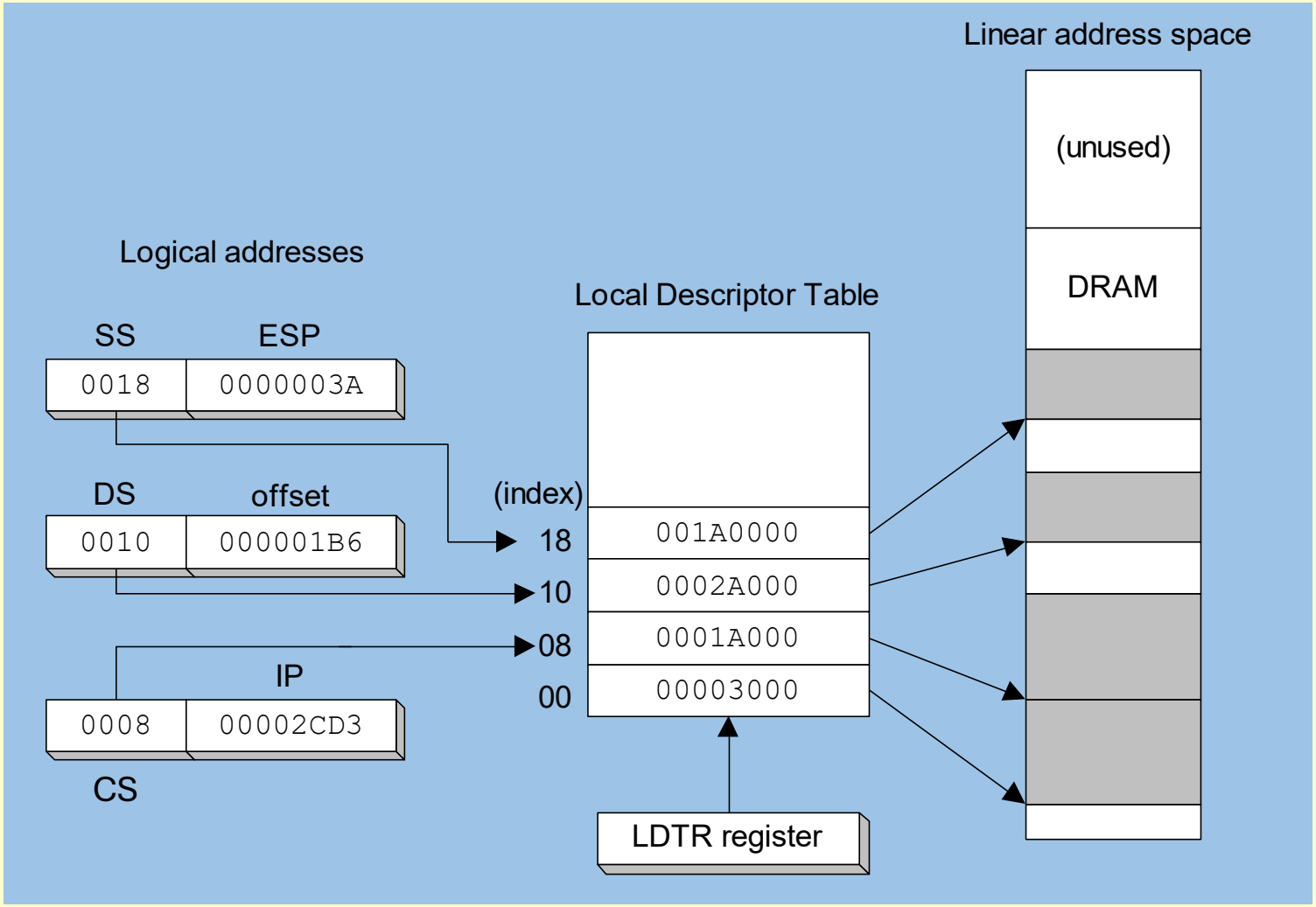
a. Conversia Adresei Logice în Adresă Liniară



- Selectorul de segment indică un descriptor de segment, which contains the base address of a memory segment.
- The 32-bit offset from the logical address is added to the segment's base address, generating a 32-bit adresă liniară pe 32 biți.

Indexare în Tabela de Descriptori

- Fiecare *descriptor de segment indexează* tabela ***tabelei locale de descriptori (LDT)***. Fiecare intrare din tabelă este mapată la o adresă liniară:

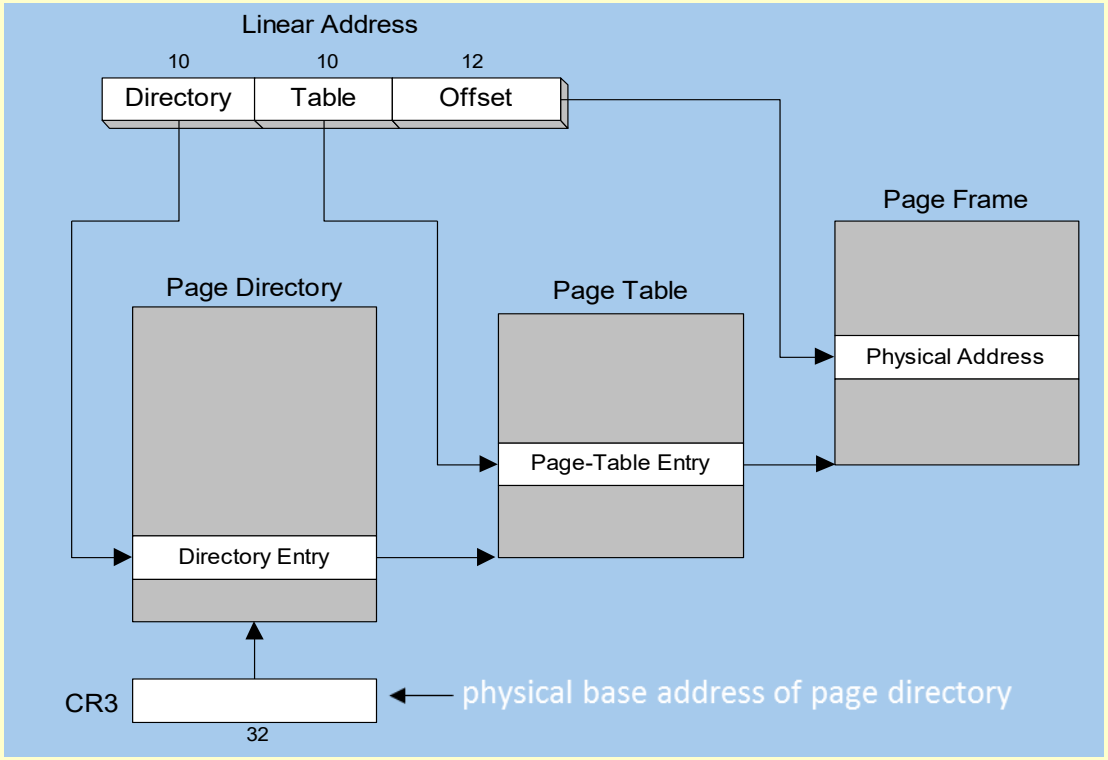


b. Paginare

- Memoria virtuală utilizează *HDD/SSD ca parte din memoria principală*, permițând ca suma tuturor programelor să depășească memoria fizică
- Doar o parte a unui program trebuie menținută în memorie, restul fiind păstrat pe disc.
- Memoria utilizată de program este împărțită în unități mai mici numite pagini (4kB)
- Pe măsură ce programul rulează, procesorul descarcă selectiv paginile inactive și încarcă altele necesare imediat

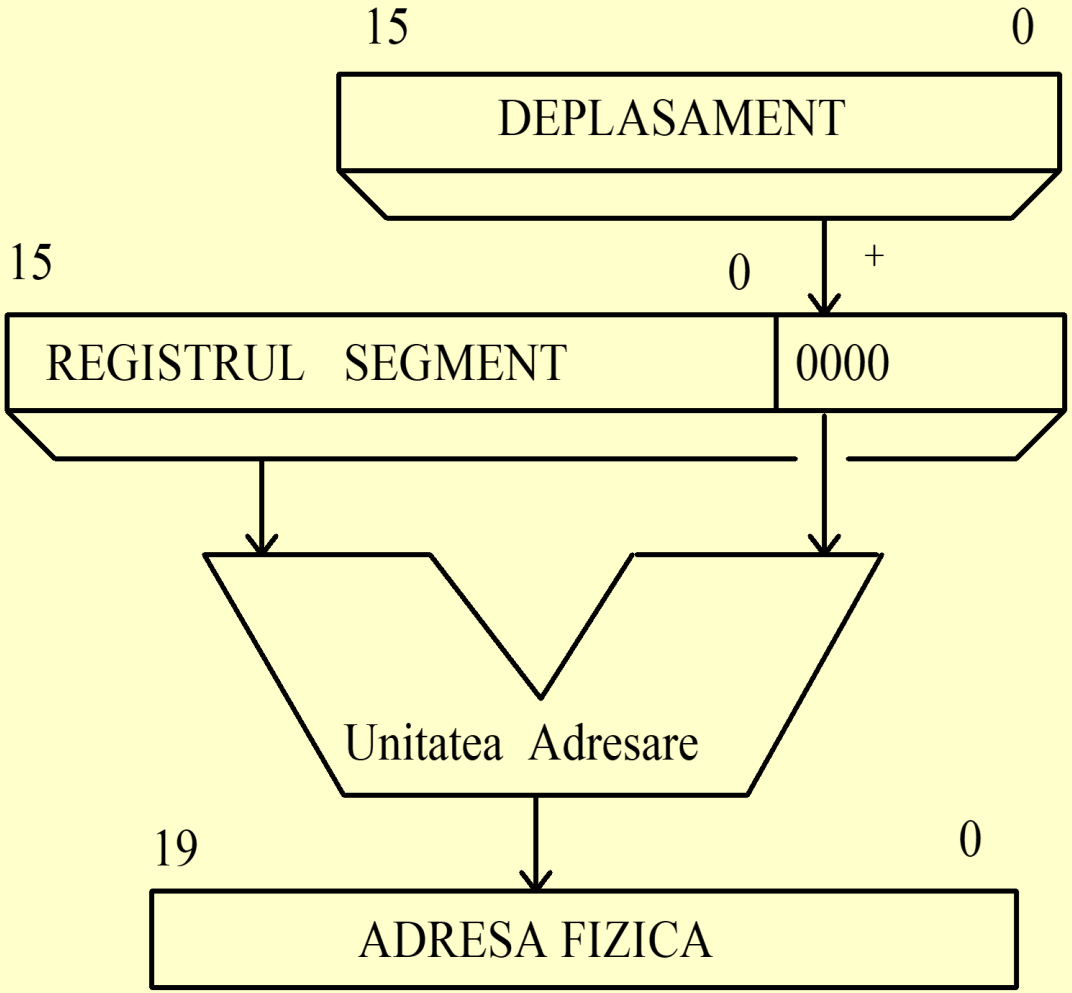
Translatarea Paginilor - CPU convertește adresa liniară >>>adresă fizică

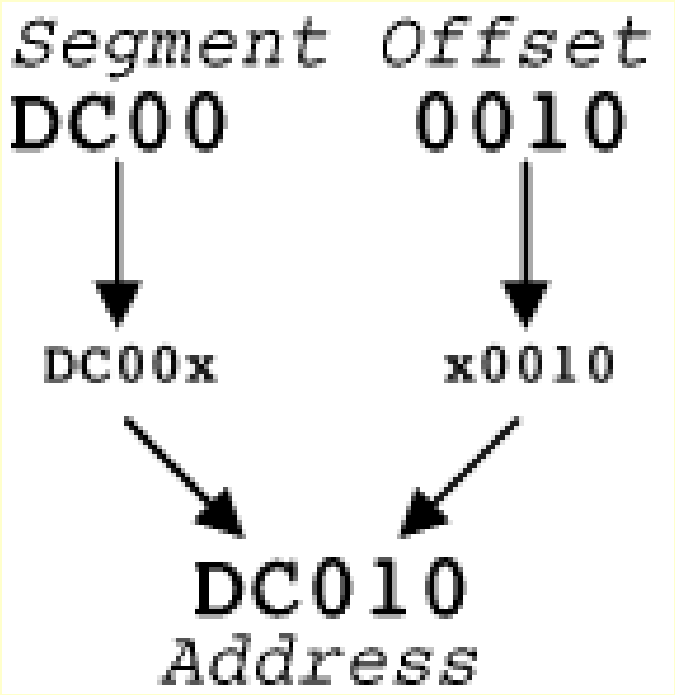
- O adresă liniară este împărțită în *câmpul directorului de pagini, câmpul tabelei de pagini și offsetul cadrului de pagini*. CPU utilizează toate trei pentru a calcula adresa fizică.
- SO menține directorul de pagini și și tabelele de pagini



- Eroare de pagină: apare când o pagină necesară nu se află în memorie și CPU întrerupe programul
- Gestionarul de memorie virtuală (VMM) – utilitar SO care gestionează încărcarea și descărcarea paginilor
- SO copiază pagina în memorie, programul își reia execuția

4. Organizarea memoriei in mod real de adresare





Modul de calcul al adresei fizice la 8086/286
in modul real de lucru

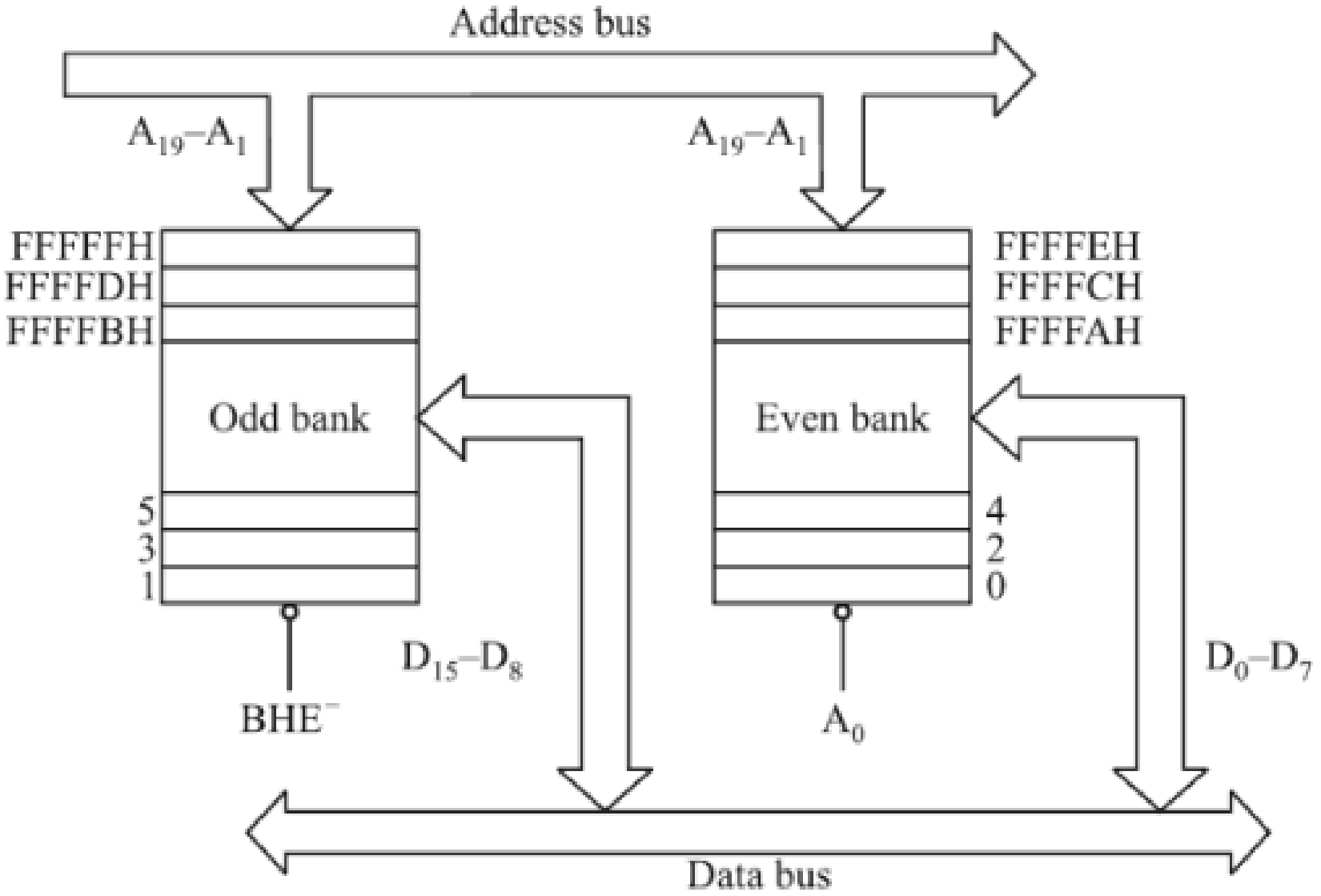
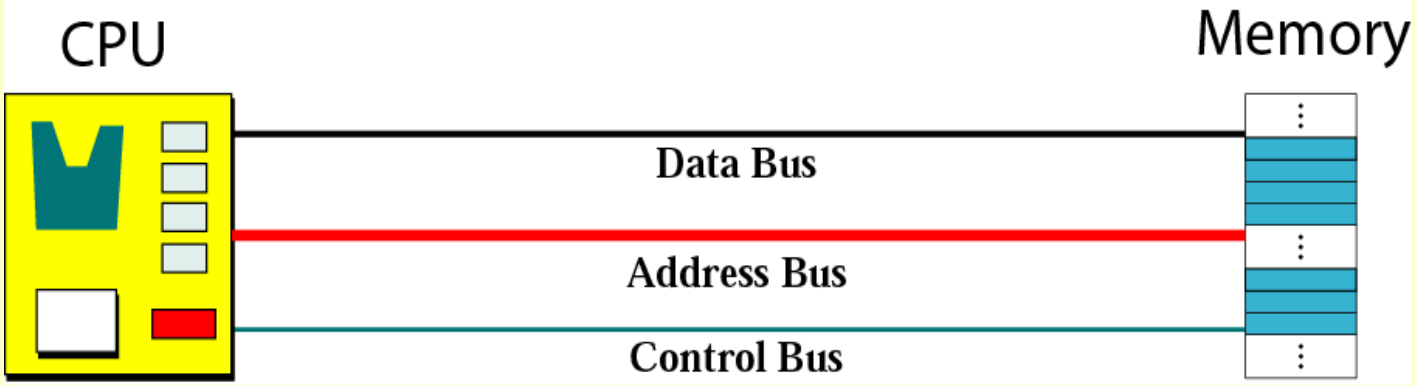
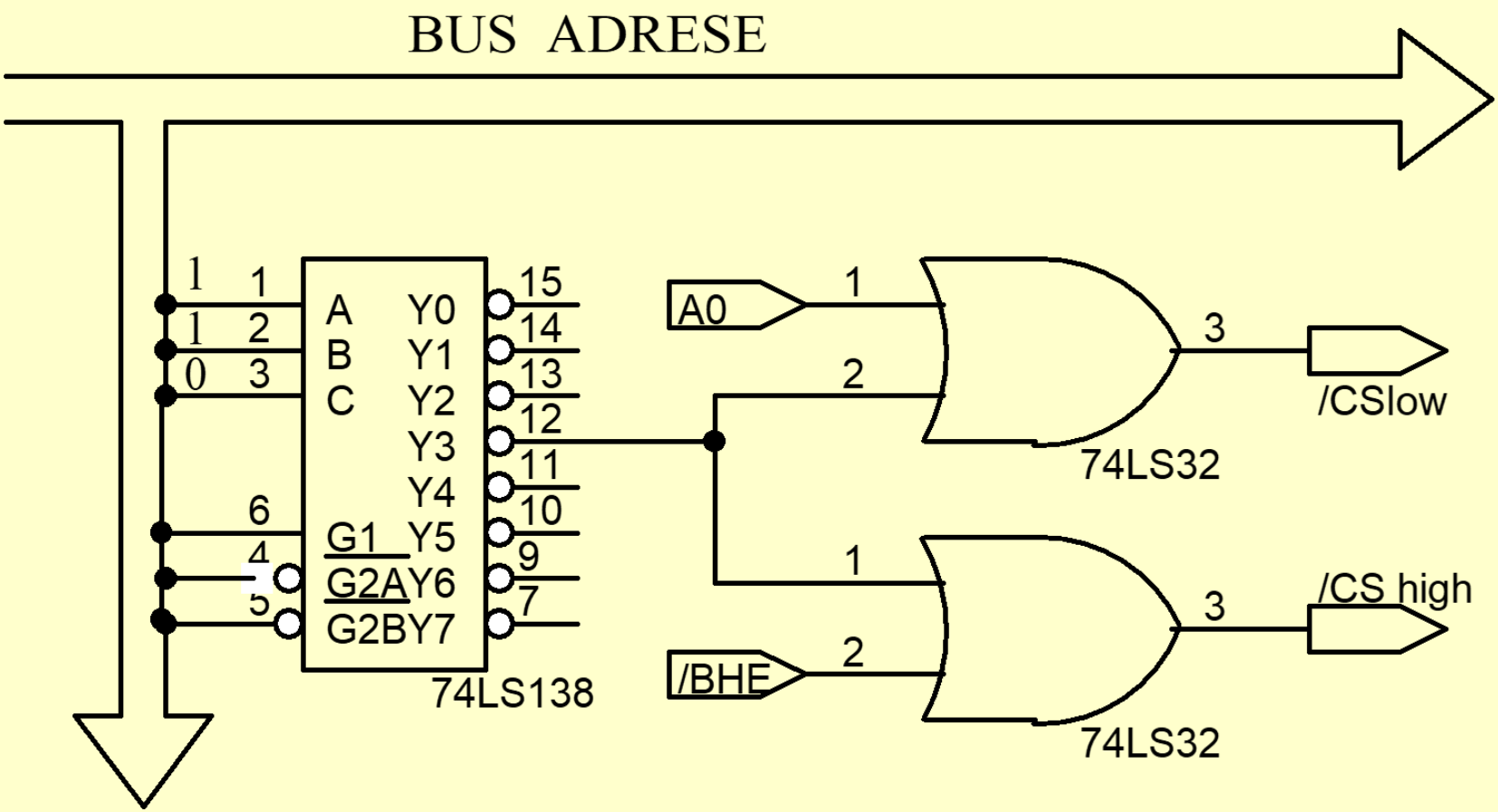


Figure 2.7 Memory banks.

https://www.plantation-productions.com/Webster/www.artofasm.com/Linux/HTML/SystemOrganizationa2.html





Circuit decodor pentru selectia memoriei

Byte or Word Transfer on Data Bus in 8086 System

Signal	*Byte to/from Even Address*	*Word to/from Even Address*	*Byte to/from Odd Address*	*Word to/from Odd Address*	
				I cycle	II cycle
A0	Low	Low	High	High	Low
$\overline{BHE}$	High	Low	Low	Low	High
Memory bank	Even	Even, odd	Odd	Odd	Even
D7-D0	DATA	DATA	Tri-state	Tri-state	DATA
D15-D8	Tri-state	DATA	DATA	DATA	Tri-state

Example 3.1

Describe the flow of data on the data bus for the following data transfer operations:

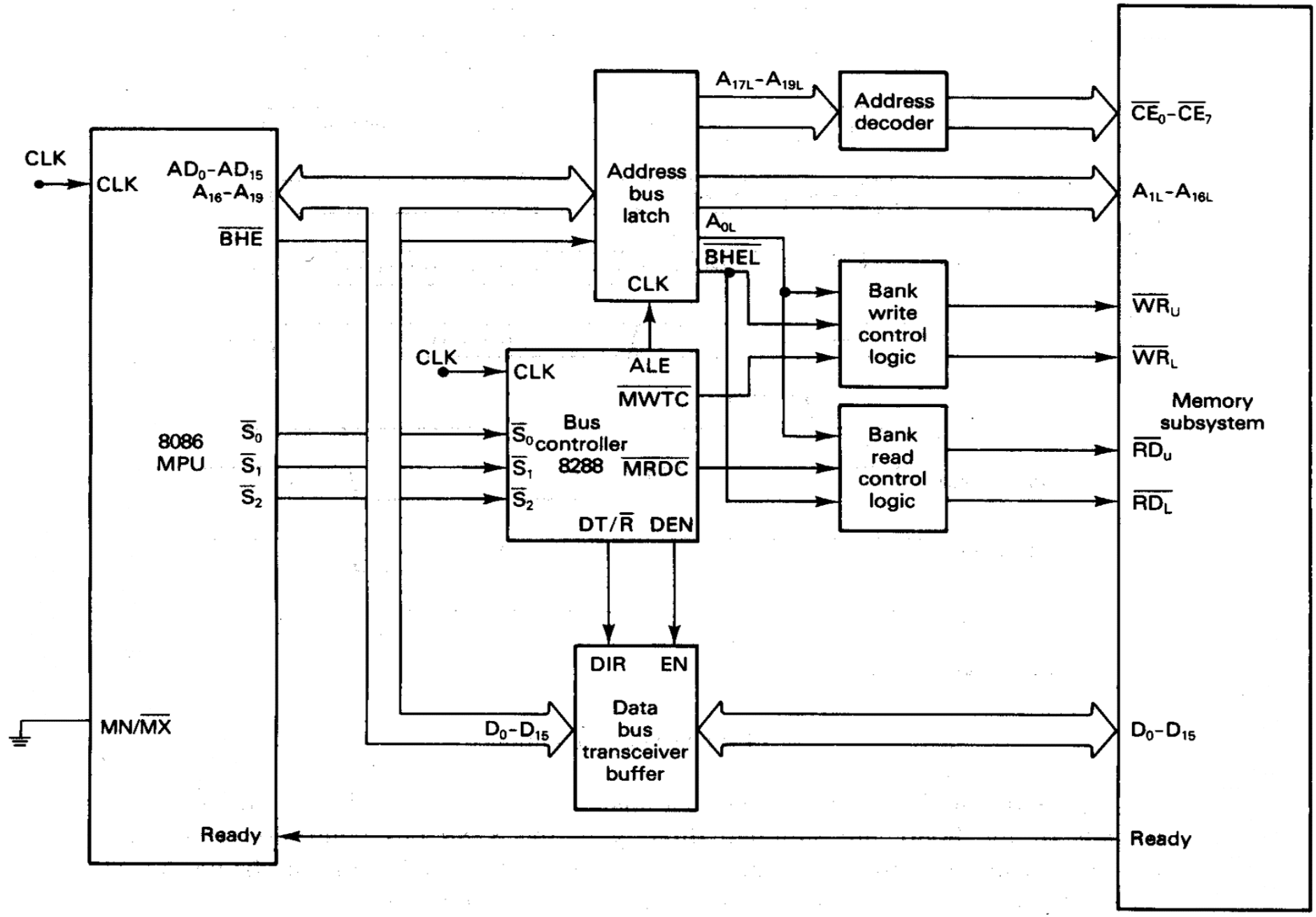
(i) Microprocessor writes a byte 11H into even addressed memory location 1000:0002H
(ii) Microprocessor writes a word 2211H into even addressed location 1000:0002H
(iii) Microprocessor writes a byte 11H into odd addressed location 1000:0003H
(iv) Microprocessor writes a word 2211H into odd addressed location 1000:0003H

Solution

Data flow for data transfer operations can be described in a table format as follows:.

Table 3.4 Data Flow on the Data Bus for Various Data Transfer Operations

Data transfer Operations	*i*	*ii*	*iii*	*iv*	
				I cycle	II cycle
A0	Low	Low	High	High	Low
$\overline{BHE}$	High	Low	Low	Low	High
Locations accessed	10002H	10002H, 10003H	10003H	10003H	10004H
Memory bank	Even	Even, odd	Odd	Odd	Even
D7-D0	11H	11H	Tri-state	Tri-state	22H
D15-D8	Tri-state	22H	11H	11H	Tri-state



Interfata cu memoria în modul "maxim" (MN/MX=„0")

Status Inputs			CPU Cycle	8288 Command
$\overline{S}_2$	$\overline{S}_1$	$\overline{S}_0$		
0	0	0	Interrupt acknowledge	$\overline{INTA}$
0	0	1	Read I/O port	$\overline{IORC}$
0	1	0	Write I/O port	$\overline{IOWC}$, $\overline{AIOWC}$
0	1	1	Halt	None
1	0	0	Instruction fetch	$\overline{MRDC}$
1	0	1	Read memory	$\overline{MRDC}$
1	1	0	Write memory	$\overline{MWTC}$, $\overline{AMWC}$
1	1	1	Passive	None

4. Memoria interna a PC in modul real de adresare

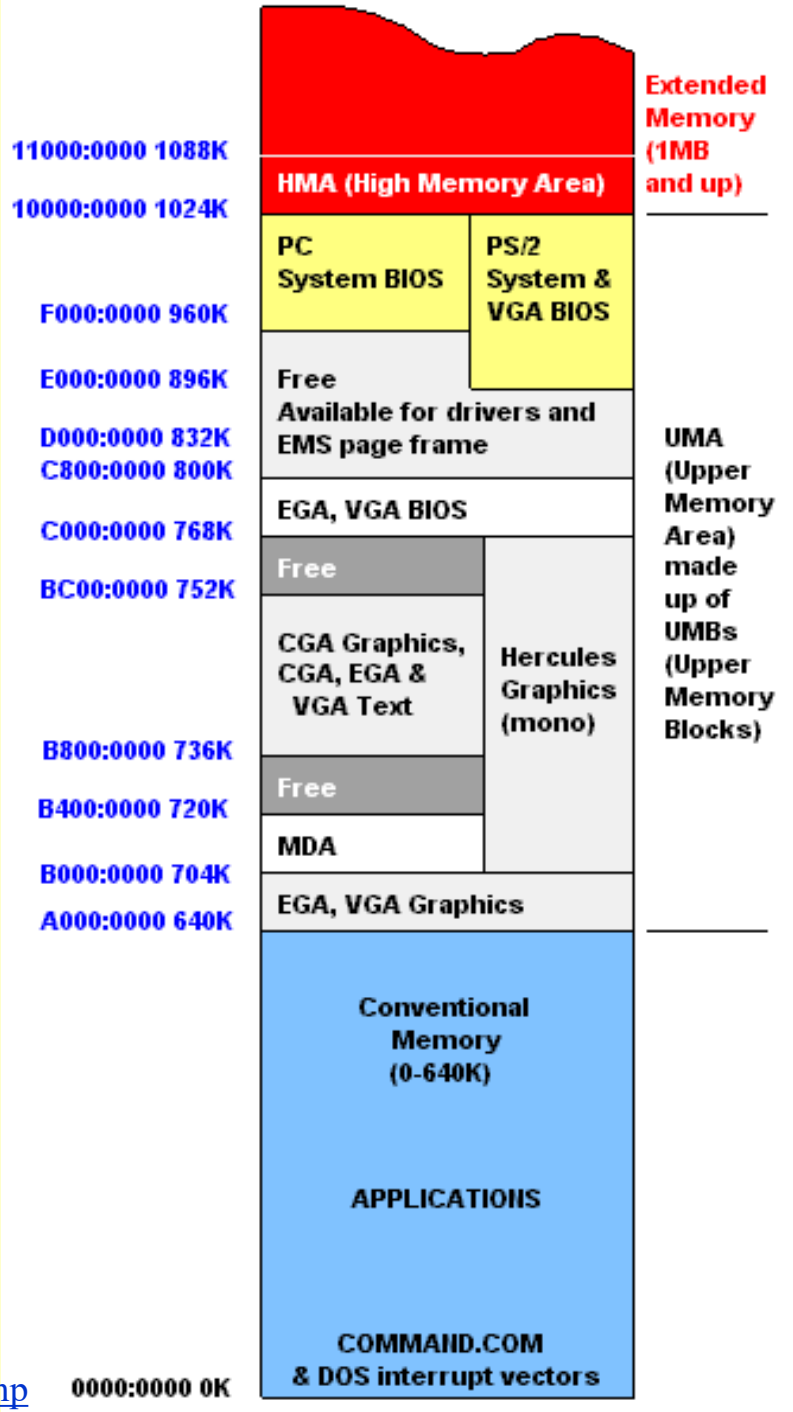
- Memoria de baza 0-9FFFFh
- Upper Memory Area-UMA (A0000h-FFFFFh)
  - → ROM Shadowing
  - → Mem. Expandata (EMS)*
  - → ROM Scan
- High Memory Area-HMA (100000h-10FFEFh)
- eXtended Memory Specification-XMS (>1Mo)

•EMS=Memorie aditionala in zona UMA accesibila prin “bank-switching”

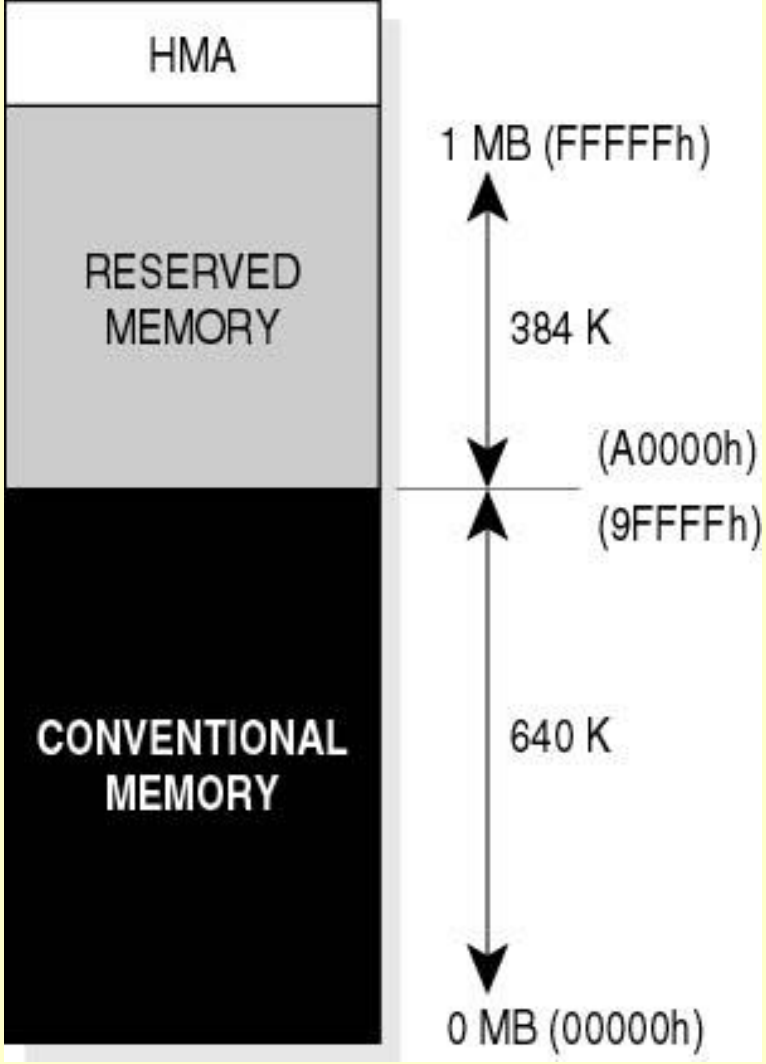
•FFFF:0.....FFFF:FFFF

•FFFF0.....10FFEF HMA

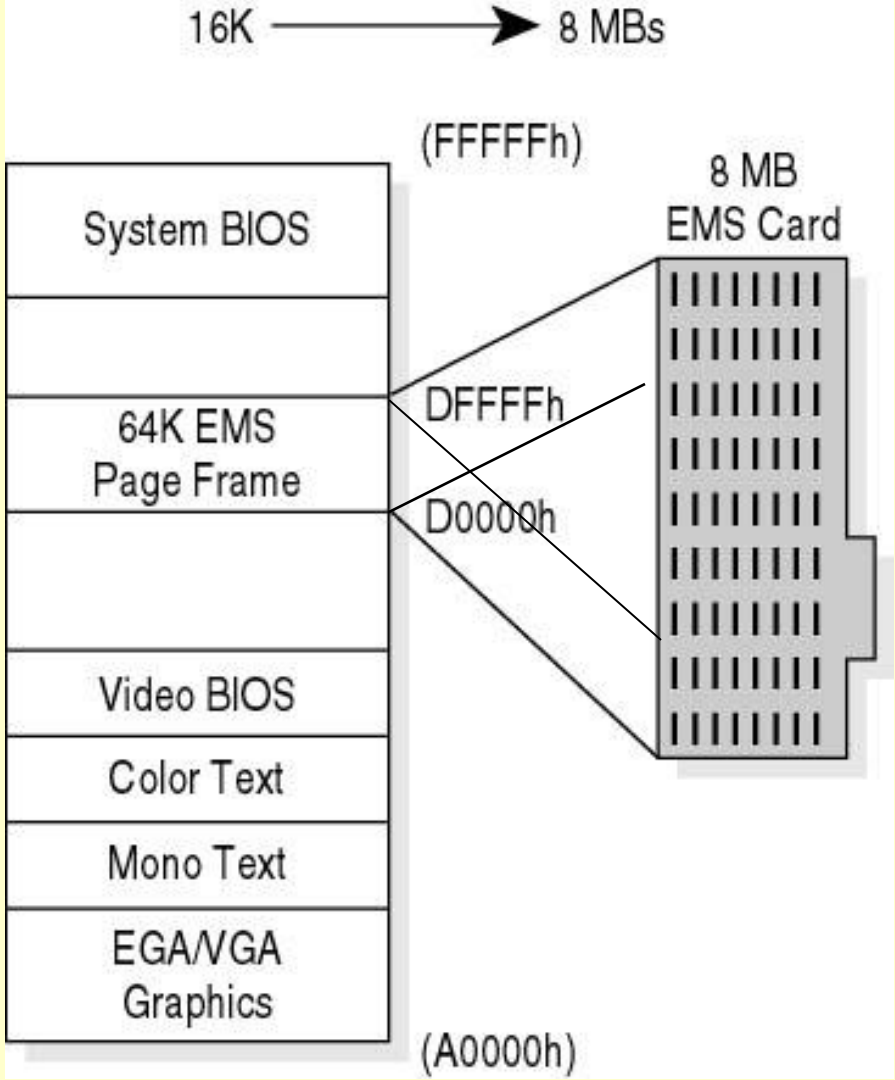
FFFF0+
FFFF
10FFEF



http://www.equestionanswers.com/bios-boot-pc-firmware/mmio-layout.php



High memory area

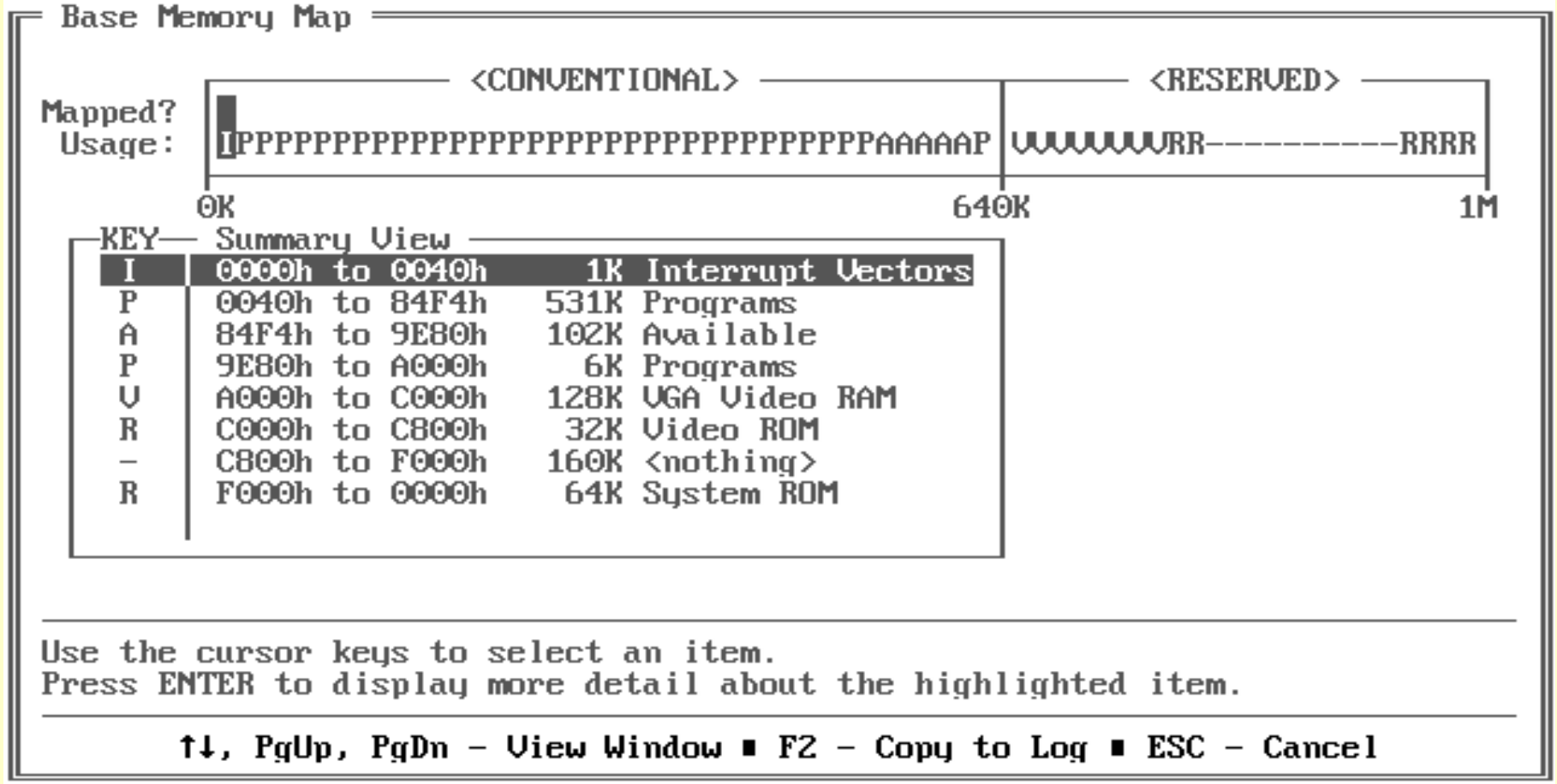


Expanded memory

FFFF:0.....FFFF:FFFF

FFFF0.....10FFEF

FFFF0+
FFFF
10FFEFh



Harta memoriei intr-un PC-AT

REFERINTE

http://savage.net.au/Ron/html/hex-ram-tutorial.html

http://en.wikipedia.org/wiki/Upper_Memory_Area

https://software.intel.com/en-us/articles/intelr-memory-latency-checker

5. Aplicatie. Conectati la un PC :

16kB de EPROM (2x 2764) incepand cu adresa CC000h si 16kB SRAM (2x6264) la adresa D0000h.

Figure 1. Logic Diagram

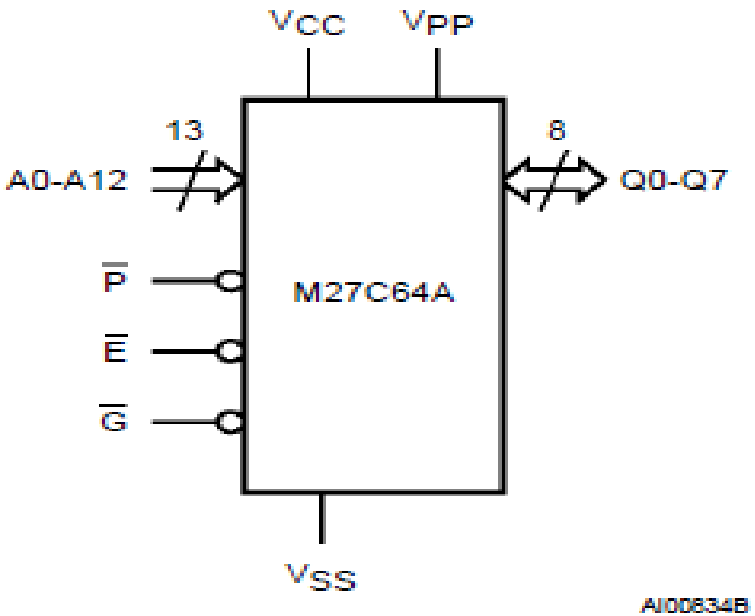


Table 1. Signal Names

A0 - A12	Address Inputs
Q0 - Q7	Data Outputs
$\overline{E}$	Chip Enable
$\overline{G}$	Output Enable
$\overline{P}$	Program
V_{PP}	Program Supply
V_{CC}	Supply Voltage
V_{SS}	Ground

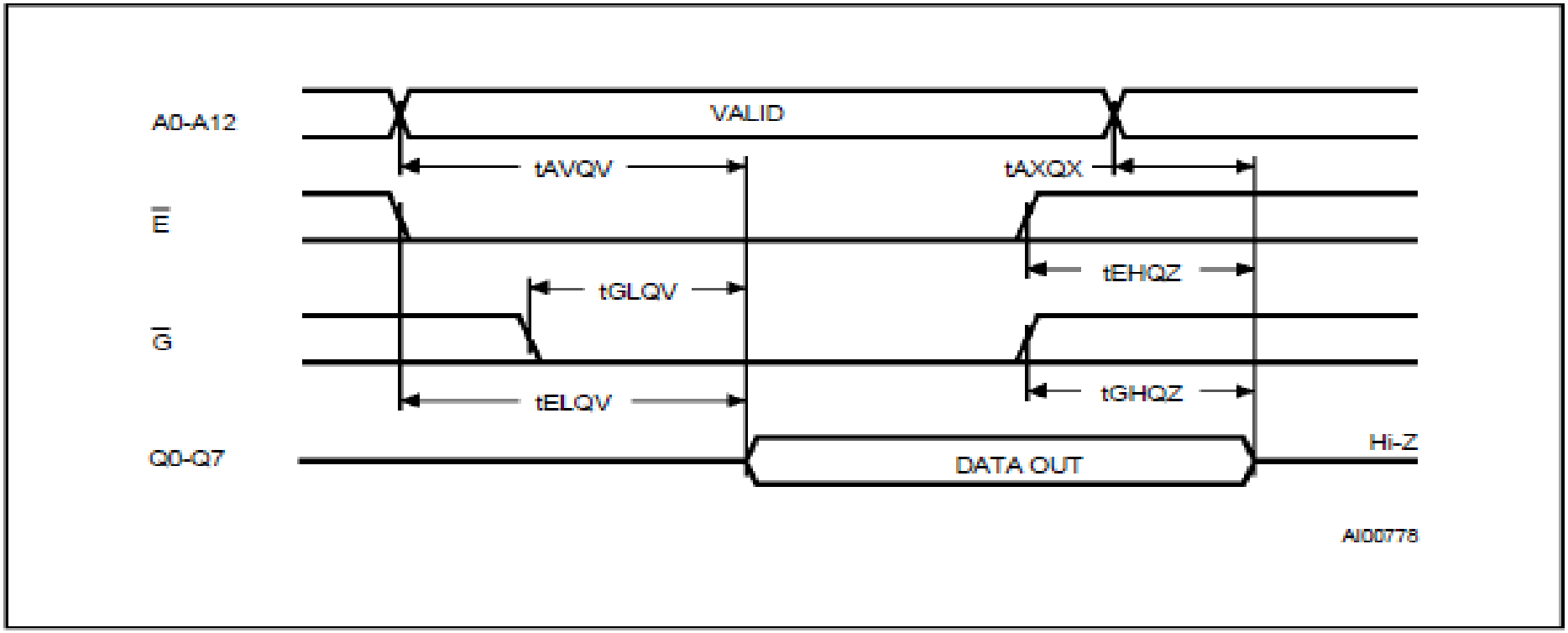
2764- 64K (8K x 8) UV EPROM

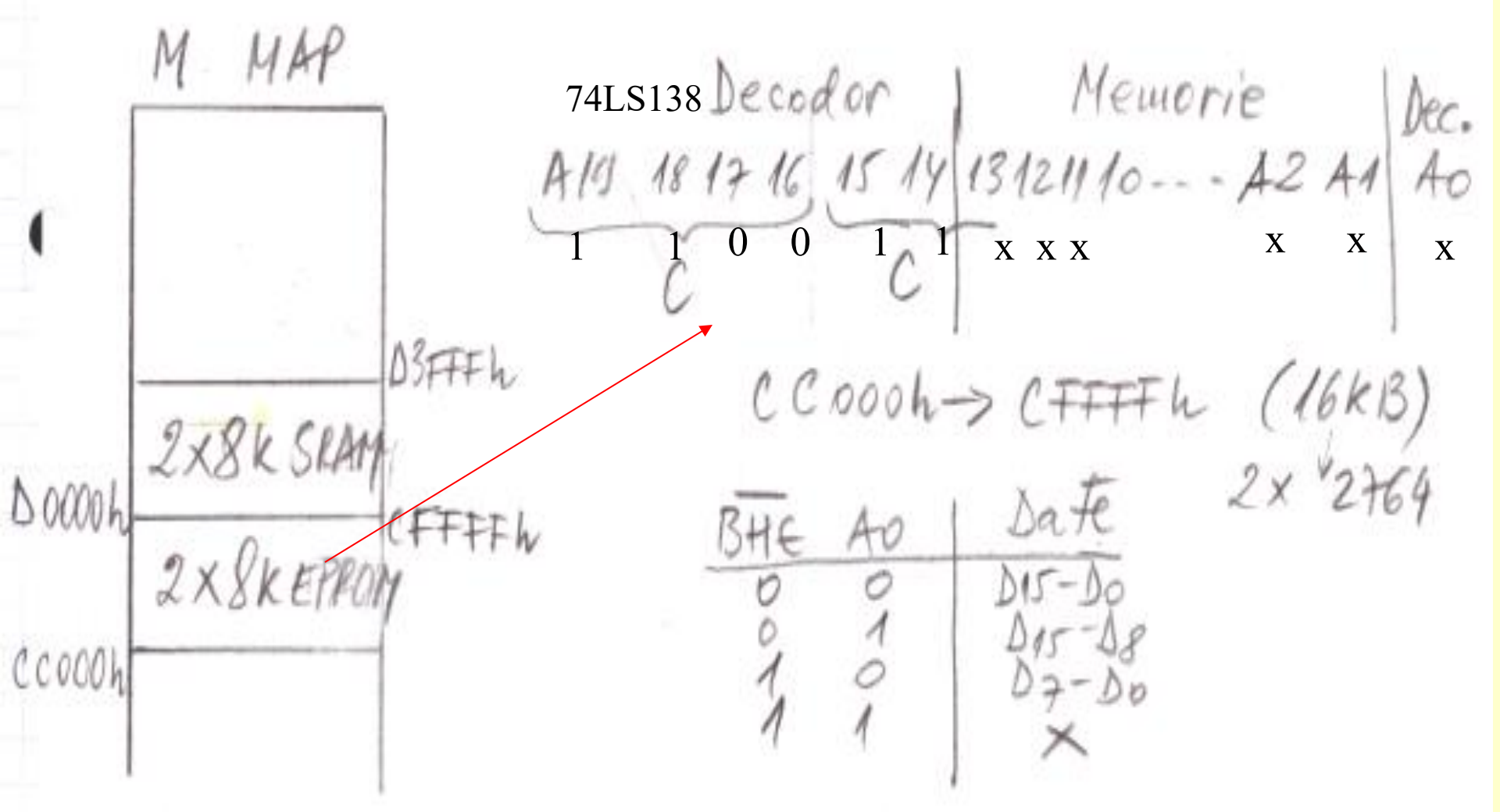
Table 3. Operating Modes

Mode	$\overline{E}$	$\overline{G}$	$\overline{P}$	A9	V_{PP}	Q0 - Q7
Read	V_{IL}	V_{IL}	V_{IH}	X	V_{CC}	Data Out
Output Disable	V_{IL}	V_{IH}	V_{IH}	X	V_{CC}	Hi-Z
Program	V_{IL}	V_{IH}	V_{IL} Pulse	X	V_{PP}	Data In
Verify	V_{IL}	V_{IL}	V_{IH}	X	V_{PP}	Data Out
Program Inhibit	V_{IH}	X	X	X	V_{PP}	Hi-Z
Standby	V_{IH}	X	X	X	V_{CC}	Hi-Z
Electronic Signature	V_{IL}	V_{IL}	V_{IH}	V_{ID}	V_{CC}	Codes

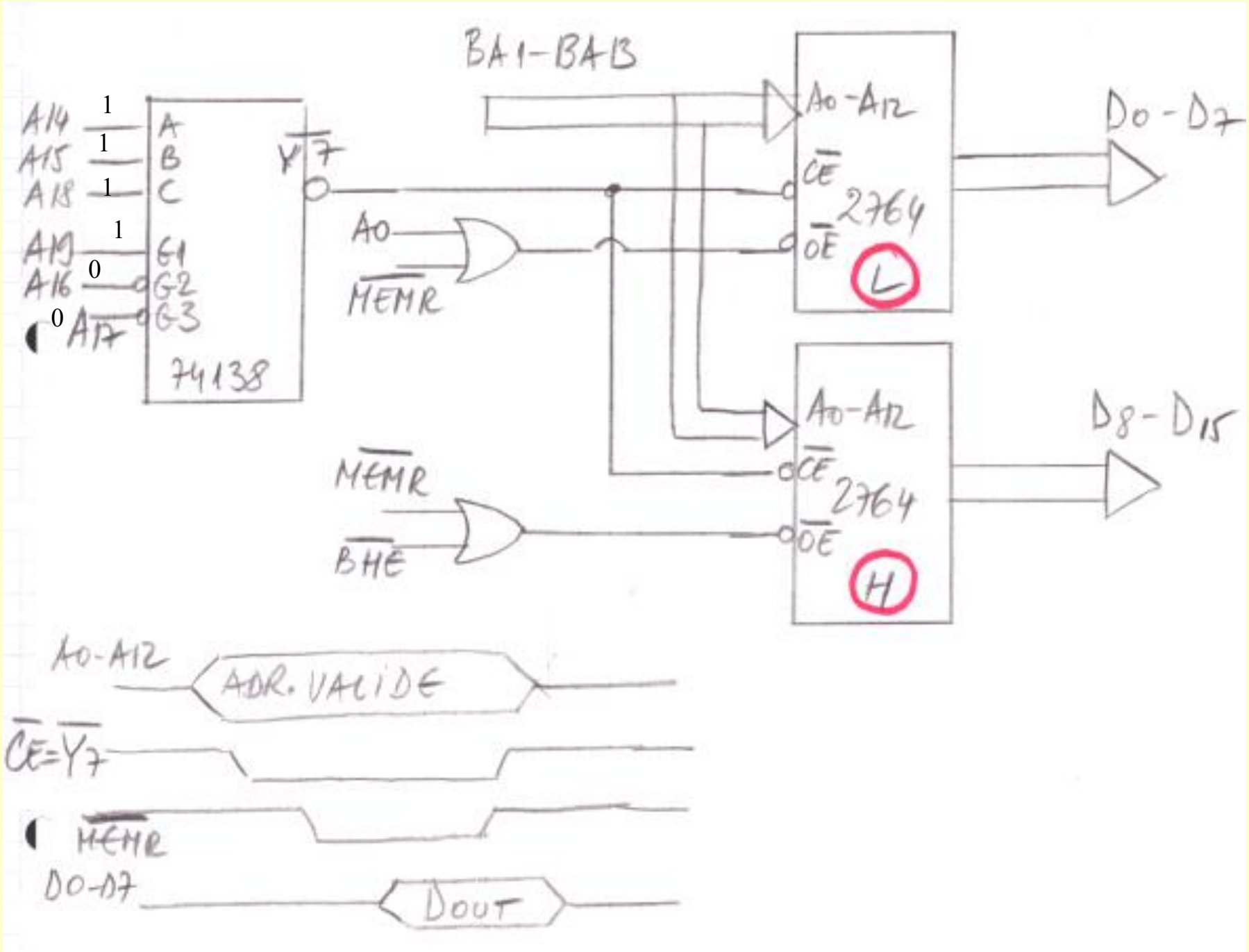
Note: X = V_{IH} or V_{IL}, V_{ID} = 12V ± 0.5V

Figure 5. Read Mode AC Waveforms

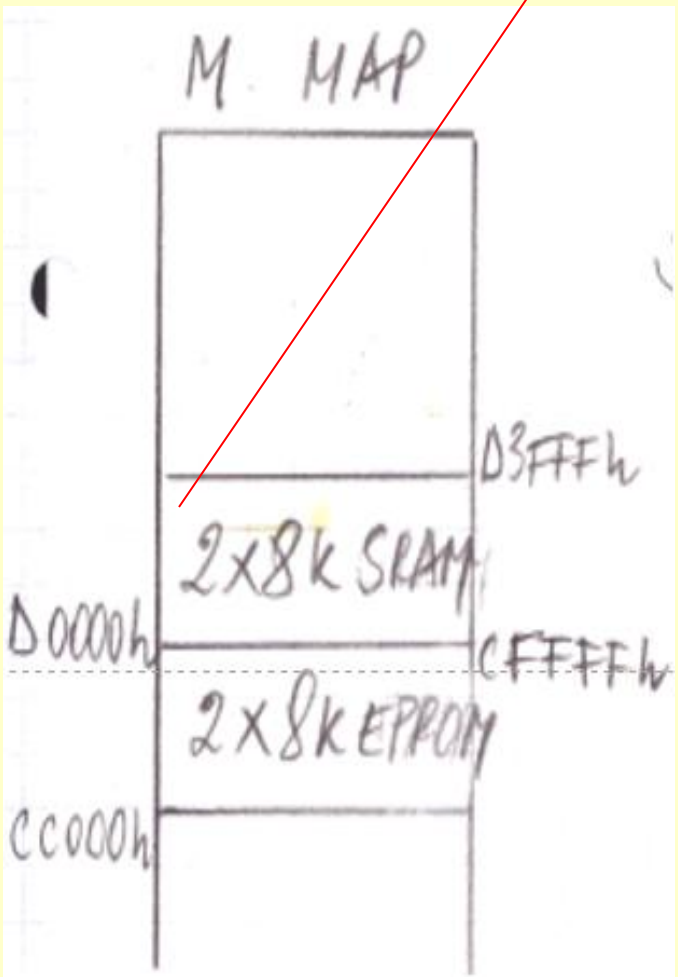


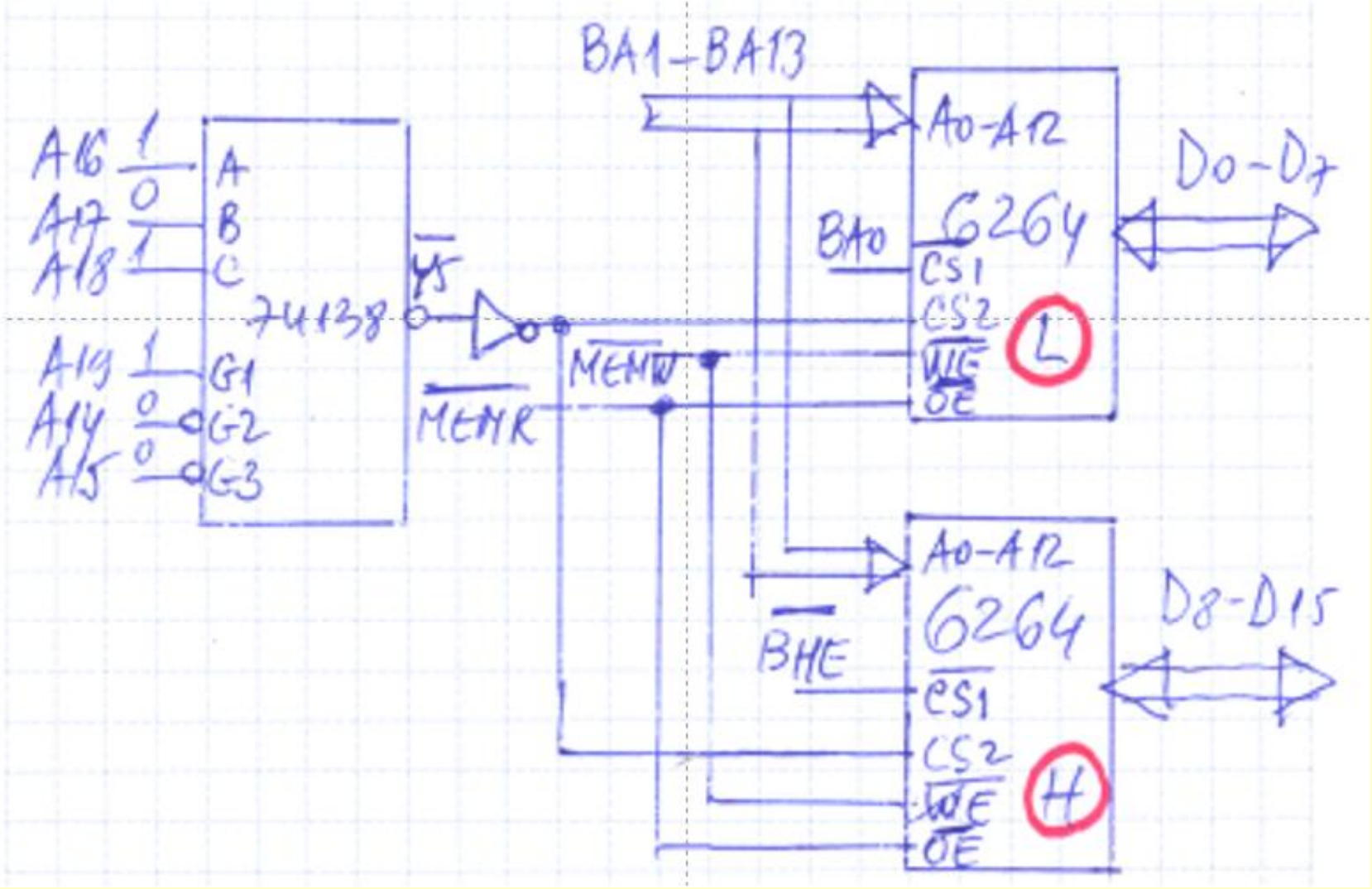


https://www.eeeguide.com/address-decoding-techniques-in-8086-microprocessor/

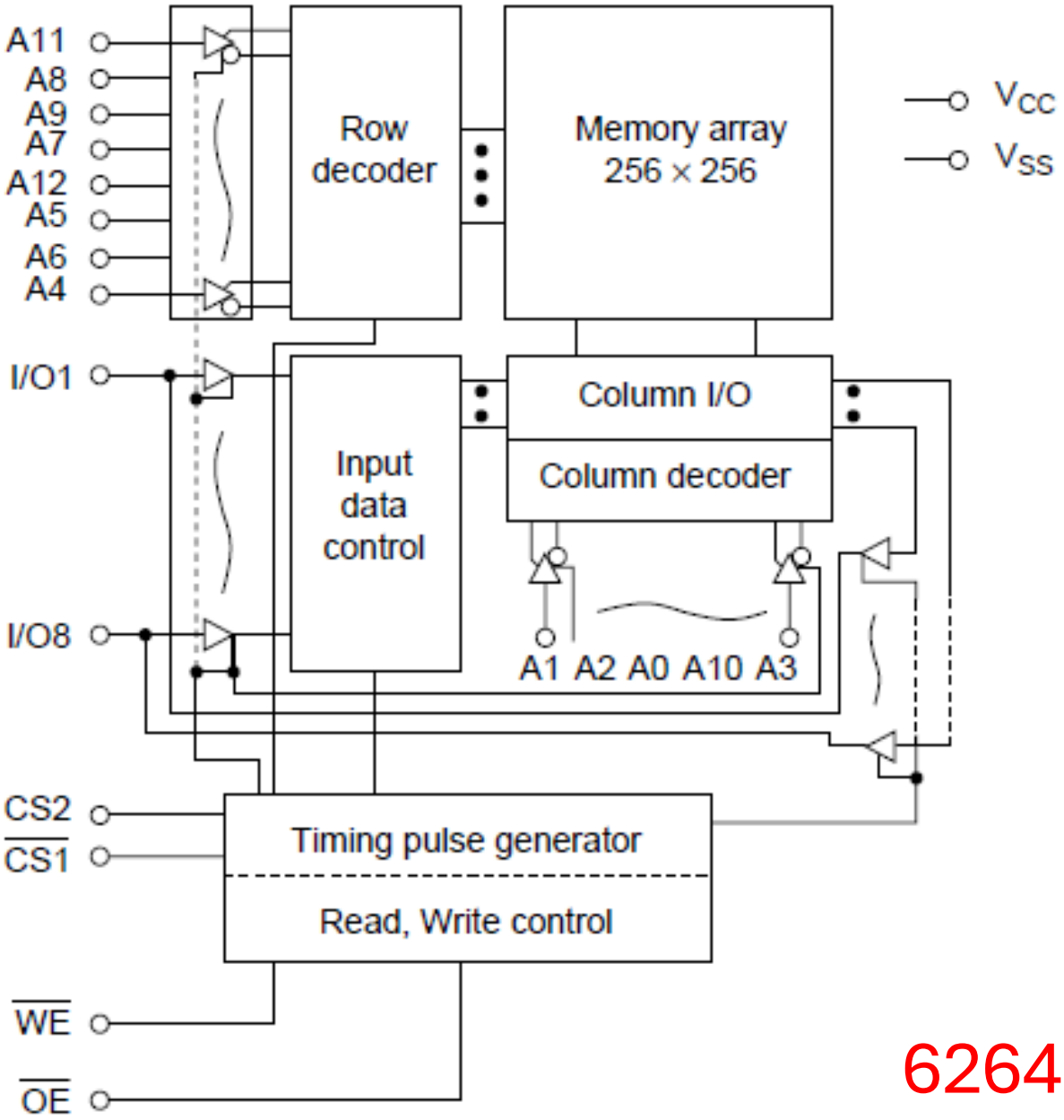

BA1-BAB
A14
A15
A18
A19
A16
A17
1
1
1
1
0
0
A
B
C
G1
G2
G3
74138
Y7
A0
MEMR
A0-A12
CE
OE
2764
L
D0-D7
A0-A12
CE
OE
2764
H
D8-D15
MEMR
BHE
A0-A12
ADR. VALIDE
CE=Y7
MEMR
D0-D7
DOUT

A19	18	17	16	15	14	13	12	...	A2	A1	A0
1	1	0	1	0	0	X	X		X	X	X
Decodor						Memorie					Dec





A19	18	17	16	15	14	13	12	...	A2	A1	A0
1	1	0	1	0	0	X	X		X	X	X
Decodor						Memorie					Dec


A11
A8
A9
A7
A12
A5
A6
A4
Row decoder
Memory array
256 × 256
Vcc
Vss
I/O1
I/O8
Input data control
Column I/O
Column decoder
A1 A2 A0 A10 A3
CS2
CS1
Timing pulse generator
Read, Write control
WE
OE

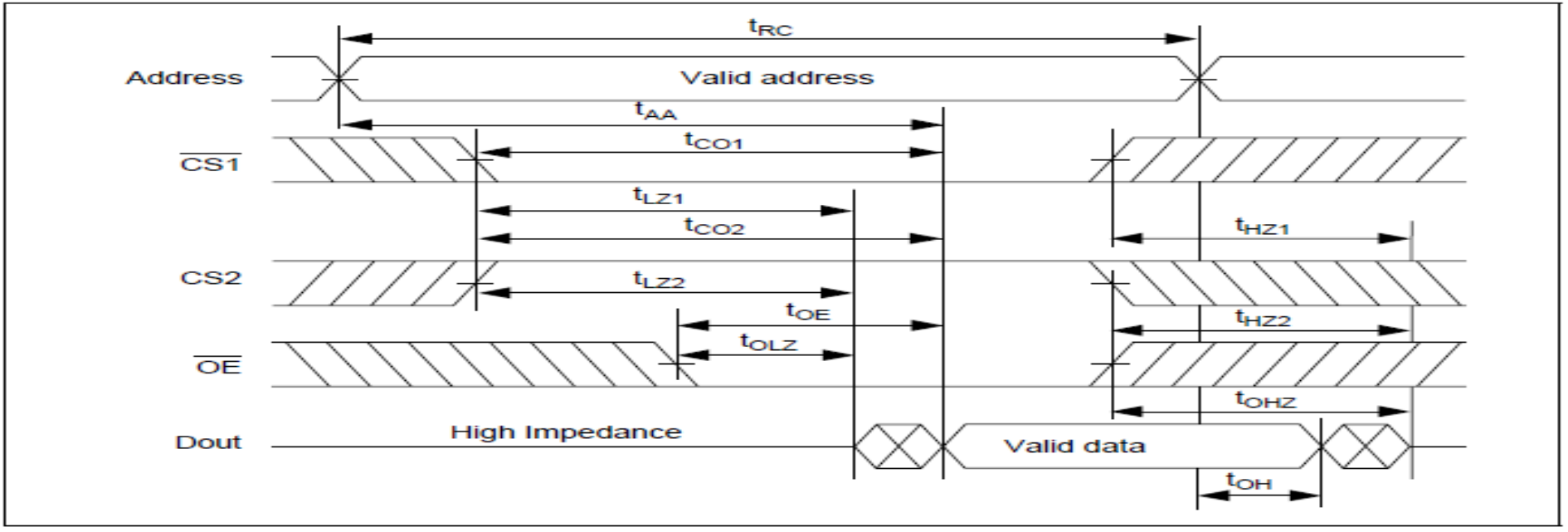
6264 SRAM

Function Table

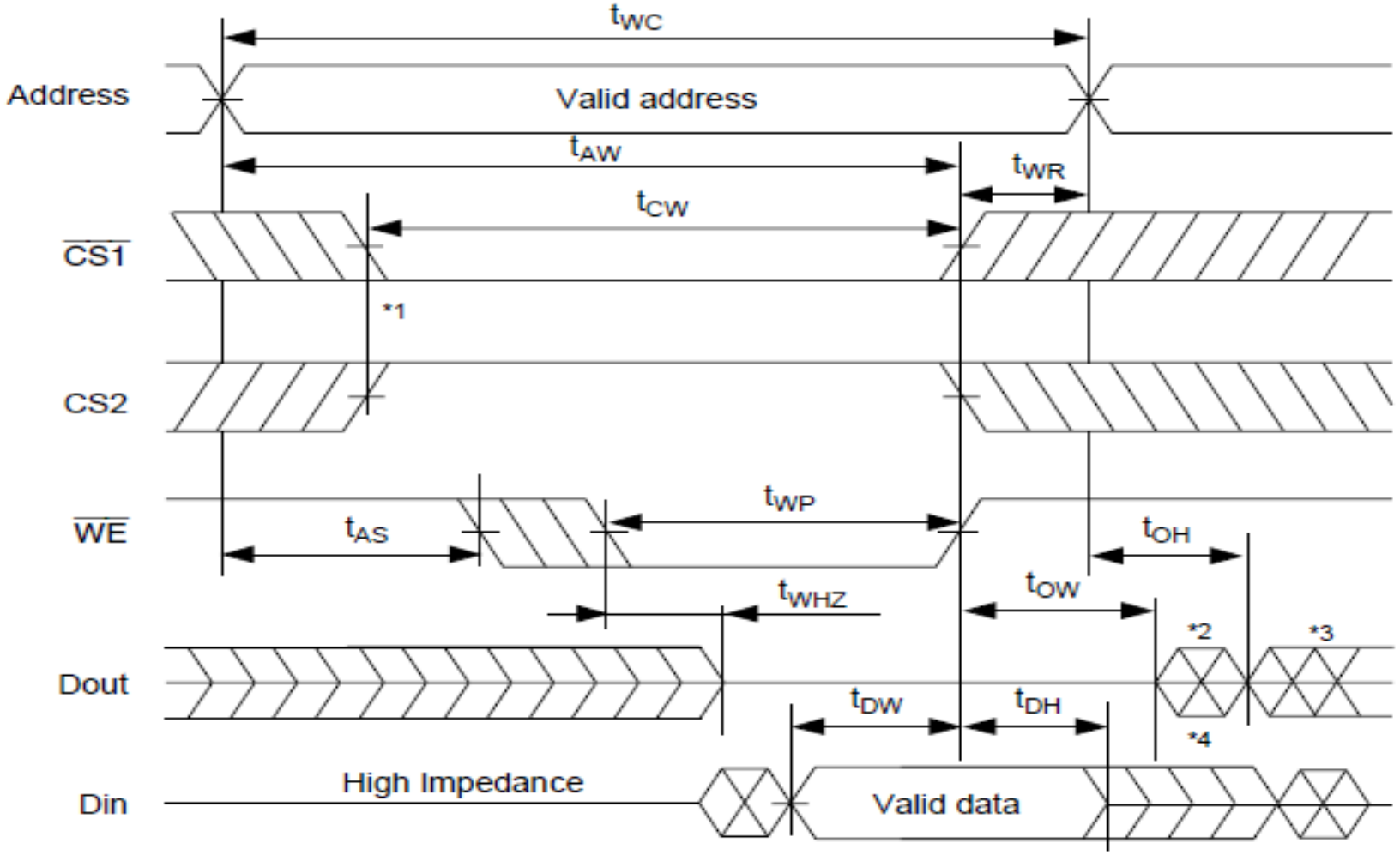
$\overline{WE}$	$\overline{CS1}$	CS2	$\overline{OE}$	Mode	V_{CC} current	I/O pin	Ref. cycle
×	H	×	×	Not selected (power down)	I_{SB}, I_{SB1}	High-Z	—
×	×	L	×	Not selected (power down)	I_{SB}, I_{SB1}	High-Z	—
H	L	H	H	Output disable	I_{CC}	High-Z	—
H	L	H	L	Read	I_{CC}	Dout	Read cycle (1)–(3)
L	L	H	H	Write	I_{CC}	Din	Write cycle (1)
L	L	H	L	Write	I_{CC}	Din	Write cycle (2)

Note: ×: H or L

Read Timing Waveform (1) ($\overline{WE} = V_{IH}$)



Write Timing Waveform (2) ($\overline{OE}$ Low Fixed) ($\overline{OE} = V_{IL}$)



Notes:

1. If $\overline{CS1}$ goes low simultaneously with $\overline{WE}$ going low or after $\overline{WE}$ goes low, the outputs remain in high impedance state.
2. Dout is the same phase of the written data in this write cycle.
3. Dout is the read data of the next address.
4. If $\overline{CS1}$ is low and CS2 is high during this period, I/O pins are in the output state. Input signals of opposite phase to the outputs must not be applied to I/O pins.

TEMA 1. Memoria DRAM

Memoria RAM dinamică (DRAM) este memoria cu cea mai mare densitate si cel mai mic pret dintre memoriile disponibile în prezent. Din aceste motive, sunt universal utilizate în orice sistem bazat pe microprocesor care necesită mai mult decât o mică memorie de stocare nevolatilă, care poate stoca date.

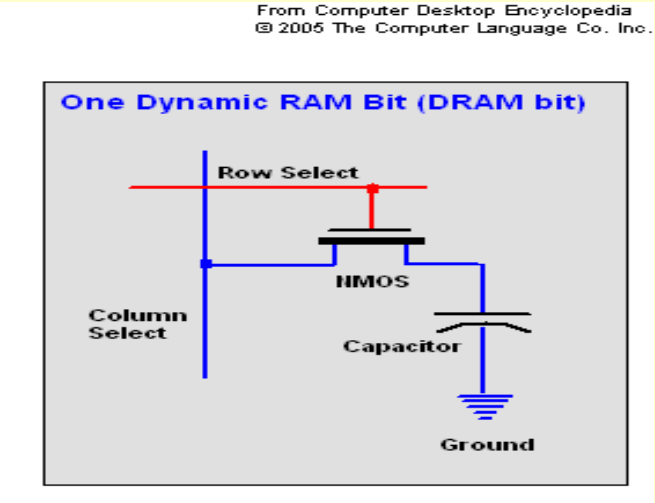
✓ Un tranzistor pe celulă (+ condensator)

✓Sunt implicate sarcini foarte mici in stocarea informatiei

- liniile de biți trebuie preîncărcate pentru a detecta valorile de biți
- oscilația tensiunii pe liniile de biți este mică; sunt necesari senzori amplificatori pentru a converti tensiunile in nivele logice
- citirile sunt distructive; dispozitivele DRAM scriu intern datele înapoi la citire
- curentul de scurgere poate comuta nivelul 1 la 0: valorile trebuie actualizate (rescrise) la fiecare câteva milisecunde sau datele vor fi pierdute

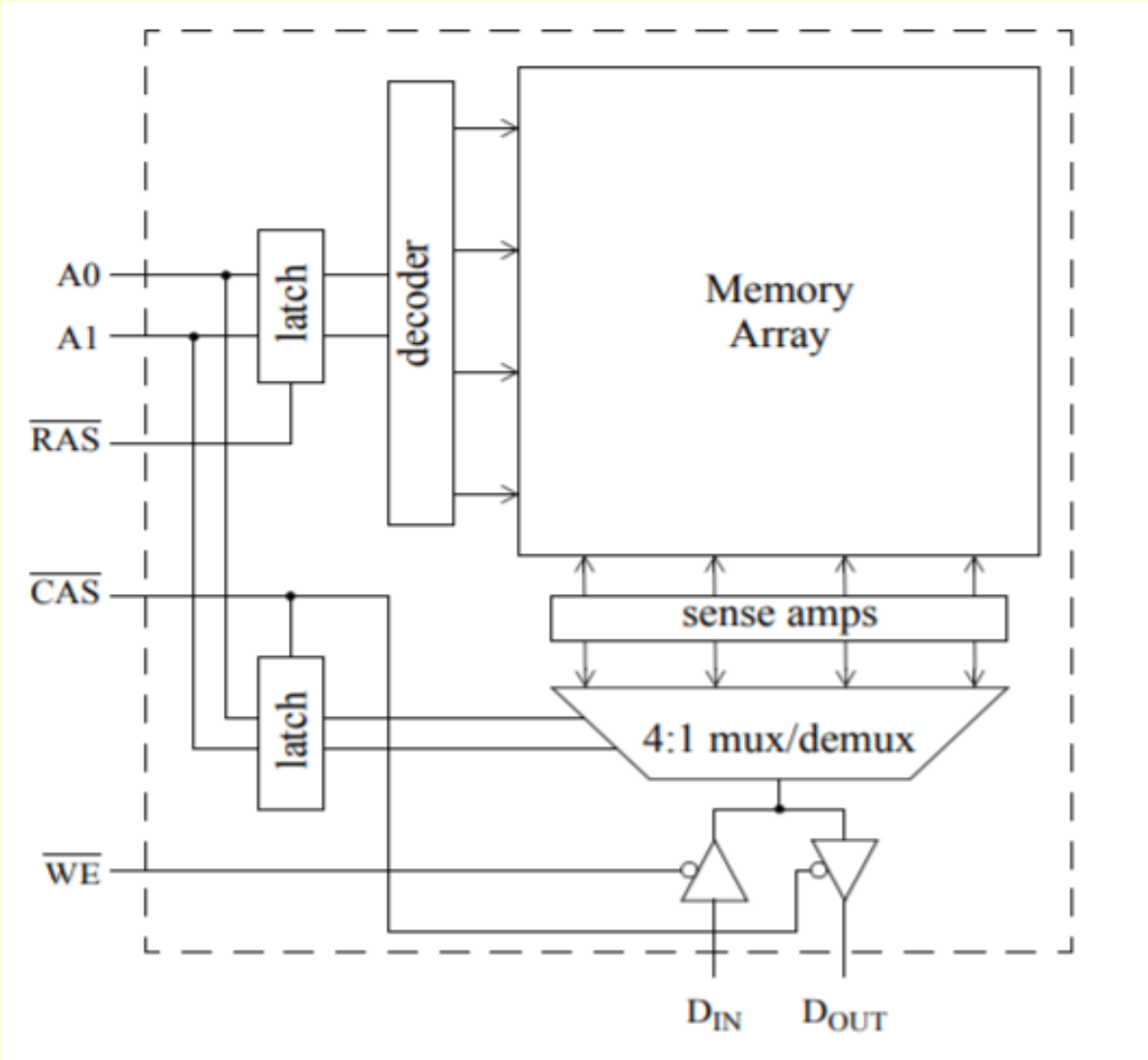
✓Pentru a reduce costul și dimensiunea capsulei, la memoriile DRAM se minimizează numărul de pini:

- folosind configurații logice înguste (x1, x4...) biti
- multiplexarea adreselor de rând și de coloană interne pe aceiași pini



Intern, DRAM-urile seamănă cu alte memorii, cu excepția:

- Semnalele RAS și CAS strobeaza latch-urile de rând și coloană ale jumătăților de adresa multiplexate
- CAS poate servi și ca activare de ieșire (output enable)
- Majoritatea dispozitivelor x1 au pini de date de intrare și ieșire separate

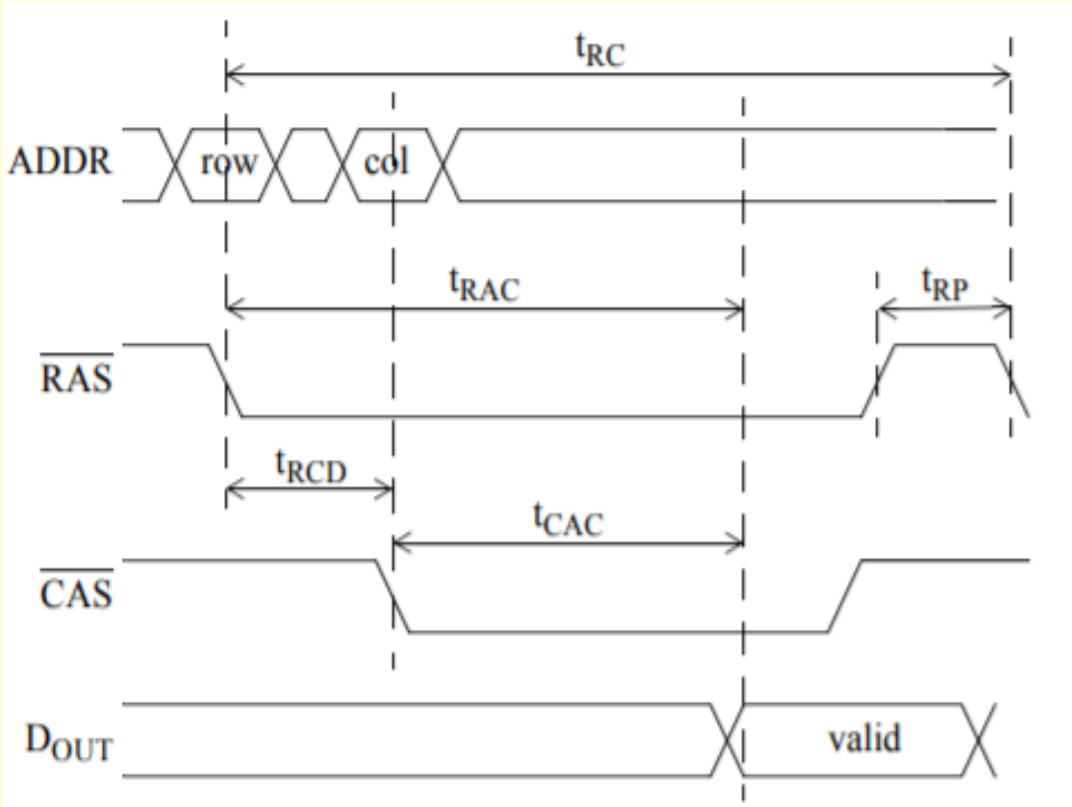


Citire DRAM

•*Fronturile căzatoare ale RAS și CAS* strobează biții de adresă în latch-urile rândului și, respectiv, ale coloanei. Acestea trebuie separate de cel puțin t_{RCD}.

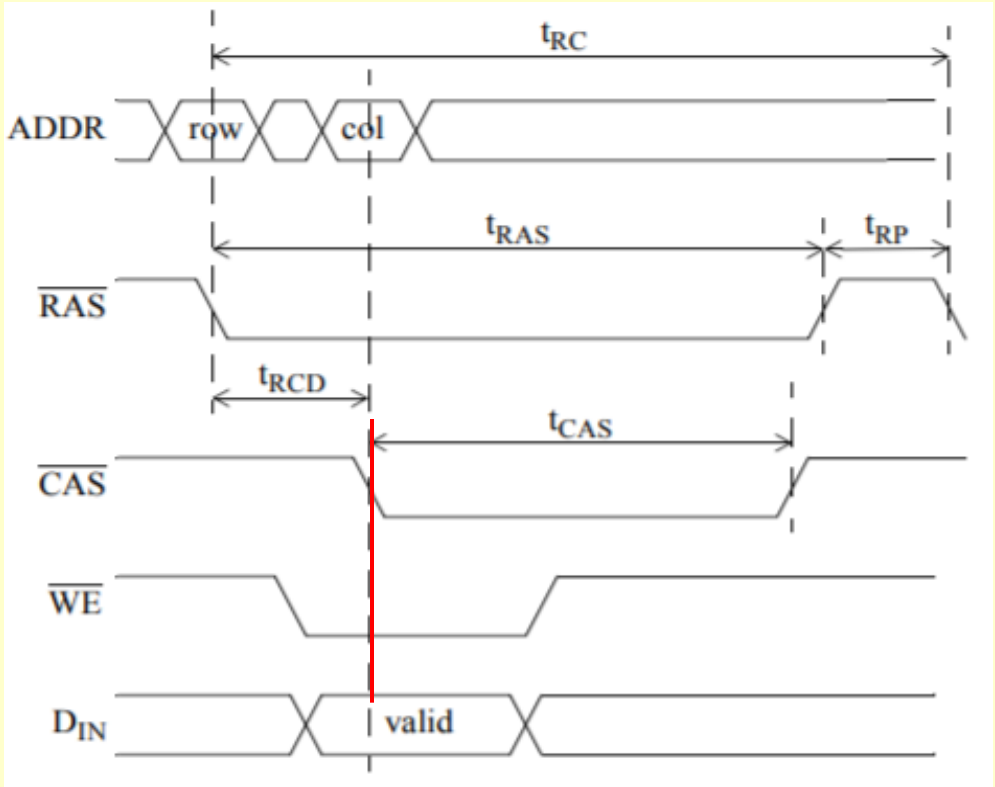
• Ca și în cazul altor memorii, sunt specificati mai multi timpi de acces, iar timpul până la validarea datelor va depinde de care este calea critică. Pentru DRAM-uri, există doi timpi de acces, t_{RAC} și t_{CAC}, pentru timpul de acces de la adresa de rând validă, respectiv adresa de coloană validă.

- Timpii de setup și de mentinere pentru adresele de rând și de coloană există, dar nu sunt afișati.
- *Timpul ciclului de citire* ***(t_{RC})*** este de obicei mult mai mare decât timpul de access, datorită *timpului de reîncărcare necesar* t_{RP} (nu este reprezentat la scară).



Scriere DRAM

- Scrierea in DRAM este similara. Dacă WE este valid pe frontul descendent al CAS, datele sunt scrise din D_{IN} în loc să fie citite în D_{OUT}. De reținut că acest lucru este diferit de SRAM-uri, care efectuează scrierile la sfârșitul ciclului (frontul ascendent al WE).
- Majoritatea parametrilor de sincronizare sunt identici cu ciclul de citire. (t_{RAS} și t_{CAS} au lățimi de impuls minime care se aplică și ciclului de citire, dar au fost lăsate în afara diagramei pentru claritate.)
- Setup-ul necesar și timpii de păstrare pe D_{IN} și WE nu sunt afișați.



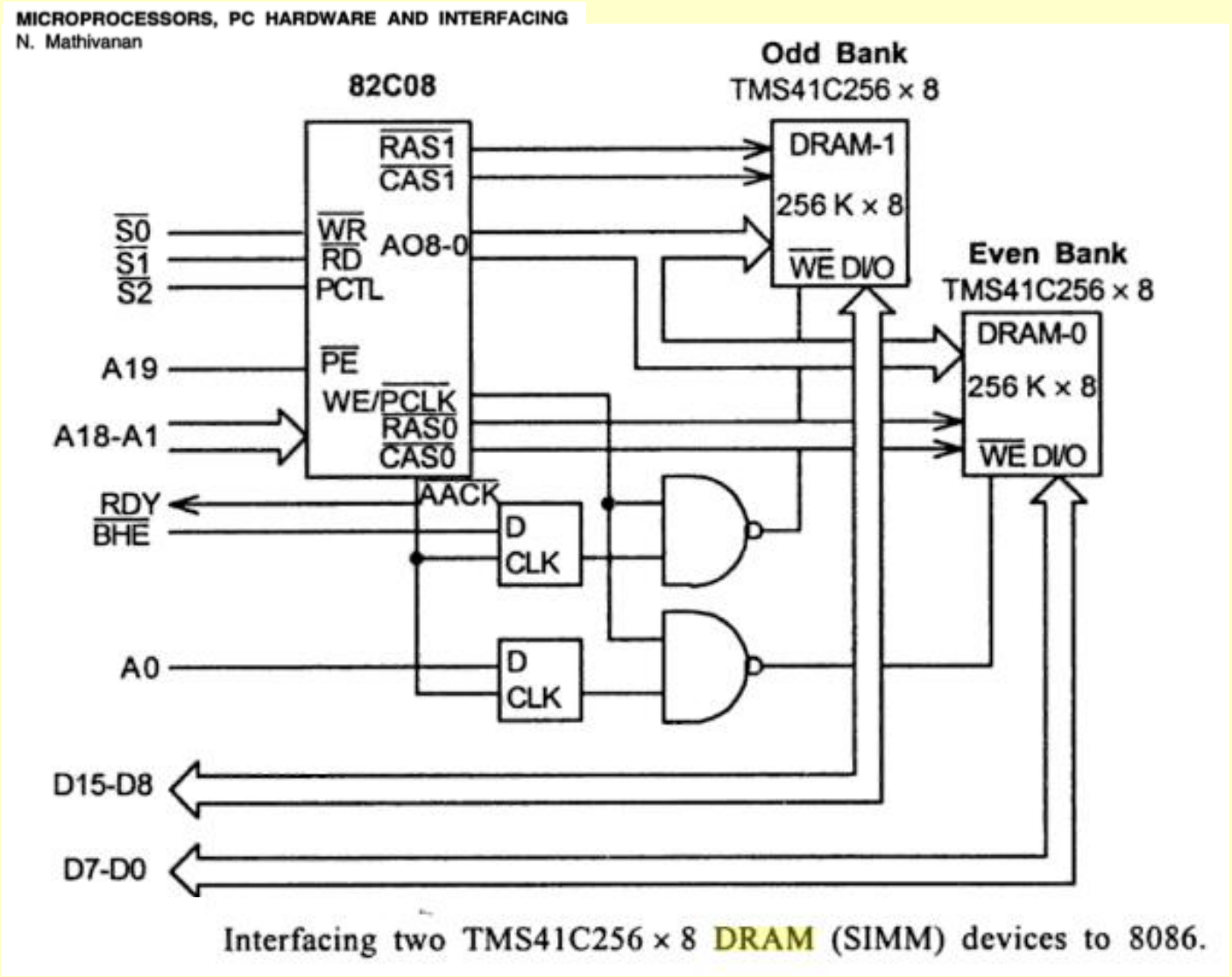
DRAM Refresh

Fiecare celulă trebuie actualizată la fiecare câteva ms pentru a evita pierderea datelor. Ori de câte ori este citit un rând, amplificatoarele de sensing rescriu automat întregul rând, deci trebuie să accesăm fiecare rând o singură dată în intervalul de reîmprospătare.

- Reimprospatarea se face rând cu rând (nu în rafală) pentru a evita accesarea DRAM pentru o perioadă lungă de timp
- De obicei este realizat de hardware, folosind contor (pentru a urmări indexul rândului următor) și un temporizator (pentru a iniția o reîmprospătare)

- Ex: Hitachi 64Mbit DRAM necesită 8192 reîmprospătări la fiecare 64 ms. Fiecare rând este accesat la (64 ms / 8192) = 7,8 us.
- Este nevoie a se furniza adresa de coloana si : “RAS-only refresh”

- Se poate insera un ciclu refresh la sfarsitul unui acces de citire separat; dacă CAS nu este activ, datele citite rămân valabile: *„hidden refresh”*
- Unele DRAM-uri au contor intern; sistemul trebuie sa indice numai cand sa se faca urmatoarea improspatare a randului prin activarea CAS apoi RAS (opus accesului regulat): “CAS-before-RAS (CBR) refresh”



Interfacing two TMS41C256 × 8 DRAM (SIMM) devices to 8086.

DRAM controller-ul are 3 sarcini de indeplinit:

- Sa multiplexeze adresele de rand si coloana
- Sa translateze (converteasca) semnalele de control sistem $\overline{MEMW}$ si $\overline{MEMR}$ in semnalele de control compatibile cu DRAM: $\overline{RAS}$, $\overline{CAS}$ si $\overline{WE}$
- Sa reimprospateze DRAM-ul

DRAM Interfacing

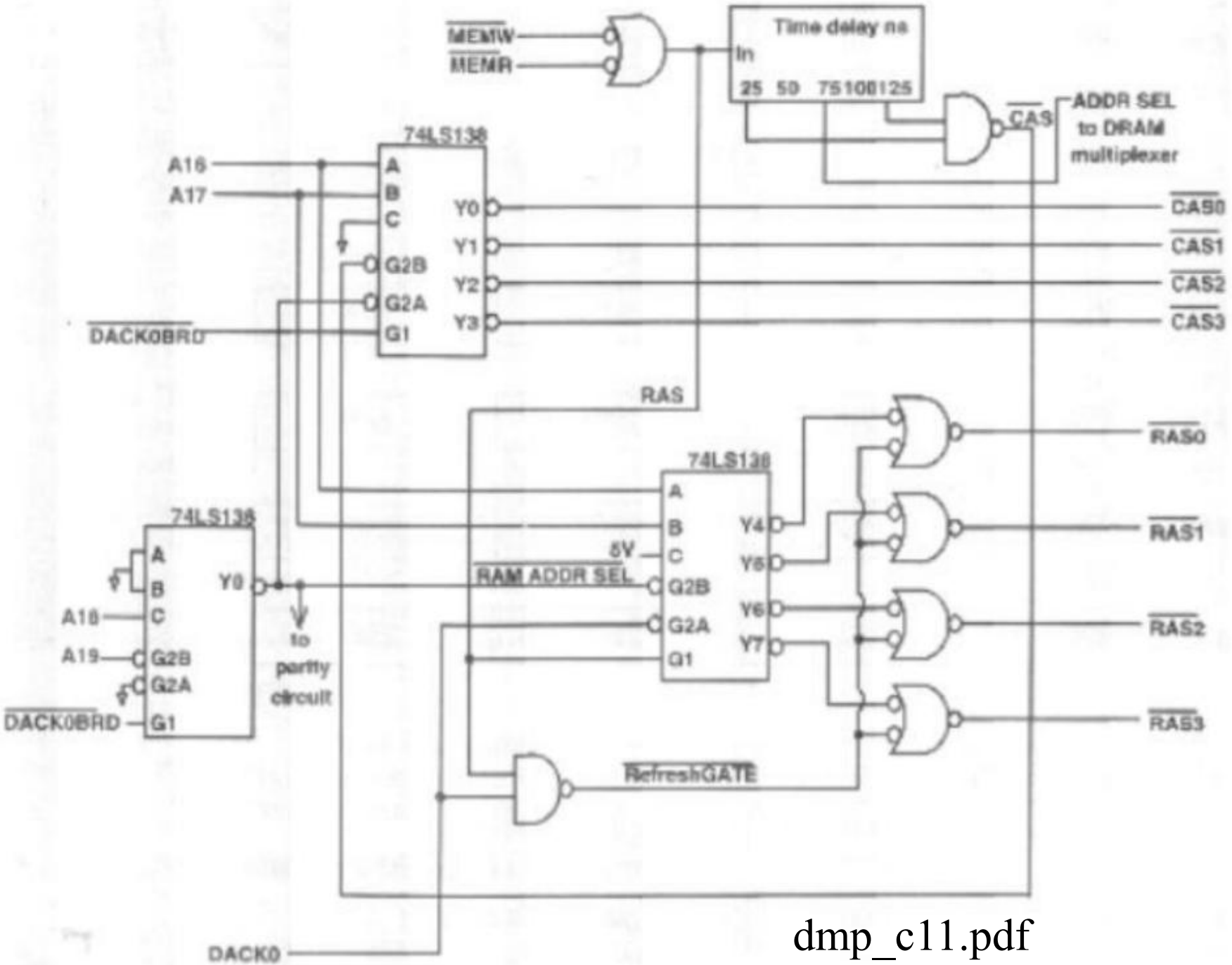
As mentioned earlier, DRAM must be refreshed periodically. DRAM interfacing requires a DRAM controller to read, write and refresh the DRAM. The DRAM controller performs three basic tasks, (i) multiplexes the row and column addresses, (ii) translates the system $\overline{\text{MEMR}}$ and $\overline{\text{MEMW}}$ control signals to $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, and $\overline{\text{WE}}$ control signals compatible for the DRAM, and (iii) keeps the DRAM refreshed.

Intel 82C08 is a CMOS DRAM controller, that can control two banks of 256 K × 16 DRAM chips. It contains a refresh counter, refresh timer, and address multiplexers to select the row, column, or refresh address. It also includes arbitration and control logic to coordinate refresh cycles with microprocessor accesses and provides the required control signals to directly drive the DRAM.

TMS41C256 is a 256 K × 1 DRAM. Eight such devices are connected in parallel to store binary information in bytes and a 256 K × 8 Single In-line Memory Module (SIMM) is constructed. Figure 3.19 shows two TMS41C256 × 8 SIMM devices interfaced to the 8086 microprocessor on odd and even memory banks using the 82C08 DRAM controller.

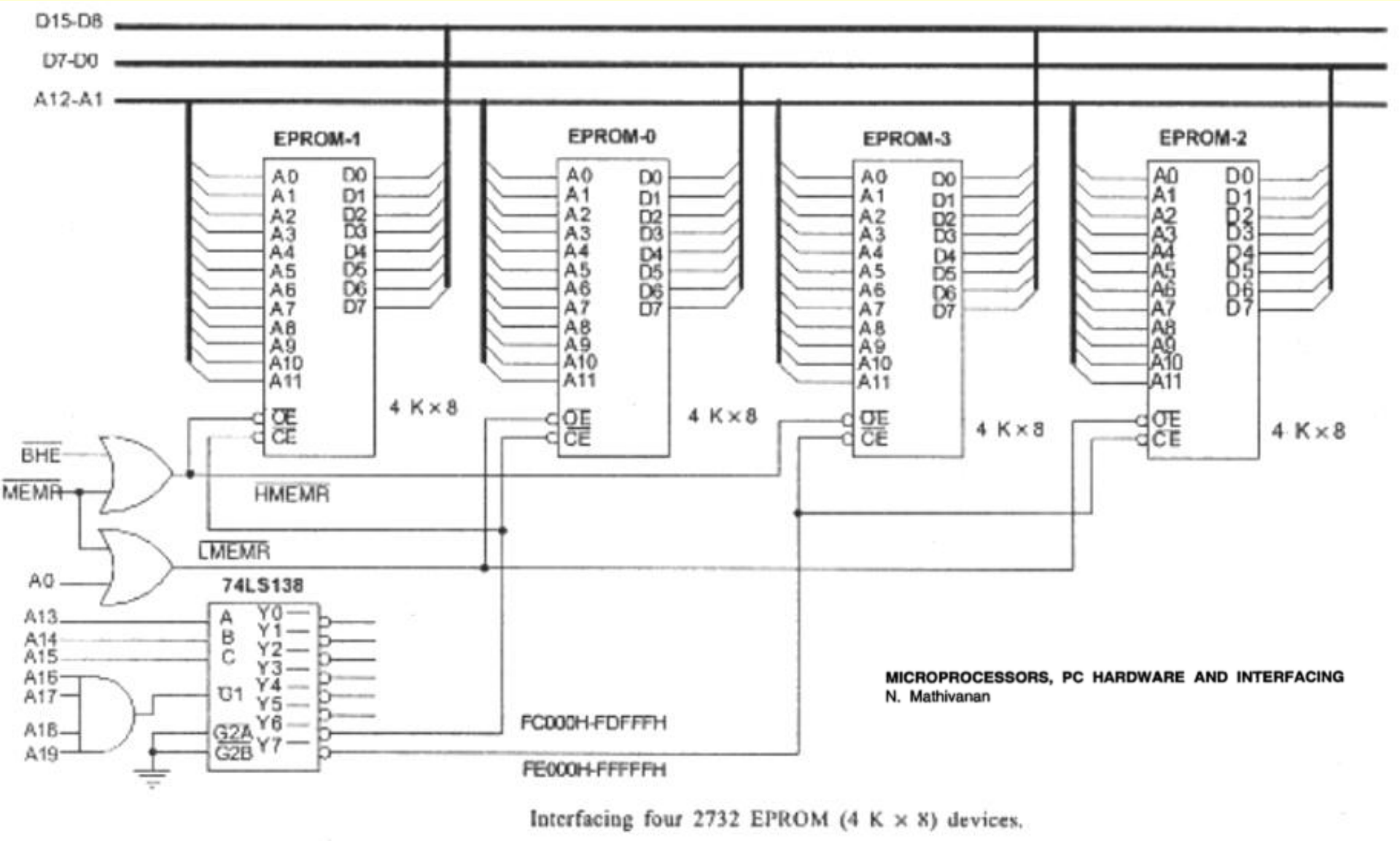
The 82C08 multiplexes the 18-bit address (A18-A1) onto the 9-bit address output lines to access the 256 K DRAM devices. The A19 bit is connected to the Port Enable $\overline{\text{PE}}$ input

of 82C08, which enables the controller whenever the microprocessor accesses memory locations in the address range 00000H–7FFFFH. The $\overline{\text{AACK}}$ signal clocks in the A0 and $\overline{\text{BHE}}$ signals into two latches and the outputs of the two latches are connected to the Write Enable $\overline{\text{WE}}$ inputs of DRAM-0 and DRAM-1. It assigns even addresses to the 256 K memory locations in DRAM-0 and odd addresses to the 256 K memory locations in DRAM-1. The 82C08 decodes the status signals S2-S0 and generates the read and write control signals required for the DRAM chips. If the microprocessor attempts to access the memory in the middle of a refresh cycle, the DRAM controller activates the $\overline{\text{AACK}}$ signal, which in turn forces the microprocessor to introduce wait states in the current cycle. During a read or write operation, the 82C08 multiplexes the address into DRAM chips and generates appropriate row and column strobe signals.


MEMW
MEMR
Time delay ns
In
25 50 75 100 125
CAS
ADDR SEL
to DRAM
multiplexer
74LS138
A16
A17
A
B
C
G2B
G2A
G1
Y0
Y1
Y2
Y3
CAS0
CAS1
CAS2
CAS3
DACK0BRD
RAS
74LS138
RAS0
RAS1
RAS2
RAS3
74LS138
A
B
C
G2B
G2A
G1
Y4
Y5
Y6
Y7
5V
RAM ADDR SEL
A16
A19
DACK0BRD
Y0
to
parity
circuit
RefreshGATE
DACK0

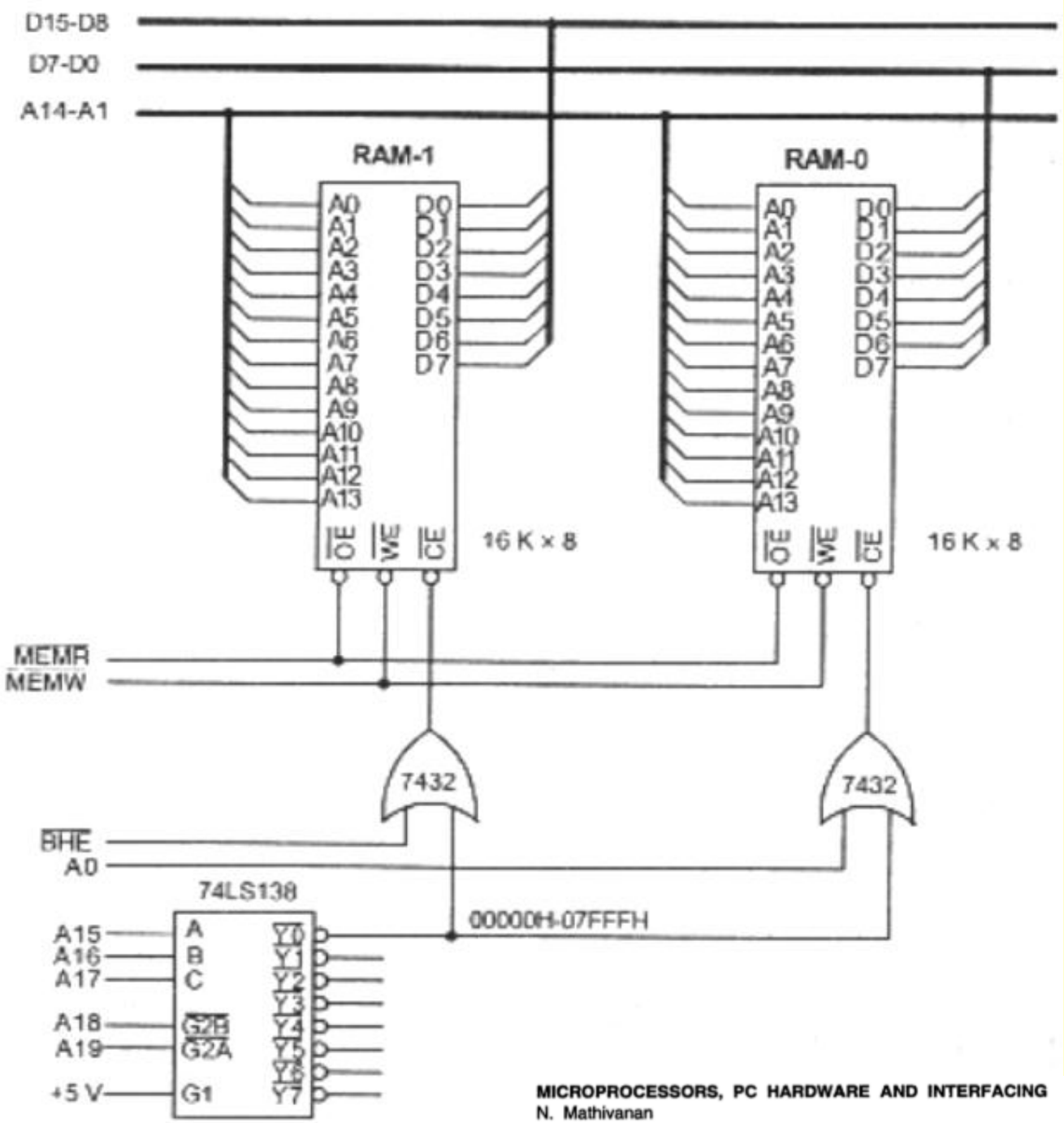
dmp_c11.pdf

Tema2. Studiati schemele de mai jos a modului de selectie a adreselor memoriilor EPROM si SRAM



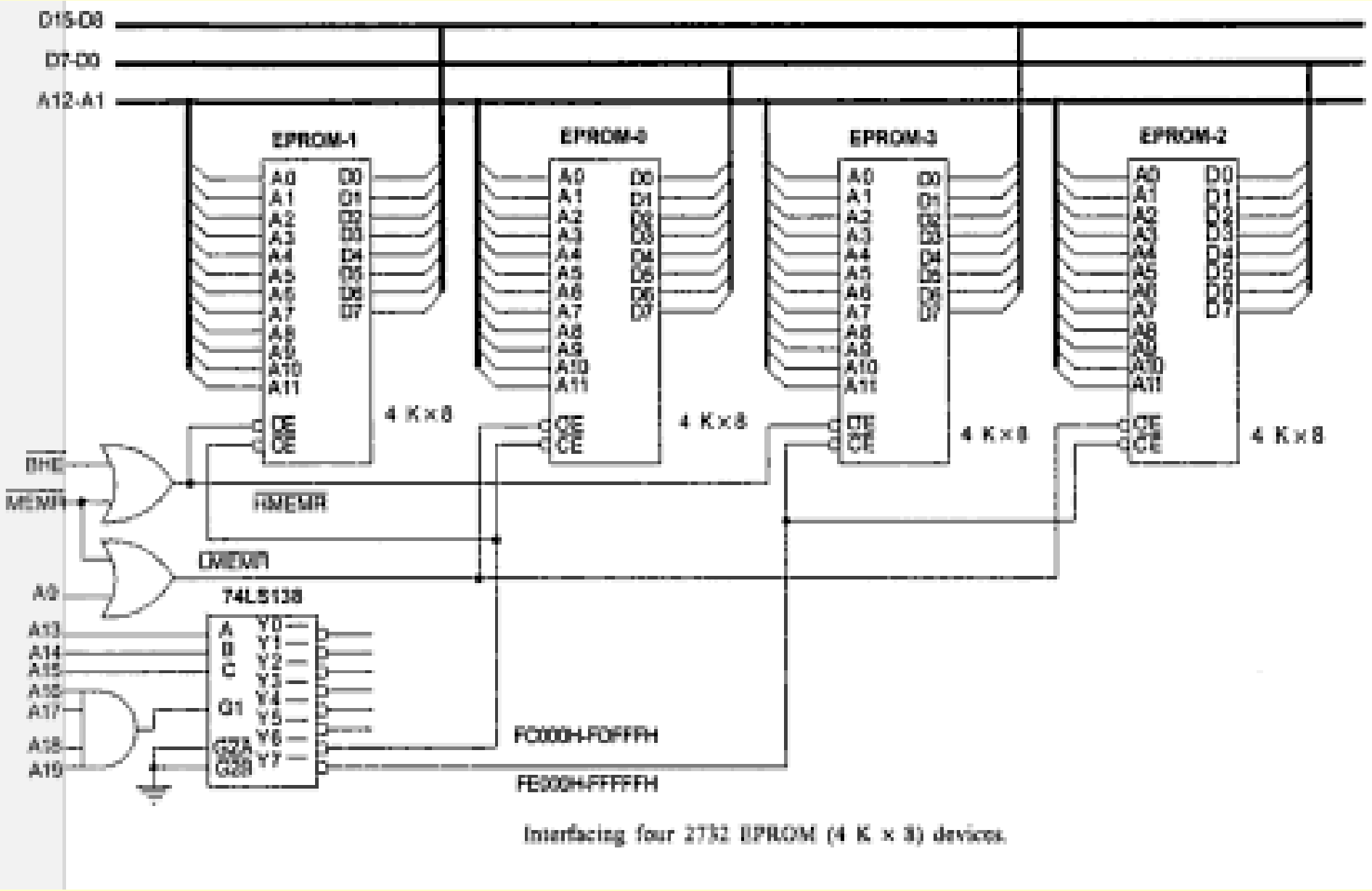
MICROPROCESSORS, PC HARDWARE AND INTERFACING
N. Mathivanan

Interfacing four 2732 EPROM (4 K × 8) devices.



Interfacing two 62128 RAM (16 K × 8) devices.

Tema 3. Studiati schema de conectarea a memoriei.



Interfacing four 2732 EPROM (4 K × 8) devices.

Tema 4. Cautati si studiati principalele tendinte in dezvoltarea memoriilor

DDR5: The Next-Generation DRAM Standard (2021+)

Overview

•DDR5 officially launched in 2021, replacing DDR4 in desktop/laptop platforms from Intel Alder Lake onwards

•JEDEC standard: JESD79-5 (2020). First DDR5 modules shipped at 4800 MT/s; roadmap extends beyond 12800 MT/s

•DDR5 doubles bandwidth per channel vs DDR4 while improving power efficiency (1.1V vs 1.2V DDR4)

Key Technical Improvements over DDR4

•On-Die ECC (ODECC): Each DRAM die corrects single-bit errors before data leaves the chip.

•Two 32-bit sub-channels per DIMM (vs one 64-bit DDR4) - improves memory controller efficiency

•Burst Length increased to BL16 (from BL8 in DDR4) for higher bandwidth per access

•Module capacity: up to 128 GB per DIMM (vs 32 GB typical DDR4) using 3D stacking

•On-DIMM PMIC (Power Management IC): voltage regulation moved from motherboard to DIMM itself

DDR4 vs DDR5 Comparison

•DDR4: 1600-3200 MT/s, 1.2V, 64-bit channel, max 32GB/DIMM, no on-die ECC

•DDR5: 4800-8800+ MT/s, 1.1V, 2x32-bit channels, up to 128GB/DIMM, on-die ECC

•Bandwidth: DDR5-6400 = ~51.2 GB/s per channel vs DDR4-3200 = ~25.6 GB/s

LPDDR5 / LPDDR5X: Mobile & Embedded Memory

What is LPDDR?

- Low Power DDR (LPDDR) is a specialized DRAM standard for mobile devices, laptops, and embedded systems
- Optimized for ultra-low power : features deep sleep modes, partial array self-refresh (PASR)
- Soldered directly on the PCB (PoP - Package on Package) rather than using removable DIMMs

LPDDR5 Key Features (JEDEC 2019)

- Speed: up to 6400 MT/s (LPDDR5X extends to 8533 MT/s), bandwidth ~68 GB/s
- Power: 1.05V-1.1V, ~50% lower power than LPDDR4 at same performance
- New features: Link ECC, WCK clock architecture, Data Copy command for in-memory operations
- Used in: flagship smartphones (Snapdragon 8 Gen 3, Apple A17/M-series), automotive ADAS, AI edge chips

LPDDR5X & Future: LPDDR6

- LPDDR5X (2022): pushes to 8533 MT/s - used in Samsung Galaxy S24, Google Pixel 9
- LPDDR6 (expected 2025+): targets 14 Gbps per pin, optimized for on-device AI/LLM inference
- On-device AI requirement: running 7B parameter LLMs needs ~56 GB/s sustained bandwidth

HBM (High Bandwidth Memory): Memory for AI & GPU

What is HBM?

- High Bandwidth Memory (HBM) stacks multiple DRAM dies vertically using Through-Silicon Vias (TSV)
- Connected to the processor via a silicon interposer in a 2.5D/3D package (no PCB trace delays)
- Standard governed by JEDEC/JEDEC HBM consortium (AMD, Intel, SK Hynix, Samsung, Micron)

HBM Generations

- HBM1 (2013): 128 GB/s bandwidth, 1024-bit interface, 2 or 4 dies - used in AMD Fury GPU
- HBM2 (2016): 256 GB/s, up to 8 dies = 8 GB per stack - NVIDIA V100 (Tesla), AMD Vega
- HBM2E (2018): 460 GB/s, 16 GB per stack - NVIDIA A100 (6912 CUDA cores), AMD MI250
- HBM3 (2022): 819 GB/s per stack, 24 GB per stack - NVIDIA H100 (80 GB with 5 stacks)
- HBM3E (2024): 1.2 TB/s per stack - NVIDIA H200, AMD MI300X (192 GB total)

Why HBM for AI?

- AI training is memory-bandwidth bordered: larger models require moving more weights per second
- NVIDIA H100: 3.35 TB/s total HBM3 bandwidth vs 77.6 GB/s DDR5 in a CPU system
- Memory capacity critical: GPT-4 requires ~700 GB for inference, requiring multi-GPU HBM

3D NAND Flash: Storage Technology Evolution

From Planar to 3D NAND

- Traditional 2D (planar) NAND hit physical limits ~16nm: interference between adjacent cells
- 3D NAND stacks cell layers vertically - Samsung V-NAND (2013) was the commercial pioneer
- 2024: leading manufacturers produce 200+ layer 3D NAND (Samsung 9th gen, Micron 232L, SK Hynix 238L)

Cell Types: TLC, QLC, PLC

- SLC (1 bit/cell): fastest, most reliable, expensive - enterprise applications only
- MLC (2 bit/cell): good balance of speed/density - high-end SSDs
- TLC (3 bit/cell): mainstream 2024 consumer SSDs - 1-4 TB drives common
- QLC (4 bit/cell): highest density, lower cost - 4-8 TB consumer drives, data centers
- PLC (5 bit/cell): under development by Intel/Micron for archival-density storage

NVMe Interface Revolution

- NVMe (Non-Volatile Memory Express) over PCIe 4.0/5.0 replaced SATA for performance SSDs
- PCIe 5.0 NVMe (2023): sequential reads up to 14 GB/s (vs 550 MB/s SATA SSD)
- NVMe over Fabrics (NVMeOF): extends NVMe protocol over Ethernet/InfiniBand for data centers

CXL (Compute Express Link): Memory Expansion Standard

What is CXL?

- CXL is an open interconnect standard based on PCIe physical layer for high-bandwidth, low-latency CPU-memory-accelerator communication
- CXL 1.1 (2019), CXL 2.0 (2020), CXL 3.0 (2022, 256 GB/s bidirectional)
- Supported by Intel (Sapphire Rapids+), AMD (Genoa/EPYC), ARM, NVIDIA, Samsung, SK Hynix, Micron

CXL Memory Protocols

- CXL.io: PCIe-compatible register and configuration access (I/O devices)
- CXL.cache: Allows accelerators (GPU, FPGA) to cache host CPU memory with coherence.
- CXL.mem: Expands host memory with attached memory devices (CXL DIMMs, memory-semantic SSDs)

Use Cases & Industry Impact

- Memory pooling: Multiple servers share a large pool of DDR5/HBM via CXL fabric
- Memory tiring: Hot data in DDR5, warm data in CXL-attached DDR4, cold data in NVMe SSD
- Replaces Intel Optane: CXL-attached DRAM/flash provides byte-addressable persistent memory without proprietary 3D XPoint
- AI workloads: Large language models (LLMs) with 100B+ parameters benefit from CXL memory expansion beyond local DRAM

Processing-in-Memory (PIM) / Near-Memory Computing

The Memory Wall Problem

- Moving data between CPU/GPU and DRAM consumes 40-60% of total system energy
- *Memory bandwidth* is the bottleneck for AI inference, genomics, graph processing
- Solution: bring computer closer to data - Processing-in-Memory (PIM) or Near-Data Processing (NDP)

Commercial PIM Implementations (2021-2024)

- Samsung HBM-PIM (Aquabolt-XL, 2021): 16 GFLOPS PIM engine inside HBM2E stack, 2.7x energy saving for AI inference
- SK Hynix AiM (Accelerator-in-Memory, 2022): GDDR6 with embedded SIMD computing for ML at 1 TB/s effective bandwidth
- UPMEM PIM (2021): Commercial DRAM DIMMs with 8 DPU cores per chip, 20 DIMMs = 2560 cores for DNA sequencing/database
- Samsung HBM3 PIM (2024): Integrated MAC units for LLM inference, reduces host GPU memory transfers by 60%

PIM Programming & Challenges

- Programming models: OpenCL extensions, vendor SDKs (Samsung PIM SDK, UPMEM DPU SDK)
- Challenges: limited computing precision, programming complexity, limited ISA, thermal constraints
- Near-Memory vs In-Memory: Near-Memory (computing in logic die near DRAM) vs In-Memory (computing in DRAM cell array itself)

MRAM: Magnetic RAM - Technology & Status (2024)

How MRAM Works

- Stores data as magnetic orientation of a thin ferromagnetic layer (not charge like DRAM/Flash)
- MTJ (Magnetic Tunnel Junction): two magnetic layers separated by thin insulator - resistance changes with magnetic alignment
- Read: sense current through MTJ; Write: STT (Spin-Transfer Torque) - spin-polarized current switches magnetization

Advantages

- Non-volatile: retains data without power (like Flash), but with DRAM-like speed
- Endurance: >10^15 write cycles (vs ~10^5 for Flash), unlimited like DRAM
- Speed: sub-ns write speed, suitable as L3/L4 cache replacement
- No wear-out: data retention >20 years at operating temperature

Current Status & Products (2024)

- Ever spin: ships 1Gb and 4Gb STT-MRAM for industrial applications (controllers, IoT)
- TSMC, GlobalFoundries: offer embedded MRAM (eMRAM) as NVM option in advanced CMOS nodes (28nm, 22nm, 12nm)
- Samsung, Intel, IBM: eMRAM replacing embedded Flash in microcontrollers and IoT chips below 28nm
- Limitation: cost/bit still 10-100x higher than NAND Flash; density limited vs DRAM for standalone applications

Emerging Non-Volatile Memory (NVM) Technologies

Phase-Change Memory (PCM / 3D XPoint heritage)

- PCM uses chalcogenide glass (GST) - crystalline (low resistance) vs amorphous (high) states
- Originally the basis for Intel Optane (3D XPoint). After Optane discontinuation (2023), PCM research continues in academia.
- Key specs: ~200ns read latency, ~1us write, 10^8 endurance - fills DRAM/Flash gap

ReRAM / Memristor (Resistive RAM)

- ReRAM stores data as resistance of a metal oxide film (forming/breaking conductive filament)
- Panasonic ships embedded ReRAM (eReRAM) for IoT MCUs; TSMC offers at 40nm, 22nm nodes
- Crossbar 3D ReRAM: claim 1TB/cm3 density for storage-class memory applications
- Weebit Nano, Adesto (Dialog): commercializing ReRAM at 22nm-40nm for NVM-MCU market

FeRAM (Ferroelectric RAM)

- FeRAM uses polarization state of ferroelectric material (PZT or HfO2) for non-volatile storage
- Fastest NVM: sub-ns switching, ultra-low energy write (unlike NAND requiring 15V Vpgm)
- Texas Instruments: ships FeRAM MCUs (MSP430FR series) for ultra-low-power data loggers
- HfO2-based FeRAM (FeFET): integrated into 28nm CMOS by GlobalFoundries for neuromorphic computing

Memory for AI / Machine Learning Workloads

Memory Requirements of Modern AI

- LLM inference (GPT-4 level, ~1.7T parameters): requires ~1.7 TB of memory for FP16 weights
- Training workload is compute-bound; inference is memory-bandwidth bound
- Rule of thumb: ~2 bytes per parameter for FP16 model storage + ~3x for activations/KV-cache
- KV-cache for attention mechanisms can exceed model weight size for long-context inference (128K tokens)

Memory Technology Choices for AI

- Training clusters: HBM3/HBM3E (NVIDIA H100/H200, AMD MI300X) - 3.35 TB/s bandwidth
- Edge AI inference: LPDDR5X (phone SoCs), on-chip SRAM (Apple Neural Engine, TPU), ReRAM-based CIM
- Compute-in-Memory (CIM): weights stored and multiplied in analog SRAM/ReRAM arrays, 100x energy reduction potential
- Model quantization: INT8/INT4/FP8 reduces memory 2-8x - NVIDIA H100 supports FP8 natively

Memory Scaling Challenges for AI

- Scaling law: doubling model size requires doubling memory AND doubling interconnect bandwidth
- NVLink/NVSwitch (NVIDIA): 900 GB/s GPU-GPU bandwidth for multi-GPU training
- Memory capacity ceiling: H200 = 141 GB HBM3E; next: HBM4 (2025) targeting 2 TB/s per stack

Future Memory Technology Roadmap (2024-2030)

Near-Term (2024-2026)

- DDR5-8800+ and LPDDR6 (2025+): speed scaling continues, on-die ECC mandatory for AI reliability
- HBM4 (2025): 2 TB/s per stack, embedded computing (PIM), 12-16 dies per stack
- 3D NAND 300+ layers: QLC mainstream consumer, PLC for archival cold storage

Medium-term (2026-2028)

- Monolithic 3D DRAM: DRAM cells stacked over logic using wafer-bonding (Samsung, SK Hynix)
- Analog Compute-in-Memory (CIM): SRAM/ReRAM arrays for DNN inference with ~10 TOPS/W
- CXL 3.0 fabric: pooled memory across racks, memory disaggregation in data centers

Long-Term (2028+): Research Horizons

- DNA data storage: 1 exabyte/gram density - Microsoft and Catalog exploring automated write/read systems
- Neuromorphic memory: memristor crossbar arrays for brain-inspired in-memory AI computing
- The memory wall remains the central challenge: energy and bandwidth, not transistor count, limit AI scaling