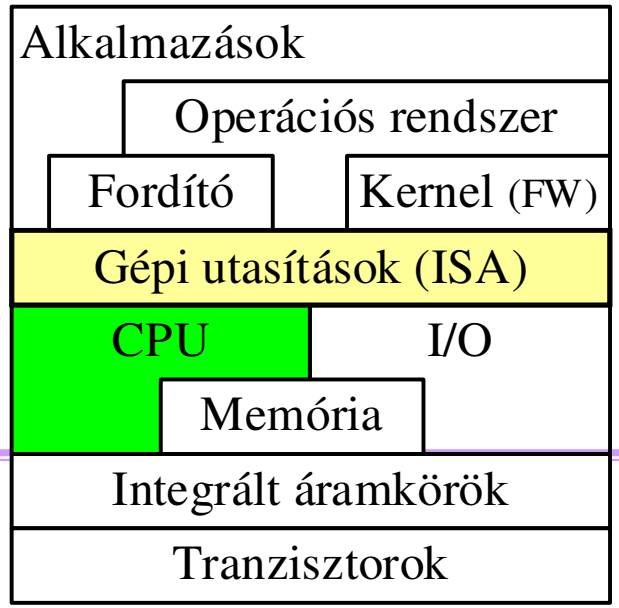
INFORMATIKA I.

BMEVIIIAB08

Számítógép architektúrák

ESETTANULMÁNY IA32 architektúra



❖ Paraméterek

- ❑ Az IA-32 ISA -t megvalósító mikroarchitektúra
- ❑ 32 bites mikroprocesszor
- ❑ 4 Gigabyte fizikai címtartomány
- ❑ 64 Terabyte virtuális címtartomány
- ❑ Beépített lapszervezésű MMU
- ❑ Külső /486-nál belső/ matematikai társprocesszor /X87/
- ❑ Külső /486-nál belső/ gyorsító tár
- ❑ 386 33Mhz
- ❑ 486DX4 /33/100Mhz órajelszorzó/
- ❑ 3-25 MIPS
- ❑ Felülről kompatibilis a 8086, 80286 mikroprocesszorokkal

➢ Üzemmódok, kompatibilitás

Típus	Adat	Cím	Real Üzemmód *Címtartomány*	Védett üzemmód *Címtartomány*
8086 /1978/	16 bit	20	Maximum mód *1MB*	X
80286 /1982/	16 bit	20/24	Valós /real/ mód *1MB*	Többszintű védelem Védett mód /protected/ 16MB
80386/ /1985/ 80486 /1989/	16/32	20/32	Valós /real/ mód *1MB*	*Többszintű védelem* Virtuális tárkezelés *4GB* Virtuális VM86 mód *16 adat bit* *1MB* /20bit/

❖ Valós üzemmód

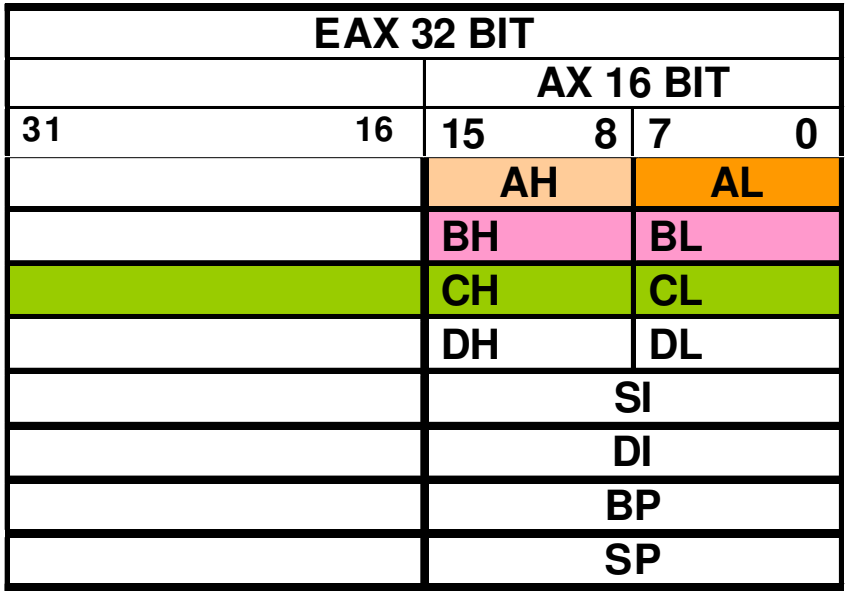
- ➢ Címzés valós módban
- ➢ Fizikai címtér *=[(Szegmensregiszter x 16)+EA]*

Szegmens regiszter	S_{15} S_0 SSSSSSSSSSSSSSSS	0000
Effektív cím	0000	EA_{15} EA_0 EEEEEEEEEEEEEEEE
Fizikai cím	F_{19} F_0 FFFF FFFFFFFFFFFFF	

- ❑ Szegmens regiszter 16 bit
- ❑ Szegmens méret max 64KB
- ❑ EA címzési mód szerint kiszámított cím pl:IP,SP,BP,stb.
- ❑ Bázisregiszteres /címbővítés/ és szegmentálás

❖ Felhasználói Regiszterkép

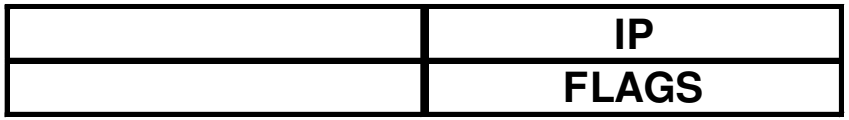
GENERAL REGISTERS



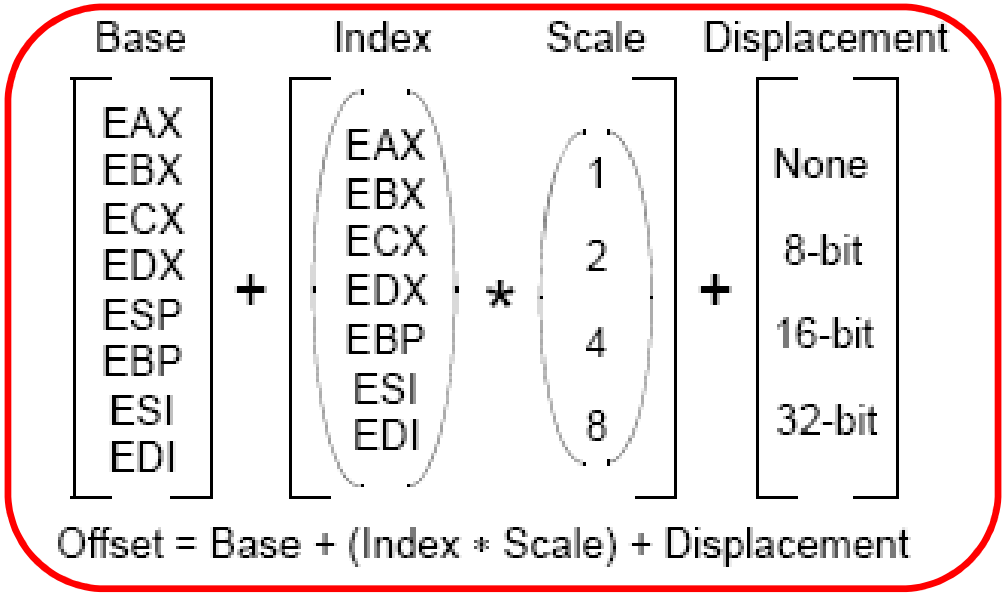
AX	EAX	EAX akkumulátor
BX	EBX	EBX bázisregiszter /pointer/ /DS/
CX	ECX	ECX számláló sztringhez és hurokhoz
DX	EDX	EDX általános I/O cím
SI	ESI	ESI forrás mutató /DS/
DI	EDI	EDI cél mutató /ES/
BP	EBP	EBP bázisregiszter /SS/
SP	ESP	ESP stack pointer /SS/

SZEGMES SZELEKTOR REGISZTEREK

CS	Kód
SS	Stack
DS	Adat
ES	Adat
FS	Adat
GS	Adat



IP	EIP	Utasítás számláló
EFLAGS		Flag regiszter



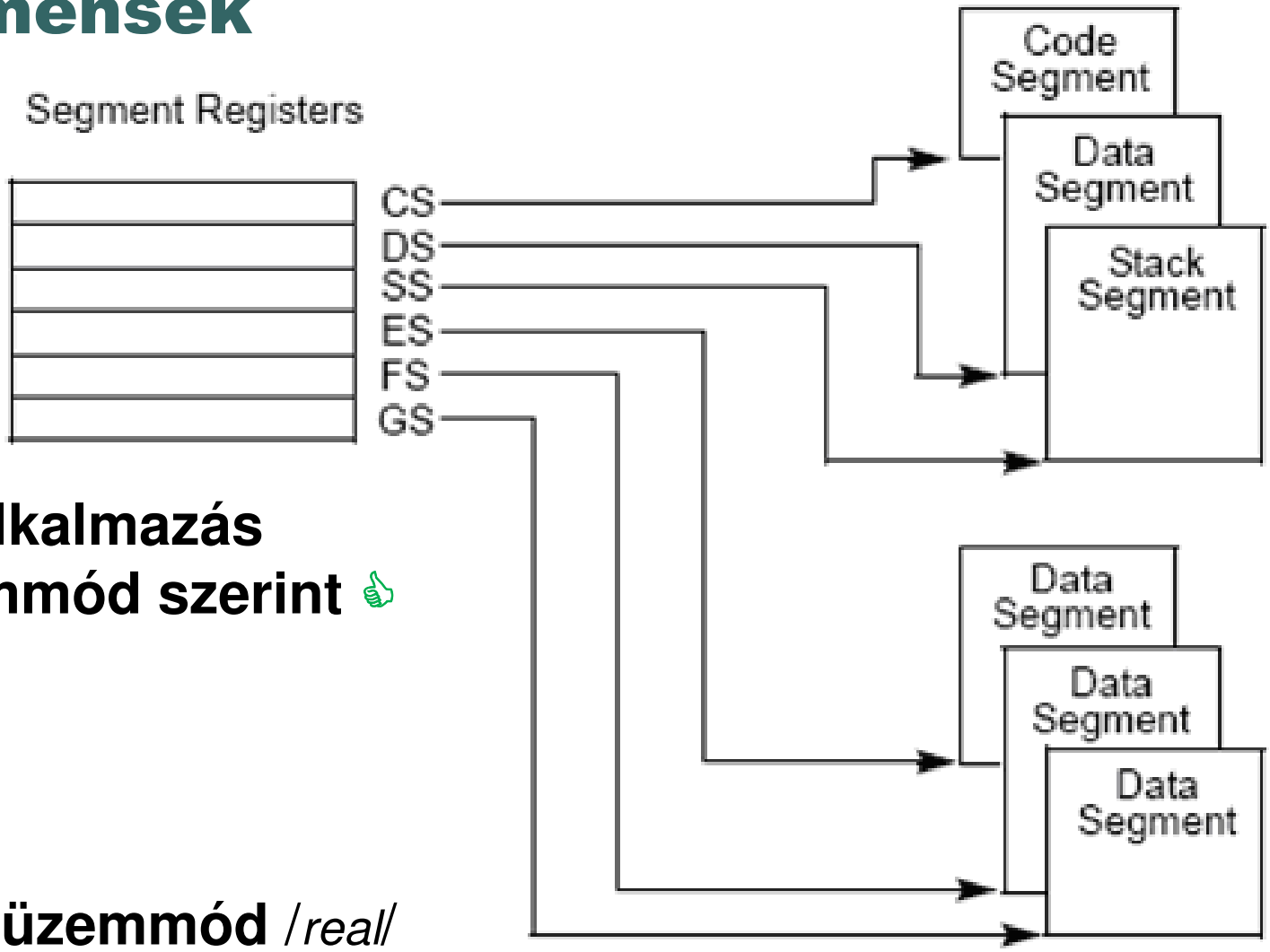
Effektív címszámítás általános modellje

Bázis és index regiszter használata

	16-Bit Addressing	32-Bit Addressing
BASE REGISTER	BX,BP	Any 32-bit GP Register
INDEX REGISTER	SI,DI	Any 32-bit GP Register Except ESP
SCALE FACTOR	none	1, 2, 4, 8
DISPLACEMENT	0, 8, 16 bits	0, 8, 32 bits

***16*, illetve *32* bites címzésnél**

❖ Szegmensek



➢ **Eltérő alkalmazás az üzemmód szerint**

❑ **Valós üzemmód** /*real*/

❑ **Védett üzemmód** /protected/

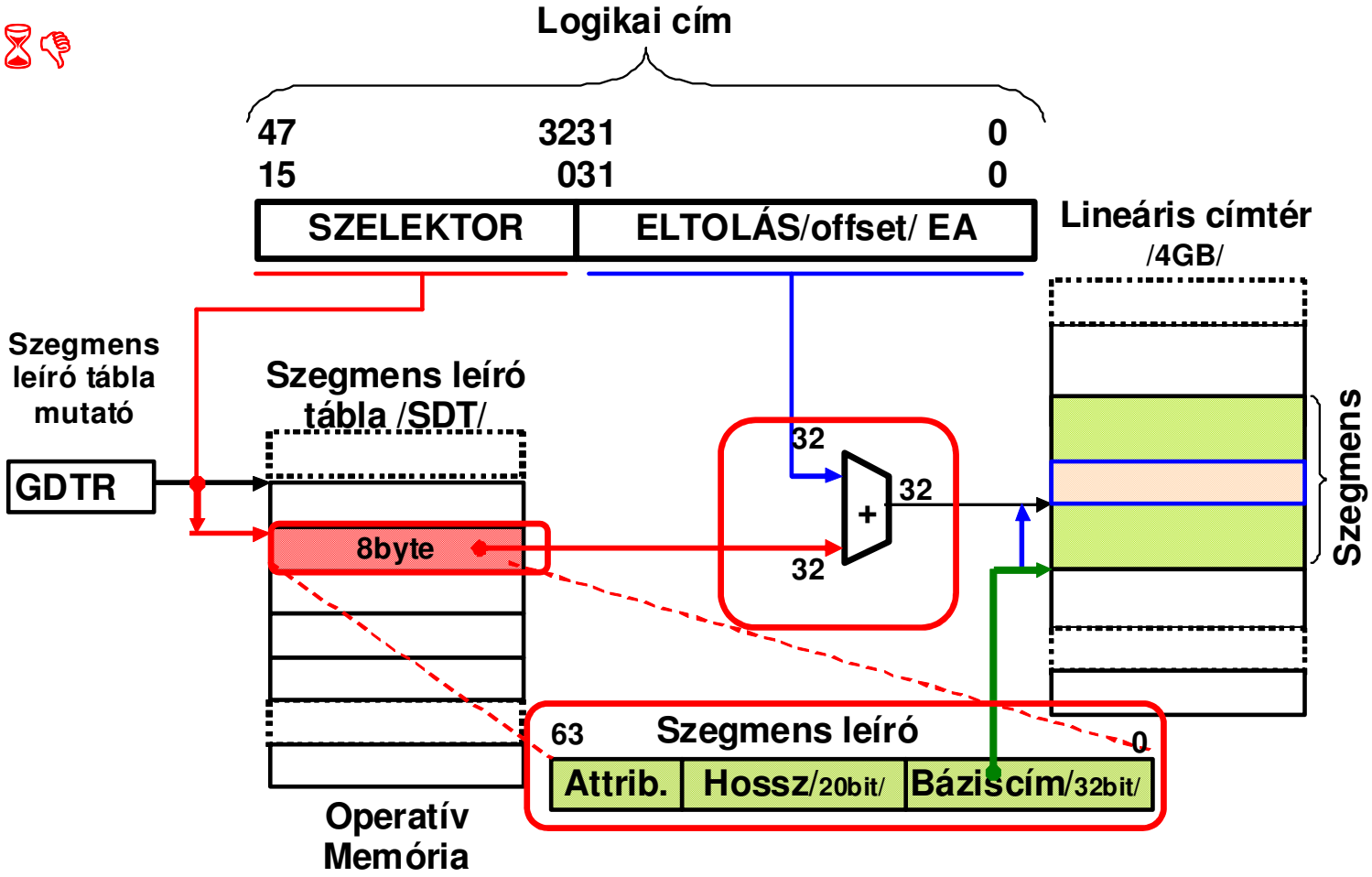
❖ Szegmensek

➢ Címzés védett üzemmódban *Logikai ⇔ Lineáris címtér*

❑ Logikai cím: *16 bit szelektor*, *32 bit eltolás* /*offset*/ *EA*

❑ 32 bit lineáris címtér, *1byte-4Gbyte* címtartomány

❑ ⌛



Privilégium szint
/CPL, DPL, RPL/

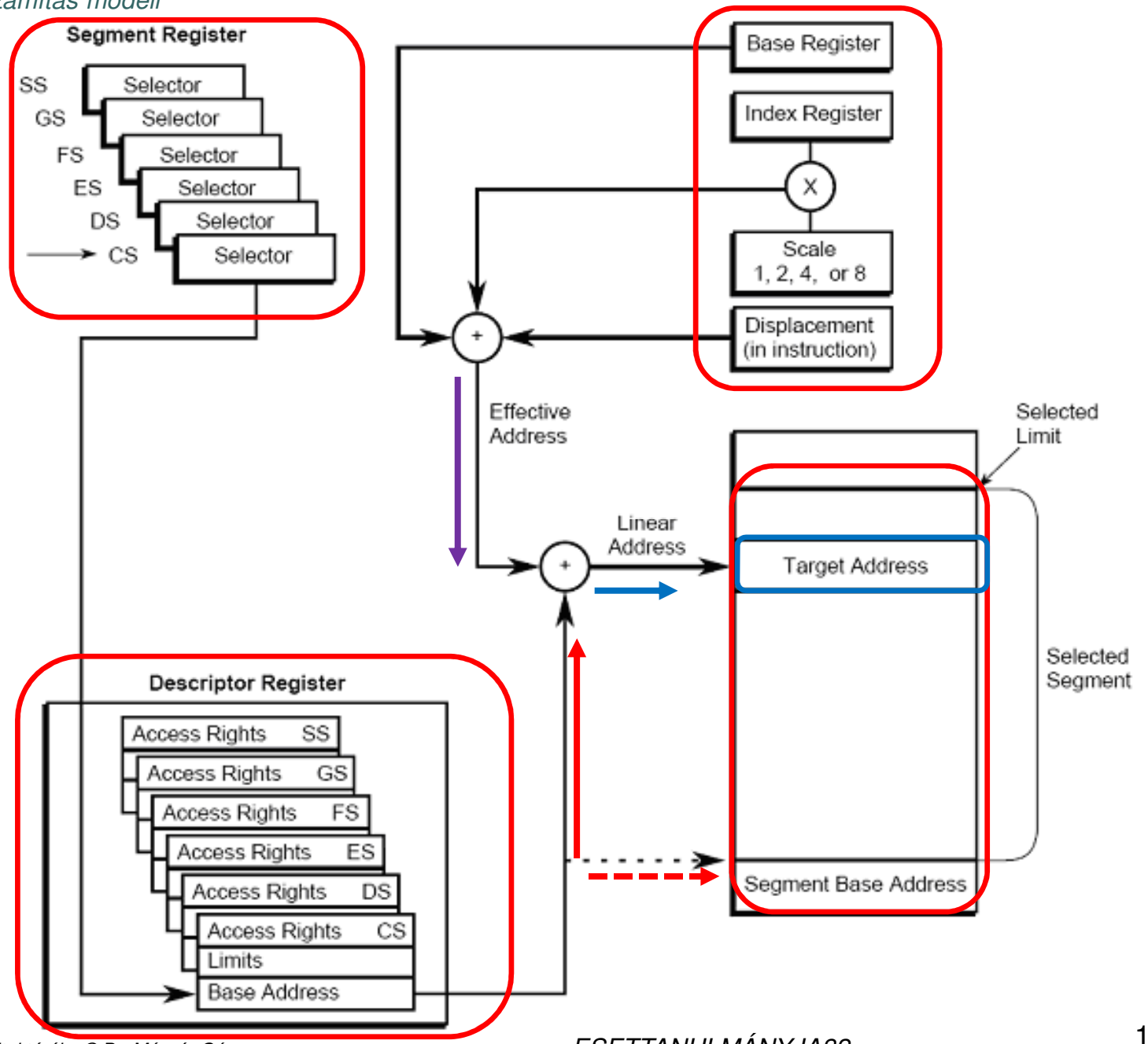
TI tábla indikátor
Globális leíró tábla /TI=0/
Lokális leíró tábla /TI=1/
$2x2^{13}$szegmens /max 4GB

- **64 Tera byte virtuális címtér**
- **→Gyorsítás 64 bit /8byte/ szegmensleíró kiolvasás→ tárolás árnyékregiszterben**

Szegmensregiszterek 15 … 0	Háttér /árnyék/ regiszterek 63 … 0
CS	BÁZIS, Limit, attributumok /CS/
SS	BÁZIS, Limit, attributumok /SS/
DS	BÁZIS, Limit, attributumok /DS/
ES	BÁZIS, Limit, attributumok /ES/
FS	BÁZIS, Limit, attributumok /FS/
GS	BÁZIS, Limit, attributumok /GS/

Programozottan nem elérhetők /HW kezeli/
új szegmensregiszter érték betöltésekor automatikusan
leíró kiolvasás, töltés, második memóriaciklusban hozzáférés

❖ *Lineáris címszámítás modell*



❖ Virtuális tárkezelés

Lineáris címtér

❑Logikailag strukturált

❑Strukturálatlan

Max 4GB

Változó fizikai tároló méret →max4GB

Virtuális tárkezelés/LVTK/

❑Lapszervezés

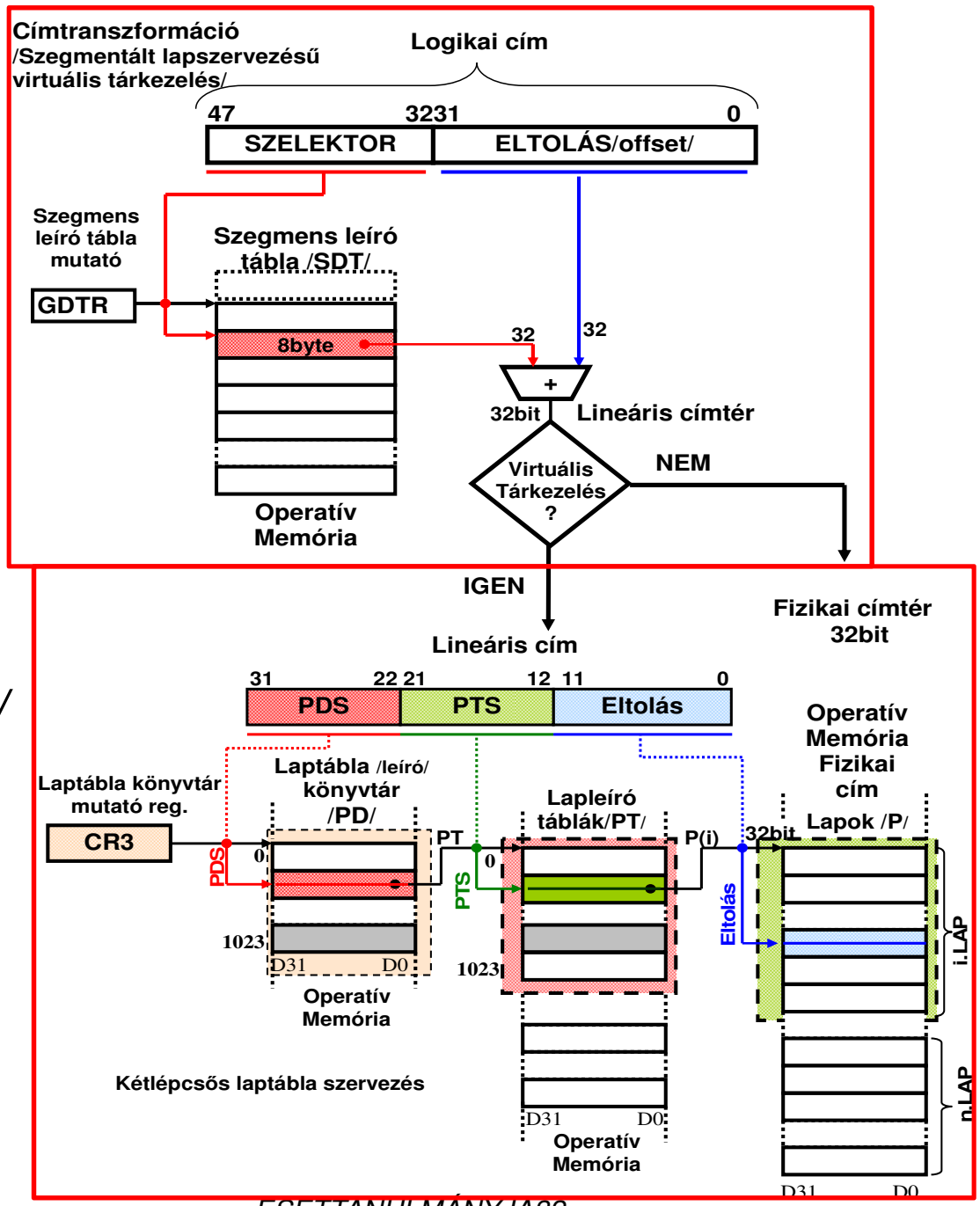
❑Ki/Be kapcsolható!!!

❑HW MMU

❑4kB-os lapok

❑kétszintű lapkezelés

Változó fizikai tároló méret →max4GB,..



ESETTANULMÁNY IA32

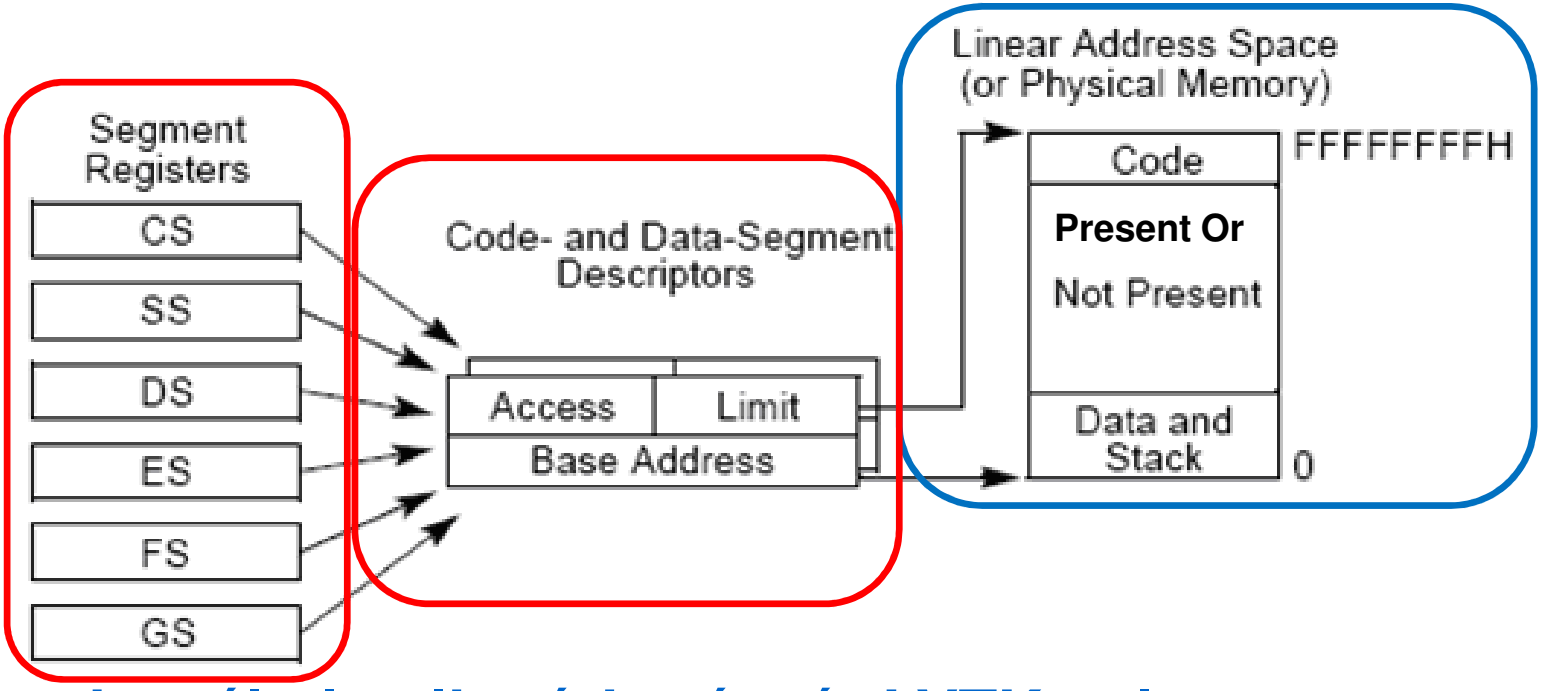
❖ i386 címképzési lehetőségei védett üzemmódban

❑ *Az OR határozza meg, hogy melyiket alkalmazza*

♦ **Strukturálatlan lineáris címtér LVTK nélkül**

■ *Lapszervezés kikapcsolva*

A szegmensek fedik egymást, minden szegmens
0 kezdőcím /báziscím/ 4GB hossz → Flat model

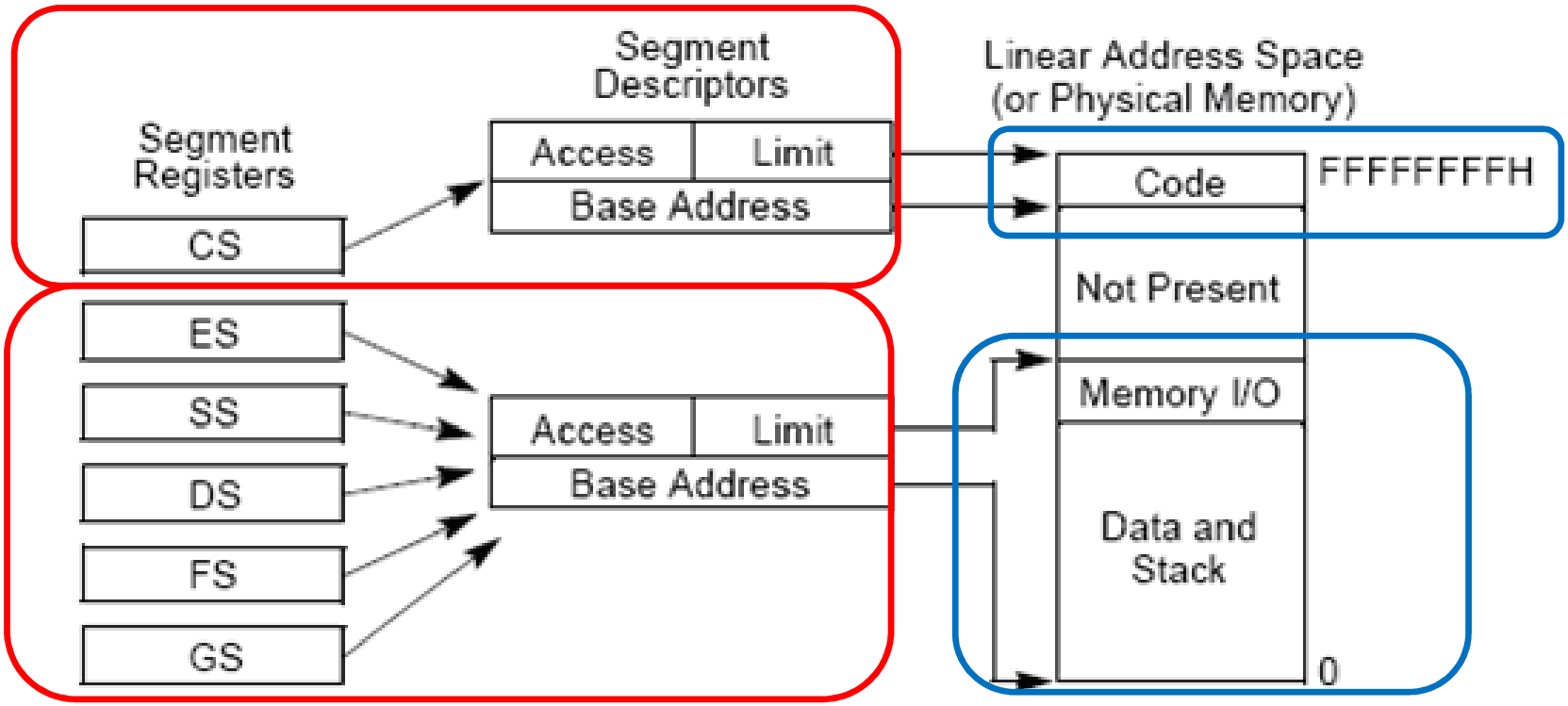


♦ **Strukturálatlan lineáris címtér LVTK-vel**

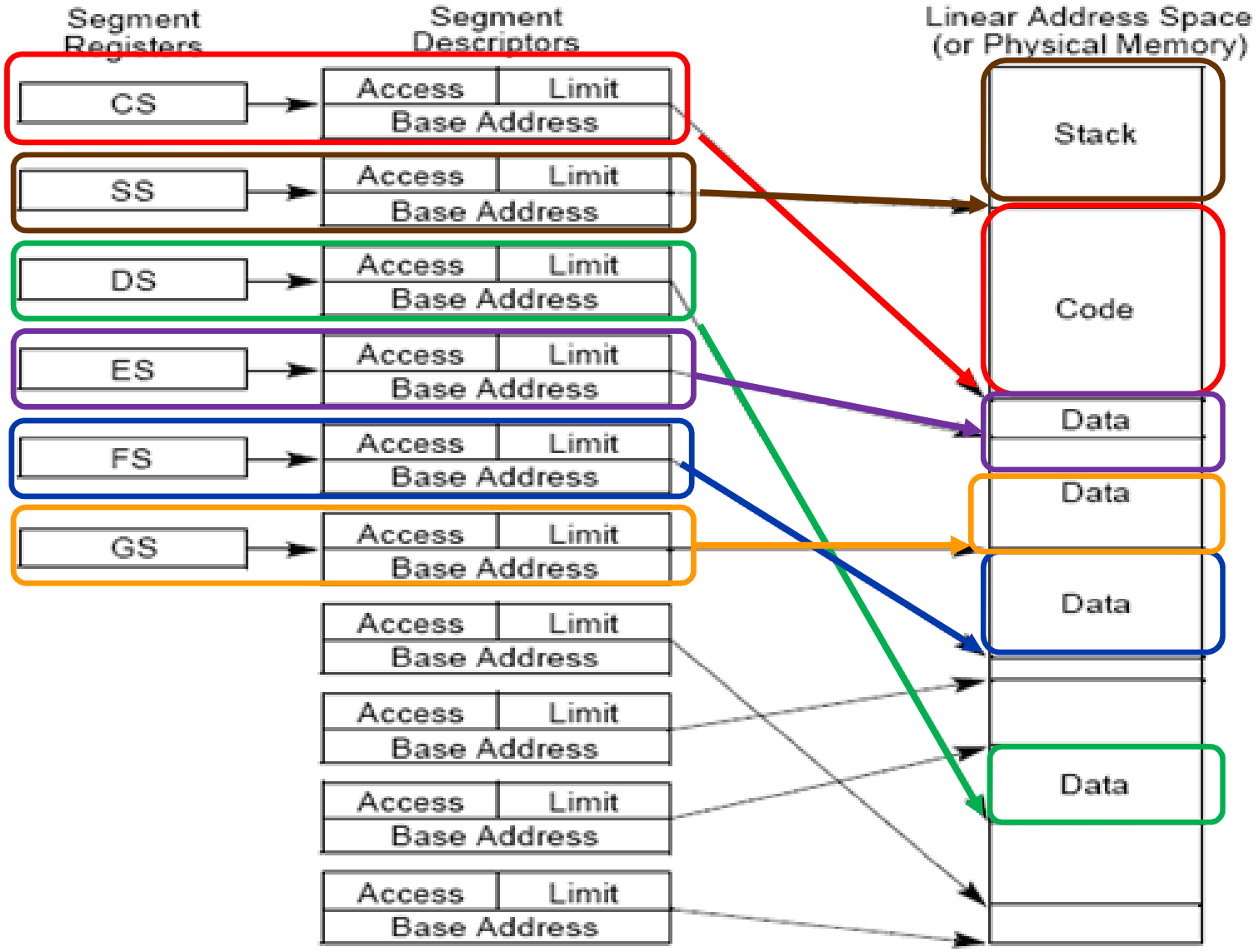
■ *Lapszervezés bekapcsolva*

♦ Részben strukturált lineáris címtér LVTK nélkül

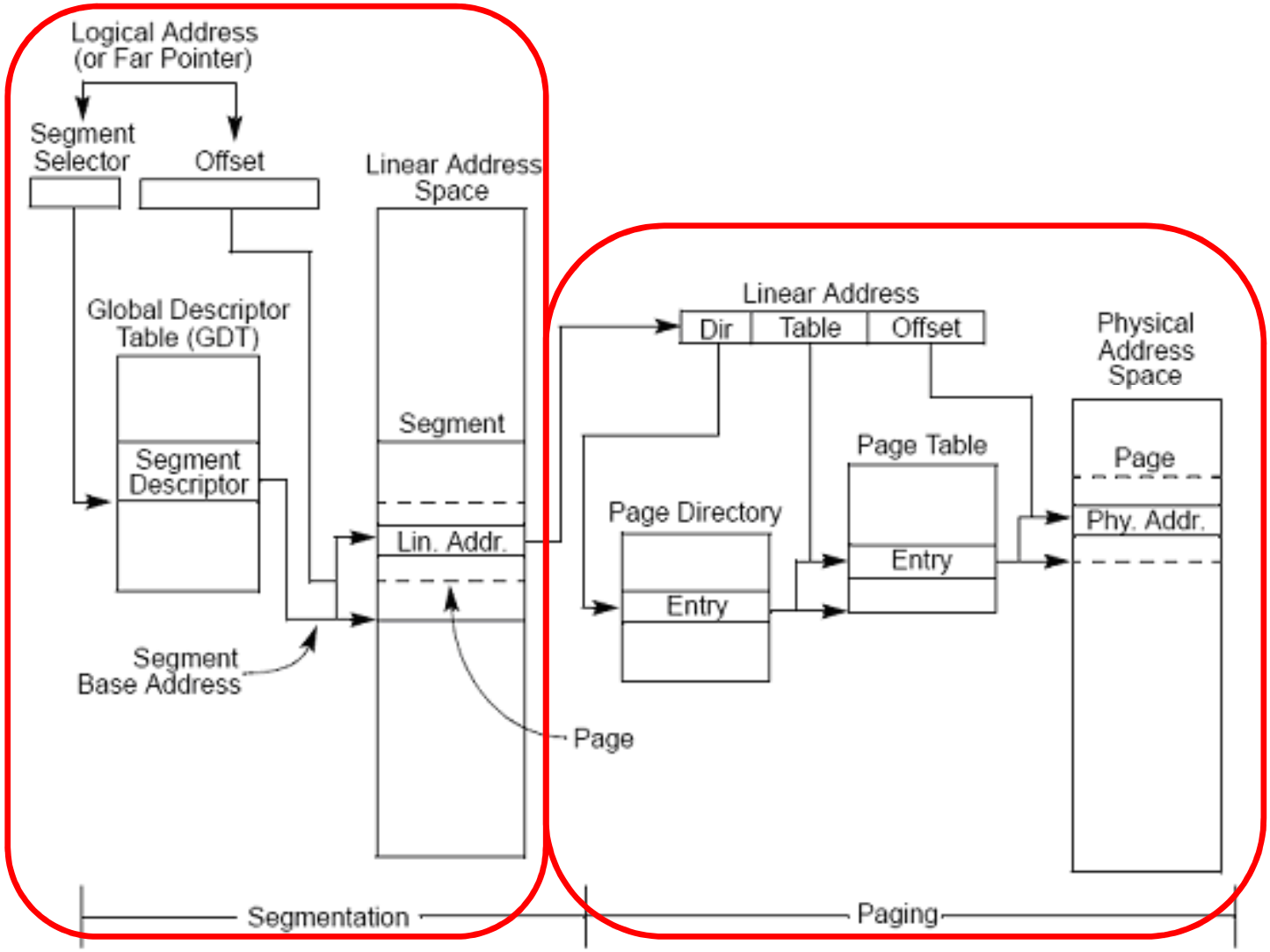
- ***Nincs lapszervezésű tárkezelés***
- ***Fizikailag osztott lineáris címtér***
- ***Csak a kiépített memória mérete szerint szegmentál***
  →Protected flat model



♦ Strukturált lineáris címtér /*Multi-segment model*/



♦ Strukturált lineáris címtér VTK-vel (*Szegmentált lapszervezés)* Lsd. 11



❖ Rendszerregiszterek

➢ *Csak privilegizált utasítások kezelhetik* 👍👎

❑ **GDTR** *globális szegmensleíró tábla báziscíme, hossza, attr.*

❑ **IDTR** *megszakításhoz rendelt leíró tábla b.cím, hossz, attr.*

❑ **TR** *éppen futó taszk TSS-ének szelektorát tartalmazza* */+háttérregiszterét/*

❑ **LDTR** *a taszk lokális leírótáblájának szelektorát tartalmazza*

♦ ***Szelektor a TSS-ben, LDT leírója GDT-ben***
/betöltés után a háttérregiszterben/

Rendszer címregiszterek

	47 … 16	15 … 0
GDTR	32 bit lineáris Báziscím	16 bit Limit /hossz/
IDTR	32 bit lineáris Báziscím	16 bit Limit /hossz/

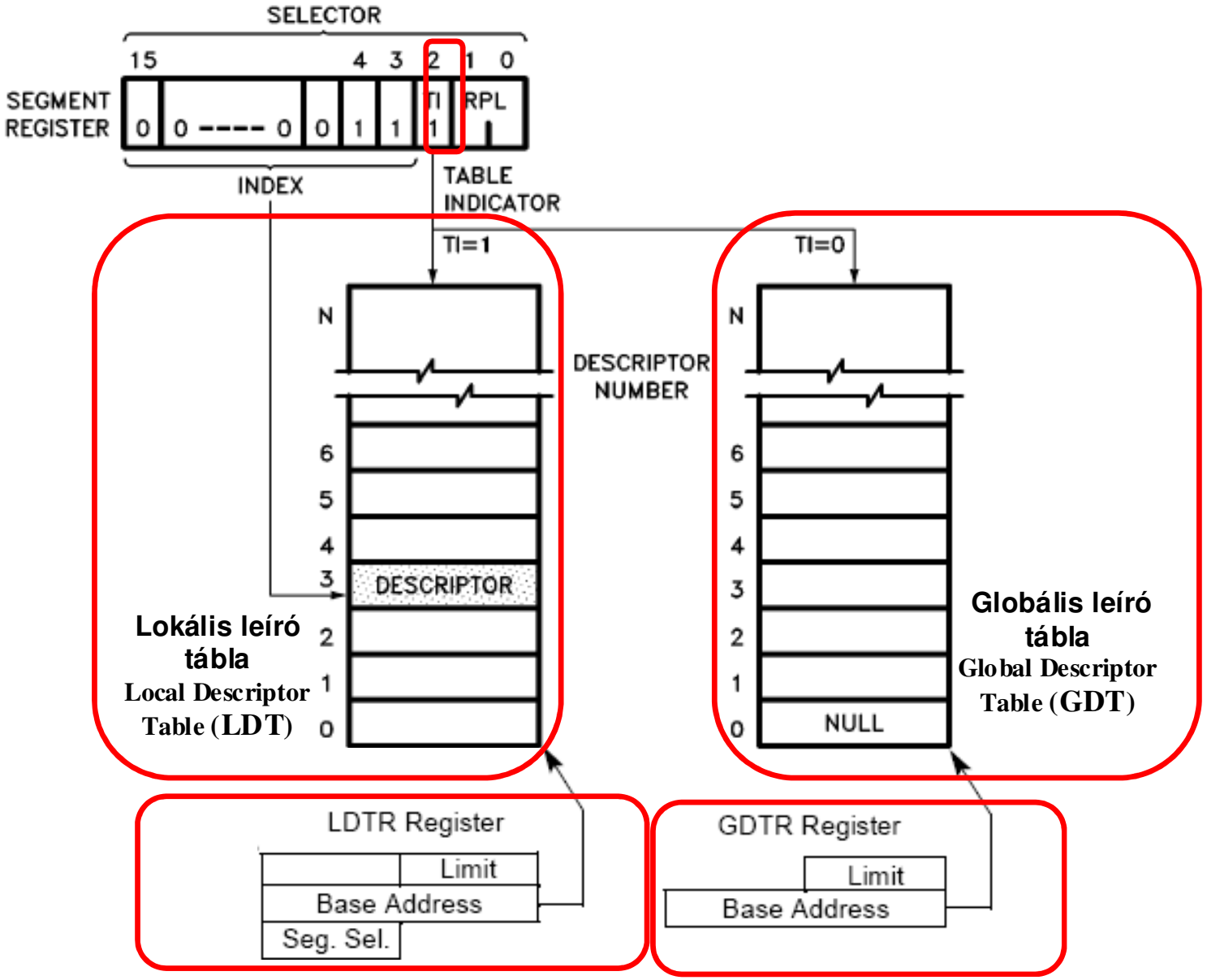
Rendszer szegmens regiszterek

	15 … 0
TR	SELECTOR
LDTR	SELECTOR

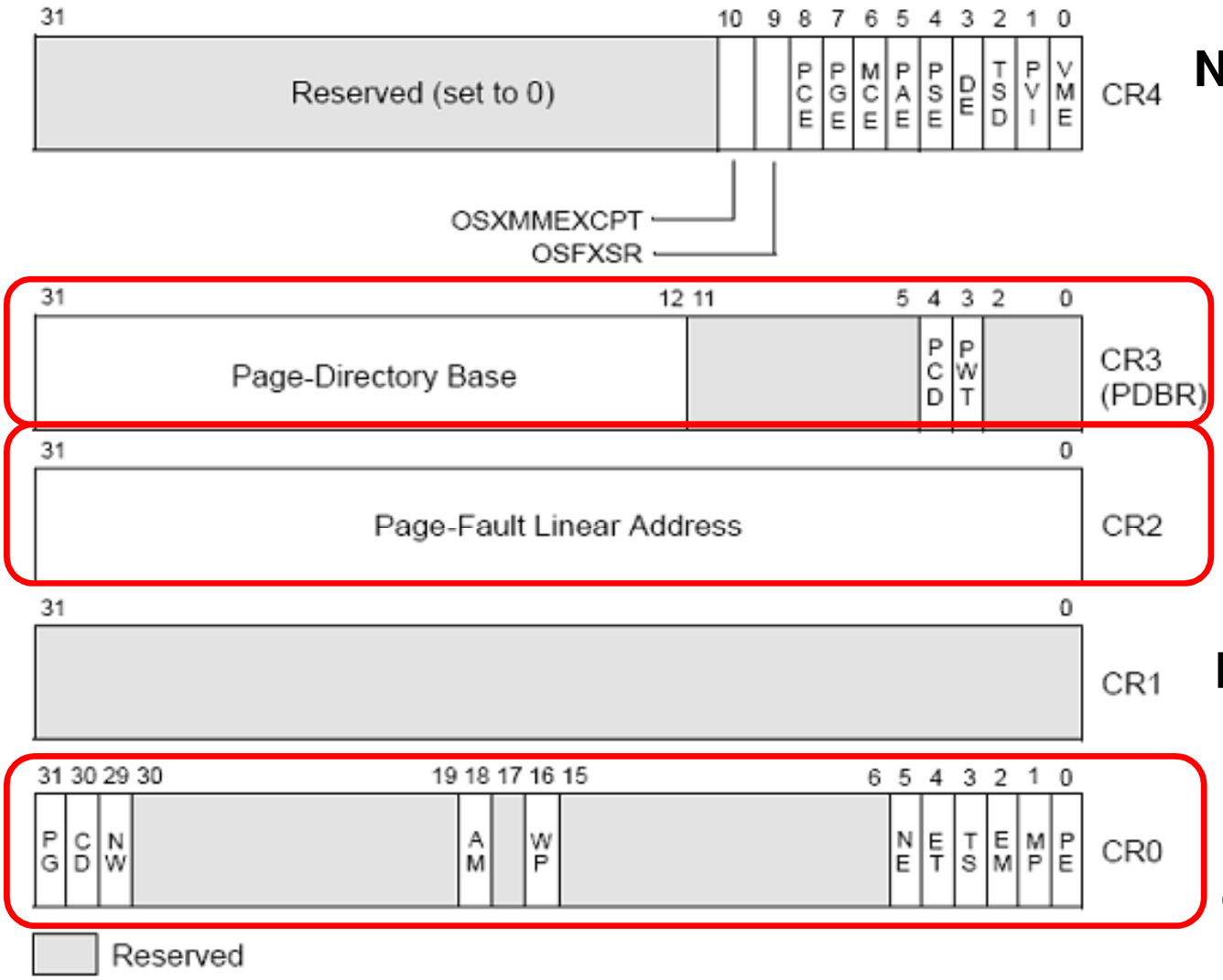
Leíró regiszterek /automatikusan töltődnek/

32 bit lineáris Báziscím	20 bit Limit	ATTRIB.

❖ Szegmens leíró táblák



❖ Vezérlő regiszterek



Nincs 386/486-nál

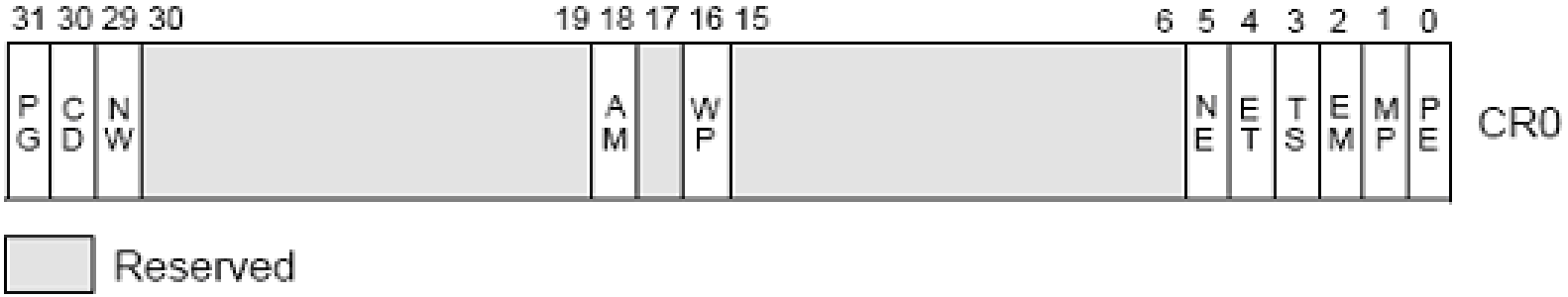
Laptábla könyvtár címe /20 bit/

Laphiba lineáris címe

Foglalt

Rendszervezérlő és jelző /*MSW*/

➢ Rendszervezérlő és jelző /*MSW*/



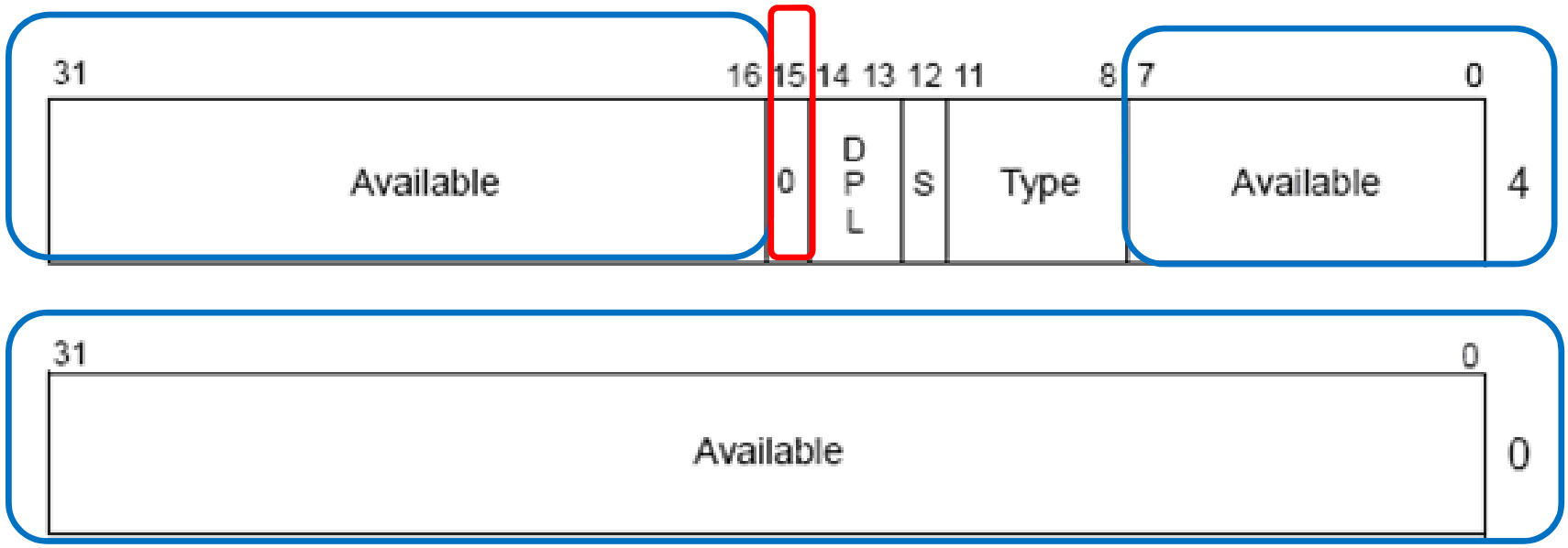
PE ***Védett üzemmód***
MP ***Math present***
EM ***Emulált Coprocessor*** */szoftver trap lebegőpontos utasításkor/*
ET ***387 kiterjesztés /i386, 486SX/ NE x87numerikus hiba***
TS ***Taszk kapcsolat***
WP ***Írás eng./tilt. system privilégium szintnél***

/486 ***lsd. lapszervezés/***

AM ***Alignment maszk /illesztés ellenőrzés maszkolása)***
PG ***Lapszervezés be/ki***
CD ***Cache tiltás /486/***
NW ***CACHE keresztülírás/érvénytelenítés engedélyezése /486/***

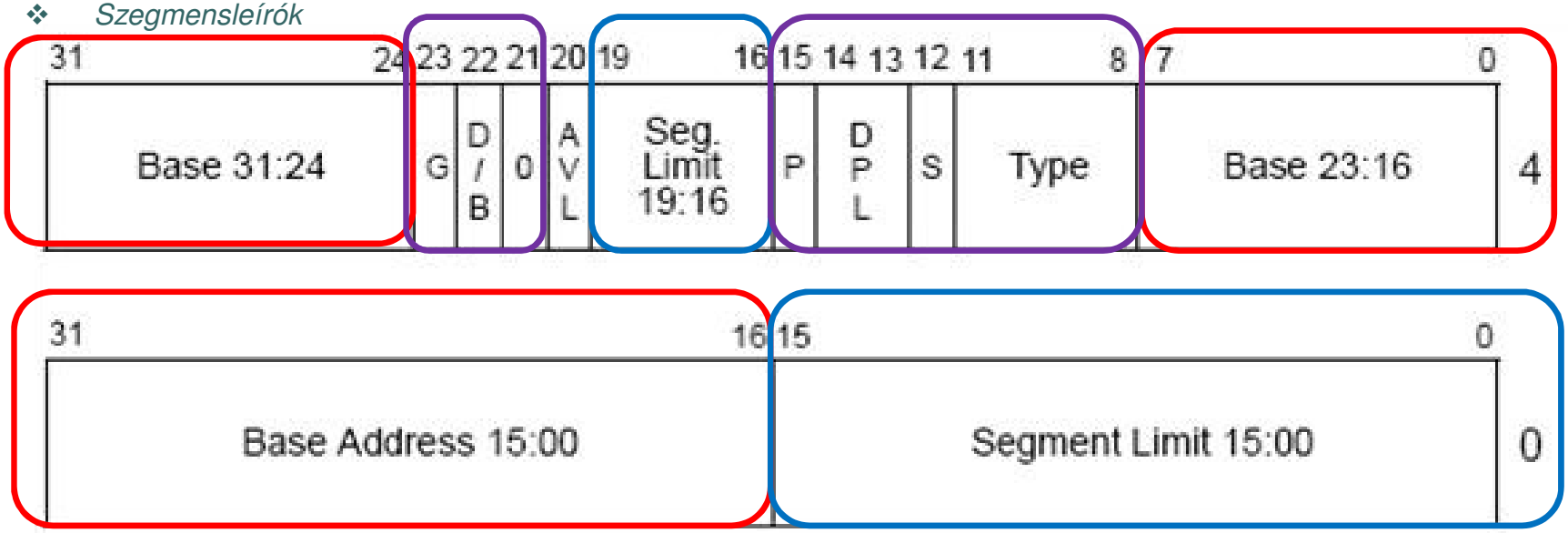
❖ Szegmensleírók

➢ Szegmens nincs bent /*P bit=0*/



- ❑ ***OR az Available mezőben tárolt információk alapján keresi meg a háttértárolón a szegmenst***
- ❑ ***Memória típusú leírók /S=0/***
- ❑ ***Rendszer leírók /S=1/***
- ❑ ***Leíró privilégium szint /DPL/***

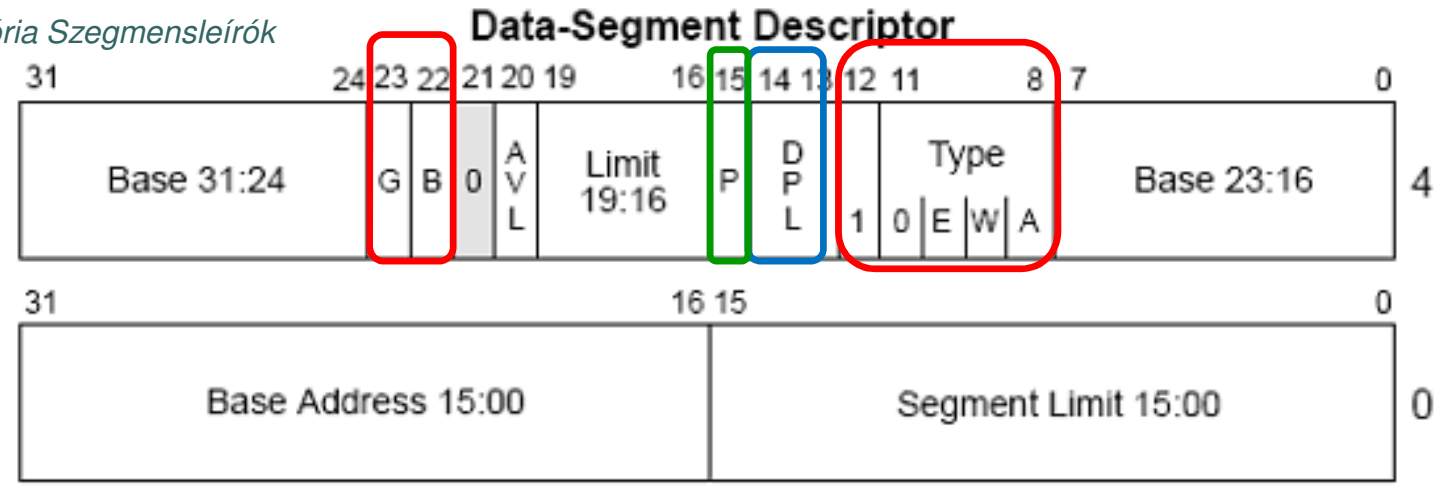
❖ *Szegmensleírók*

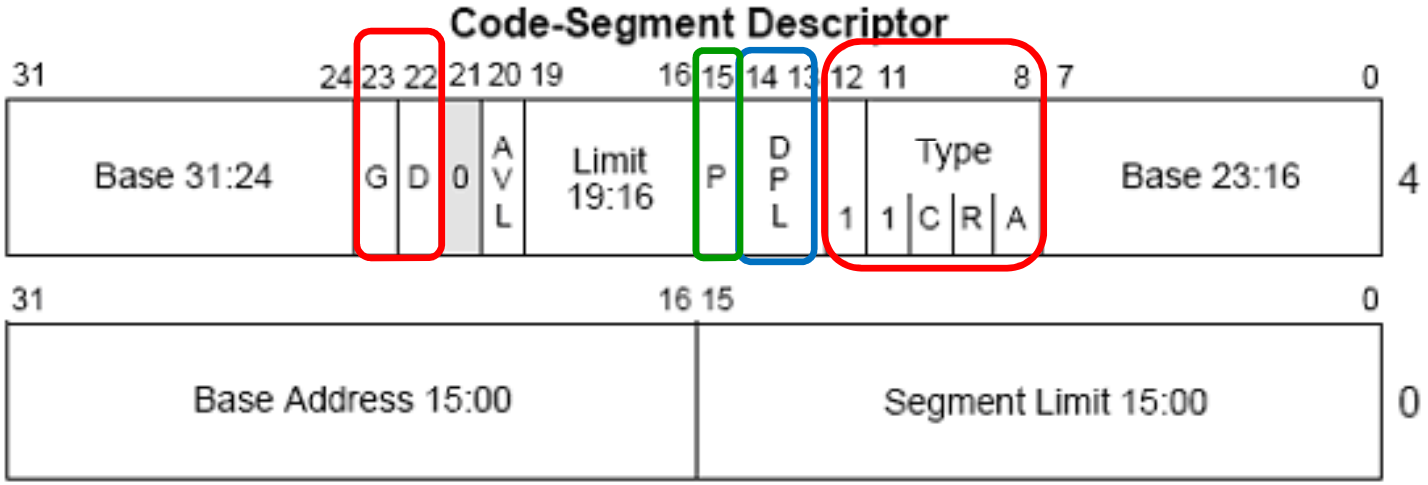


AVL — Available for use by system software
BASE — Segment base address
D/B — Default operation size (0 = 16-bit segment; 1 = 32-bit segment)
DPL — Descriptor privilege level
G — Granularity - *A limit mérete byte/ vagy 4KB egységekben*
LIMIT — Segment Limit
P — Segment present
S — Descriptor type (0 = system; 1 = code or data)
TYPE — Segment type

D/B - *SP méretét (16/32bit) és a stack limit (64KB/4GB) ellenőrzését is befolyásolja*

❖ *Memória Szegmensleírók*





A	Accessed
AVL	Available to Sys. Programmer's
B	Big
C	Conforming
D	Default
DPL	Descriptor Privilege Level
	Reserved

E	Expansion Direction
G	Granularity
R	Readable
LIMIT	Segment Limit
W	Writable
P	Present

Accessed:
Op. rendszer törölheti, processzor 1-be állítja hozzáférés esetén

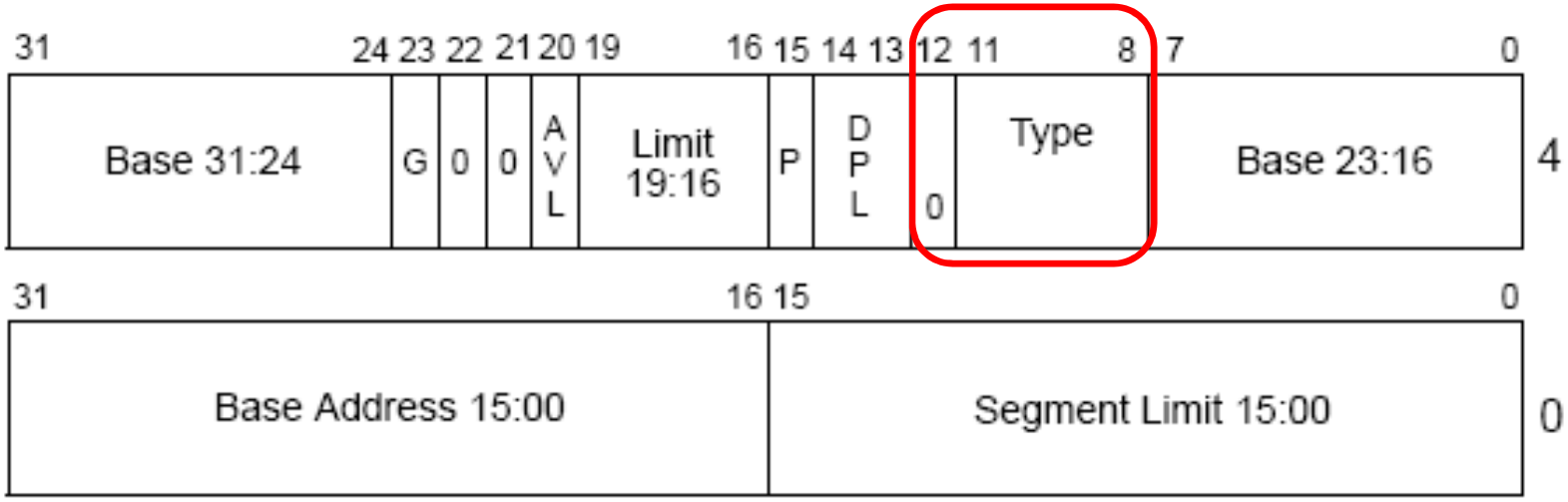
❖ Rendszerleírók

A rendszer működéséhez szükséges objektumok leírói

- **Taszkok leírói**
  - **LDT** ***Lokális leíró tábla leírója /GDT-ben/***
    - ***Szelektora TSS-ben***
  - **TSS** ***Taszk állapot szegmens leírója /GDT-ben/***
    - **Szelektora TR-ben**
- **Elérést biztosító objektumok /*kapuk - leírók*/**

  A privilégiumszintekre épülő védelmet is biztosítják
  - **Call Gate** ***/GDT-ben, LDT-ben/***
  - **Interrupt Gate** ***/IDT-ben/***
  - **Trap Gate** ***/IDT-ben/***
  - **Task Gate** ***/GDT-ben, IDT-ben, LDT-ben/***

➢ LDT, TSS leíró



Típus mező bit				*Leíró*	*Típus mező bit*				*Leíró*
11	*10*	*9*	*8*		*11*	*10*	*9*	*8*	
0	0	0	1	*TSS 16bit*	0	1	0	0	*CALL Gate 16 bit*
0	0	1	1	*TSS 16bit Foglalt*	.	.	.	.	.....
1	0	0	1	*TSS 32 bit*	1	1	0	0	*CALL Gate 32 bit*
1	0	1	1	*TSS 32 bit Foglalt*	1	1	1	0	INT Gate 32 bit
0	0	1	0	*LDT*	1	1	1	1	TRAP Gate 32 bit
					0	1	0	1	TASK Gate

➢ Kapu leírók

❖ Privilégium szintek

15 szegmensregiszter 2 1 0

TI tábla indikátor

RPL privilégium szint *[CPL, RPL, DPL]*

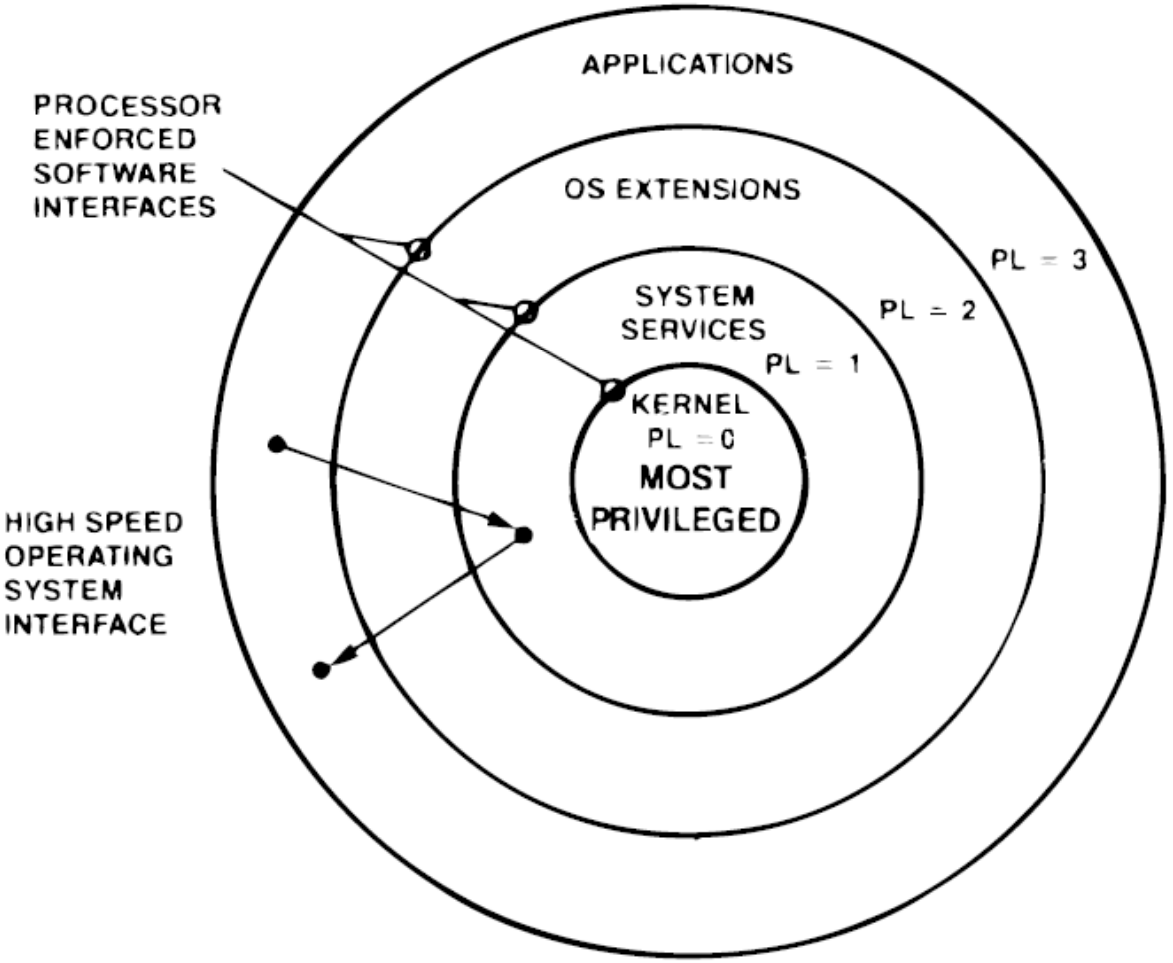
0-3 privilégium szint → 0 a legmagasabb

- **CPL** ***A futó taszk vagy program kód privilégium szintje***
  Aktuális privilégium szint
  - **Kivétel konform** *[conforming]* **kódra történő áttéréskor**
  - **Az aktuális privilégium szint meghatározza a *privilegizált - és I/O* utasítások *végrehajtását*, szegmensek, leírók *elérését***
- **RPL** ***Szelektor privilégium szintje*** → **operandus, hivatkozás**
  - ***Kérő szintje***
  - ***Effektív privilégium szint /EPL/***
    [RPL,CPL]max=EPL

- **DPL** ***A szegmensleíró vagy kapu privilégium szintje***
  - ***Adatszegmensnél az a szint, amelynél alacsonyabb /nagyobb számértékű/ szintről nem férhetnek hozzá***
  - ***Kódnál azt a privilégium szintet mutatja, amellyel egy taszknak vagy programnak rendelkeznie kell hozzáféréskor***
  - ***Call gate –nél azt a legnagyobb számértékű privilégium szintet jelzi, amelynél még a hozzáférés megengedett /mint az adatnál/***
  - ***TSS mint az adatszegmensnél***
- **Minden objektum elérésekor *ellenőrzés***
  - ***Még az adott szegmensregiszter tényleges átírása előtt***
  - ***A szabályok megsértése #GP védelmi szint megsértése kivételkezelést indít***

➢ Egy lehetséges védelmi elrendezés /gyűrűs védelem/

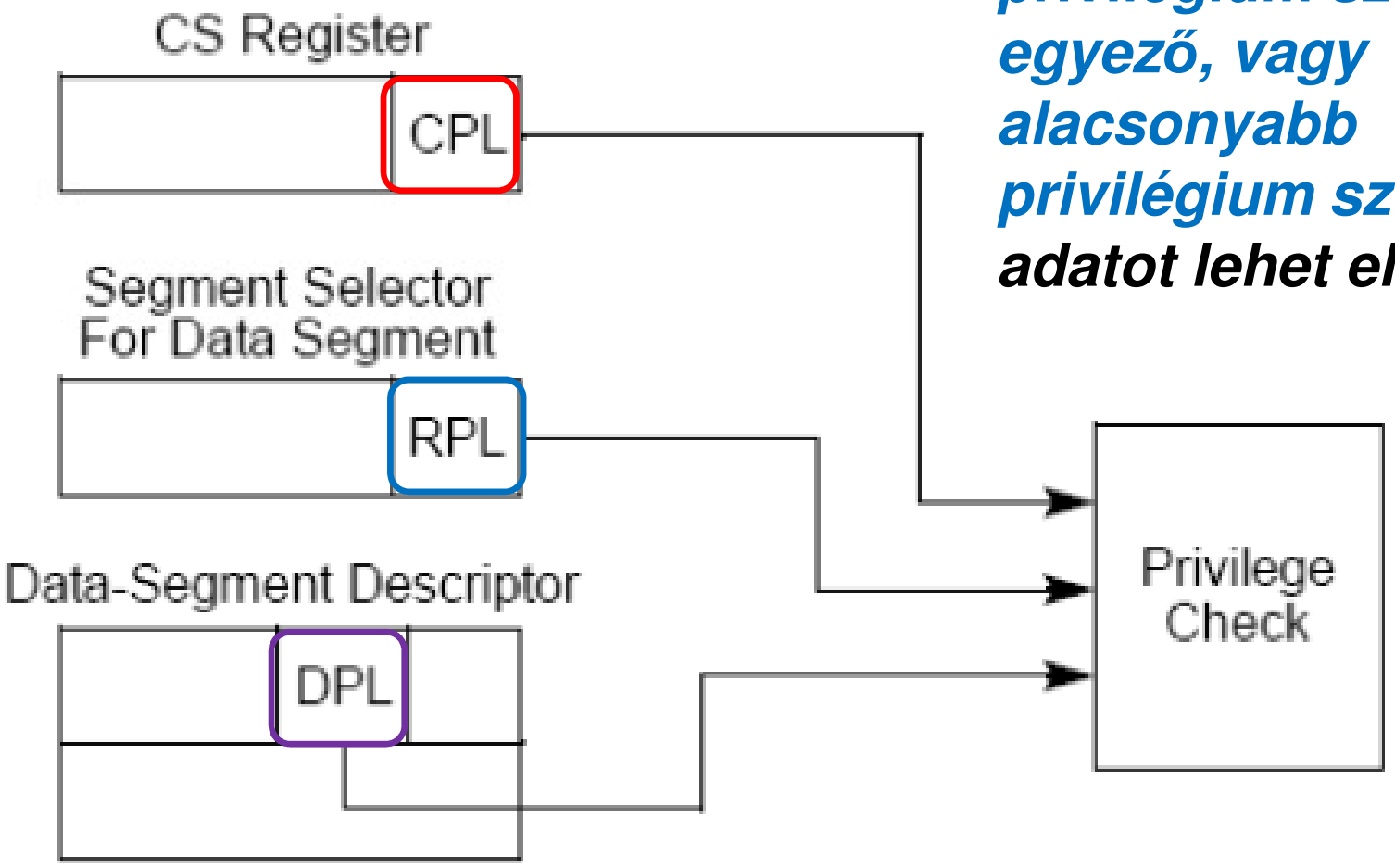
❑☞ Használata Opercíós rendszer függő 👍👎



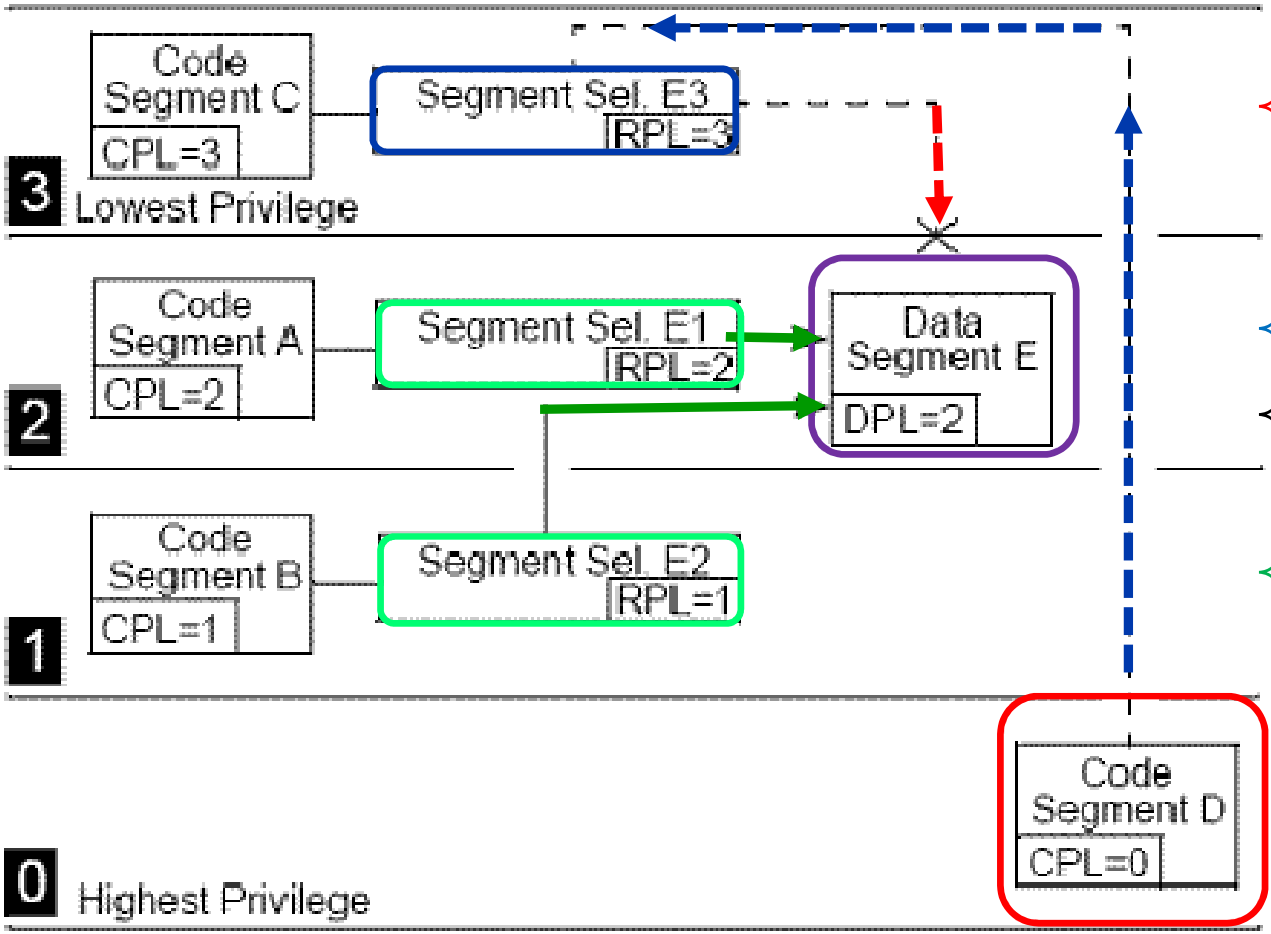
❖ Adatelérés privilégium ellenőrzés

➢ *[CPL, RPL]max ≤DPL*

Csak az aktuális privilégium szinttel egyező, vagy alacsonyabb privilégium szintű adatot lehet elérni



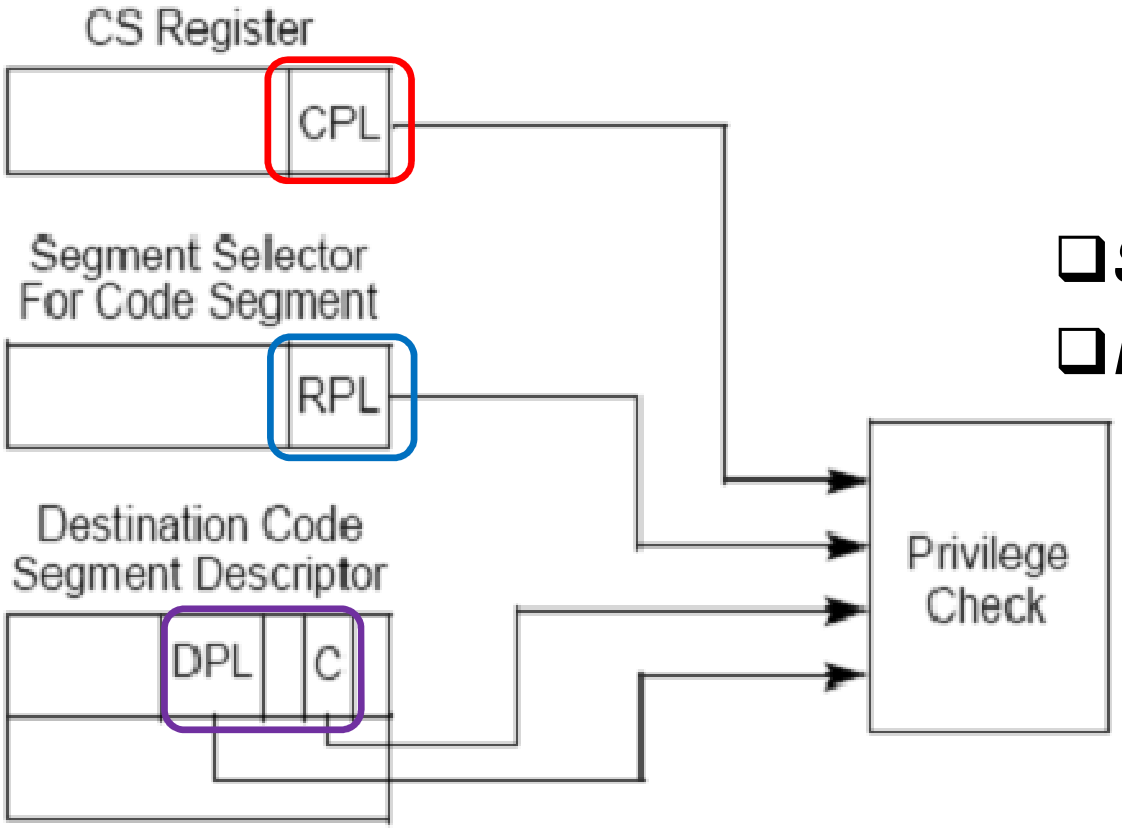
➢ [CPL, RPL]max ≤DPL



- ✧ ***D szegmens /CPL=0/ sem érheti el E-t***
- ✧ ***E3 szelektorral***
- ✧ ***elérhetné pl.:E1*** 👍 ***vagy***
- ✧ ***E2 szelektorral*** 👍 ☞ ***/adatszegmens szelektorának RPL-je*** *programból állítható*

❖ Privilégium szint ellenőrzés vezérlés átadáskor

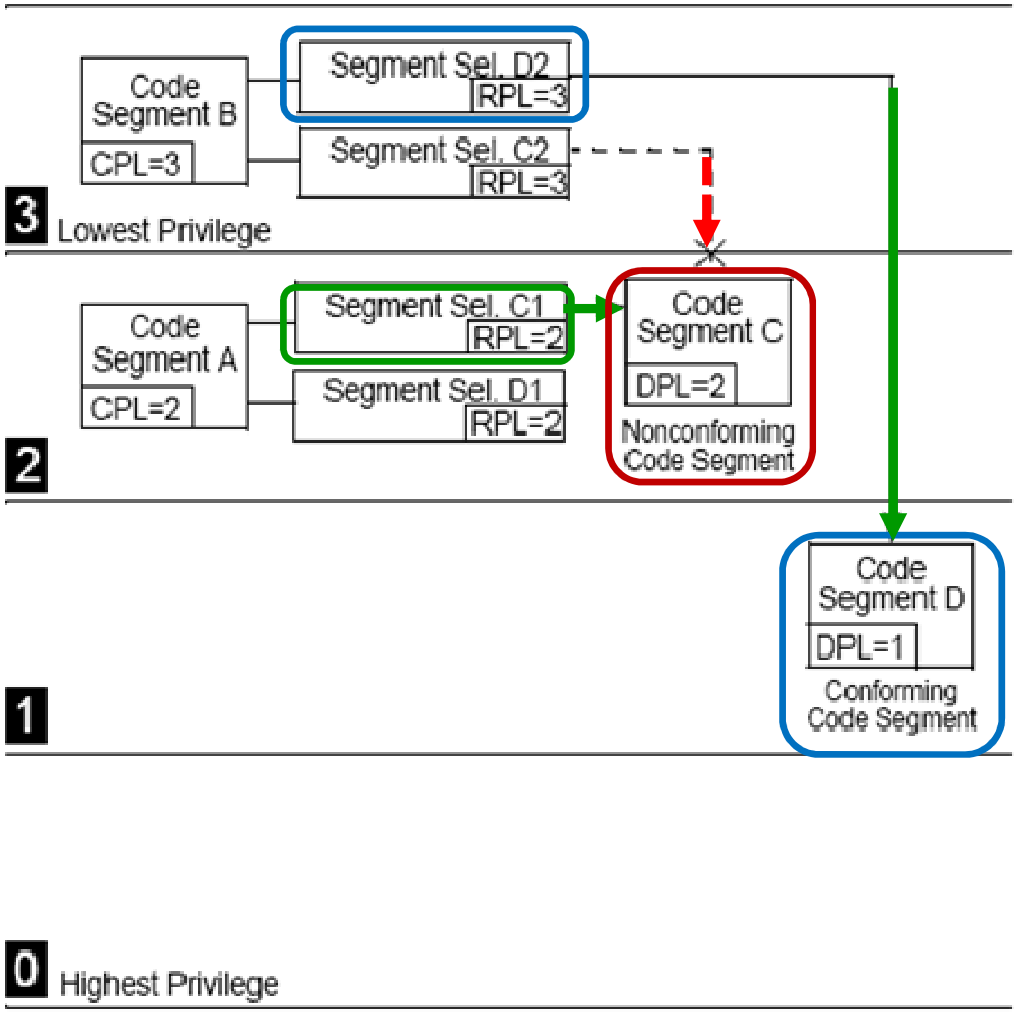
- ➢ **Utasítások**
  - ❑ **JMP, CALL, RET,**
  - ❑ **INTn, IRET**
  - ❑ **SYSENTER,SYSEXIT**



CS betöltésekor /előtt/ CPU megvizsgálja a cél kód szegmens leíróját

- ❑ ***Végrehajtja***
  - ■ ***Típus***
  - ■ ***Privilégium***
  - ■ ***Limit ellenőrzést***
- ❑ ***Siker esetén betölti***
- ❑ ***Egyébként #GP /kivételkezelés/***

➢ Példa közvetlen vezérlés átadásra /JMP, CALL/



Nem konform kódnál

- ♦ **CPL=DPL és RPL≤CPL kell**
- ♦ ***Ellenkező esetben #GP***

Példában

C1 RPL

lehetne 0,1,2 👍

de 3 nem

Konform kódnál

- ♦ ***pl.: mat. lib.***

❑ **CPL≥DPL vagy #GP**

❑ ***RPL nem számít***

❑ ***Végrehajtáskor CPL nem íródik át !!*** 👍

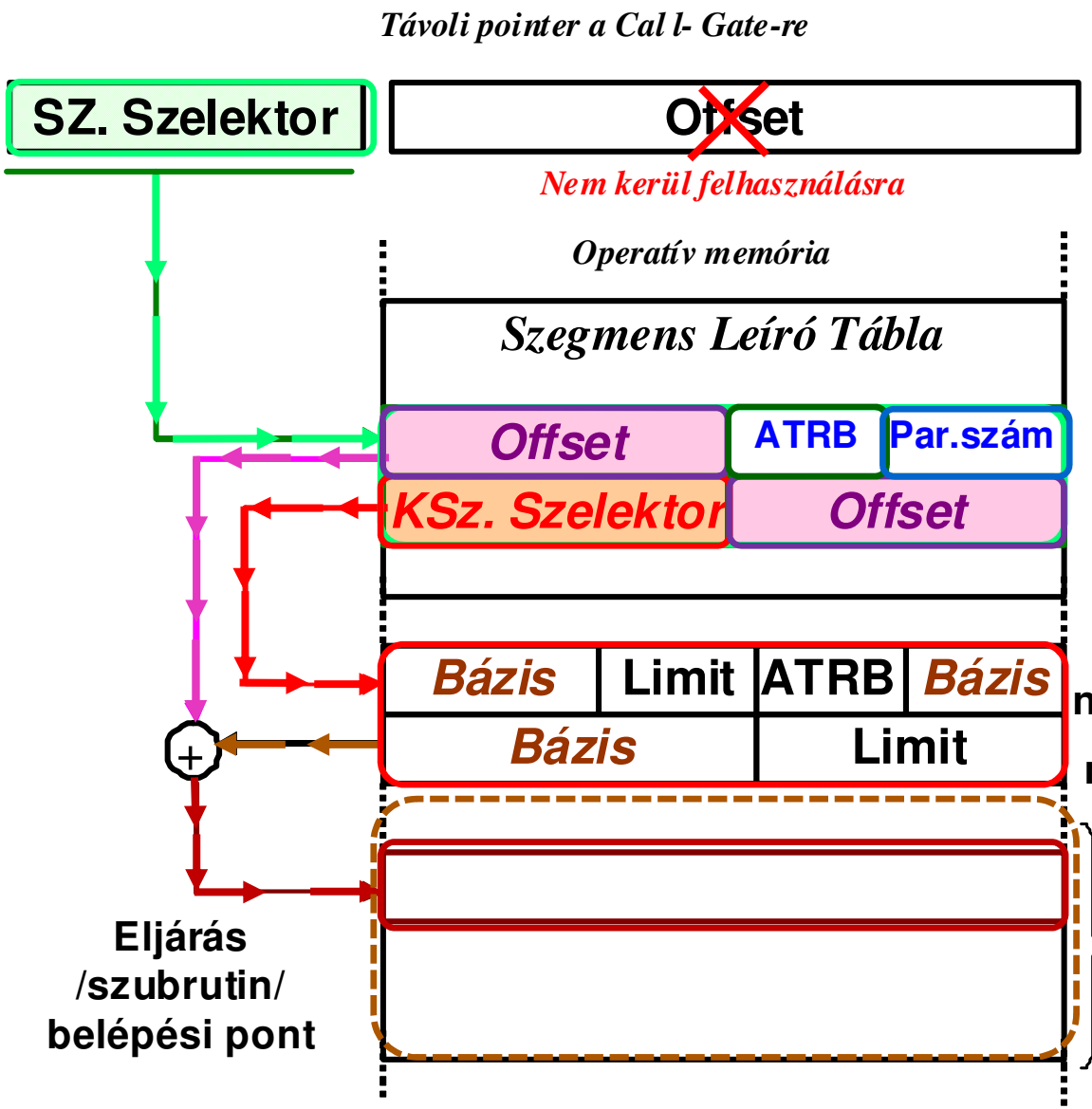
❖ Call Gate Vezérlés átadás kapun keresztül

DPL Descriptor Privilege Level
P Gate Valid

A kapu

- ***Meghatározza az elérendő kódszegmenst***
- ***Definiálja a kódszegmens eljárásának belépési pontját***
- ***Meghatározza az eljárás meghívását megkísérlő hívó hozzáféréshez szükséges privilégium szintjét***
- ***Ha stack kapcsolás történik, akkor meghatározza a stack-kek között átmásolandó opcionális paraméterek számát***
- ***Meghatározza a cél stack - be töltendő értékek méretét /16 bites kapu 16 bites töltést, 32 bites kapu 32bites töltést kényszerít.***
- ***Meghatározza, hogy érvényes e a CALL gate leíró /P bent van?! Normálisan ez 1, de OR használhatja speciális célra pl.:számláló/***

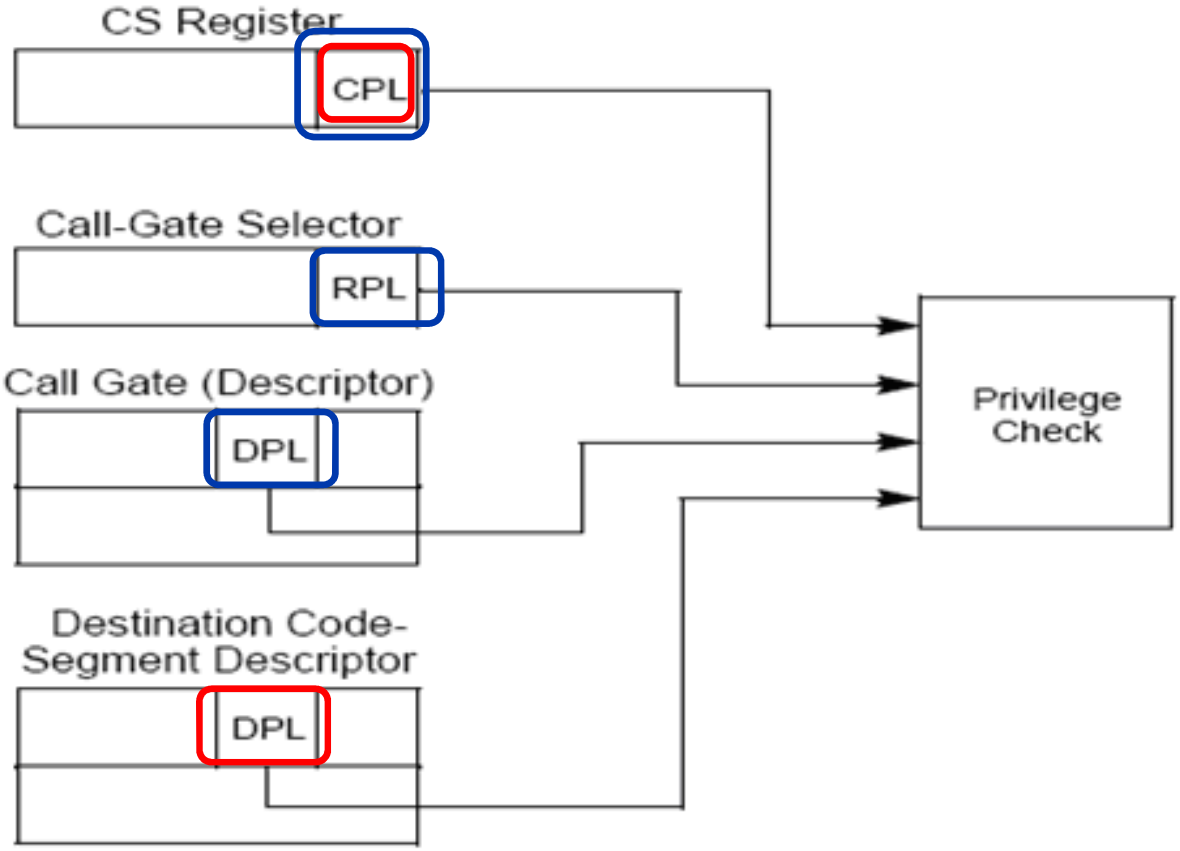
➢ ***Interrupt Gate a paraméter szám kivételével a fentivel hasonló***



Meghatározza

- A belépéshez szükséges prv. szintet
- Cél Kódszegmenst
- Cél Belépési pontját
- Stackban átadott paraméterek számát

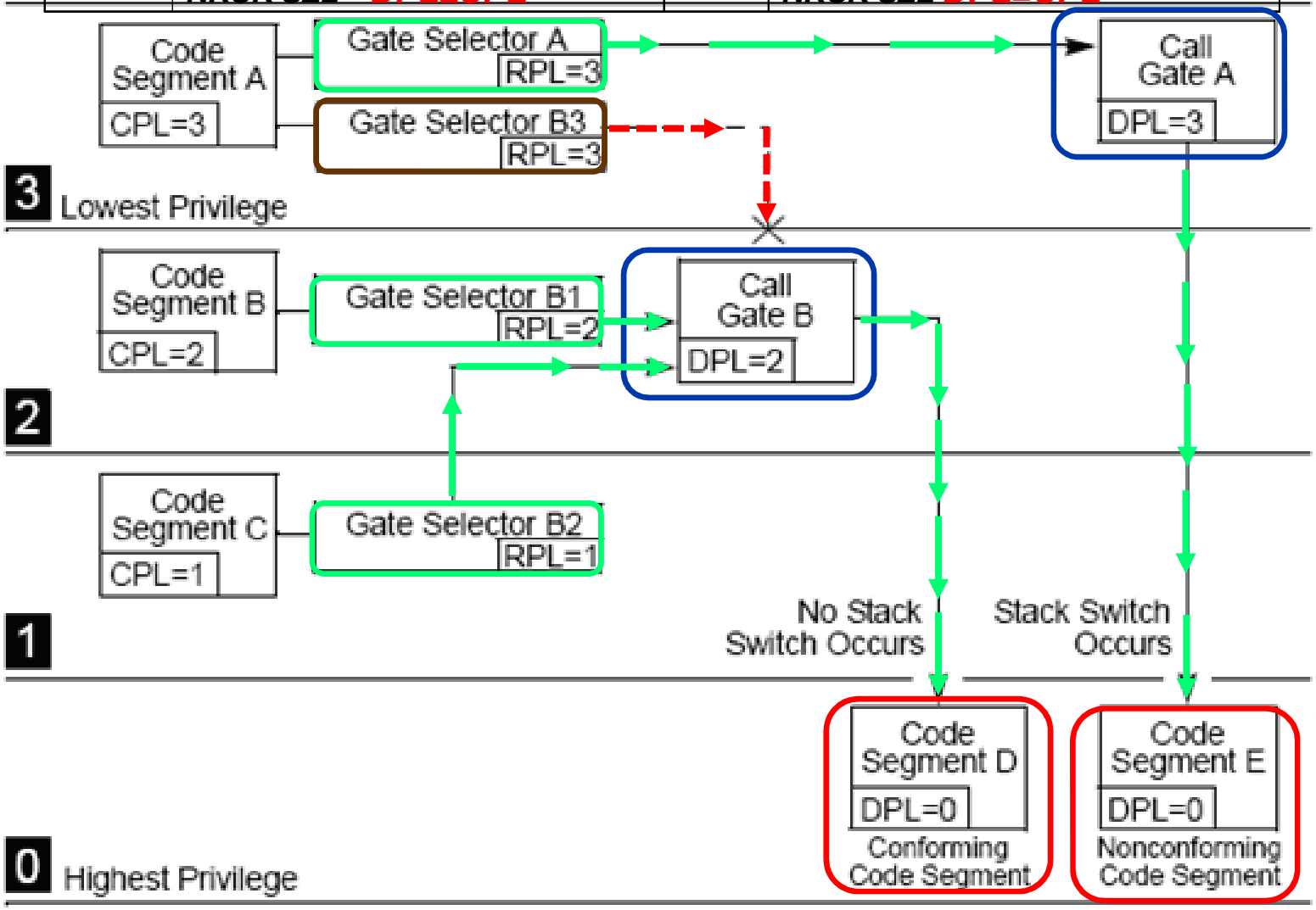
❖ Call Gate privilégium szint ellenőrzés



CALL	***CPL≤kapuDPL, RPL≤kapuDPL*** ***Konform célkód szegmensleíró DPL≤CPL*** ***Nemkomform célkód szegmensleíró DPL≤CPL***
JMP	***CPL≤kapuDPL, RPL≤kapuDPL*** ***Konform célkód szegmensleíró DPL≤CPL*** ***Nemkomform célkód szegmensleíró DPL=CPL***

❖ *Call Gate Vezérlés átadás kapun keresztül*

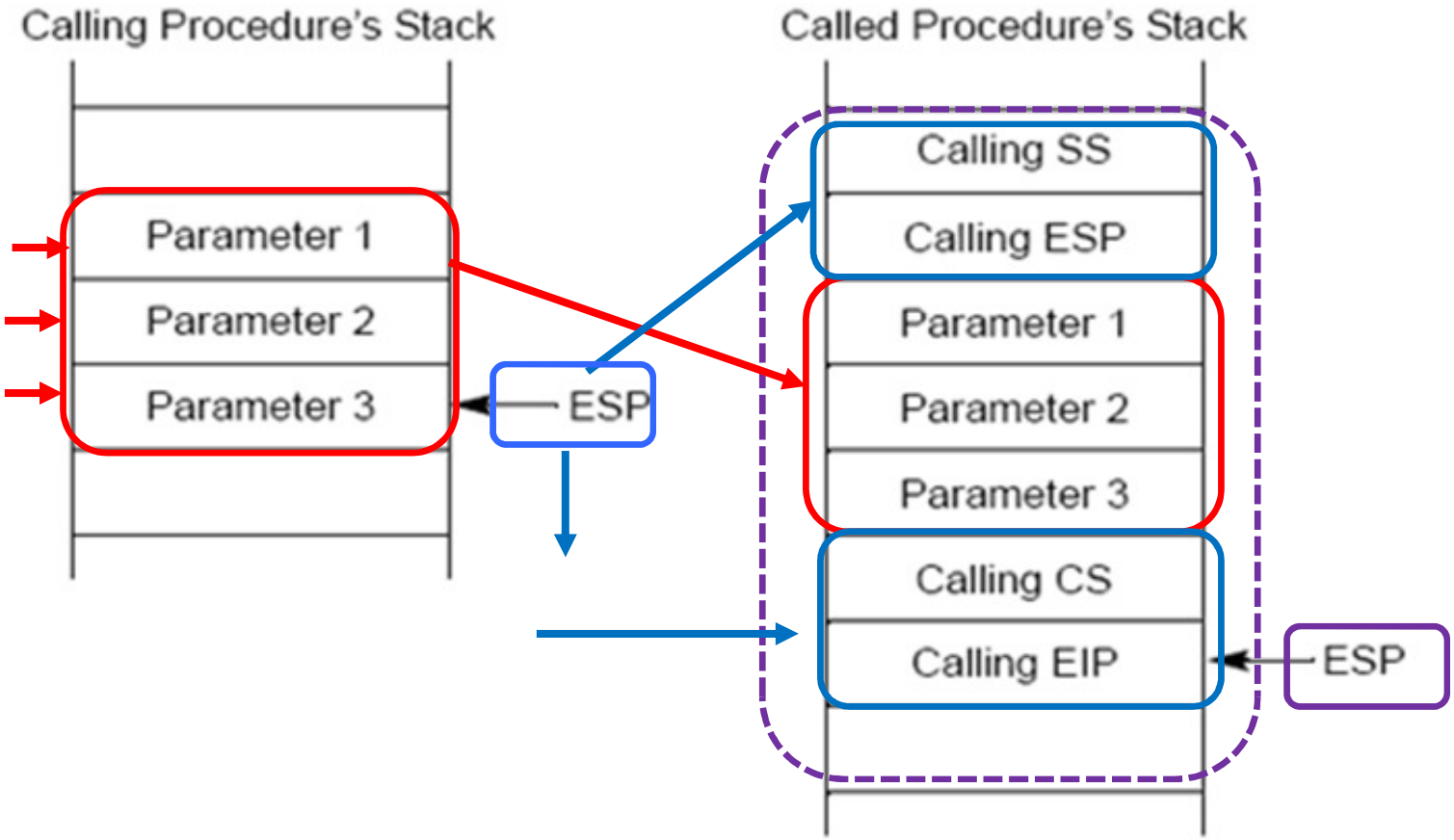
CALL	CPL≤kapuDPL, RPL≤kapuDPL KCK SZL DPL≤CPL NKCK SZL DPL≤CPL	JMP	CPL≤kapuDPL, RPL≤kapuDPL KCK SZL DPL≤CPL NKCK SZL DPL=CPL



- **Paraméter átadás /*STACK switch*/**
  - ***Minden szintnek saját SS, ESP a hibaterjedés megakadályozására*** 👍
  - ***Privilégium szint váltáskor /nem komform kódnál/***
  - **Processzor végrehajtja →a *CALL Gate kiválasztásakor*** 👍
    - ***Hozzáférési jogok, limit ellenőrzése CPL, RPL, kapuDPL, célkódDPL alapján*** 👍
    - ***Átmeneti -belső mentés- /SS, ESP, CS, EIP/*** 👍
    - ***SS, ESP töltése az aktuális /hívó rutin/ TSS-ből*** 👍
    - ***Átmásolja az átmenetileg mentett SS, ESP- t a cél szintjéhez rendelt stak-be*** 👍
    - ***Átmásolja a paramétereket a hívó stack-jéből az új stack-be*** 👍
    - ***Rámásolja az átmenetileg mentett CS-t, EIP-t*** 👍

✧ ***Betölti az új kódszegmens szelektorát CS-be az új EIP-t a kapu leíróból /ofset/ és megkezdi a végrehajtást***

❑ **Bonyolult műveletsorozat** 💣 ***Mikroprogramozott vezérlés*** 👍

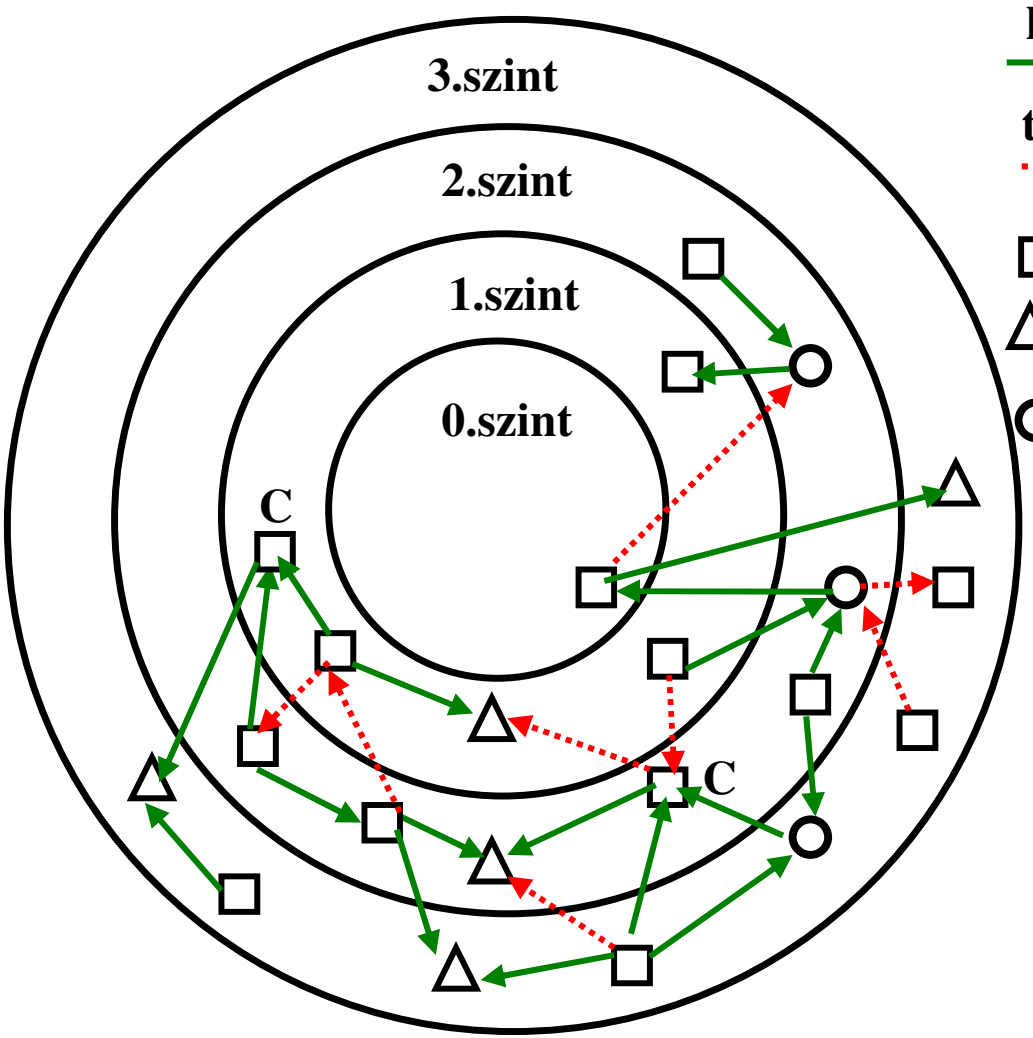


❑ Pointer ellenőrző utasítások

A kivételkezelés elkerülésére az OR SW is ellenőrizheti 👍

- ♦ **ARPL** ***[CPL, RPL]max=EPL Paraméterátadás után***
- ♦ **VERR/VERW** ***Olvashatóság, írhatóság ellenőrzésére***
- ♦ **LAR** ***Hozzáférési jogok ellenőrzésére*** 👍
- ♦ ***LSL Szegmens limit ellenőrzésére*** 👍

Hozzáférési szabályok



legális

tiltott

kód

adat

kapu

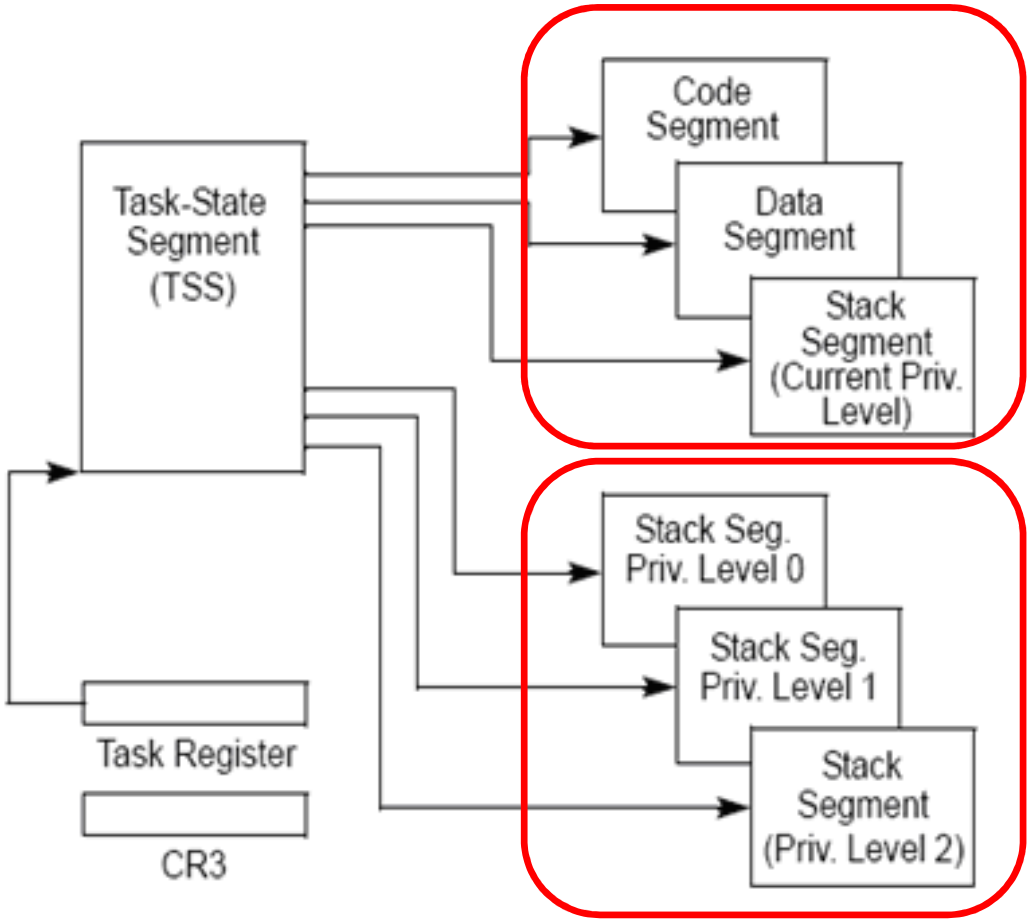
Hogyan lehet alacsonyabb privilégium szintre lépni?

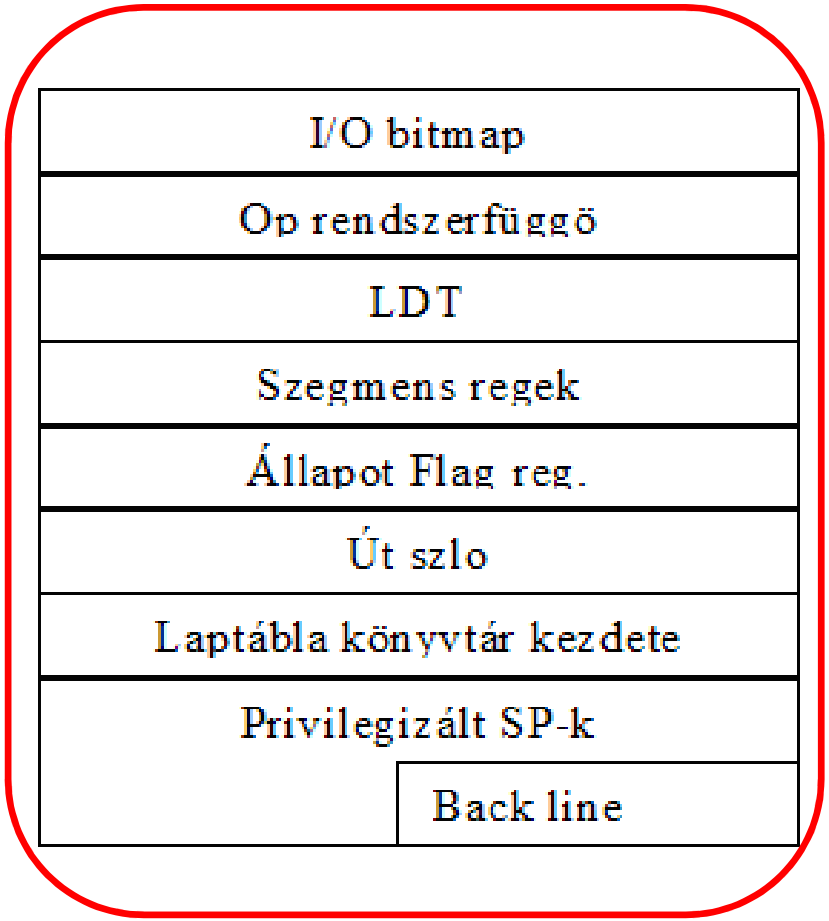
Várom a választ!

❖ Taszkkezelés

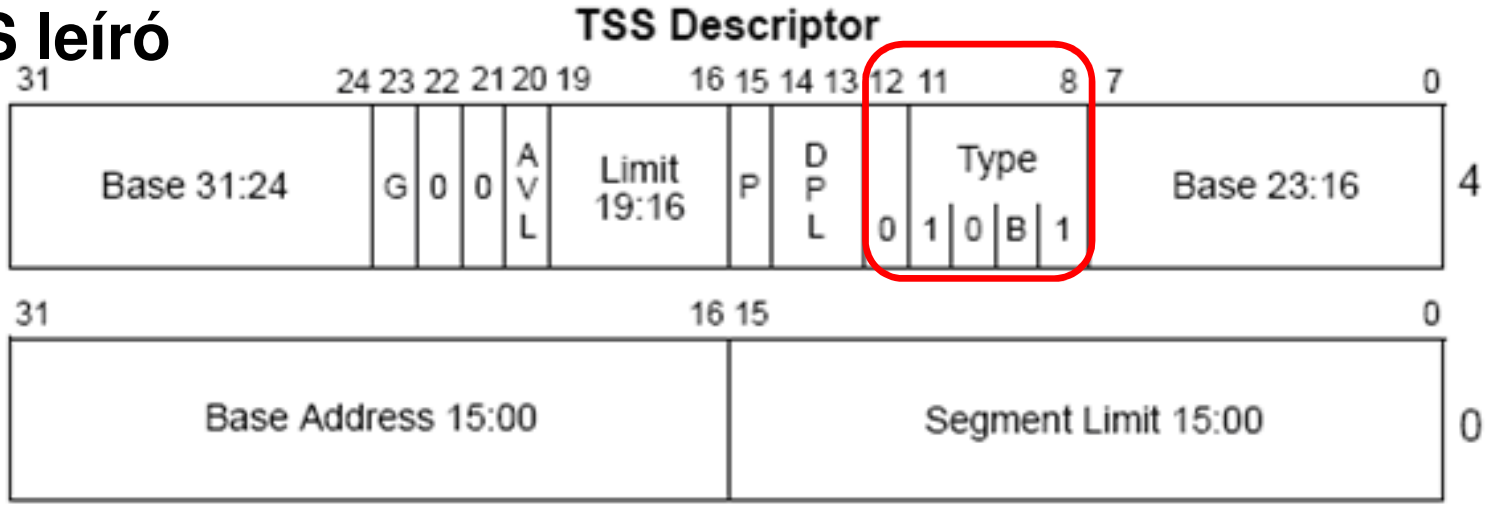
➢ Taszk állapot szegmens /TSS/

- ❑ ***A taszk pillanatnyi állapotát és környezetét írja le***
  - ☞ ***Virtuális <<>>fizikai processzor összerendeléshez***





➢ TSS leíró

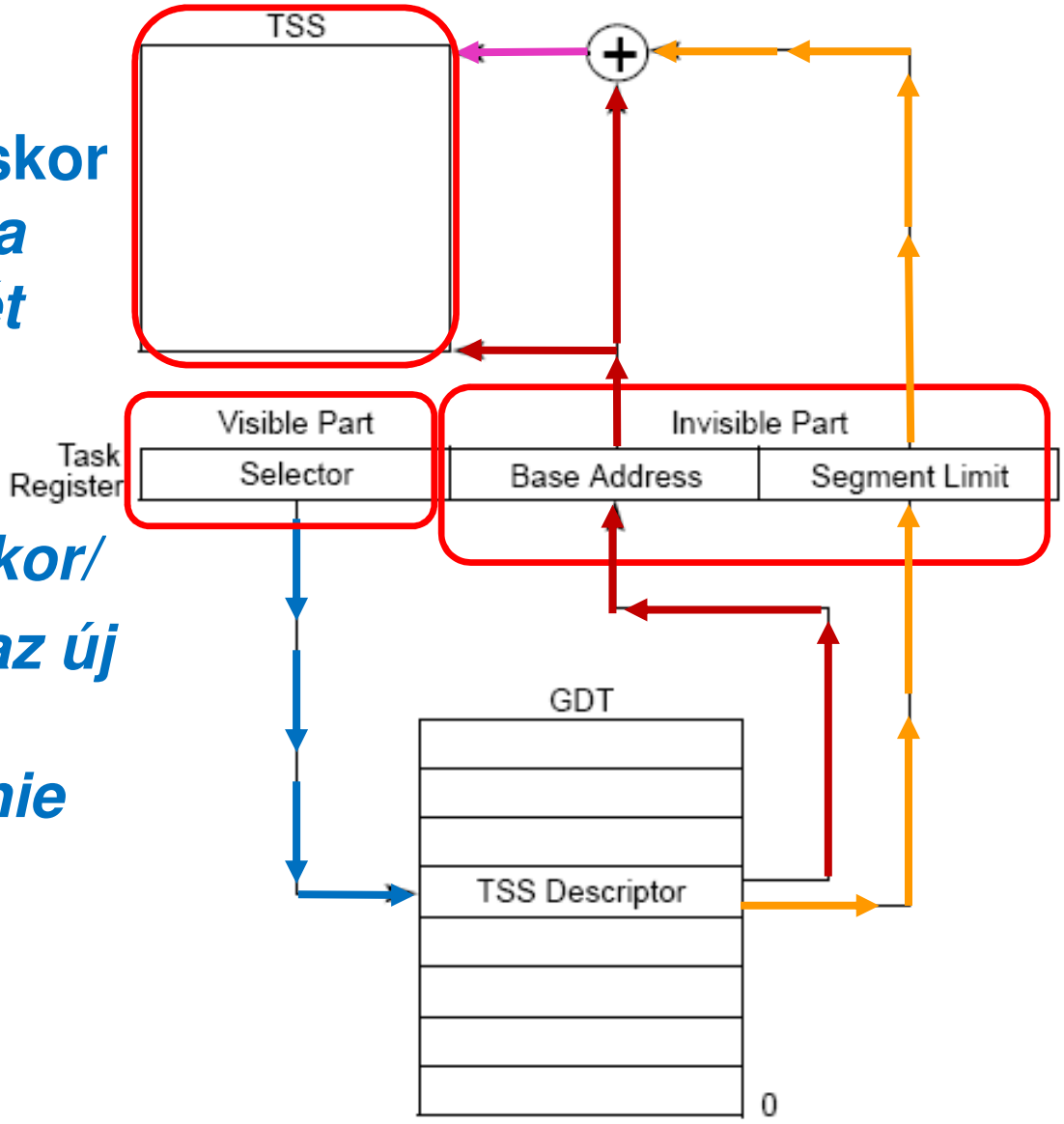


AVL	Available for use by system software
B	Busy flag
BASE	Segment Base Address
DPL	Descriptor Privilege Level
G	Granularity
LIMIT	Segment Limit
P	Segment Present
TYPE	Segment Type

- ❑ ***Új taszk létrehozásakor , illetve telepítésekor az OR létrehozza a TSS leíróját a GDT- ben***
- ❑ ***TSS kezdeti értékekkel a háttértárolón***

➢ TSS betöltése - *TR használata*

- ❑ **Rendszer indításkor** ***LTR utasítás adja TR induló értékét***
- ❑ ***Taszkváltáskor töltődik /TSS kiolvasásakor/***
- ❑ ***Taszkváltáskor az új TSS-nek is az OM-ben kell lennie***

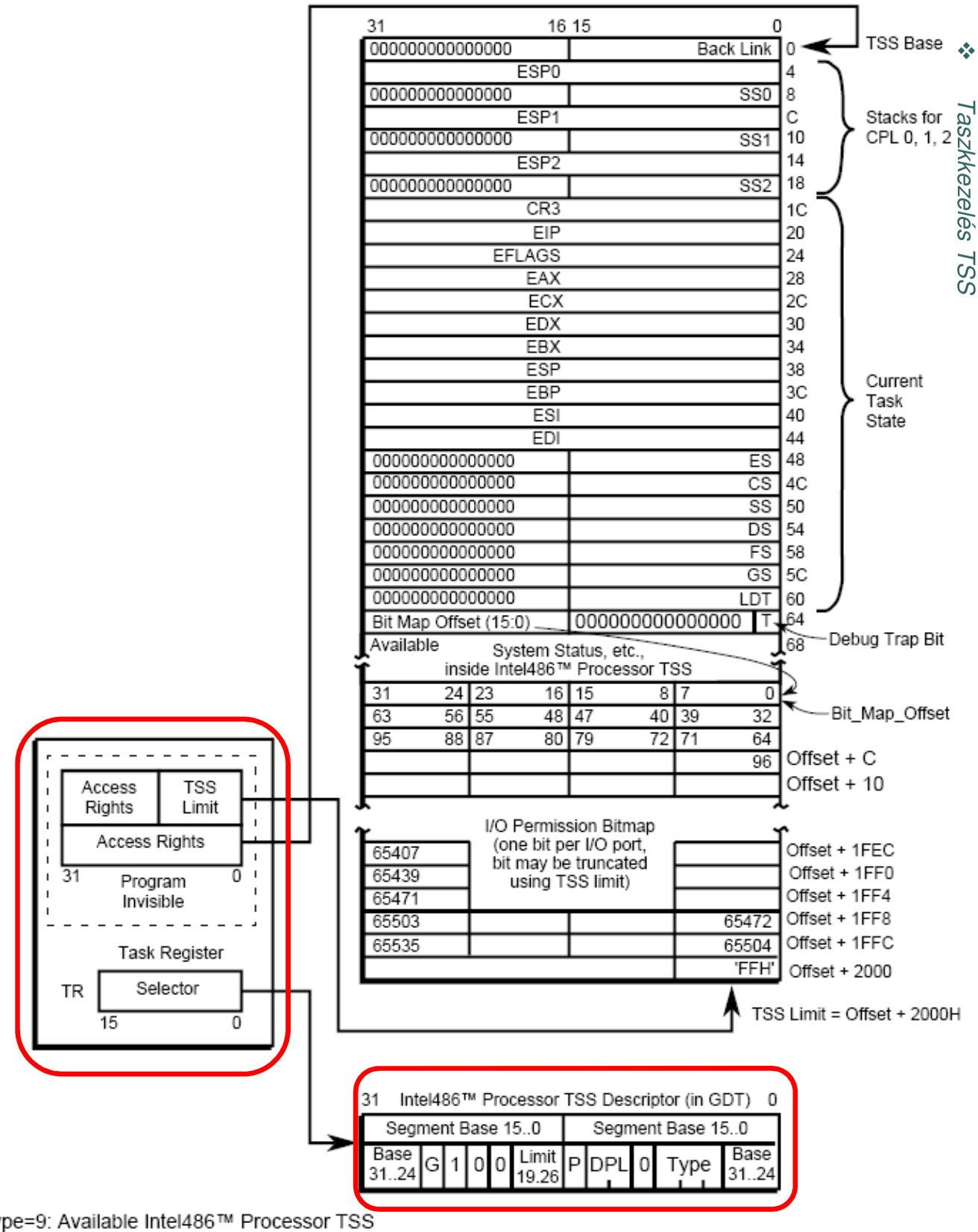


➢ TSS *HW függő része*

- ❑ ***A processzor taszkváltáskor automatikusan kezeli*** 👍
  - ☞ ***Nem léphet át laphatárt*** 💣

31 — 15	0	
I/O Map Base Address	T	100
	LDT Segment Selector	96
	GS	92
	FS	88
	DS	84
	SS	80
	CS	76
	ES	72
EDI		68
ESI		64
EBP		60
ESP		56
EBX		52
EDX		48
ECX		44
EAX		40
EFLAGS		36
EIP		32
CR3 (PDBR)		28
	SS2	24
ESP2		20
	SS1	16
ESP1		12
	SS0	8
ESP0		4
	Previous Task Link	0

Reserved bits. Set to 0.

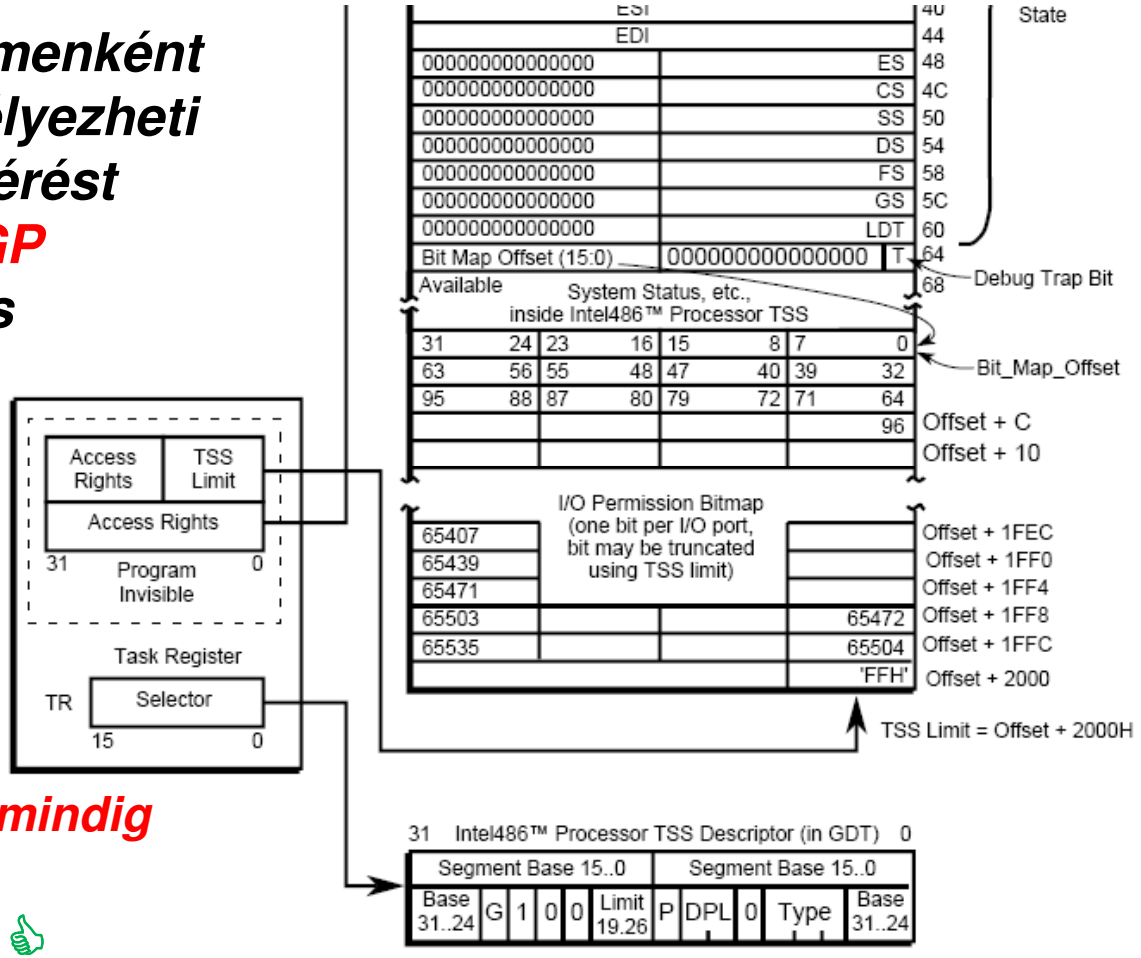


Type=9: Available Intel486™ Processor TSS
Type=B: Busy Intel486™ Processor TSS

Note:
BIT_MAP_OFFSET must be ≤ DFFFH

➢ I/O szenzitív utasítások csak ha CPL≤IOPL /EFLAGS/

- ❑ **Bitmap I/O címenként külön engedélyezheti /0 a bit/ az eltérést egyébként #GP kivételkezelés**



- ❑ **VM86 módban mindig ellenőrzi az I/O védelem miatt**

Type=9: Available Intel486™ Processor TSS
Type=B: Busy Intel486™ Processor TSS

Note:
BIT_MAP_OFFSET must be ≤ DFFFH

➢ Több helyről elérhető taszk kérdése

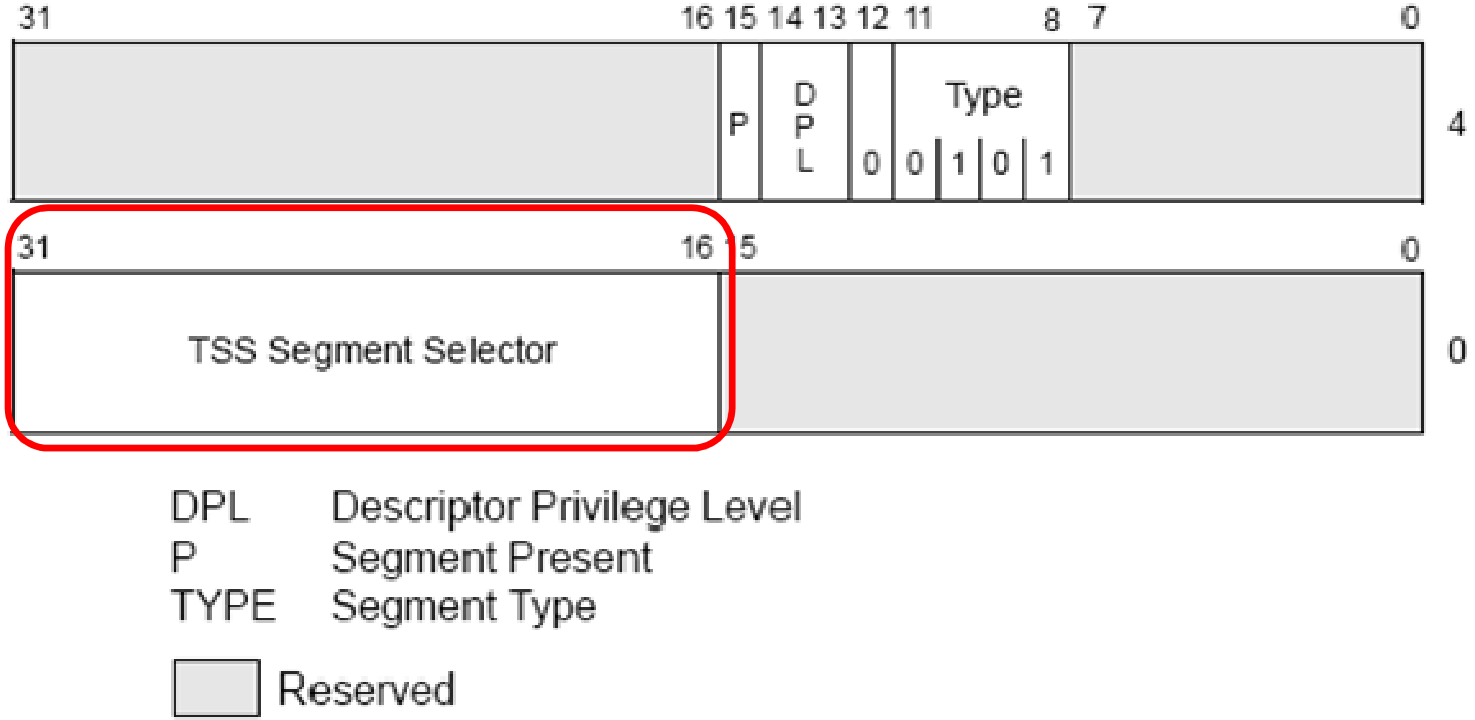
❑ Csak egy TSS van

Taszk nem lehet rekurzív, vagy reentrant

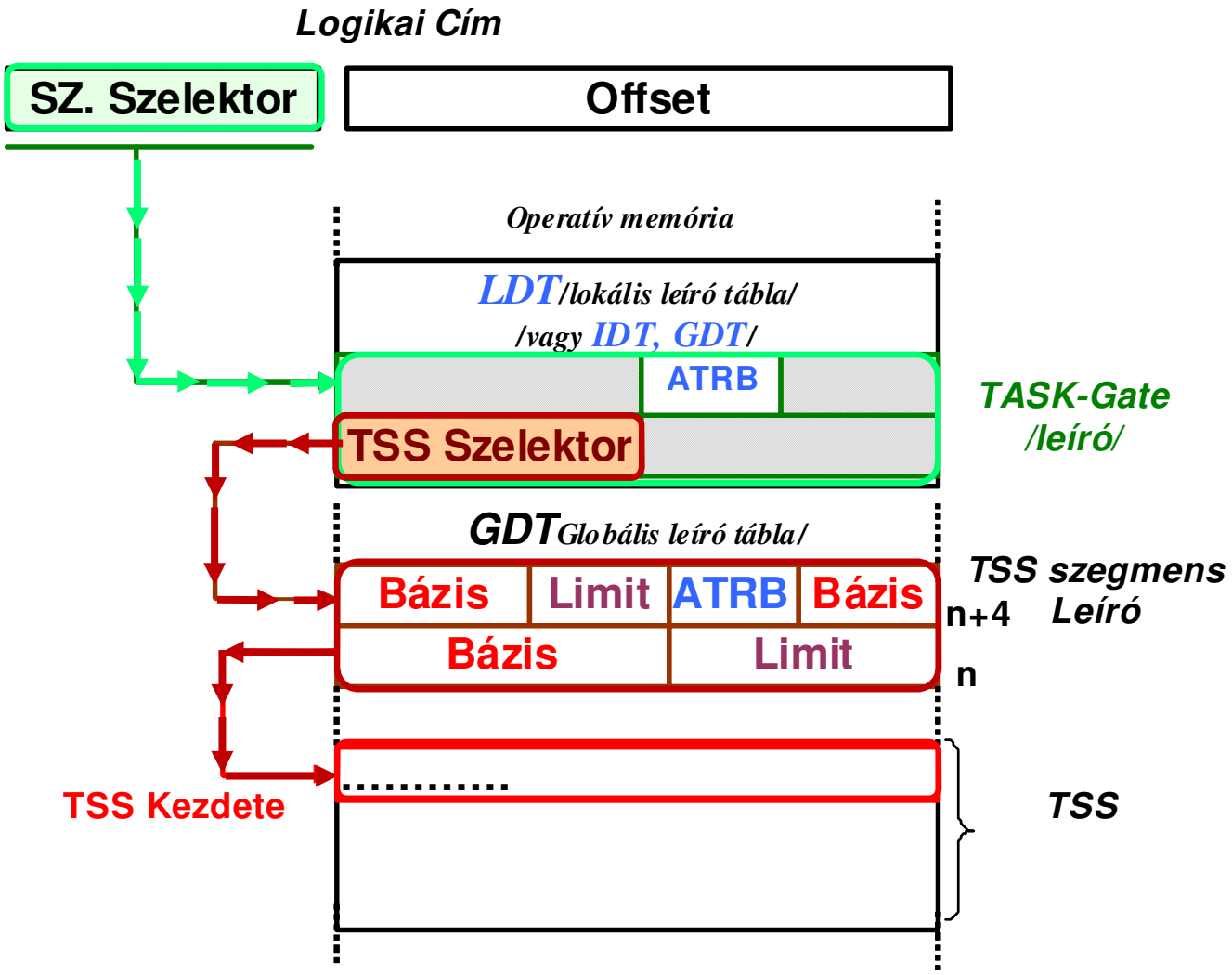
B /busy/ bit a leíróban

❑ *Task Gate →Leíró* /rendszer objektum/

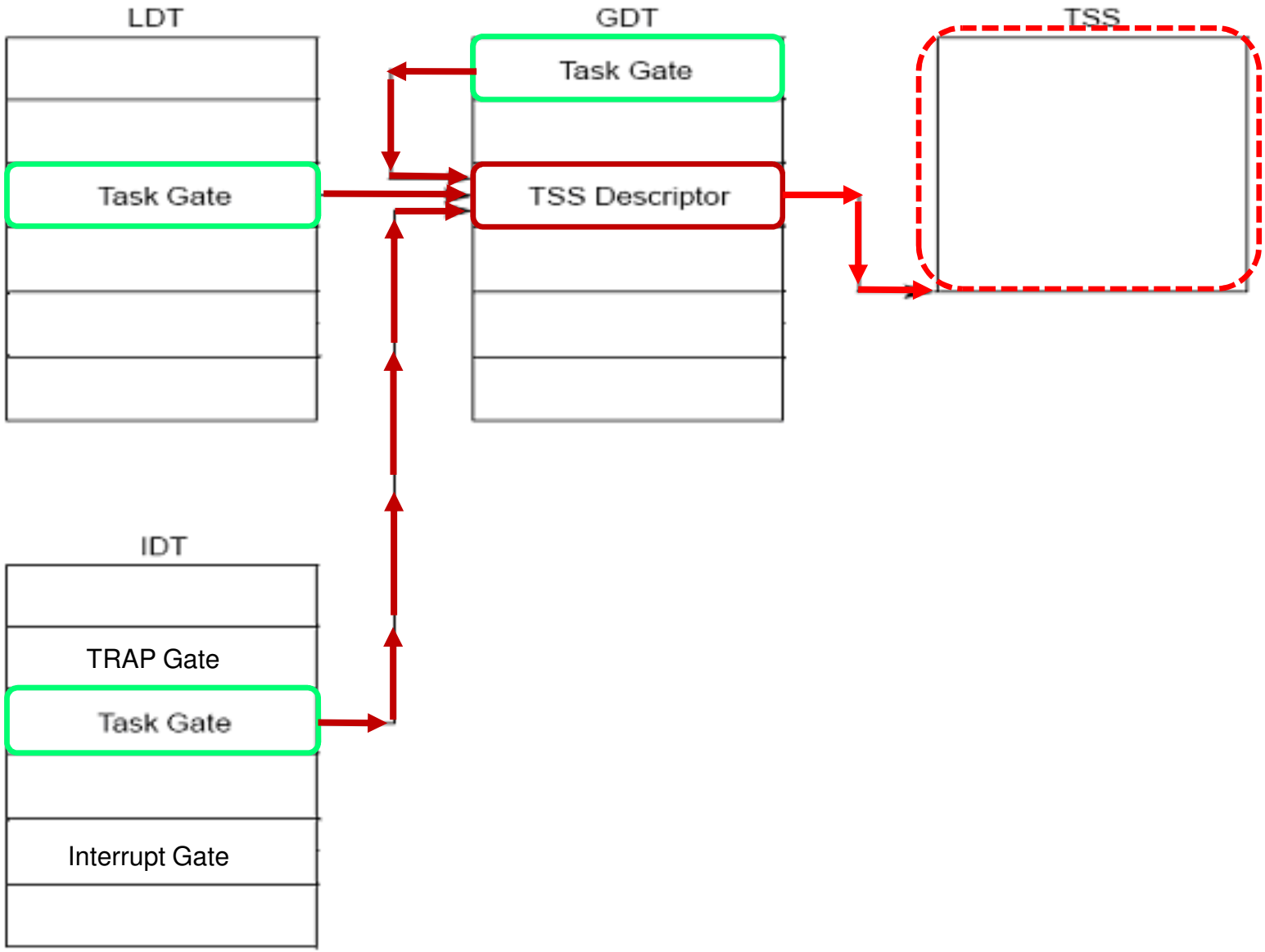
Ebből használja a szelektort aTSS leírójához



➢ Task Gate „működése”



➢ Alkalmazás


LDT
GDT
TSS
Task Gate
Task Gate
TSS Descriptor
IDT
TRAP Gate
Task Gate
Interrupt Gate

❖ Taszkváltás

- ***Új taszk futtatására való áttéréskor az OR***

  pl.: JUMP TSSi; utasítást /Call, Interrupt, Exeption/ ad ki /operandus az új taszk /TSSi/, vagy Task Gate szelektora/

- **A processzor**

  - ***Ellenőrzi a hozzáférési jogosultságát /CPL, RPL, DPL/*** 👍
  - ***Ellenőrzi az új TSS meglétét /betöltöttségét- limitjét /*** 👍
  - ***Ellenőrzi az új taszk rendelkezésre állását /B=?busy?/*** 👍
  - ***Ellenőrzi, hogy mindkét TSS és taszkváltáshoz szükséges összes szegmens leíró lapja, bent van a memóriában*** 👍
  - ***Törli a /vagy nem/ a régi B /busy/ bitjét*** 👍
  - ***Menti a régi TSS-t*** 👍
  - ***Beállítja az új TSS leíró B bitjét*** 👍
  - ***Betölti a TR-be az új TSS szelektorát és leíróját*** 👍
  - ***Beállítja a processzor regisztereit az új TSS-ből*** 👍
  - ***Elindítja az új taszk futtatását*** 👍

❖ Taszk védelem

- ***Szétválasztás szegmensekre***
  - ***Kód, adat, stack, stb.*** 👍
- ***Szegmensekre és lapokra védelmi hierarchia*** 👍
- ***Saját LDT és PDT ☞ elválasztás a többi taszktól*** 👍
- ***Hozzáférés ellenőrzés***
  - ***Privilégium, Típus, Limit, stb.*** 👍
- ***Hibák hatását taszkon belül tartja***
  - ***Privilegizált stack pointerek*** 👍
  - ***Paraméterátadás, stb.*** 👍
- ***Közös objektumok kezelése***
  - ***GDT leírókon*** 👍
  - ***Azonos célra mutató LDT – és lapleírókon keresztül*** 👍

❖ Virtuális tárkezelés

Lineáris címtér

- ❑Logikailag strukturált
- ❑Strukturálatlan

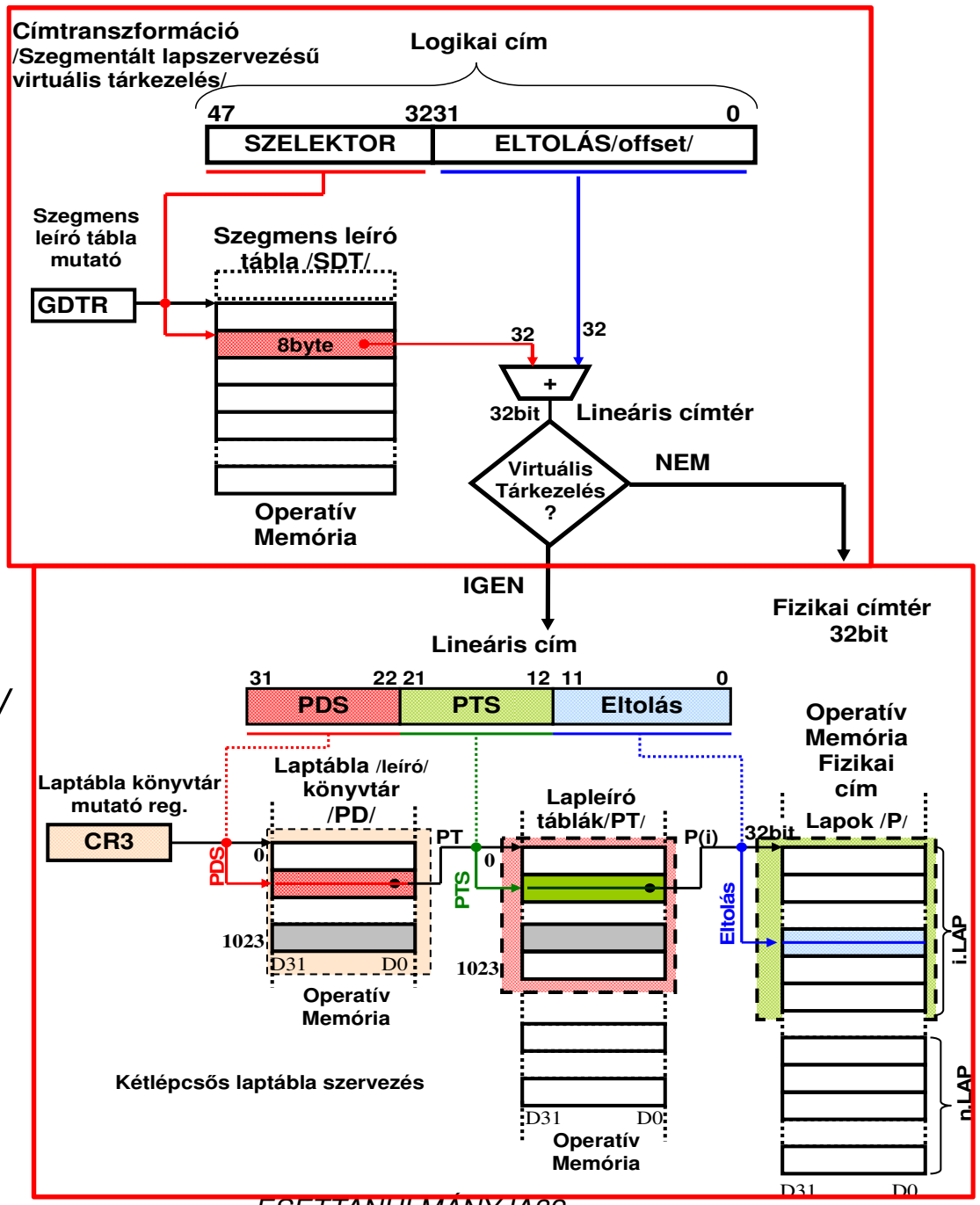
Max 4GB

Változó fizikai tároló méret →max4GB

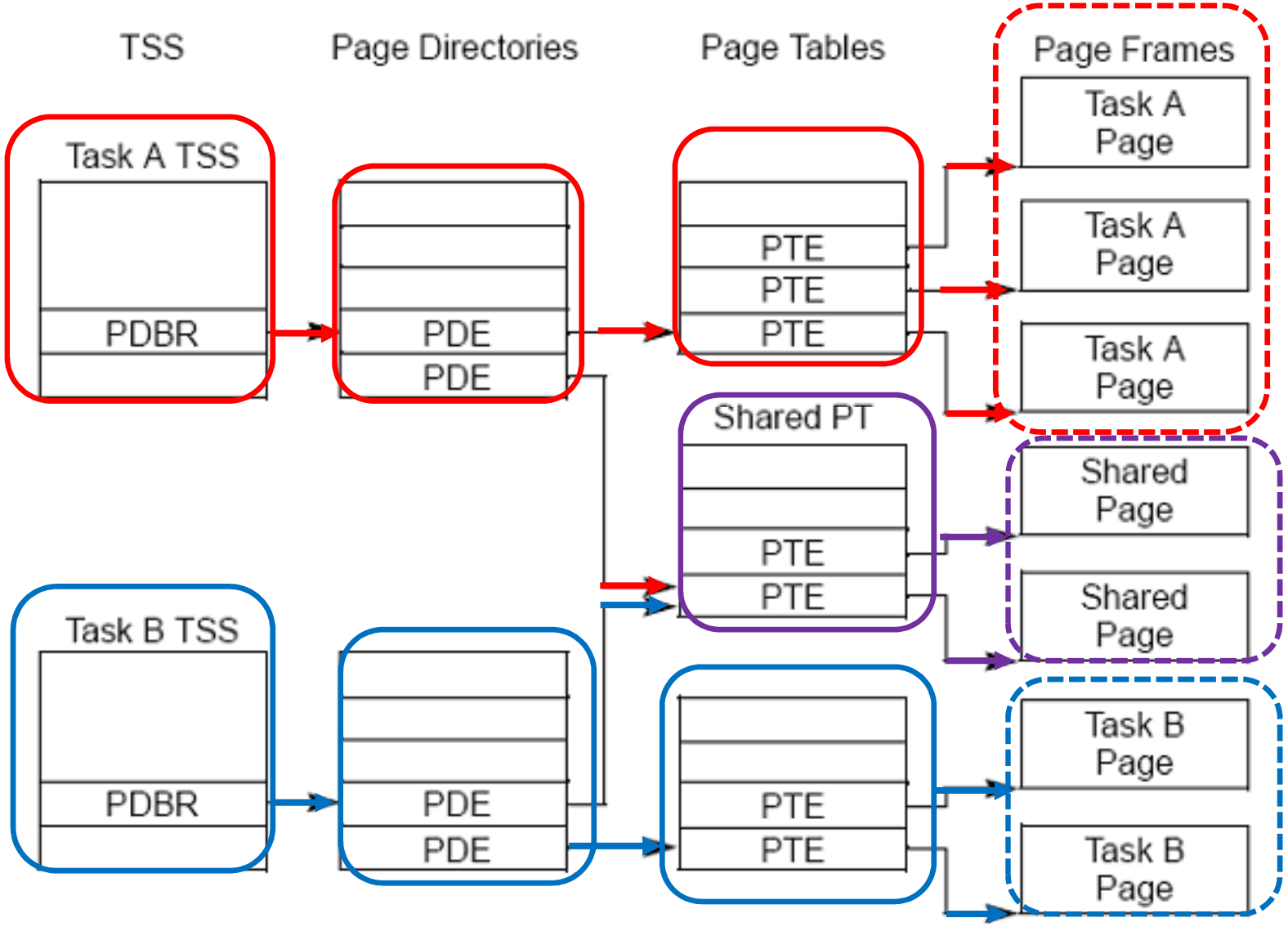
Virtuális tárkezelés/LVTK/

- ❑Lapszervezés
- ❑Ki/Be kapcsolható!!!
- ❑HW MMU
- ❑4kB-os lapok
- ❑kétszintű lapkezelés

Változó fizikai tároló méret →max4GB



➢ Példa megosztott hozzáférésű lapok használatára

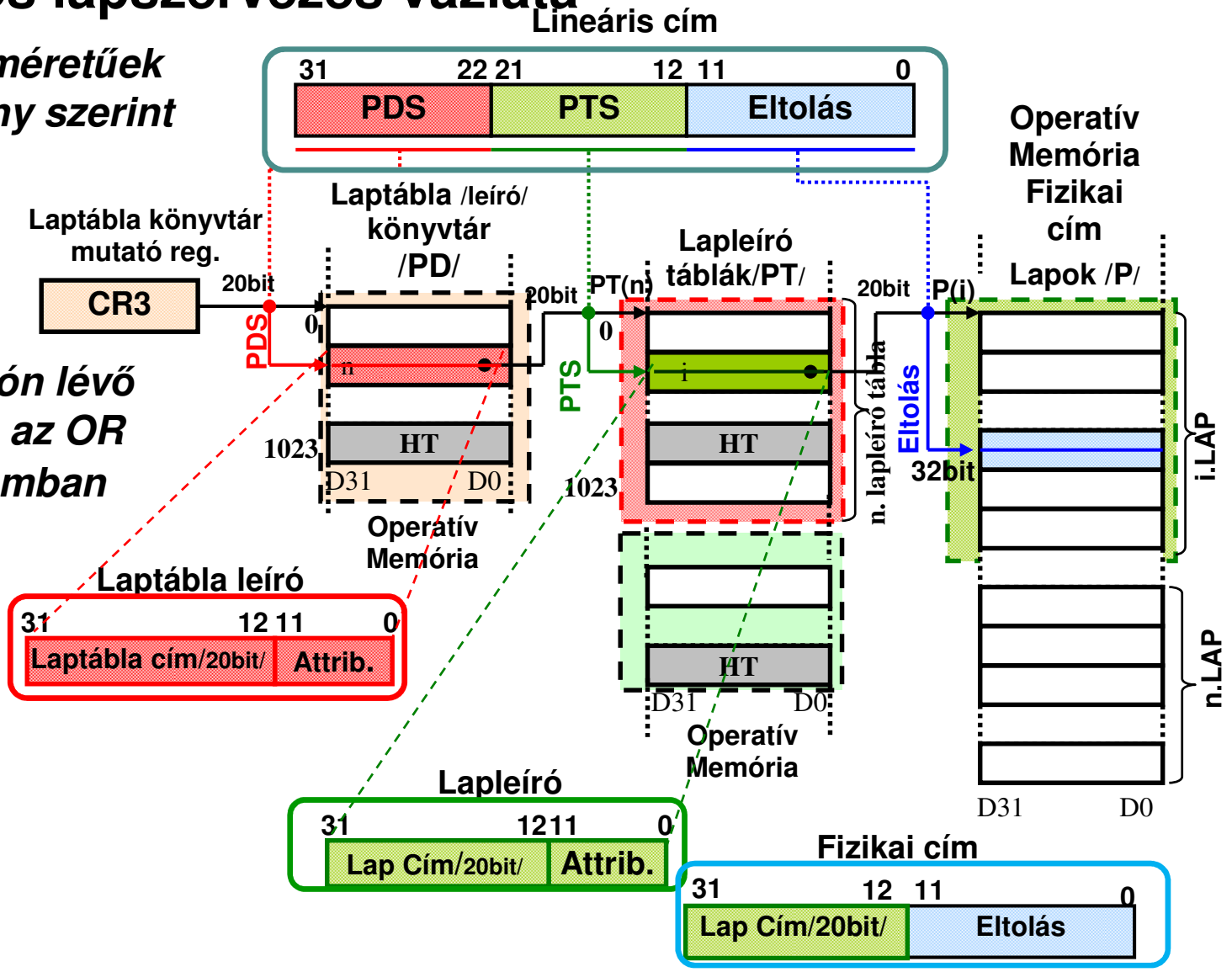


❖ i386 Lapszervezésű virtuális tárkezelés

A kétlépcsős lapszervezés vázlata

A táblák is lap méretűek
A laptáblák igény szerint betölthetők

HT a háttértárolón lévő helyet adja meg az OR szerinti formátumban



➢ Laptábla leíró */könyvtárban/*

31 — 12	11 — 9	8	7	6	5	4	3	2	1	0
PAGE TABLE ADDRESS 31..12	OS RESERVED	0	0	D	A	0	0	U — S	R — W	P

➢ Lapleíró */laptáblában/*

31 — 12	11 — 9	8	7	6	5	4	3	2	1	0
PAGE FRAME ADDRESS 31..12	OS RESERVED	0	0	D	A	0	0	U — S	R — W	P

P *Laptábla, vagy a lap bent van az Operatív memóriában /P=1/*

R/W *Írásengedélyezés /R/W=1/*

U/S *Az engedélyezett hozzáférés szintje*

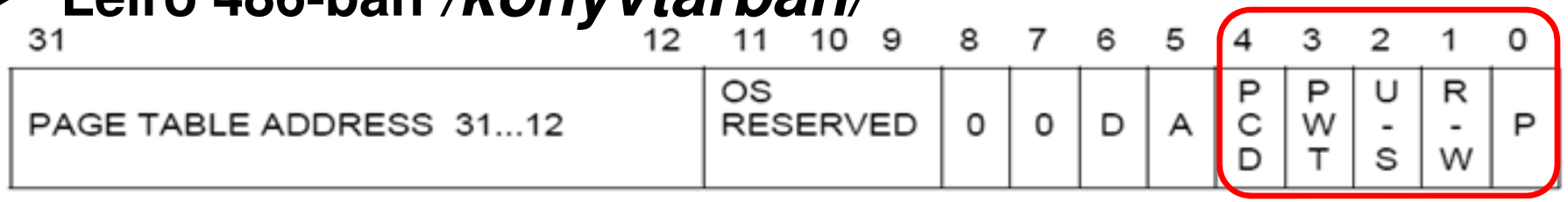
*1 esetén **felhasználói** /3. privilégium szint/*

*0 esetén **rendszer szint** /0, 1, 2 privilégium szint/* 👎👍

A *hozzáfértek /használták/ R/W esetén a processzor 1-be állítja* 👍

D *Dirty a betöltés után írtak a lapra* 👍

➢ Leíró 486-ban */könyvtárban/*



U/S	R/W	3 szint /user/ engedély	0, 1, 2 szint /system/ engedély
0	0	Nincs	Read/Write*
0	1	Nincs	Read/Write
1	0	Read only	Read/Write
1	1	Read/Write	Read/Write

***486-nál R/W=0 –nál, WP=1 /CRO-ban/ tiltja az írást rendszer módban is**

PCD Laptartalom Cache- betöltésének tiltása

PWT Write through tiltás /0 write back/ külső Cache-nek /L2/

Laponként eltérő írási stratégiátalkalmazhat

➢ Többszintű hozzáférés engedélyezés

az adott leíró U/S R/W bejegyzése szerint

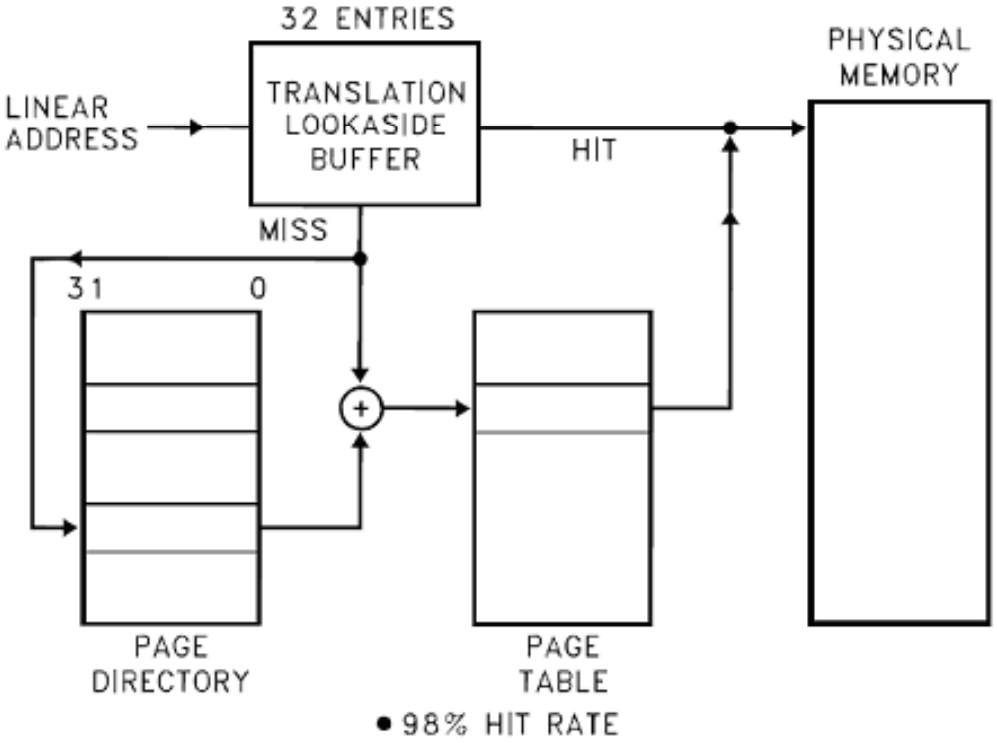
Pl.: Könyvtár 11 /user szint, írható/

Laptábla 01 /system szint, írható

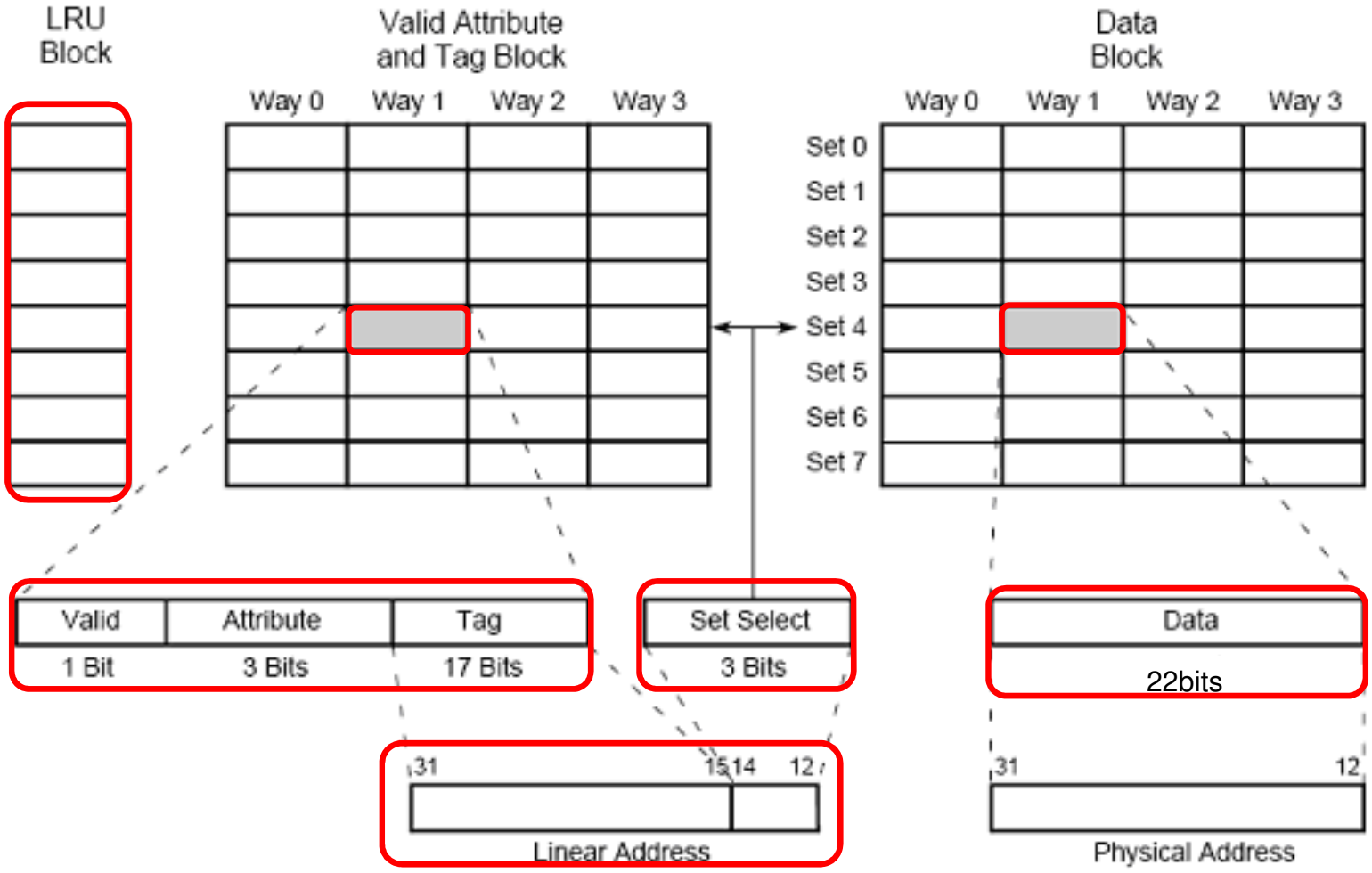
Eredmény 01 az adott laphoz csak 0,1,2 szint férhet hozzá

➢ Leírók olvasásának gyorsítása TLB-vel

- ❑ ***Négyutas szet asszociatív cache az utolsó 32 lapra***
- ❑ ***Laptábla leíró beolvasása a könyvtárból***
  - ☞ ***Ha kell locked utasítás /Test and Set/ A bit írására SW***
- ❑ ***Lapleíró beolvasása laptáblából és betöltése a TLB-be /HW/***



➢ TLB szervezése /i486/



ADAT: ***20 bit lapcím +PCD, WT***
TAG: ***V, U/S, R/W, D, 17címbit***
LRU: ***3bit pszeudo LRU***

CR3 írásakor */pl.:taszk váltáskor/* ***az egészet érvényteleníteni kell!***

A lapszervezés csökkenti a szegmentálás hátrányait

A fix lapméret jól illeszthető a HT szektor méretéhez

Elkerülhető a memória „szétesés”

Nagy objektumoknál /pl.: kód szegmens/ a fizikai tárban nem kell folytonos tárterület, nem is kell az egész szegmensnek egyszerre bent lenni az OM-ben

Láthatatlan a felhasználó számára

Probléma:

Így is előfordulhat memória töredezettség

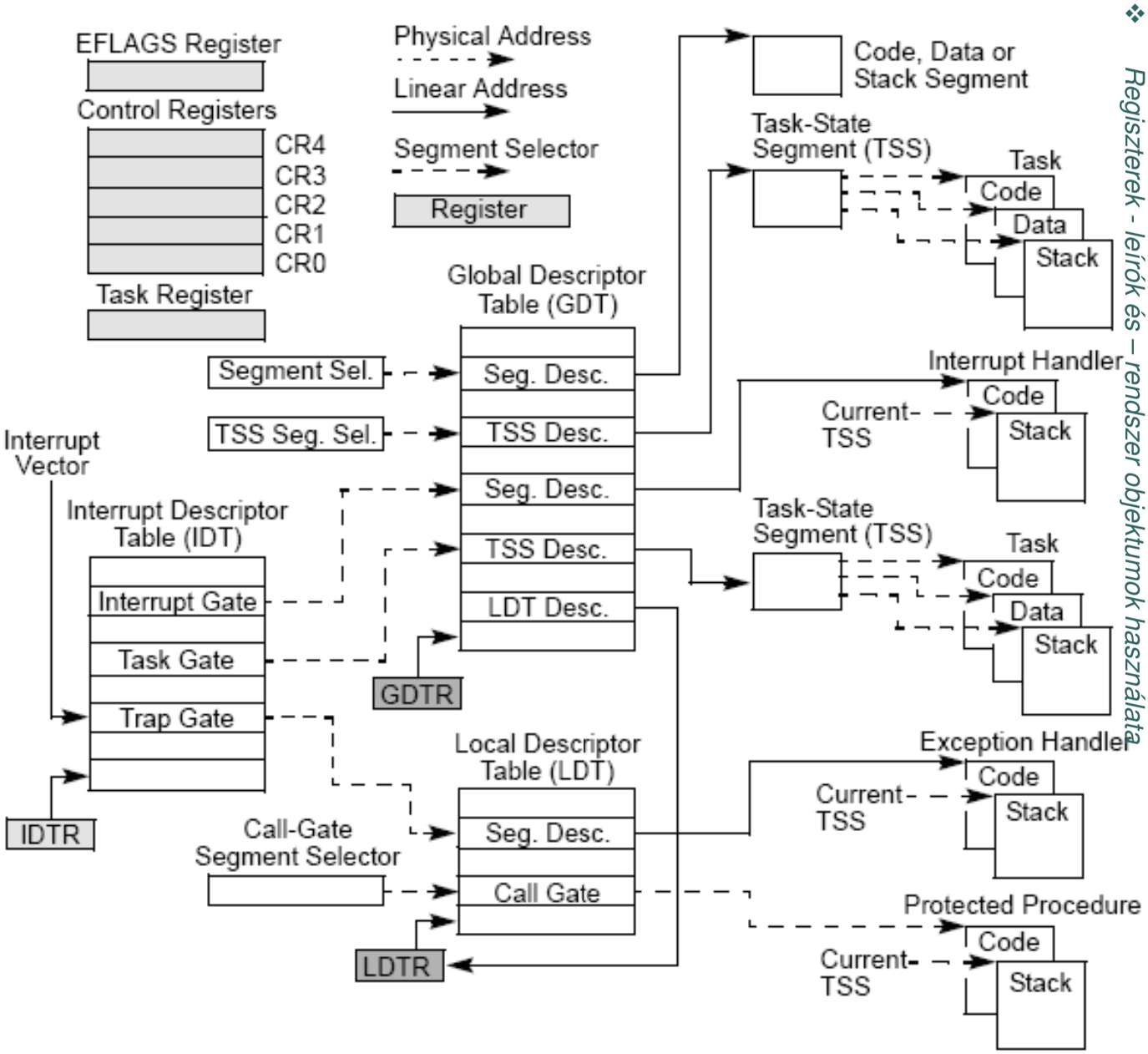
Internal fragmentation **pl.:4KB van, 5000 byte kell**

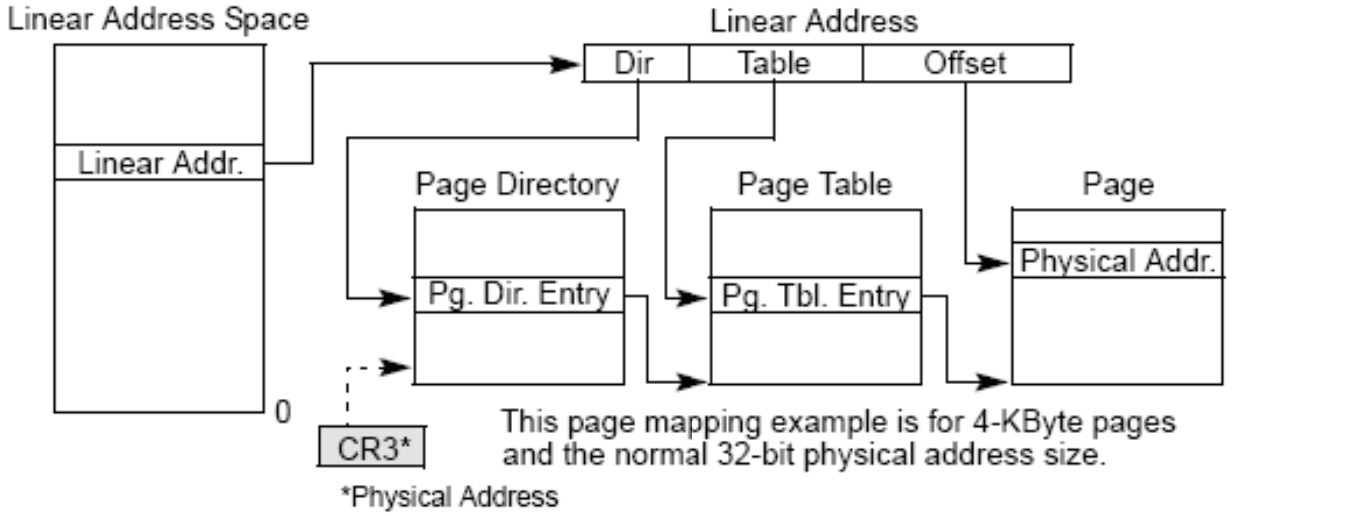
Nagyobb szoftver kiszolgálás kell

Hardver igény

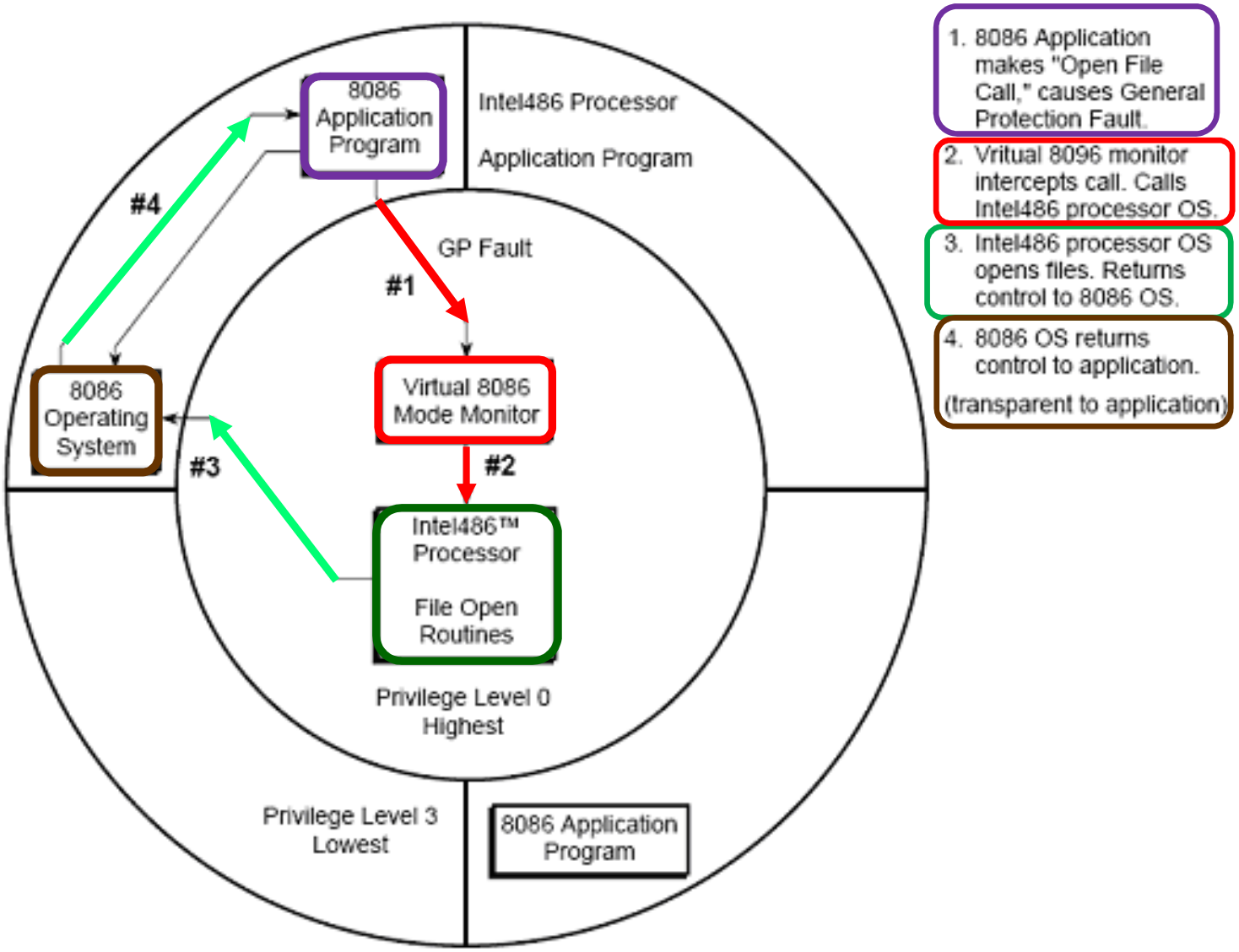
❖ Szegmentált lapszervezés - *védelem*

- **Típus ellenőrzés** */pl.: CS nem lehet írható, SS csak írható/olvasható lehet, CS nem mutathat adat - vagy stack szegmensre, stb./*
- **Hozzáférési jogosultság ellenőrzése** */Privilégium szintek, az adatok programok és rendszerobjektumok elérésének korlátozása CPL, DPL, RPL/*
- **Limit ellenőrzés**
- **Vezérlés átadás korlátozása** */kapuk alkalmazása/*
- **Utasításhasználat korlátozása** */privilegizált utasítások pl.: LGDT, LLDT, stb.; csak 0 szintnél használhatók*
  - ☞ *I/O utasítások korlátozása*
    - ☞ *I/OPL, I/O bit map*
- **Kétszintű U/S, R/W laptábla/lap védelem**
- **Mi lenne csak lapszervezés esetén** 👎👍

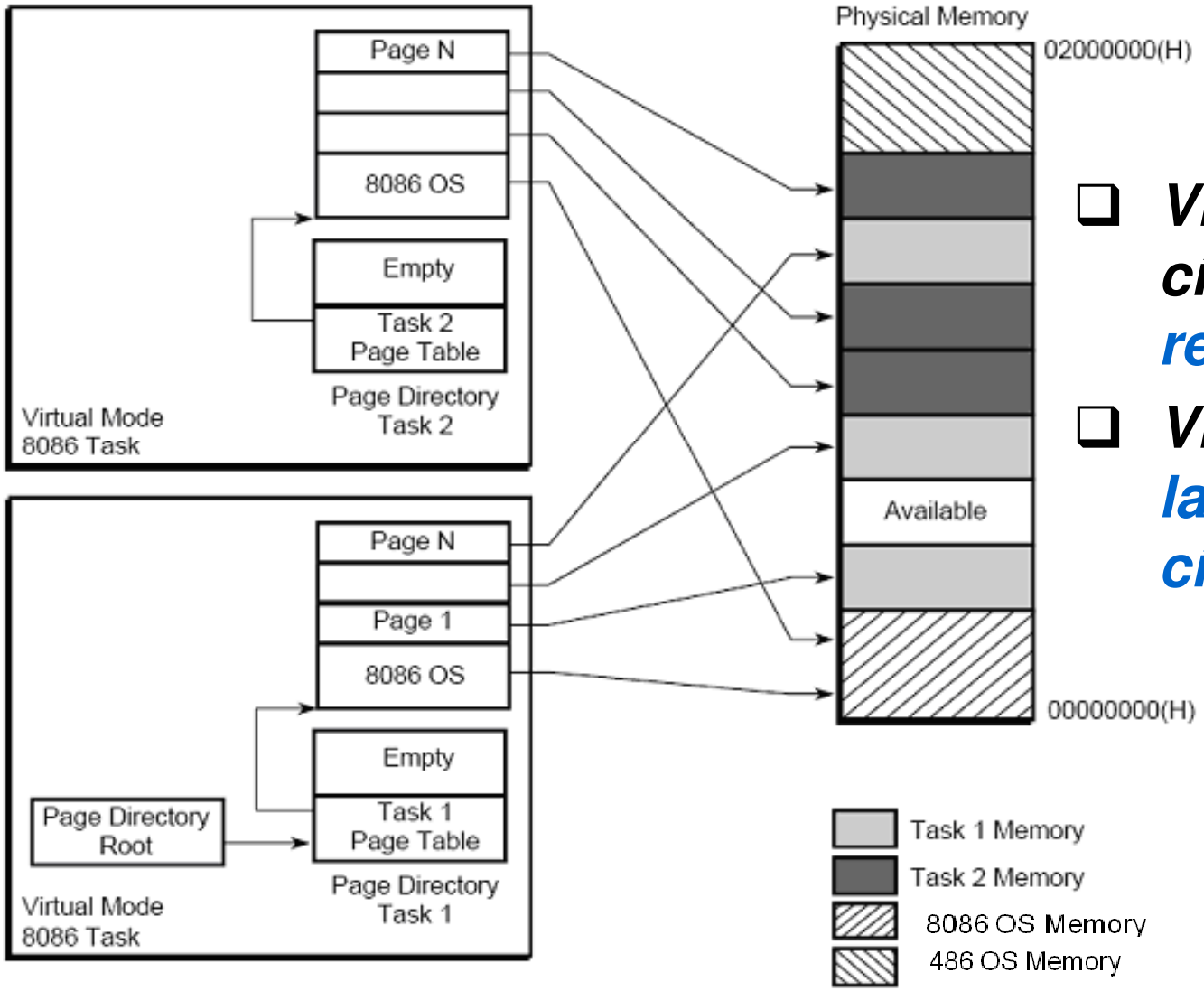

EFLAGS Register
Control Registers
CR4
CR3
CR2
CR1
CR0
Task Register
Physical Address
Linear Address
Segment Selector
Register
Code, Data or Stack Segment
Task-State Segment (TSS)
Task
Code
Data
Stack
Global Descriptor Table (GDT)
Segment Sel.
Seg. Desc.
TSS Seg. Sel.
TSS Desc.
Seg. Desc.
TSS Desc.
LDT Desc.
Interrupt Handler
Code
Current-TSS
Stack
Interrupt Vector
Interrupt Descriptor Table (IDT)
Interrupt Gate
Task Gate
Trap Gate
GDTR
IDTR
Task-State Segment (TSS)
Task
Code
Data
Stack
Local Descriptor Table (LDT)
Call-Gate Segment Selector
Seg. Desc.
Call Gate
LDTR
Exception Handler
Code
Current-TSS
Stack
Protected Procedure
Code
Current-TSS
Stack


Linear Address Space
Linear Addr.
0
Linear Address
Dir
Table
Offset
Page Directory
Pg. Dir. Entry
Page Table
Pg. Tbl. Entry
Page
Physical Addr.
CR3*
This page mapping example is for 4-KByte pages and the normal 32-bit physical address size.
*Physical Address

❖ Virtuális 86 üzemmód

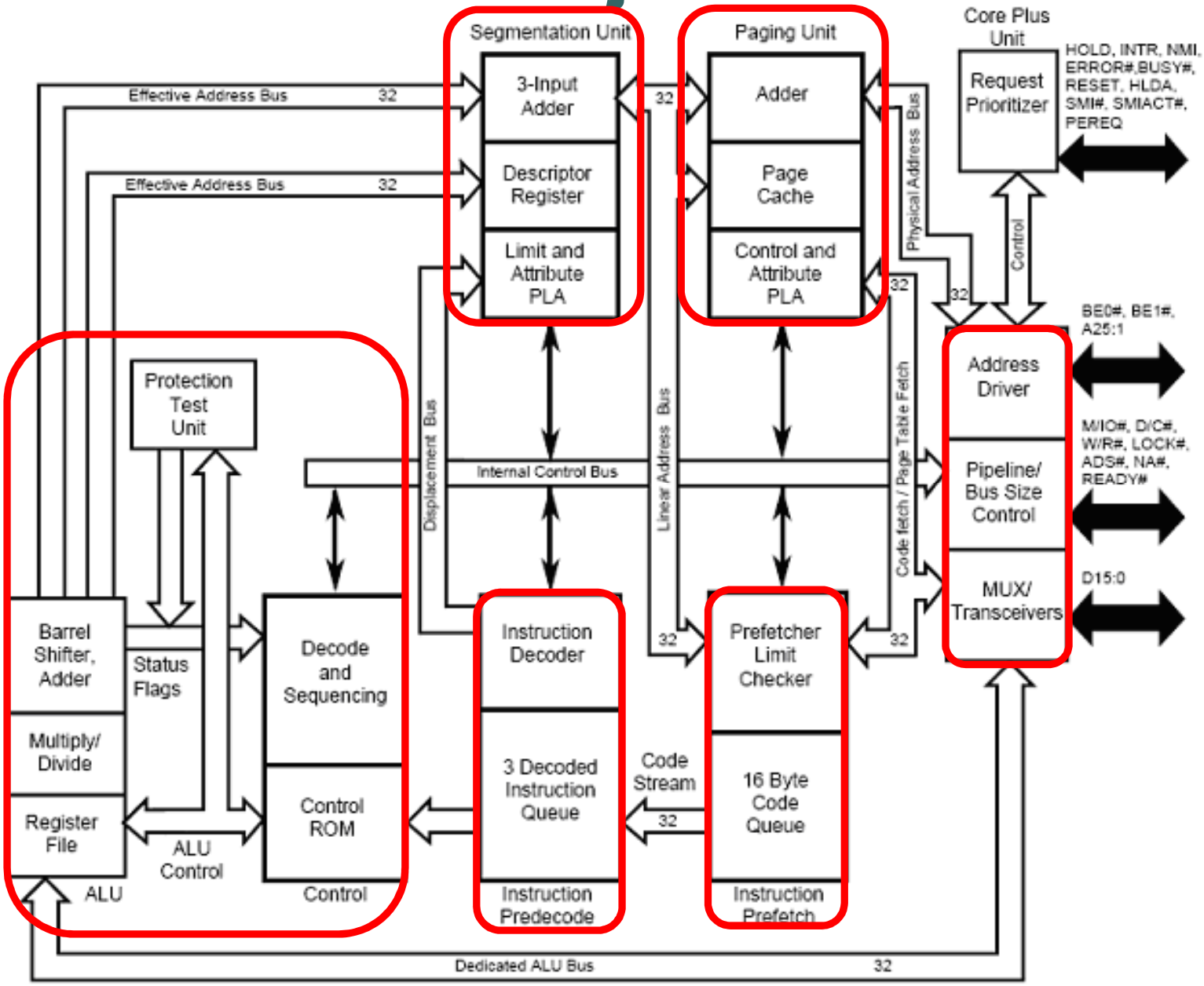


➢ *8086 virtuális módú taszkok futtatása*



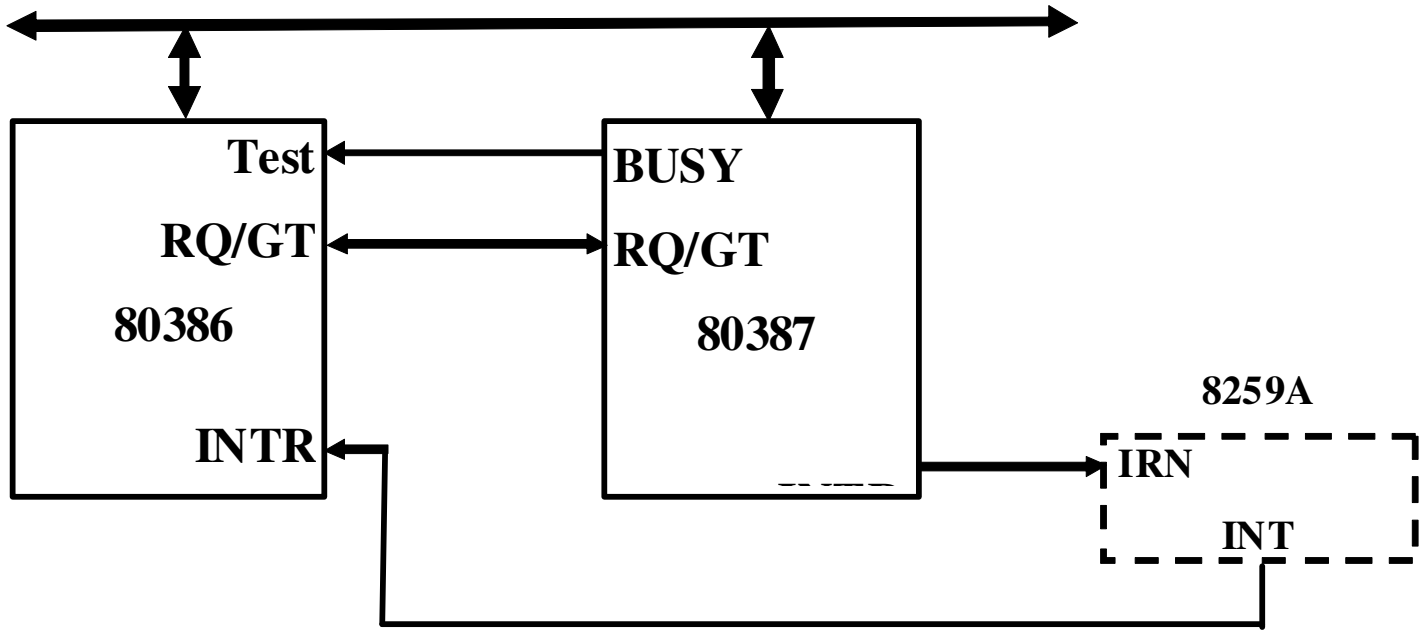
- ***VM86 módban címszámítás mint real módban***
- ***VM86 módban 256 lap a 4 GB címtartományban***

❖ i386 belső struktúrája

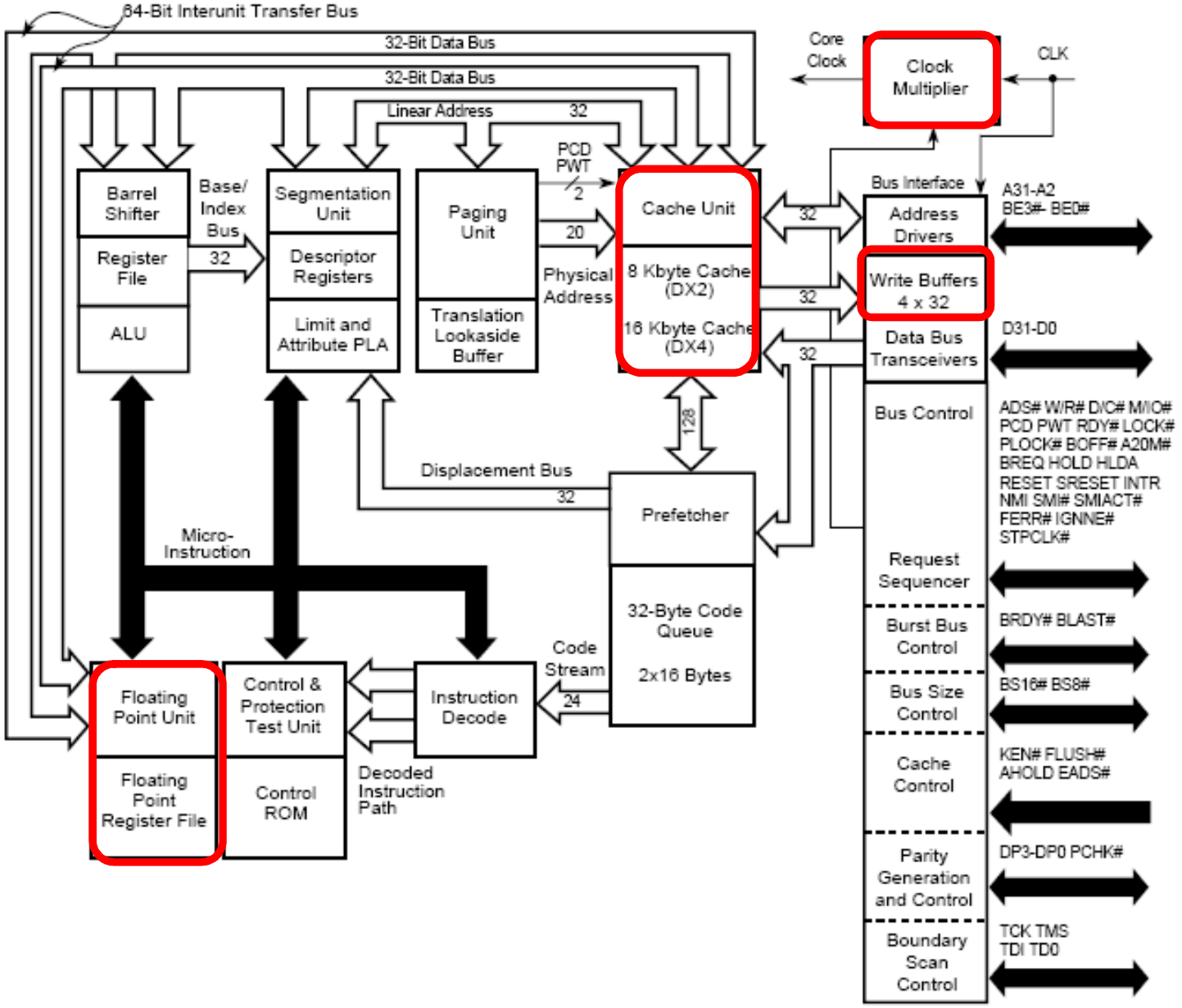


❖ Matematikai társprocesszor *i387-ix87*

- ❑ Lebegőpontos aritmetika
- ❑ Az ehhez rendelt utasításoknál *átveszi az utasítás végrehajtást* */ha kell, a memóriában lévő operandus kezdőcímével/*
- ❑ Belső FILO */STACK/* *szervezésű 80 bites* Regiszterek */R1-R8/*
- ❑ Átlapolt működés
- ❑ A processzor *WAIT* utasítással várhatja be az eredményt

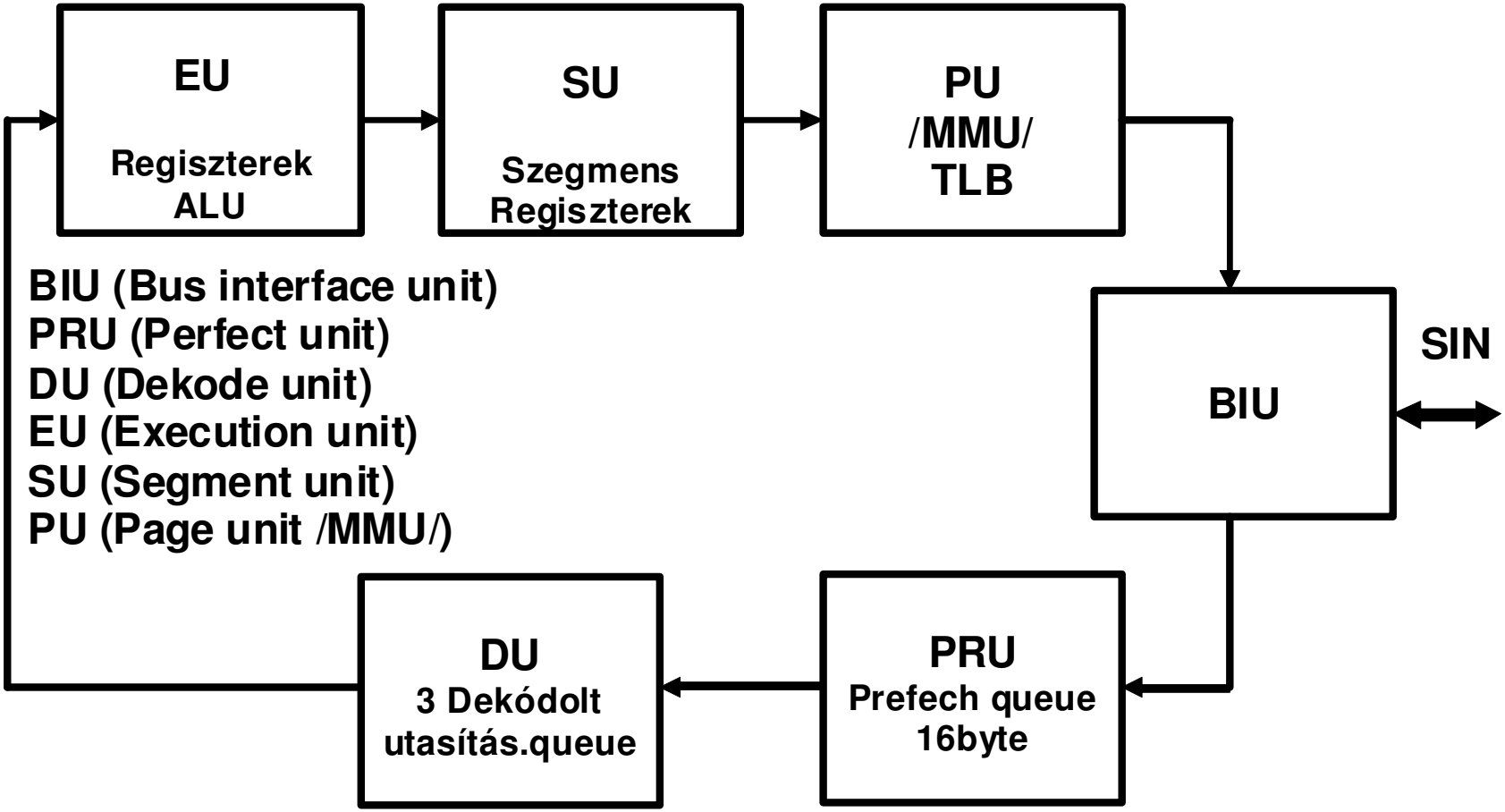


❖ i486 belső struktúrája



❖ Utasítás feldolgozó szalag

➢ i386 pipe-line vázlata



- **A szalag működését támogatja**
  - **Utasítás behozatal várósor** 👍
    /a sín nem érhető el mindig periodikusan/ 👎
  - **Dekódolt utasítás váró sor** 👍
    /az utasítások végrehajtási ideje eltérő/ 👎
- **A feldolgozó szalag hatása az utasítás végrehajtásra**
  - **utasítás egymásra hatás kezelése** ***/pl.: feltételes ugrás/***
    - ***Új utasítás lehívását felfüggeszti a feltétel kiértékeléséig /386/*** 👎👍
    - ***Előre eldöntött ágon folytatja az előfeldolgozást/486/***
- **Címzési mód miatti hatások**
  - **Operandus *lehívás/kivitel kell/nem kell***
  - **Címzési módtól függő *címszámítás***
  - **Van/nincs *virtuális tárkezelés***