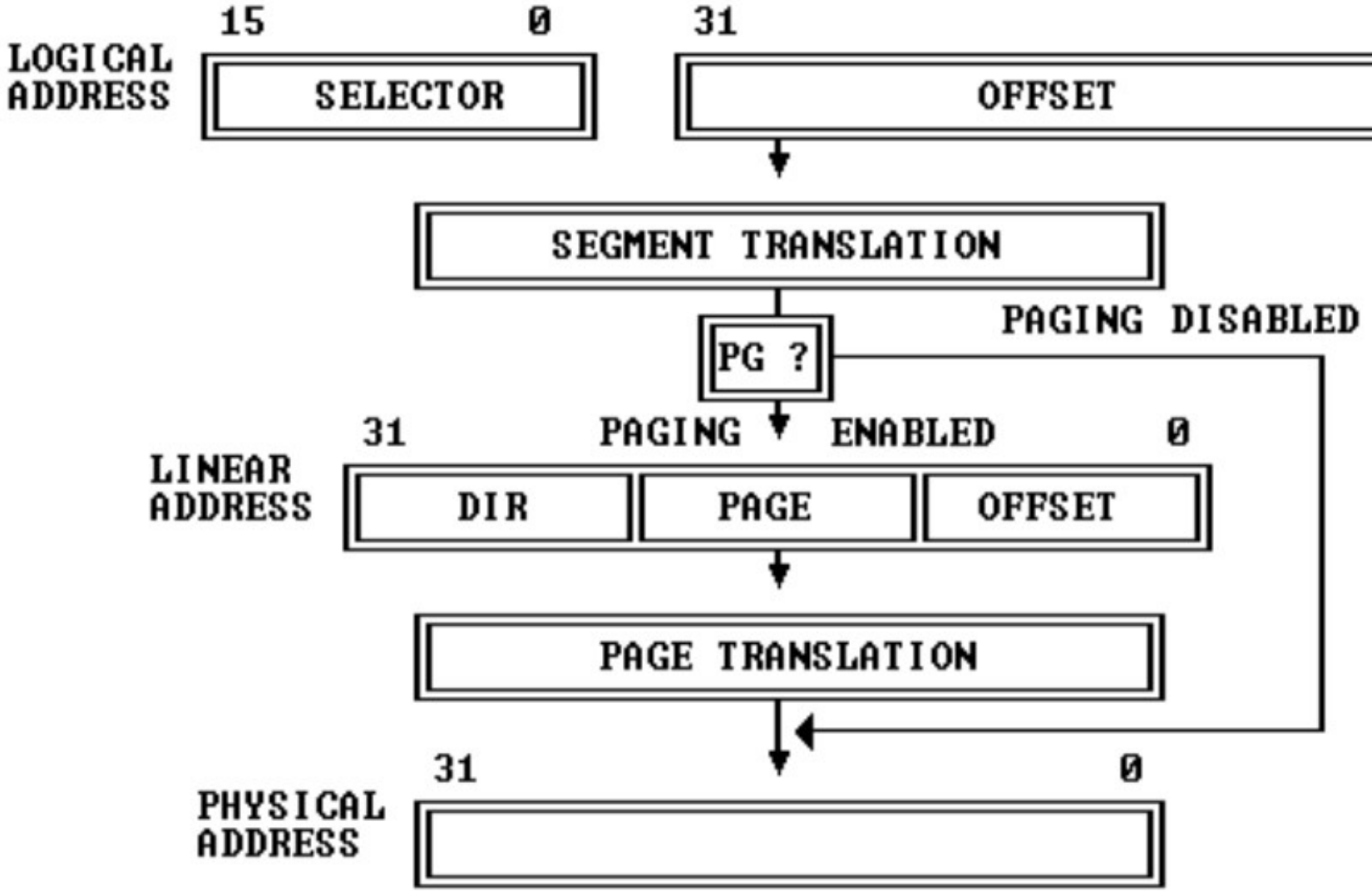
OS 2018 v. 04

MIT ;)

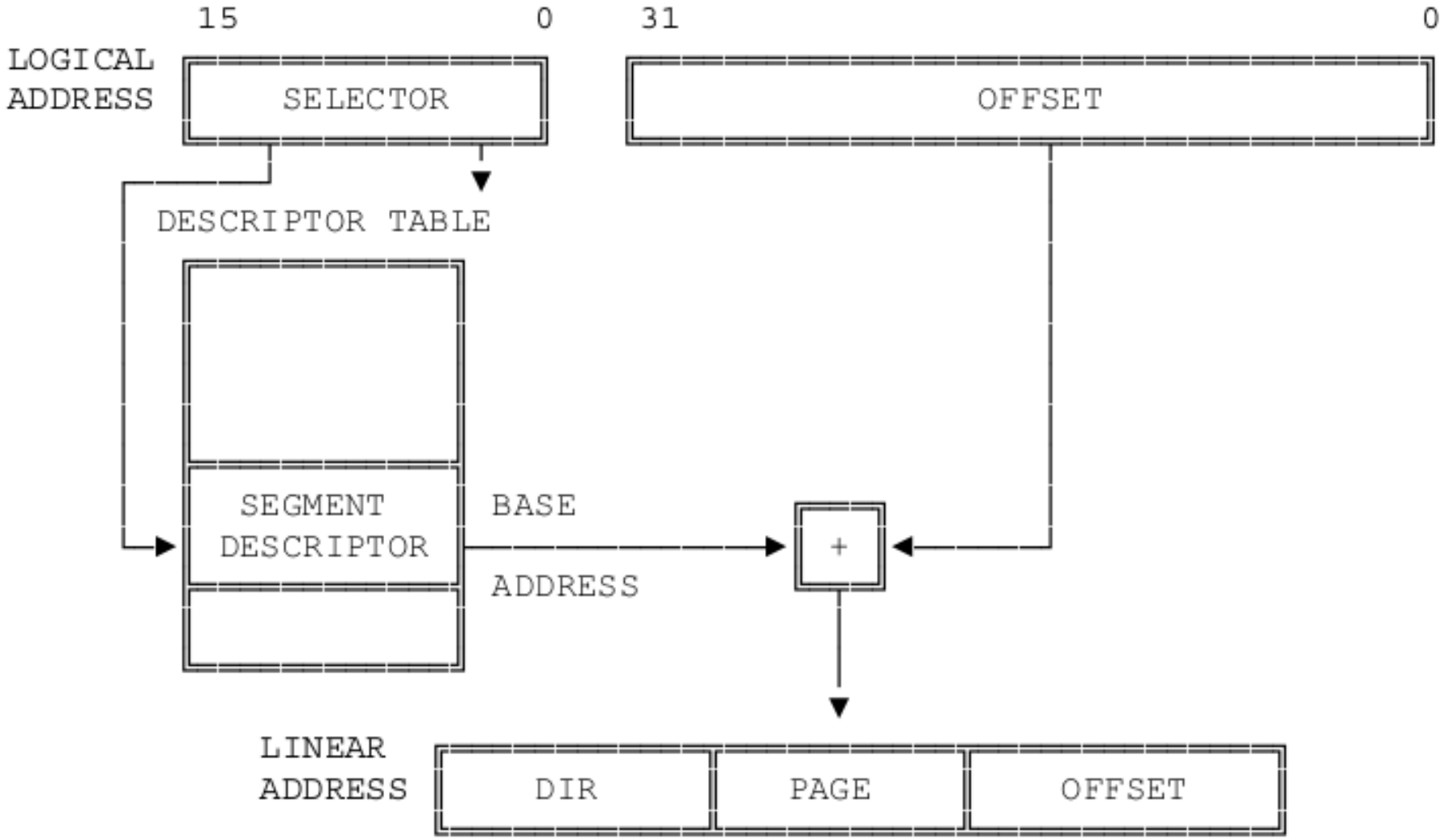
https://pdos.csail.mit.edu/6.828/2018

1. SEGMENTACIA a STRANKOVANIE

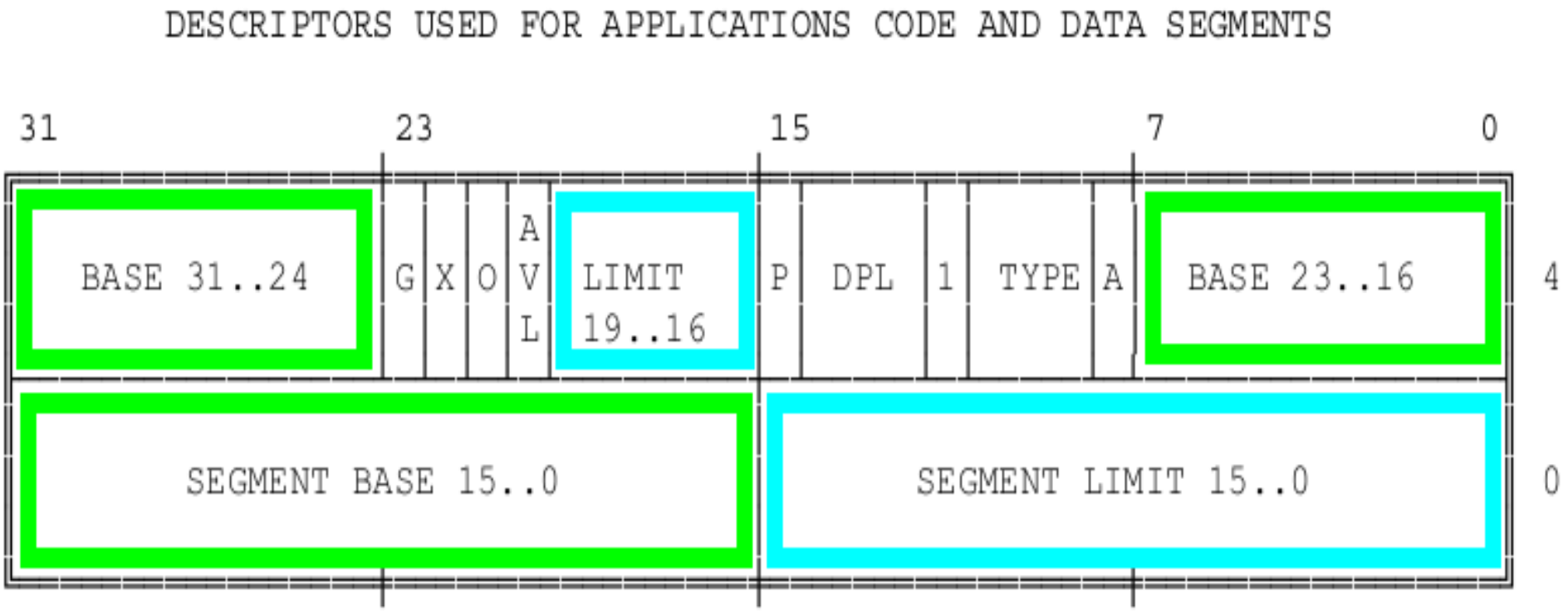
Preklad logickej adresy na fyzicku



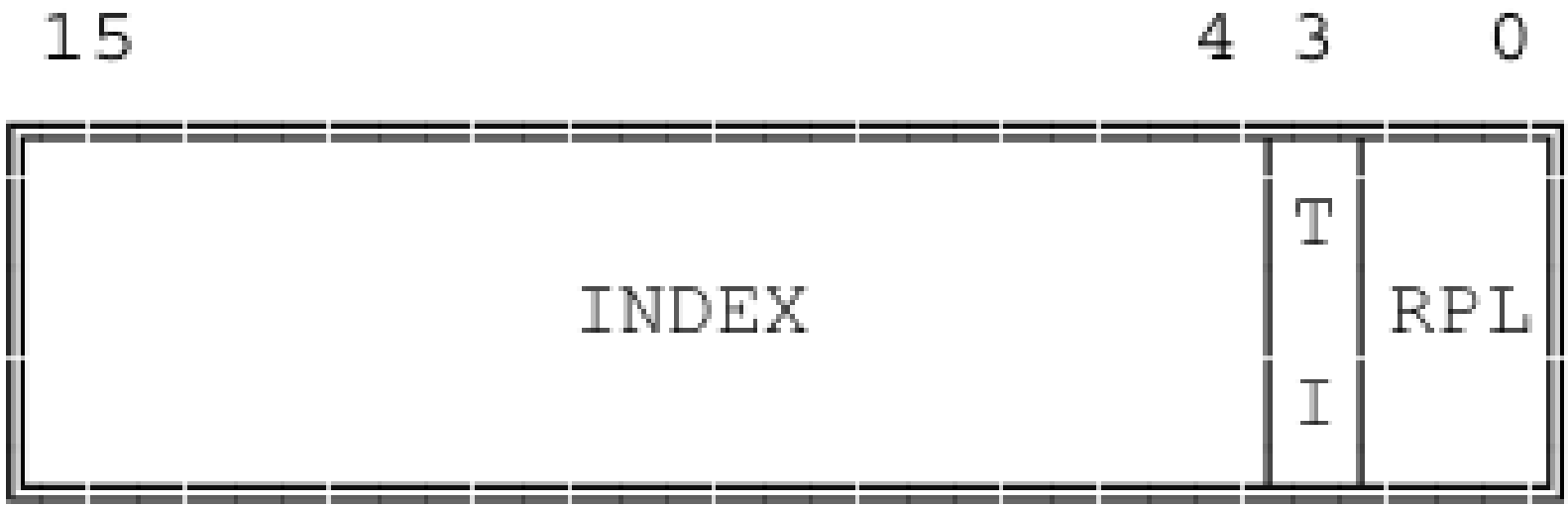
Segmentacia



Deskriptor segmentu

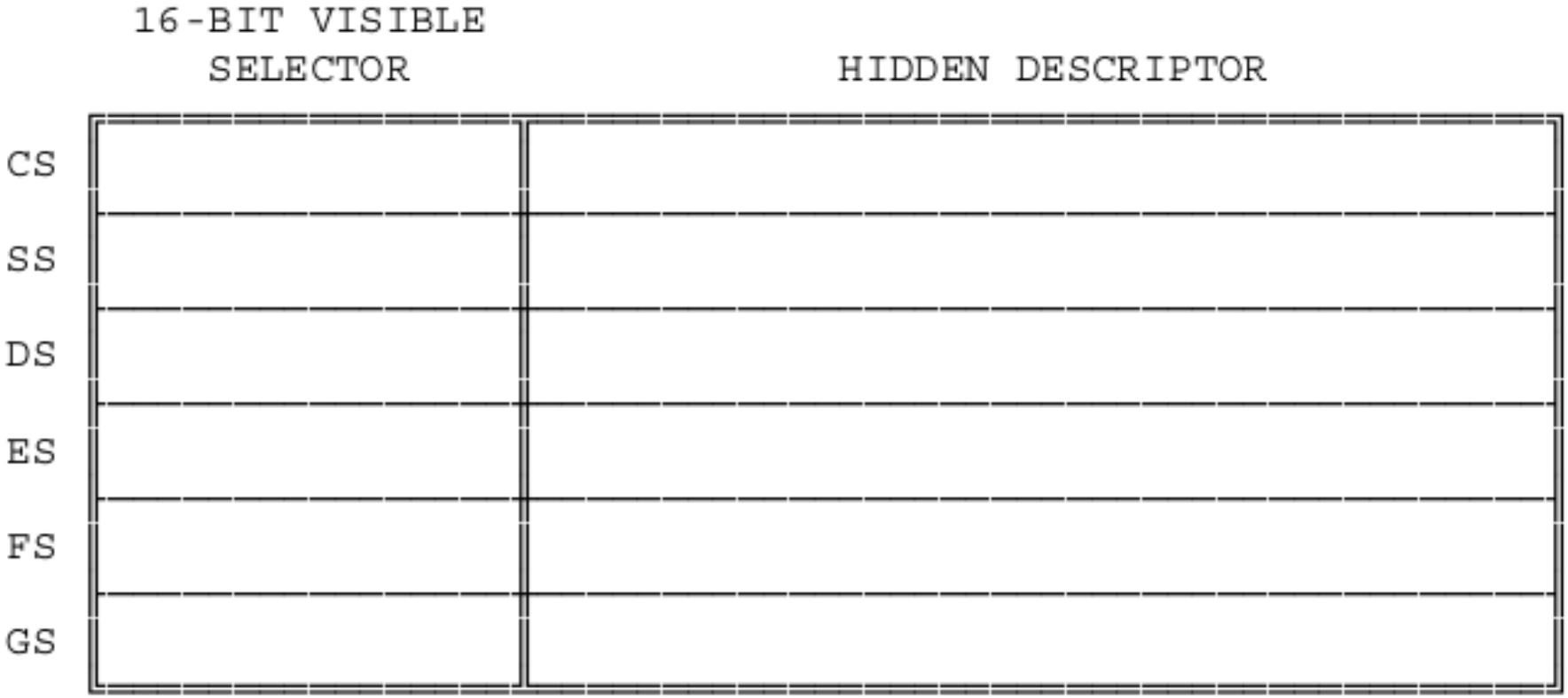


Selektor



TI - TABLE INDICATOR
RPL - REQUESTOR'S PRIVILEGE LEVEL

Segmentove registre



Objasnenie asm ver. 1

- **boot/boot.S**
  - **lgdt, cr0**
  - **cs, ds, es, fs, gs, ss**
- kern/entry.S
  - cr3, RELOC(), cr0
  - bootstack, bootstacktop
- kern/entrypgdir.c
  - entry_pgdir 2 polozky!!!, entry_pgtable

Vysledok segmentacie

VirtA (LogA) --->
Linearna Adresa

Strankovanie

- Ramec (page frame) versus stranka (page)
- Linearna adresa
- Tabulky stranok
- Polozka tabulky stranok
- Preklad

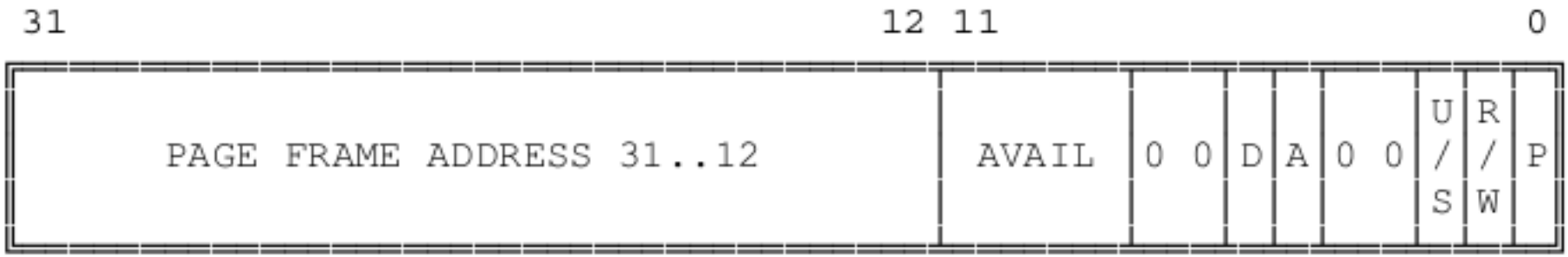
Princip

- Jedna velka tabulka stranok
- Kolko ma poloziek?
- Kolko miesta zabera v pamati?

- Dvoj urovnove strankovanie
- Troj urovnove, stvor urovnove

Format linearnej adresy

Polozka tabulky stranok



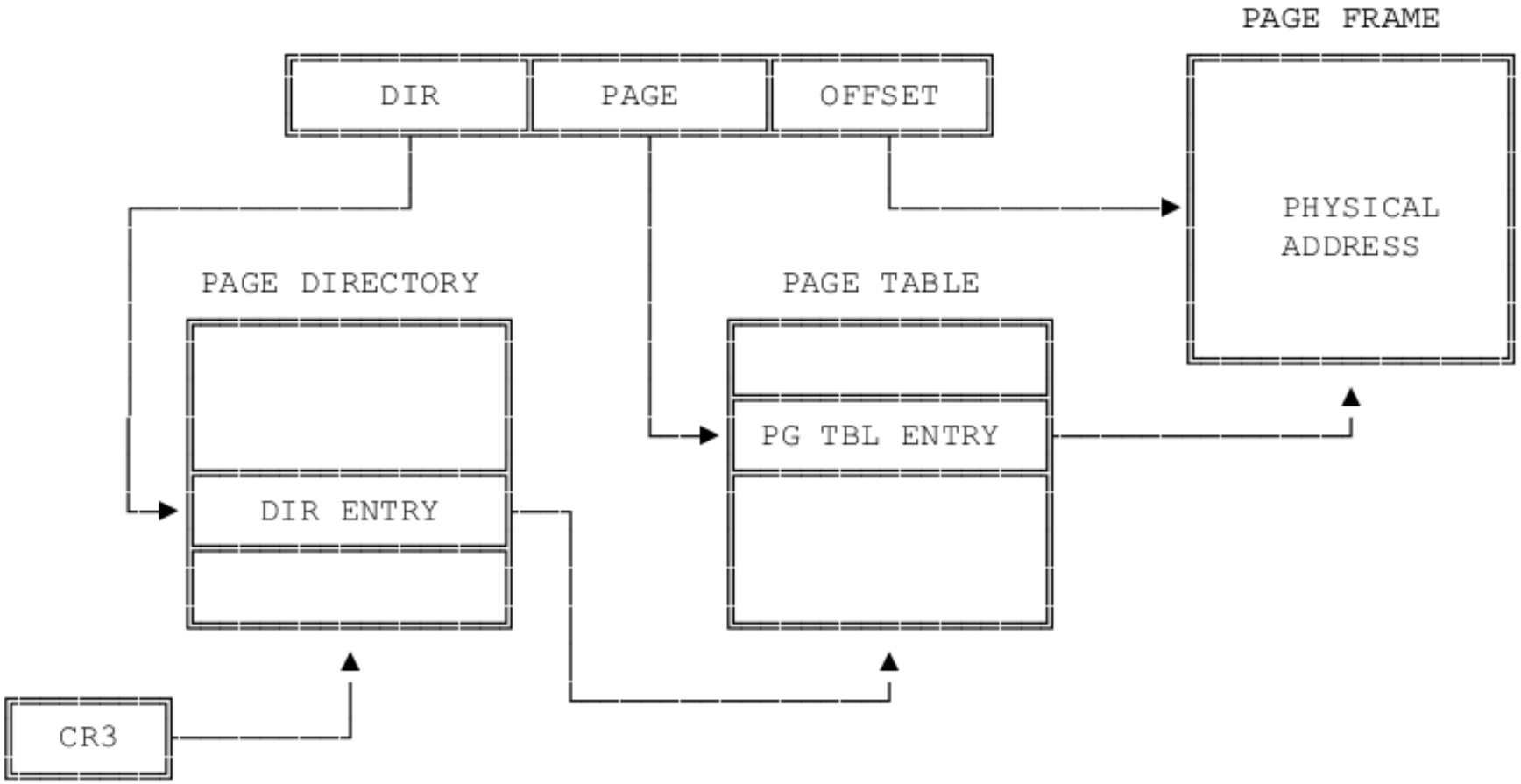
P - PRESENT
R/W - READ/WRITE
U/S - USER/SUPERVISOR
D - DIRTY
AVAIL - AVAILABLE FOR SYSTEMS PROGRAMMER USE

NOTE: 0 INDICATES INTEL RESERVED. DO NOT DEFINE.

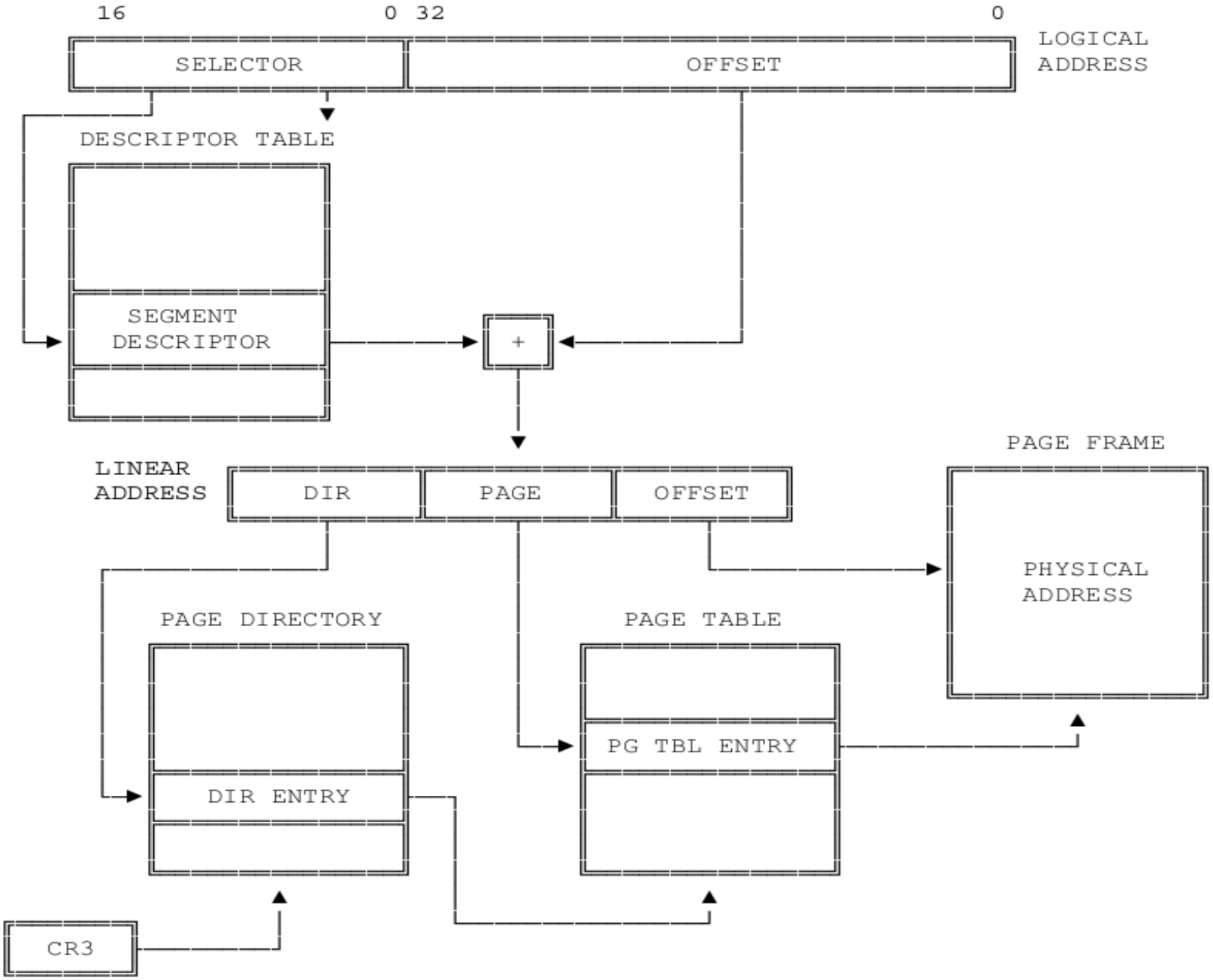
Kontrola opravneni

Page Directory Entry		Page Table Entry		Combined Protection	
U/S	R/W	U/S	R/W	U/S	R/W
S-0	R-0	S-0	R-0	S	x
S-0	R-0	S-0	W-1	S	x
S-0	R-0	U-1	R-0	S	x
S-0	R-0	U-1	W-1	S	x
S-0	W-1	S-0	R-0	S	x
S-0	W-1	S-0	W-1	S	x
S-0	W-1	U-1	R-0	S	x
S-0	W-1	U-1	W-1	S	x
U-1	R-0	S-0	R-0	S	x
U-1	R-0	S-0	W-1	S	x
U-1	R-0	U-1	R-0	U	R
U-1	R-0	U-1	W-1	U	R
U-1	W-1	S-0	R-0	S	x
U-1	W-1	S-0	W-1	S	x
U-1	W-1	U-1	R-0	U	R
U-1	W-1	U-1	W-1	U	W

Preklad LinA-->FyzA



Seg + Pag

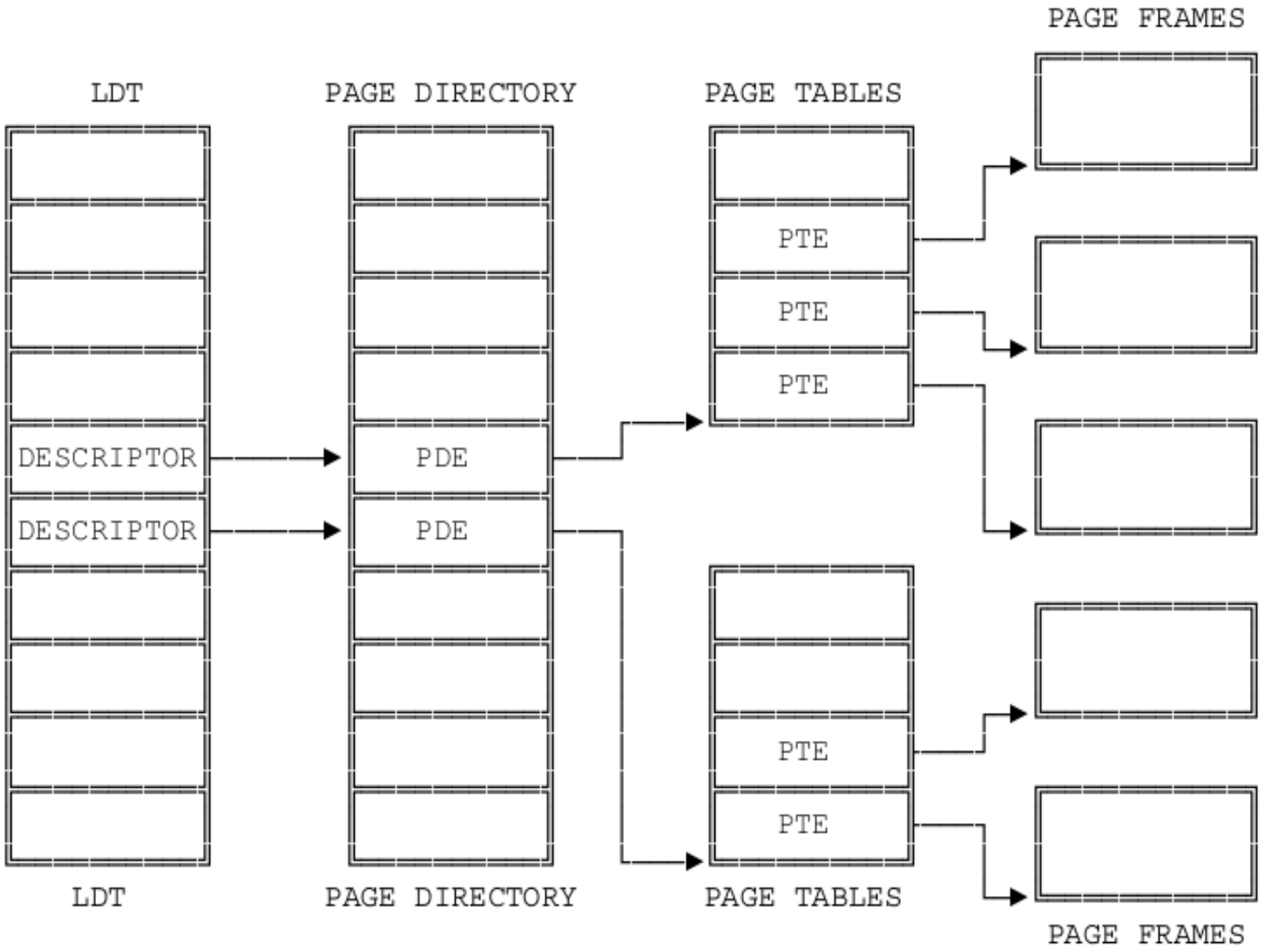

16
0 32
0
SELECTOR
OFFSET
LOGICAL ADDRESS
DESCRIPTOR TABLE
SEGMENT DESCRIPTOR
+
LINEAR ADDRESS
DIR
PAGE
OFFSET
PAGE FRAME
PHYSICAL ADDRESS
PAGE DIRECTORY
DIR ENTRY
PAGE TABLE
PG TBL ENTRY
CR3

Kombinacie Seg + Pag

- FLAT model (“vypnuta” segmentacia)
- Jeden segment cez viac stranok
- Jedna stranka obsahujuca viac segmentov (napr. semafory)
- 1 tabulka stranok na 1 segment (t.j. 1024 stranok per seg)

- Licujuce hranice segmentov a stranok
- Nerovnake hranice segmentov a stranok

Deskriptor per stranka



2. ADRESACIA V JADRE JOS

Objasnenie asm ver. 2

- boot/boot.S
  - lgdt, cr0
  - cs, ds, es, fs, gs, ss
- **kern/entry.S**
  - **cr3, RELOC(), cr0**
  - **bootstack, bootstacktop**
- **kern/entrypgdir.c**
  - **entry_pgdir 2 polozky!!!, entry_pgtable**

Alokacia fyzickych stranok (cca 30r, 40m)

- boot_alloc() cca 8 riadkov, 15 min
- mem_init() cca 3 riadky, 2 min
- page_init() cca 5 riadkov, 10 min
- page_alloc() cca 8 riadkov, 10 min
- page_free() cca 5 riadkov, 3 min

- Pocet stranok RAM – premenna 'npages'
- Velkost stranky – 4096 bajtov – konstanta 'PGSIZE'

PADDR(x) versus KADDR(x)

- uintptr_t – ukazatel jadra (adresy zacinajuce 0xF...)
- physaddr_t – fyzicka adresa v RAM

- PADDR(x): uintptr_t → physaddr_t
- (napr. 0xF010BABA → 0x0010BABA)

- KADDR(x): physaddr_t → uintptr_t
- (napr. 0x0030DEAD → 0xF030DEAD)

boot_alloc()

- Ukazatel na prvy volny bajt ZA jadrom
- Praca v mode strankovania, volna pamat (i pouzita) zaokruhlovana na stranky!
- 3 stavy:
  - Nedostatok pamate! (ako zistime???) (vieme velkost RAM a mame ukazatel na prve volne miesto spolu s pozadovanou velkostou... ale... to nestaci! Uvazuj mapovanie!!!)
  - Vstupny paramater == 0 !!!!!!!!
  - Vstupny parameter < 0 ?????
- Pomocne makra: PADDR, KADDR

mem_init()

- Vyuzit boot_alloc()
- Alokacia pola struktur PageInfo, zaciatok do 'pages'

- Praca podla pokynov vo funkcii!!
- Citat komentare!!!

page_init()

- Inicializuje ZOZNAM volnych ramcov, nie samotne ramce v pamati!

- <IOPHYSMEM; EXTPHYSMEM) nesmu byt v zozname
- <EXTPHYSMEM; first_free_page) nesmu byt v zozname
  - Co je od EXTPHYSMEM v ramke?
  - Pokial siaha kernel?
  - Ako zistime prvu volnu pamat 'nad' jadrom?

- Pomocne makra PADDR, KADDR, PGNUM...

page_alloc()

- Pracuje so zoznamom 'page_free_list'
- Ak je prazdny, vraciame NULL
- Inak z neho 'odoberieme' prvy prvok (opakovanie prace so zretazenym zoznamom... - odoberame prvok zo ZACIATKU zoznamu)
- Ak je nastaveny ALLOC_ZERO, pouzijeme memset()
  - Ako zistime adresu, kde nam zacina stranka?
  - Vid pmap.h (page2pa, pa2page, page2kva)
- Pozor na spravnu inicializaciu poloziek struktury PageInfo

page_free()

- Ak sa snazi niekto uvolnit stranku, ktora sa pouziva (polozka pp_link a/alebo pp_ref struktury PageInfo su nenulove), panic!

- Pridaj stranku do zoznamu volnych stranok
- Pridavame na ZACIATOK

!!! DOMACA ULOHA !!!

Kapitola 5 Memory Management z knizky Intel 80386 Programmer's Reference Manual 1986