11. přednáška

27. listopadu 2008
14:24

Úrovně oprávnění (Privilege Levels)

- **Úroveň 0** - *jádro operačního systému* (řízení procesoru, I/O operací)
- **Úroveň 1** - *služby poskytované operačním systémem* (plánování procesů, organizace I/O, přidělování prostředků)
- **Úroveň 2** - *systémové programy a podprogramy z knihoven* (systémy obsluhy souborů, správa knihoven)
- **Úroveň 3** - *uživatelské aplikace*

DPL (Descriptor Privilege Level) - uložen ve **dvou bitech slabiky přístupová práva** popisovače segmentu, obsahuje úroveň oprávnění přidělenou obsahu segmentu.
CPL (Current Privilege Level) - zapsán ve **dvou nejnižších bitech selektoru CS** (RPL), představuje momentální úroveň oprávnění přidělenou právě prováděnému procesu
RPL (Request Privilege Level) - uložen v **bitech 0 a 1 selektoru segmentového registru**, obsahuje úroveň oprávnění, kterou proces nabízí při přístupu k určitému segmentu
EPL (Effective Privilege Level) - numerické maximum CPL a RPL (hodnota nižší úrovně zabezpečení)

Zpřístupnění datového registru

- `MOV DS, AX` - naplnění a kontrola přístupových práv
- `MOV DL, DS:adresa` - čtení datového segmentu
- `MOV DS:adresa,DL` - zápis datového segmentu (pokud `W = 1`)

$$\frac{CPL \leq DPL \quad RPL \leq DPL}{Max(CPL,RPL) \leq DPL}$$

$$EPL \leq DPL$$

Zpřístupnění datového registru

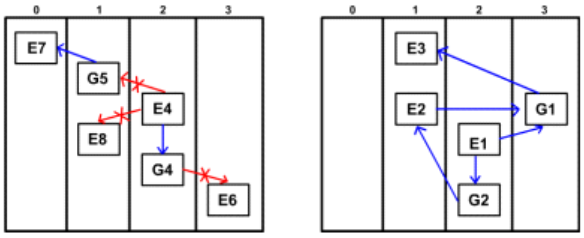
- Proces přistupuje k datům **pouze na stejné nebo nižší úrovni oprávnění**.

Předání instrukcí do instrukčního segmentu

- `JMP FAR PTR Navesti` – Skok do jiného instrukčního segmentu
- `CALL FAR PTR Navesti` – Volání jiného instrukčního segmentu
- `RET` – Návrat do jiného instrukčního segmentu
- `MOV IDL,CS:Adresa` – Čtení instrukčního segmentu (je-li `R = 1`)

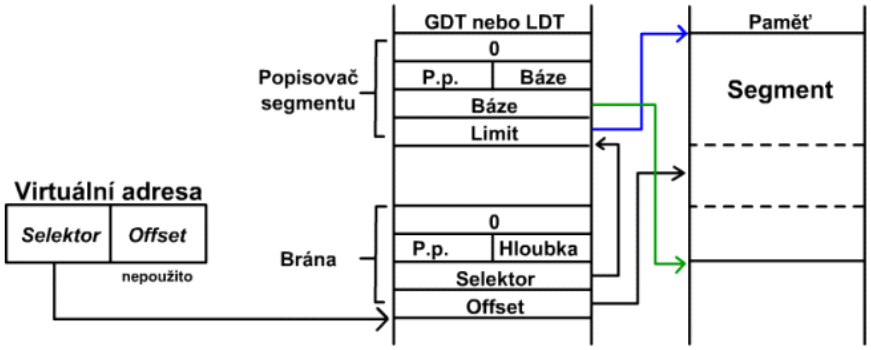
Brána pro předávání řízení (Call Gate)

- Brána je popisovač uložený v tabulce popisovačů segmentů
- **CPL ≤ DPL brány**
- **CPL ≥ DPL podprogramu**
- **Brána** je na **nižší** úrovni oprávnění
- **Podprogram** je na **vyšší** úrovni oprávnění



Brána pro předání řízení (Call Gate)

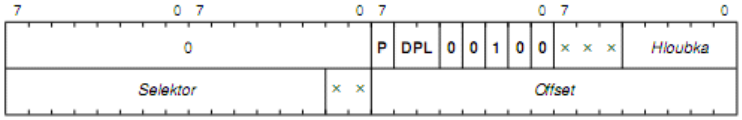
Použití brány pro předávání řízení



Použití brány pro předávání řízení

- Do segmentu se **nedá skočit přímo** (ochrana před zneužitím funkce podprogramu, definujeme přístupové body)
- **Globální adresový prostor** - viditelný pro všechny
- **Lokální adresové prostory** - viditelné pouze pro jednotlivý proces

Předávání parametrů pomocí brány



- `PUSH Parametr_1`
- `PUSH Parametr_2`
- `CALL Podprogram`
- **Každý proces** má **vlastní zásobník**
- **Každá úroveň oprávnění** uvnitř procesu má **vlastní zásobník**
- Parametry se do podprogramu předávají přes zásobník

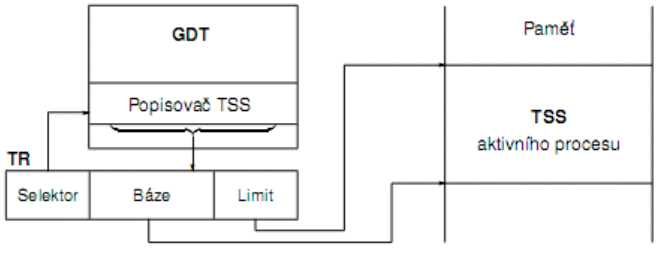
Privilegované instrukce

- instrukce, které mohou použít pouze procesy s **CPL 0** (např. *UNIX* rozlišuje mezi uživatelské úrovně **0** [*systém*] a **1** [*uživatel*])
- **LGDT** - naplnění registru **GDTR**
- **LIDT** - naplnění registru IDTR
- **LLDT** - naplnění registru LDTR
- **LTR** - naplnění registru TR
- **LMSW** - naplnění registru MSW
- **CLTS** - nulování bitu TS v registru MSW
- **HLT** - zastavení procesoru (procesor čeká na přerušení, nižší energetické nároky)
- **POPF** a **IRET** smějí měnit pouze **IOPL** s **CPL 0**

Podmíněné privilegované instrukce (CPL ≤ IOPL)

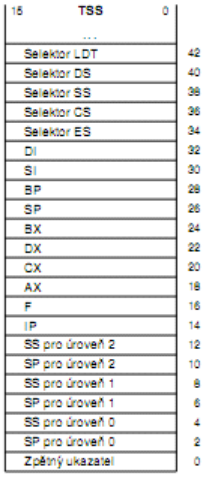
- **IN**, **INS**, **INSB**, **INSW** - čtení z I/O brány
- **OUT**, **OUTS**, **OUTSB**, **OUTSW** - zápis na I/IO bránu
- **STI**, **CLI** změna příznaku **IF**
- prefix **LOCK** - blokování směrnice
- **POPF** smí měnit pouze **IF** s **CPL** ≤ **IOPL** (jinak se změna **IF** ignoruje), porušení ► **INT 13**

Task State Segment (TSS)



Task State Segment

- **datová struktura**, ve které je od offsetu nula **definováno tolik segmentů**, kolik je **registrů**
- firmware **automaticky** posbírá **všechny registry** a uloží je do **TSS** (neukládá se tedy pouze návratová adresa přerušení, ale uloží se všechny registry do datové struktury TSS)
- Na segment s **TSS** ukazuje popisovač systémového segmentu (smí být umístěn pouze v **GDT**)
- **Typ 3** - **TSS právě aktivního** procesu
- **Typ 1** - **TSS neaktivního** procesu
- **EPL** <= **DPL**



Tak State Segment 2

- **TR** - registr patřící pravě jednomu aktivnímu procesu
- **TSS** - může mít velikost až do velikosti segmentu, předpokládá se, že operační systém si přeje ukládat dodatečné informace o každém procesu (limit procesu si může nastavit programátor dle sebe - u 80286 od 64. bitu)

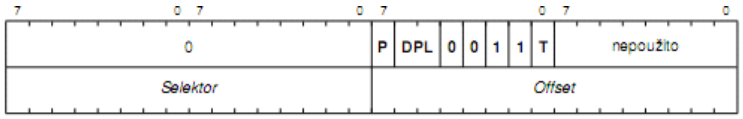
Brána zpřístupňující TSS

Přepnutí procesu může být vyvoláno

- **Vzdáleným JMP** nebo **CALL**, jehož selektor ukazuje na **popisovač TSS** nového procesu v **GDT**
- **Vzdáleným JMP** nebo **CALL**, jehož selektor ukazuje na **bránu zpřístupňující TSS**
- **IRET** s nastavením **NT 1**
- **Přerušením**, jehož přerušovací vektor ukazuje na bránu zpřístupňující TSS

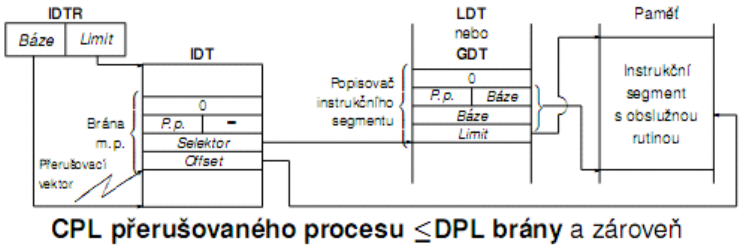
V reálném režimu

- **IDT** (Interrupt Descriptor Table) - obsahuje až 256 popisovačů rutin obsluhujících přerušení
- **IDTR** obsahuje adresu **IDT** (stejný jako **GDTR**)
- **Popisovače v IDT**
  - **Typ 1** - **Brána zpřístupňující TSS**
  - **Typ 2** - **Brána pro maskující přerušení** (Interrupt Gate) - po přerušení se zakáže další přerušení vynulováním **IF**
  - **Typ 3** - **Brána pro nemaskující přerušení** (Trap Gate) - po přerušení se nezakazuje přerušení (nenuluje se **IF**)



T=1 … Brána pro nemaskující přerušení (nenuluje IF).
T=0 … Brána pro maskující přerušení (nuluje IF).

Brána pro přerušení



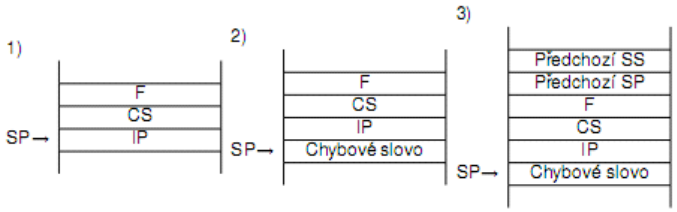
CPL přerušovaného procesu ≤DPL brány a zároveň

CPL≥DPL rutiny

Brána pro přerušení 2

Informace ukládaná do zásobníku

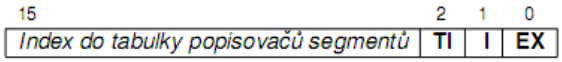
- některá přerušení navíc ukládájí chybové slovo



1. Žádné chybové hlášení.
2. Přerušení předává chybové slovo.
3. Přerušení předává chybové slovo a obsluha přerušení pracuje na jiné (vyšší) úrovni oprávnění – je uloženo původní SS:SP.

Informace ukládané do zásobníku

Formát chybového slova předávaného přerušeními 10 až 13



Chybové slovo

- **I** - index ukazující do **IDT** (nikoli do **GDT** nebo **IDT** podle **TI**)
- **EX** (External) - přerušení bylo způsobeno vnější události bez zavinění procesu

Přerušení generovaní procesorem 80286

- **Fault** - do zásobníku uloží **CS:IP** ukazující **na instrukci, která způsobila přerušení**
- **Trap** - do zásobníku uloží CS:IP ukazující **za instrukci** (na následující instrukci), která způsobila přerušení
- **Abort** - v procesu nelze pokračovat a musí být náročně ukončen

Číslo vektoru	*Určení vektoru*	*Typ přerušení*	*Chybové slovo?*
0	Dělení nulou	Fault	ne
1	Krokovací režim	Trap	ne
2	Nemaskovatelná přerušení	–	ne
3	Ladící bod	Trap	ne
4	Přeplnění	Trap	ne
5	Kontrola mezí	Fault	ne
6	Chybný operační kód	Fault	ne
7	Nedostupnost koprocesoru	Fault	ne
8	Dvojnásobný výpadek segmentu	Abort	ano (=0)
9	Překročení segmentu koprocesorem	Abort	ne
10	Chybný TSS	Fault	ano
11	Výpadek segmentu	Fault	ano
12	Výpadek segmentu se zásobníkem	Fault	ano
13	Obecná chyba ochrany	Fault	ano
16	Chyba koprocesoru	Fault	ne

Rezervovaná přerušeni 80286

Počáteční nastavení procesoru

- nastává po signálu **RESET** (reálný režim)

Registr	Obsah
F	0002h
MSW	FFF0h
IP	FFF0h
Selektor CS	F000h
Selektor DS	0000h
Selektor SS	0000h
Selektor ES	0000h
Báze CS	FF0000h
Báze DS	000000h
Báze SS	000000h
Báze ES	000000h
Limit CS	FFFFh
Limit DS	FFFFh
Limit SS	FFFFh
Limit ES	FFFFh
Báze IDT	000000h
Limit IDT	FFFFh

Počáteční nastavení procesoru

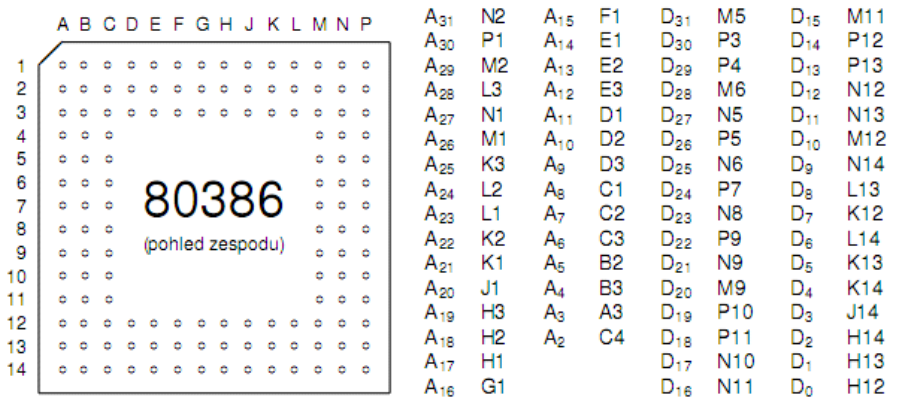
Zapnutí chráněného režimu

1. Do paměti zavést programy a odpovídající tabulky popisovačů
2. Nastavit GDTR a IDTR
3. Zapnout chráněný režim nastavením bitu PE registru MSW na 1
4. Provést blízký skok JMP proto, aby se zrušil obsah interních front procesoru, ve kterých jsou uloženy

předvybrané instrukce (výběr instrukcí totiž závisí na zvoleném režimu procesoru)
5. Vytvořit TSS inicializačního procesu a nastavit obsah TR
6. Naplnit LDTR
7. Inicializovat ukazatel vrcholu zásobníku SS:SP

Procesor Intel 80386

- *1986-1994*
- 16-40 MHz
- první procesor s kontakty zespodu
- 32-bitový procesor, datová a adresová (až 4GB RAM)
- zakladatel architektury IA-32
- **Varianty**
  - **i386DX** - plnohodnotný procesor
  - **i386SX** - 16-bitovou datová a 24-bitovou adresová sběrnicí
  - **i386SL** - pro laptop počítače (nižší spotřeba)
- **matematický koprocesor (i387) separátně**
- bez chlazení



A_{31}	N2	A_{15}	F1	D_{31}	M5	D_{15}	M11
A_{30}	P1	A_{14}	E1	D_{30}	P3	D_{14}	P12
A_{29}	M2	A_{13}	E2	D_{29}	P4	D_{13}	P13
A_{28}	L3	A_{12}	E3	D_{28}	M6	D_{12}	N12
A_{27}	N1	A_{11}	D1	D_{27}	N5	D_{11}	N13
A_{26}	M1	A_{10}	D2	D_{26}	P5	D_{10}	M12
A_{25}	K3	A_9	D3	D_{25}	N6	D_9	N14
A_{24}	L2	A_8	C1	D_{24}	P7	D_8	L13
A_{23}	L1	A_7	C2	D_{23}	N8	D_7	K12
A_{22}	K2	A_6	C3	D_{22}	P9	D_6	L14
A_{21}	K1	A_5	B2	D_{21}	N9	D_5	K13
A_{20}	J1	A_4	B3	D_{20}	M9	D_4	K14
A_{19}	H3	A_3	A3	D_{19}	P10	D_3	J14
A_{18}	H2	A_2	C4	D_{18}	P11	D_2	H14
A_{17}	H1			D_{17}	N10	D_1	H13
A_{16}	G1			D_{16}	N11	D_0	H12

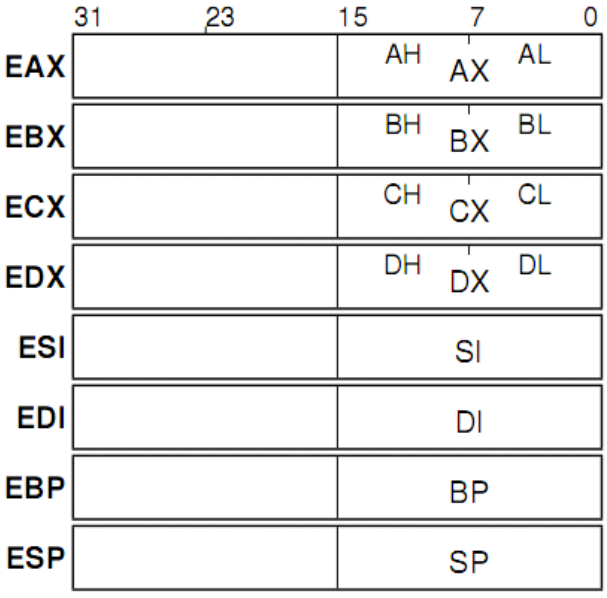
$\overline{ADS}$	E14	$\overline{BS16}$	C14	$\overline{LOCK}$	C10	W/$\overline{R}$	B10	INTR	B7
$\overline{BE_3}$	A13	$\overline{BUSY}$	B9	M/$\overline{IO}$	A12	CLK2	F12	NMI	B8
$\overline{BE_2}$	B13	D/$\overline{C}$	A11	$\overline{NA}$	D13	HOLD	D14	PEREQ	C8
$\overline{BE_1}$	C13	$\overline{ERROR}$	A8	$\overline{READY}$	G13	HLDA	M14	RESET	C9
$\overline{BE_0}$	E12								

GND: A2 A6 A9 B1 B5 B11 B14 C11 F2 F3 F14 J2 J3 J12 J13 M4 M8 M10 N3 P6 P14
U_{cc}: A1 A5 A7 A10 A14 C5 C12 D12 G2 G3 G12 G14 L12 M3 M7 M13 N4 N7 P2 P8

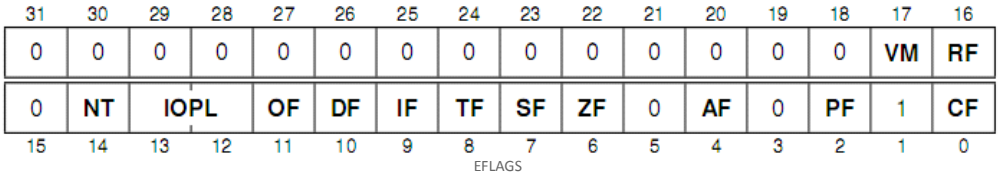
Procesor 80386 (i386DX)

- z paměti lze vytáhnout minimálně 4B a to jen z adres dělitelných 4
- při přesahu nutno vytáhnout 8B
- snaha o optimalizaci ukládání proměnných (např. do adres dělitelných 4)

Registry procesoru 80386



Registry procesoru Intel 80386 (i386DX)



- **VM** (Virtual 8086 Mode) - zapíná režim *virtuální 8086* pro proces, jemuž obsah příznakového registru náleží, příznak VM smí programátor nastavovat pouze v chráněném režimu, a to instrukcí IRET, a jenom na úrovni oprávnění 0, příznak je také modifikován mechanismem přepnutí procesu, libovolný počet virtuálních 8086
- **RF** (Resume Flag) - maskuje opakování ladícího přerušení