Lecture 3

Interrupt handling in PM

Interrupt handling in Real Mode

- interrupts can be caused by: hardware (maskable INTR, nonmaskable NMI), software (explicitly with INT instruction, exceptions),

- upon receipt of an interrupt, control is redirected to an interrupt handling routine,

- the entry point to the interrupt handling routine is taken from the table of interrupt vectors that is hold in memory starting from address: 0000h:0000h,

Interrupt handling in Protected Mode

The PM interrupt structure is fully analogous to that of Real Mode.

- interrupts can be caused by: hardware (maskable INTR, nonmaskable NMI), software (explicitly with INT instruction, exceptions),

- upon receipt of an interrupt, control is redirected to an interrupt handling routine,

- the entry points to interrupt routines are hold in **interrupt gates** and **trap gates**,

- interrupt gates are well suited for hardware interrupts while trap gates for software ones,

- interrupt and trap gates are listed in an array called **Interrupt Descriptor Table** (IDT). The IDT is the counterpart of Real Mode interrupt-vector table.

Interrupt Gate

Interrupt gates are designed for vectoring hardware interrupts. The interrupt gate is 8-byte long structure containing vectoring and privilege information (see Fig. 3.1).

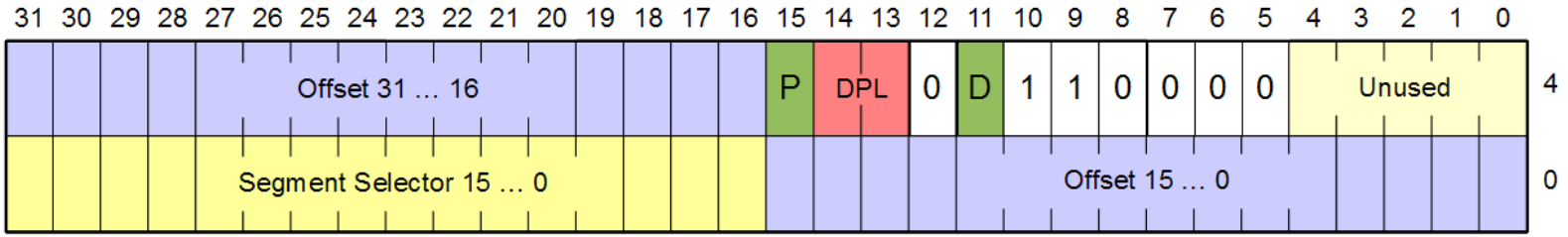

Fig. 3.1 Interrupt gate

- **Selector**: selector to destination code segment containing the interrupt routine,
- **Offset**: the offset within code segment identifying the routine entry point,
- **P** (Present): indicates whether the gate is available in memory (P=1) or not (P=0),
- **DPL**: indicates the privilege level of the gate. A hardware interrupt regardless to DPL may always vector through the gate if it is present (P=1). DPL applies to software interrupts generated with INT, INTO, BOUND instructions and exceptions. For a software interrupt or exception the instruction causing an interrupt must have a CPL at least as privileged as DPL of the gate. Otherwise GP would be generated.
- **D** (Default): 16-bit code (D=0) and 32-bit code (D=1).

Interrupt handling in PM

Transfer of Control through Interrupt Gate
While control is transferred to interrupt routine via interrupt gate:
- three 32-bit wide words are pushed on the stack: a copy of EFLAGS register, a copy of CS and a copy of EIP (the return address),

- the IF bit of EFLAGS is cleared disabling further maskable interrupts,

- the EIP register is loaded with offset field of the interrupt gate and CS is loaded with the selector field from the interrupt gate.

Privilege checks
In case of hardware interrupts the DPL of interrupt gate is of no concern. However the DPL of destination code segment must be less or equal to the CPL.

It should be noted that in case of hardware interrupt the peripheral devices such as interrupt controller I8259A play the same role both in Real and Protected Modes. It means that: I8259A generates the interrupt request signal INT, processor detects that signal and after finishing current instruction sends the interrupt aknowledge signal ~INTA, I8259A updates its internal registers, processor sends the second ~INTA and in response the interrupt controller provides the 8-bit number of interrupt on the system data bus.

The return from the interrupt routine is realized with use of IRET instruction.

Trap Gate

Trap gates are designed for vectoring software interrupts. The trap gate is 8-byte long structure containing vectoring and privilege information (see Fig. 3.2).

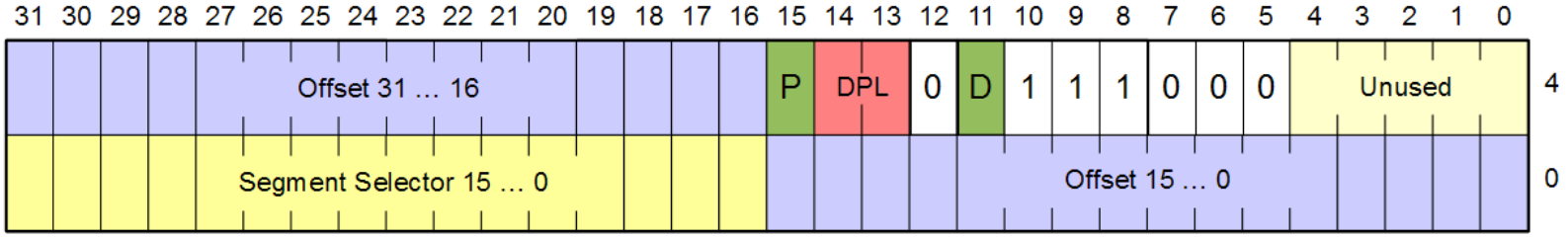

Fig. 3.2 Trap gate

- **Selector**: selector to destination code segment containing the interrupt routine,
- **Offset**: the offset within code segment identifying the routine entry point,
- **P** (Present): indicates whether the gate is available in memory (P=1) or not (P=0),
- **DPL**: indicates the privilege level of the gate. The instruction causing an interrupt/exception must have a CPL at least as privileged as DPL of the gate. Otherwise GP would be generated.
- **D** (Default): 16-bit code (D=0) and 32-bit code (D=1).

Transfer of Control through the Trap Gate
Control transfer via trap gate is identical to the transfer process through the interrupt gate, except that IF flag in EFLAGS register is not cleared.

Interrupt Descriptor Table
The IDT holds interrupt gates. Due to the fact that maximum offset (limit) may exceed up to 7FFh there is the possibility to hold up to 256 interrupt gates. The location and limit of IDT is described by pseudodescriptor hold in IDTR register (see Fig. 3.3).

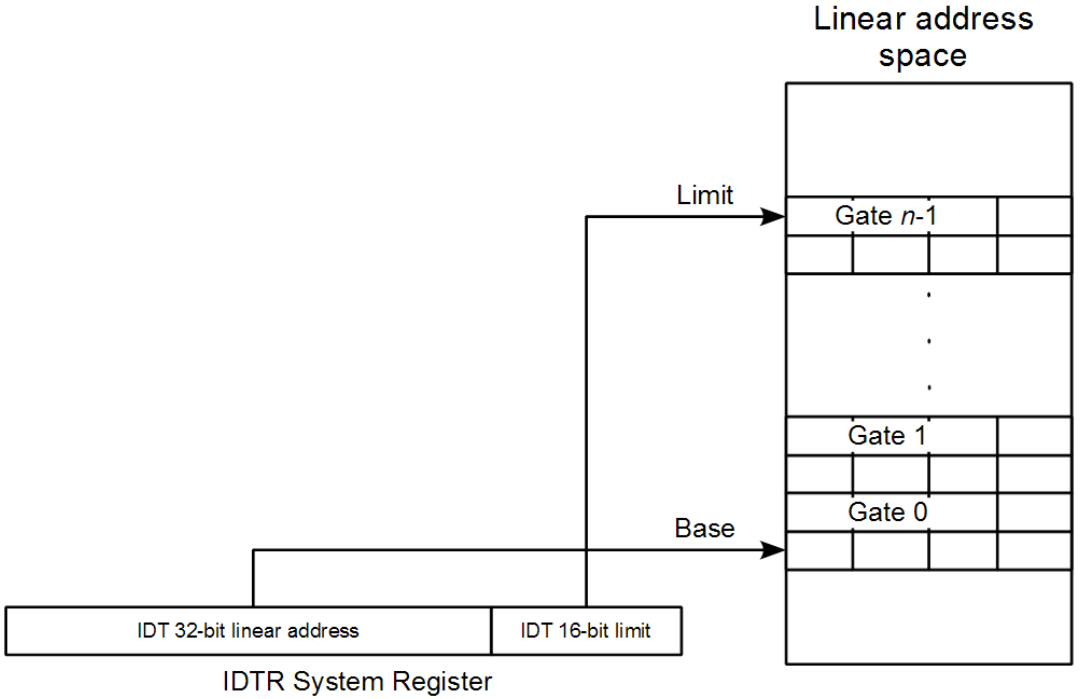

Fig. 3.3 Description of IDT

Interrupt handling in PM

The structure of supervisor application with basic interrupt handling

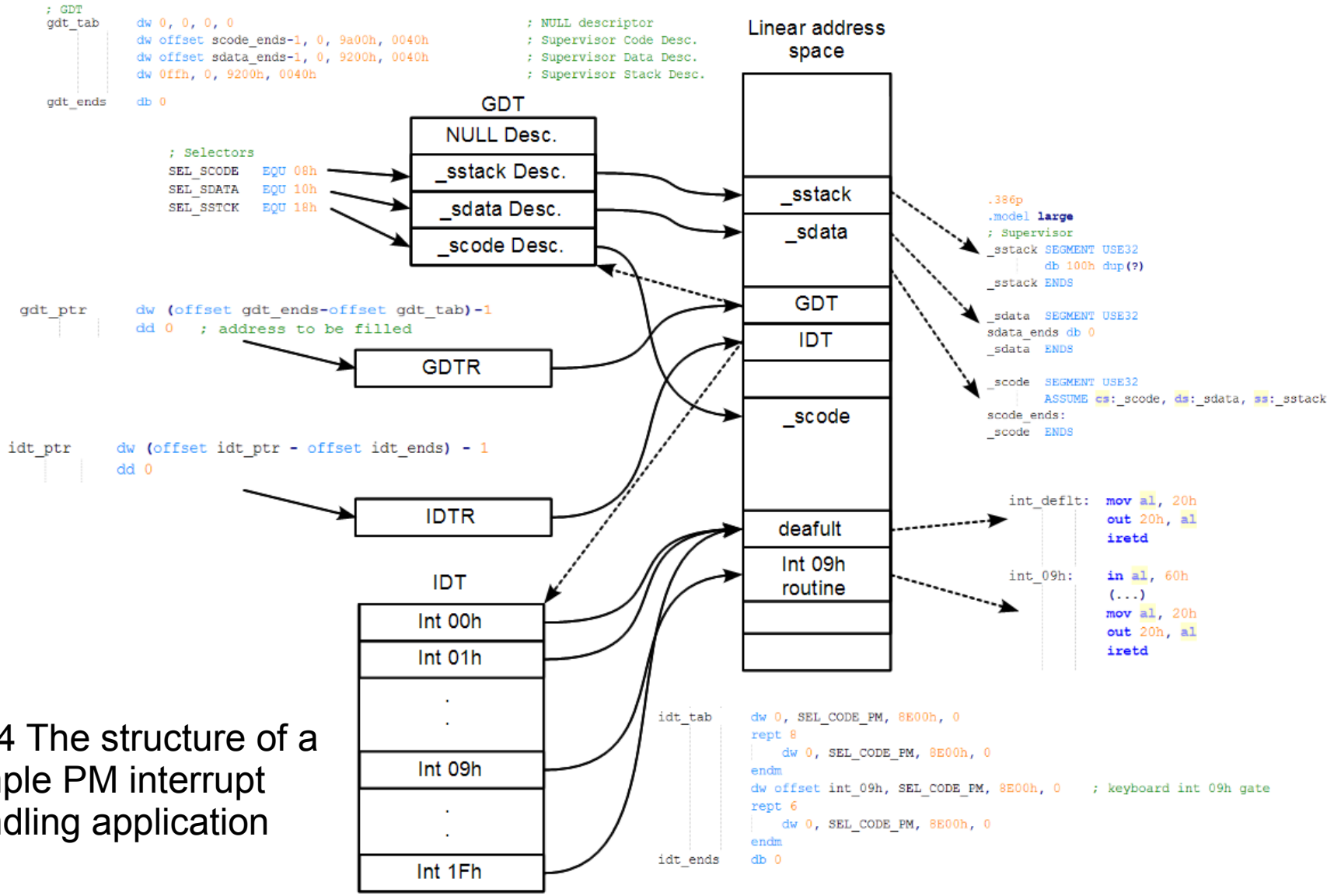

Fig. 3.4 The structure of a simple PM interrupt handling application

Default interrupt handling routine

The default handling routine should have at least the following functionality: send EOI word to I8259A interrupt controller.

```
int_deflt:  mov al, 20h
            out 20h, al
            iretd
```

Fig. 3.5 Default (basic) interrupt handling routine

Requirements for IDTR content in Real Mode

In Real Mode the same internal processor registers are used as in case of Protected Mode. Hence they must hold properly prepared data. In Real Mode interrupt-vector table starts from 0000h:0000h and extends up to the address 0000h:3FFh. In order to make Real Mode interrupts system work properly we must provide the following pseudodescriptor for IDT table: linear address 00000000h, limit 000003FFh and write it into the IDTR (see code below).

Fig. 3.6 IDTR contents preparation for Real Mode

```
            (...)
idt_ptr     dw 0
            dd 0
            (...)
            mov bx, offset idt_ptr
            mov word ptr [bx], 3ffh
            mov word ptr [bx+2], 0
            mov word ptr [bx+4], 0
            lidt [bx]
            (...)
```