

Short Notes Chapter 15 (Lecture 39-43):

Protected Mode:

Switching processor in the newer 32-bit mode is a very easy task. Just turn on the least significant bit of a new register called CRO (Control Register 0) and the processor switches into 32-bit mode called protected mode. However, manipulations in the protected mode are very different from those in the read mode.

Extended registers:

All registers in 386 have been extended to 32-bits. The new names are EAX, EBX, ECX, EDX, ESI, EDI, ESP, EBP, EIP, and EFLAGS. The original names refer to the lower 16-bits of these registers. A 32-bit address register can access up to 4GB of memory so memory access has increased a lot.

Segment selectors:

We call segment registers as segment selectors and they are still 16-bits wide. We are also given two other segment selectors FS and GS for no specific purpose just like ES.

The working of segment registers as being multiplied by 10 and added into the offset for obtaining the physical address is totally changed.

Working of segment selector:

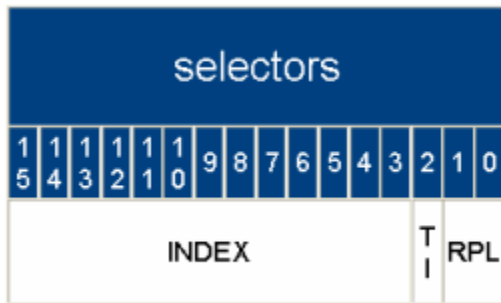
Now the selector is just an index into an array of segment descriptors where each descriptor describes the base, limit, and attributes of a segment.

Role of selector is to select on descriptor from the table of descriptors and the role of descriptor is to define the actual base address.

For example, an operating system can define the possible descriptors for a program and the program is bound to select one of them and nothing else. This sentence also hints that the processor has some sense of programs that can or cannot do certain things like change this table of descriptors. This is called the privilege level of the program and varies for 0 (highest privilege) to 3 (lowest privilege).

Format of a selector:

The format of a selector is shown below.



The table index (TI):

TI is set to 0 to access the global table of descriptors called the GDT (Global Descriptor Table).

It is set to 1 to access another table, the local descriptor table (LDT).

RPL:

RPL is the requested privilege level that ranges from 0-3 and informs what privilege level the program wants when using this descriptor.

Index:

The 13-bit index is the actual index into the GDT to select the appropriate descriptor. 13 bits mean that a maximum of 8192 descriptors are possible in the GDT.

GDT:

The GDT itself is an array of descriptors where each descriptor is an 8-byte entry. The base and limit of GDT is stored in a 48-bit register called the **GDTR**.

This register is loaded with a special instruction LGDT and is given a memory address from where the 48-bits are fetched.

The first entry of the GDT must always be zero. It is called the null descriptor. After that any number of entries up to a maximum of 8191 can follow.

Format of code & data descriptor:

The format of a code and data descriptor is shown below.

code (application) segment descriptor																																					
offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
+4	BASE (bit31..24)								G	D	r.	A	LIMIT (bit19..16)								P	DPL	S	X	=	=	C	R	A	BASE (bit23..16)							
+0	BASE (bit15..0)																LIMIT (bit15..0)																				

data (application) segment descriptor																																					
offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
+4	BASE (bit31..24)								G	B	r.	A	LIMIT (bit19..16)								P	DPL	S	X	=	=	E	W	A	BASE (bit23..16)							
+0	BASE (bit15..0)																LIMIT (bit15..0)																				

The **32-bit base** in both descriptors is scattered into different places because of compatibility reasons.

The **limit** is stored in 20 bits but the **G bit** defines that the limit is in terms of bytes of 4K pages therefore a maximum of 4GB size is possible.

The **P bit** must be set to signal that this segment is present in memory.

DPL is the descriptor privilege level again related to the protection levels in 386.

D bit defines that this segment is to execute code is 16-bit mode or 32bit mode.

C is conforming bit.

R signals that the segment is readable.

A bit is automatically set whenever the segment is accessed.

The combination of **S (system)** and **X (executable)** tell that the descriptor is a code or a data descriptor.

B (big) bit tells that if this data segment is used as stack SP or ESP is used.

VESA and VBE:

VESA is the Video Electronics Standards Association and VBE is the set of Video BIOS Extensions proposed by them. The VESA VBE 2.0 standard includes a linear frame buffer mode. This mode allows direct access to the whole video memory.

Some important VESA services are listed below:

INT 10 – VESA – Get SuperVGA Information

AX = 4F00h

ES:DI -> buffer for SuperVGA information

Return:

AL = 4Fh if function supported

AH = status

INT 10 – VESA – Get SuperVGA Mode Information

AX = 4F01h

CX = SuperVGA video mode

ES:DI -> 256-byte buffer for mode information

Return:

AL = 4Fh if function supported

AH = status

ES:DI filled if no error

INT 10 – VESA – Set VESA Video Mode

AX = 4F02h

BX = new video mode

Return:

AL = 4Fh if function supported

AH = status

VESA mode 4117:

One of the VESA defined modes is 4117 which is a 1024x768 mode with 16-bit color and a linear frame buffer. The 16 color bits for every pixel are organized in 5:6:5 format with 5 bits for red, 6 for green, and 5 for blue. This makes 32 shades of red and blue and 64 shades of green and 64K total possible colors. The 32-bit linear frame buffer base address is available at offset 28 in the mode information buffer.

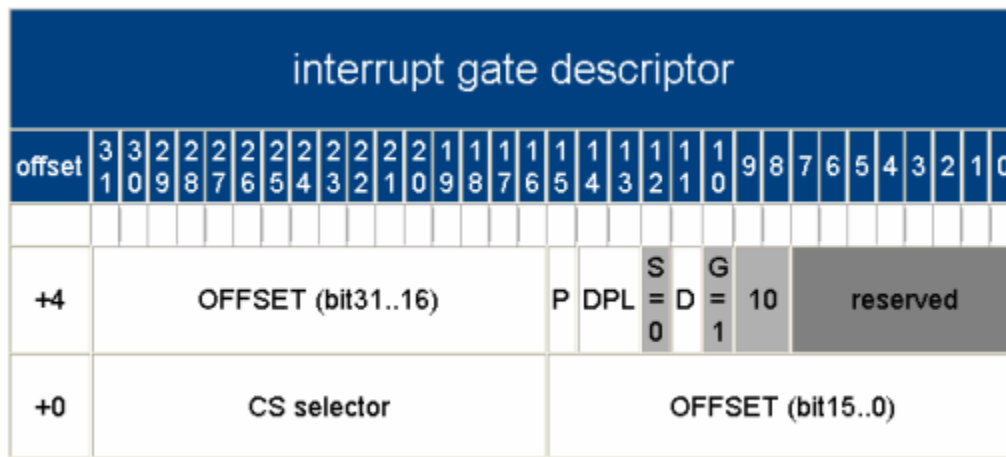
INTERRUPT HANDLING

Handling interrupts in protected mode is also different. Instead of the IVT at physical address 0 there is the IDT (interrupt descriptor table) located at physical address stored in IDTR, a special purpose register.

IDTR:

The IDTR is also a 48-bit register similar in structure to the GDTR and loaded with another special instruction LGDT.

The format of the interrupt descriptor is as shown below:



The **P** and **DPL** have the same meaning as in data and code descriptors.

The **S bit** tells that this is a system descriptor while the 1110 following it tells that it is a 386-interrupt gate.