Chapter Two

Protected Mode Memory Addressing

2.1 Protected Mode Memory Addressing

Protected mode memory addressing (80268 and above) allows access to data and programs located above the first 1M byte of memory as well as within the first 1M byte of memory. Addressing this extended section of the memory system requires a change to the segment plus offset addressing used with real mode memory addressing. When data and programs are addressed in extended memory, the offset address is still used to access information located within the memory segment. The difference is that the segment address, as discussed with real mode memory addressing, is no longer present in the protected mode. ***In place of the segment address, the segment register contains a selector that selects a descriptor table. The descriptor describes the memory segments location, length and access rights****. Because the segment register and offset address still access memory, protected mode instructions are identical to real mode instructions. In fact, most programs Witten to function in real mode will function without change in the protected mode. The difference between modes is in the way that the segment register is interpreted by the microprocessor to access the memory segment.*

2.2 Selectors and Descriptor:

The selector, located in the segment register, selects one of 8192 descriptors from one of two tables of descriptor. The descriptor describes the memory segments location, length and access rights.

There are two descriptor tables used with the segment registers: one contains global descriptors and other contains local descriptors. The global descriptors contain segment definitions that apply to all programs, while the local

descriptors are usually unique to an application. Each descriptor table contains 8192 descriptor, so a total of 16384 descriptors are available to an application at any time. Because the descriptor describes a memory segment, this allows up to 16384 memory segment to be described for each application.

Figure (1) shows the format of a descriptor for the 80268 through Pentium Pro. Note that each descriptor is 8byte in length, so the global and local descriptor tables are each a maximum of 64kbyte in length. Descriptors for 80268 and 8036 through Pentium Po differ slightly, but the 80286 descriptor is upward compatible.

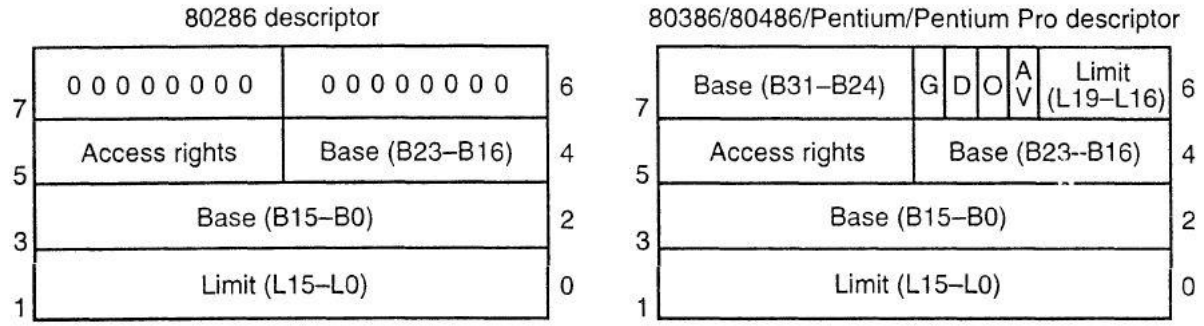


Figure (1) the descriptor formats for the 80268 through Pentium Pro microprocessor.

- *The **Segment Limit** contains the last offset address found in a segment. The 80286 has a 16-bit limit, and 80386 through the Pentium Pro have a 20-bit limit. The 80268 has accesses memory segments that are between 1 and 64K byte in length. The 80368 and above access memory segments that are between 1 and 1M byte or 4K and 4G bytes in length.*
- *The **Base Address** portion of the descriptor indicates the starting location of memory segment. For the 80268 microprocessor, the base address is a 24-bit address, so segments begin at any location in its 16M byte of memory. The 80368 and above use a 32-bit base address that allows segments to begin at any location in its 4G bytes of memory. Notice how the 80286 descriptors base address is*

upward compatible to the 80386 though the Pentium Pro descriptor because it's most-significant 8-bits.

For example, if a segment begins at memory location F00000H and ends at location F000FFH, the base address isF00000H and the limit is FFH. For the 80268 microprocessor, the base address is F00000H and the limit is 00FFH. For the 80368 and above, the base address is 00FF00000H and the limit is 000FFH.

- *The **Granularity bit (G bit)** is found in the 80368 through the Pentium Pro descriptor that is not found in the 80268 descriptor. If G=0, the limit specifies a segment limit of from 1 to 1M byte in length. If G=1 the value of limit multiplied by 4K bytes. This allows a segment length of 4K to 4G bytes in steps of 4K bytes. The reason that the segment length is 64K bytes in 80286 is that the offset address is always 16-bit because of its 16-bit internal architecture. The 80368 and above use a 32-bit architecture, which allows an offset address, in the protected mode operation, of the 32-bit. This 32-bit offset address allows segment length of 4G bytes, and the 16-bit offset address allows segment lengths of 64K bytes. Operating systems operate in either a 16 or 32-bit environment.*

- *The **Available Segment (AV):** is found in 80386 and above descriptor, is used by some operating systems to indicate that the segment is available (AV=1) or not available (AV=0).*

- *The **D bit:** indicates how the 80386 though the Pentium Pro instructions access register and memory data in the protected and real mode. If D=0, the instructions are 16-bit instructions compatible with 8086-80286 multiprocessors. If D=1, the instructions are 32-bit instructions.*

- *The **Access Rights byte**: control access to the protected mode memory segment. This byte describes how the segment functions I the system. The access*

rights byte allows complete control over the segment. Figure (2) show the access right bytes.

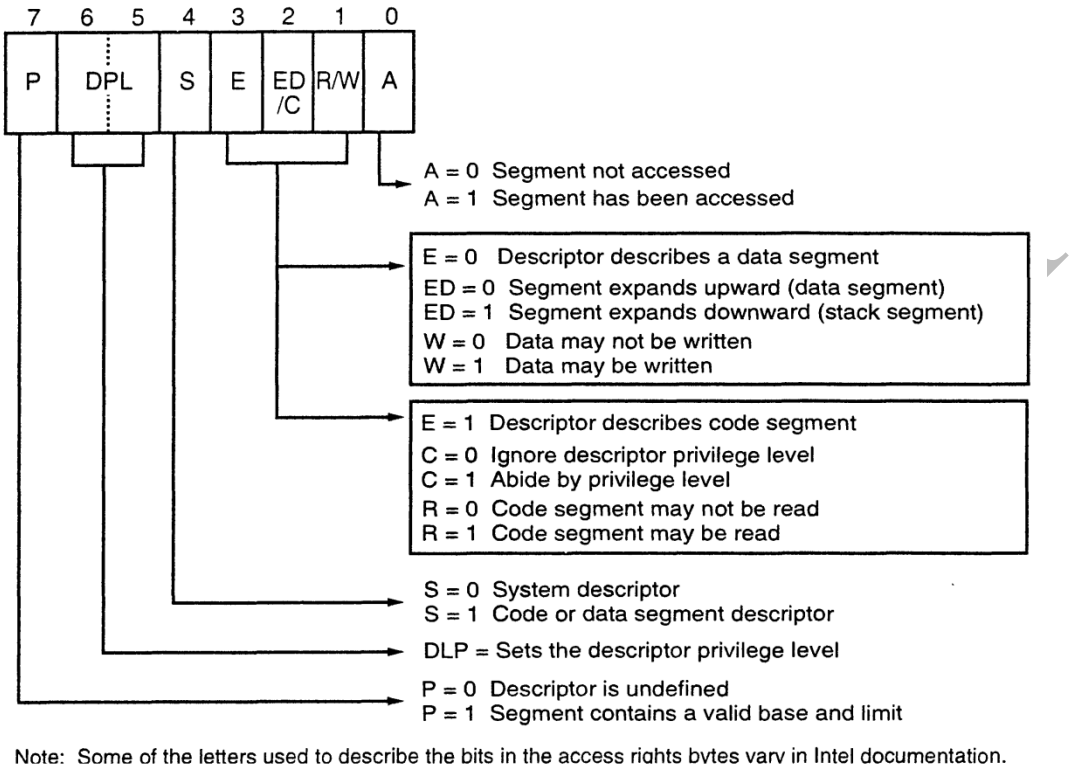


Figure (2) the access rights byte for 80268 through Pentium Pro microprocessor.

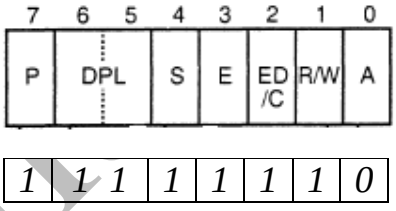
- ***A (Accesses) bit**: if the data or code segment is accessed, A=1; otherwise, A=0 (segment not accessed).*
- ***R/W (Read/ Write) bit**: this bit allows code or data to be read protected or write protected.*
- ***ED/C**: this has a different meaning for the data segment and code segment. In the case of data, it indicates if the segment should expand downward as the stack segment grows, or upward as the data segment grows. In the case of the code segment, it is used to enforce certain rules of privilege level access.*
- ***E bit**: this indicates if the information is executable (E=1), such as code, or no executable (E=0), such as data or stack.*

- ***S bit****: this indicates if the descriptor belongs to the code and data segment (S=1) or if it is a system segment descriptor (S=0).*
- ***DPL (Descriptor Privilege Level) bits****: this allows one of the combinations 00, 01, 10, or 11, to be assigned to the code or data, indicating the privilege level.*
- ***P (Present) bit****: this indicates if the piece of code or data is present in the main memory (DRAM). If it is present (P=1), the CPU will process it. if it is not (P=0), the CPU cases an exception and the exception handler of the operating system will bring the desired piece of code or data into main memory from the hard disk.*

Example (1): *The access rights byte of a segment descriptor contains FEH? What are its characteristics?*

Solution:

Expressing the access rights in binary form FEH ⟶ *1111 1110B*



P=1: segment descriptor mapped into Physical memory.

DPL = 11: Level 3 lowest.

S =1: Code /data segment descriptor.

E =1: conform code segment.

ED/C = 1: conforming CS.

R/W =1: readable CS

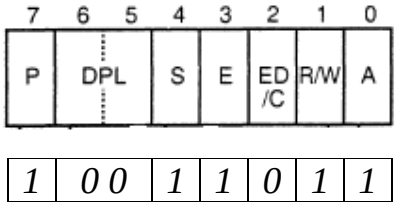
A =0: segment not accessed.

Example (2): *State the characteristic of each of the following access bytes. State if it is for code or data?*

1) 10011011 2)10010111 3)11110001

Solution:

1)



1	0 0	1	1	0	1	1

P=1: segment descriptor mapped into Physical memory.

DPL = 00: Level 0.

S =1: Code /data segment descriptor.

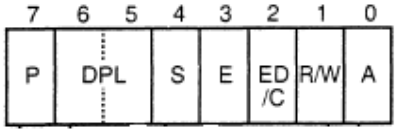
E =1: conform code segment.

ED/C = 0: ignore CS.

R/W =1: readable CS

A =1: segment accessed.

2)



1	0 0	1	0	1	1	1

P=1: segment descriptor mapped into Physical memory.

DPL = 00: Level 0 highest.

S =1: Code /data segment descriptor.

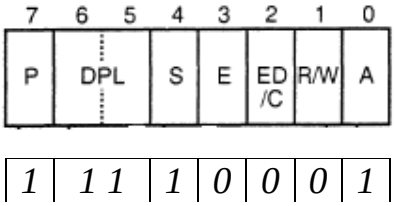
E =0: conform data segment.

ED/C = 1: expand downward SS.

R/W =1: data may be written DS.

A =1: segment accessed.

3)



P=1: segment descriptor mapped into Physical memory.

DPL = 11: Level 3 lowest.

S =1: Code /data segment descriptor.

E =0: conform data segment.

ED/C = 0: expand upward DS.

R/W =0: data may be not written DS.

A =1: segment accessed.

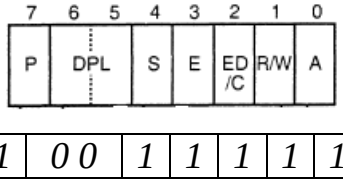
Example (3): *Code a descriptor that describes a memory segment that begins at location 210000H and ends at location 21001FH. This memory segment is a code segment that can be read. The descriptor is for an 80286 microprocessor.*

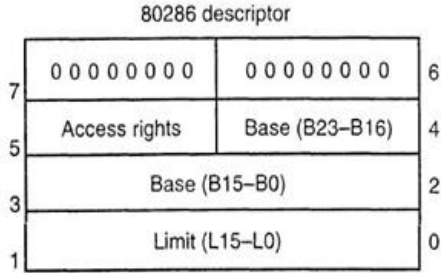
Solution:

Base (start) = 210000H

End = base + limit

21001FH = 210000H + limit ⟶ *limit = 21001FH-210000H =001FH*





2.3 Local and Global Descriptor Tables:

Descriptors are chosen from the descriptor table by the segment register as shown in Figure (3). The segment register contains a 13-bit selector field, a table selector bit, and a requested privilege level field. The 13-bit selector chooses one of the 8192 descriptors from the descriptors table. The TI bit selects either the global descriptor table (TI=0) or the local descriptor table (TI=1). The requested privilege level (RPL) requests access privilege of memory segment. The highest level is 00 and the lowest is 11. If the requested privilege level matches or is higher in priority than the privilege level set by success rights byte, access is granted. For example, if the requested privilege level is 10 and the access rights byte sets the segment privilege level at 11, access is granted because 10 is higher in priority than privilege level 11.

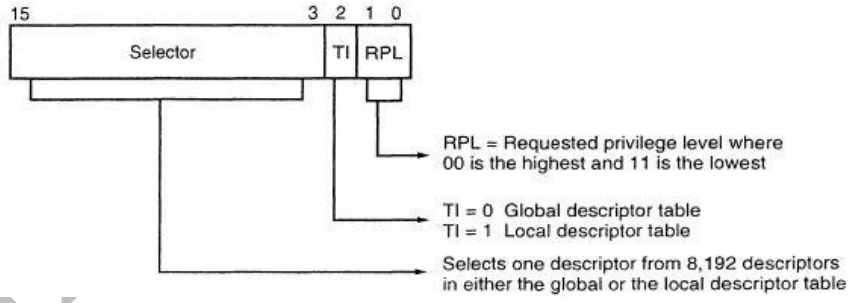


Figure (3) the content of a segment register during protected mode operation of the 80286 though Pentium Pro.

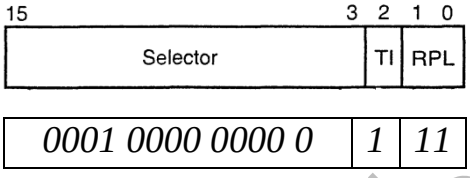
Example (4): *Assume that the base address of the LDT is 00120000Hand the GDT base address*

00	00
9F	21
0000	
001F	

is 00100000H. If the value of the selector loaded into the CS register is 1007H. What is the request privilege level? Is the segment descriptor in the GDT or LDT? What is the address of the segment descriptor?

Solution:

Expressing the selector in binary form 1007H ⟶ 0001 0000 0000 0111B



RPL =11=3

TI = 1: This means that the segment descriptor is LDT

Descriptor (address) = Base + Offset

*Offset = Index * 8 = 0000 0010 0000 0000 * 8= 0200H*8 = 1000H*

Descriptor (address) = Base (LDT) + Offset

= 0012 0000H + 1000H = 0012 1000H

2.4 Program-invisible Register:

The global and local descriptor tables are found in the memory system. In order to access and specify the address of these table, the 80286, 80386, 80486, Pentium and Pentium Pro contain program-invisible registers as shown in Figure (4). The program invisible registers are not directly addressed by software, so they are given this name, although some of these registers are accessed by system software. These registers control the microprocessor when operated in the proteced mode.

Each of the segment registers contians a program-invisible portion used in the proteced mode. The program-invisible portion of these registers is offten called cache memory becouse a cache memory is any memory that stores information.

This cache is not to be confused with the normal level1 or level2 caches found with the microprossor. The program-invisible portion of the segment register is loaded with the base address, limit and access rights each time the number in the segment register is changed. When a new segment number is placed in a segment register, the microprosser accesses a descriptor table and loads the descriptor into the program-invisible cache portion of the segment register. It is held thee and used to access the memory segment until the segment number us again changed. This allows the microprossor to access a memory segment repeatly without referring back to the descriptor table for each access, hence the term cache.

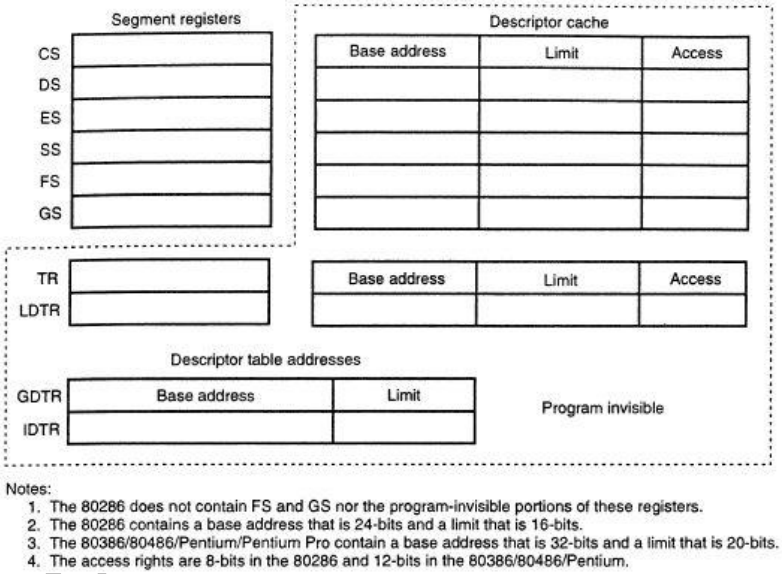


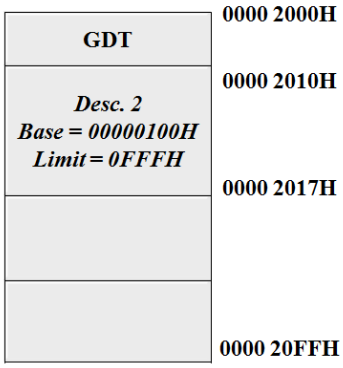
Figure (4) The program-invisible register within 80286 though Pentium Pro.

The GDTR (Gloable Descriptor Table Register) and IDTR (Interrupt Descriptor Table Register) contian the base address of the descriptor table and its limit. The limit of each descriptor table is 16-bits becouse the maximum table length is 64K bytes. When protected mode operation is desired, the address of the global descriptor table and its limit are loaded into GDT. Before using protected mode, the interrrupt descriptor table and the IDTR must also initialized.

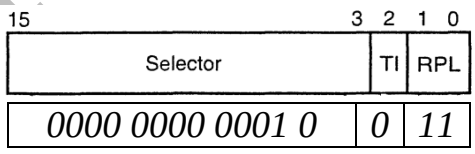
The location of the local descriptor table is selected from the global descriptor table. One of the global descriptors is set up to address the local desciptor table. To access the local descriptor table, the LDTR (Local Descriptor Table Rigister) is loaded with a selector, just as a segment register is loaded with a selector. This selector accesses the global descriptor table and loads the base addess, limit, and access rights of the local descriptor table into cache portion of the LDTR.

Example (5): *If the content of DS = 0013H and the descriptor table shown below. Calculate the Descriptor address and physical address for the following instruction.*

MOV AX, [10H]



Solution:



RPL =11=3
TI = 0: This means that the segment descriptor is GDT
End GDT = 0000 20FFH, Start (Base) = 0000 2000H
End = Base +limit
0000 20FFH = 0000 2000H + limit ⟶ limit =0000 20FFH- 0000 2000H
Limit = 00FFH
Descriptor (address) = Base + Offset

*Offset = Index * 8 = 0000 0000 0000 0010 * 8= 0002H*8 = 0010H*

Descriptor (address) = Base (LDT) + Offset

= 0000 2000H + 0010H = 0000 2010H

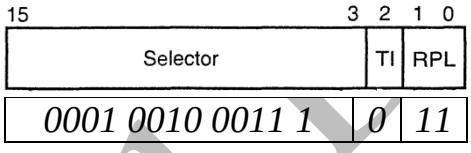
Physical Address = segment Base + Offset

= 0000 0100H + 0010H = 0000 0110H

Example (6): *If the content of DS = 123BH and the descriptor table contents are given below, find out the descriptor number, table index, RPL, base address, end address, and also give the status of access rights bits.*

1F A0 00 30 20 93 9F 12

Solution:



RPL =11=3(lowest)

TI = 0: This means that the segment descriptor is GDT

Descriptor number= 0 0 0 1 0 0 1 0 0 0 1 1 1

$2^{12}2^{11}2^{10}2^{9}\ 2^{8}2^{7}2^{6}2^{5}\ 2^{4}2^{3}2^{2}2^{1}\ 2^{0}$ *=1+2+4+64+512=583*

1F	*A0*
00	*30*
20 93	
9F 12	

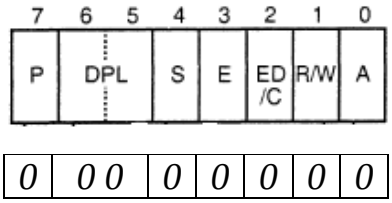
Limit = 0 9F12H

Start (Base) = 1F30 2093H

End = Base +limit

= 1F30 2093H + 0 9F12H =1F30 BFA5H

Access rights bits= 00H



0	0 0	0	0	0	0	0

P=0: segment descriptor undefined.

DPL = 00: Level 0 highest.

S =0: system descriptor.

E =0: conform data segment.

ED/C = 0: expand upward DS.

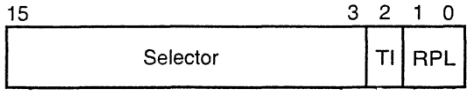
R/W =0: data may be not written DS.

A =0: segment not accessed.

Example (7): *If the content of DS = 1003H and the descriptor table contents are given below, find out the descriptor number, table index, RPL, base address, end address, and also give the status of access rights bits.*

00 00 FE 0F 00 00 FF FF

Solution:



0001 0000 0000 0	0	11

RPL =11=3(lowest)

TI = 0: This means that the segment descriptor is GDT

Descriptor number= 0 0 0 1 0 0 0 0 0 0 0 0 0

$2^{12}2^{11}2^{10}2^{9}\ 2^{8}2^{7}2^{6}2^{5}\ 2^{4}2^{3}2^{2}2^{1}\ 2^{0}$ *=1+512=513*

00	00

FE	0F
00 00	
FF FF	

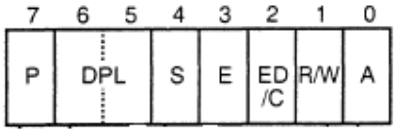
Limit = FFFFH

Start (Base) = 0F 00 00H

End = Base +limit

= 0F 00 00H + FFFFH =0F FF FFH

Access rights bits= FEH



1	1 1	1	1	1	1	0

P=1: segment descriptor mapped into physical memory.

DPL = 11: Level 3 lowest.

S =1: segment descriptor.

E =1: conform code segment.

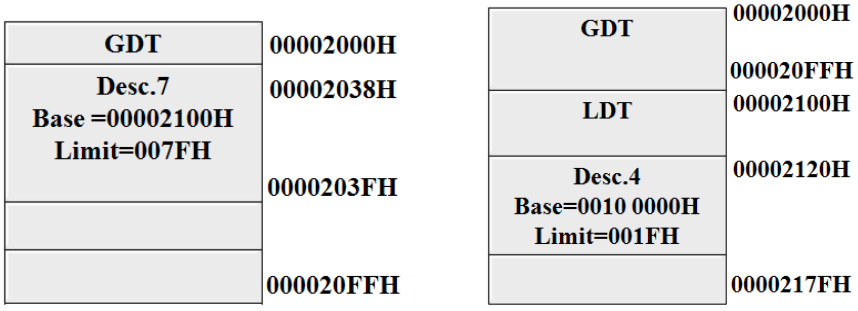
ED/C = 1: conforming CS.

R/W =1: data readable CS.

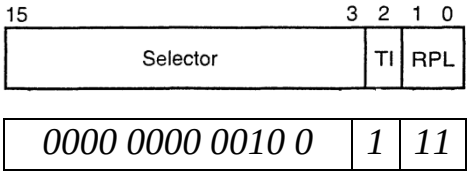
A =0: segment not accessed.

Example (8): *If the content of DS = 0027H, LDTR=0038H and the descriptor table shown below. Calculate the Descriptor address and physical address for the following instruction.*

MOV AX, [10H]



Solution:



0000 0000 0010 0	1	11

RPL =11=3

TI = 1: This means that the segment descriptor is LDT

End GDT = 0000 20FFH, Start (Base) = 0000 2000H

End = Base +limit

0000 20FFH = 0000 2000H + limit ⟶ limit =0000 20FFH- 0000 2000H

Limit = 00FFH

Descriptor (address) = (GDT Base) + LDTR

= 0000 2000H+ 0038H = 0000 2038H

End = Base +limit

End (LDTR) = 0000 2100H+007FH = 0000 217FH

Descriptor (address) = (LDT Base) + Offset

*Offset = Index * 8 = 0000 0000 0000 0100 * 8= 0004H*8 = 0020H*

Descriptor (address) = Base (LDT) + Offset

= 0000 2100H + 0020H = 0000 2120H

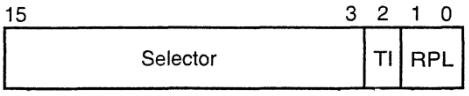
Physical Address = segment Base + Offset

= 0010 0000H + 0010H = 0010 0010H

Example (9): *A Pentium microprocessor is working in protected mode, the data segment register DS content is 2007H, LDTR = 0080H, GDTR= 0060 0000 0FFFH, LDT descriptor= 0000 8240 0000 00F5H and segment descriptor =0040 D320 0000 FFFFH. Determine the physical address affected by the following instruction and the content of this address after execution where AX=220CH?*

MOV [40H], AX

Solution:



0000 0000 0010 0	*1*	*11*

RPL =11=3 lowest level

TI = 1: This means that the segment descriptor is LDT

Index = 0 0100 0000 0000

*Offset = index*8= 0000 0100 0000 0000*8= 0400H*8= 2000H*

GDTR = 0060 0000 0FFFH

GDT (Base) =0060 000H, limit=0FFFH

Descriptor (address) = (GDT Base) + LDTR

= 0060 0000H+ 0080H = 0060 0080H

00	*00*
82	*40*
0000	
00F5	

Base (LDTR) = 00 0040H, limit =00F5H

Descriptor (address) = (LDT Base) + Offset

= 00 0040H + 2000H = 002040H

00	*40*
D3	*20*
0000	
FFFF	

Physical Address = segment Base + Offset

= 0020 0000H +40H = 0020 0040H

0C	0020 0040H
22	0020 0041H
	0020 0042H
	0020 0043H