Unit III - Memory Management

Global Descriptor Table, Local Descriptor Table, Interrupt Descriptor Table, GDTR, LDTR, IDTR. Formats of Descriptors and Selector, Segment Translation, Page Translation, Combining Segment and Page Translation.

Address Translation Overview:

- It uses two modes : **Real Mode and Protected Mode**
- **Real Mode** : Segmentation Unit shifts the selector left 4 bits and adds the result to the offset to from the physical address.
- **Protected Mode** : transforms logical addresses into physical address(actual address).
- 80386 transforms logical addresses into physical address two steps:
- **Segment translation:** a logical address is converted to a linear address.
- **Page translation:** a linear address is converted to a physical address.(optional)
- These translations are performed in a way that is not visible to applications programmers.

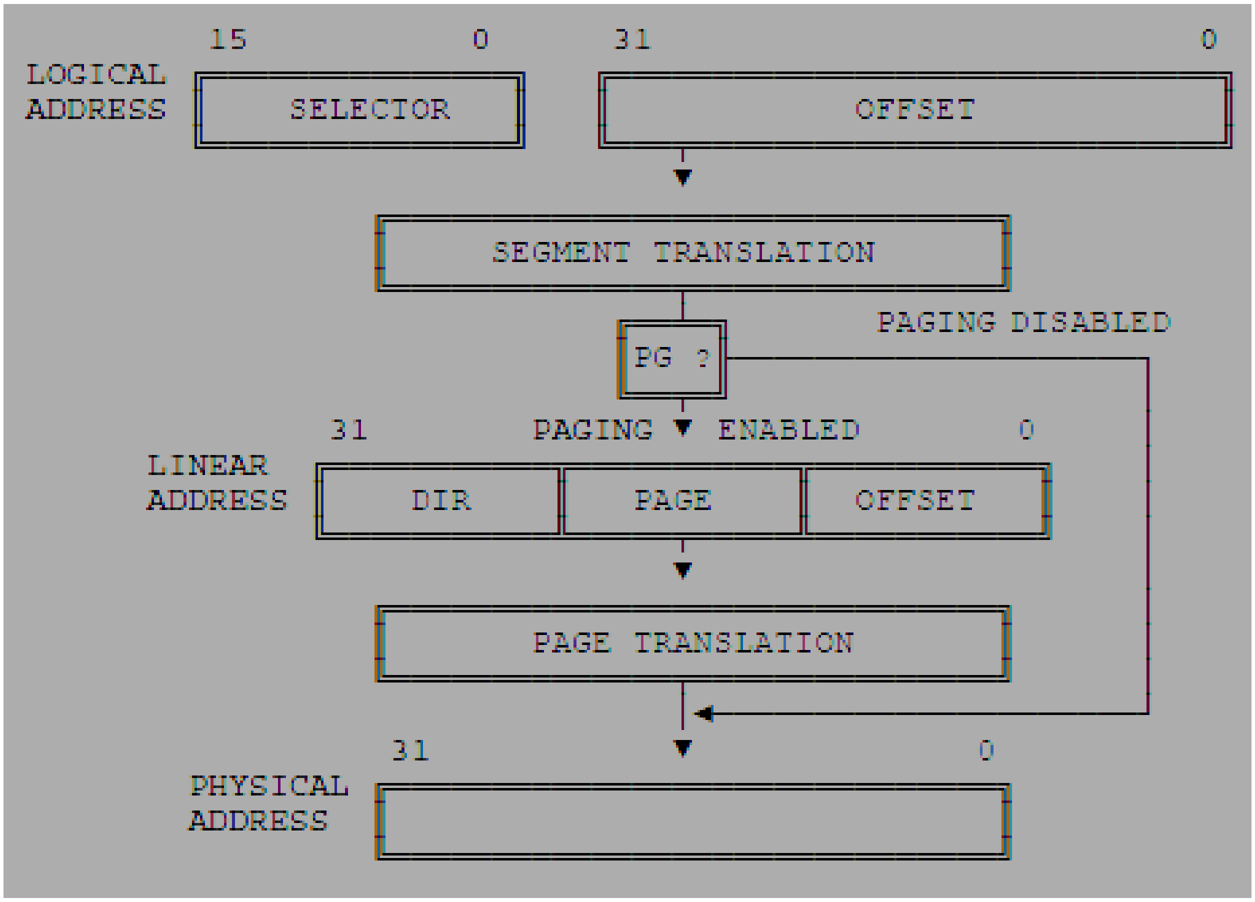


Fig. Address Translation Overview

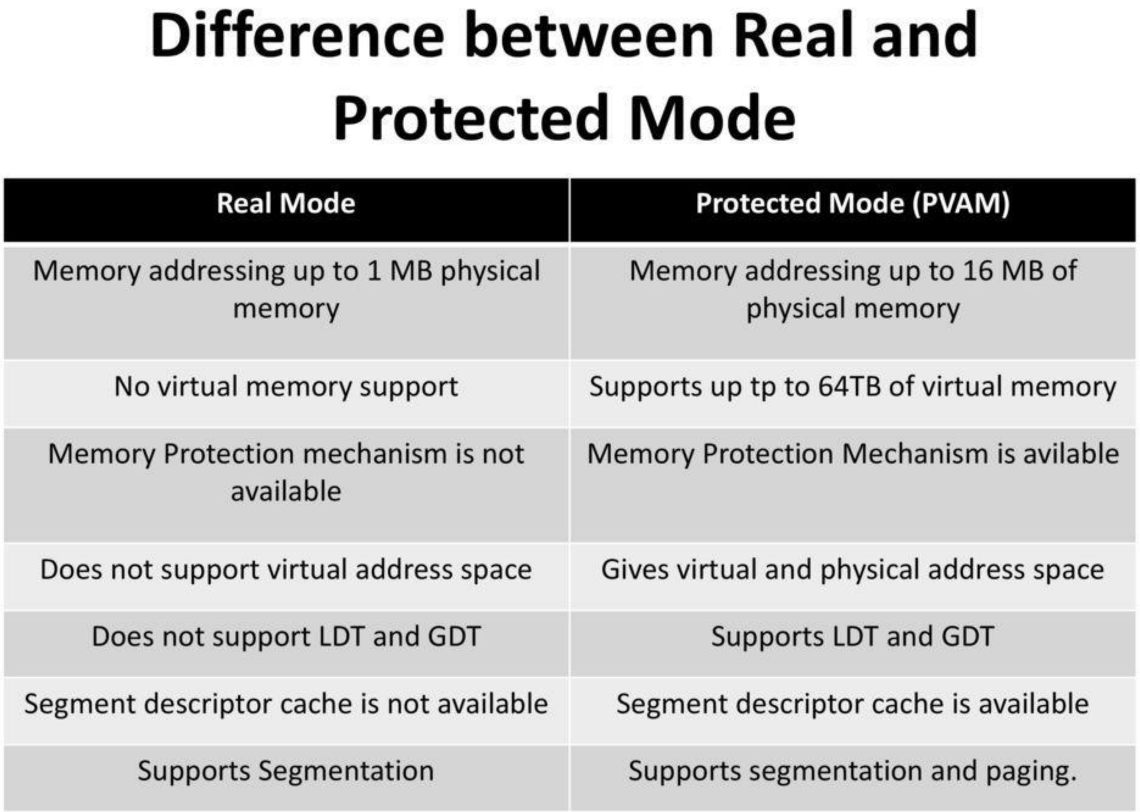
Q.What is a Logical address, Linear address and Physical address?

Ans. logical address/ Virtual address : Logical addresses are used by an application program. They consist of 16-but selector and 32-bit offset.in the flat memory model, the selectors are preloaded into segment registers CS,DS,SS and ES, which all refer to the same linear address.

Virtual address : Virtual address is same as logical address. It will be only used by user mode in the operating system. Operating system will map virtual address from logical address.

Linear address : linear addresses are calculated from virtual/logical addresses by segment translation(Segmentation). The base of the segment referred to by the selectors added to the virtual offset, giving a 32-bit linear address. In Segmentation, when paging is disabled linear address=physical address. It may be in the ram or in the disk.

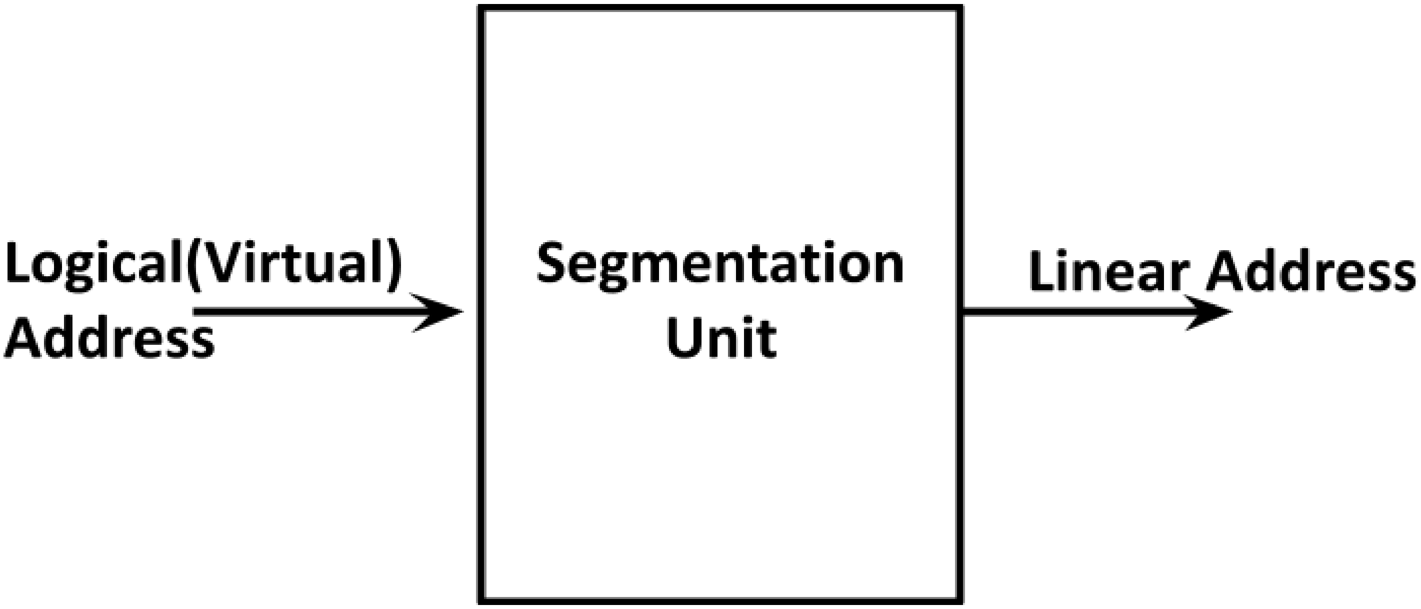
Physical address : Physical address is nothing but, the address value that appears on pins of processor during a memory read / memory write operations. Physical address are calculated from linear addresses through paging. The linear address is used as an index into page table where the CPU locates the corresponding physical address.



Difference between Real and Protected Mode

Real Mode	Protected Mode (PVAM)
Memory addressing up to 1 MB physical memory	Memory addressing up to 16 MB of physical memory
No virtual memory support	Supports up tp to 64TB of virtual memory
Memory Protection mechanism is not available	Memory Protection Mechanism is avilable
Does not support virtual address space	Gives virtual and physical address space
Does not support LDT and GDT	Supports LDT and GDT
Segment descriptor cache is not available	Segment descriptor cache is available
Supports Segmentation	Supports segmentation and paging.

Segmentation :



Segment Translation :

Q. Explain the Segment Translation Process with a neat diagram of 80386.

- Segment Translation is a process of converting logical address into a linear address.

- To perform this translation, the processor uses the following data structures:

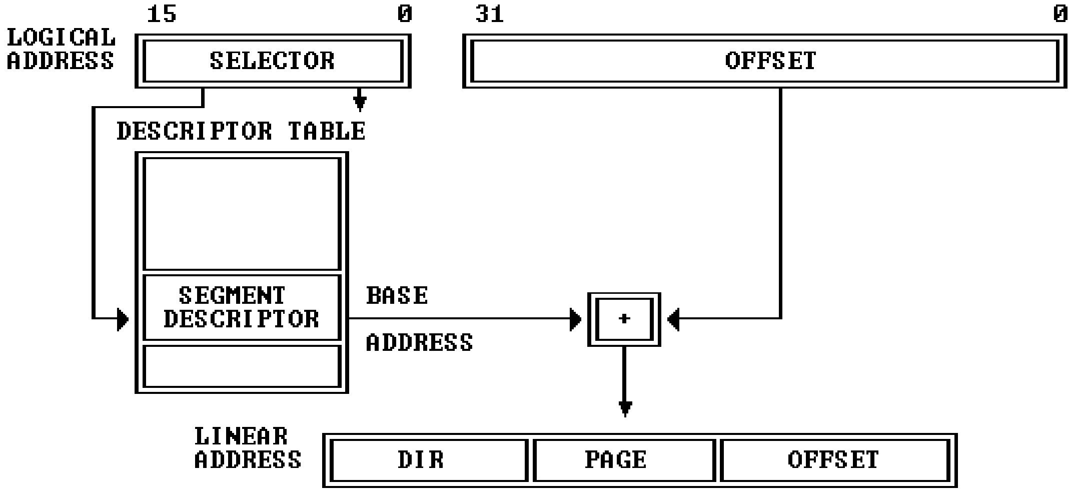
– Descriptors

– Descriptor tables

– Selectors

– Segment Registers

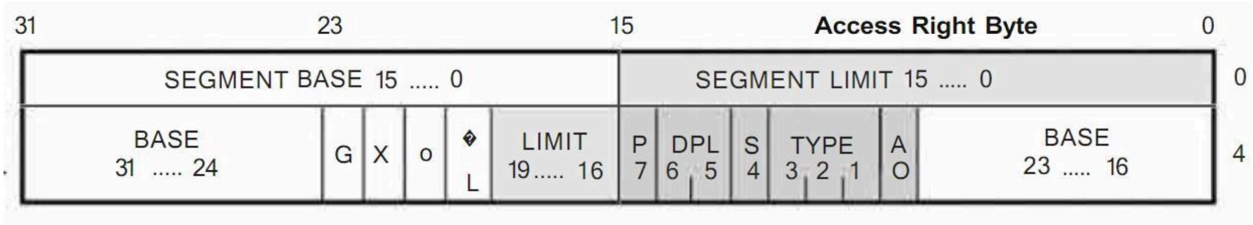
Figure 5-2. Segment Translation



1)Segment Descriptors :

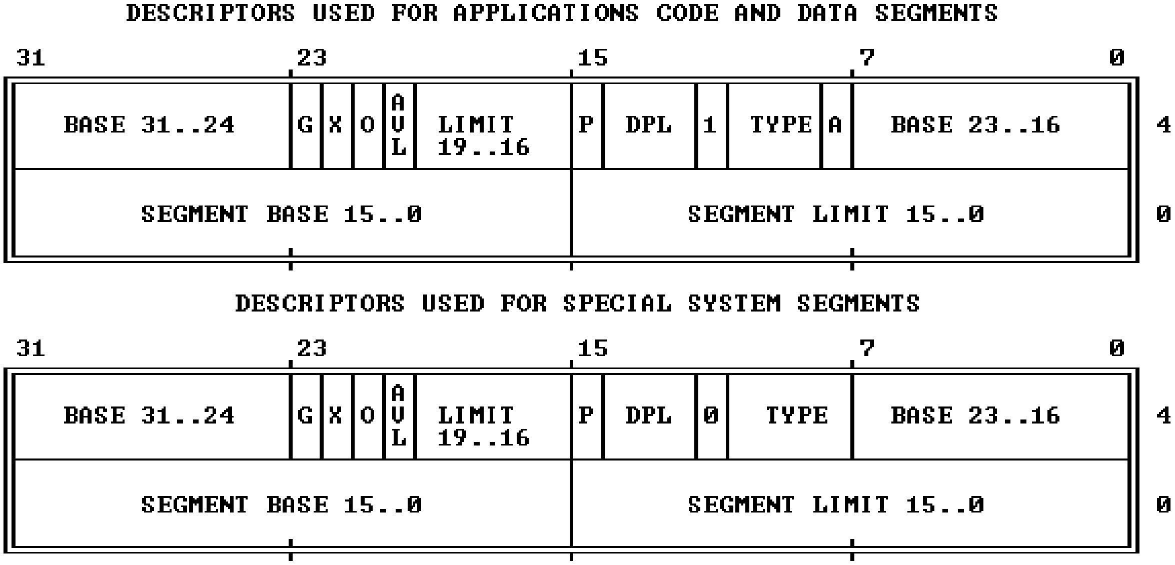
- The segment descriptor provides the processor with the data it needs to map a logical address into a linear address.
- Descriptors are created by compilers, linkers, loaders, or the operating system, not by applications programmers.

General Format of Descriptor :



OR

Figure 5-3. General Segment-Descriptor Format


DESCRIPTORS USED FOR APPLICATIONS CODE AND DATA SEGMENTS

DESCRIPTORS USED FOR SPECIAL SYSTEM SEGMENTS

A	- ACCESSED
AVL	- AVAILABLE FOR USE BY SYSTEMS PROGRAMMERS
DPL	- DESCRIPTOR PRIVILEGE LEVEL
G	- GRANULARITY
P	- SEGMENT PRESENT

BASE: Defines the location of the segment within the 4 gigabyte linear address space.

The processor concatenates the three fragments of the base address to form a single 32-bit value.

TYPE: Distinguishes between various kinds of descriptors.

DPL (Descriptor Privilege Level): Used by the protection mechanism

LIMIT: Defines the size of the segment.

When the processor concatenates the two parts of the limit field, a 20-bit value results.

The processor interprets the limit field in one of two ways, depending on the setting of the granularity bit(G):

- **If G bit 0 :** In units of one byte, to define a limit of up to 1 megabyte.

- **If G bit 1 :** In units of 4 Kilobytes, to define a limit of up to 4 gigabytes.

The limit is shifted left by 12 bits when loaded, and low-order one-bits are inserted.

Granularity bit(G):
It Specifies the units with which the LIMIT field is interpreted.
When the bit is clear(0), the limit is interpreted in units of one byte;
when set, the limit is interpreted in units of 4 Kilobytes.

AVL/U(User bit) :

This bit is completely undefined, and 80386 ignores it.(available for user or OS)

Segment-Present bit(P):

If this bit is zero, the descriptor is not valid for use in address transformation; the processor will signal an exception when a selector for the descriptor is loaded into a segment register Operating systems that implement segment-based virtual memory clear the present bit in either of these cases:

● When the linear space spanned by the segment is not mapped by the paging mechanism.

● When the segment is not present in memory.

Accessed bit(A):

The processor sets this bit when the segment is accessed;

i.e., a selector for the descriptor is loaded into a segment register or used by a selector test instruction. Operating systems that implement virtual memory at the segment level may, by periodically testing and clearing this bit, monitor frequency of segment usage.

2) Descriptor Tables :

- The segment descriptors are grouped together and placed in a continuous memory location and this group arrangement is known as **Descriptor tables** .

- Each descriptor requires 8 bytes in order to store the data of particular segment.

- Total descriptors =8192 and Max length(Size) =64KB.
  E.g: 8 bytes

- A descriptor table is variable in length and may contain up to 8192 (2^13) descriptors.

There are 3 types of descriptor tables.1)GDT 2) LDT 3) IDT

The Global descriptor table (GDT)

- is a general purpose table of descriptors ,

- it can be used by all programs to reference segments of memory.

- Main table & most important one
- The same GDT can be used by all programs to refer to the segment of memory.
- Processor in protected mode can have many LDT's but only one GDT.
- It may contain special system descriptors.

Local descriptor table (LDT)

LDT are set up in the system for individual task or closely related group of tasks.

Multitasking system is defined on a per task basis.

Each task can have access to own private descriptor table(LDT) in addition to GDT.

It can also be shared with other tasks.

Each task can have its own segment of local memory.

So there may be many LDT’s in protected mode, say LDT-0 to LDT-n.

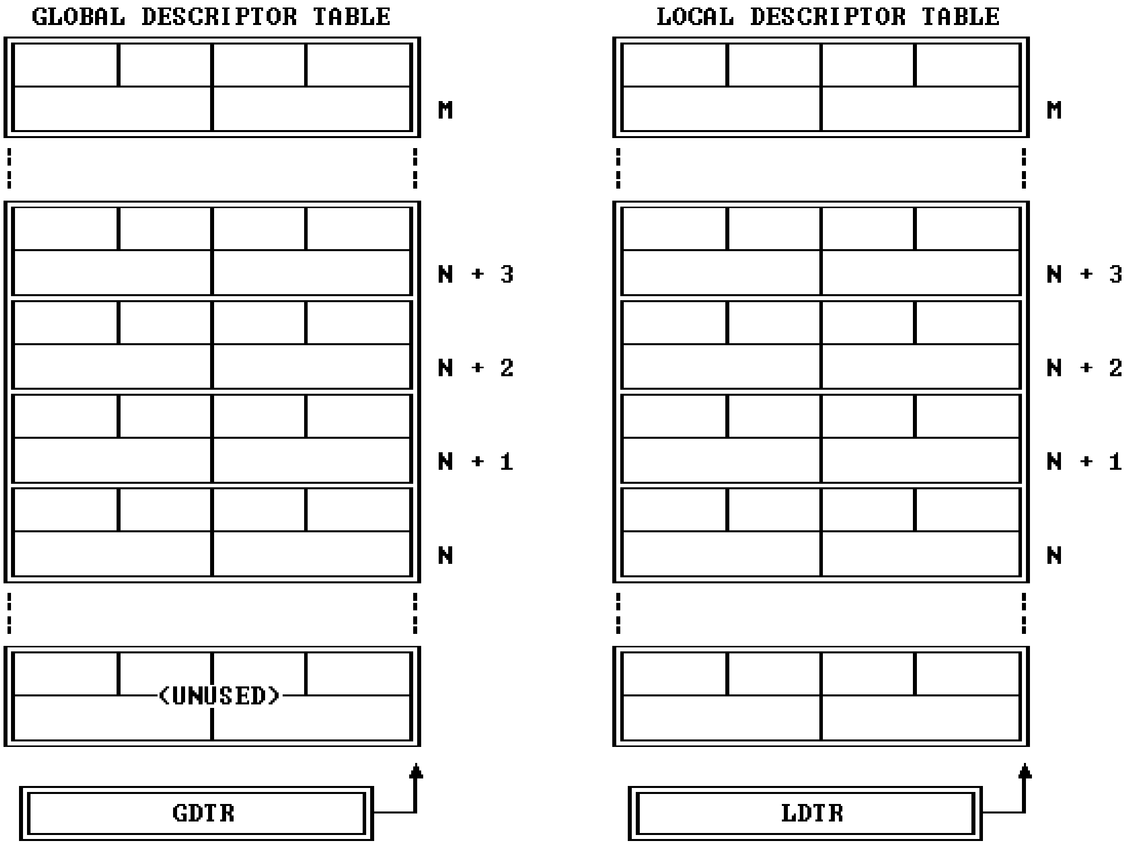
It can be smaller or larger than the GDT

Function: Expand the total number of available descriptors.

The LDT is also called as “private table” which defines a local memory address space for use by the task.

Contains descriptors that provide access to code and data in segments of memory.

Figure 5-5. Descriptor Tables



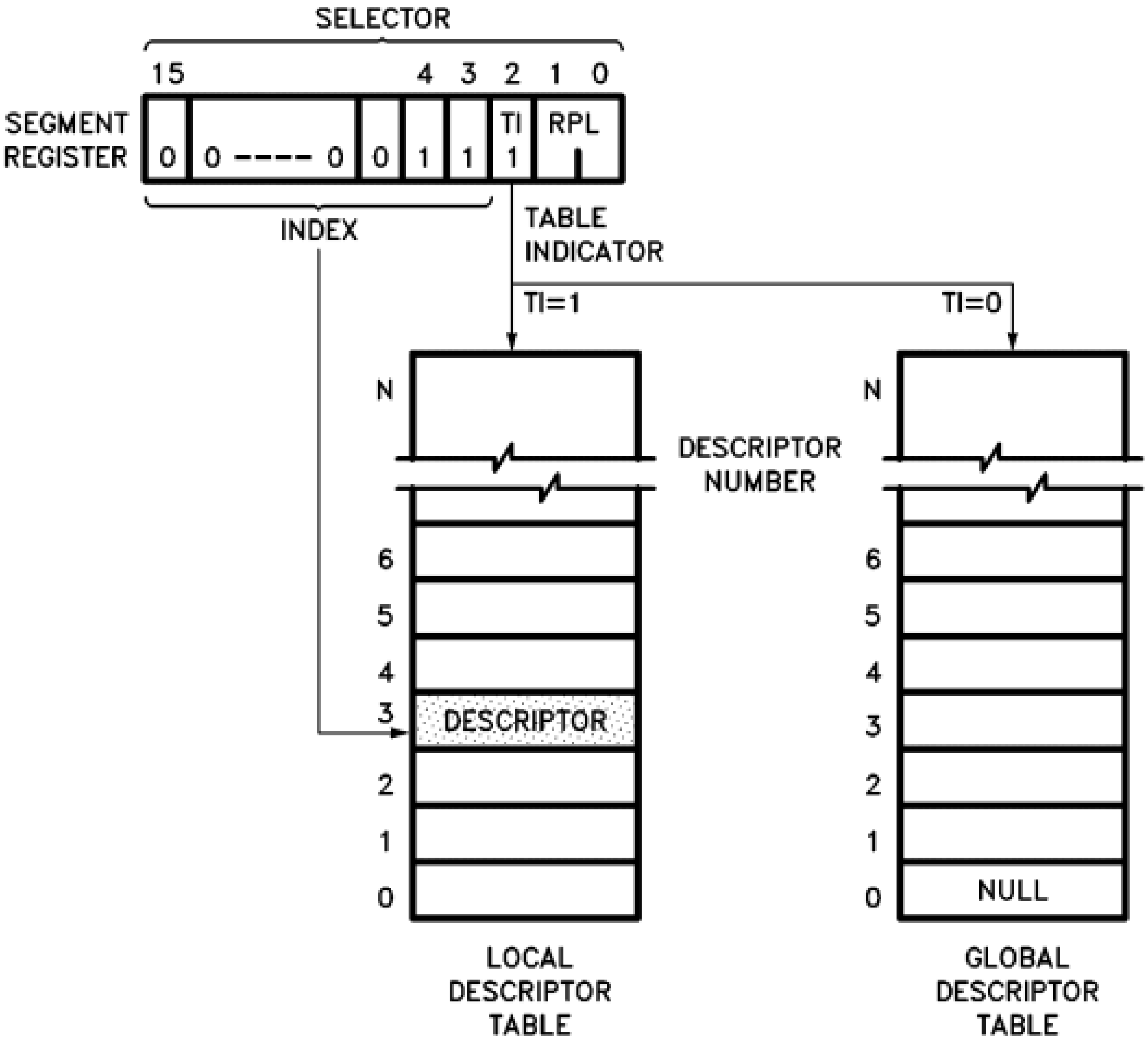


Table Indicator(TI):

- Specifies to which descriptor table the selector refers.
- A **zero** indicates the GDT; a **one** indicates the current LDT.

Interrupt Descriptor Table (IDT)

-Holds the descriptors that are used

1) Trap Gate Descriptor

2) Interrupt Gate Descriptor

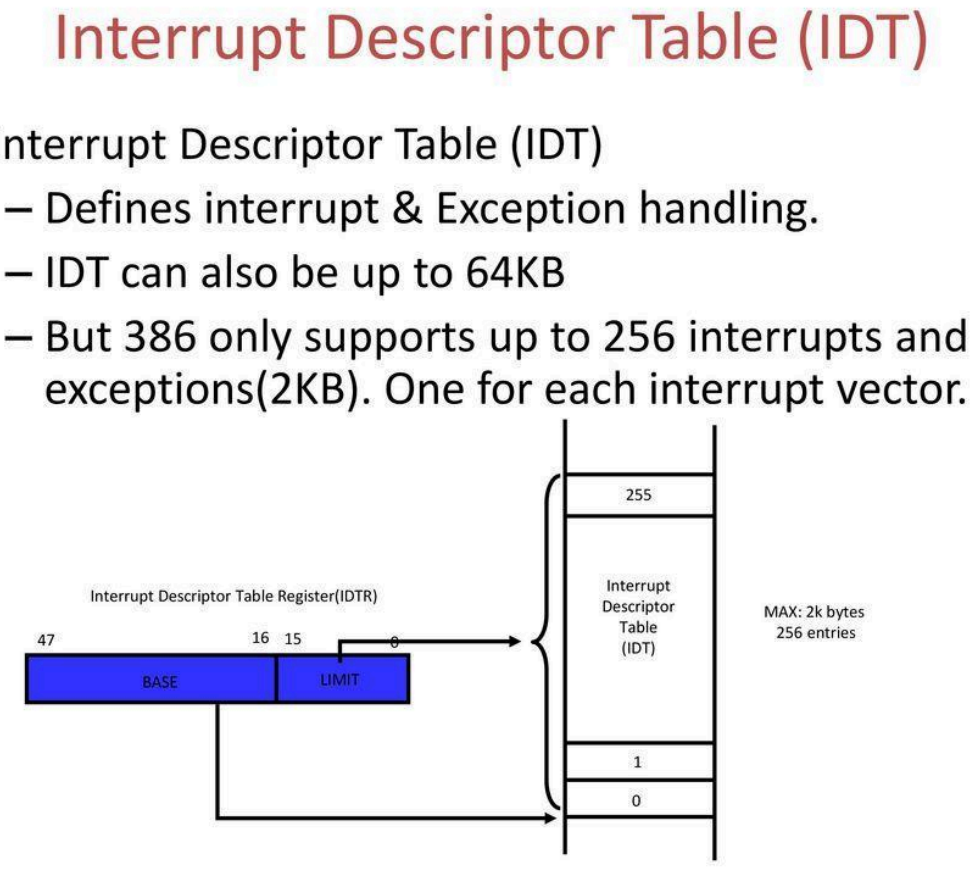
3) Task Gate Descriptor

The user program can never select a descriptor in IDT like GDT or the LDT

It Maintains ISR

The default value that loads into IDTR as

Base address=0, Limit = 03FFH



Interrupt Descriptor Table (IDT)

- Interrupt Descriptor Table (IDT)
  - Defines interrupt & Exception handling.
  - IDT can also be up to 64KB
  - But 386 only supports up to 256 interrupts and exceptions(2KB). One for each interrupt vector.

Interrupt Descriptor Table Register(IDTR)
47
16
15
0
BASE
LIMIT
255
Interrupt Descriptor Table (IDT)
1
0
MAX: 2k bytes 256 entries

Descriptor Register : (GDTR,LDTR,IDTR)

Global Descriptor Table Register (GDTR):

-48 bit register.

-Used to point GDT. Divided into two components viz. Base and limit.

-Base value(32 bit) indicates the starting address of GDT.

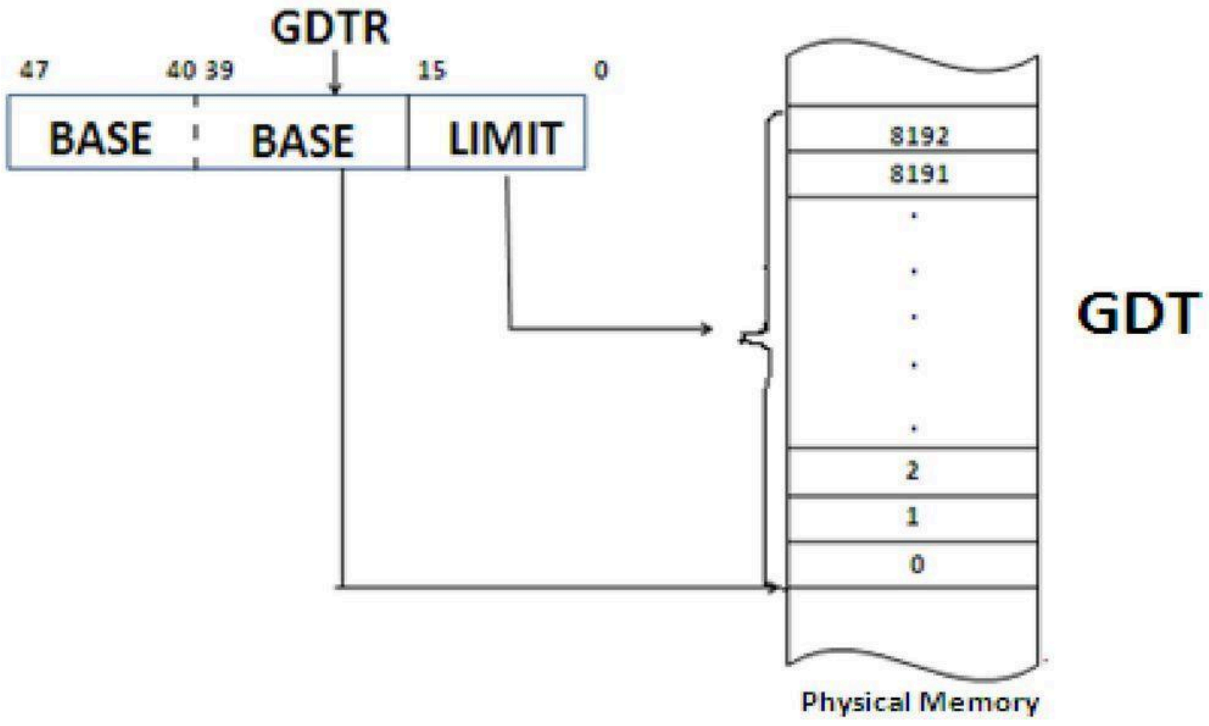
-Limit value(16 bit) indicates the size of GDT.

-Used by OS only(GDTR).Initialized in real mode.

-Defines characteristics of global address space.

-It has no cache register.

Global Descriptor Table Register (GDTR) and GDT



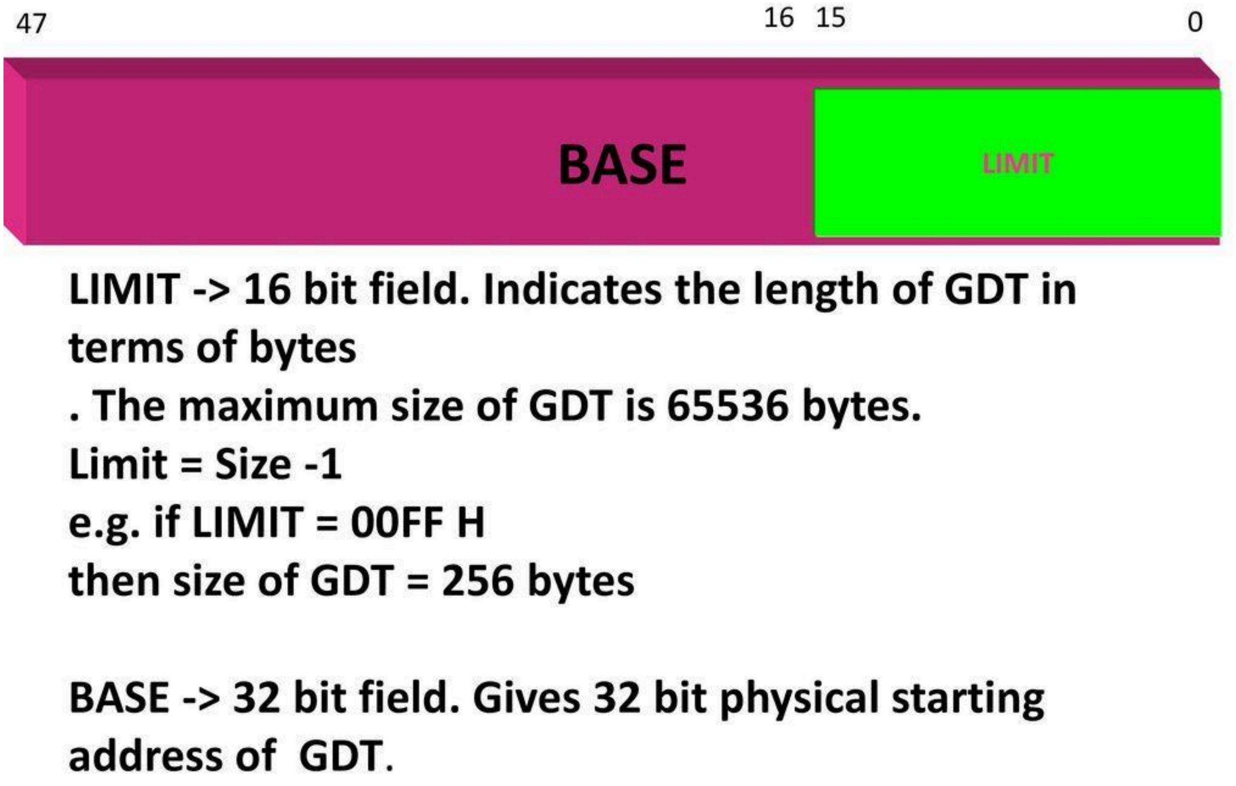
The lower two bytes of this register specifies the LIMIT (in bytes) for the GDT.

The value of limit is 1 less than the actual size of the table.

For example, if LIMIT is 03FFH then the table is 1024(1023 +1) bytes in length (03FFH =102316).

Since the LIMIT field is 16 bit long, the GDT can grow up to 65,536 bytes long.

The upper four bytgs of GDTR specifies the 32-bit linear address of the base of the Global Descriptor Table (GDT).



LIMIT -> 16 bit field. Indicates the length of GDT in terms of bytes
. The maximum size of GDT is 65536 bytes.
Limit = Size -1
e.g. if LIMIT = 00FF H
then size of GDT = 256 bytes

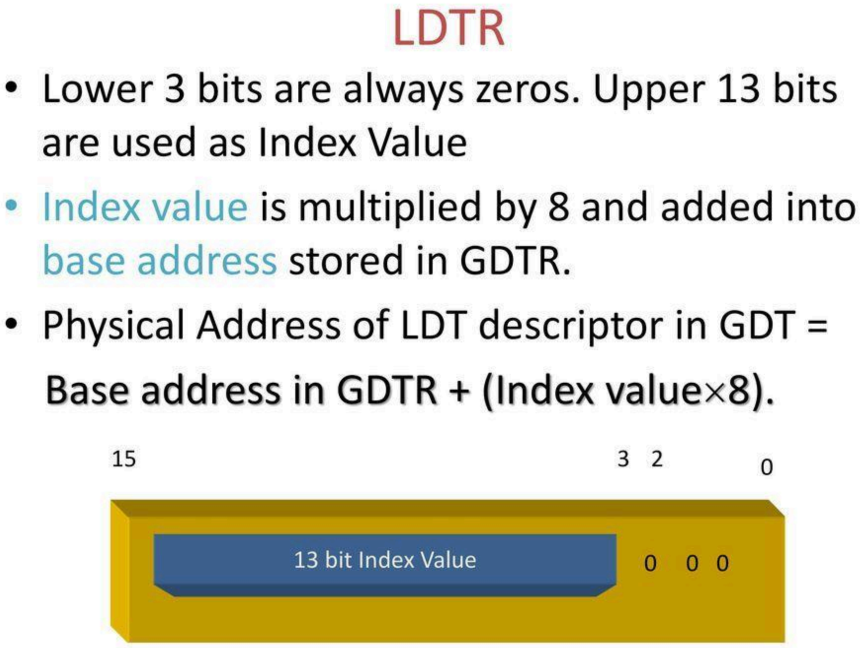
BASE -> 32 bit field. Gives 32 bit physical starting address of GDT.

2) Local Descriptor Table Register :

-This **register holds the 32-bit** base address, **16-bit segment limit**, and **16-bit segment selector** for the **local descriptor table (LDT).**

-The segment which contains the LDT has a segment descriptor in the GDT.

- There is no segment descriptor for the GDT.
- Used as a local selector.
- Points LDT descriptor stored in GDT.
- GDT contains many LDT descriptors.
- Each LDT has LDT descriptor in GDT.
- Points only one LDT descriptor at a time.
- Used to change LDT.
- Provides 48 bit cache register.
- A 48 bit cache register is used to hold current LDT descriptor.
- When a reference is made to data in memory, a segment selector is used to find a segment descriptor in the GDT or LDT.
- A segment descriptor contains the base address for a segment



LDTR

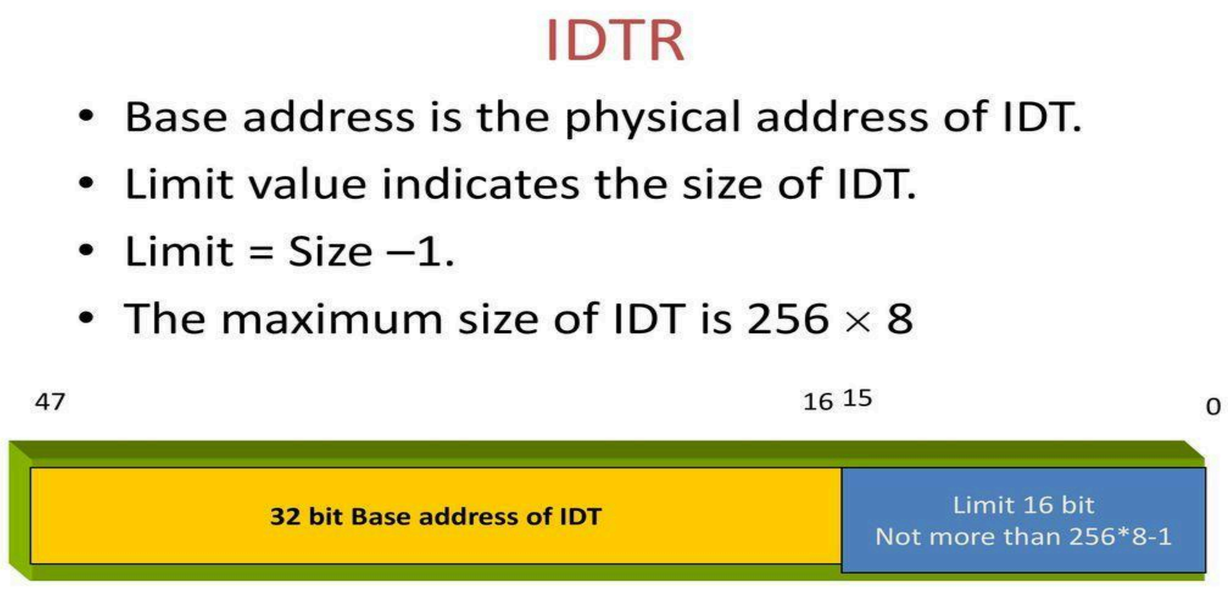
- Lower 3 bits are always zeros. Upper 13 bits are used as Index Value
- Index value is multiplied by 8 and added into base address stored in GDTR.
- Physical Address of LDT descriptor in GDT = Base address in GDTR + (Index value×8).

15
3 2
0
13 bit Index Value
0 0 0

3) Interrupt Descriptor Table Register :

- 48 bit register.Points IDT.
- Divided into two components viz. Base and limit.Base value(32 bit) indicates the starting address of IDT.
- Limit value(16 bit) indicates the size of IDT.Used by interrupts and exceptions only.

- It has no cache register.
- The descriptors used in the IDT are called as “interrupt gates” which gives the beginning of an interrupt-service routine(ISR).
- This register holds the 32-bit base address and 16-bit segment limit for the interrupt descriptor table (IDT).
- When an interrupt occurs, the interrupt vector is used as an index to get a gate descriptor from this table.
- The gate descriptor contains a far pointer used to start up the interrupt handler.

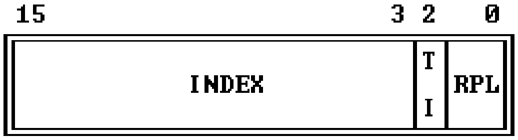


IDTR

- Base address is the physical address of IDT.
- Limit value indicates the size of IDT.
- Limit = Size –1.
- The maximum size of IDT is 256 × 8

3) Selectors :

Figure 5-6. Format of a Selector



TI - TABLE INDICATOR
RPL - REQUESTOR'S PRIVILEGE LEVEL

- The selector portion of a logical address identifies a descriptor by specifying a descriptor table and indexing a descriptor within that table.
- Selectors may be visible to applications programs as a field within a pointer variable, but the values of selectors are usually assigned (fixed up) by linkers or linking loaders

Index :

- Selects one of 8192 descriptors in a descriptor table.
- The processor simply multiplies this index value by 8 (the length of a descriptor), and adds the result to the base address of the descriptor table in order to access the appropriate segment descriptor in the table.

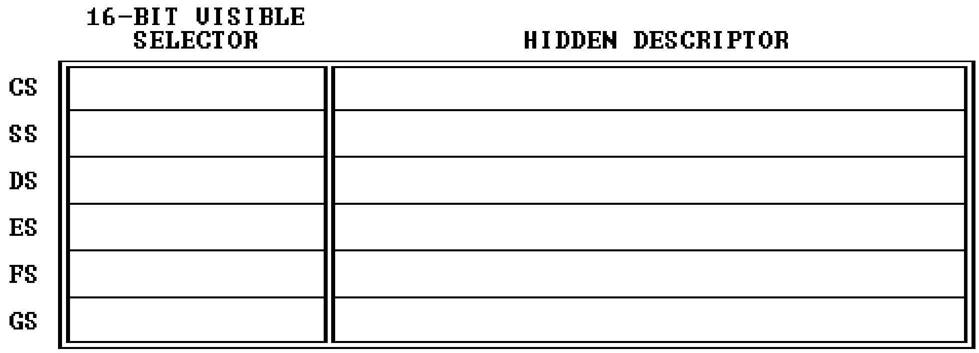
Table Indicator(TI):

- Specifies to which descriptor table the selector refers.
- A **zero** indicates the GDT; a **one** indicates the current LDT.

Requested Privilege Level(RPL): Used by the protection mechanism.

4) Segment Registers :

Figure 5-7. Segment Registers

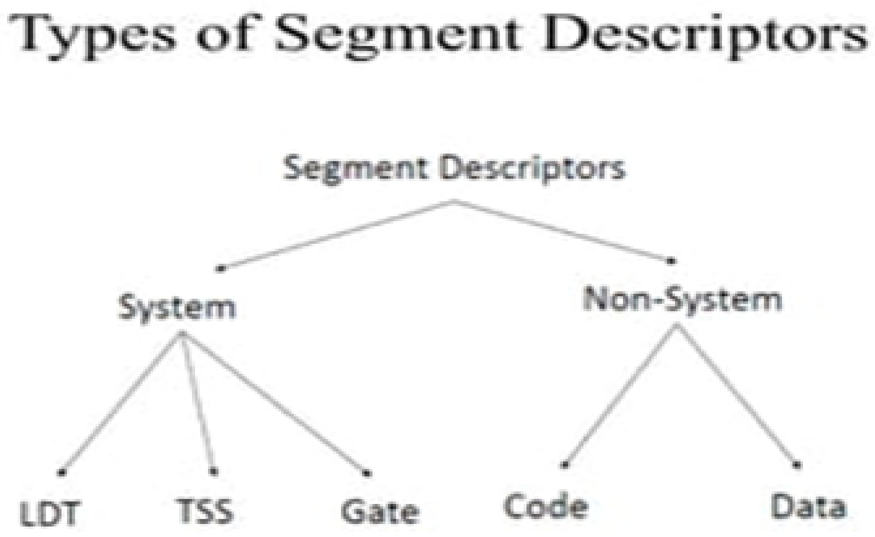


- The 80386 stores information from descriptors in segment registers, thereby avoiding the need to consult a descriptor table every time it accesses memory.
- Every segment register has a **"visible"** portion and an **"invisible"** portion.
- The visible portions of these segment address registers are manipulated by **programs as if they were simply 16-bit registers**.
- The invisible portions are manipulated by the **processor.**

The operations that load these registers are normal program instructions. These instructions are of two classes:

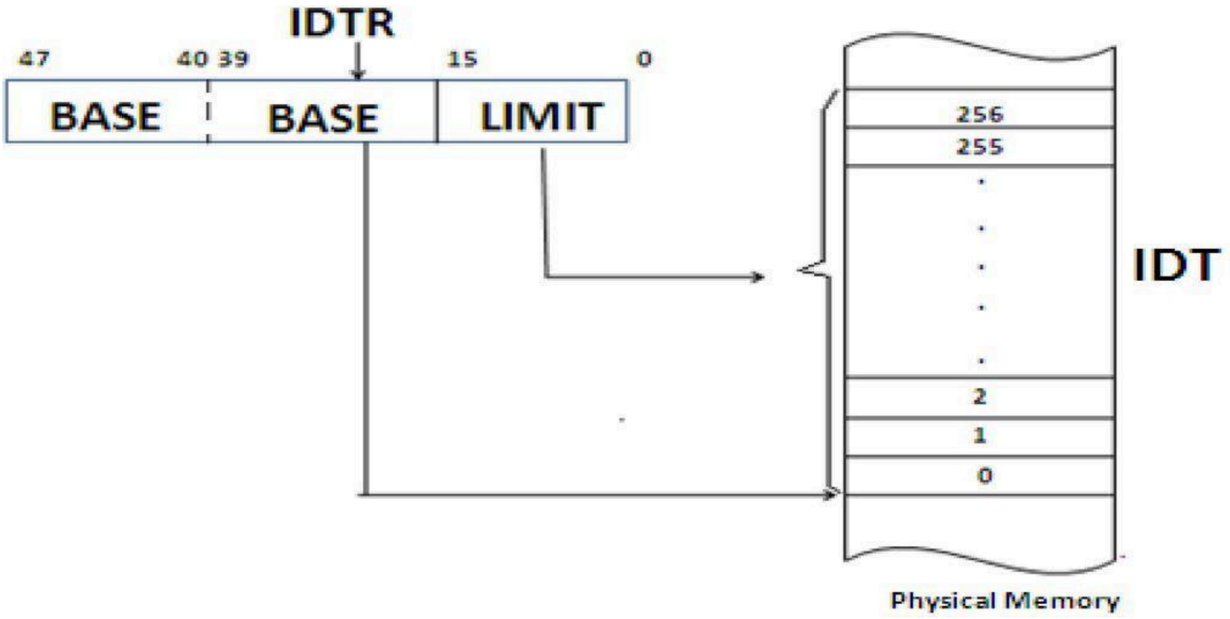
1. **Direct load instructions**; for example, MOV, POP, LDS, LSS, LGS, LFS. These instructions explicitly reference the segment registers.
2. **Implied load instructions**; for example, far CALL and JMP. These instructions implicitly reference the CS register, and load it with a new value.

Types of Segment Descriptors and Their Formats :

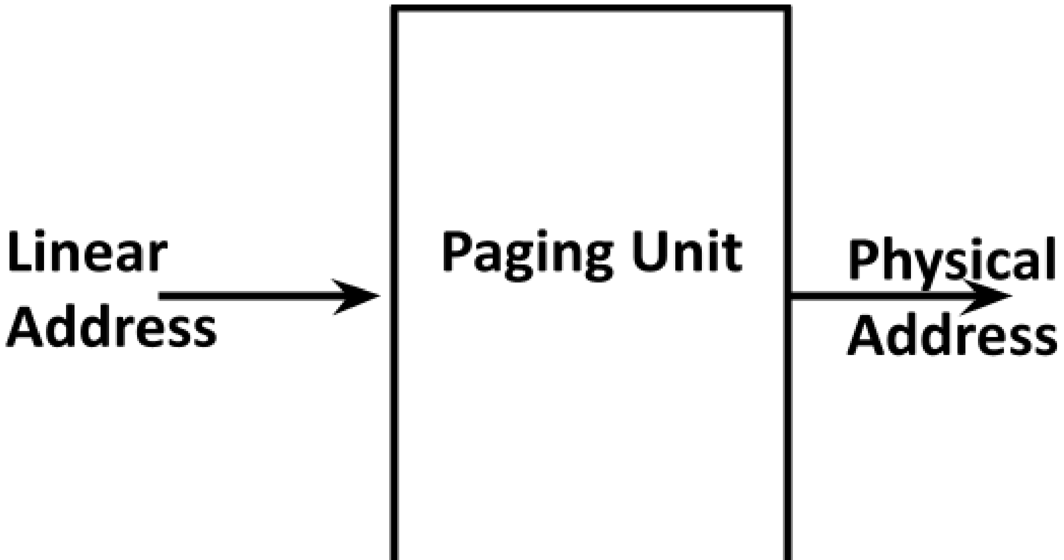


Types of Segment Descriptors

Interrupt Descriptor Table Register (IDTR) and IDT



Page Translation :



In the second phase of address transformation, the 80386 transforms a linear address into a physical address.
This phase of address transformation implements the basic features needed for page-oriented virtual-memory systems and page-level protection.

The page-translation step is optional.

Page translation is in effect only when the PG bit of CR0 is set. This bit is typically set by the operating system during software initialization.

The PG bit must be set if the operating system is to implement multiple virtual 8086 tasks, page-oriented protection, or page-oriented virtual memory.

Components of Paging Mechanism

- Page Directory
- Page tables
- Page frame

Page Frame

A page frame is a 4K-byte unit of contiguous addresses of physical memory.

Pages begin on byte boundaries and are fixed in size.

Linear Address

A linear address refers indirectly to a physical address by specifying a page table, a page within that table, and an offset within that page.

Figure 5-8. Format of a Linear Address

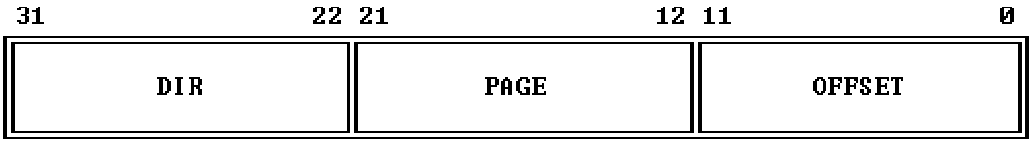


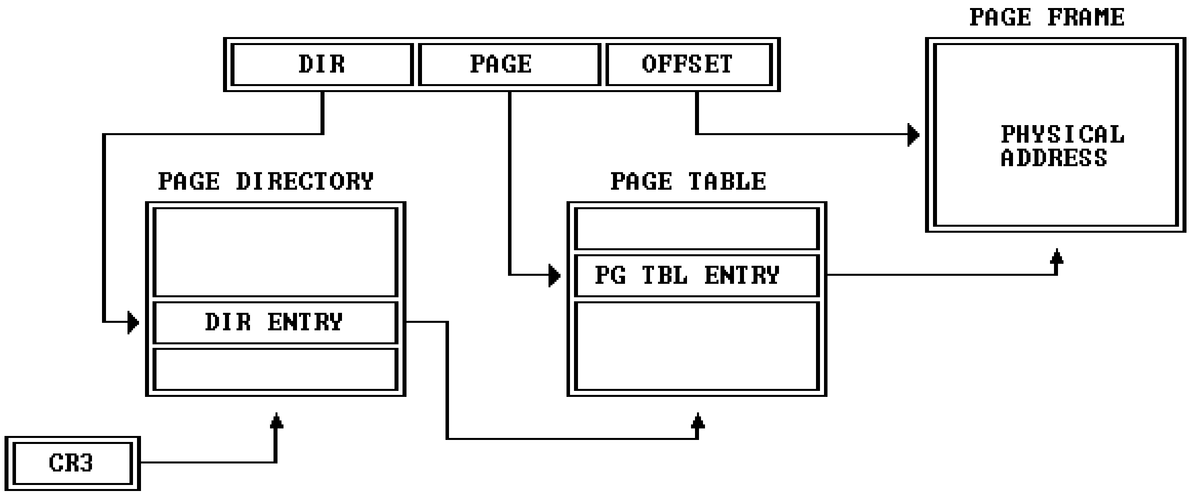
Fig. shows how the processor converts the DIR, PAGE, and OFFSET fields of a linear address into the physical address by consulting two levels of page tables.

The addressing mechanism uses the **DIR field** as an index into a page directory,

uses the **PAGE field** as an index into the page table determined by the page directory,

and uses the **OFFSET field** to address a byte within the page determined by the page table.

Figure 5-9. Page Translation



Page Tables :

A page table is simply an array of 32-bit page specifiers.
A page table is itself a page, and therefore contains 4 Kilobytes of memory or at most 1K 32-bit entries.

Two levels of tables are used to address a page of memory.

At the higher level is a page directory. The page directory addresses up to 1K page tables of the second level.

A page table of the second level addresses up to 1K pages.

All the tables addressed by one page directory, therefore, can address 1M pages (2^(20)).

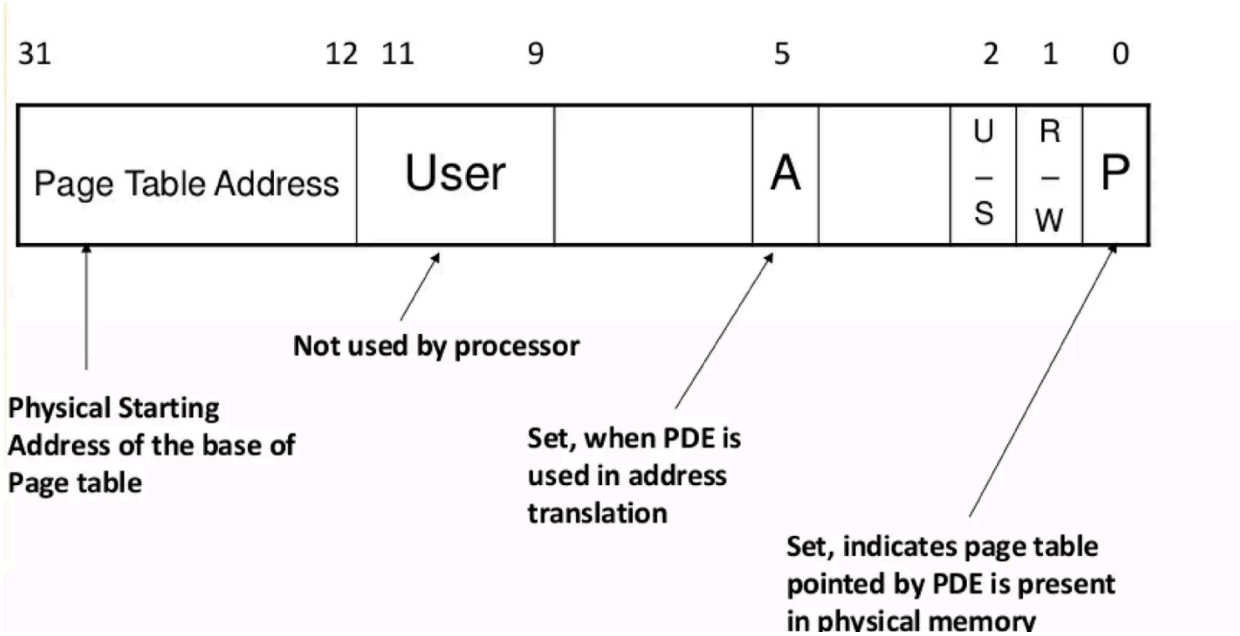
Because each page contains 4K bytes 2^(12) bytes), the tables of one page directory can span the entire physical address space of the 80386 (2^(20) times 2^(12) = 2^(32)).

The physical address of the current page directory is stored in the CPU register CR3, also called the page directory base register (PDBR).

Memory management software has the option of using one page directory for all tasks, one page directory for each task, or some combination of the two

Page Directory (PDE):

PDE Descriptor



Not used by processor

Physical Starting Address of the base of Page table

Set, when PDE is used in address translation

Set, indicates page table pointed by PDE is present in physical memory

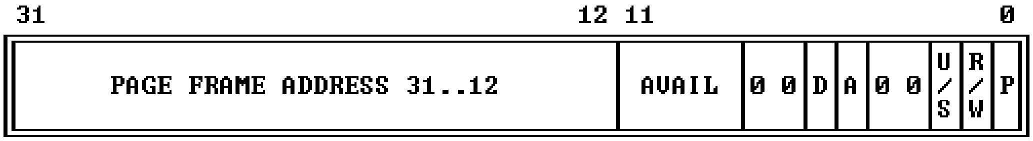
- PDE:-It is at the most **4KB in size** and allows upto (32 bit) **1024** entries are allowed.
- The **upper 10 bits** of the linear address are used as an index to corresponding page directory entry
- Page directory entry points to page tables.

Page-Table Entries(PTE):

Entries in either level of page tables have the same format.

A page directory entry is having six fields.

Figure 5-10. Format of a Page Table Entry



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.

1) Page Frame Address

The page frame address specifies the physical starting address of a page.

Because pages are located on 4K boundaries, the low-order 12 bits are always zero.

In a page directory, the page frame address is the address of a page table.

In a second-level page table, the page frame address is the address of the page frame that contains the desired memory operand.

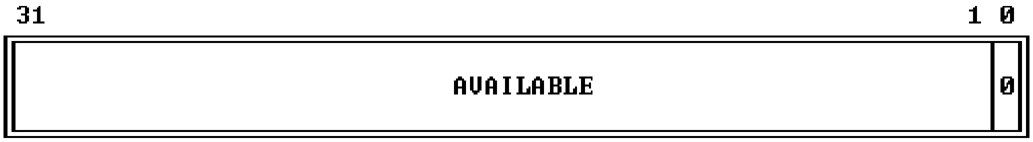
2) Present Bit

The Present bit indicates whether a page table entry can be used in address translation.

P=1 indicates that the entry can be used.

When P=0 in either level of page tables, the entry is not valid for address translation, and the rest of the entry is available for software use; none of the other bits in the entry is tested by the hardware.

Figure 5-11. Invalid Page Table Entry



3) Accessed and Dirty Bits

- **A (Accessed) Bit:** It is set before any access to the page.

- **D (Dirty) bit:** It is set before a write operation to the page is carried out. The D bit is undefined for PDEs.
- **The 80386 never clears this bit.**

The processor sets the corresponding accessed bits in both levels of page tables to one before a read or write operation to a page.

The processor sets the dirty bit in the second-level page table to one before a write to an address covered by that page table entry.

The dirty bit in directory entries is undefined.

4) User/Supervisor Bits :

if set : pages covered by this entry are accessible to all users if clear : only for PL0,1,2

5) Read/Write and User/Supervisor Bits

These bits are not used for address translation, but are used for page-level protection, which the processor performs at the same time as address translation .

If U/S` is clear R/W` has no effect But if U/S`=1 then pages covered by this entry will be write protected.

If R/W` is set : write privileges are allowed from PL3 code

U/S`	**R/W`**	**Permitted level**	**Permitted access levels 0,1,2**
0	0	None	Read/Write
0	1	None	Read/Write
1	0	Read-Only	Read/Write
1	1	Read/Write	Read/Write (PL0)

6)User/Avail :

Bits 9,10,11 are not used by the 80386. Users are free to use them.

Page Translation Cache / Translation Lookaside Buffer(TLB) :

- TLB has 4 sets of eight entries each.
- Each entry consists of a TAG and a DATA.
- Tags are 24 bit wide. They contain 20 upper bits of linear address, a **valid bit** (Validation of Entry) and three attribute **bits(D,U/S and R/W**)
- Data portion of each entry contains upper 20 bits of the Physical address.

• For greatest efficiency in address translation, the processor stores the most recently used page-table data in an on-chip cache.

• Only if the necessary paging information is not in the cache must both levels of page tables be referenced.

•The 80386DX paging mechanism has designed to support demand paged virtual memory systems.

• Performance would degrade substantially if the processor was required to access two levels of tables (Page directory and page table) for every memory access.

- To solve this problem, the 80386DX stores the most recently used page table entries in an on-chip cache.
- This cache is called the Translation Lookaside Buffer (TLB).
- The TLB holds up to 32 page table entries.
- The 32-entry TLB coupled with a 4K page size, results in coverage of 128K bytes of memory addresses.
- Whenever program generates linear address that maps to a page table entry (PTE) already in the cache, the 80386DX can use the cached information it has internally.
- This saves two outside memory references, improving performance in address translation.

The page-translation cache can be flushed by either of two methods:

1. By reloading CR3 with a MOV instruction;

   for example: MOV CR3, EAX

2. By performing a task switch to a TSS that has a different CR3 image than the current TSS .

Combining Segment and Page Translation :

Figure 5-12. 80306 Addressing Machanism

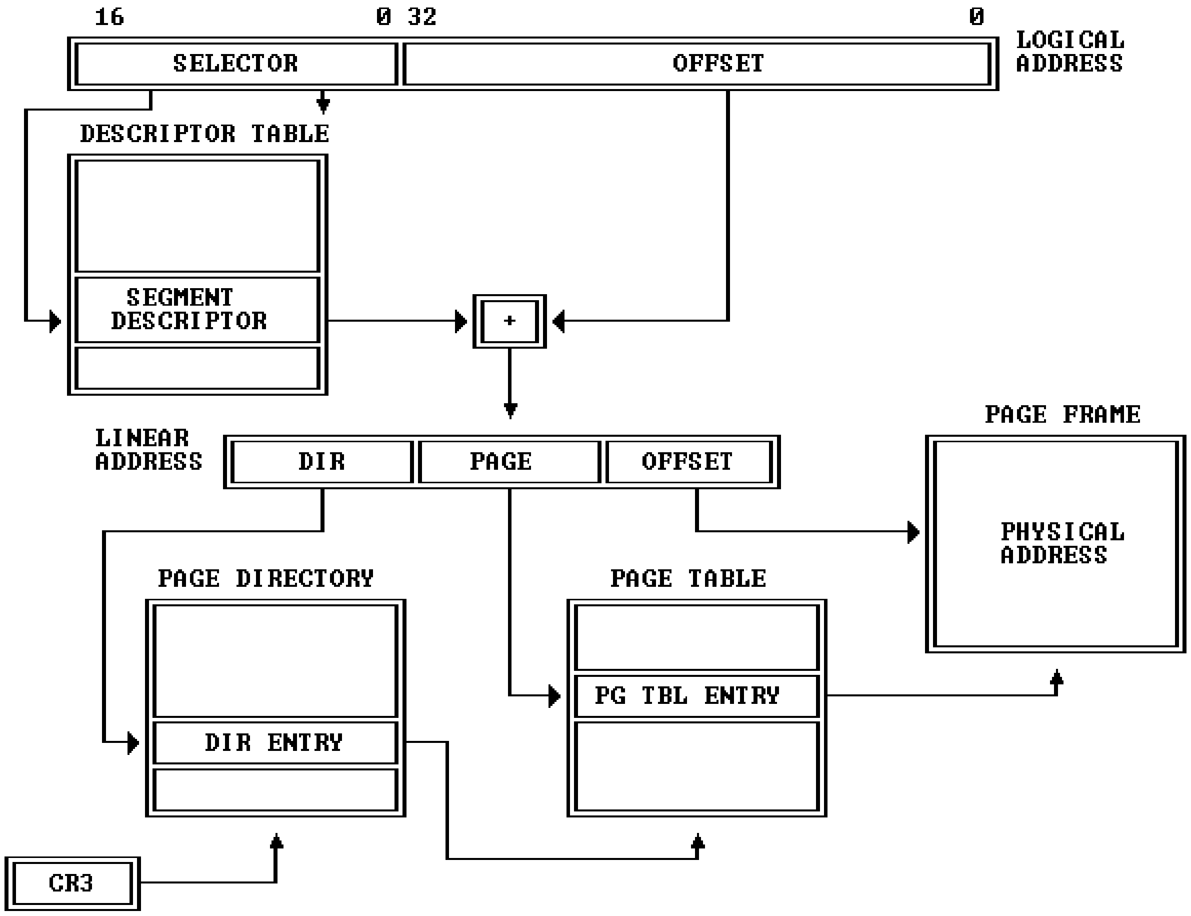


Fig. shows both phases of the transformation from a logical address to a physical address when paging is enabled.

In a combined paging/segmentation system a user’s address space is broken up into a number of segments. Each segment is broken up into a number of fixed-sized pages which are equal in length to a main memory frame Segmentation is visible to the programmer Paging is transparent to the programmer

By appropriate choice of options and parameters to both phases, memory-management software can implement several different styles of memory management.