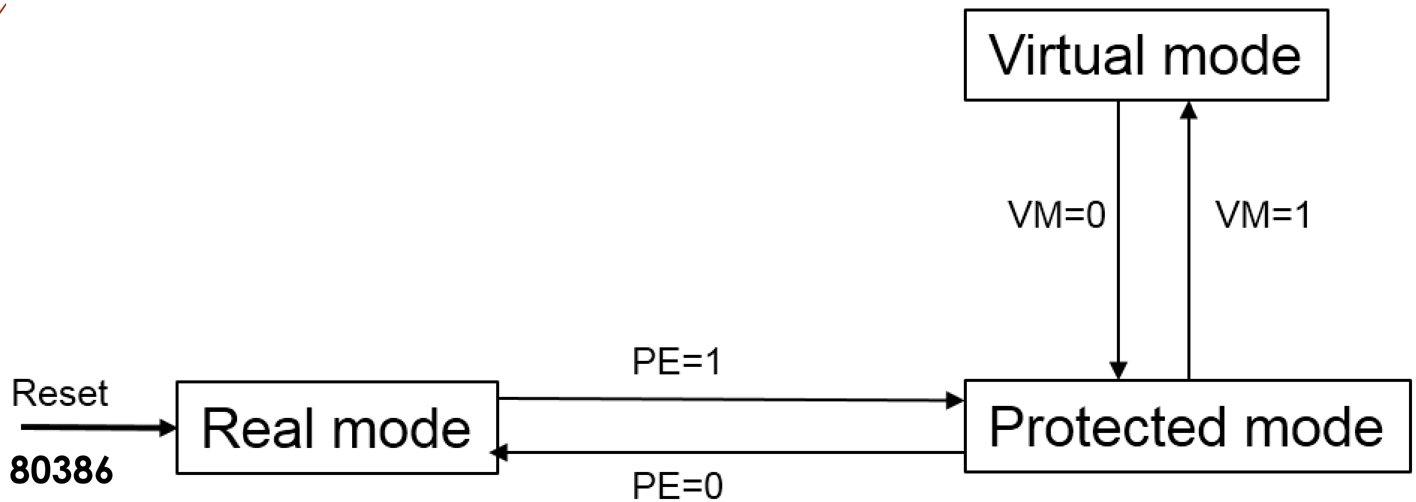
The 80386 Microprocessor Family

Processing modes of 80386

- The processing modes determine the features that are accessible
  - Real address mode
  - Protected mode
  - Virtual 8086 mode



Real address mode

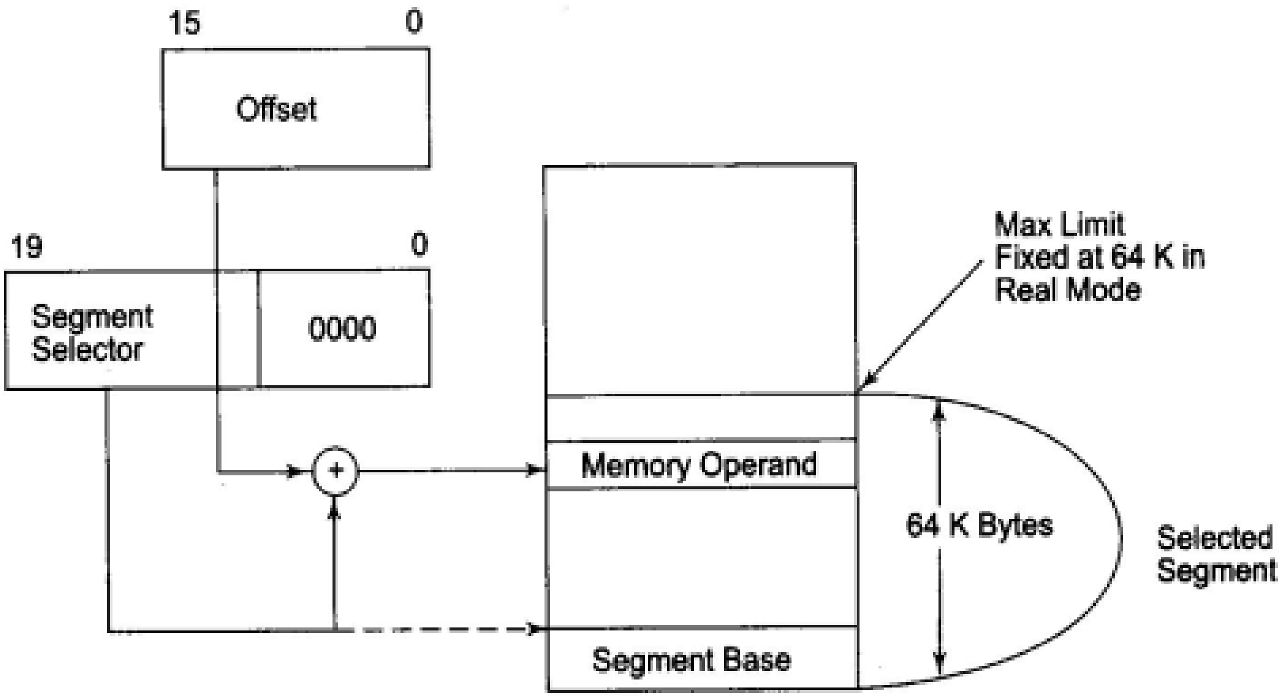
- After RESET, processor is always in real mode.
- In real mode, 80386 will correctly execute the program designed for 8086.
- After reset, the 80386 starts from memory location FFFFFFF0H under the real address mode. Here, 80386 works as a **fast 8086** with 32-bit registers and data types.
- The memory is sub divided into segments.
- Segment can start at any base address (multiple of 10H) in memory.

Real address mode

- Overlapping (partial/fully), disjoint segments are allowed.
- The segment size in real mode is 64KB, hence the 32-bit effective addressing must be less than 0000FFFFH.
- The real mode initializes the 80386 and prepares it for protected mode.

Real address mode

- Addressing in real mode
  - 16 bit segment register, used to determine the linear base address
  - 16 bit offset register



Physical Address Formation in Real Mode of 80386 (Intel Corp.)

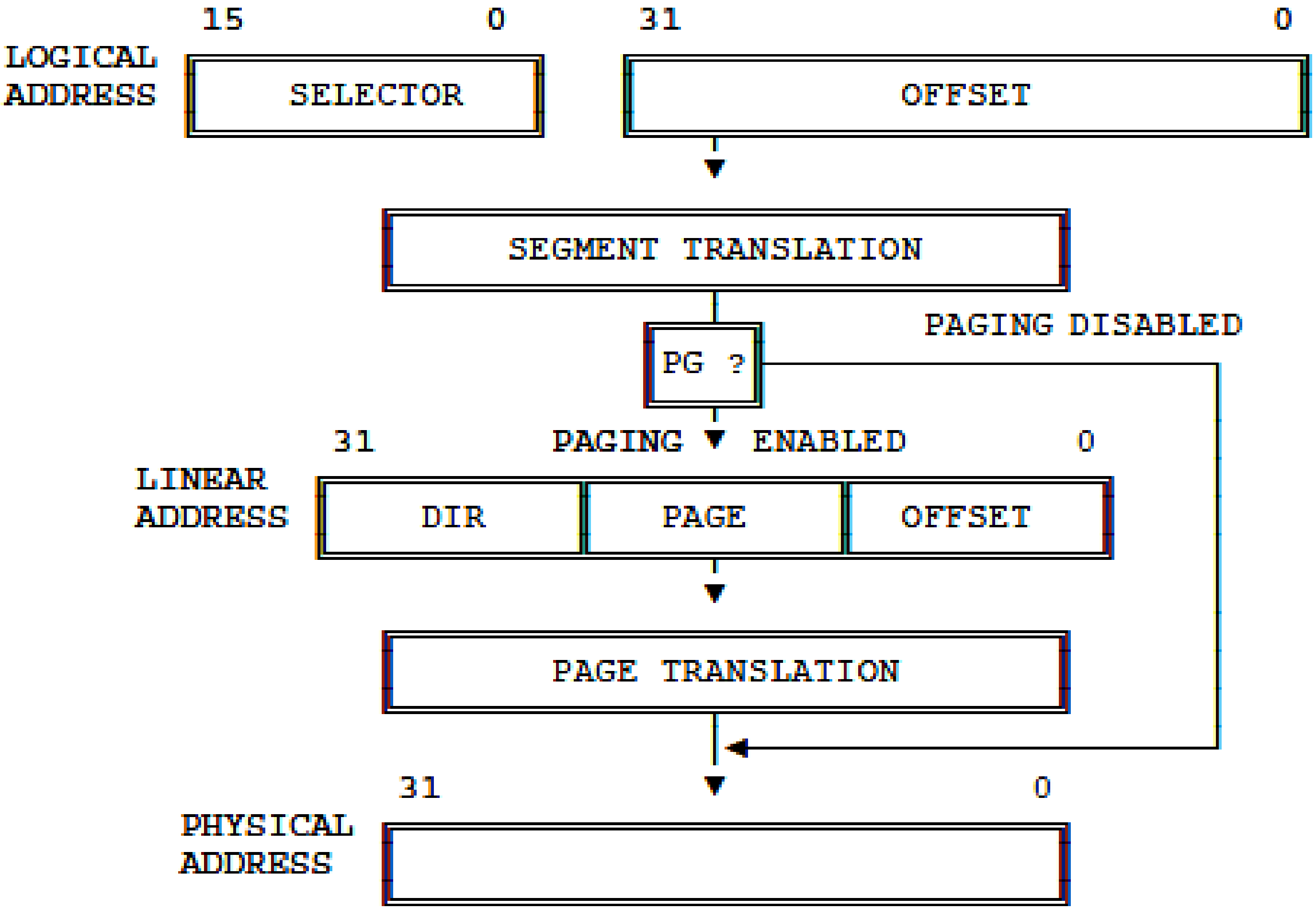
Real address mode

- Addressing in real mode
  - In real mode, the 80386 can address at the most **1Mbytes** of **physical memory** using address lines A_0-A_{19} .
  - **Paging unit** is **disabled** in real addressing mode, hence real addresses are same as physical addresses.
  - The segment in 80386 real mode can be read, written or executed. No protection is available.
  - Any access past the end of the segment limit generates exception 13 in real address mode.

Protected mode

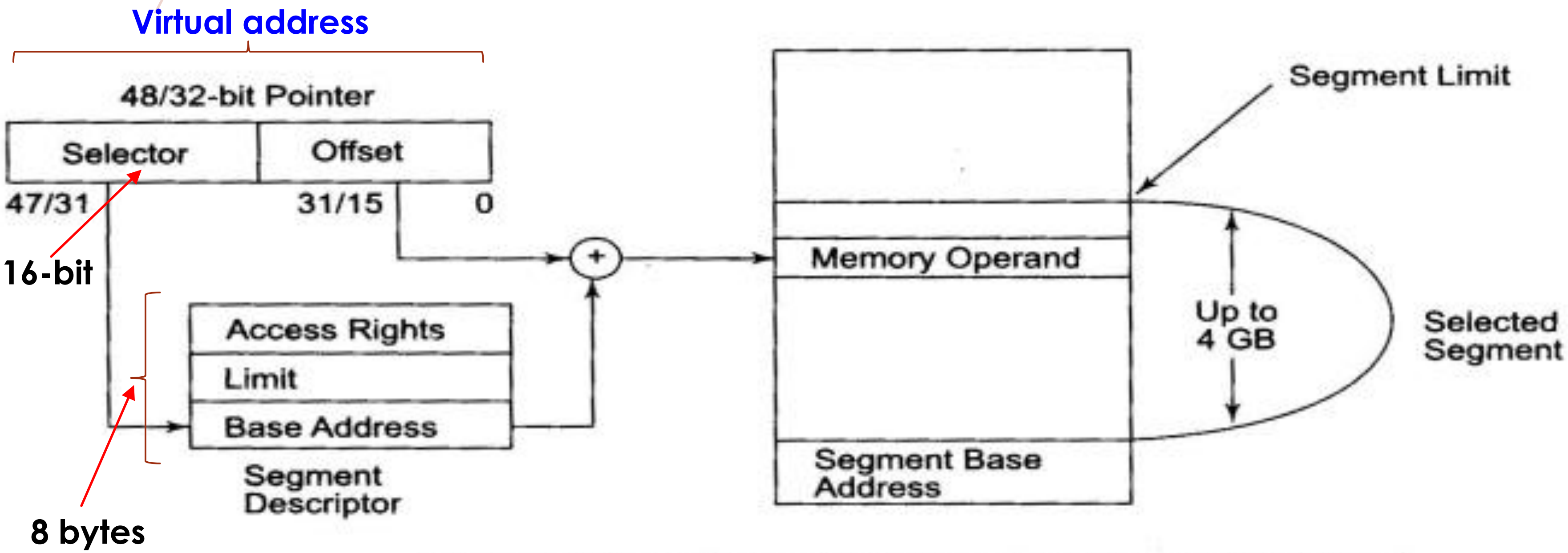
- All the capabilities of 80386 are available for utilization in its protected mode of operation.
- The protected mode allows the use of additional instructions, addressing modes and capabilities of 80386.
- Addressing in protected mode -
  - In this mode, the contents of segment registers are used as **selectors** to segment descriptors.
  - **Segment descriptors** contains segment limit, base address and access rights bytes of the segment.
  - The offset is added with segment base address to calculate linear address.

Virtual to Linear Address Translation

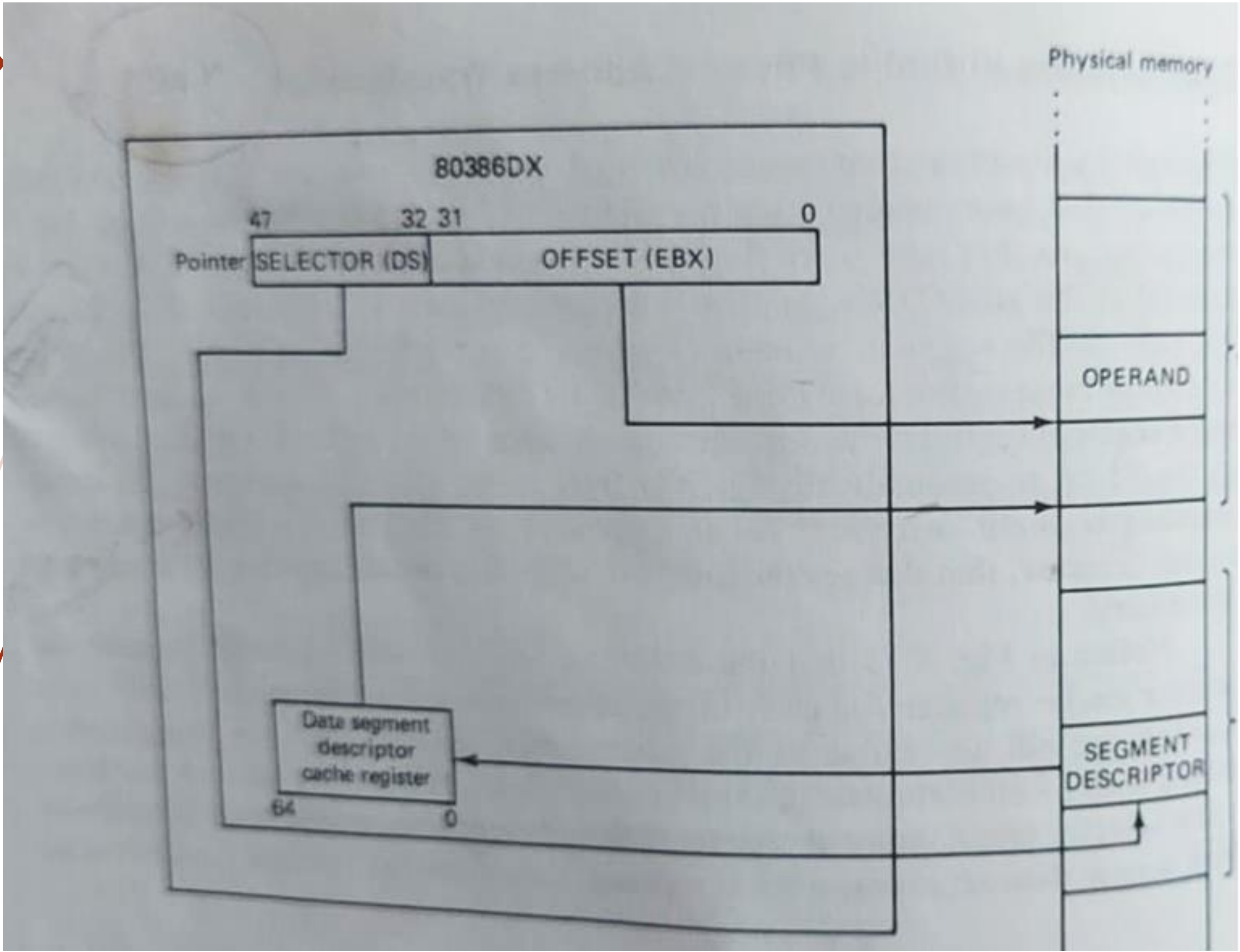


Protected mode

- Addressing in protected mode
  - If paging unit is disabled, then linear address is same as physical address.



Protected Mode Addressing without Paging Unit (Intel Crop.)



Data segment

Local descriptor table

Selector (D0-D15)

- RPL – 2 request privilege level bits (D0-D1)
- Table select bit TI (D2) : 0->GDT, 1->LDT
- The no. of segments is decided by D2-D15 -> 14bits
- The total no. of segments = 2^{14} = 16,384 = 16K

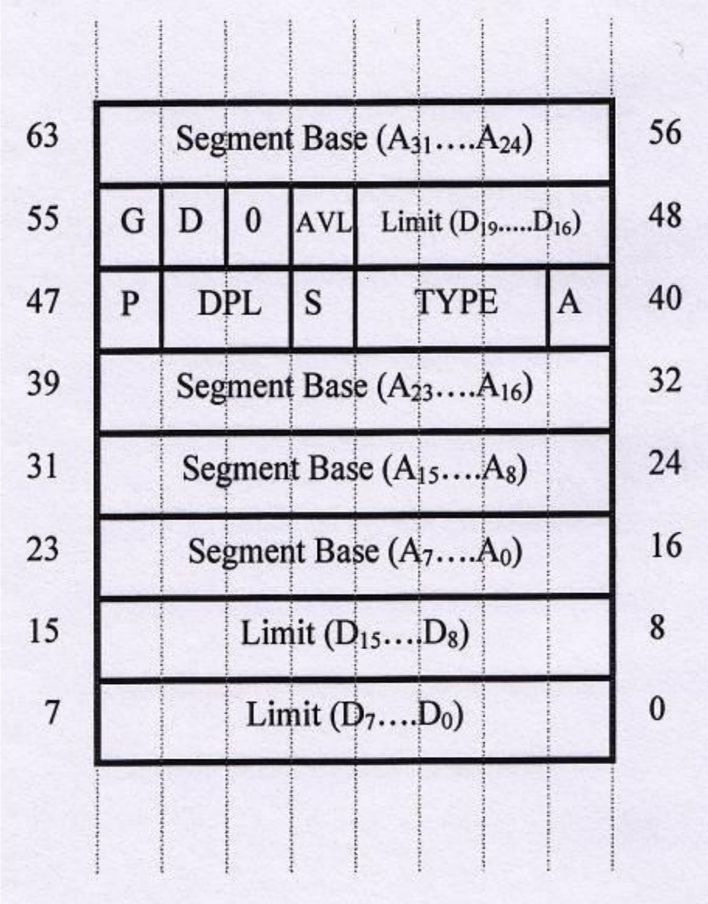
Segmentation

- Segmentation offers protection to different types of data and code.
- Descriptor tables
  - Global Descriptor Table (GDT)
  - Local Descriptor Table (LDT)
  - Interrupt Descriptor Table (IDT)

Segmentation

➢ Descriptors

- ❖ The 80386 descriptors are 8 byte quantities containing **20-bit segment limit**, **32-bit segment address** and **12-bits for access rights**.

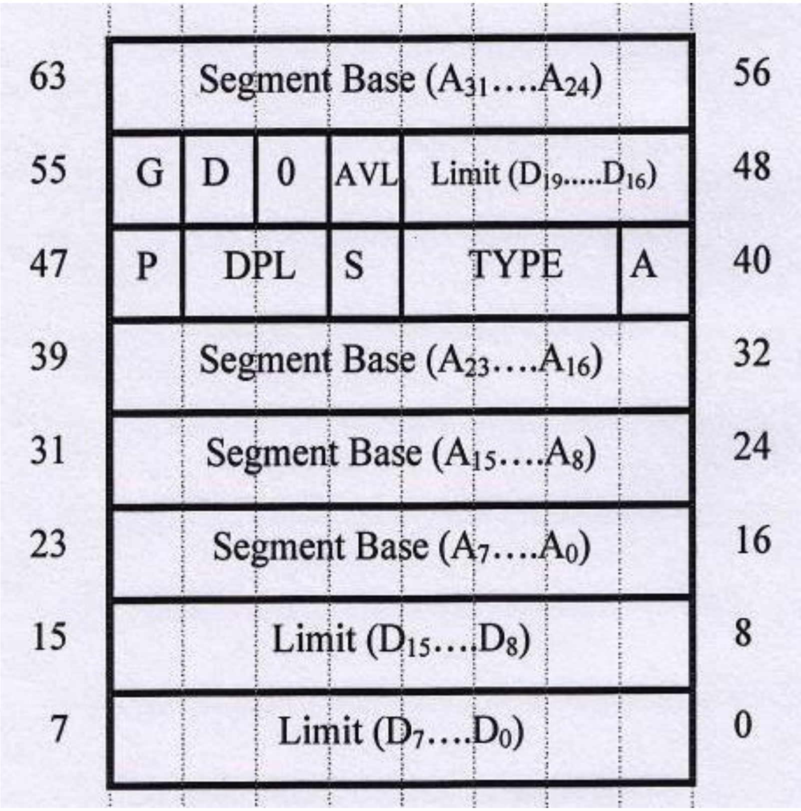


<table>
<tr><td>63</td><td colspan="8">Segment Base (A_{31}….A_{24})</td><td>56</td></tr>
<tr><td>55</td><td>G</td><td>D</td><td>0</td><td>AVL</td><td colspan="4">Limit (D_{19}.....D_{16})</td><td>48</td></tr>
<tr><td>47</td><td>P</td><td colspan="2">DPL</td><td>S</td><td colspan="3">TYPE</td><td>A</td><td>40</td></tr>
<tr><td>39</td><td colspan="8">Segment Base (A_{23}….A_{16})</td><td>32</td></tr>
<tr><td>31</td><td colspan="8">Segment Base (A_{15}….A_{8})</td><td>24</td></tr>
<tr><td>23</td><td colspan="8">Segment Base (A_{7}….A_{0})</td><td>16</td></tr>
<tr><td>15</td><td colspan="8">Limit (D_{15}….D_{8})</td><td>8</td></tr>
<tr><td>7</td><td colspan="8">Limit (D_{7}….D_{0})</td><td>0</td></tr>
</table>

8 byte = 64-bits

Segmentation

- Descriptors



63	Segment Base (A_{31}….A_{24})					56
55	G	D	0	AVL	Limit (D_{19}.....D_{16})	48
47	P	DPL	S	TYPE	A	40
39	Segment Base (A_{23}….A_{16})					32
31	Segment Base (A_{15}….A_{8})					24
23	Segment Base (A_{7}….A_{0})					16
15	Limit (D_{15}….D_{8})					8
7	Limit (D_{7}….D_{0})					0

BASE	Base Address of the segment
LIMIT	The length of the segment
P	Present Bit-1 = Present, 0 = Not Present
DPL	Descriptor Privilege Level 0-3
S	Segment Descriptor-0 = System Descriptor, 1 = Code or Data Segment Descriptor
TYPE	Type of Segment
A	Accessed Bit
G	Granularity Bit-1 = Segment length is page granular, 0 = Segment length is byte granular
D	Default Operation Size (recognized in code segment descriptors only)-1 = 32-bit segment, 0 = 16-bit segment
0	Bit must be zero (0) for compatibility with future processors
AVL	Available field for user or OS

Structure of an 80386 Descriptor (Intel Corp.)

- **Type** Bits
  - E, ED/C, RW
  - E = 0 (data or stack) ,
    - ED = 0 Expand Up (data segment)
    - ED =1 Expand down (stack Segment)
    - W = 0 Data seg Read only , W = 1 may be Written into
  - E =1 (code seg)
    - C = 1 Conforming; Code seg may only be executed if Current PL>=Descriptor PL
    - R = 1 may be read

Segmentation

- The five types of descriptors that 80386 has are
  - Code or data segment descriptors
  - System descriptors
  - Local descriptors
  - TSS (Task State Segment) descriptors
  - GATE descriptors

Segmentation

- Base address is 32-bit and limit is 20-bit. Hence, maximum segment size can be 1MB.
- 80386 can handle 16K descriptors and hence 16K segments.
- 80386 may be able to address a virtual memory of 16K × 1MB = 16GB per task if operated with segmentation scheme.
- In case of operation under paging scheme, the memory is managed in terms of segments of 4GB size which are further organized in terms of pages of 4KB size each.
- The page table has entries for pages. The page table can store 1024 entries for pages.
- Page table directory contains entries for page table (1024 entries)

Segmentation

- Thus, the page management structure can have 1024 × 1024 = 1M such page entries under each segment.
- Each page is of 4KB size.
- Thus 1M × 4KB = 4GB can be maximum size of each segment.
- The 80386 can thus address maximum of 16K × 4GB = 64TB of virtual memory per task.

Paging

- Segmentation scheme may divide the physical memory into variable size segments but the paging divides the memory into fixed size pages.
- The segments are supposed to be logical segments of the program, but the pages do not have any logical relation with the program.
- The pages are just the fix size portions of the program module or data.
- The **advantage** of the paging scheme is that the complete segment of a task need not be in the physical memory at any time. Only a few pages of the segments, which are required currently for the execution need to be available in the physical memory.

Paging

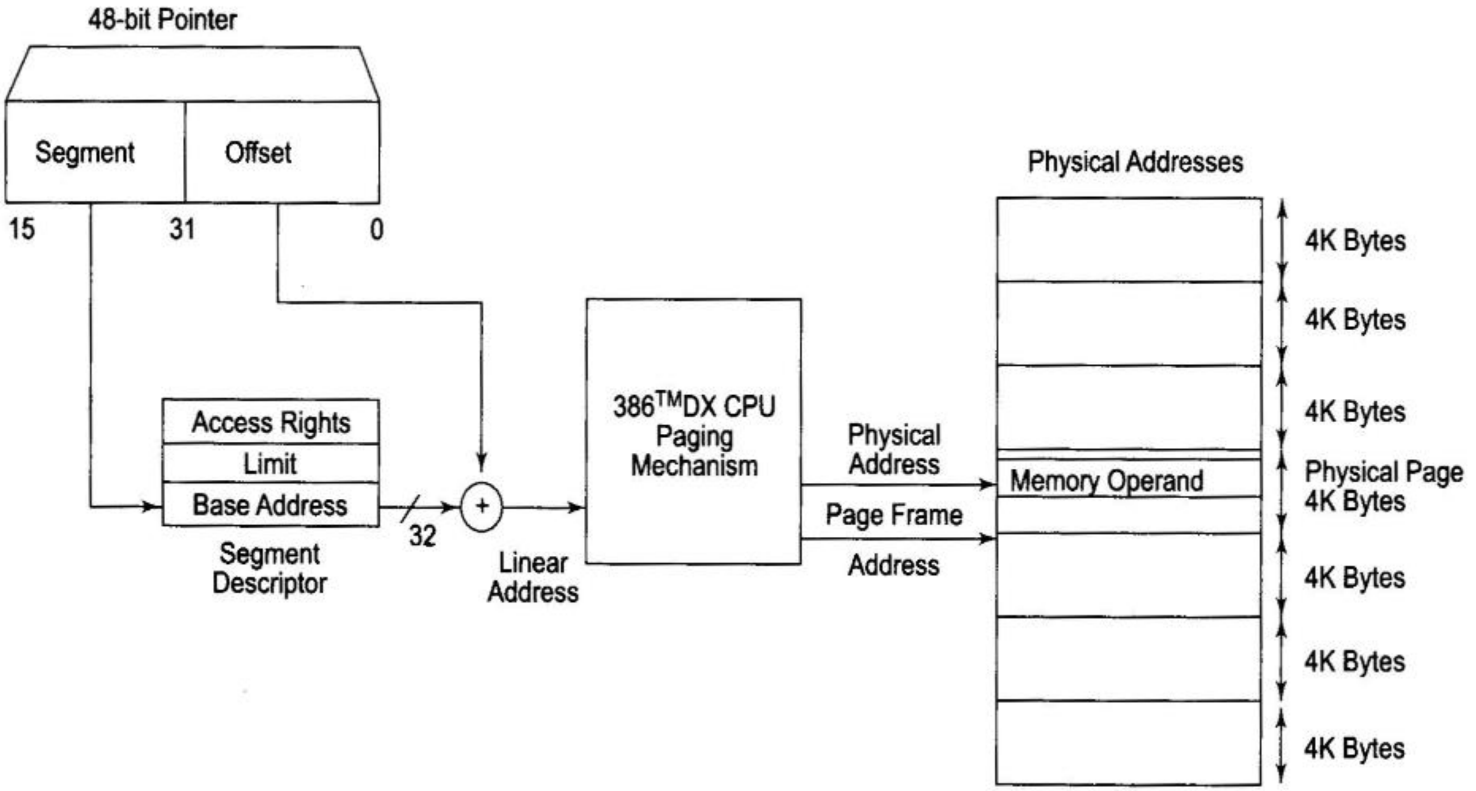
- Thus, the memory requirement of the task is substantially reduced, relinquishing the available memory for other tasks.
- Whenever the other pages of the task are required for execution, they may be fetched from the secondary storage.
- The previous pages which are executed, need not be available in the memory. Hence, the space occupied by them may be relinquished for other tasks.
- Thus, the paging mechanism provides an effective technique to manage the physical memory for multitasking.

Paging

- Paging unit
  - Paging unit converts linear address provided by segmentation unit into physical address.
  - Paging unit converts the complete map of a task into pages, each of size 4K.
  - The task is then handled in terms of pages, rather than segments.
  - Three components – page directory, page table and page itself

Protected mode

- Addressing in protected mode
  - If paging unit is enabled, then paging unit converts linear address into physical address.



Paging Unit Enabled in Protected Mode Addressing (Intel Corp.)

Paging

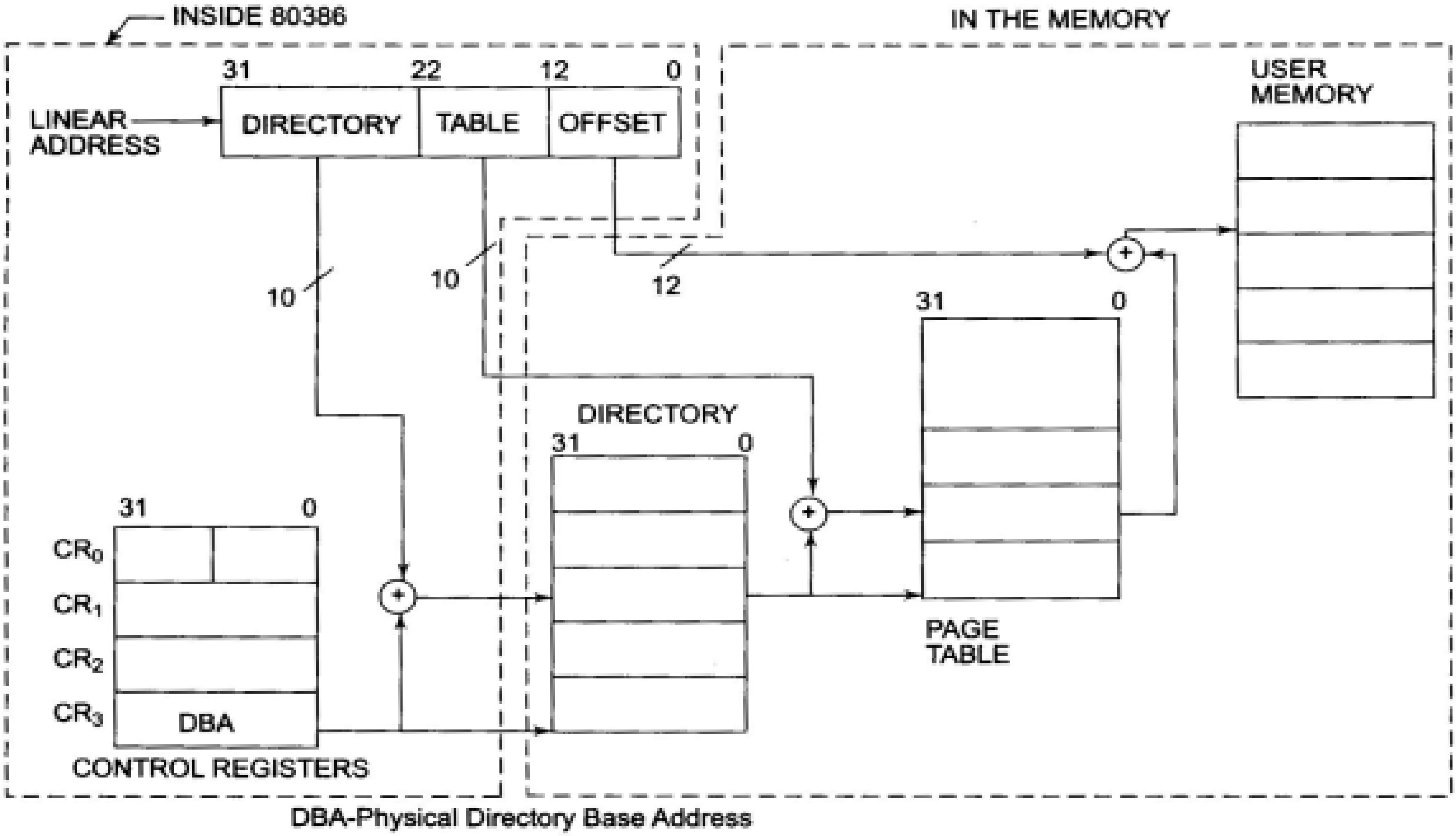
- **Page Directory Base Register**
  - The **CR3** is used as a page directory base address register, to store starting address of the page directory.
- **Page Directory**
  - This is at most **4Kbytes** in size.
  - Each directory entry is of **4 bytes**, thus total of **1024 entries** are allowed in a directory.
  - The **upper 10 bits** of the linear address are used as an index to the corresponding page directory entry.
  - The page directory entries point to the page table.

Paging

- **Page Table**
  - Each page table is of **4Kbytes** in size and contains maximum of **1024 entries**.
  - The address bits $A_{12} – A_{21}$ are used to select 1024 page table entries.
  - The page table entry contains starting address of the page. It is combined with **lower 12 bits** of the linear address.
  - The page tables can be shared between the tasks.

Paging

Block diagrammatic representation of complete paging mechanism of 80386



Paging Mechanism of 80386 (Intel Corp.)

physical memory. point to a separate page

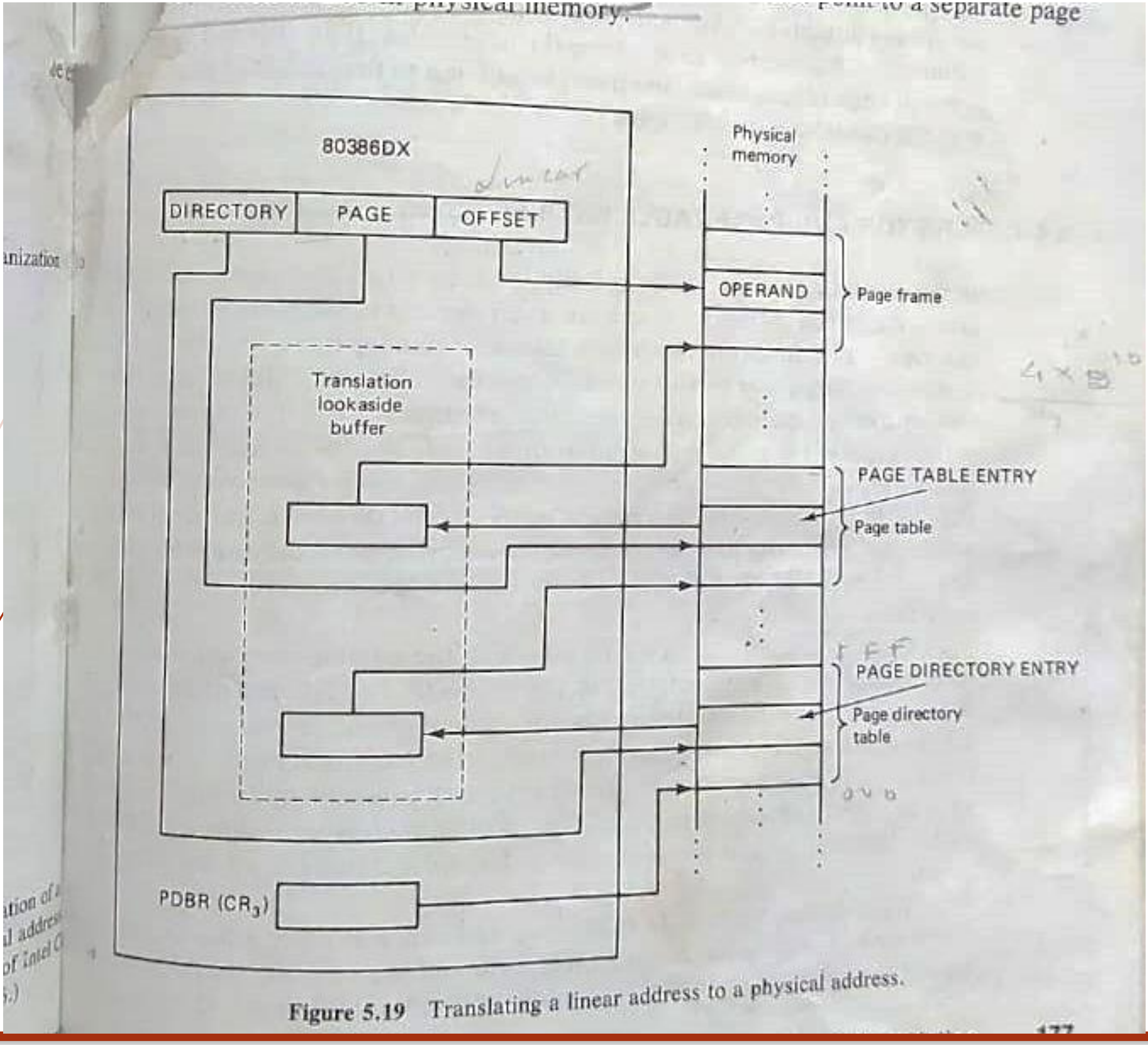
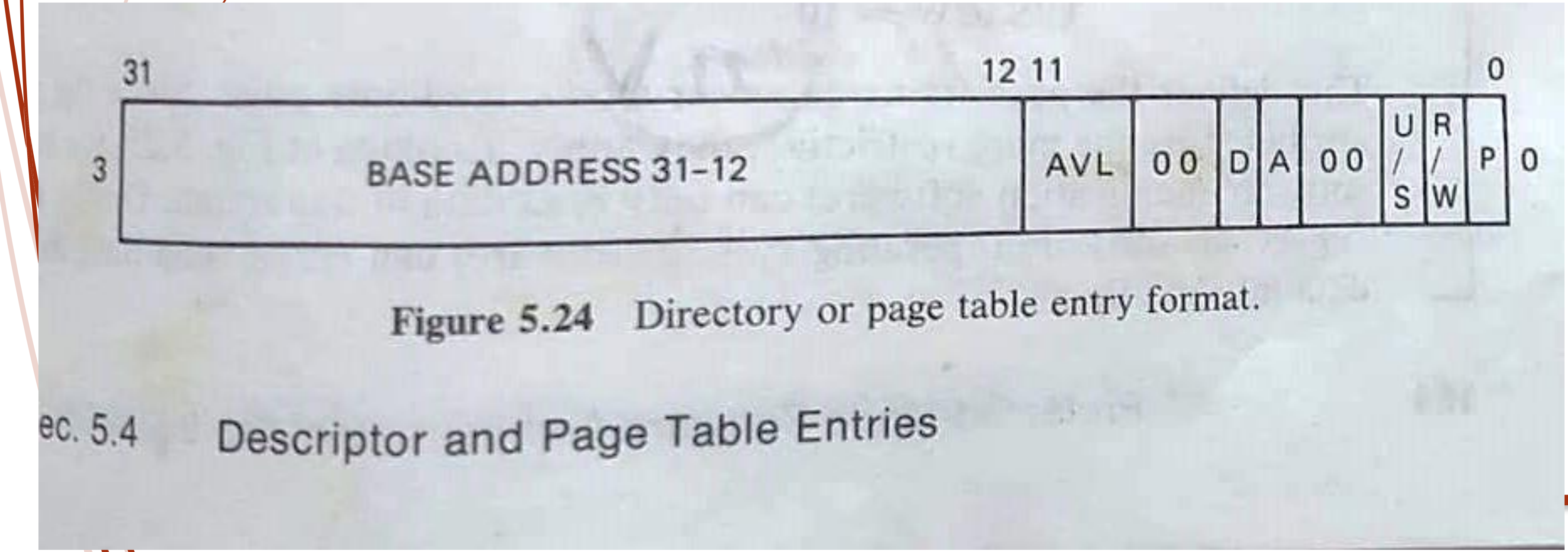


Figure 5.19 Translating a linear address to a physical address.

Page table/ directory table entry

- D – Dirty bit
- A – accessed
- U/S – user (1) /supervisor (0, level assigned to OS resources)
- R/W – read (1, read only)/write
- P – present
- Avl – available for the use by the programmer



	31 12	11							0	
3	BASE ADDRESS 31–12	AVL	0 0	D	A	0 0	U/S	R/W	P	0

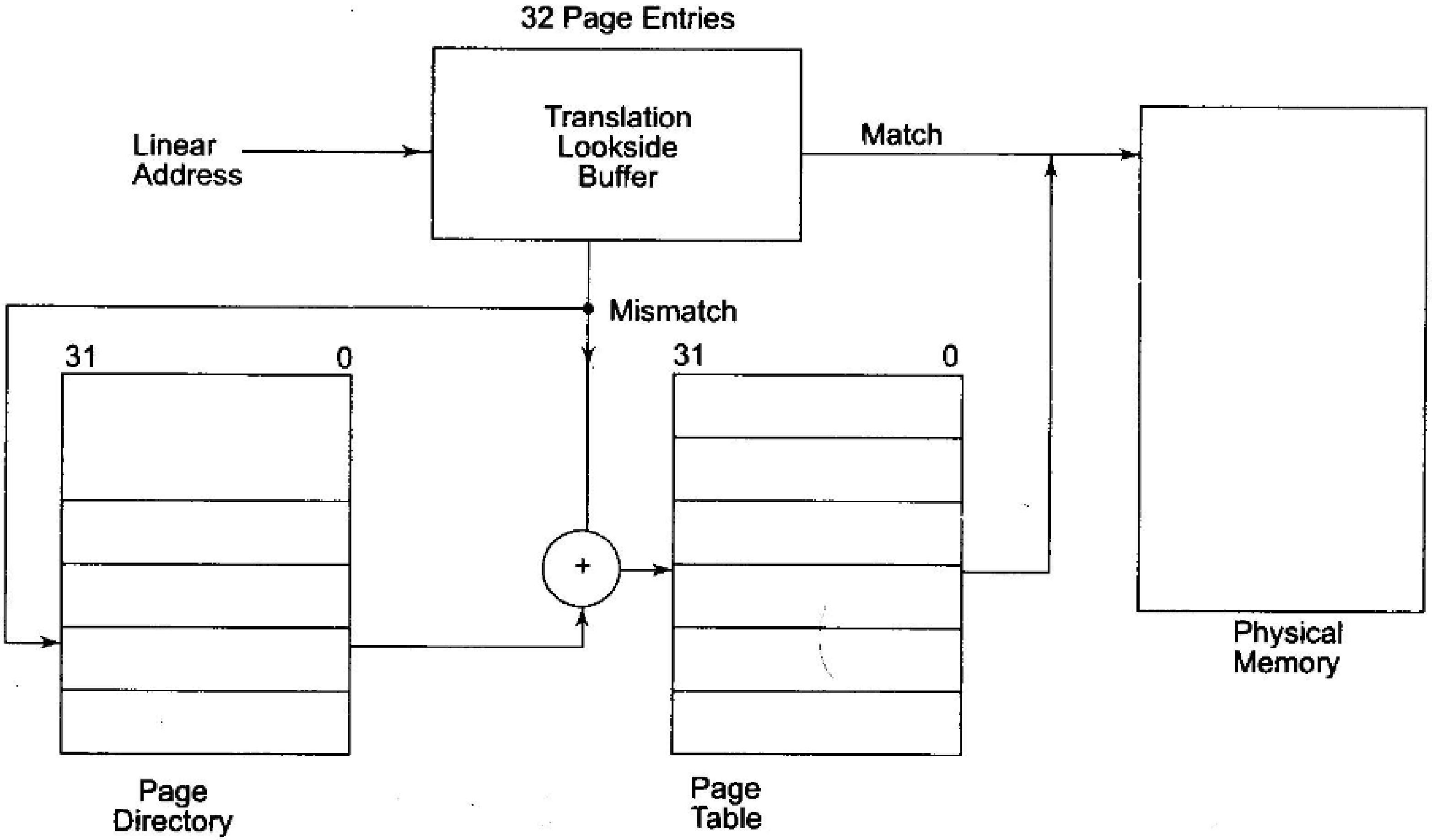
Figure 5.24 Directory or page table entry format.

ec. 5.4 Descriptor and Page Table Entries

Conversion of linear address to a physical address

- The paging unit receives 32-bit linear address from the segmentation unit.
- The upper 20 bits ($A_{12} - A_{31}$) are compared with all the 32 entries in the translation look aside buffer to check if it matches with any of the entries.
- If it matches, then the 32 bit physical address is taken from the matching TLB entry and placed on the address bus.
- TLB is a 32-entry (32 × 4bytes) page table cache which stores the 32 recently accessed page table entries.
- TLB is checked before converting linear address to physical address as a considerable time is wasted in the process.

Conversion of linear address to a physical address



Paging Operation with TLB (Intel Corp.)

Virtual mode

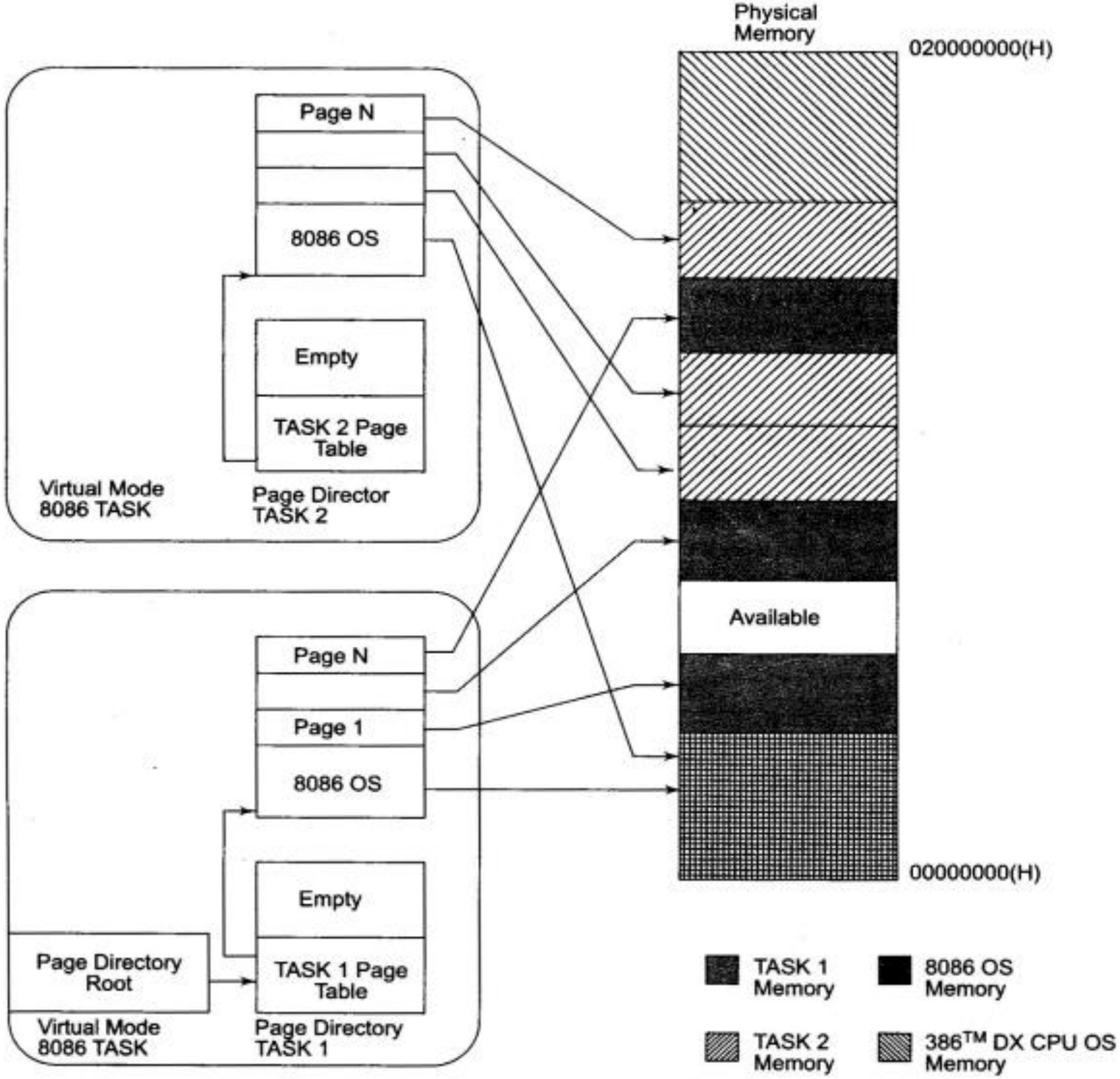
- 80386DX provides a virtual 8086 operating environment to execute 8086 programs.
- The real mode can also be used to execute the 8086 programs. But once the 80386 enters the protected mode from its real mode, it cannot return back to the real mode without a reset operation. Thus, the virtual 8086 mode of operation of 80386 offers an advantage of executing 8086 programs while in protected mode.
- The address forming mechanism in virtual mode is exactly identical with that of 8086 real mode.
- In virtual mode, 8086 can address 1Mbytes of physical memory that may be anywhere in the 4Gbytes address space of the protected mode of 80386.

Virtual mode

- In virtual mode, paging mechanism and protection capabilities are available to programmer.
- Paging unit may not be necessarily enabled in the virtual mode, but may be needed to run the 8086 programs which require more than 1 Mbytes of memory for memory management functions.

Virtual mode

- Memory management in multitasking virtual 8086 environment



Memory Management in Virtual 8086 Mode (Multitasking) (Intel Corp.)

Virtual mode

- In virtual mode, paging unit allows only 256 pages, each of 4Kbytes size.
- The virtual mode allows the multiprogramming of 8086 applications.