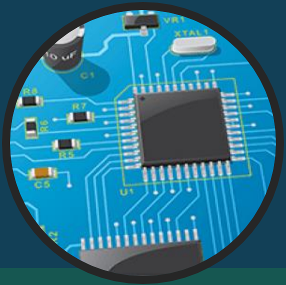
CS-235: Computer Organization & Assembly Language

CPU Memory Management



Topic # 03

Fall 2020

Muhammad Imran Abeel

<imran.abeel@seecs.edu.pk>

Outlines

- General Concepts of CPU Architectures
- Internal Registers
- Floating Point Unit
- IA-32 Modes of Operation
- IA-32 Memory Management

IA-32 Modes of Operations

Real Addressing Mode

Protected Mode

System Management Mode

Virtual 8086 Mode

[3]

IA-32 Modes of Operations

Real Address Mode

- Can access only **1MB** Memory (beyond 80236)
- For **MS-DOS**
- One task at a time (**Single-Tasking**)
- Programs can **access Shared Memory** Locations (Problem???)

IA-32 Modes of Operations

Real Address Mode

- Can access only **1MB** Memory (beyond 80236)
- For **MS-DOS**
- One task at a time (**Single-Tasking**)
- Programs can **access Shared Memory** Locations

Protected Mode

- Can access only **4GB** RAM
- For **Windows, Linux**
- Allows multi-tasking
- Memory Reservation for each Program

IA-32 Modes of Operations

System Management Mode

- Implementing Functions : **Power Management** & **System Security**
- Usually implemented by **Computer Manufacturers**

Virtual 8086 Mode

- Being in Protected Mode:
  the processor can directly execute a program in Real Mode

[6]

IA-32 Memory Management

Segmented Memory Model

- Real Mode
- Segmented Memory

Flat Memory Model

- Protected Mode

Segmented Memory Model

- Intel 8088/8086 is a so-called 16-bit Machine
- Each Register has 16 Bits
- 2^16 = 65536 = 64K
- But we want to use more memory (640K, 1M)…

[8]

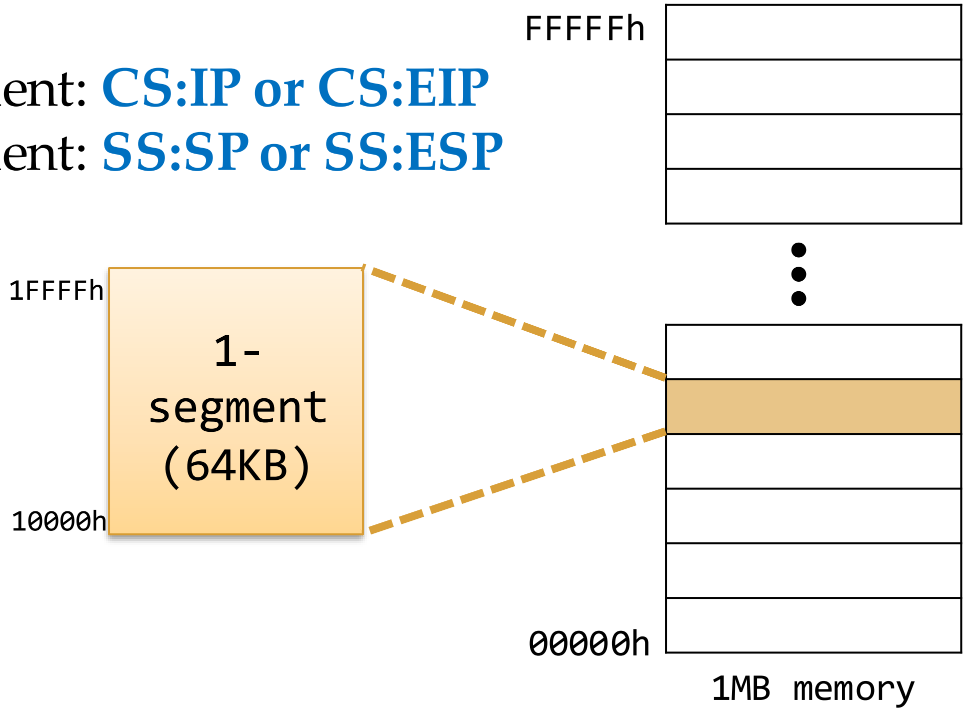
Segmented Memory Model

- **1MB** Total Memory (Real Mode)
  - Base Address
- Divided into **64KB** called Segments
  - Offset Address

- **Linear Address:** resultant of 16-bits Base Address and 16 bits Offset Address

Segmented memory model

For Code Segment: **CS:IP or CS:EIP**
For Stack Segment: **SS:SP or SS:ESP**



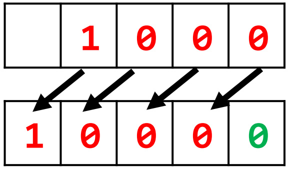
[10]

Segmented Memory Model

- For Code Segment: **CS:IP or CS:EIP**
- For Stack Segment: **SS:SP or SS: ESP**

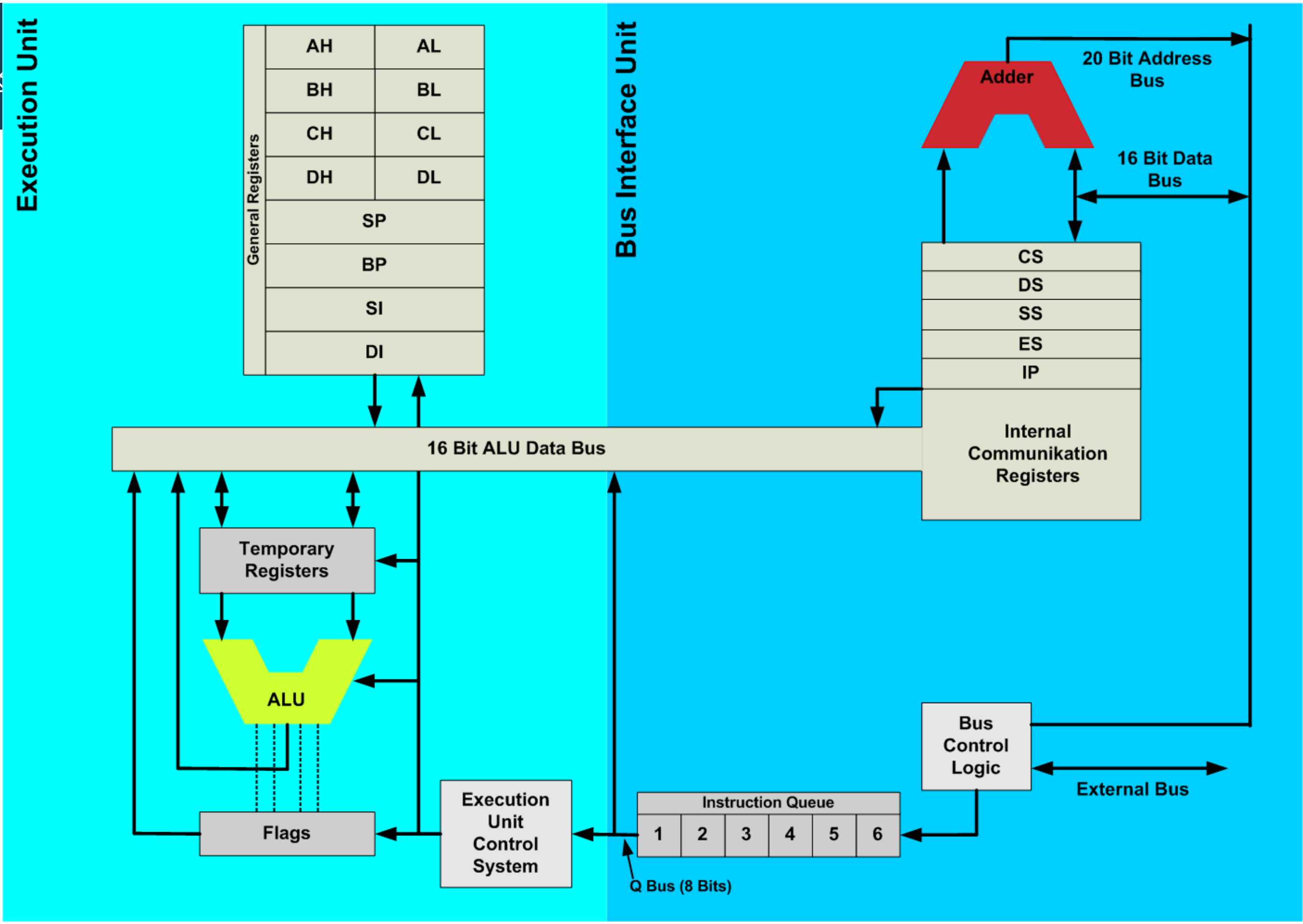
1000h:5000h calculate linear address?

Linear Address Calculations:



Left shift Segment Address by 4 locations	New Seg. Address 10000h
Add offset into new Address	Offset Address 5000h
Linear Address =	Linear Address 15000h

[11]


Execution Unit
General Registers
AH
AL
BH
BL
CH
CL
DH
DL
SP
BP
SI
DI
16 Bit ALU Data Bus
Temporary Registers
ALU
Flags
Execution Unit Control System
Bus Interface Unit
Adder
20 Bit Address Bus
16 Bit Data Bus
CS
DS
SS
ES
IP
Internal Communikation Registers
Bus Control Logic
External Bus
Instruction Queue
1
2
3
4
5
6
Q Bus (8 Bits)

Segmented Memory Model

Segment : Offset

- Segment: one of CS, DS, SS, ES
- Real address = Segment * 16 + Offset
- Overlapping Segments. For example:

  0000:01F0 = 0001:01E0 = 0010:00F0

Linear Address Calculation

- Activity

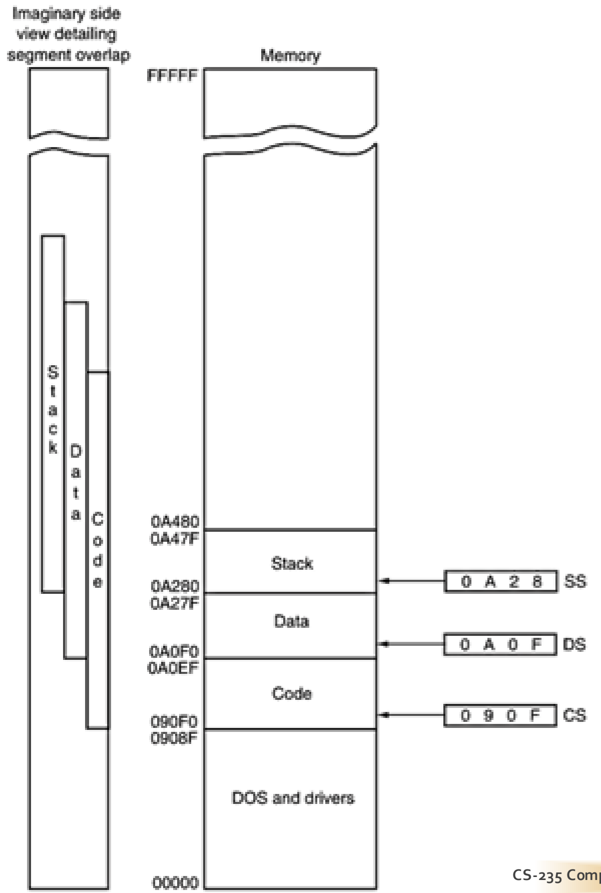
CS: 1200h **IP: F000h**

Calculate Linear Address?

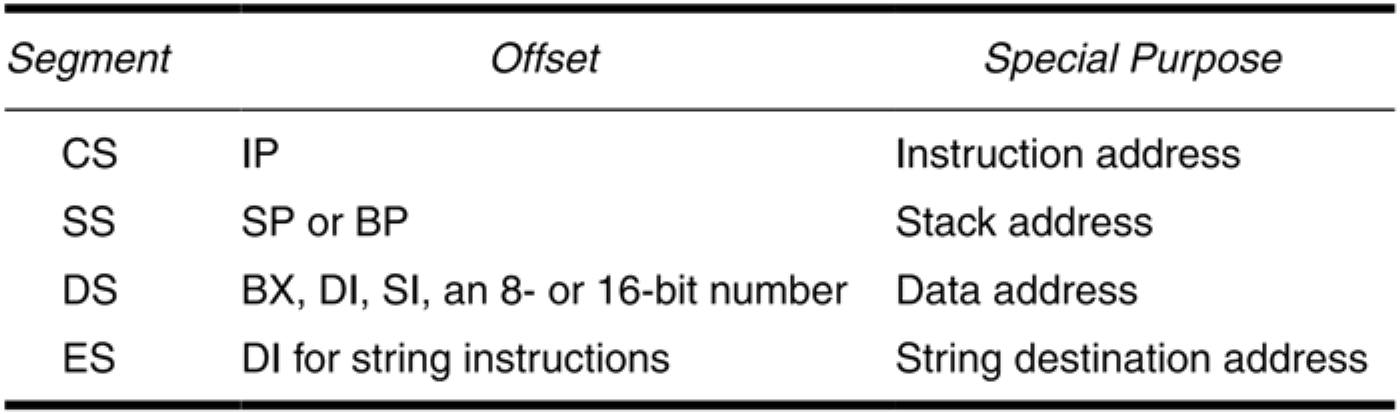
Linear Address : 21000h

Real Mode Operation

FIGURE 2–5 An application program containing a code, data, and stack segment loaded into a DOS system memory.



Segment & Offset Combination



Segment	*Offset*	*Special Purpose*
CS	IP	Instruction address
SS	SP or BP	Stack address
DS	BX, DI, SI, an 8- or 16-bit number	Data address
ES	DI for string instructions	String destination address

[16]

Protected Mode Memory Model

In place of the segment address, the **segment register contains a selector** that selects a descriptor from a descriptor table.

The **descriptor** describes the **memory segment's location, length, and access rights**.

Global descriptor table

- segment definitions that apply to all programs

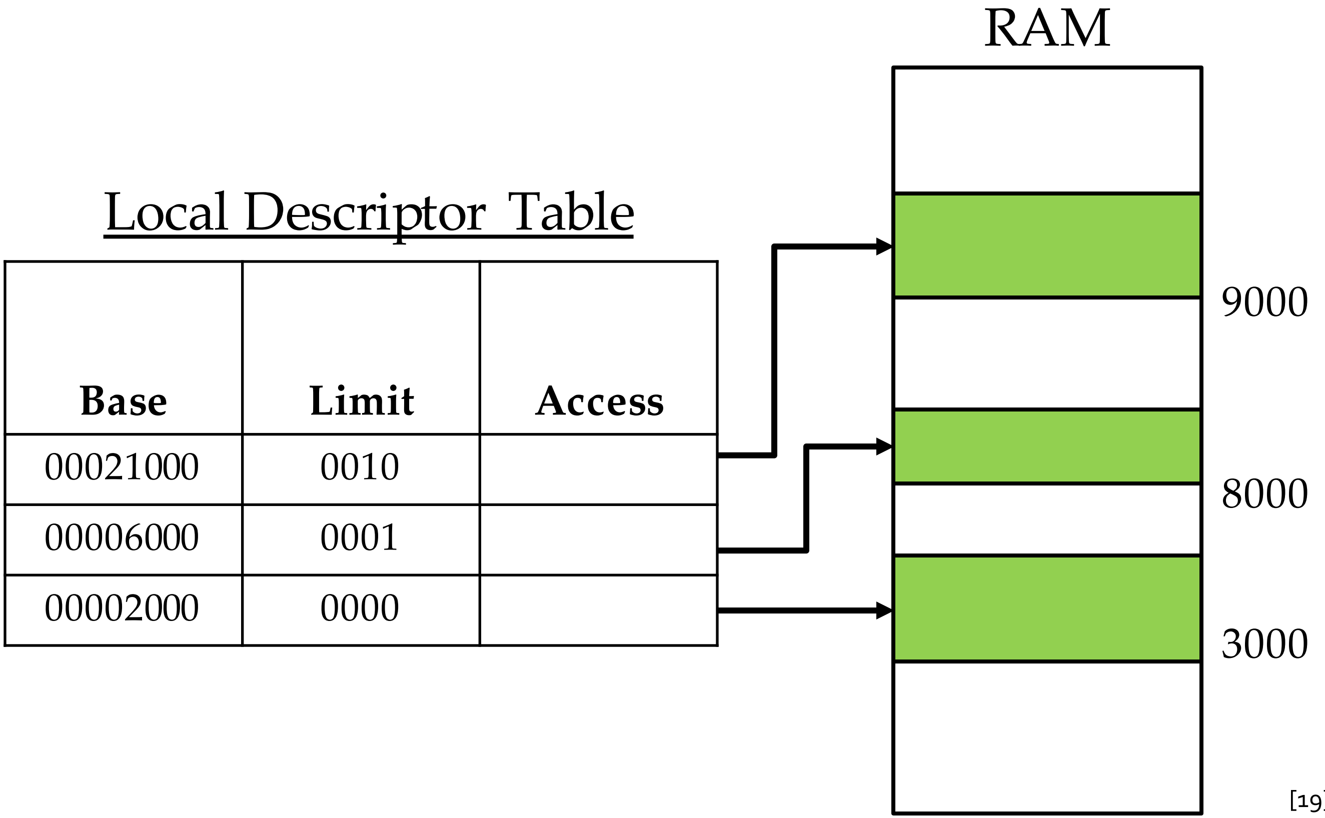
Local descriptor table

- segment definitions unique to a program.

Protected Mode Memory Model

- Segment descriptor tables
- Program structure
  - code, data, and stack areas
  - CS, DS, SS segment descriptors
  - global descriptor table (GDT)
- MASM Programs use the Microsoft flat memory model

Protected Mode Memory Model

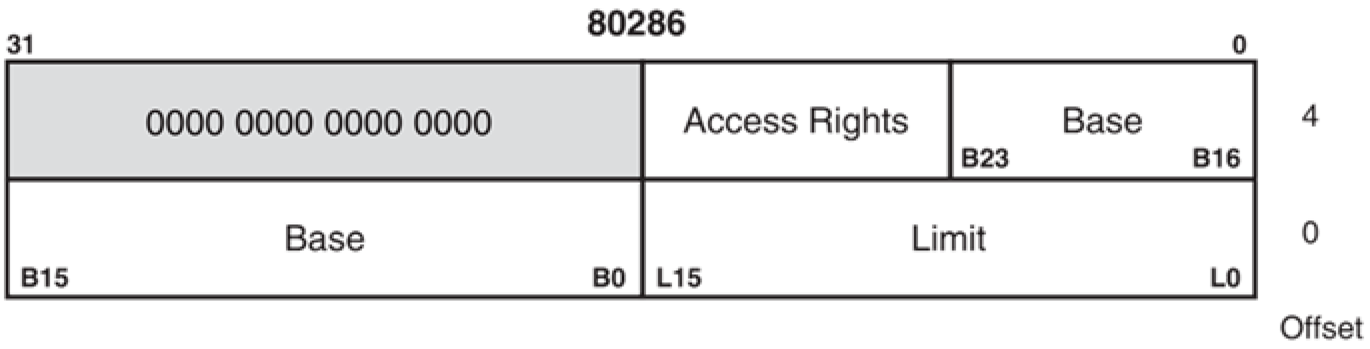


Base	Limit	Access
00021000	0010	
00006000	0001	
00002000	0000	

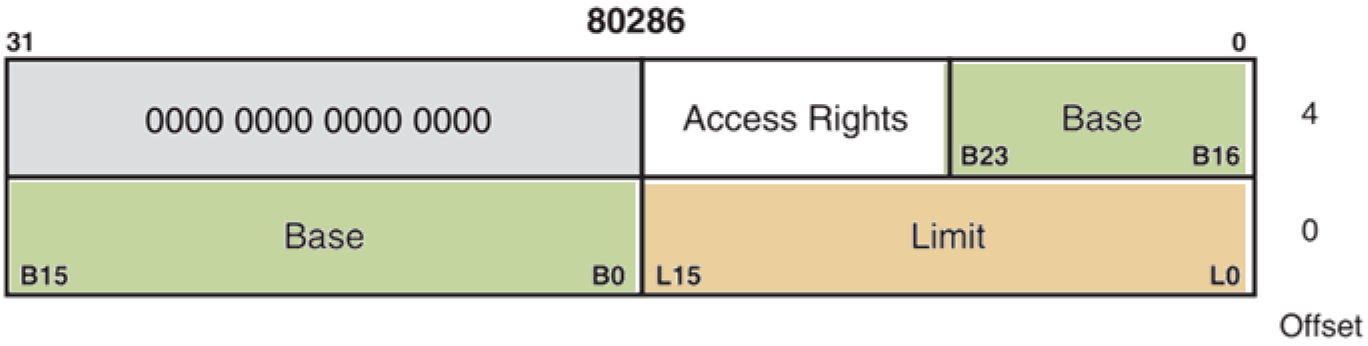
[19]

Descriptor Table

- Each descriptor table contains **8192 Descriptors**
- A total of **16,384 Descriptors** are available to an application at any time
- Max Memory 4G X 16383 = 4TB

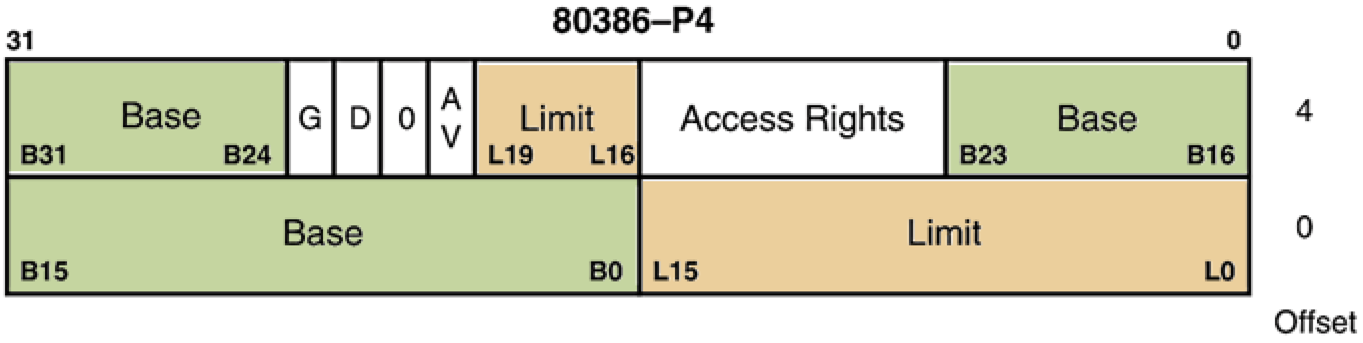


Descriptor Table: 80286



- **Base Address:** 24 bits
- **Limit Address (Offset):** 16 bits
- Each Descriptor is of **8-bytes**
- An 80286 can access Memory Segments that are **between 1 and 64K bytes** in length

Descriptor Format: 80386



- **Base Address:** 32 bits
- **Limit Address (offset):** 20 bits
- The 80386 and above access memory segments that are between **1 and 1M byte**, or **4K and 4G** bytes in length.

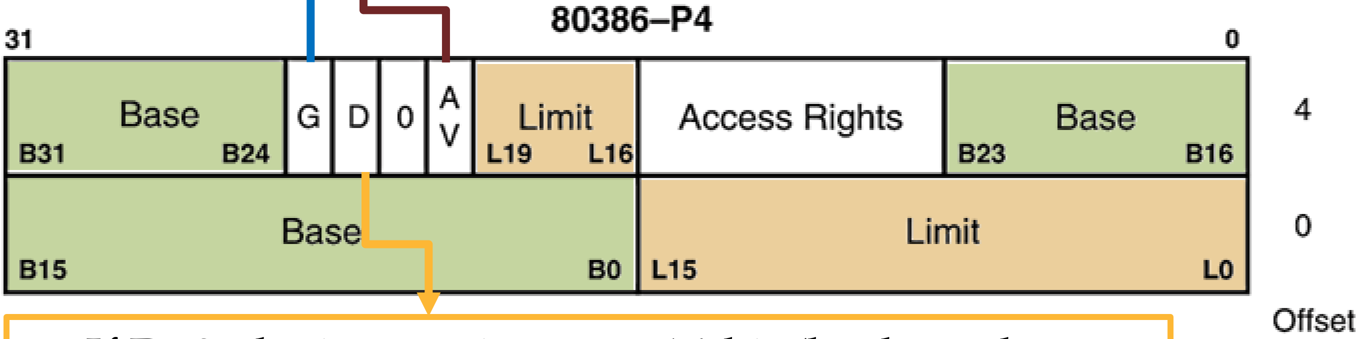
Descriptor Table

Granularity Bit:

- If G=0, segment limit of 00000H to FFFFFH.
- If G=1. multiplied by 4K bytes (appended with FFFH). The limit is then 00000FFFFH to FFFFFFFFH.

Available Bit:

- If AV=1, mean memory segment available.

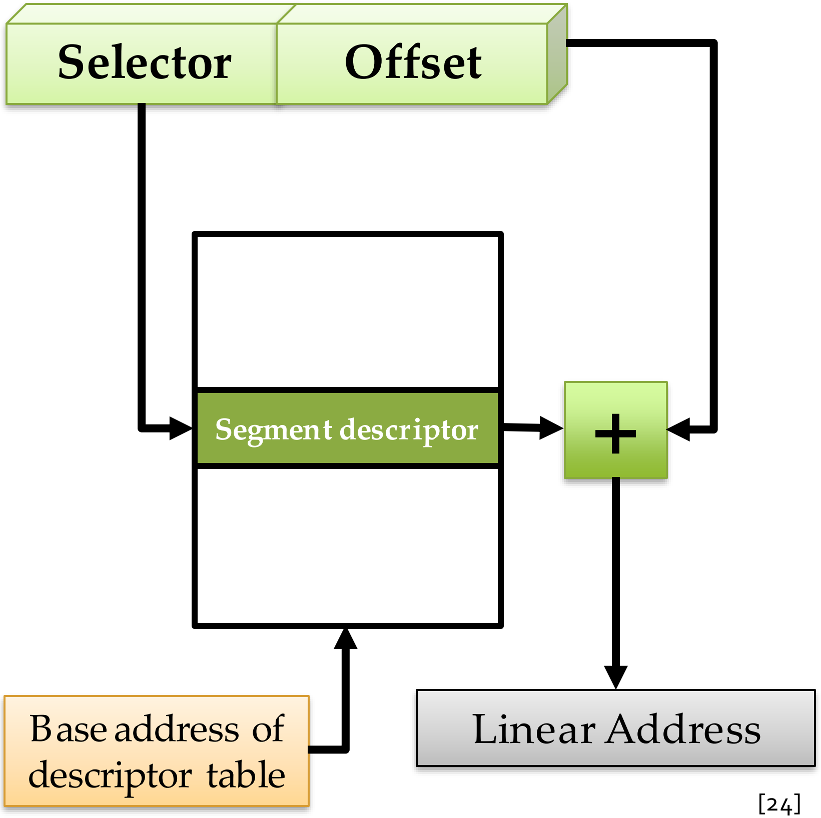


- If D=0, the instructions are 16-bit (backward compatible).

Protected Mode Memory Management

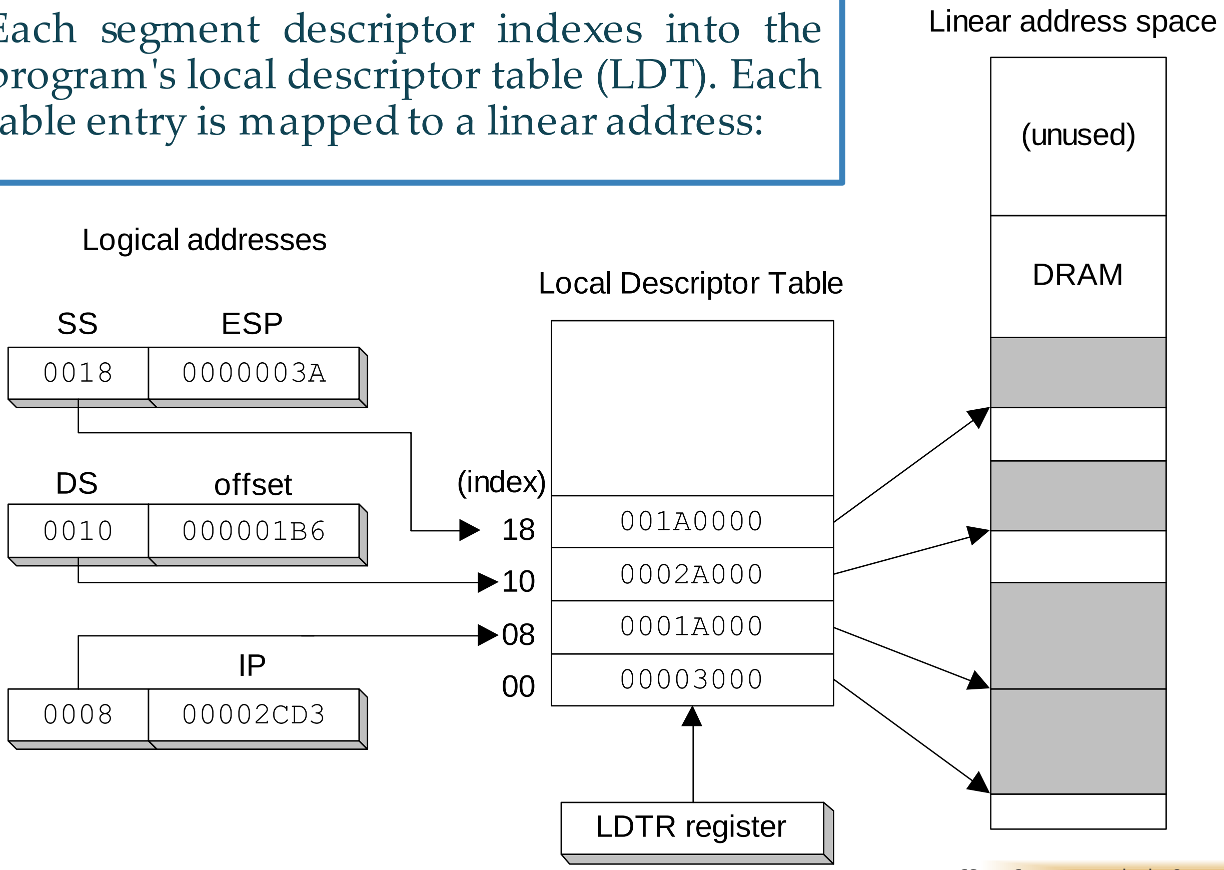
The segment selector points to a **segment descriptor**, which contains the **base address of a memory segment**.

The 32-bit offset from the logical address is added to the segment's base address, generating a 32-bit linear address



Example

Each segment descriptor indexes into the program's local descriptor table (LDT). Each table entry is mapped to a linear address:



Paging

- Virtual Memory uses disk as part of the memory, thus allowing **sum of all Programs can be larger than Physical Memory**

- Only part of a **Program must be kept in Memory**, while the remaining parts are kept on disk

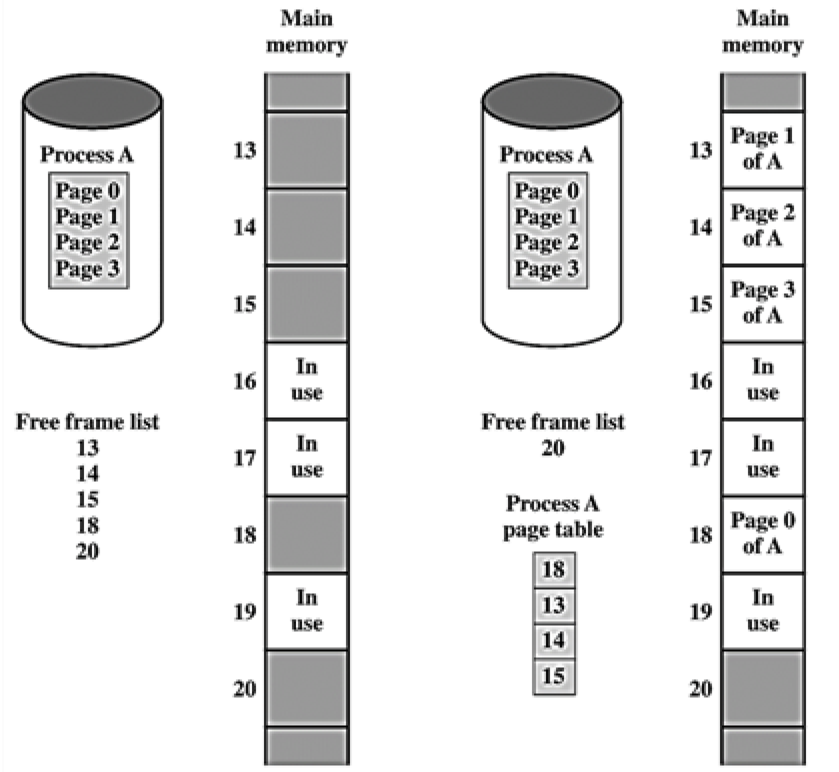
- The Memory used by the Program is divided into small units called Pages (4096-byte)

- As the Program runs, the Processor selectively unloads inactive pages from memory and loads other pages that are immediately required

Paging

- OS maintains Page Directory and Page Tables
- **Page Translation:** CPU converts the Linear Address into a Physical Address
- **Page Fault:** occurs when a needed page is not in memory, and the CPU interrupts the program
- **Virtual Memory Manager (VMM) –** OS utility that manages the loading and unloading of pages
- OS copies the page into memory, program resumes execution

Paging



(a) Before (b) After

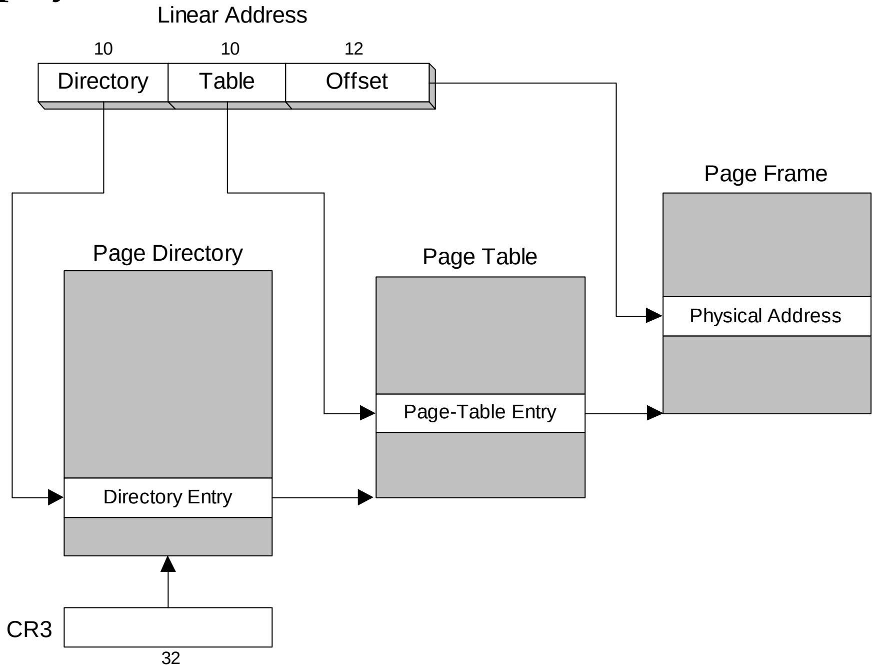
Figure 8.15 Allocation of Free Frames

Chunk of Program:
Page

Chunk of Memory:
Frame

Page Translation

A linear address is divided into a **page directory field, page table field, and page frame offset.** The CPU uses all three to calculate the physical address.



Page Translation

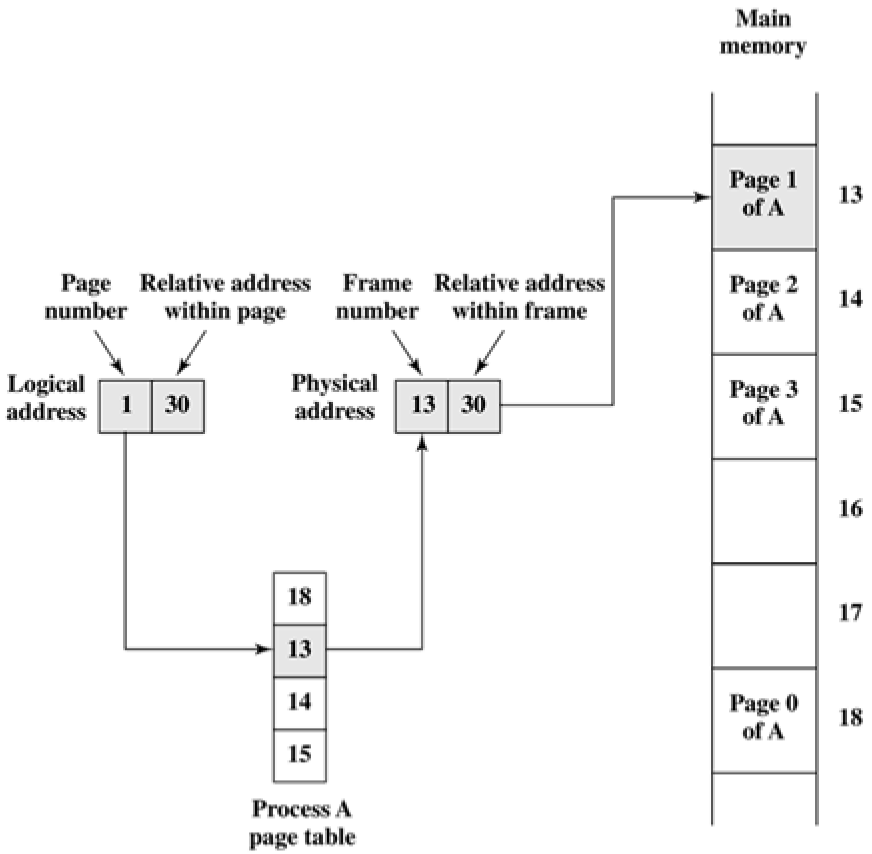


Figure 8.16 Logical and Physical Addresses

CISC vs RISC

- **CISC – Complex Instruction Set**

  large instruction set

  high-level operations

  requires microcode interpreter

  examples: Intel 80x86 family

  - Mult 2,3

CISC vs RISC

- **RISC – Reduced Instruction Set**
  - Simple instruction formats
  - Small instruction set
  - Directly executed by hardware

Examples:

ARM (Advanced RISC Machines)

Questions?

THANK YOU!