INTEL 80286

Unit 7

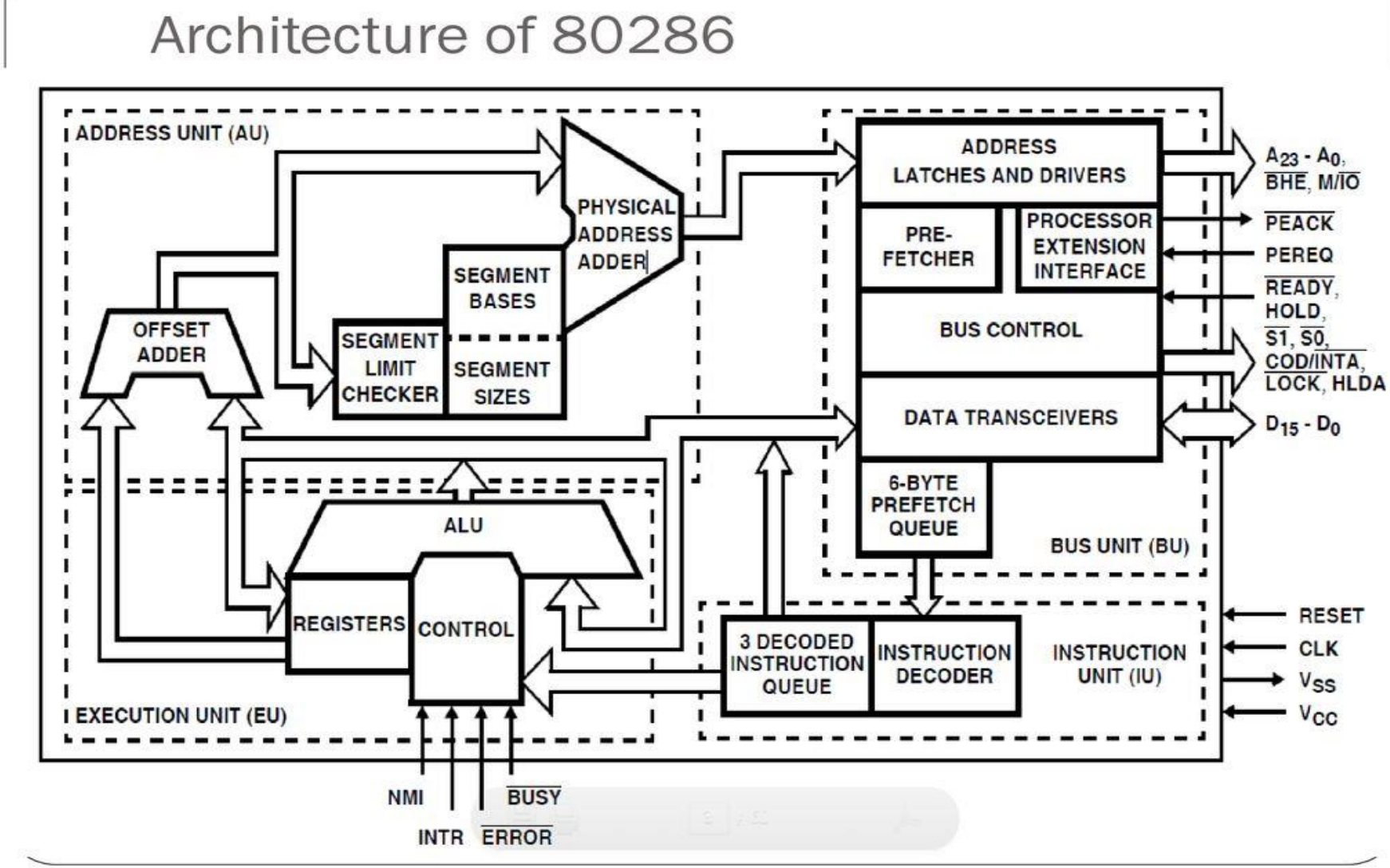
INTRODUCTION

- The Intel 80286 also called iAPX 286
- Introduced on 1 February 1982, was a 16-bit x86 microprocessor with 134,000 transistors .
- The 80286 is the first member of the family of advanced microprocessors with memory management and wide protection abilities.
- The 80286 was designed for multi-user systems with multitasking applications, including communications (such as automated PBXs*) and real-time process control.

PBX- private **automatic branch exchange*

- 80286 offers two mode of operation:
  1. ***Real mode***
  2. ***Protection mode***
- A no: of changes have been made to both s/w and h/w architecture of the 80286,primarily to improve its performance.
  - ✓Additional pipeline for higher performance
  - ✓Instruction set is modified
  - ✓Address and Data bus are de-multiplexed to simplify system design
- It is manufactured using the high-performance metal oxide semiconductor.
- It consist of four independent units: **address unit, bus unit, instruction unit and execution unit.**

ARCHITECTURE OF INTEL 80286



INTERNAL ARCHITECTURE

- The 80286 is internally partitioned into four units:
  - ✓ **Bus Unit**
  - ✓ **Instruction Unit**
  - ✓ **Execution Unit**
  - ✓ **Address Unit**
- **Bus Unit**
  - The bus unit interface with the outside world .
  - It provide a 16 bit data bus, 24 bit address bus and the signals needed to control bus transfers buses are de-multiplexed.
  - It is responsible for performing all external bus operation.
  - Processing unit contain the latches and drives for address bus and transceivers for the data buses.
  - Bus unit also contains element called the pre-fetcher and pre-fetch queue..together these elements implement a mechanism know as an **instruction stream queue.**

Instruction Unit

- The IU access the output end of the pre-fetch queue. It read one instruction byte after the other from the output of the queue and decode them into the 69-bit instruction format used by the 80286 EU.
- IU off-load the responsibility for instruction decoding from the execution unit.
- Instruction queue within the instruction unit permit three fully decoded instruction to be waiting for execution by the execution unit.

ADDRESS UNIT

- Address unit provides the memory management and protection services for the 80286.
- It off-load the responsibility for performing high speed address calculation, virtual to physical transition and limit and access right attributes checks.
- Address unit calculate the address of the next instruction to be fetched.
- It also performs the various address transaction and protection checks needed when performing protected mode bus cycle.

EXECUTION UNIT

- EU includes the ALU and a control ROM.
- General purpose registers –accessible registers to users during programming.
- Control Rom contain the microcode sequences that define the operation performed by the 80286's instruction.
- EU read decoded instructions from the instruction queue and perform the operations that they specify.

REGISTER ORGANIZATION OF 80286

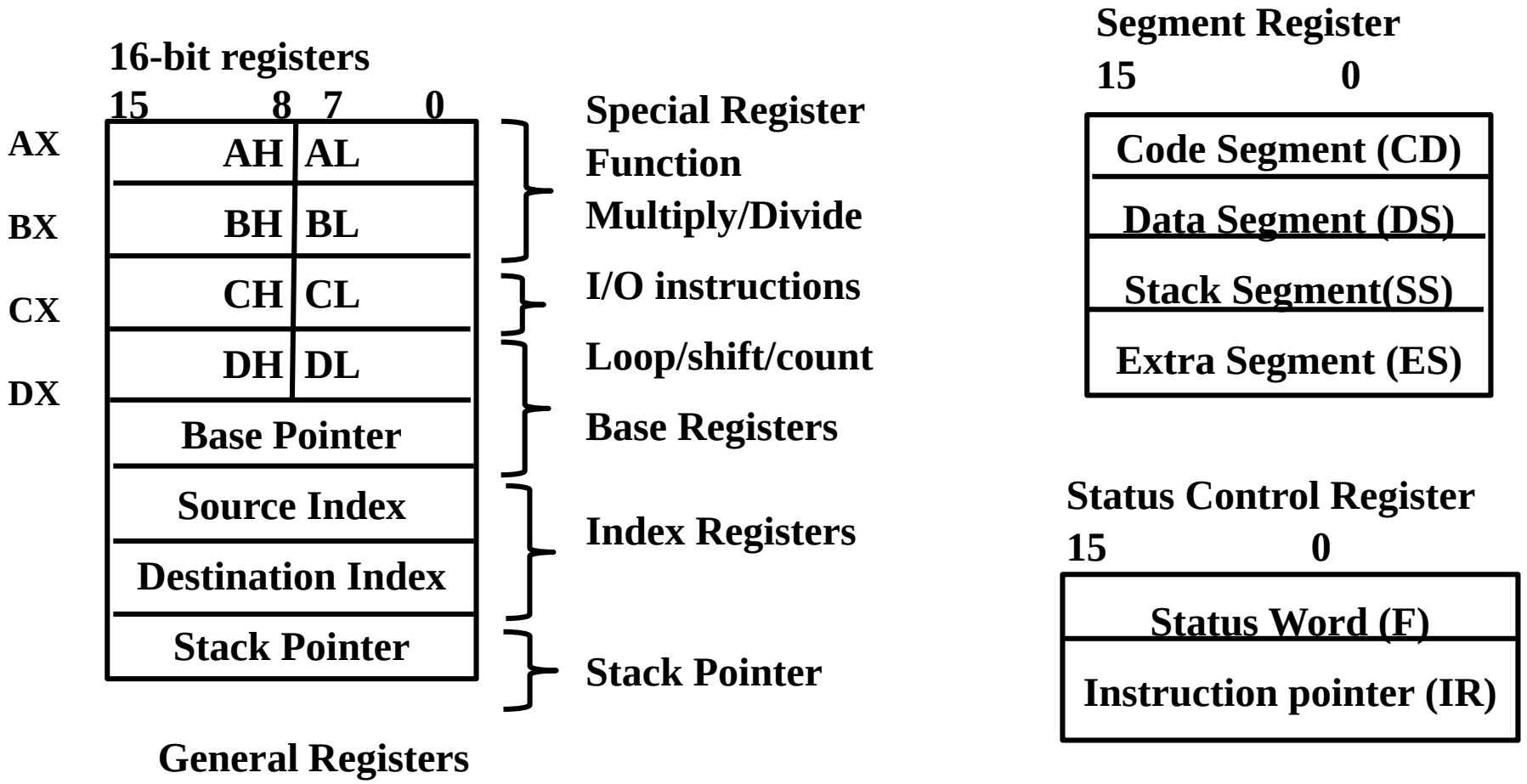


Fig: The programming model of 80286

MEMORY ADDRESSING IN 80286

- 80286 have two modes of operations
  - ***Real mode***
  - ***Protected mode or software mode***

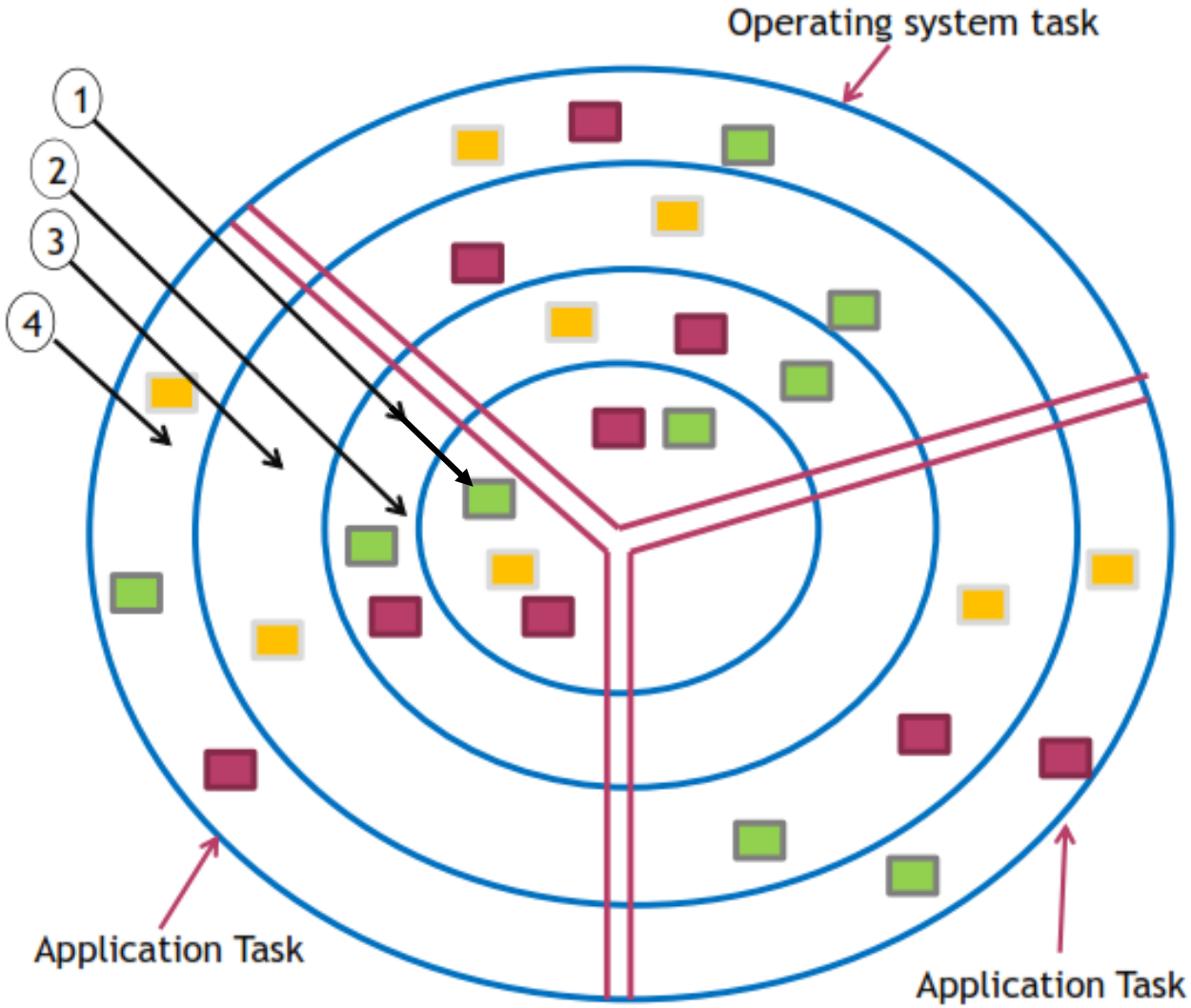
1. Real mode:

- Real mode is a normal mode or when 80286 is power on it comes up in the real mode or when 80286 is hardware reset, it automatically enters real address mode.
- When in the real mode , the 80286 can be used to execute the base instruction set of the 8086.

- The code for the base instruction of the 80286 is identical to that of the 8086.
- Number of new instructions have been added in the instruction set of the 80286 to enhance performance and function .
- A new register called the machine status word register (MSW) is used to switch the 80286 from real mode to protected mode.
- Address is 20 bit wide with16 bit segment and 16 bit offset.

2. PROTECTED MODE

- 1-GB of virtual memory and 16 Mbyte of physical memory. The address is 24 bit.
- To enter Protected virtual address mode (PVAM) mode, Processor Status Word (PSW) is loaded by the instruction LPSW.
- In PVAM the 80286 has its multitasking capability.
- The 80286 PVAM extends use of segments in such a way that the segment of several tasks can reside in memory at the same time and can be dynamically loaded as the need arises.
- In order to provide protection between segments within a task, segments are assigned to one of four privileged levels. Thus the various tasks and segment are protected form unwanted access by type segment ,level and task protection mechanism.



code segment

Stack segment

Data segment

1 Level 0-Most privileged

2 Level 1

3 Level 2

4 Level 3-Least Privileged

- **Level 0** would normally include only the very critical modules in the operating system.
- **Level 1** modules would consist chiefly of the less critical operating system services.
- **Level 3** would be for application program and **Level 2** most often would be used for custom extension to these programs.

DESCRIPTOR CACHE

- To allow for fast accesses to segmented memory, the x86 processor keeps a copy of each segment descriptor in a special descriptor cache.
- Two types: ***Global and Local Descriptor Table***
- Saves the processor from accessing the ***GDT*** for every memory access made.
- The cache for each segment selector (cs, ss, ds, es, fs, gs) contains all of the bits and fields you would also find in the ***GDT***.
- Include descriptor type, access rights, base and limit.
- In ***protected mode*** these fields are filled from the GDT or LDT whenever there is written to a segment selector (both implicitly and explicitly).
- In ***real mode***, the processor generates entries internally as there is no GDT involved in this mode.
- Interestingly, not all fields are updated in real mode. Most notably that includes the segment limit

MEMORY ACCESS IN GDT AND LDT

- **Selectors and descriptors:**
  - **2x8192 descriptors**
  - **Selector selects one of the descriptor from descriptor table**
  - **Selector is inside segment register**

0000 0000 0000	Access Rights	Base
Base	Limit	

MULTITASKING

- In 80286 we improve the multitasking capability by adding necessary memory management and task switching in such way that there is adequate protection
- Between the application task and the system and other more privileged task
  - *For separating the from each other*
  - *Between the code and data module*
  - *Again unwanted accessing or changing the data or code.*
- The primary key to multi-tasking in the ability to break the various entities to be processed into part in such a way that the part can easily function together and adequate protection is maintained .

- Modularity is important in that it allow the programmer to view the entire system as collection of modules that make up the task and subtask.so that operating system module that include critical code and data table can be protected from application program.
- Second key to multi tasking is the ability to change quickly and smoothly from one task to another.
- The task containing the module that is currently executing is called current task.
- As the time progresses ,it is necessary to change the current task ie:- to set aside logically the segments that comprise the current task and make the segment in another task the current task . This is called ***task switching***.

Selectors and Descriptors

- Allows access to data and programs located within & above the first 1M byte of memory.
- Protected mode is where Windows operates.
- In place of a segment address, the segment register contains a selector that ***selects a descriptor from a descriptor table***.

SELECTORS AND DESCRIPTORS

- The descriptor is located in the segment register & describes the location, length, and access rights of the segment of memory.
  - ✓ it selects one of ***8192 descript***ors from ***one of two tables*** of descriptors
- In protected mode, this segment number can address any memory location in the system for the code segment.
- Indirectly, the register still selects a memory segment, but not directly as in real mode.
- **Global descriptors contain segment**
  - ✓ definitions that apply to all programs.
- **Local descriptors are usually unique to an** application.
  - ✓ a global descriptor might be called a **system descriptor, and local descriptor an application descriptor**

- Figure 1 shows the format of a descriptor for the 80286 through the Core2.
  - each descriptor is 8 bytes in length
  - global and local descriptor tables are a maximum of 64K bytes in length

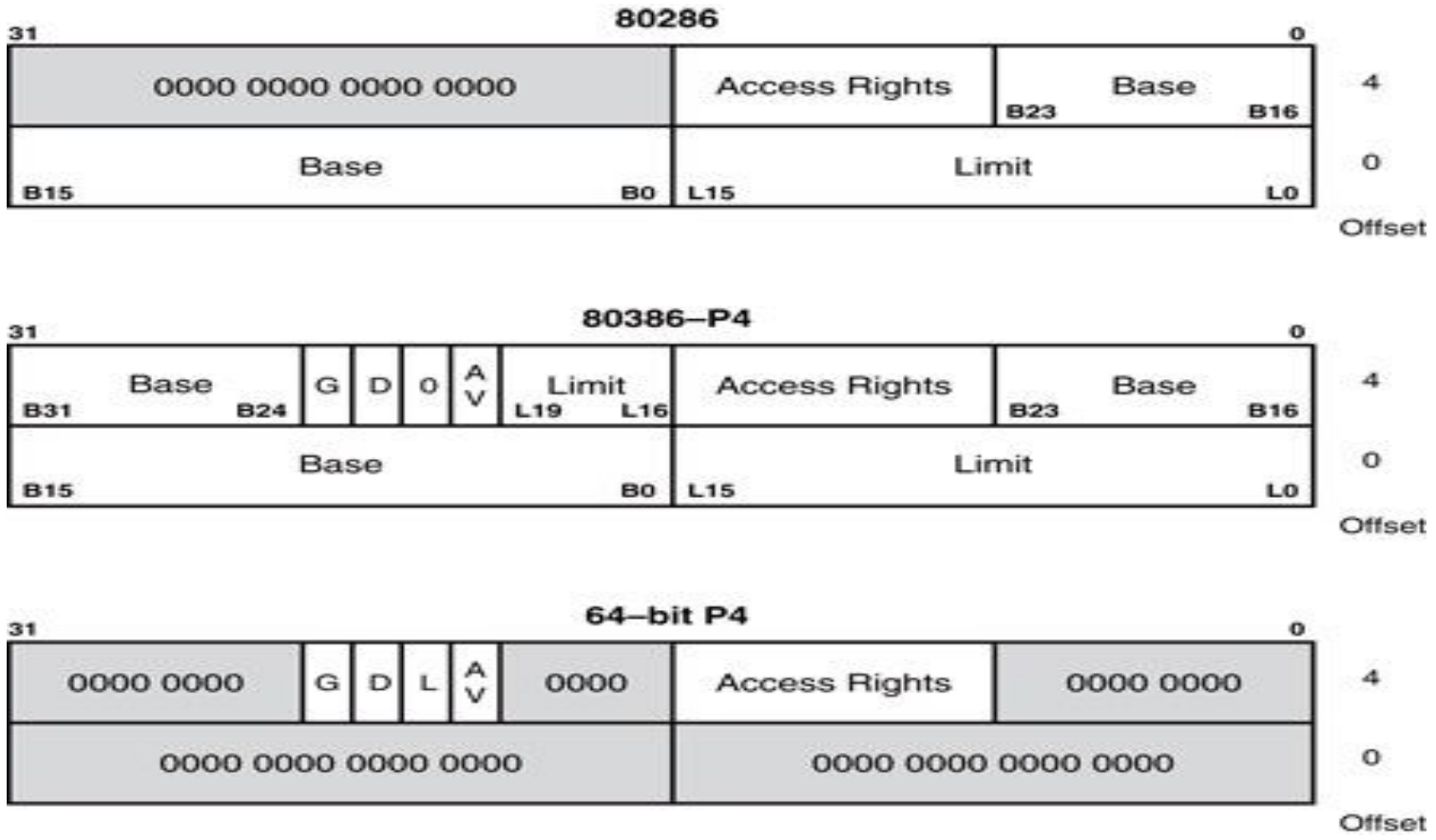


Figure 1 The 80286 through Core2 64-bit descriptors.

- The **base address of the descriptor indicates** the starting location of the memory segment.
  - ✓the paragraph boundary limitation is removed in protected mode
  - ✓segments may begin at any address
- •The G, or **granularity bit allows a segment** length of 4K to 4G bytes in steps of 4K bytes.
- Descriptors are chosen from the descriptor table by the segment register.
- The **TI bit selects either the global or the local** descriptor table.
- **Requested Privilege Level (RPL) requests** the access privilege level of a memory segment.
  - ✓ If the requested privilege level matches or is higher in priority than the privilege level set by the access rights byte, access is granted.

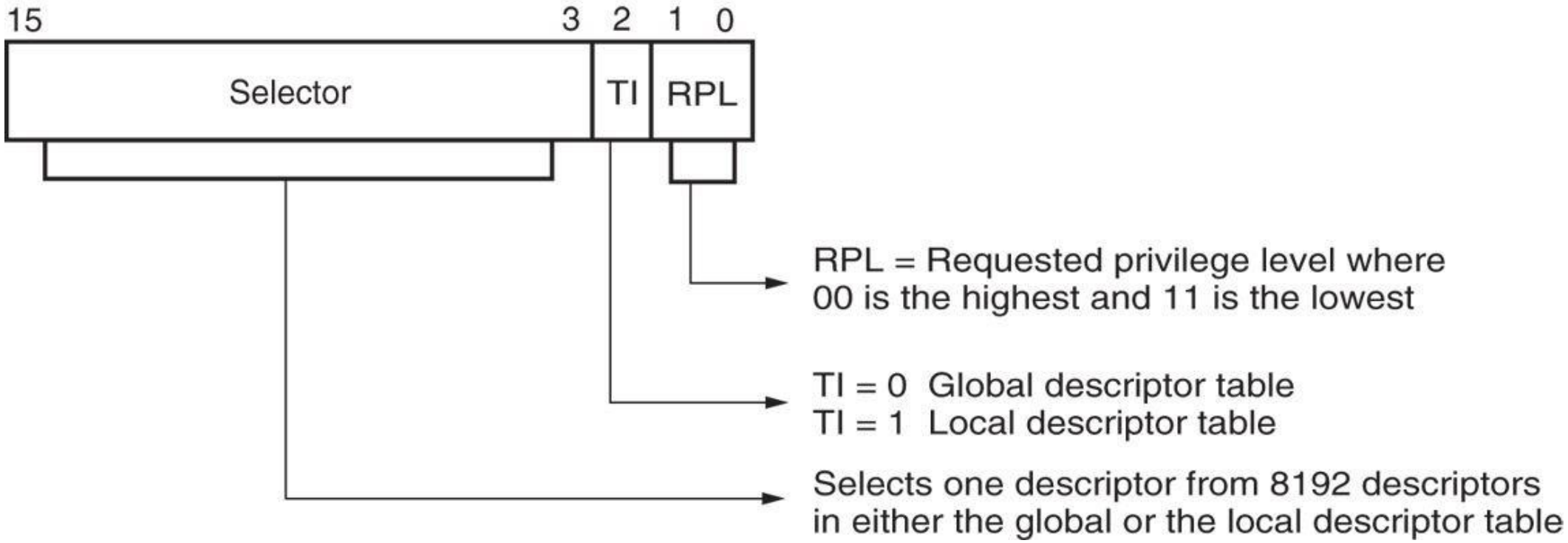


Figure 2: The contents of a segment register during protected mode operation of the 80286 through Core2 microprocessors.

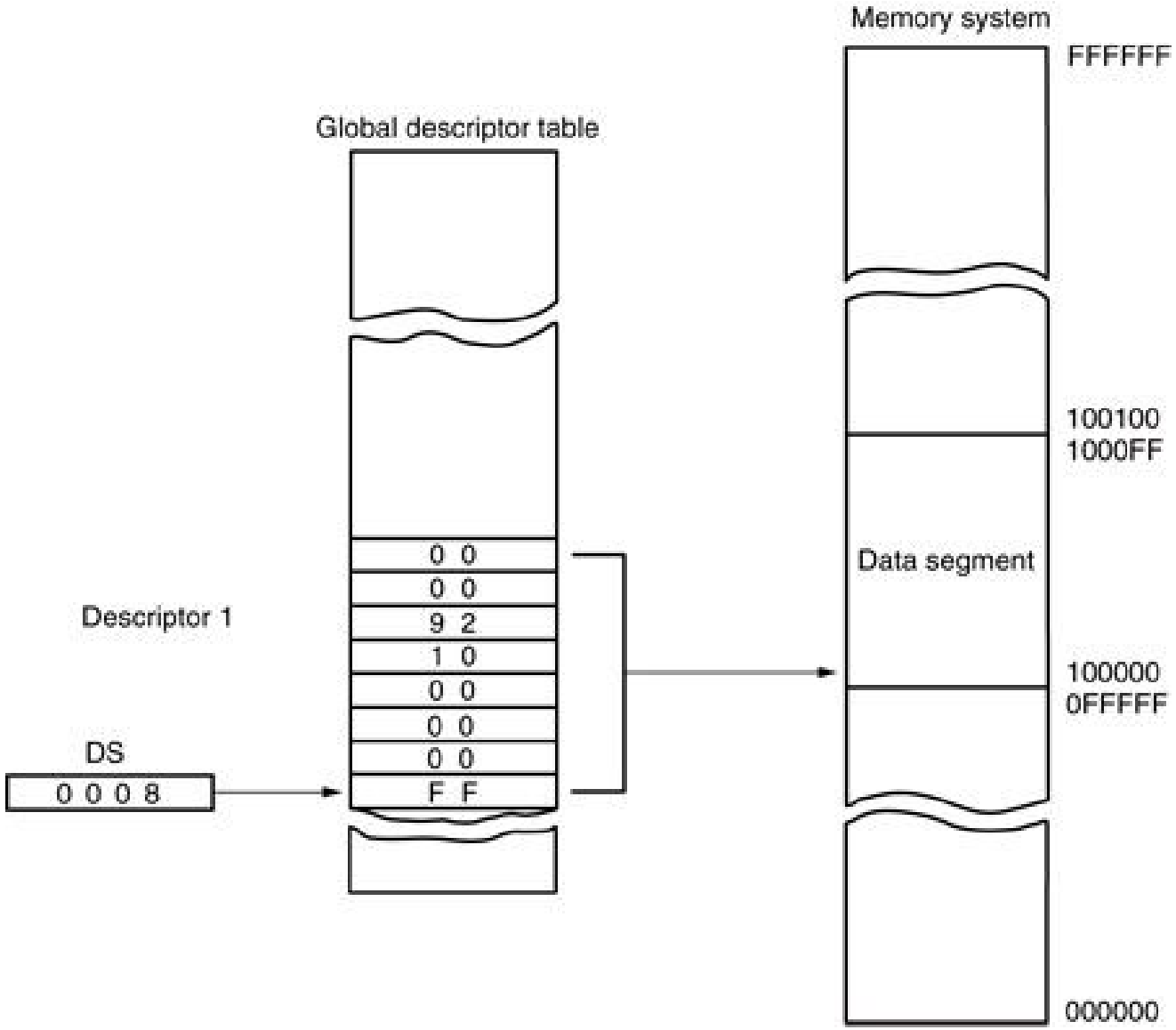
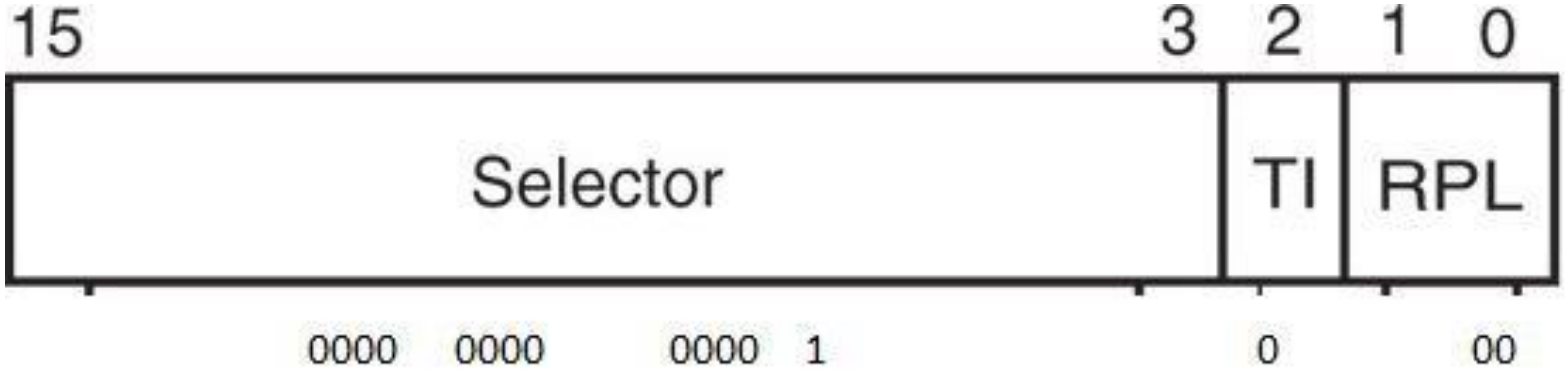
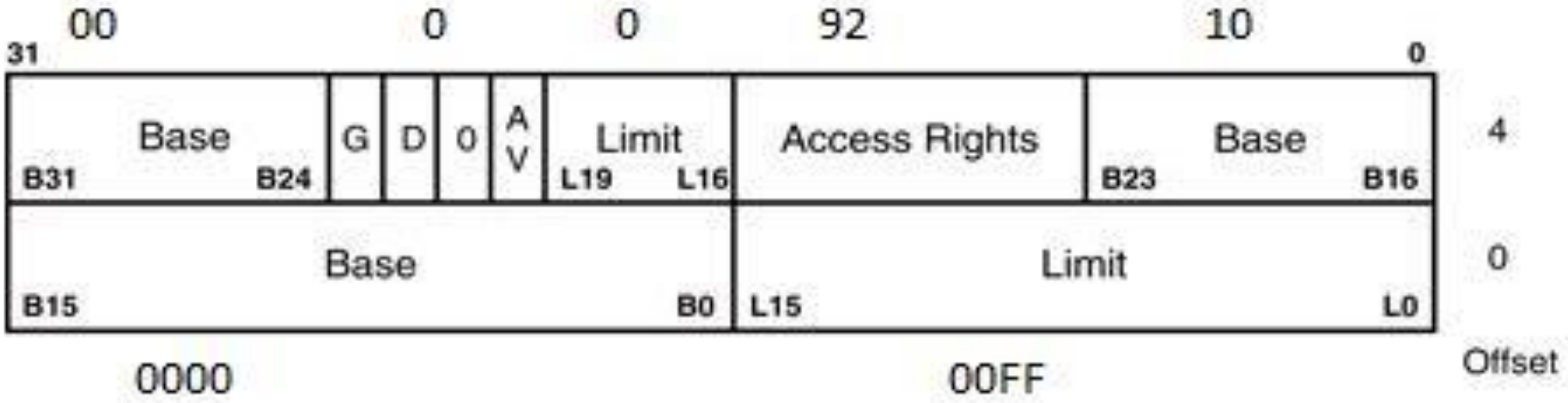


Figure 3: Using the DS register to select a description from the global descriptor table. In this example, the DS register accesses memory locations


15
3
2
1
0
Selector
TI
RPL
0000
0000
0000
1
0
00


00
0
0
92
10
31
0
Base
B31
B24
G
D
0
A
V
Limit
L19
L16
Access Rights
Base
B23
B16
4
Base
B15
B0
Limit
L15
L0
0
Offset
0000
00FF

- DS contains 0008H, which accesses the descriptor number 1 from the global descriptor table using a requested privilege level of 00.
- Descriptor number 1 contains a descriptor that defines the base address as 00100000H with a segment limit of 000FFH.
- This means that a value of 0008H loaded into DS causes the microprocessor to use memory locations 00100000H–001000FFH.