Important Instructions to examiners:

1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills).
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q1. Solve any five: **[5x4=20]**

a) Describe divide by zero and single step interrupt of 8086.

Ans. **Divide by zero interrupt** **[2 marks]**

It is the type 0 interrupt in 80x86 family of processors.

When the result (quotient) of division operation (execution of DIV or IDIV instruction) is too large to be stored in destination register, the microprocessor generates the type 0 interrupt. The division by zero is the worst case of such division.

The 4 byte vector for this interrupt starts at address 0000H: 0000H. The first two bytes hold the IP (offset) and next two bytes hold the CS (base address) of the ISR for divide-by-zero interrupt.

Single step interrupt **[2 marks]**

It is the type 1 interrupt in 80x86 family of processors.

Single step execution is used for debugging the assembly language program. In single step mode, the processor executes one instruction and then stops. Thereafter the contents of register and memory location can be examined to see the results of the executed instruction. The programmer then gives the command to execute the next instruction.

When Trap Flag is set, the processor enters the single step mode.

The 4 byte vector for this interrupt starts at address 0000H: 0004H. The first two bytes hold the IP (offset) and next two bytes hold the CS (base address) of the ISR for single step interrupt.

b) Explain segment registers and base and index registers of 80286**.**

Ans. ***Note: diagram not compulsory***

Segment registers: **[½ mark each – total 2marks]**

CS: **C**ode **S**egment – it is 16-bit register used to hold the starting/ base/ segment base address of code segment of memory.

DS: **D**ata **S**egment – it is 16-bit register used to hold the starting/ base/ segment base address of data segment of memory.

SS: **S**tack **S**egment – it is 16-bit register used to hold the starting/ base/ segment base address of stack segment of memory.

ES: **E**xtra **S**egment – it is 16-bit register used to hold the starting/ base/ segment base address of extra segment of memory.

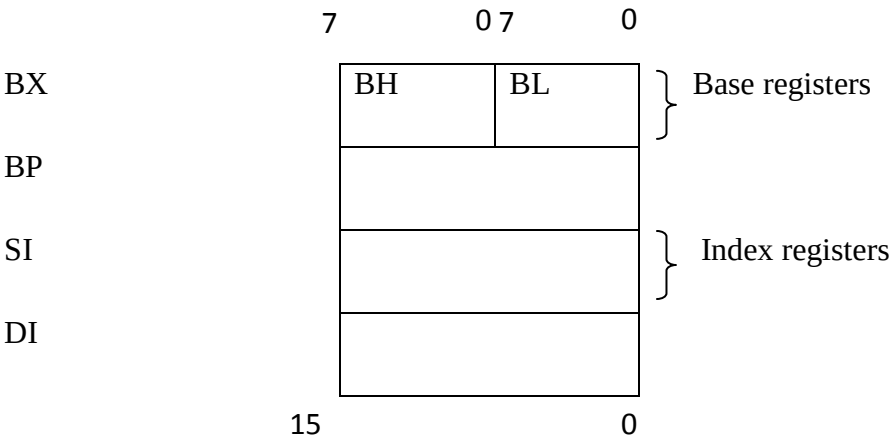
15 0

CS		Code segment selection
DS		data segment selection
SS		stack segment selection

ES | | extra segment selection

Segment Registers

16-bit register name



General registers

Base registers: **[½ mark each- total 1mark]**

BX & BP are known as Base Registers.

BX: it is generally referred to as **B**ase register. It is used as a 16-bit general purpose register or two individual registers BH & BL, each 8-bit in size.

BP: it is a **B**ase **P**ointer. It is used to store the offset address and it is associated with the Stack Segment.

Index registers: **[½ mark each - total 1mark]**

SI & DI are the Index Registers.

SI: it is **S**ource **I**ndex Register. It is associated with data segment. It is used to store the 16-bit offset address of the source. (It is implicitly used in string instructions)

DI: it is **D**estination **I**ndex Register. It is associated with extra segment. It is used to store the 16-bit offset address of the destination. (It is implicitly used in string instructions)

c) List the salient features of 80386.

Ans. Any Four features **[1 mark each]**

1. It is a 32-bit microprocessor i.e. all the registers are 32-bit in size.
2. It can operate on 8-bit, 16-bit or 32-bit data.
3. It has 32-bit data bus.
4. It has 32-bit address bus. Thus it can access 2^{32} i.e.4GB of physical memory and 64 TB of virtual memory.
5. The memory management of 80386 supports virtual memory, paging and four level protection. The available physical memory is divided into pages of 4KB each.
6. The supports 80387 math coprocessor. The 80387 supports higher precision (32-bit) numerical operations.
7. It has 8 debug registers (DR0-DR7) which provides hardware debugging and control.
8. It has on-chip address translation cache.
9. It is upward compatible all the previous members of 80x86 family.
10. It runs at 20MHz and 33MHz.
11. It supports three operating modes: Real, Protected Virtual (PVAM) and Virtual 8086.

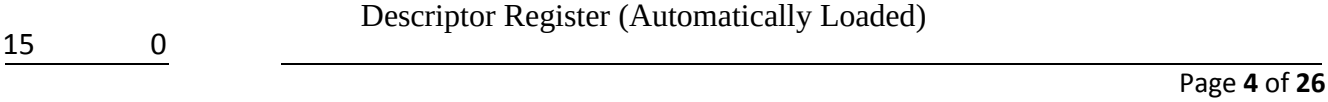
d) Explain GDTR and LDTR of 80386.

Ans. GDTR – Global Descriptor Table Register **[1M for diagram, 1M for description]**

GDTR is a 48-bit system address register. The lower 16-bit (2 bytes) hold the limit and higher 32-bits (4 bytes) hold the base address of the Global Descriptor Table (GDT).

The GDTR is loaded when the instruction LGDT (Load Global Descriptor Table) is executed at privilege level 0.

LDTR – Local Descriptor Table Register **[1M for diagram, 1M for description]**



LDTR	Selector	32-bit linear base address	32-bit segment limit	Attributes

LDTR is a system segment register which holds the 16-bit selector that points to the LDT descriptor in GDT. Whenever a selector is loaded into the LDTR, the corresponding descriptor is located in Global Descriptor Table. The contents of this descriptor are automatically copied to the corresponding descriptor register. The descriptor register holds the 32-bit linear base address, the 32-bit limit and the attributes for the LDT. The LDT pointed by this 32-bit linear base address is the descriptor table for the current task.

e) Mention and explain the functions of four processing units of 80286.

Ans. 80286 has four functional units: **[1/2 mark for listing the four units]**

Address Unit (AU)
Bus Unit (BU)
Instruction Unit (IU)
Execution Unit (EU)

Address Unit **[1/2 mark]**

The Address Unit (AU) is responsible for calculating the physical address of the memory location. Using this address, processor can access the code (instructions) & data from the memory and the peripherals connected to the microprocessor. This physical address is forwarded to the Bus Unit (BU).

Bus Unit **[1 mark]**

The physical address received by the Bus Unit (BU) is transmitted over the address bus A23-A0 through Address Latches and Drivers. The BU has Bus Control module which controls the Prefetcher module. The Prefetcher performs the task of prefetching the instructions from the memory. And these prefetched instructions are stored in 6-byte Prefetch Queue. The Data Transceivers interfaces and controls the internal data buses with the system bus. Another major module in the BU is Processor Extension Interface, which takes care of communication between the CPU and the co-processor.

Instruction Unit **[1 mark]**

The Instruction Decoder in the Instruction Unit (IU) accepts the instruction from the 6 Byte Prefetch Queue. These instructions are decoded by the Decoder one by one and stored into the Decoded Instruction Queue. This Decoded Instruction Queue can store 3 instructions which are already decoded and are ready for execution.

Thus the 6-byte instruction queue and the prefetcher module contribute for pipelining. And 3 byte decoded instruction queue along with pipelining help in overall improvement of performance (speed) of the microprocessor.

Execution Unit **[1 mark]**

These decoded instructions drive the control circuitry in the Execution Unit (EU), which is responsible for executing the instructions. The ALU (Arithmetic and Logic Unit) is the heart of EU, which carries out arithmetic and logical operations. The data bus sends the data required for the instruction execution. The result of the operation may be stored in the registers and it can be also stored back to memory through internal data bus by the BU. The EU also has the Register Bank for storing the data as scratch pad or used as Special purpose registers.

f) Explain function of debug register and test register in 80386.

Ans. Debug Registers: **[1M for diagram, 1M for description]**

31 0

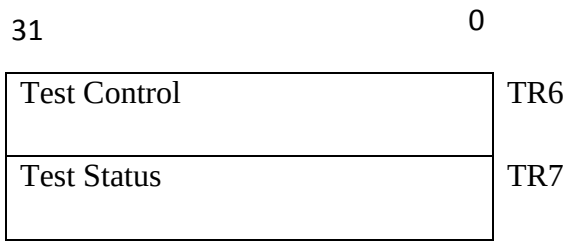
Linear Breakpoint Address 0	DR0
Linear Breakpoint Address 1	DR1
Linear Breakpoint Address 2	DR2
Linear Breakpoint Address 3	DR3
Intel Reserved	DR4
Intel Reserved	DR5
Breakpoint Status	DR6
Breakpoint Control	DR7

There are eight debug registers DR0 to DR7 for hardware debugging. The DR0 to DR3 are used to store program controllable breakpoint addresses. The DR4 and DR5 are not used and are reserved by Intel. The DR6 and DR7 are used to hold the breakpoint status and breakpoint control information respectively.

Test registers of 80386: **[1 mark for diagram, 1 mark for description]**

The 80386 has two test registers for page caching. The registers are TR6 – Test Control and TR7 – Test Status. TR6 & TR7 are used for translation look aside buffer (TLB). TLB holds page table address translation to reduce the no. of memory required for page table translation.

The test registers are used to perform the confidence checking on the paging. TR6 is the TLB testing command register. By writing into this register, you can either initiate a write directly into the TLB or perform a mock TLB lookup. TR7 is the TLB testing data register. When a program is performing writes, the entry to be stored is contained in this register, along with cache set information.



g) List the pins of Intel 80386.

Ans. Any 8 pins should be listed **[1/2 mark for each pin]**

Clk_2	D/C#	READY#	ERROR#
D_0-D_{31}	M/IO#	BS_{16}#	PEREQ#
A_2-A_{31}	LOCK#	HOLD	INTR
BE_0# to BE_3#	ADS#	HLDA	NMI
W/R#	NA#	BUSY#	RESET

(Note: # indicates the active low signals)

Q2. Solve any two: **[2x8=16]**

1) Explain the following signals of Intel 80286. a) $\overline{\text{LOCK}}$ b) $\overline{\text{READY}}$ c) M/$\overline{\text{IO}}$ d) COD/ $\overline{\text{INTA}}$

Ans.

LOCK: **[2 marks]**

It is the active low output pin. It is used to prevent the other masters from gaining the control of the bus for the current bus cycles. This pin is activated by a LOCK instruction prefix, or automatically by the hardware during XCHG, interrupt acknowledge or descriptor table access.

READY: **[2 marks]**

This is the active low input pin, used to insert the wait states in a bus cycle, for interfacing low speed peripherals. This signal is neglected during HLDA cycle.

M/IO: **[2 marks]**

This output line differentiates memory operations from I/O operations. If this signal is low (i.e. "0"), it indicates that an I/O cycle or INTA cycle is in process and if it is high (i.e. "1"), it indicates that the memory or a HALT cycle is in process.

COD/INTA: **[1M for description,1M for the table]**

This output signal, in combination with M/IO# signal and S1# - S0# distinguishes different memory, I/O and INTA cycles.

Table 9.4 *80C286 Cycle Status Definition (Intel Corp.)*

80C286 Bus Cycle Status Definition				
COD/$\overline{INTA}$	*M/$\overline{IO}$*	$\overline{S_1}$	$\overline{S_0}$	*Bus Cycle*
0 (LOW)	0	0	0	Interrupt acknowledge
0	0	0	1	Will not occur
0	0	1	0	Will not occur
0	0	1	1	None; not a status cycle
0	1	0	0	IF A_1 = 1 then halt; else shutdown
0	1	0	1	Memory data read
0	1	1	0	Memory data write
0	1	1	1	None; not a status cycle
1 (HIGH)	0	0	0	Will not occur
1	0	0	1	I/O read
1	0	1	0	I/O write
1	0	1	1	None; not a status cycle
1	1	0	0	Will not occur
1	1	0	1	Memory instruction read
1	1	1	0	Will not occur
1	1	1	1	None; not a status cycle

2) Explain separate code and data cache and branch prediction of Pentium µp.

Ans. Separate Code and Data cache: **[4 marks]**

The Pentium processor contains two 8KB cache memories, one for code and other for data. Both have 32-bit line size. The provision of separate code and data cache saves the time in searching pre-specified segments rather than the entire cache. This is supplemented by 64-bit data bus of the processor which ensures that the dual cache and dual pipelines are continuously supplied with data. Separate on-chip code and data cache increases the performance and reduces the bus conflicts.

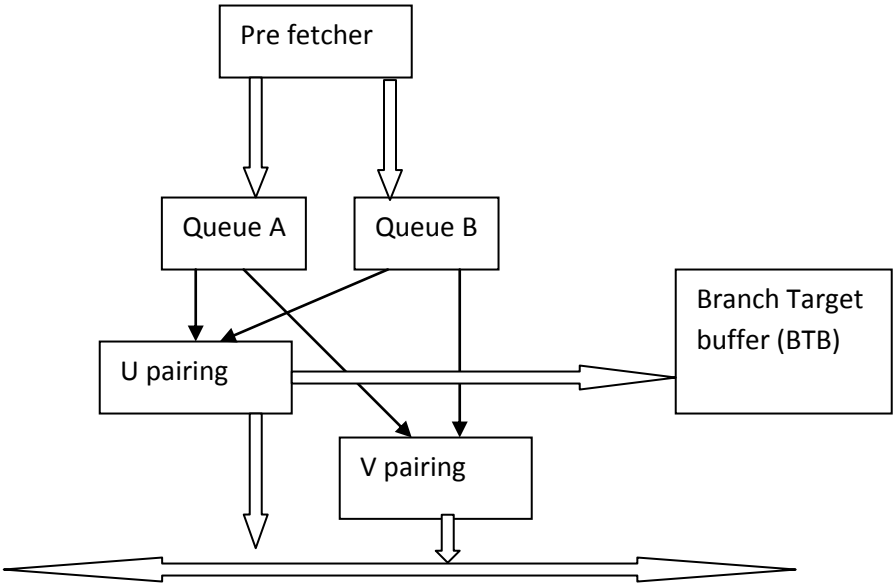
The separate cache also supports the superscalar organization. Also it is employed for efficient execution of Branch Prediction.

Branch Prediction: **[3 marks Description and 1 Marks for Diagram]**

The branch instructions occur frequently while running any application. These instructions change the normal sequential control flow of the program and may stall the pipelined execution in the Pentium system. Branches may be of two types: Conditional branch and unconditional branch. In case of conditional branch, the CPU has to wait till the execution stage to determine whether the condition is met or not.

The Pentium processor makes the dynamic branch prediction using a Branch Target Buffer (BTB). To efficiently predict branches, the Pentium uses two prefetch buffers. One buffer prefetches code in linear fashion, while the other prefetches instructions based on address in the branch target buffer. As a result the needed code is prefetched before it is required for execution. The Pentium processors prediction algorithm not only forecast the simple branch choices but also supports more complex branch prediction for example, within nested loops.

This is achieved by storing multiple branch address in the branch prediction buffer. The design of the branch target buffer allows 256 addresses to be stored and thus the prediction algorithm can forecast up to 256 branches.



3) Describe the loading of MS-DOS

Ans. . **[Description 4M, diagram 4M]**

When the system is reset or started, the program execution begins at the address 0FFFF0H. The control is transferred to system test code, power on self-test (POST). Then the control is transferred to the ROM bootstrap routine, which reads the bootstrap from the first sector of the system startup disk into memory at some arbitrary address and transfers control to it.

The disk bootstrap checks to see if the "boot" disk contains DOS by checking the first sector of the root directory for the file IO.SYS and MSDOS.SYS. If these are not found in the boot disk, the user gets a prompt for changing the disk. If the two files are found, the disk bootstrap reads the files into memory and transfers the control to IO.SYS

The IO.SYS file consists of two separate modules. The first is the BIOS, which contains the linked set of resident device drivers for the console, auxiliary port, printer, clock devices and some hardware specific initialization code. Second module consists of system initialization program, which determines the RAM size in the PC. Then it loads the MSDOS.SYS program to its final memory location of the DOS Kernel program.

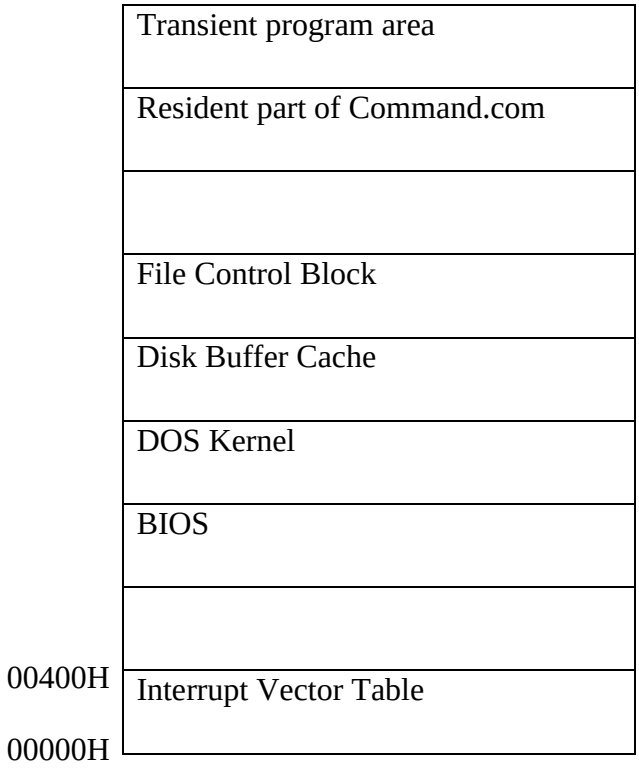
The DOS Kernel initializes its tables and sets up its various work areas. It sets up the various interrupt vectors for the DOS interrupts20H-2FH pointing them to appropriate service routine. It then loads and executes the device drivers. Now it returns the control to system initialization program (SYSINIT).

The SYSINIT calls MS-DOS file service to open the CONFIG.SYS file. It contains the list of additional device drivers that the user wants in his system. The required drivers are loaded into the memory, initialized by calls to their INIT modules, and linked into their device driver list.

After SYSINIT calls the EXEC function to load the command interpreter (shell). Once the interpreter is loaded, it displays a prompt and waits for the user to enter the command.

Top of the RAM

ROM Bootstrap routine
:
:
Transparent part of Command.com



Sharing of system by different components of DOS

Q.3 Solve any two: **(2x8=16)**

1) Draw and explain PVAM of 80286

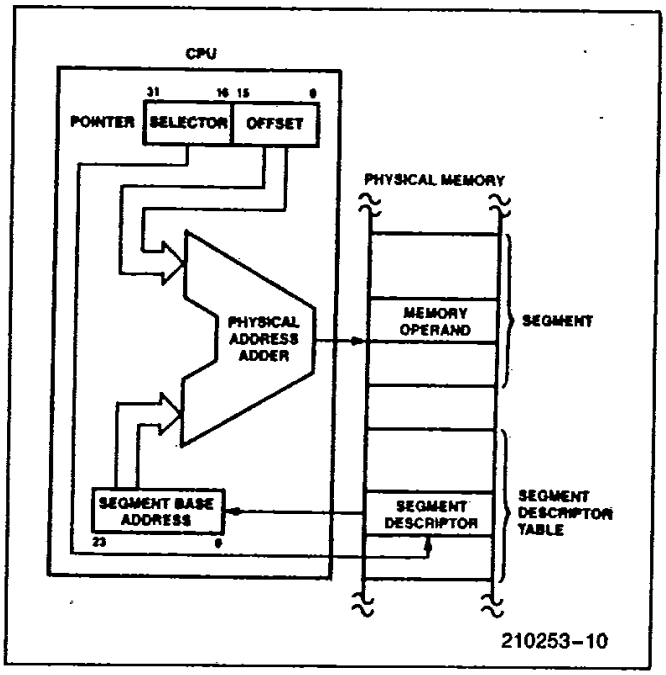
Ans. **[4 marks for diagram and 4 marks for description]**

The 80286 executes a fully upward compatible superset of the 8086 instruction set in protected virtual address mode (protected mode). Protected mode also provides memory management and protection mechanisms and associated instructions.

The 80286 enters protected virtual address mode from real address mode by setting the PE (Protection Enable) bit of the machine status word with the Load Machine Status Word (LMSW) instruction. Protected mode offers extended physical and virtual memory address space, memory protection mechanisms, and new operation to support operating systems and virtual memory.

All registers, instructions, and addressing modes described in the 80286 Base Architecture section of this Functional Description remain the same. Programs for the 8086, 88, 186, and real address mode 80286 can be run in protected mode; however, embedded constants for segment selectors are different.

The protected mode 80286 provides a 1 gigabyte virtual address space per task mapped into a 16 megabyte physical address space defined by the address pin $A_{23} - A_0$ and $\overline{BHE}$. The virtual address space may be larger than the physical address space since any use of an address that does not map to a physical memory location will cause a restart able exception.



As in real address mode, protected mode uses 32 – bit pointers, consisting of 16 – bit selector and offset components. The selector, however, specifies an index into a memory resident table rather than the upper 16 – bits of a real memory address. The 24 – bit base address of desired segment is obtained from the tables in memory. The 16 – bit offset is added to the segment base address to form the physical address as shown in fig above. The tables are automatically referenced by the CPU whenever a segment register is loaded with a selector. All 80286 instructions which load a segment register will reference the memory based tables without additional software. The memory based tables contain 8 byte values called descriptors.

2) Illustrate Difference between .com and .exe programs.

Ans: **[2 Marks each for Any four Points]**

Sr. No.	.COM	.EXE
1	In .COM program data, code and Stack reside in one segment	.EXE program can have multiple code,data an stack segment
2	The .COM files are compact and are loaded slightly faster than equivalent .EXE file, since these contain only the execution code	.EXE files contain unique header, a relocation map, a checksum and other information used by DOS along with the execution code
3	Near subroutine are used in .COM files	.EXE programs can contain more than one code segment so both near and far CALLS are used
4	CS starts at 0000H and IP at 0100H	Not fixed
5	Maximum length of program (code and data) is 65536 bytes(64 K) minus 256 bytes of PSP	.EXE program can be as large as available memory
6	In .COM format the size of the file is exacte the size of the program	In .EXE format the size of the file is size of the program plus the size of the header
7	.COM program does not require file header	.EXE program need file header for relocation process.

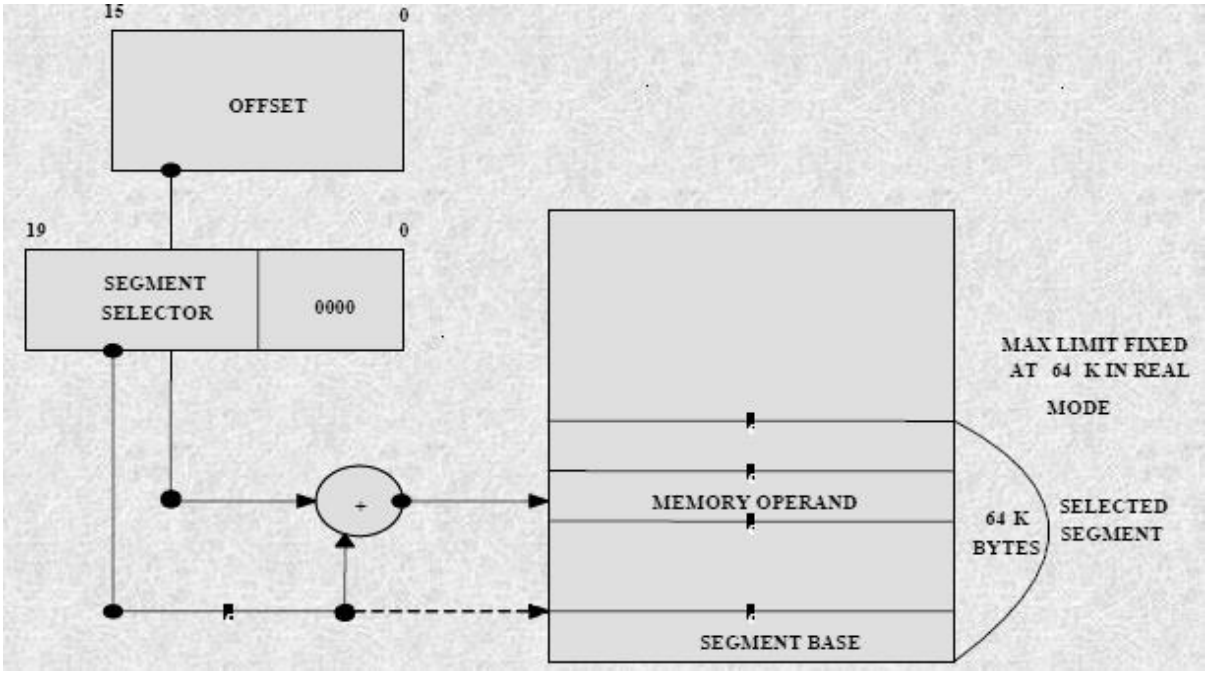
3) Explain Real Mode of 80386.

Ans. **[6 Marks for Description and 2 Marks for Diagram]**

After reset, the 80386 starts from memory location FFFFFFF0H under the real address mode. In the real mode, 80386 works as a fast 8086 with 32-bit registers and data types.

In real mode, the default operand size is 16 bit but 32- bit operands and addressing modes may be used with the help of override prefixes.

The segment size in real mode is 64k, hence the 32-bit effective addressing must be less than 0000FFFFFH. The real mode initializes the 80386 and prepares it for protected mode.



Memory Addressing in Real Mode:

In the real mode, the 80386 can address at the most 1Mbytes of physical memory using address lines A0-A19.

Paging unit is disabled in real addressing mode, and hence the real addresses are the same as the physical addresses.

To form a physical memory address, appropriate segment registers contents (16-bits) are shifted left by four positions and then added to the 16-bit offset address formed using one of the addressing modes, in the same way as in the 80386 real address mode.

The segment in 80386 real mode can be read, write or executed, i.e. no protection is available.

Any fetch or access past the end of the segment limit generates exception 13 in real address mode.

The segments in 80386 real mode may be overlapped or nonoverlapped.

The interrupt vector table of 80386 has been allocated 1Kbyte space starting from 00000H to 003FFH.

4. Solve any Two **(2x8=16)**

1) Explain following signals of 80286. a) $\overline{BHE}$ b) NMI c) PEREQ d) $\overline{PEACK}$

Ans. **[2 Marks for each Pin Description]**

$\overline{BHE}$ This output signals indicates that there is a transfer of data on the higher byte of data bus ($D_{15} – D_8$).

NMI: - the non maskable interrupt request is an active high, edge triggered input that is equivalent to an INTR signal of type 2. No acknowledgement cycle is needed to be carried out

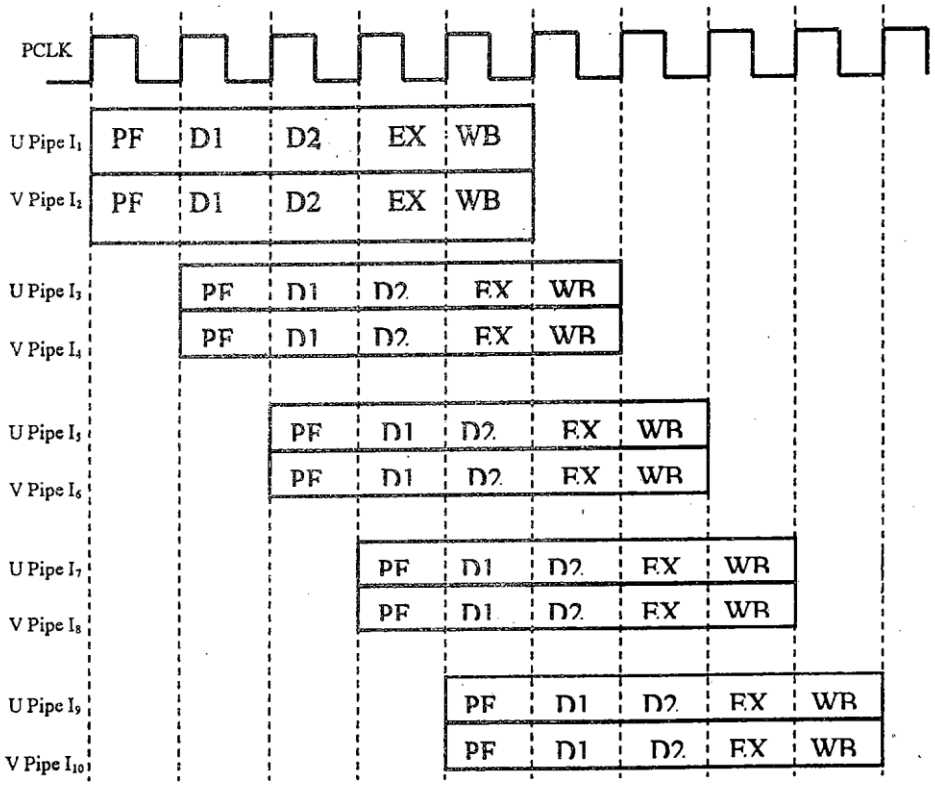
PEREQ and $\overline{PEACK}$:- (processor extension request and acknowledgement) the processor extension refers to coprocessor 80287. This pair of pins extended the memory management and protection capabilities of 80286 to the processor extension 80287. The PEREQ input request the 80286 to perform a data operand transfer for a processor extension. The $\overline{PEACK}$ active low output indicates to the processor extension that the requested operand is being transfer.

2. Explain Pipelining and Super Scalar Execution of Intel Pentium Microprocessor

Ans. **[6 Marks for Description and 2 Marks for diagram]**

In Pentium processor the integer instruction are executed in 5 stage pipeline. The pipeline stages are as follows:
PF – Prefetch
D1 – Instruction decode
D2 – Address generate
EX – Execute – ALU and Cache access
WB – Write back
Pentium processor is a superscalar machine, capable of executing two instruction in parallel. The five stage pipelines operate in parallel allowing integer instruction to execute in a single clock in each pipeline. The pipelines in Pentium processor are called U & V pipes and a process of issuing two instruction in parallel is term as 2 issue superscalar. There are two execution units in Pentium and the instruction pairing allows each unit to complete the execution of an instruction at the same time.

Assuming that all the instruction have followed the pairing rule, the five instruction pairs are shown in fig. the five clock cycles are used to perform five pipeline stages. In the clock cycle 1, the prefetch (PF) action is implemented. A pair of instruction is prefetched from the on- chip code cache during clock 1. This first pair is issued in parallel to the U and V pipelines for decoding purpose (D1 stage, while another pair is being prefetched (PF stage during the clock 2 cycle. In clock 3 cycle, the first instruction pair moves to decode 2 (D2) stage , while the second pair is now issued to the decode 1 (D1) stage of both the pipelines and the third pair of instructions is being fetched (PF stage). In this way, each pair of instruction can proceed to the next stage in the pipeline with each cycle of the processor clock (PCLK) . During clock cycle 5, the first instruction pair completes its execution. If we observe the column of CLK 5, the first pair in the last stage (WB) of the pipeline whereas the second pair is implementing the 4^{th} stage (EX) and the third instruction pair is at the 3^{rd} stage (D2) of the pipeline and so on. Thus, ten different instruction are present at the various pipeline stages during a single cycle. After the clock cycle 5, each succeeding clock cycle shows the completion of another instruction pair.



PF (Prefetch): During the PF stage the processor prefetches code from the instruction cache and aligns the code to the initial byte of the next instruction to be decoded. Because instructions are of variable length, this stage includes buffers to hold the line containing the instruction being decoded and the next consecutive line.

D_1 (First decode): In the D_1 stage, the processor decodes the instruction to generate a control word. A single control word executes instruction directly; more complex instructions require micro coded control sequencing in D_1

D2 (Second decode): in the D2 stage, the processor decodes the control word from D1 for use in the E stage. In addition, the processor generates addresses for data memory references.

E (Execute): In the E stage, the processor either accesses the data cache or calculates results in the MU (arithmetic logic unit), Barrel shifter or other functional units in the data path.

WB (Write back): In the WB stage, the processor updates the registers and flags with the instruction's results. All exceptional conditions must be resolved before an instruction can advance to WB.

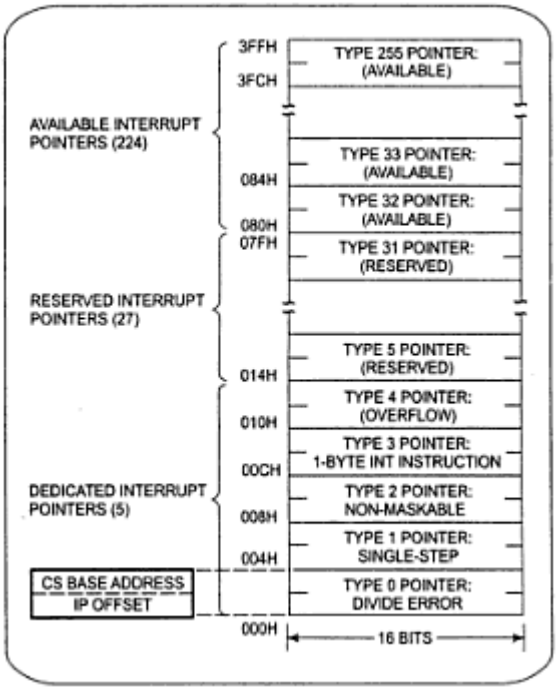
3. Draw and Explain IVT of 8086

Ans. **[4 Marks for description and 4 Marks for diagram]**

Figure shows the 256 interrupt vectors are arranged in the table in memory. Note that the instruction pointer value is put in as the low word of the vector, and the code segment register is put in as the high word of the vector. Each double word interrupt vector is identified by number from 0 to 255. Intel calls this number the type of interrupt.

The lowest five types are dedicated to specific interrupts, such as the divide – by – zero interrupt, the single step interrupt, and the non maskable interrupt. Interrupts types 5 to 31 are reserved by intel for using more complex microprocessor, such as the 80286, 80386, and 80486. The upper 224 interrupts types, from 32 to 255, are available for use of hardware and software interrupts,

In the figure the vector for each interrupt types requires four memory location. therefore, when the 8086 represent to a particular type interrupt, it automatically multiplies the type by 4 to produce the desired address in vector table. It then goes to the address in the table to get the starting address of the interrupt – service procedure.

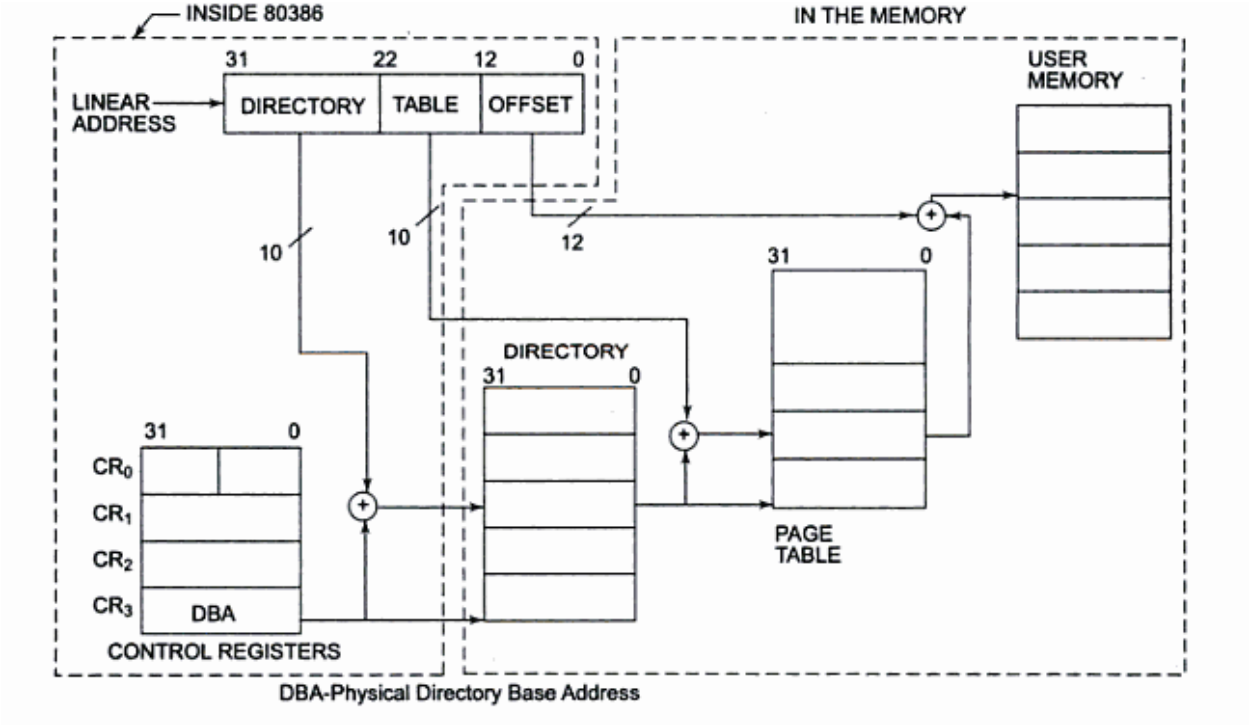


Q.5 Solve any two. **(2 x 8=16)**

1) Explain paging and segmentation of Intel 80386.

Ans: **(diagram :4 marks and description 4 marks)**

Segmentation is the process of dividing the main memory in 4 different segments (may not be of equal size.) The process of paging and segmentation together is as shown as below in diagram:



The paging unit operates under the control of segmentation unit. The paging unit if enabled converts linear addresses into physical address, in protected mode.

•**Paging Operation**: Paging is one of the memory management techniques used for virtual memory multitasking operating system.

•The segmentation scheme may divide the physical memory into a variable size segments but the paging divides the memory into a fixed size pages.

•The segments are supposed to be the logical segments of the program, but the pages do not have any logical relation with the program.

•The pages are just fixed size portions of the program module or data.

•The advantage of 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. Thus the memory requirement of the task is substantially reduced, relinquishing the available memory for other tasks.

•Whenever the other pages of task are required for execution, they may be fetched from the secondary storage.

•The previous page which are executed, need not be available in the memory, and hence the space occupied by them may be relinquished for other tasks.

•Thus paging mechanism provides an effective technique to manage the physical memory for multitasking systems.

•**Paging Unit:** The paging unit of 80386 uses a two level table mechanism to convert a linear address provided by segmentation unit into physical addresses.

The paging unit converts the complete map of a task into pages, each of size 4K. The task is further handled in terms of its page, rather than segments.

The paging unit handles every task in terms of three components namely page directory, page tables and page itself.

•**Paging Descriptor Base Register:** The control register CR2 is used to store the 32-bit linear address at which the previous page fault was detected.

The CR3 is used as page directory physical base address register, to store the physical starting address of the page directory.

The lower 12 bit of the CR3 are always zero to ensure the page size aligned directory. A move operation to CR3 automatically loads the page table entry caches and a task switch operation, to load CR0 suitably.

•**Page Directory :** This is at the most 4Kbytes in size. Each directory entry is of 4 bytes,thus a 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 page tables.

•**Page Tables:** Each page table is of 4Kbytes in size and many contain a maximum of 1024 entries. The page table entries contain the starting address of the page and the statistical information about the page.

•The upper 20 bit page frame address is combined with the lower 12 bit of the linear address. The address bits A12- A21 are used to select the 1024 page table entries. The page table can be shared between the tasks.

2) Explain different types of interrupts in 8086.

Ans : (main types: hardware ,software , exceptions:6 marks, maskable and non maskable -2 marks)

INTERRUPTS

Interrupts in 8086 can further be divided in three main types as :

(i)**Hardware Interrupts** and

(ii)**Software Interrupts**

(iii) Exceptions

(i) Hardware Interrupts (External Interrupts). The Intel microprocessors **support hardware interrupts** through:

- Two pins that allow interrupt requests, INTR and NMI
- One pin that acknowledges, INTA, the interrupt requested on INTR.

INTR and NMI

- **INTR** is a maskable hardware interrupt. The interrupt can be enabled/disabled using STI/CLI instructions or using more complicated method of updating the FLAGS register with the help of the POPF instruction.
- **NMI** is a non-maskable interrupt. Interrupt is processed in the same way as the INTR interrupt. Interrupt type of the NMI is 2, i.e. the address of the NMI processing routine is stored in location 0008h. This interrupt has higher priority than the maskable interrupt.
- – **Ex: NMI, INTR.**

(ii) Software Interrupts (Internal Interrupts and Instructions) .S**oftware interr**upts can be caused by:

- INT instruction - breakpoint interrupt. This is a type 3 interrupt.
- INT <interrupt number> instruction - any one interrupt from available 256 interrupts.
- Single-step interrupt - generated if the TF flag is set. This is a type 1 interrupt. When the CPU processes this interrupt it clears TF flag before calling the interrupt processing routine.

(iii) Exceptions

Exceptions are the interrupts which when occur must be processed by the processor. They don' t allow the processor to execute the remaining part of the program unless they are processed.

Exceptions take the control of execution to some other part for their processing which is executed by the processor.

Processor exceptions are: Divide Error (Type 0), Unused Opcode (type 6) and Escape opcode (type 7).

There are further two sub types of interrupt in the 8086 microprocessor, internal and external hardware interrupts. Hardware interrupts occur when a peripheral device asserts an interrupt input pin of the microprocessor. Whereas internal interrupts are initiated by the state of the CPU (e.g. divide by zero error) or by an instruction.

Maskable Interrupts

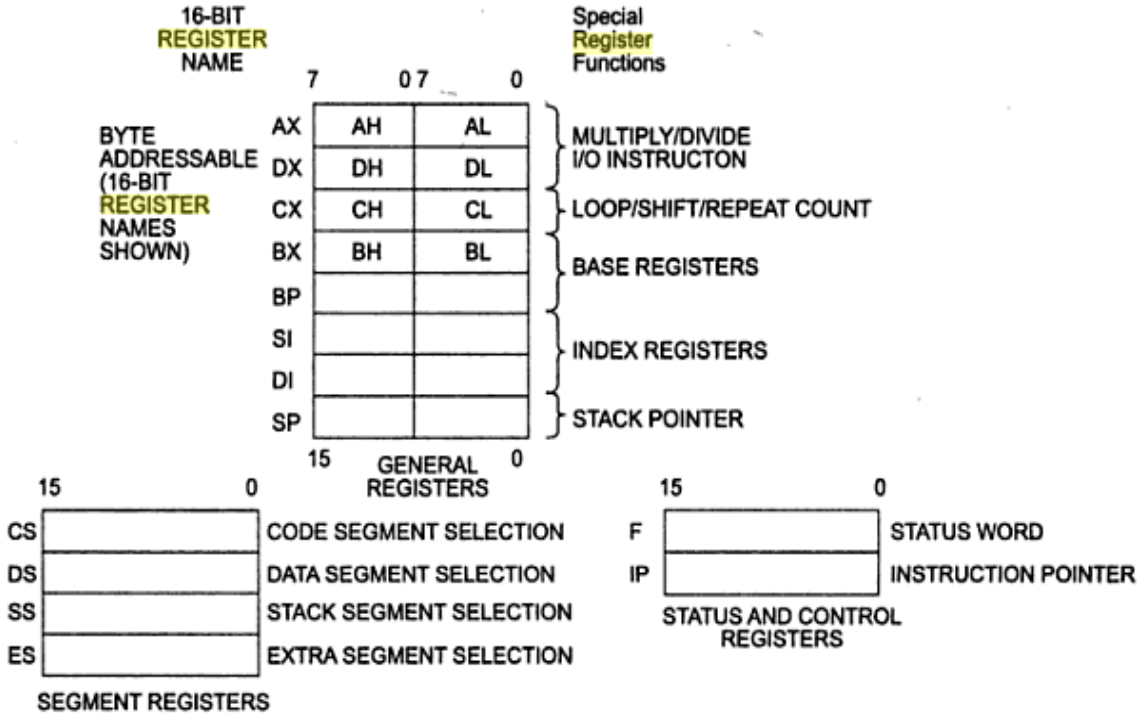
- The processor can inhibit certain types of interrupts by use of a special interrupt mask bit. This mask bit is part of the flags/condition code register, or a special interrupt register. In the 8086 microprocessor if this bit is clear, and an interrupt request occurs on the Interrupt Request input, it is ignored.
- **Non-Maskable Interrupts**

There are some interrupts which cannot be masked out or ignored by the processor. These are associated with high priority tasks which cannot be ignored (like memory parity or bus faults). In general, most processors support the Non-Maskable Interrupt (NMI). This interrupt has absolute priority, and when it occurs, the processor will finish the current memory cycle, then branch to a special routine written to handle the interrupt request.

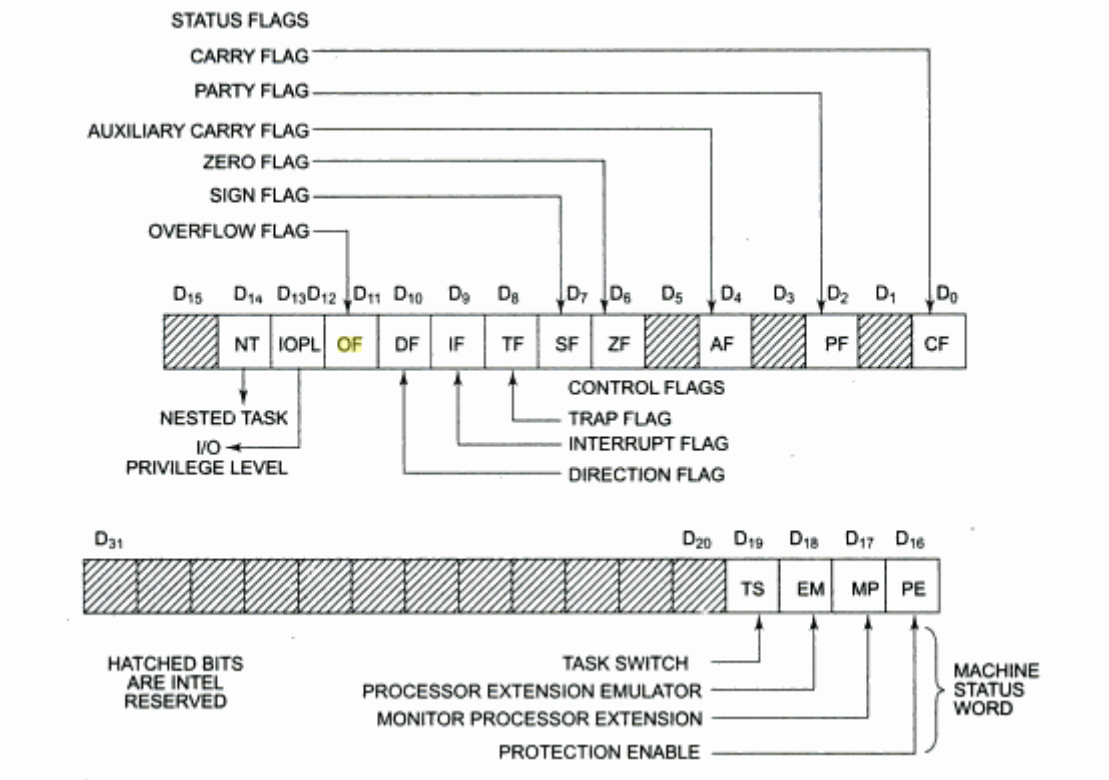
3) Draw different registers of Intel 80286.

Ans. **[Neat labeled diagram of register set :8 marks]**

The register set in 80286 is as shown below:



Flag register organsation is as shown below :

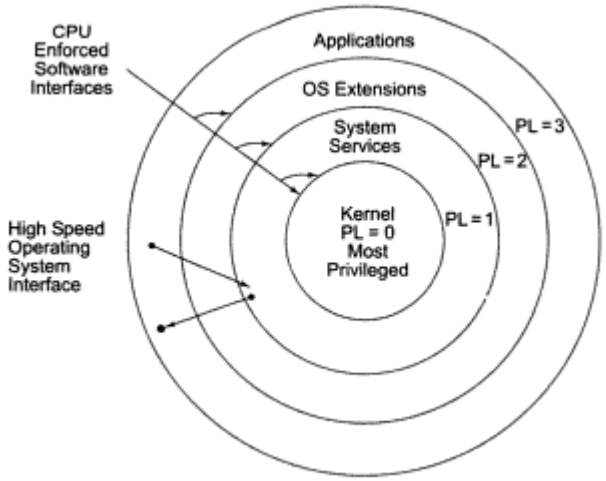


Q.6 Solve any two: **(2 x 8=16)**

1) Describe privilege protection of 80286.

Ans. **[diagram: 2 marks, (description of task , descriptor and selector privilege :2 marks each)]**

The four level protection mechanism in 80286 is as shown below:



The task always executes at one of the four privilege levels. A task privilege level at any specific instant is called the current privilege level (CPL) and is defined by the lower two bits of the CS register. CPL cannot change during execution in a single coded segment. A task's CPL may only be changed by control transfer through gate descriptors to a new code segment (See control Transfer). Task new begin executing at the CPL value specified by the code segment when the task is initiated by a task switch operation. A task executing at level 0 can access all data segments defined in the GDT and the task's LDT and is considered the most trusted level. A task executed at level 3 has the most restricted access to data and is considered the least trusted level.

Descriptor privilege:

The descriptor privilege is specified by the DPL field of the access rights byte. The DPL specifies the least task privilege level (CPL) that may be used to refer to the descriptor. Hence the task with privilege level 0. Can refer to all the lower level privilege descriptors. However, the task with privilege level 3 can refer to only level 3 descriptors. This rule applies to all the descriptors except the LTD descriptors.

Selector privilege:

This privilege is specified by the RPL field of a segment register (selector). A selector RPL may use a less trusted privilege than the current privilege level for further use. This is known as the effective privilege level (EPL) of the task. The effective privilege level is thus the maximum of RPL and CPL (i.e. numeric maximum and privilege minimum). The RPL is used to ensure that the pointer parameters passed to a more privileged procedure are not given the access of data at the privilege higher than the caller routine. The pointer testing instructions are used for this purpose.

2) List features of Pentium µp.

Ans.

(Any 8 Features 1 Mark each)

Features of Pentium Processor are as given below :

1. Superscalar Execution
2. Pipeline Architecture
3. Branch Target Buffer
4. Dual 8-KB On-Chip Caches
5. Write-Back Cache
6. 64-Bit Bus
7. Instruction Optimization

8. On chip Floating-Point unit and Optimization
9. Pentium Extensions

3)Explain descriptors of Intel 80386.

Ans : **[3 Marks for Diagram and 5 Marks for Description]**

•The 80386 has five types of descriptors listed as follows:

1.Code or Data Segment Descriptors.

2.System Descriptors.

3.Local descriptors.

4.TSS (Task State Segment) Descriptors.

5.GATE Descriptors.

•***DESCRIPTORS***: The 80386 descriptors have a 20-bit segment limit and 32-bit segment address. The descriptor of 80386 are 8-byte quantities access right or attribute bits along with the base and limit of the segments.

•***Descriptor Attribute Bits***: The A (accessed) attributed bit indicates whether the segment has been accessed by the CPU or not.

•The TYPE field decides the descriptor type and hence the segment type.

•The S bit decides whether it is a system descriptor (S=0) or code/data segment descriptor (S=1).

•The DPL field specifies the descriptor privilege level.

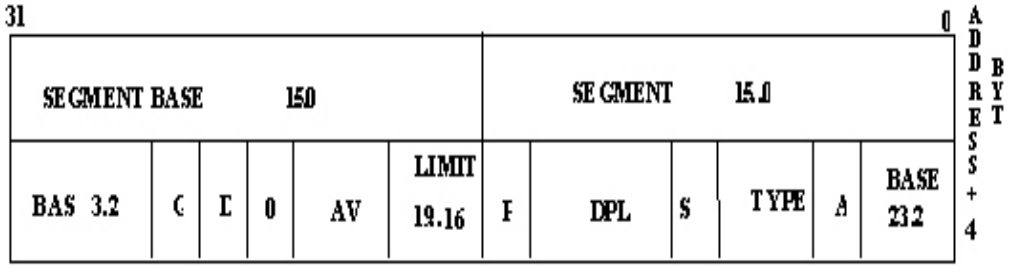
•The D bit specifies the code segment operation size. If D=1, the segment is a 32-bit operand segment, else, it is a 16-bit operand segment.

•The P bit (present) signifies whether the segment is present in the physical memory or not. If P=1, the segment is present in the physical memory.

•The G (granularity) bit indicates whether the segment is page addressable. The zero bit must remain zero for compatibility with future process.

•The AVL (available) field specifies whether the descriptor is for user or for operating system.

•The 80386 provides a four level protection mechanism exactly in the same way as the 80286 does.



Structure as a descriptor

BASE Base Address of the Segment

LIMIT The Length of the Segment

P Present Bit -1= present, 0= not present

S segment descriptor -0= system descriptor

1= code or data segment descriptor

TYPE Type of segment

G Granularity Bit -1 = segment length is page granular

0= Segment length is byte granular

D Default Operation size

0 Bit must be zero

AVL available field for user or OS