UNIT – IV

ADVANCED MICROPROCESSORS

SALIENT FEATURES OF 80386:

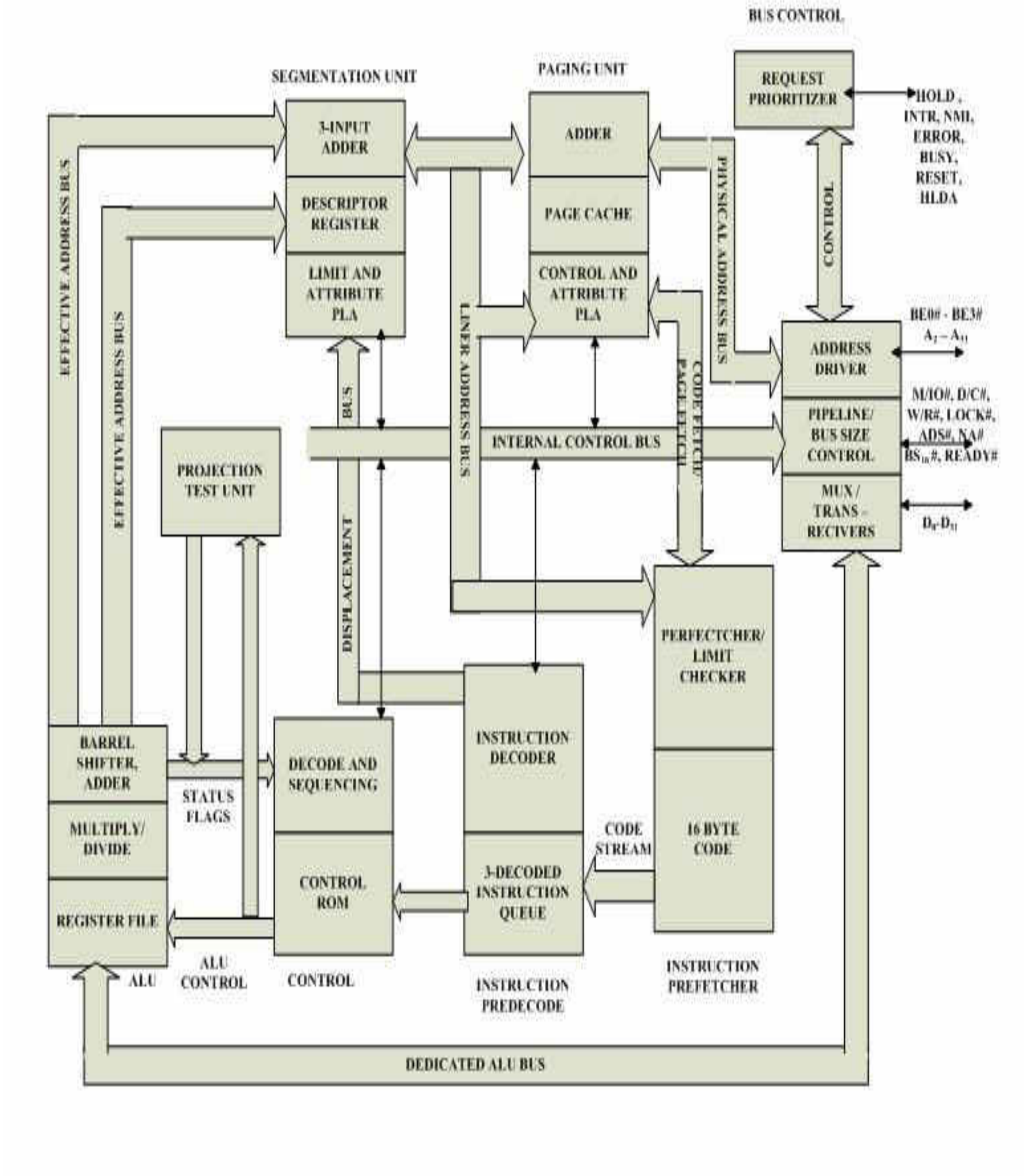
- 8, 16, 32-Bit Data Types
  - 8 General Purpose 32-Bit Registers
- Very Large Address Space
  - 4 Gigabyte Physical
  - 64 Terabyte Virtual
  - 4 Gigabyte Maximum Segment Size
- Integrated Memory Management Unit
  - Virtual Memory Support
  - Optional On-Chip Paging
  - 4 Levels of Protection
  - Fully Compatible with 80286
- Object Code Compatible with All 8086 Family Microprocessors
- Virtual 8086 Mode Allows Running of 8086 Software in a Protected and Paged System
- Hardware Debugging Support
- Optimized for System Performance
  - Pipelined Instruction Execution
  - On-Chip Address Translation Caches
  - 20, 25 and 33 MHz Clock
  - 40, 50 and 66 Megabytes/Sec Bus Bandwidth
- Numeric Support via Intel387TM DX Math Coprocessor
- Complete System Development Support
  - Software: C, PL/M, Assembler
  - System Generation Tools
  - Debuggers: PSCOPE, ICETM-386
- High Speed CHMOS IV Technology
  - 132 Pin Grid Array Package
  - 132 Pin Plastic Quad Flat Package

The processing mode of the 80386 also determines the features that are accessible. The 80386 has three processing modes

1. Protected Mode.
2. Real-Address Mode.
3. Virtual 8086 Mode.

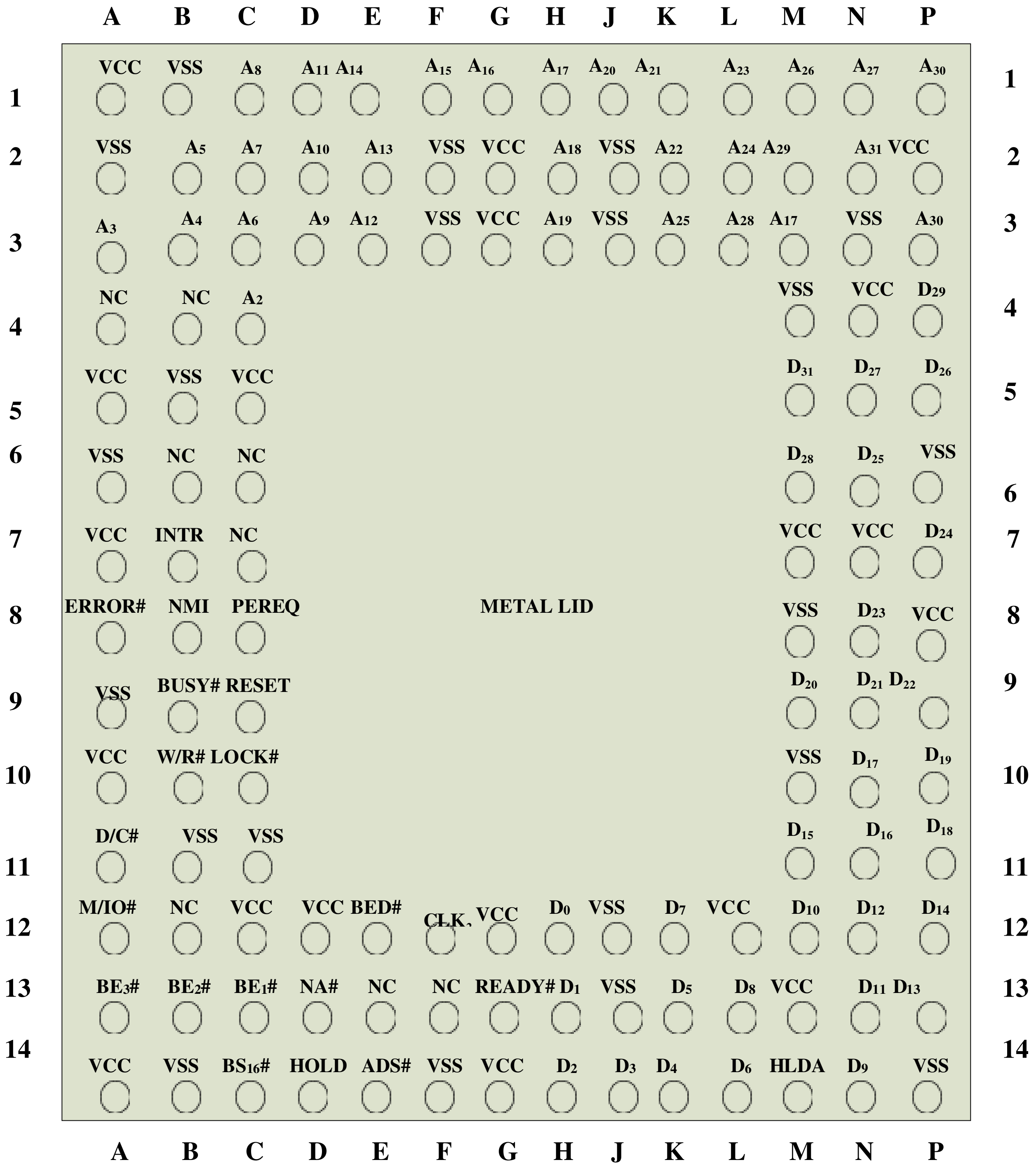
Architecture and Signal description of 80386

- The Internal Architecture of 80386 is divided into 3 sections.
- Central processing unit
- Memory management unit
- Bus interface unit
- Central processing unit is further divided into Execution unit and Instruction unit
- Execution unit has 8 General purpose and 8 Special purpose registers which are either used for handling data or calculating offset addresses.



80386 ARCHITECTURE

- The **Instruction unit** decodes the opcode bytes received from the 16-byte instruction code queue and arranges them in a 3- instruction decoded instruction queue.
- After decoding them pass it to the control section for deriving the necessary control signals. The barrel shifter increases the speed of all shift and rotate operations.
- The multiply / divide logic implements the bit-shift-rotate algorithms to complete the operations in minimum time.
- Even 32- bit multiplications can be executed within one microsecond by the multiply / divide logic.
- The Memory management unit consists of a Segmentation unit and a Paging unit.
- Segmentation unit allows the use of two address components, viz. segment and offset for relocability and sharing of code and data.
- Segmentation unit allows segments of size 4Gbytes at max.
- The Paging unit organizes the physical memory in terms of pages of 4kbytes size each.
- Paging unit works under the control of the segmentation unit, i.e. each segment is further divided into pages. The virtual memory is also organizes in terms of segments and pages by the memory management unit.
- The Segmentation unit provides a 4 level protection mechanism for protecting and isolating the system code and data from those of the application program.
- Paging unit converts linear addresses into physical addresses.
- The control and attribute PLA checks the privileges at the page level. Each of the pages maintains the paging information of the task. The limit and attribute PLA checks segment limits and attributes at segment level to avoid invalid accesses to code and data in the memory segments.
- The Bus control unit has a prioritizer to resolve the priority of the various bus requests. This controls the access of the bus. The address driver drives the bus enable and address signal A_0 – $A_{31.}$ The pipeline and dynamic bus sizing unit handle the related control signals.
- The data buffers interface the internal data bus with the system bus.



	A	B	C	D	E	F	G	H	J	K	L	M	N	P
1	VCC	VSS	A_8	A_{11}	A_{14}	A_{15}	A_{16}	A_{17}	A_{20}	A_{21}	A_{23}	A_{26}	A_{27}	A_{30}
2	VSS	A_5	A_7	A_{10}	A_{13}	VSS	VCC	A_{18}	VSS	A_{22}	A_{24}	A_{29}	A_{31}	VCC
3	A_3	A_4	A_6	A_9	A_{12}	VSS	VCC	A_{19}	VSS	A_{25}	A_{28}	A_{17}	VSS	A_{30}
4	NC	NC	A_2									VSS	VCC	D_{29}
5	VCC	VSS	VCC									D_{31}	D_{27}	D_{26}
6	VSS	NC	NC									D_{28}	D_{25}	VSS
7	VCC	INTR	NC									VCC	VCC	D_{24}
8	ERROR#	NMI	PEREQ					METAL LID				VSS	D_{23}	VCC
9	VSS	BUSY#	RESET									D_{20}	D_{21}	D_{22}
10	VCC	W/R#	LOCK#									VSS	D_{17}	D_{19}
11	D/C#	VSS	VSS									D_{15}	D_{16}	D_{18}
12	M/IO#	NC	VCC	VCC	BED#	CLK	VCC	D_0	VSS	D_7	VCC	D_{10}	D_{12}	D_{14}
13	BE_3#	BE_2#	BE_1#	NA#	NC	NC	READY#	D_1	VSS	D_5	D_8	VCC	D_{11}	D_{13}
14	VCC	VSS	BS_{16}#	HOLD	ADS#	VSS	VCC	D_2	D_3	D_4	D_6	HLDA	D_9	VSS

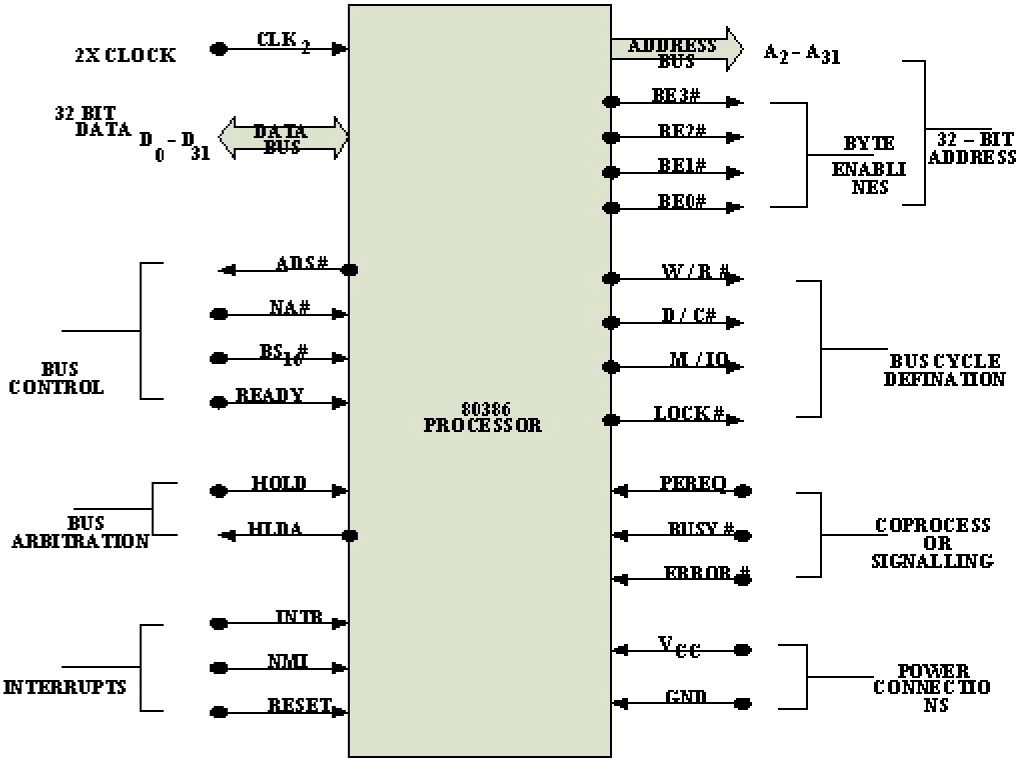
PIN DIAGRAM OF 80386

UNIT – IV
ADVANCED MICROPROCESSORS

Signal Descriptions of 80386

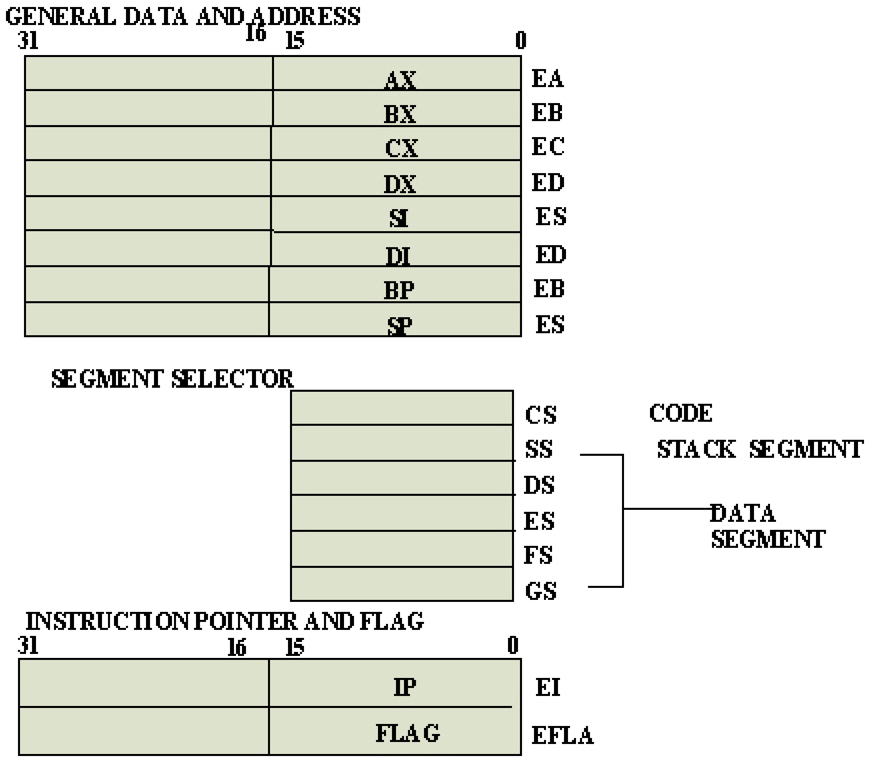
•**CLK$_2$:**The input pin provides the basic system clock timing for the operation of 80386.
•**D$_0$ – D$_{31}$**:These 32 lines act as bidirectional data bus during different access cycles.
•**A$_{31}$ – A$_2$**: These are upper 30 bit of the 32- bit address bus.
• **$\overline{\text{BE0}}$ to $\overline{\text{BE3}}$** : The 32- bit data bus supported by 80386 and the memory system of 80386 can be viewed as a 4- byte wide memory access mechanism. The 4 byte enable lines $\overline{\text{BE0}}$ to $\overline{\text{BE3}}$, may be used for enabling these 4 blanks. Using these 4 enable signal lines, the CPU may transfer 1 byte / 2 / 3 / 4 byte of data simultaneously.
•**ADS#:** The address status output pin indicates that the address bus and bus cycle definition pins(W/R#, D/C#, M/IO#, BE$_0$# to BE$_3$#) are carrying the respective valid signals. The 80383 does not have any ALE signals and so this signals may be used for latching the address to external latches.
•**READY#:** The ready signals indicates to the CPU that the previous bus cycle has been terminated and the bus is ready for the next cycle. The signal is used to insert WAIT states in a bus cycle and is useful for interfacing of slow devices with CPU.
•**VCC**: These are system power supply lines.
•**VSS**: These return lines for the power supply.
•**BS$_{16}$#:** The bus size – 16 input pin allows the interfacing of 16 bit devices with the 32 bit wide 80386 data bus. Successive 16 bit bus cycles may be executed to read a 32 bit data from a peripheral.
•**HOLD**: The bus hold input pin enables the other bus masters to gain control of the system bus if it is asserted.
•**HLDA**: The bus hold acknowledge output indicates that a valid bus hold request has been received and the bus has been relinquished by the CPU.
•**BUSY#:** The busy input signal indicates to the CPU that the coprocessor is busy with the allocated task.
•**ERROR#:** The error input pin indicates to the CPU that the coprocessor has encountered an error while executing its instruction.
•**PEREQ**: The processor extension request output signal indicates to the CPU to fetch a data word for the coprocessor.
•**INTR**: This interrupt pin is a maskable interrupt, that can be masked using the IF of the flag register.
•**NMI:** A valid request signal at the non-maskable interrupt request input pin internally generates a non- maskable interrupt of type2.
•**RESET**: A high at this input pin suspends the current operation and restart the execution from the starting location.
•**N / C** : No connection pins are expected to be left open while connecting the 80386 in the circuit.

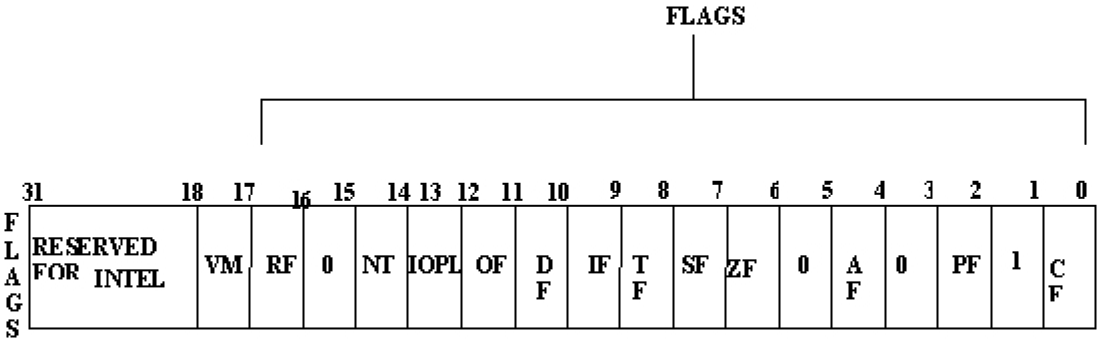
UNIT – IV
ADVANCED MICROPROCESSORS



Register Organisation of 80386:

•The 80386 has eight 32 - bit general purpose registers which may be used as either 8 bit or 16 bit registers.
•A 32 - bit register known as an extended register, is represented by the register name with prefix E.
•Example : A 32 bit register corresponding to AX is EAX, similarly BX is EBX etc.
•The 16 bit registers BP, SP, SI and DI in 8086 are now available with their extended size of 32 bit and are names as EBP,ESP,ESI and EDI.
•AX represents the lower 16 bit of the 32 bit register EAX.
• BP, SP, SI, DI represents the lower 16 bit of their 32 bit counterparts, and can be used as independent 16 bit registers.
•The six segment registers available in 80386 are CS, SS, DS, ES, FS and GS.
•The CS and SS are the code and the stack segment registers respectively, while DS, ES, FS, GS are 4 data segment registers.
•A 16 bit instruction pointer IP is available along with 32 bit counterpart EIP.





FLAG REGISTER OF 80386

•***Flag Register of 80386***: The Flag register of 80386 is a 32 bit register. Out of the 32 bits, Intel has reserved bits D_{18} to D_{31}, D_5 and D_3, while D_1 is always set at 1.Two extra new flags are added to the 80286 flag to derive the flag register of 80386. They are VM and RF flags.

•**VM - *Virtual Mode Flag***: If this flag is set, the 80386 enters the virtual 8086 mode within the protection mode. This is to be set only when the 80386 is in protected mode. In this mode, if any privileged instruction is executed an exception 13 is generated. This bit can be set using IRET instruction or any task switch operation only in the protected mode.

•***RF- Resume Flag***: This flag is used with the debug register breakpoints. It is checked at the starting of every instruction cycle and if it is set, any debug fault is ignored during the instruction cycle. The RF is automatically reset after successful execution of every instruction, except for IRET and POPF instructions.

•Also, it is not automatically cleared after the successful execution of JMP, CALL and INT instruction causing a task switch. These instruction are used to set the RF to the value specified by the memory data available at the stack.

•***Segment Descriptor Registers***: This registers are not available for programmers, rather they are internally used to store the descriptor information, like attributes, limit and base addresses of segments.

•The six segment registers have corresponding six 73 bit descriptor registers. Each of them contains 32 bit base address, 32 bit base limit and 9 bit attributes. These are automatically loaded when the corresponding segments are loaded with selectors.

•***Control Registers***: The 80386 has three 32 bit control registers CR_0, CR_2 and CR_3 to hold global machine status independent of the executed task. Load and store instructions are available to access these registers.

•***System Address Registers***: Four special registers are defined to refer to the descriptor tables supported by 80386.

•The 80386 supports four types of descriptor table, viz. global descriptor table (GDT), interrupt descriptor table (IDT), local descriptor table (LDT) and task state segment descriptor (TSS).

•***Debug and Test Registers***: Intel has provide a set of 8 debug registers for hardware debugging. Out of these eight registers DR_0 to DR_7, two registers DR_4 and DR_5 are Intel reserved.

•The initial four registers DR_0 to DR_3 store four program controllable breakpoint addresses, while DR_6 and DR_7 respectively hold breakpoint status and breakpoint control information.

•Two more test register are provided by 80386 for page cacheing namely test control and test status register.

Addressing modes of 80386:

The 80386 supports overall eleven addressing modes to facilitate efficient execution of higher level language programs.

•In case of all those modes, the 80386 can now have 32-bit immediate or 32- bit register operands or displacements.

•The 80386 has a family of scaled modes. In case of scaled modes, any of the index register values can be multiplied by a valid scale factor to obtain the displacement.

•The valid scale factor are 1, 2, 4 and 8.

•The different scaled modes are as follows.

•***Scaled Indexed Mode***: Contents of the an index register are multiplied by a scale factor that may be added further to get the operand offset.

•***Based Scaled Indexed Mode***: Contents of the an index register are multiplied by a scale factor and then added to base register to obtain the offset.

•***Based Scaled Indexed Mode with Displacement***: The Contents of the an index register are multiplied by a scaling factor and the result is added to a base register and a displacement to get the offset of an operand.

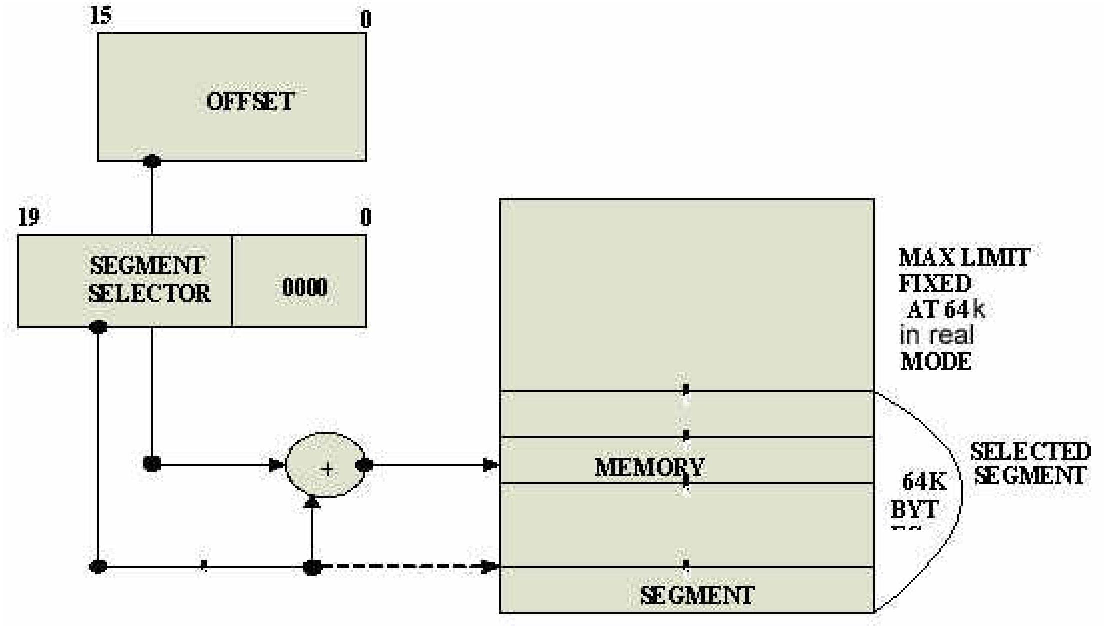
Data types of 80386:

The 80386 supports the following data types they are

- Bit
- Bit Field: A group of at the most 32 bits (4bytes)
- Bit String: A string of contiguous bits of maximum 4Gbytes in length.
- Signed Byte: Signed byte data
- Unsigned Byte: Unsigned byte data.
- Integer word: Signed 16-bit data.
- Long Integer: 32-bit signed data represented in 2's complement form.
- Unsigned Integer Word: Unsigned 16-bit data
- Unsigned Long Integer: Unsigned 32-bit data
- Signed Quad Word: A signed 64-bit data or four word data.
- Unsigned Quad Word: An unsigned 64-bit data.
- Offset: 16/32-bit displacement that points a memory location using any of the addressing modes.
- Pointer: This consists of a pair of 16-bit selector and 16/32-bit offset.
- Character: An ASCII equivalent to any of the alphanumeric or control characters.
- Strings: These are the sequences of bytes, words or double words. A string may contain minimum one byte and maximum 4 Gigabytes.
- BCD: Decimal digits from 0-9 represented by unpacked bytes.
- Packed BCD: This represents two packed BCD digits using a byte, i.e. from 00 to 99.

Real Address Mode of 80386:

•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.

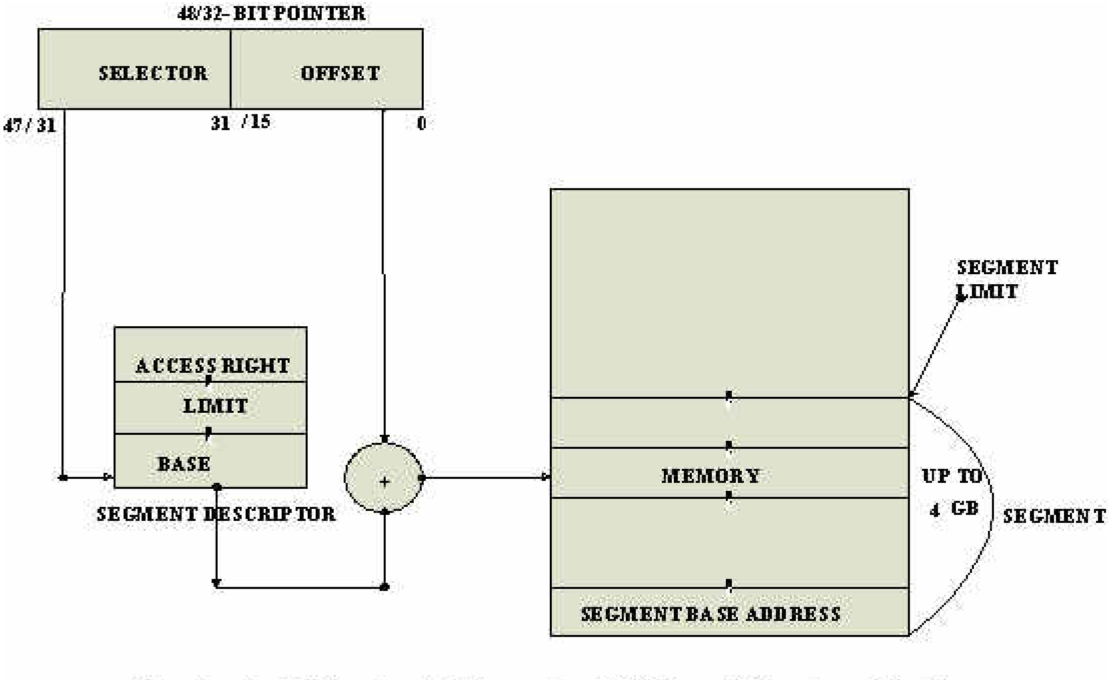


Physical Address Formation In Real Mode Of 80386

•***Memory Addressing in Real Mode***: In the 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, 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 generate exception 13 in real address mode.
•The segments in 80386 real mode may be overlapped or non-overlapped.
•The interrupt vector table of 80386 has been allocated 1Kbyte space starting from 00000H to 003FFH.

Protected Mode of 80386

•All the capabilities of 80386 are available for utilization in its protected mode of operation.
•The 80386 in protected mode support all the software written for 80286 and 8086 to be executed under the control of memory management and protection abilities of 80386.
•The protected mode allows the use of additional instruction, addressing modes and capabilities of 80386.



Protected Mode Addressing Without Paging Unit

ADDRESSING IN PROTECTED MODE:

In this mode, the contents of segment registers are used as selectors to address descriptors which contain the segment limit, base address and access rights byte of the segment.
•The effective address (offset) is added with segment base address to calculate linear address. This linear address is further used as physical address, if the paging unit is disabled, otherwise the paging unit converts the linear address into physical address.
•The paging unit is a memory management unit enabled only in protected mode. The paging mechanism allows handling of large segments of memory in terms of pages of 4Kbyte size.
•The paging unit operates under the control of segmentation unit. The paging unit if enabled converts linear addresses into physical address, in protected mode.

Segmentation:

•**Descriptor** ***tables***: These descriptor tables and registers are manipulated by the operating system to ensure the correct operation of the processor, and hence the correct execution of the program.
•Three types of the 80386 descriptor tables are listed as follows:
•GLOBAL DESCRIPTOR TABLE (GDT)
•LOCAL DESCRIPTOR TABLE (LDT)
•INTERRUPT DESCRIPTOR TABLE (IDT)
•***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.

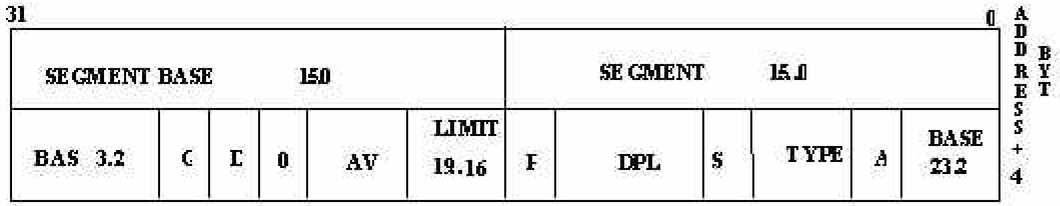
•The G (granularity) bit indicates whether the segment is page addressable. The zero bit must remain zero for compatibility with future process.

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

•The 80386 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.

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



Structure of 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

Paging:

•*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

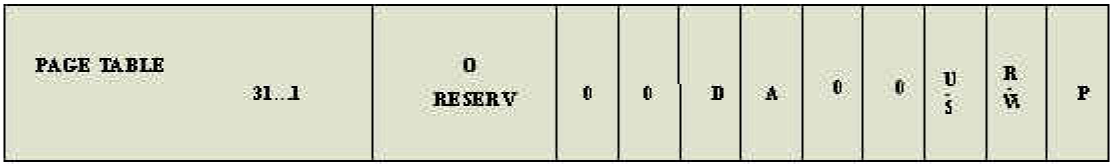
the space occupied by them may be relinquished for other tasks.
•Thus paging mechanism provides an effective technique to manage the physical memory for multitasking systems.

•***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 CR_2 is used to store the 32-bit linear address at which the previous page fault was detected.
•The CR_3 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 CR_3 are always zero to ensure the page size aligned directory. A move operation to CR_3 automatically loads the page table entry caches and a task switch operation, to load CR_0 suitably.

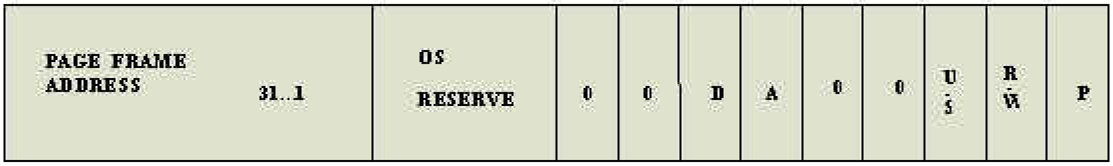
•***Page Directory*** : This is at the most 4Kbytes in size. Each directory entry is of 4 bytes, thus a total of 1024 entries are allowed in a directory.
•The upper 10 bits of the linear address are used as an index to the corresponding page directory entry. The page directory entries point to page tables.

•***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 A_{12}- A_{21} are used to select the 1024 page table entries. The page table can be shared between the tasks.
•The P bit of the above entries indicate, if the entry can be used in address translation.
•If P=1, the entry can be used in address translation, otherwise it cannot be used.
•The P bit of the currently executed page is always high.
•The accessed bit A is set by 80386 before any access to the page. If A=1, the page is accessed, else unaccessed.
•The D bit (Dirty bit) is set before a write operation to the page is carried out. The D-bit is undefined for page director entries.
•The OS reserved bits are defined by the operating system software.
•The User / Supervisor (U/S) bit and read/write bit are used to provide protection. These bits are decoded to provide protection under the 4 level protection model.
•The level 0 is supposed to have the highest privilege, while the level 3 is supposed to have the least privilege.
•This protection provide by the paging unit is transparent to the segmentation unit.



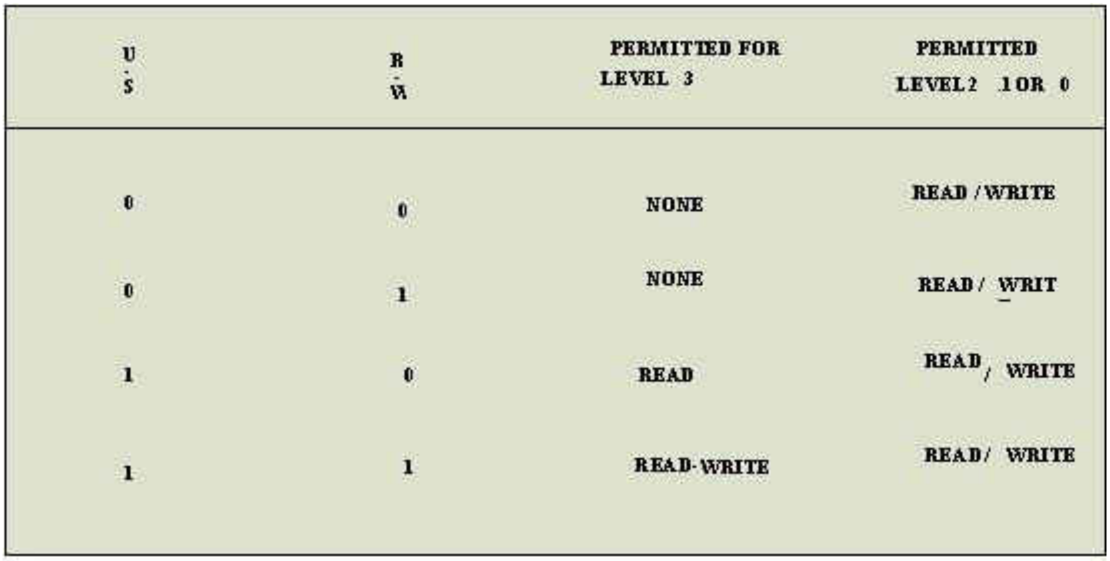
PAGE TABLE 31..1	O RESERV	0	0	D	A	0	0	$U/\overline{S}$	$R/\overline{W}$	P

PAGE DIRECTORY ENTRY

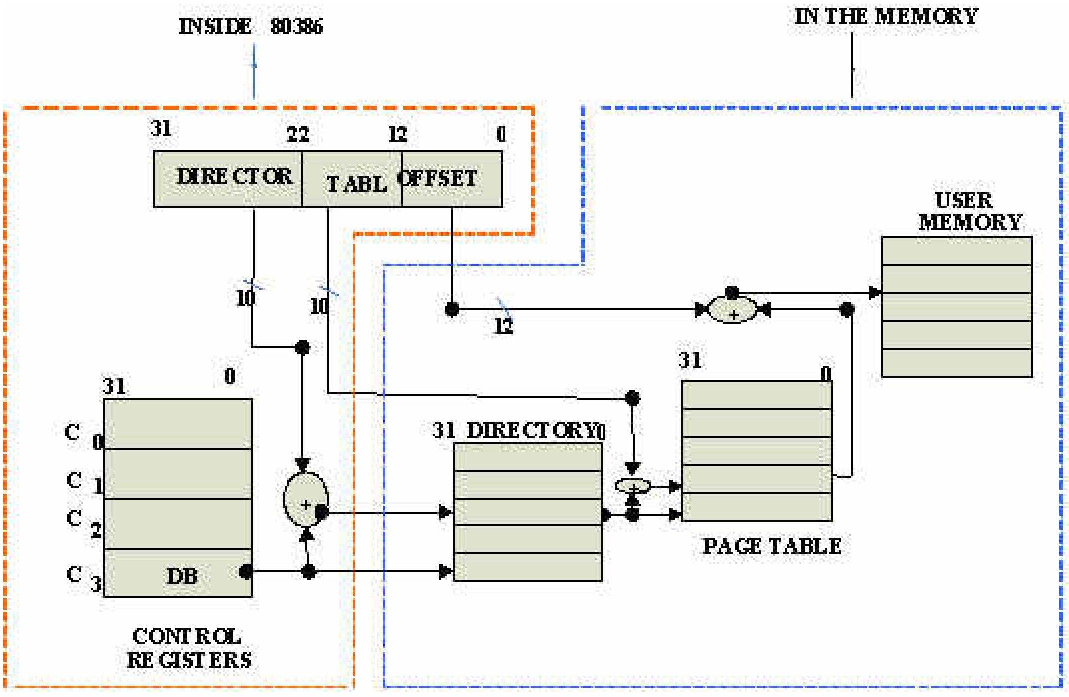


PAGE FRAME ADDRESS 31..1	OS RESERVE	0	0	D	A	0	0	$U/\overline{S}$	$R/\overline{W}$	P

PAGE TABLE ENTRY



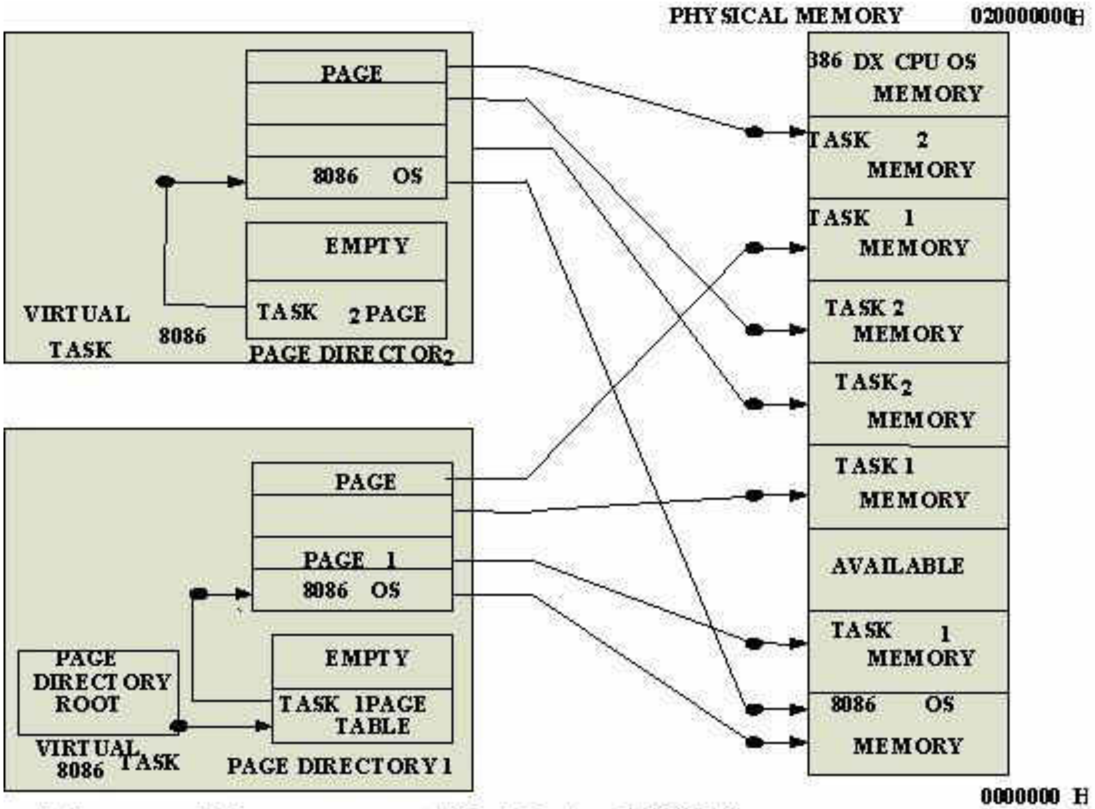
$U/\overline{S}$	$R/\overline{W}$	PERMITTED FOR LEVEL 3	PERMITTED LEVEL2 1OR 0
0	0	NONE	READ /WRITE
0	1	NONE	READ/ WRIT
1	0	READ	READ/ WRITE
1	1	READ-WRITE	READ/ WRITE



DBA Physical directory base address

Virtual 8086 Mode:

•In its protected mode of operation, 80386DX provides a virtual 8086 operating environment to execute the 8086 programs.
•The real mode can also used to execute the 8086 programs along with the capabilities of 80386, like protection and a few additional instructions.
•Once the 80386 enters the protected mode from the 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 8086 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.
•Like 80386 real mode, the addresses in virtual 8086 mode lie within 1Mbytes of memory.
•In virtual mode, the paging mechanism and protection capabilities are available at the service of the programmers.
•The 80386 supports multiprogramming, hence more than one programmer may be use the CPU at a time.



Memory Management In Virtual 8086

•Paging unit may not be necessarily enable in virtual mode, but may be needed to run the 8086 programs which require more than 1Mbyts of memory for memory management function.
•In virtual mode, the paging unit allows only 256 pages, each of 4Kbytes size.
•Each of the pages may be located anywhere in the maximum 4Gbytes physical memory. The virtual mode allows the multiprogramming of 8086 applications.
•The virtual 8086 mode executes all the programs at privilege level 3.Any of the other programmes may deny access to the virtual mode programs or data.

•However, the real mode programs are executed at the highest privilege level, i.e. level 0.
•The virtual mode may be entered using an IRET instruction at CPL=0 or a task switch at any CPL, executing any task whose TSS is having a flag image with VM flag set to 1.
•The IRET instruction may be used to set the VM flag and consequently enter the virtual mode.
•The PUSHF and POPF instructions are unable to read or set the VM bit, as they do not access it.
•Even in the virtual mode, all the interrupts and exceptions are handled by the protected mode interrupt handler.
•To return to the protected mode from the virtual mode, any interrupt or execution may be used.
•As a part of interrupt service routine, the VM bit may be reset to zero to pull back the 80386 into protected mode.

UNIT – IV
ADVANCED MICROPROCESSORS

Enhanced Instruction Set of 80386:

The instruction set of 80386 contains all the instructions supported by·80286. The 80286 instructions are designed to operate with 8-bit or 16-bit data, while the same mnemonics for 80386 instruction set may be executed over 32-bit operands, besides 8-bit and 16-bit operands. The newly added instructions may be categorized into the following functional groups.

1. Bit scan instructions
2. Bit test instructions
3. Conditional set byte instructions
4. Shift double instructions
5. Control transfer via gates instructions.

1. Bit Scan Instructions: 80386 instruction set has two bit scan mnemonics, such as BSF (bit scan forward) and BSR (bit scan reverse). Both of these instructions scan the operand for a '1' bit, without actually rotating it. The BSF instruction scans the operand from right to left. If a '1' is encountered during the scan, zero flag is set and the bit position of '1' is stored into the destination operand. If no '1' is encountered, zero flag is reset. The BSR instruction also performs the same function but scans the source operand from the left most bit towards right.

2. Bit Test Instructions: 80386 have four bit test instructions, those are BT (test a bit), BTC (test a bit and complement), BTR (test and reset a bit) and BTS (test and set a bit). All these instructions test a bit position in the destination operand, specified by the source operand. If the bit position of the destination operand specified by the source operand satisfies the condition specified in the mnemonic, the carry flag is affected appropriately. For example, in the case of BT instruction, if the bit position in the destination operand, specified by the source operand, is '1', the carry flag is set, otherwise, it is cleared.

3. Conditional Set Byte instruction: This instruction sets all the operand bits, if the condition specified by the mnemonic is true. This instruction group has 16 mnemonics corresponding to 16 conditions as shown below.

E.g. SETO EAX; this instruction sets all the bits of EAX, if the overflow flag is set.

1. SETO Set on overflow
2. SETNO Set on no overflow
3. SETB/SETNAE Set on below/not above or equal
4. SETNB/SETAE Set on not below/above or equal
5. SETE/SETZ Set on equal/zero
6. SETNE/SETNZ Set on not equal/not zero
7. SETBE/SETNA Set on below or equal/not above
8. SETNBE/SETA Set on not below or equal/above
9. SETS Set on sign
10. SETNS Set on not sign
11. SETP/SETPE Set on parity/parity even
12. SETNP Set on not parity/parity odd
13. SETUSETNGE Set on less/not greater or equal
14. SETNUSET GE Set on not less/greater or equal
15. SETLE/SETNG Set on less or equal/not greater
16. SETNLE/SETG Set on not less or equal/greater

4. Shift Double Instructions: These instructions shift the specified number of bits from the source operand into the destination operand. The 80386 instruction set has two mnemonics under this category, such as SHLD (shift left double) and SHRD (shift right double). The SHLD instruction shifts the specified number of bits (in the instruction) from the upper side, i.e. MSB of the source

operand into the lower side, i.e. LSB of the destination operand. The SHRD instruction shifts the number of bits specified in the instruction from the lower side, i.e. LSB of the source operand into the upper side, .i.e. MSB of the destination operand.

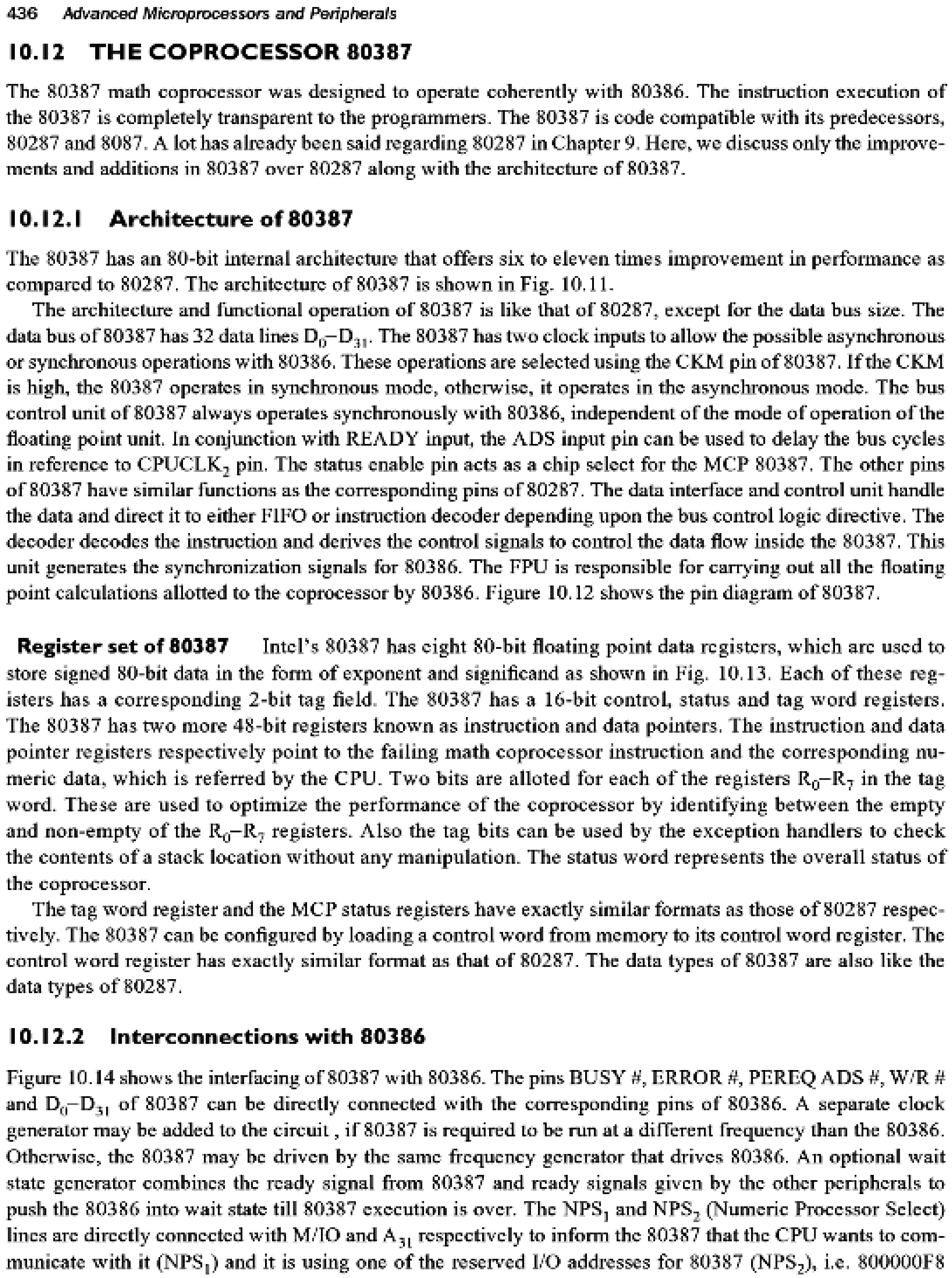
Ex: 1. SHLQ EAX, ECX, 5; This instruction shifts 5 MSB bits of ECX into the LSB positions of EAX one by one starting from the MSB of ECX. 2. SHRD EAX, ECX, 8; this instruction shifts 8 LSB bits of ECX into the MSB positions of EAX one by one starting from the LSB of ECX.

5. Control Transfer Instructions: The 80386 instruction set does not have any additional instructions for the intrasegment jump. However, for intersegment jumps, it has got a set of new instructions which are variations of the previous CALL and JUMP instructions, and are to be executed only in the protected mode. These instructions are used by 80386 to transfer the control either at the same privilege or at a different privilege level. Also, different versions of control transfer instructions are available to switch between the different task types and TSS (task state segment). The corresponding RET instructions are also available to switch back from the new task initiated via CALL, JMP or INT instructions to the parent task. Intel's 80387 has eight 80-bit floating point data registers, which are used to store signed 80-bit data in the form of exponent and significant. Each of these registers has a corresponding 2-bit tag field. The 80387 has a 16-bit control, status and tag word registers. The 80387 has two more 48-bit registers called as instruction and data pointers. The instruction and data pointer registers respectively point to the failing math coprocessor instruction and the corresponding numeric data, which is referred by the CPU. Two bits are allotted for each of the registers R0-R7 in the tag word. Also the tag bits can be used by the exception handlers to check the contents of a stack location without any manipulation. The status word represents the overall status of the coprocessor.

The 80387 can be configured by loading a control word from memory to its control word register. The control word register has exactly similar format as that of 80287.

80387 co - processor:

The 80387 has an 80-bit internal architecture that offers six to eleven times improvement in performance as compared to 80287. The architecture of 80387. The architecture and functional operation of 80387 is exactly similar to the 80287, except for the data bus size. The data bus of 80387 has 32 data lines D0-D31. The 80387 has two clock inputs to allow the possible asynchronous or synchronous operations with 80386. These operations are selected using the CKM pin of 80387. If CKM is high, the 80387 operates in synchronous mode, otherwise, it operates in asynchronous mode. The bus control unit of 80387 always operates synchronously with 80386, independent of the mode of operation of the floating point unit. In conjunction with READY input, the ADS input pin can be used to delay the bus cycles in reference to CPUCLK2 pin. The status enable pin acts as a chip select for the MCP 80387. The other pins of 80387 have similar functions as the corresponding pins of 80287. The data interface and control unit handle the data and direct into either FIFO or instruction decoder depending upon the bus control logic directive. The decoder decodes the instruction and derives the control signals to control the data flow inside the 80387. This unit generates the synchronization signals for 80386. The FPU is responsible for carrying out all the floating point calculations allotted to the coprocessor by 80387.



10.12 THE COPROCESSOR 80387

The 80387 math coprocessor was designed to operate coherently with 80386. The instruction execution of the 80387 is completely transparent to the programmers. The 80387 is code compatible with its predecessors, 80287 and 8087. A lot has already been said regarding 80287 in Chapter 9. Here, we discuss only the improvements and additions in 80387 over 80287 along with the architecture of 80387.

10.12.1 Architecture of 80387

The 80387 has an 80-bit internal architecture that offers six to eleven times improvement in performance as compared to 80287. The architecture of 80387 is shown in Fig. 10.11.

The architecture and functional operation of 80387 is like that of 80287, except for the data bus size. The data bus of 80387 has 32 data lines D_0–D_{31}. The 80387 has two clock inputs to allow the possible asynchronous or synchronous operations with 80386. These operations are selected using the CKM pin of 80387. If the CKM is high, the 80387 operates in synchronous mode, otherwise, it operates in the asynchronous mode. The bus control unit of 80387 always operates synchronously with 80386, independent of the mode of operation of the floating point unit. In conjunction with READY input, the ADS input pin can be used to delay the bus cycles in reference to $CPUCLK_2$ pin. The status enable pin acts as a chip select for the MCP 80387. The other pins of 80387 have similar functions as the corresponding pins of 80287. The data interface and control unit handle the data and direct it to either FIFO or instruction decoder depending upon the bus control logic directive. The decoder decodes the instruction and derives the control signals to control the data flow inside the 80387. This unit generates the synchronization signals for 80386. The FPU is responsible for carrying out all the floating point calculations allotted to the coprocessor by 80386. Figure 10.12 shows the pin diagram of 80387.

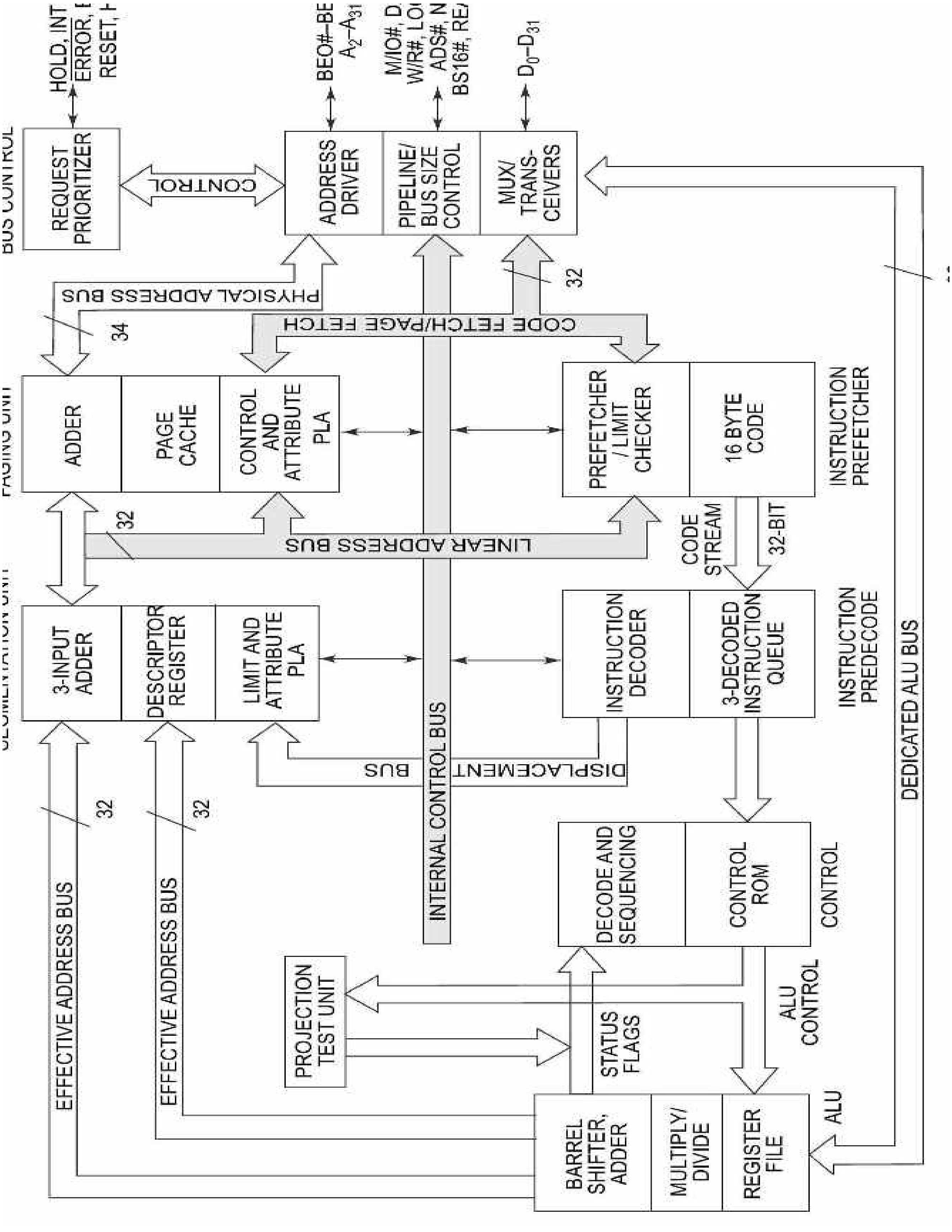
Register set of 80387 Intel's 80387 has eight 80-bit floating point data registers, which are used to store signed 80-bit data in the form of exponent and significand as shown in Fig. 10.13. Each of these registers has a corresponding 2-bit tag field. The 80387 has a 16-bit control, status and tag word registers. The 80387 has two more 48-bit registers known as instruction and data pointers. The instruction and data pointer registers respectively point to the failing math coprocessor instruction and the corresponding numeric data, which is referred by the CPU. Two bits are alloted for each of the registers R_0–R_7 in the tag word. These are used to optimize the performance of the coprocessor by identifying between the empty and non-empty of the R_0–R_7 registers. Also the tag bits can be used by the exception handlers to check the contents of a stack location without any manipulation. The status word represents the overall status of the coprocessor.

The tag word register and the MCP status registers have exactly similar formats as those of 80287 respectively. The 80387 can be configured by loading a control word from memory to its control word register. The control word register has exactly similar format as that of 80287. The data types of 80387 are also like the data types of 80287.

10.12.2 Interconnections with 80386

Figure 10.14 shows the interfacing of 80387 with 80386. The pins BUSY #, ERROR #, PEREQ ADS #, W/R # and D_0–D_{31} of 80387 can be directly connected with the corresponding pins of 80386. A separate clock generator may be added to the circuit , if 80387 is required to be run at a different frequency than the 80386. Otherwise, the 80387 may be driven by the same frequency generator that drives 80386. An optional wait state generator combines the ready signal from 80387 and ready signals given by the other peripherals to push the 80386 into wait state till 80387 execution is over. The NPS_1 and NPS_2 (Numeric Processor Select) lines are directly connected with M/IO and A_{31} respectively to inform the 80387 that the CPU wants to communicate with it (NPS_1) and it is using one of the reserved I/O addresses for 80387 (NPS_2), i.e. 800000F8

80387 ARCHITECTURE:


BUS CONTROL
REQUEST PRIORITIZER
HOLD, INT
ERROR, B
RESET, H
CONTROL
ADDRESS DRIVER
PIPELINE/ BUS SIZE CONTROL
MUX/ TRANS- CEIVERS
BEO#–BE
A2–A31
M/IO#, D
W/R#, LO
ADS#, N
BS16#, RE
D0–D31
PHYSICAL ADDRESS BUS
34
32
CODE FETCH/PAGE FETCH
PAGING UNIT
ADDER
PAGE CACHE
CONTROL AND ATTRIBUTE PLA
PREFETCHER / LIMIT CHECKER
16 BYTE CODE
INSTRUCTION PREFETCHER
32
LINEAR ADDRESS BUS
CODE STREAM
32-BIT
SEGMENTATION UNIT
3-INPUT ADDER
DESCRIPTOR REGISTER
LIMIT AND ATTRIBUTE PLA
INSTRUCTION DECODER
3-DECODED INSTRUCTION QUEUE
INSTRUCTION PREDECODE
DISPLACEMENT BUS
INTERNAL CONTROL BUS
DEDICATED ALU BUS
32
32
EFFECTIVE ADDRESS BUS
EFFECTIVE ADDRESS BUS
DECODE AND SEQUENCING
CONTROL ROM
CONTROL
PROJECTION TEST UNIT
STATUS FLAGS
ALU CONTROL
BARREL SHIFTER, ADDER
MULTIPLY/ DIVIDE
REGISTER FILE
ALU