The 80386 Microprocessor Family

Introduction

- The 80386 family of microprocessors of Intel Corporation is the first 32 bit version of the 8086 family-a switch from 16 bit to 32 bit
- 80386 has upward compatibility with 8086,8088,80286 etc
- The 80386 was launched in October 1985, but full-function chips were first delivered in the third quarter of 1986
- Memory management section of 80386 supports the virtual memory, paging and four levels of protection.

Versions of 80386

80386DX – the full version

The first member in 80386 family

this CPU could work with 16-bit and 32-bit external buses.

Comprises of both 32-bit internal registers and 32-bit external bus.

80386SX –the reduced bus version

low cost version of the 80386.

This processor had 16 bit external data bus and 24-bit external address bus.

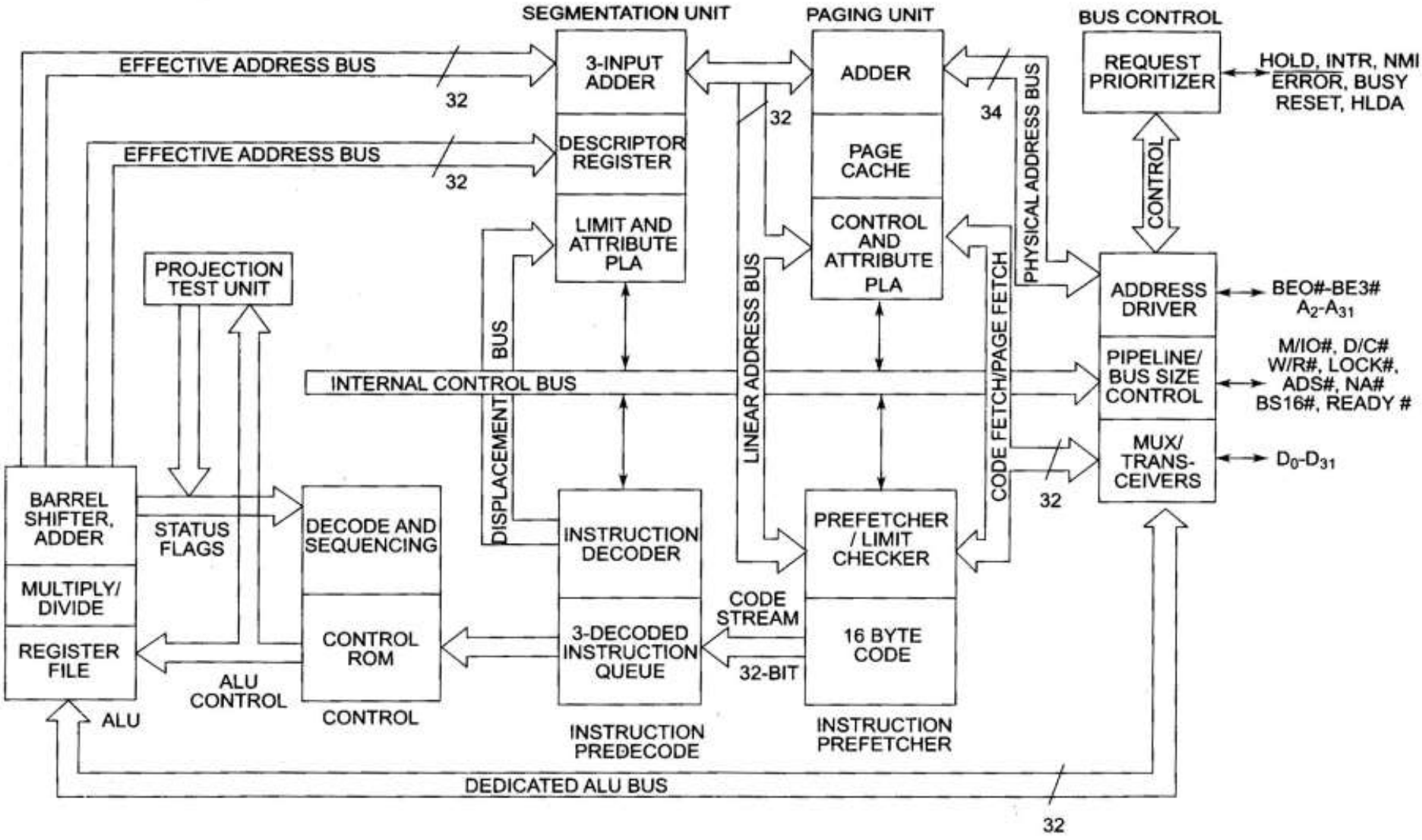
Features of 80386DX

- The 80386DX is a 32-bit processor that supports 8-bit/16-bit/32-bit data operands.
- The instruction set is upward compatible with all its predecessors.
- With its 32-bit address bus, it can address up to 4Gbytes of physical memory. The physical memory is organized in terms of segments of 4 Gbytes size at maximum.
- The 80386 CPU supports 16K (16384) number of segments and thus the total virtual memory space is 4Gbytes×16K=64 terrabytes.
- The concept of paging is introduced in 80386 that enables it to organize the available physical memory into pages of size 4Kbytes each, under the segmented memory.

Features of 80386DX

- The 80386 can be supported by 80387 for mathematical data processing.
- The 80386DX supports 8 debug registers, for hardware debugging and control.
- The 80386DX has an on chip address translation cache.
- Another version 80386SX has identical architecture but 16-bit data and 24-bit address bus.
- The 80386DX is available in 132-pin grid array package and has 20MHz and 33MHz versions.

Architecture of 80386 microprocessor

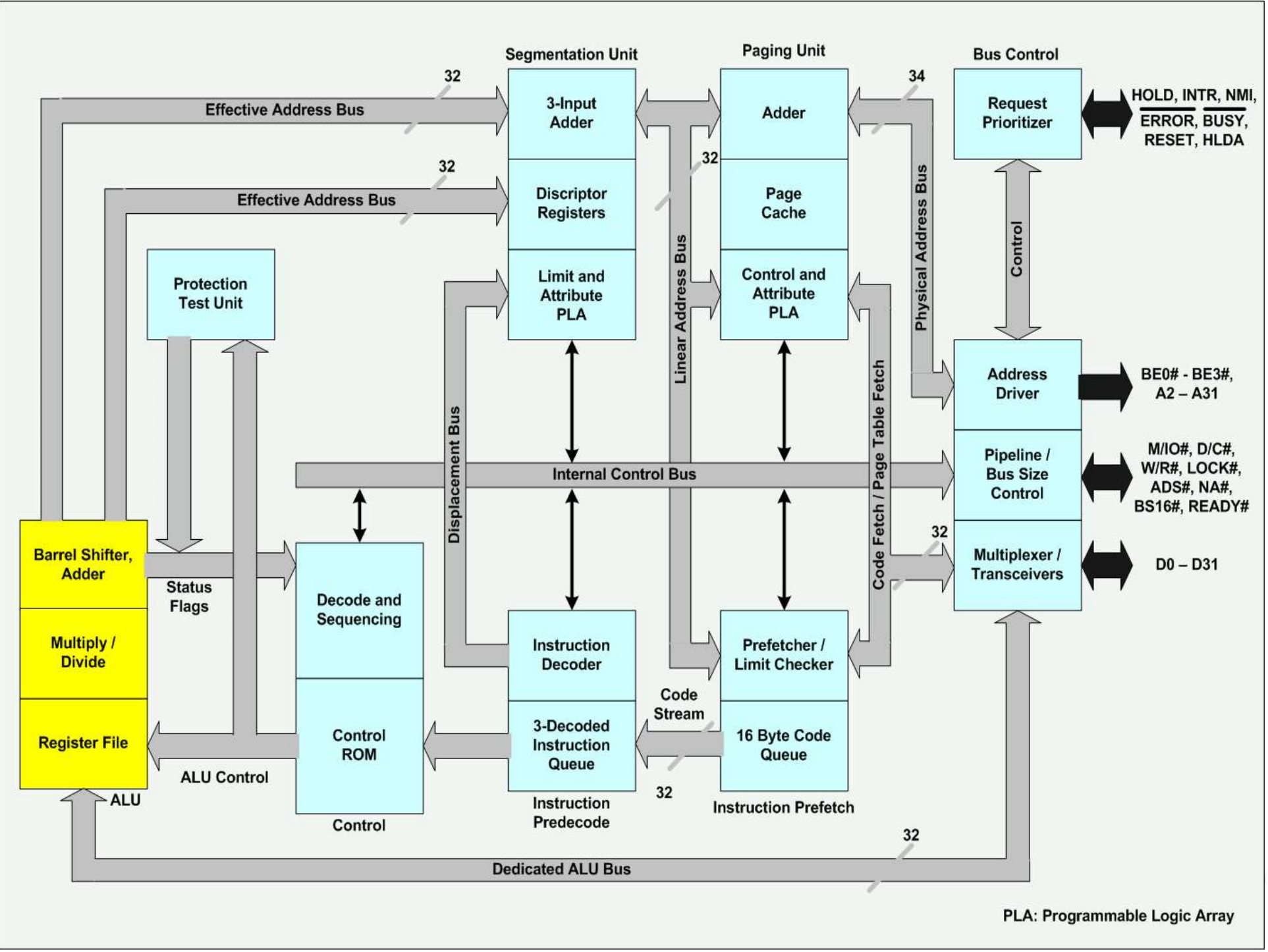


80386 Architecture

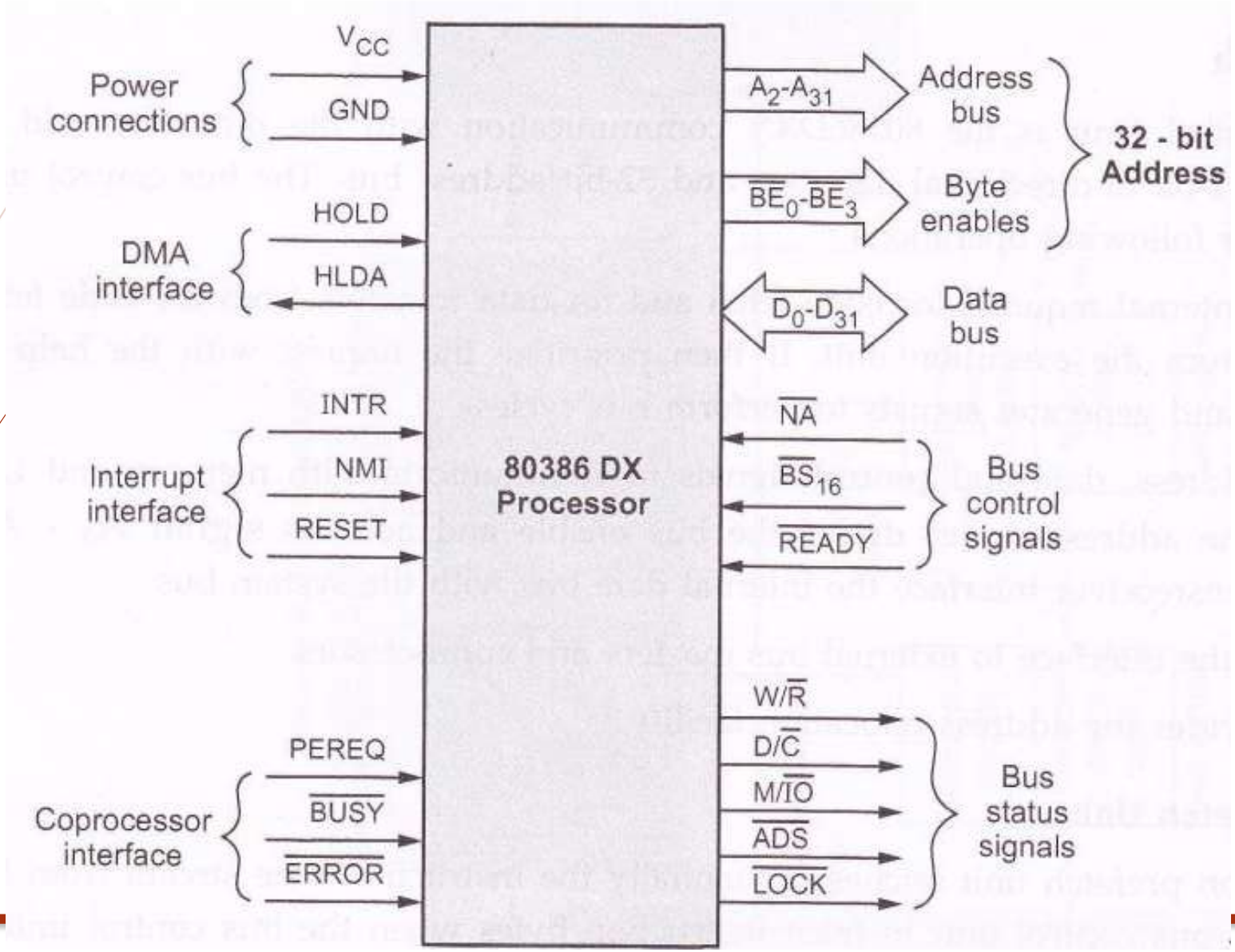
Architecture of 80386 microprocessor

The Internal Architecture of 80386 is divided into 3 sections.

- Central processing unit
  - Prefetcher and prefetch queue
  - Instruction decoder
  - Control ROM and sequencing
  - Execution unit
  - Protection unit
- Bus interface unit
- Memory management unit
  - Segmentation unit
  - Paging unit


Segmentation Unit
Paging Unit
Bus Control
32
Effective Address Bus
3-Input Adder
Adder
34
Request Prioritizer
HOLD, INTR, NMI,
ERROR, BUSY,
RESET, HLDA
32
Effective Address Bus
Discriptor Registers
32
Page Cache
Physical Address Bus
Control
Protection Test Unit
Limit and Attribute PLA
Linear Address Bus
Control and Attribute PLA
Address Driver
BE0# - BE3#,
A2 – A31
Displacement Bus
Internal Control Bus
Code Fetch / Page Table Fetch
Pipeline / Bus Size Control
M/IO#, D/C#,
W/R#, LOCK#,
ADS#, NA#,
BS16#, READY#
32
Barrel Shifter, Adder
Status Flags
Decode and Sequencing
Instruction Decoder
Prefetcher / Limit Checker
Multiplexer / Transceivers
D0 – D31
Multiply / Divide
Code Stream
Register File
Control ROM
3-Decoded Instruction Queue
16 Byte Code Queue
ALU Control
ALU
32
Instruction Predecode
Instruction Prefetch
Control
32
Dedicated ALU Bus
PLA: Programmable Logic Array

PIN DIAGRAM


Power connections
V_{CC}
GND
DMA interface
HOLD
HLDA
Interrupt interface
INTR
NMI
RESET
Coprocessor interface
PEREQ
BUSY
ERROR
80386 DX Processor
A_2-A_{31}
Address bus
$\overline{BE_0}$-$\overline{BE_3}$
Byte enables
32 - bit Address
D_0-D_{31}
Data bus
NA
$\overline{BS_{16}}$
READY
Bus control signals
W/R
D/C
M/IO
ADS
LOCK
Bus status signals

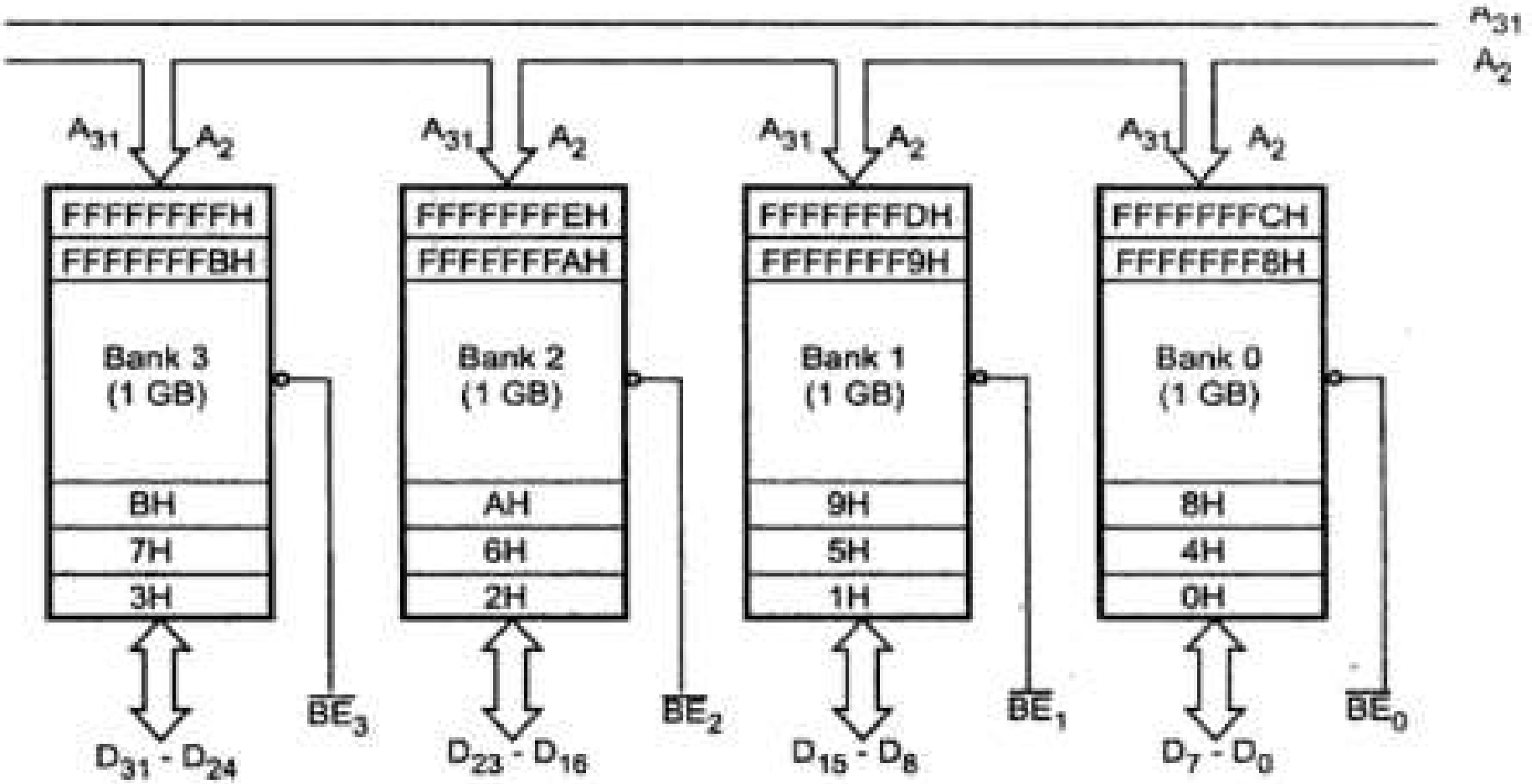
PIN DIAGRAM

- CLK2 The input pin provides the basic system clock timing for the operation of 80386
- D31-D0 - These 32 lines act as bidirectional data bus during different access cycles
- A31-A2 - ADDRESS BUS outputs physical memory or port I/O addresses.
- BE0-BE3 – These signals are generated by A0 and A1.
  - BYTE ENABLES indicate which data bytes of the data bus take part in a bus cycle. 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 BE0 toBE3, may be used for enabling these 4 banks. Using these 4 enable signal lines, the CPU may transfer 1 byte / 2 / 3 / 4 byte of data simultaneously

Signal Interface

- **Byte Enable Outputs(BE0# -- BE3#)**

BE3#	BE2#	BE1#	BE0#	Operation
1	1	1	1	No Operation
1	1	1	0	Bank0 (8-bit)
1	1	0	1	Bank1 (8-bit)
1	0	1	1	Bank2 (8-bit)
0	1	1	1	Bank3 (8-bit)
1	1	0	0	Bank 0,1 (16-bit)
1	0	0	1	Bank 1,2 (16-bit)
0	0	1	1	Bank 2,3 (16-bit)
0	0	0	0	Bank 0,1,2,3 (32-bit)


A31
A2
A31
A2
A31
A2
A31
A2
A31
A2
FFFFFFFFH
FFFFFFFBH
Bank 3
(1 GB)
BH
7H
3H
FFFFFFFEH
FFFFFFFAH
Bank 2
(1 GB)
AH
6H
2H
FFFFFFFDH
FFFFFFF9H
Bank 1
(1 GB)
9H
5H
1H
FFFFFFFCH
FFFFFFF8H
Bank 0
(1 GB)
8H
4H
0H
BE3
BE2
BE1
BE0
D31 - D24
D23 - D16
D15 - D8
D7 - D0

PIN DIAGRAM

- $\overline{W}/R$ - WRITE/READ is a bus cycle definition pin that distinguishes write cycles from read cycles. (1=W, 0=R)
- $D/\overline{C}$ - DATA/CONTROL is a bus cycle definition pin that distinguishes data cycles, either memory or I/O, from control cycles which are: interrupt acknowledge, halt, and instruction fetching. (1=D, 0=C)
- $M/\overline{IO}$ - MEMORY I/O is a bus cycle definition pin that distinguishes memory cycles from input/output cycles. (1=M, 0=I/O)

PIN DIAGRAM

- $\overline{\text{LOCK}}$ - BUS LOCK is a bus cycle definition pin that enables the CPU to prevent the other bus masters from gaining the control of the system bus.

- $\overline{\text{ADS}}$ - ADDRESS STATUS indicates that a valid bus cycle definition and address are being driven at the Intel386 DX pins.

- $\overline{\text{NA}}$ - NEXT ADDRESS is used to request address pipelining.

PIN DIAGRAM

- $\overline{\text{READY}}$ – The ready signals indicates to the CPU that the previous bus cycle has been terminated and the bus is ready for the next cycle.
- $\overline{\text{BS16}}$ – (Dynamic data bus sizing) If this pin is initialized with 0, 80386 uses 16-bit data bus and if it is initialized with 1, 80386 uses 32-bit data bus.
- HOLD - HOLD REQUEST input allows another bus master to request control of the local bus.
- HLDA - HOLD ACKNOWLEDGE output indicates that the Intel386 DX has surrendered control of its local bus to another bus master.

PIN DIAGRAM

- $\overline{\text{BUSY}}$ - signals a busy condition from a processor extension(co-processor).The busy input signal indicates to the CPU that the coprocessor is busy with the allocated task.

- $\overline{\text{ERROR}}$ - The error input pin indicates to the CPU that the coprocessor has encountered an error while executing its instruction.

- PEREQ - PROCESSOR EXTENSION REQUEST. The processor extension request output signal indicates to the CPU to fetch a data word for the coprocessor

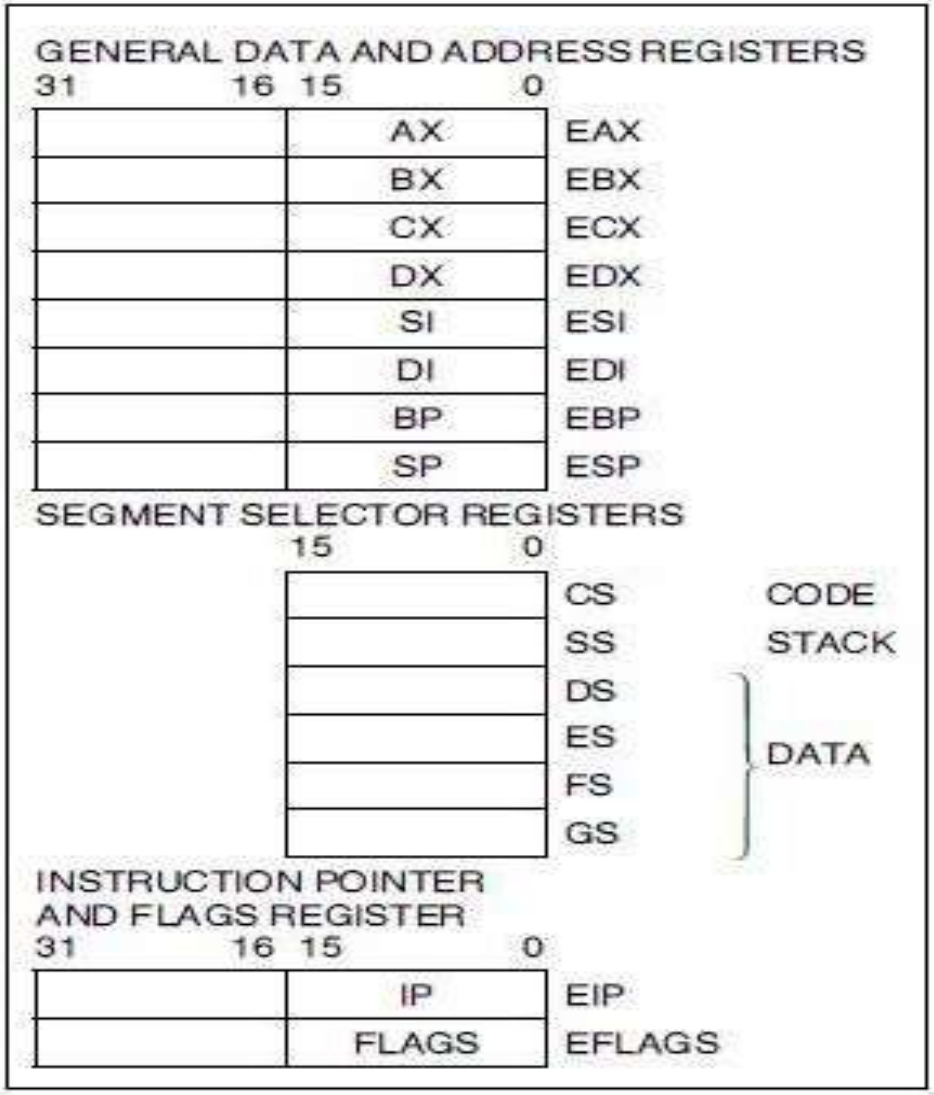
PIN DIAGRAM

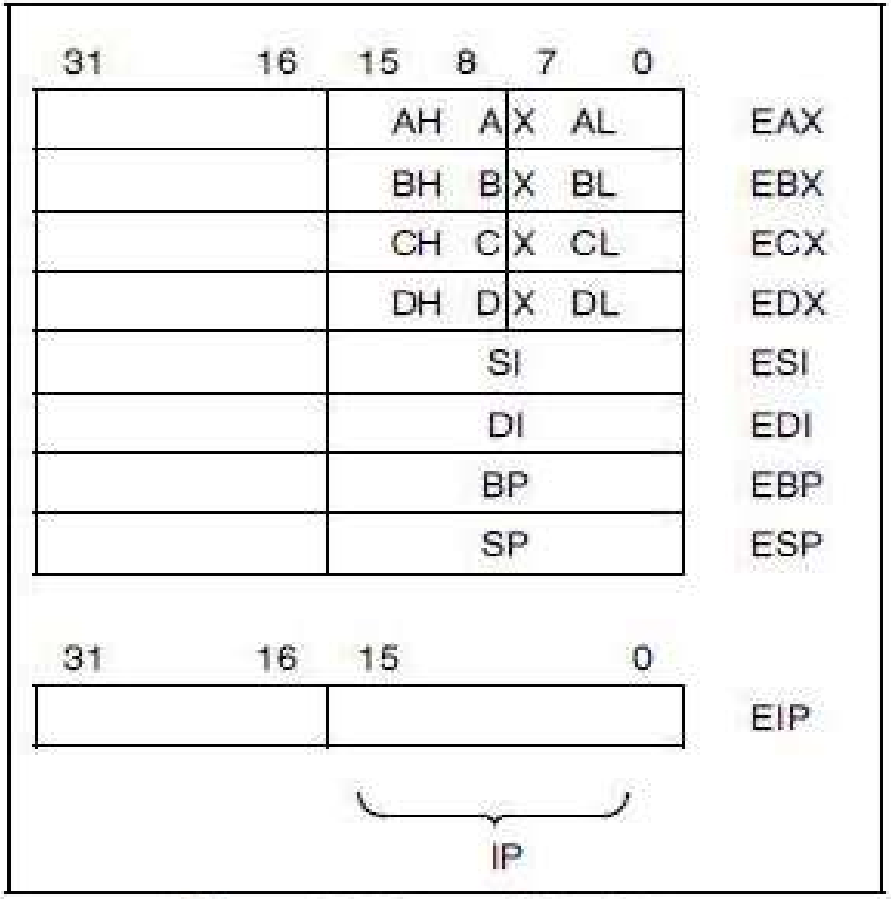
- INTR - INTERRUPT REQUEST is a maskable input that signals the Intel386 DX to suspend execution of the current program and execute an interrupt acknowledge function.
- NMI - a non-maskable input that signals the Intel386 DX to suspend execution of the current program and execute an interrupt acknowledge function.
- RESET - suspends any operation in progress and places the Intel386 DX in a known reset state.

PROGRAMMING MODEL

- Register Organization
  - **General Purpose Registers**
  - **Segment Registers**
  - **Instruction Pointer and Flags**
  - **System Address Registers**
  - **Control Registers**
  - Debug Registers
  - Test Registers.

General Purpose Registers

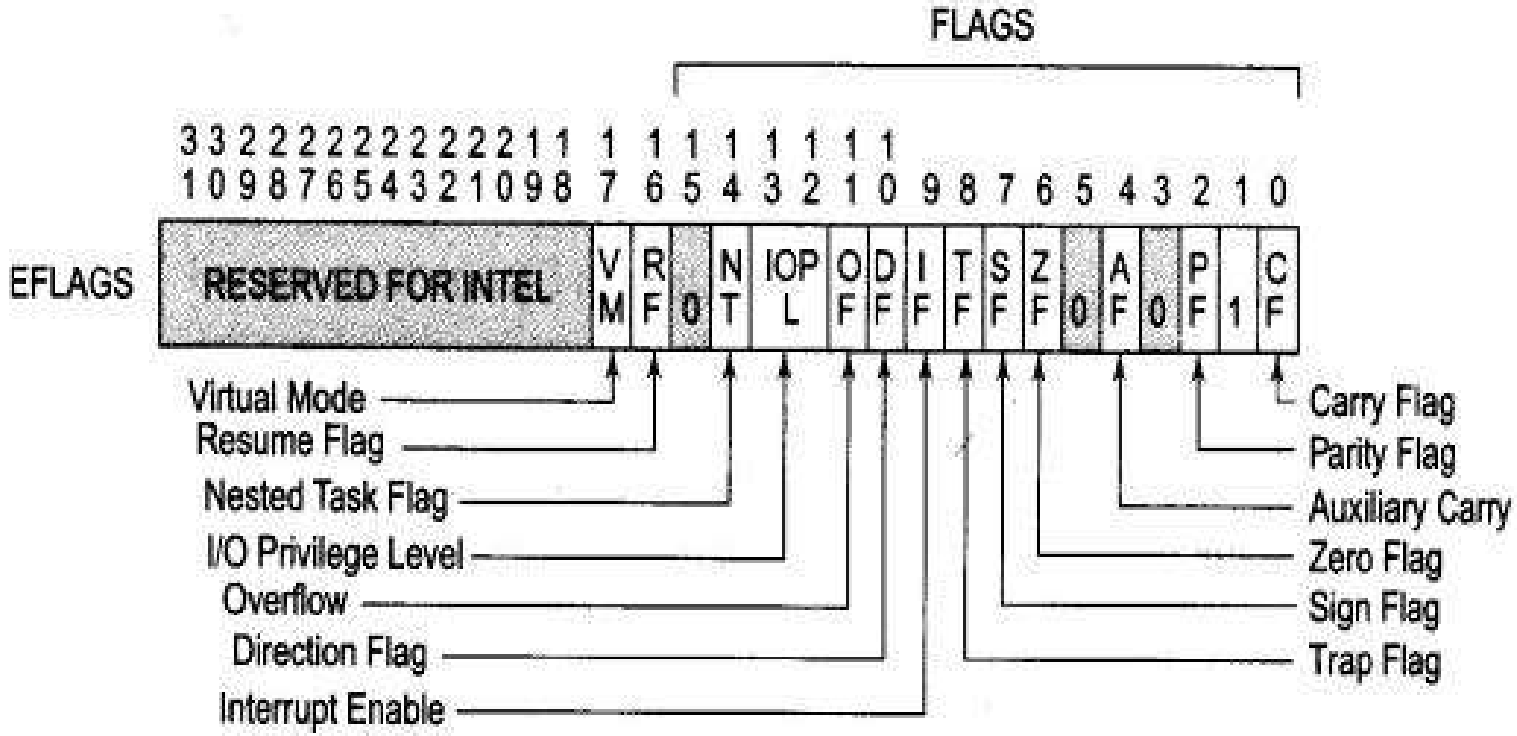




Flags Register

- The Flags Register is a 32-bit register named EFLAGS.
- The defined bits and bit fields within EFLAGS, control certain operations and indicate status of the Intel386 DX.
- The lower 16 bits of EFLAGS contain the 16-bit flag register named FLAGS, which is most useful when executing 8086 and 80286 code

EFlags Register



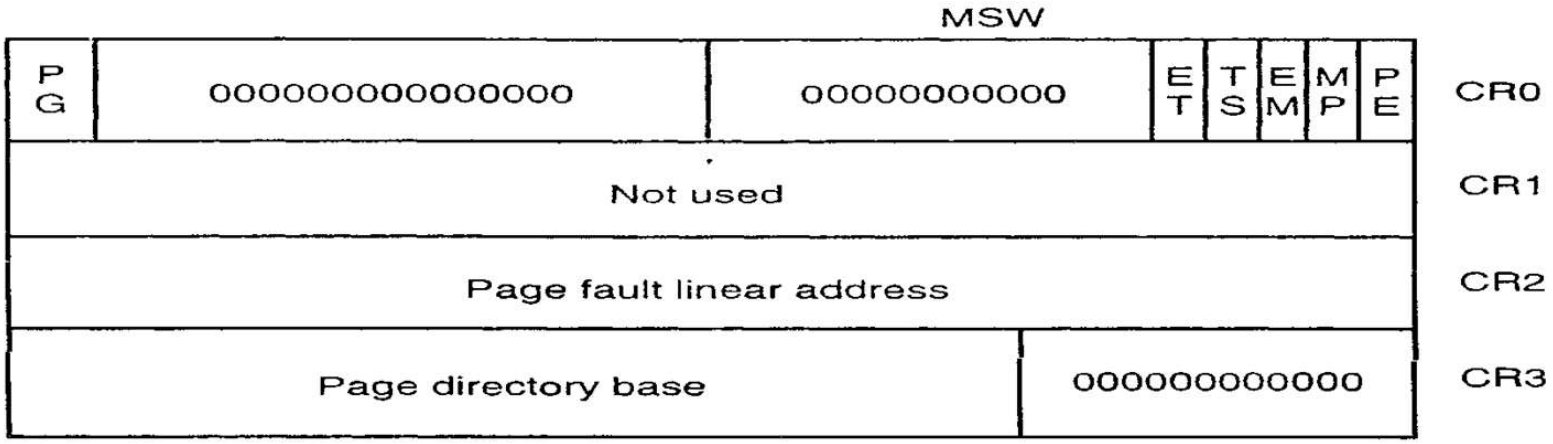
Note: 0 indicates Intel reserved

Flag Register of 80386 (Intel Corp.)

System Address Registers/Descriptor tables

- There are four new registers in the protected mode model:
  - Global Descriptor Table Register(GDTR)
  - Interrupt Descriptor Table Register(IDTR)
  - Local Descriptor Table Register(LDTR)
  - Task State Segment Descriptor Register (TR)

CR0



Register CR0 contains a number of special control bits that are defined as follows in the 80386:

PG Selects page table translation of linear addresses into physical addresses when PG = 1. Page table translation allows any linear address to be assigned any physical memory location.

ET Selects the 80287 coprocessor when ET = 0 or the 80387 coprocessor when ET = 1. This bit was installed because there was no 80387 available when the 80386 first appeared. In most systems, ET is set to indicate that an 80387 is present in the system.

TS Indicates that the 80386 has switched tasks (in protected mode, changing the contents of TR places a 1 into TS). If TS = 1, a numeric coprocessor instruction causes a type 7 (coprocessor not available) interrupt.

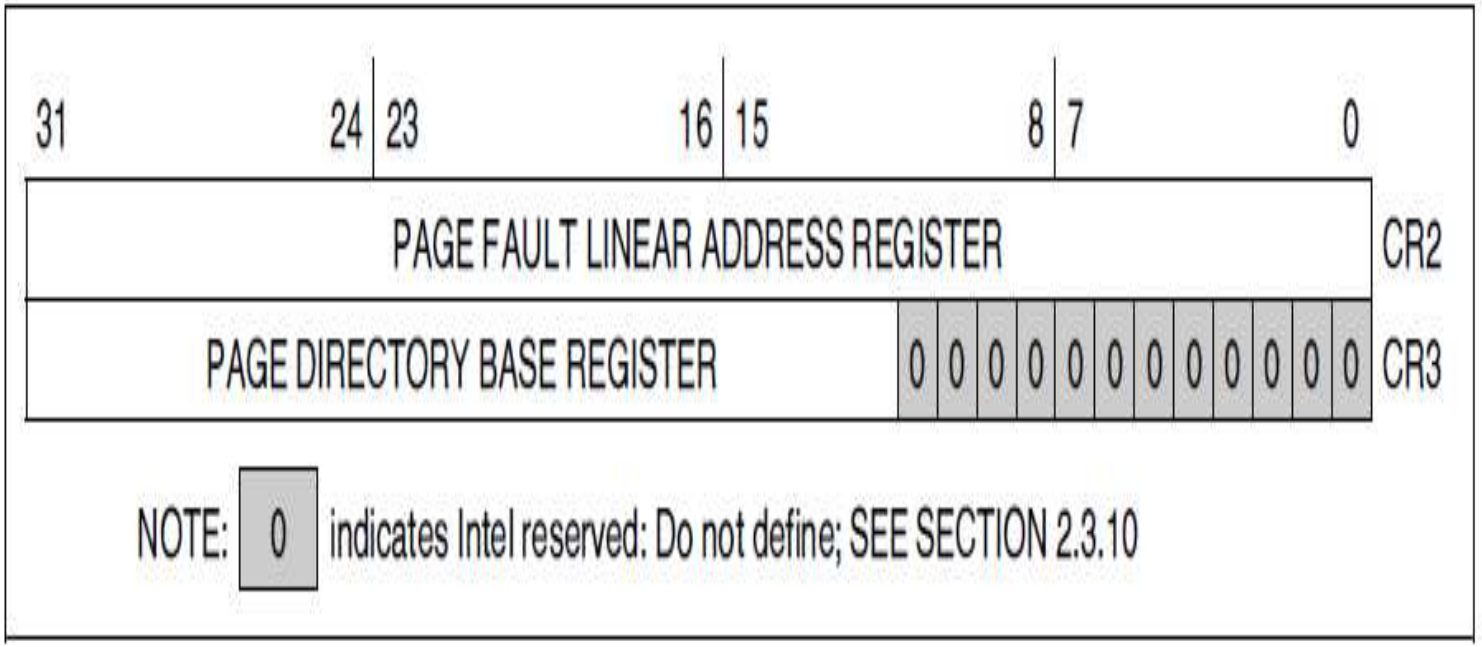
EM Is set to cause a type 7 interrupt for each ESC instruction. (ESCape instructions are used to encode instructions for the 80387 coprocessor.) We often use this interrupt to emulate, with software, the function of the coprocessor. Emulation reduces the system cost, but it often takes at least 100 times longer to execute the emulated coprocessor instructions.

MP Is set to indicate that the arithmetic coprocessor is present in the system.

PE Is set to select the protected mode of operation for the 80386. It may also be cleared to reenter the real mode. This bit can only be set in the 80286. The 80286 could not return to real mode without a hardware reset, which precludes its use in most systems that use protected mode.

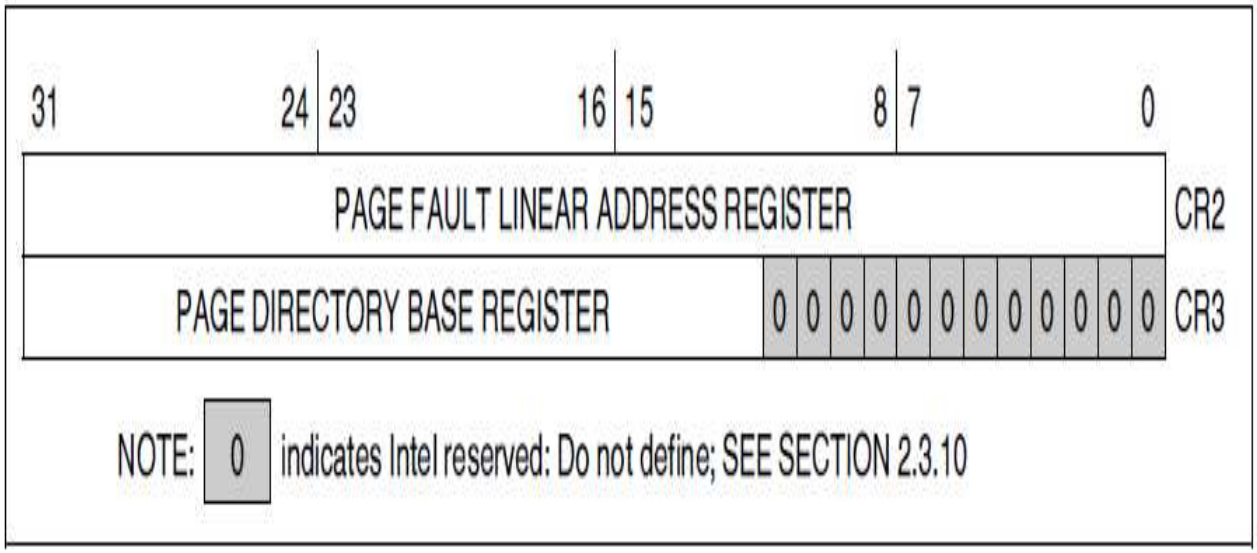
CR1 and CR2

- CR1: reserved
  - CR1 is reserved for use in future Intel processors.
- CR2: Page Fault Linear Address
  - holds the 32-bit linear address that caused the last page fault detected.



CR3: Page Directory Base Address

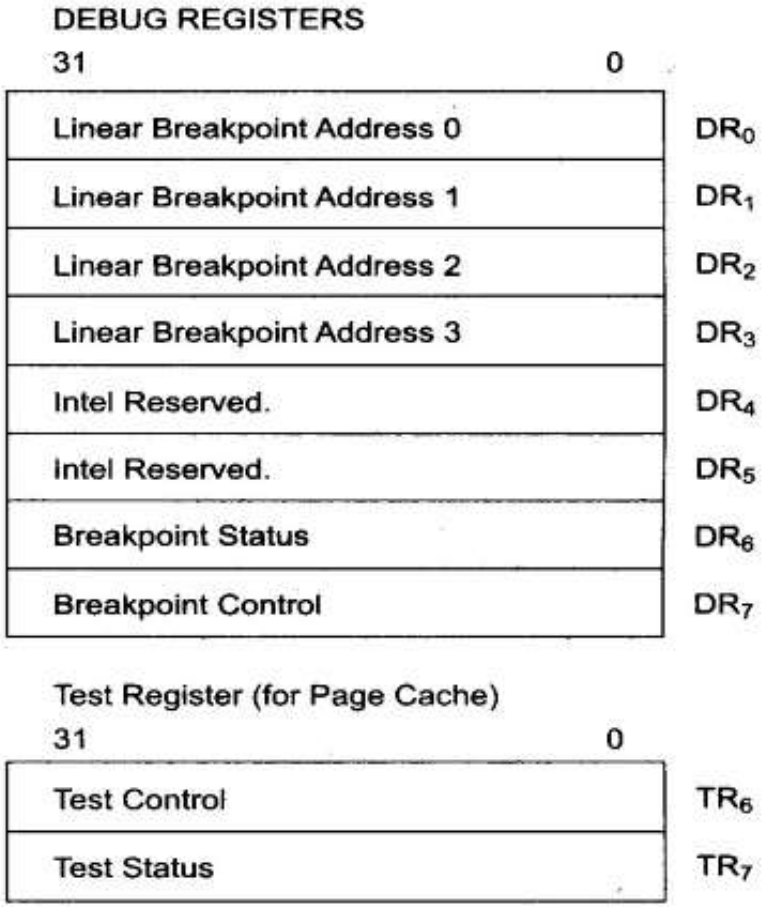
- CR3 contains the physical base address of the page directory table.
- The Intel386 DX page directory table is always page aligned (4 Kbyte-aligned). Therefore the lowest twelve bits of CR3 are ignored when written and they store as undefined.
- It gives upper 20 bits of the starting address of the page directory



Debug and Test Registers

- Debug Registers
  - 8 debug registers for hardware debugging DR0 – DR7
  - DR4, DR5 – Intel reserved
  - DR0- DR3 – store four program controllable breakpoint addresses
  - DR6, DR7 – hold breakpoint status, breakpoint control information
- Test Registers
  - Test control
  - Test status registers

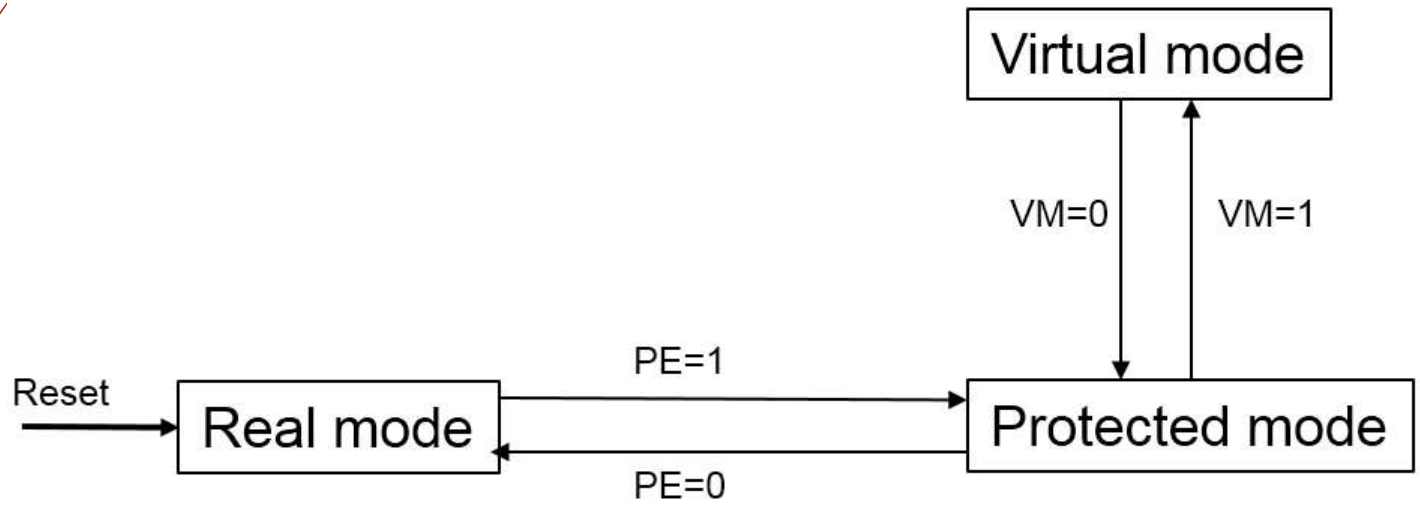
Debug Registers



Debug and Test Registers of 80386 (Intel Corp.)

Processing modes of 80386

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

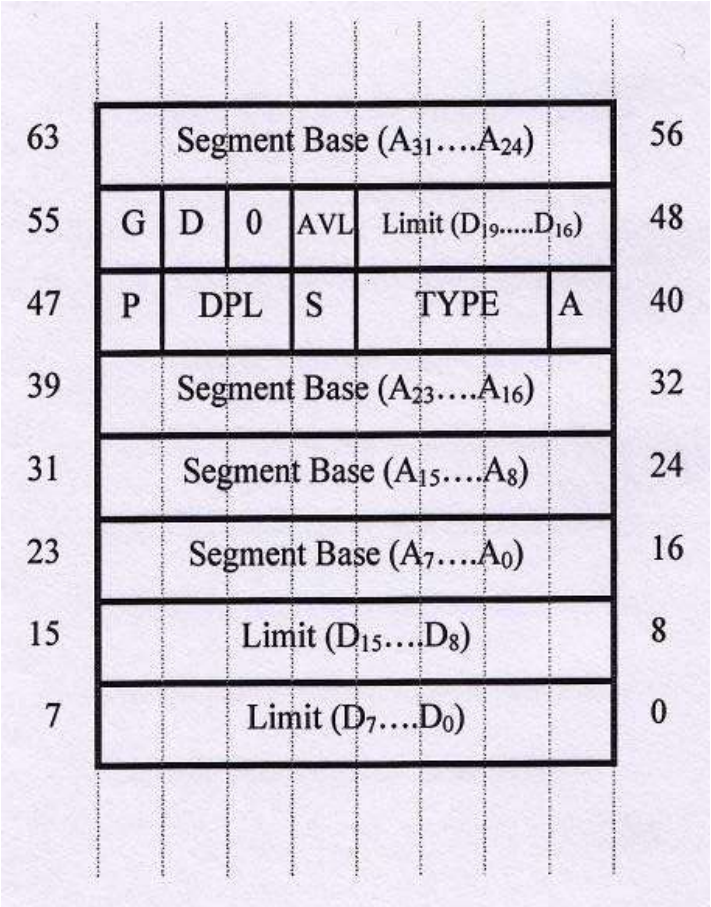


Segmentation

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

Segmentation

- Descriptors
  - The 80386 descriptors are 8 byte quantities containing 20 bit segment limit, 32 bit segment address and 16 bits for access rights.



Segmentation

- Descriptors

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

Structure of an 80386 Descriptor (Intel Corp.)

- Type Bits
  - E, ED/C, RW
  - E = 0 (data or stack) ,
    - ED = 0 Expand Up (data segment)
    - ED =1 Expand down (stack Segment)
    - W = 0 read only , 1 may ne written into
  - E =1 (code seg)
    - C = 1 Conforming
    - R = 1 may be read

Segmentation

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

Paging

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

Paging

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

Paging

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

Paging

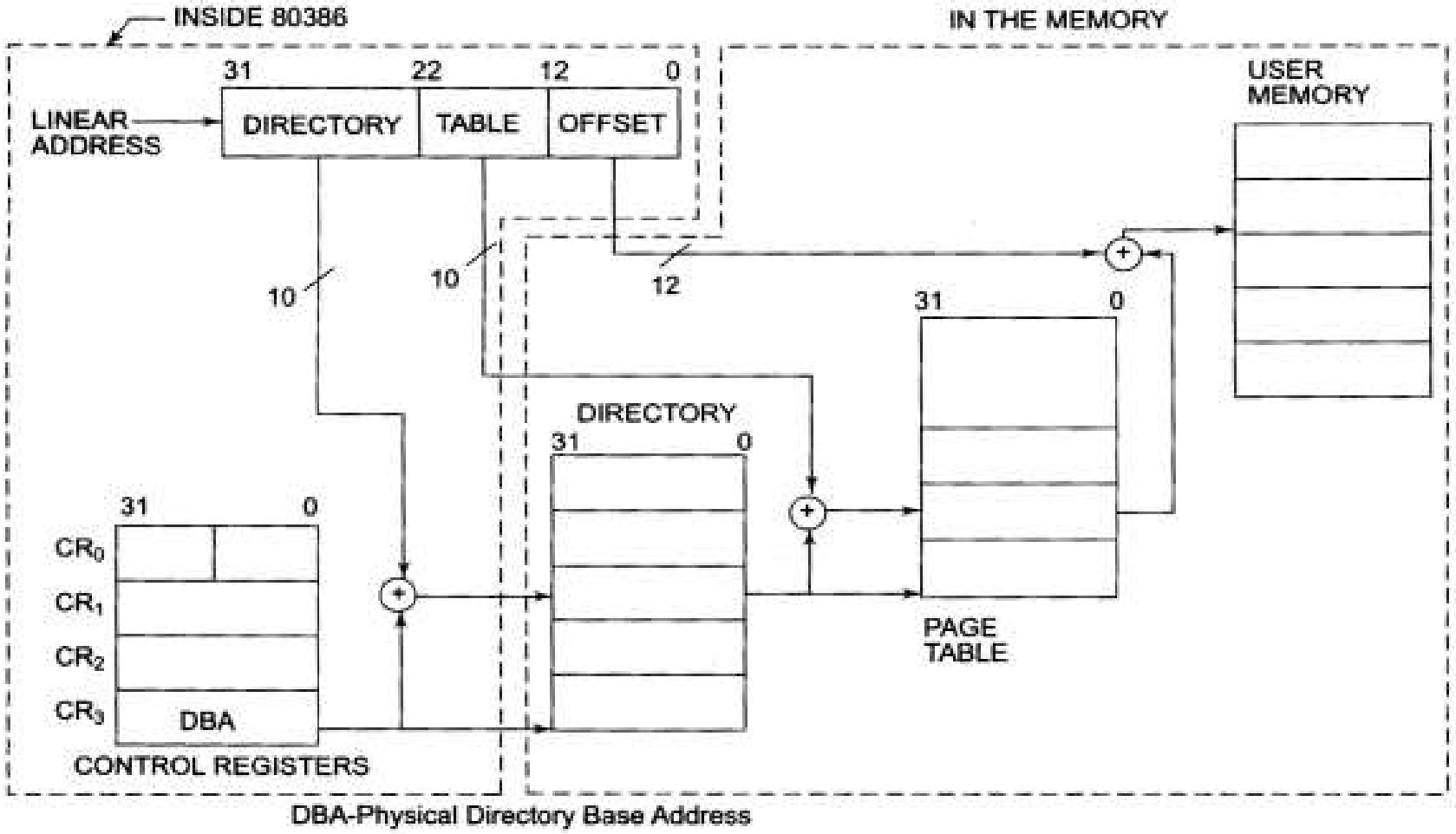
- Page Directory Base Register
  - The CR3 is used as a page directory base address register, to store starting address of the page directory.
- Page Directory
  - This is at most 4Kbytes in size.
  - Each directory entry is of 4 bytes, thus total of 1024 entries are allowed in a directory.
  - The upper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

Paging

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

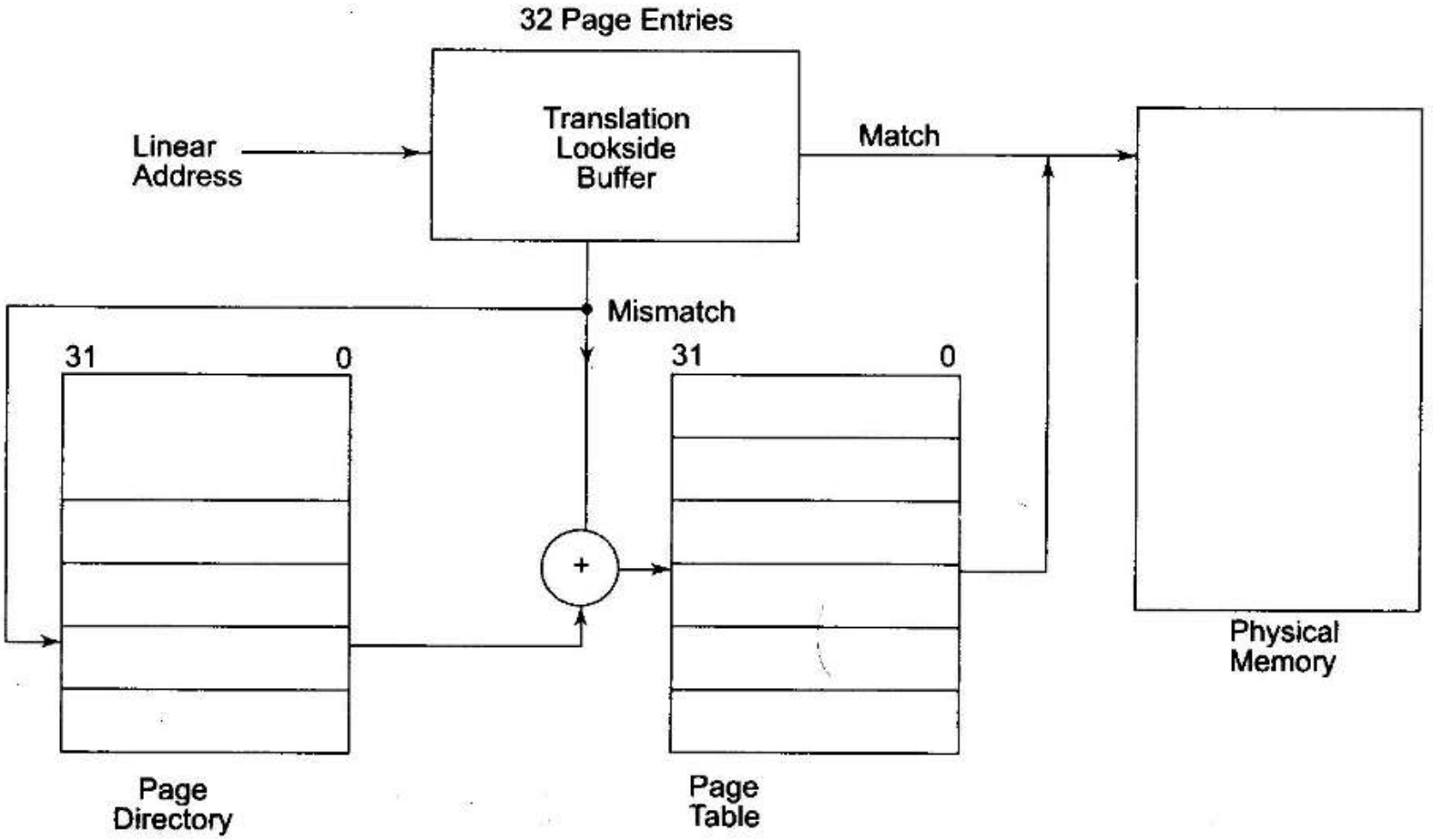
Paging

Block diagrammatic representation of complete paging mechanism of 80386



Paging Mechanism of 80386 (Intel Corp.)

Conversion of linear address to a physical address



Paging Operation with TLB (Intel Corp.)