1.6 INTEL 8086 MICROPROCESSOR

The INTEL 8086 is the first 16-bit processor released by INTEL in the year 1978. The 8086 is designed using the HMOS technology and now it is manufactured using HMOS III technology and contains approximately 29,000 transistors. The 8086 is packed in a 40-pin DIP and requires a single 5-V supply.

The 8086 does not have an internal clock circuit. The 8086 requires an external asymmetric clock source with 33% duty cycle. An 8284 clock generator is used to generate the required clock for 8086. The maximum internal clock of 8086 is 5 MHz. The other versions of 8086 with different clock rates are 8086-1, 8086-2 and 8086-4 with maximum internal clock frequency of 10 MHz,

8 MHz and 4 MHz respectively.

The 8086 uses a 20-bit address to access memory and hence it can directly address up to one megabytes (220 = 1 Mega) of memory space. The one megabyte (1MB) of addressable memory space of 8086 are organized as two memory banks of 512 kilobytes each (512 kB + 512 kB = 1MB).

The memory banks are called even (or lower) bank and odd (or upper) bank. The address line A_0 is used to select even bank and the control signal $\overline{BHE}$ is used to select odd bank.

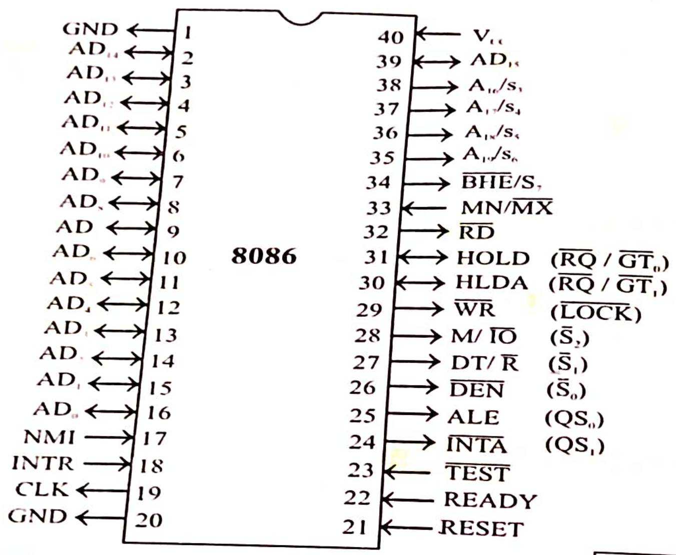
For accessing IO-mapped devices, the 8086 uses a separate 16-bit address, and so the 8086 can generate 64k (2^{16}) IO addresses. The signal M/$\overline{IO}$ is used to differentiate the memory and IO addresses. For memory address the signal M/$\overline{IO}$ is asserted **high** and for IO address the signal M/$\overline{IO}$ is asserted **low** by the processor.

The 8086 can operate in two modes: minimum mode and maximum mode. The mode is decided by a signal at MN/$\overline{MX}$ pin. When the MN/$\overline{MX}$ is tied **high** it works in minimum mode and the system is called a uniprocessor system. When MN/$\overline{MX}$ is tied **low** it works in maximum mode and the system is called a multiprocessor system. Usually the pin MN/$\overline{MX}$ is permanently tied to **low** or **high** so that the 8086 system can work in any one of the two modes. The 8086 can work with an 8087 coprocessor in maximum mode. In this mode an external bus controller 8288 is required to generate bus control signals.

The 8086 has two families of processors. They are 8086 and 8088. The 8088 uses 8-bit data bus externally but 8086 uses 16-bit data bus externally. The 8086 access memory is in words but 8088 access memory is in bytes. IBM designed its first Personal Computer (PC) using an INTEL 8088 microprocessor as the CPU.

1.6.1 Pins and Signals of INTEL 8086

The 8086 pins and signals are shown in Fig. 1.13. The 8086 is a 40-pin IC and all the 8086 pins are TTL compatible. The signal assigned to pins 24 to 31 is different for minimum and maximum mode of operation. The signal assigned to all the other pins are common for minimum and maximum mode of operation.



8086
GND 1, AD14 2, AD13 3, AD12 4, AD11 5, AD10 6, AD9 7, AD8 8, AD7 9, AD6 10, AD5 11, AD4 12, AD3 13, AD2 14, AD1 15, AD0 16, NMI 17, INTR 18, CLK 19, GND 20
40 Vcc, 39 AD15, 38 A16/S3, 37 A17/S4, 36 A18/S5, 35 A19/S6, 34 BHE/S7, 33 MN/MX, 32 RD
31 HOLD (RQ / GT0), 30 HLDA (RQ / GT1), 29 WR (LOCK), 28 M/IO (S2), 27 DT/R (S1), 26 DEN (S0), 25 ALE (QS0), 24 INTA (QS1)
23 TEST, 22 READY, 21 RESET

Fig. a : 8086 pin assignments.

Note: Signals shown in parenthesis are maximum mode signals.

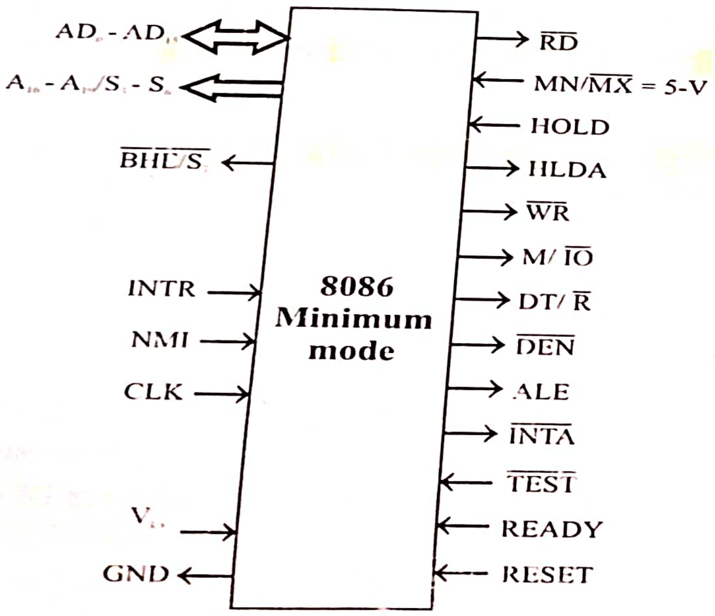


Fig. b: 8086-Minimum mode.

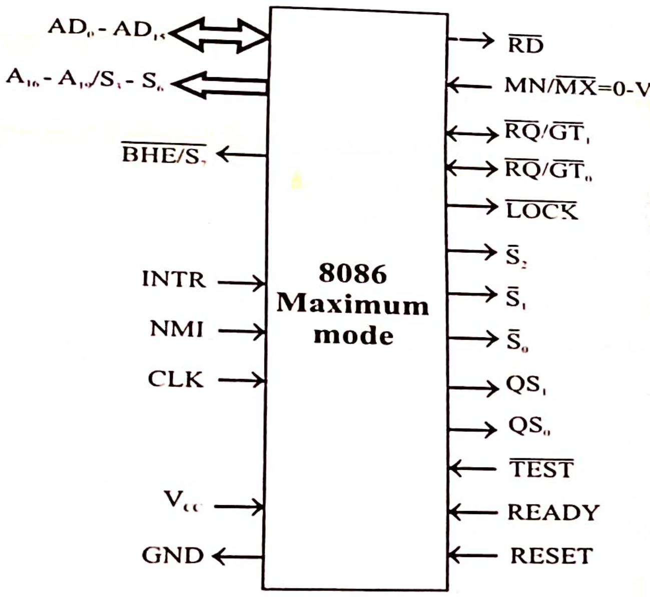


Fig. c: 8086-Maximum mode.

Fig. 1.8: 8086 signals and pin assignment.

TABLE 1.4: Common Signals

Name	Description/Function	Type
AD_{15} - AD_0	Address/Data	Bidirectional, Tristate
A_{19}/S_6-A_{16}/S_3	Address/Status	Output, Tristate
$\overline{BHE}/S_7$	Bus high enable/Status	Output, Tristate
$MN/\overline{MX}$	Minimum/Maximum mode control	Input
$\overline{RD}$	Read control	Output, Tristate
$\overline{TEST}$	Wait on test control	Input
READY	Wait state control	Input
RESET	System reset	Input
NMI	Non-maskable interrupt request	Input
INTR	Interrupt request	Input
CLK	System clock	Input
V_{cc}	+ 5-V	Power supply input
GND	Ground	Power supply ground

TABLE 1.5: Minimum Mode Signals [MN / $\overline{MX}$ = V_{CC} (Logic high)]

Name	Description / Function	Type
HOLD	Hold request	Input
HLDA	Hold acknowledge	Output
$\overline{WR}$	Write control	Output, Tristate
$M/\overline{IO}$	Memory/IO control	Output, Tristate
$DT/\overline{R}$	Data transmit/Receive	Output, Tristate
$\overline{DEN}$	Data enable	Output, Tristate
ALE	Address latch enable	Output
$\overline{INTA}$	Interrupt acknowledge	Output

TABLE 1.6: Maximum Mode Signals [MN / $\overline{MX}$ = GROUND (Logic low)]

Name	Description/Function	Type
$\overline{RQ}/\overline{GT}_1$, $\overline{RQ}/\overline{GT}_0$	Request/Grant bus access control	Bidirectional
$\overline{LOCK}$	Bus priority lock control	Output, Tristate
$\overline{S}_2$, $\overline{S}_1$, $\overline{S}_0$	Bus cycle status	Output, Tristate
QS_1, QS_0	Instruction queue status	Output

1.6.2 Common Signals

The signals common for minimum and maximum mode are listed in Table-1.4. The lower sixteen lines of address are multiplexed with data and the upper four lines of address are multiplexed with status signals. During the first clock period of a bus cycle the entire 20-bit address is available on these lines. During all other clock periods of a bus cycle, the data and status signals will be available on these lines.

The status signals on S_3 and S_4 specifies the segment register used for calculating the physical address. The output on the status lines S_3 and S_4 when the processor is accessing various segments are listed in Table 1.7.

Table 1.7: Status Signals During Memory Segment Access

Status signal		Segment register
S_4	S_3	
0	0	Extra segment
0	1	Stack segment
1	0	Code or no segment
1	1	Data segment

The status lines S_3 and S_4 can be used to expand the memory up to 4 Mb. The status line S_5 indicates the status of an 8086 interrupt enable flag. A **low** on the line S_6 indicates that 8086 is on the bus (i.e., it indicates that 8086 is the bus master) and during hold acknowledge this pin is driven to **high impedance** state. The output signal $\overline{\text{BHE}}$ on the first T-state of a bus cycle is maintained as status signal S_7 on the same pin.

The 8086 outputs a **low** on $\overline{\text{BHE}}$ pin during read, write and interrupt acknowledge cycles when the data is to be transferred to the high-order data bus. The $\overline{\text{BHE}}$ can be used in conjunction with AD_0 to select memory banks.

When the processor reads from memory or an IO location it asserts $\overline{\text{RD}}$ **low**. The $\overline{\text{TEST}}$ input is tested by the WAIT instruction. The 8086 will enter a wait state after execution of the WAIT instruction, and it will resume execution only when $\overline{\text{TEST}}$ is made **low** by an external hardware. This is used to synchronize an external activity to the processor internal operation. $\overline{\text{TEST}}$ input is synchronized internally during each clock cycle on the leading edge of the clock signal.

INTR is the maskable interrupt and INTR must be held **high** until it is recognized to generate an interrupt signal. NMI is the non-maskable interrupt input activated by a leading edge signal.

RESET is the system reset input signal. For power-ON reset it is held **high** for 50 micro-second For reset while working, it is held **high** for at least four clock cycles. When the processor is resetted the DS, SS, ES, IP and flag register are cleared, Code Segment (CS) register is initialized to $FFFF_H$ and queue is emptied. After reset the processor will start fetching instruction from 20-bit physical address $FFFF0_H$.

READY is an input signal to the processor, used by the memory or IO devices to get extra time for data transfer or to introduce **wait states** in the bus cycles. Normally READY is tied **high**. If the READY is tied **low**, the 8086 introduces wait states after second T-state of a bus cycle and it will complete the bus cycle only when READY is made **high** again.

CLK input is the clock signal that provides basic timing for the 8086 and bus controller. The 8086 does not have an on-chip clock generation circuit. Hence the 8284 clock generator chip is used to generate the required clock. A quartz crystal whose frequency is thrice that of the internal clock of 8086 must be connected to the 8284. The 8284 generates the clock at crystal frequency. It divides the generated clock by three and modifies the duty cycle to 33% and output on the CLK pin of the 8284. This CLK output of the 8284 must be connected to the 8086 CLK pin. The 8284 also provides the RESET and READY signal to an 8086.

1.6.3 Minimum Mode Signals

The minimum mode signals of an 8086 are listed in Table 1.5. For minimum mode of operation the MN/$\overline{\text{MX}}$ pin is tied to V_{cc}(logic **high**). In minimum mode, the 8086 itself generates all bus control signals. The minimum mode signals are explained below:

DT/$\overline{\text{R}}$ - [*Data Transmit / Receive*] It is an output signal from the processor to control the direction of data flow through the data transceivers.

DEN - (*Data Enable*) - It is an output signal from the processor used as output enable for the data transceivers.

ALE - (*Address Latch Enable*) - It is used to demultiplex the address and data lines using external latches.

M/$\overline{\text{IO}}$ - It is used to differentiate memory access and IO access. For IN and OUT instructions it is asserted **low**. For memory reference instructions it is asserted **high.**

$\overline{\text{WR}}$ - It is a write control signal and it is asserted **low** whenever the processor writes data to memory or IO port.

$\overline{\text{INTA}}$ - (*Interrupt Acknowledge*) - The 8086 output is asserted **low** on this line to acknowledge when the interrupt request is accepted by the processor.

HOLD - It is an input signal to the processor from other bus masters as a request to grant control of the bus. It is usually used by the DMA controller to get control of the bus.

HLDA - (*Hold Acknowledge*) - It is an acknowledge signal by the processor to the master requesting the control of the bus through HOLD. The acknowledge is asserted **high** when the processor accepts the HOLD. [*On accepting the hold the processor drives all the tristate pins to **high impedance** state and sends an acknowledgement to the device which requested HOLD. On receiving the acknowledgement the other master will take control of the bus.*]

1.6.4 Maximum Mode Signals

The maximum mode signals of an 8086 are listed in Table 1.6. An 8086-based system can be made to work in maximum mode by grounding the MN/$\overline{\text{MX}}$ pin (i.e., MN/$\overline{\text{MX}}$ is tied to logic **low**). In maximum mode, the pins 24 to 31 are redefined as follows:

$\overline{S_0}$, $\overline{S_1}$, $\overline{S_2}$ - These are status signals and they are used by the 8288 bus controller to generate bus timing and control signals. The status signals are decoded as shown in Table 1.8.

Table 1.8: Status Signals During Various Machine Cycles

Status Signal			Machine Cycle
S_2	S_1	S_0	
0	0	0	Interrupt acknowledge
0	0	1	Read IO port
0	1	0	Write IO port
0	1	1	Halt
1	0	0	Code access
1	0	1	Read memory
1	1	0	Write memory
1	1	1	Passive/Inactive

$RQ/\overline{GT_0}$, $RQ/\overline{GT_1}$ - (Bus Request/Bus Grant) These requests are used by the other local bus masters to force the processor to release the local bus at the end of the processor's current bus cycle These pins are bidirectional. The request on GT_0will have higher priority than GT_1.

The bus request to an 8086 works as follows:

1. **When a local bus master requires an system bus control, it sends a low pulse to the 8086.**
2. **At the end of the current bus cycle, the processor (8086) drives its pins to high impedance state and sends an acknowledgement as a low pulse on the same pin to the device which had requested the bus control.**
3. **On receiving the acknowledgement the local master will take control of the system bus. After completing its work, at the end, the local bus master sends a low signal on the same pin to 8086 to inform the end of control. Now 8086 regains the control of the bus.**

$\overline{\text{LOCK}}$ - It an output signal activated by the LOCK prefix instruction and remains active until the completion of the instruction prefixed by LOCK. The 8086 asserts the $\overline{\text{LOCK}}$ pin low while executing an instruction prefixed by LOCK to prevent other bus masters from gaining control of the system bus.

QS_1, QS_0 - (Queue Status) - The processor provides the status of queue on these lines. The queue status can be used by the external device to track the internal status of the queue in an 8086. The QS_0 and QS_1 are valid during the clock period following any queue operation. The output on QS_0 and QS_1 can be interpretted as shown in Table 1.9.

TABLE 1.9: QUEUE STATUS

Queue status		Queue operation
QS_1	QS_0	
0	0	No operation
0	1	First byte of an opcode from the queue
1	0	Empty the queue
1	1	Subsequent byte from the queue

1.6.5 Architecture of INTEL 8086

The 8086 has a pipelined architecture. In pipelined architecture the processor will have a number of functional units and the execution time of the functional units are overlapped. Each functional unit works independently most of the time. The simplified block diagram of the internal architecture of an 8086 is shown in Fig. 1.9. The architecture of the 8086 can be internally divided into two separate functional units: **Bus Interface Unit** (BIU) and **Execution Unit** (EU).

The BIU fetches instructions, reads data from memory and IO ports and writes data to memory and IO ports. The BIU contains segment registers, an instruction pointer, an instruction queue, an address generation unit and a bus control unit. The EU executes instructions that have already been fetched by the BIU. The BIU and EU function independently.

The instruction queue is a FIFO (**F**irst-**I**n-**F**irst-**O**ut) group of registers. The size of the queue is 6 bytes. The BIU fetches the instruction code from memory and stores it in queue. The EU fetches the instruction codes from the queue.

The BIU has four 16-bit segment registers: **C**ode **S**egment (CS) register, **D**ata **S**egment (DS) register, **S**tack **S**egment (SS) register and **E**xtra **S**egment (ES) register. The 8086 memory space can be divided into segments of 64 kB. The 4-segment registers are used to hold four segment base addresses. Hence 8086 can directly address 4 segments of 64 kB at any time instant (4 × 64 = 256 kB within 1 MB memory space). This feature of the 8086 allows the system designer to allocate separate areas for storing program codes and data.

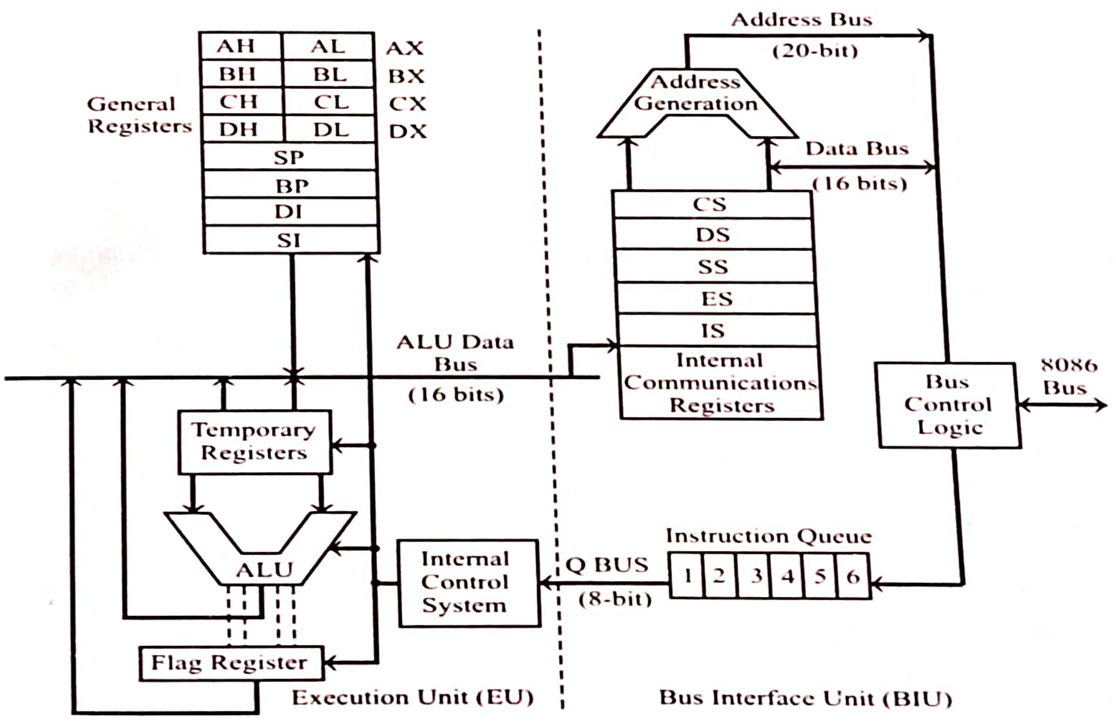


Fig. 1.9: Internal architecture of 8086.

The contents of the segment registers are programmable. Hence the processor can access the code and data in any part of the memory by changing the contents of the segment registers. The memory segment can be continuous, partially overlapped, fully overlapped or disjointed.

Note: *Since segment registers are programmable it is possible to design multitasking and multiuser systems using 8086. The program code and data for each task/user can be stored in separate segments. The program execution can be switched from one task/user to another by changing the contents of the segment registers.*

The dedicated address generation unit generates a 20-bit physical address from the segment base and an offset or effective address. The segment base address is logically shifted left four times and added to the offset. *[logically shifting left four times is equal to multiplying it by 16_{10}.]*

The address for fetching instruction codes is generated by logically shifting the content of the CS to the left four times and then adding it to the content of the IP (Instruction Pointer). The IP holds the offset address of the program codes. The content of the IP gets incremented by two after every bus cycle. *[In one bus cycle the processor fetches two bytes of the instruction code.]*

The data address is computed by using the content of the DS or ES as the base address and an offset or effective address specified by the instruction. The stack address is computed by using the content of the SS as the base address and the content of the SP (Stack Pointer) as the offset address or effective address.

The bus control logic of the BIU generates all the bus control signals such as read and write signals for memory and IO. The EU consists of the ALU, the flag register and the general purpose registers. The EU decodes and executes the instructions. A decoder in the EU control system translates the instructions.

The EU has a 16-bit ALU to perform arithmetic and logical operations. The EU has eight numbers of 16-bit general purpose registers. They are AX, BX, CX, DX, SP, BP, SI and DI.

Some of the 16-bit registers can also be used as two numbers of 8-bit registers as given below:

AX - can be used as AH and AL

BX - can be used as BH and BL

CX - can be used as CH and CL

DX - can be used as DH and DL

The general purpose registers can be used for data storage when they are not involved in any special functions assigned to them. These registers are named after special functions carried out by each one of them as given in Table 1.10.

Table 1.10: Special Functions of 8086 Registers

Register	Name of the register	Special function
AX	16-bit Accumulator	Stores the 16-bit result of certain arithmetic and logical operations.
AL	8-bit Accumulator	Stores the 8-bit result of certain arithmetic and logical operations.
BX	Base Register	Used to hold the base value in base addressing mode to access memory data
CX	Count Register	Used to hold the count value in SHIFT, ROTATE and LOOP instructions.
DX	Data Register	Used to hold data for multiplication and division operations.
SP	Stack Pointer	Used to hold the offset address of top of stack memory.
BP	Base Pointer	Used to hold the base value in base addressing using stack segment register to access data from stack memory.
SI	Source Index	Used to hold the index value of source operand (data) for string instructions.
DI	Destination Index	Used to hold the index value of destination operand (data) for string instruction.

1.6.6 8086 Flag Register

The size of an 8086 flag register is 16 bits and in this nine bits are defined as flags. The six flags are used to indicate the status of the result of the arithmetic or logical operations. Three flags are used to control the processor operation and so they are also called control bits. The various flags of an 8086 processor and their bit position in flag register are shown in Fig. 1.10.

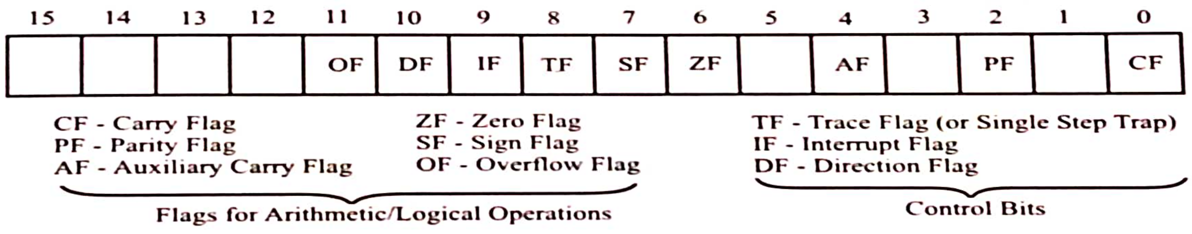


Fig. 1.10: Bit positions of various flags in the flag register of 8086.

The **Carry Flag** (CF) is set if there is a carry from the addition or borrow from the subtraction. **Auxiliary carry Flag** (AF) is set if there is a carry from low nibble to high nibble of the low order 8-bit of a 16-bit number.

The **Overflow Flag** (OF) is set to **one** if there is an arithmetic overflow, that is, if the size of the result exceeds the capacity of the destination location. Sign Flag (SF) is set to **one** if the most significant bit of the result is **one** and SF is cleared to **zero** for non-negative result. The **Parity Flag** (PF) is set to **one** if the result has even parity and PF is cleared to **zero** for odd parity of the result. The **Zero Flag** (ZF) is set to **one** if the result is zero and ZF is cleared to **zero** for a non zero result.

The three control bits in the flag register can be set or reset by the programmer. The **D**irection **F**lag (DF) is set to **one** for autodecrement and reset to **zero** for autoincrement of the SI and DI registers during string data accessing. Setting **I**nterrupt **F**lag (IF) to **one** causes the 8086 to recognize the external maskable interrupts, and clearing IF to **zero** disables the interrupts.

Setting **T**race **F**lag (TF) to **one**, places the 8086 in the single step mode. In this mode the 8086 generates an internal interrupt after execution of each instruction. The single stepping is used for debugging a program.