8086 features

- It is a 16-bit microprocessor.
- 8086 has a 20 bit address bus can access up to 2^{20} memory locations (1 MB).
- It can support up to 64K I/O ports.
- It provides 14, 16 -bit registers.
- It has multiplexed address and data bus AD0- AD15 and A16 – A19.
- It requires single phase clock with 33% duty cycle to provide internal timing.
- 8086 is designed to operate in two modes, Minimum and Maximum.
- It can prefetches upto 6 instruction bytes from memory and queues them in order to speed up instruction execution.
- It requires +5V power supply.
- A 40 pin dual in line package
- It has 256 vectored interrupts.
- It consists of 29,000 transistors.

8086 microprocessor pin out and pin functions

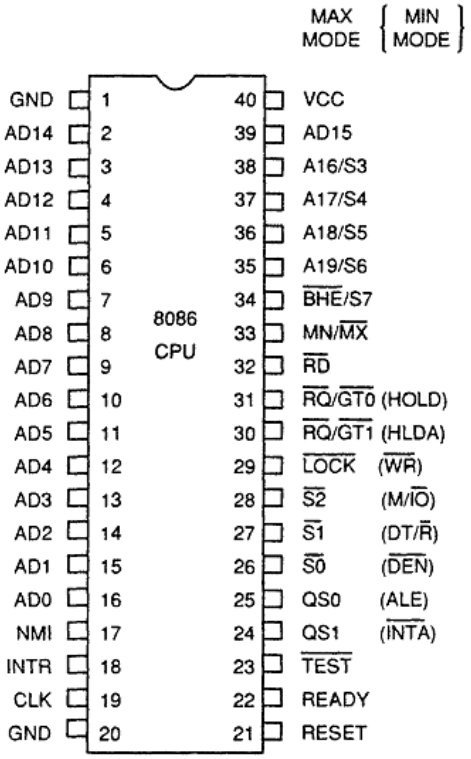


Figure 1 8086 pin diagram

AD_7–AD_0 The 8088 **address/data bus** lines compose the multiplexed address data bus of the 8088 and contain the rightmost 8-bits of the memory address or I/O port number whenever ALE is active (logic 1) or data whenever ALE is active (logic 0). These pins are at their high-impedance state during a hold acknowledge.

AD_{15}–AD_8 The 8086 **address/data bus** lines compose the upper multiplexed address/data bus on the 8086. These lines contain address bits A_{15}–A_8 whenever ALE is a logic 1, and data bus connections D_{15}–D_8. These pins enter a high-impedance state whenever a hold acknowledge occurs.

PREPARED BY
SHAHADAT HUSSAIN PARVEZ

A_{19}/S_6–A_{16}/S_3	The **address/status bus** bits are multiplexed to provide address signals A_{19}–A_{16} and also status bits S_6–S_3. These pins also attain a high-impedance state during the hold acknowledge. Status bit S_6 always remains a logic 0, bit S_5 indicates the condition of the IF flag bits, and S_4 and S_3 show which segment is accessed during the current bus cycle.

Table 1 shows the function of status bits s_4 and s_3. These status bits could be used to address four separate 1M byte memory banks by decoding them as A_{21} and A_{20}

$\overline{RD}$	Whenever the **read signal** is a logic 0, the data bus is receptive to data from the memory or I/O devices connected to the system. This pin floats to its high-impedance state during a hold acknowledge.
READY	This input is controlled to insert wait states into the timing of the microprocessor. If the READY pin is placed at a logic 0 level, the microprocessor enters into wait states and remains idle. If the READY pin is placed at a logic 1 level, it has no effect on the operation of the microprocessor.
INTR	**Interrupt request** is used to request a hardware interrupt. If INTR is held high when IF = 1, the 8086/8088 enters an interrupt acknowledge cycle (INTA becomes active) after the current instruction has completed execution.
$\overline{TEST}$	The **Test** pin is an input that is tested by the WAIT instruction. If $\overline{TEST}$ is a logic 0, the WAIT instruction functions as a NOP. If $\overline{TEST}$is a logic 1, then the WAIT instruction waits for $\overline{TEST}$ to become a logic 0. This pin is most often connected to the 8087 numeric coprocessor.

Table 1 Functions of status bits S_3 and S_4 [Applicable for minimum mode]

S_4	S_3	*Function*
0	0	Extra segment
0	1	Stack segment
1	0	Code or no segment
1	1	Data segment

Table 2 Bus control functions generated by bus controller (8288) [Applicable for maximum mode]

$\overline{S2}$	$\overline{S1}$	$\overline{SO}$	*Function*
0	0	0	Interrupt acknowledge
0	0	1	I/O read
0	1	0	I/O write
0	1	1	Halt
1	0	0	Opcode fetch
1	0	1	Memory read
1	1	0	Memory write
1	1	1	Passive

PREPARED BY
SHAHADAT HUSSAIN PARVEZ

NMI	The **non-maskable interrupt** input is similar to INTR except that the NMI interrupt does not check to see if the IF flag bit is a logic 1. If NMI is activated, this interrupt input uses interrupt vector 2.
RESET	The reset input causes the microprocessor to reset itself if this pin is held high for a minimum of four clocking periods. Whenever the 8086 or 8088 is reset, it begins executing instructions at memory location FFFF0H and disables future interrupts by clearing the IF flag bit.
CLK	The **clock pin** provides the basic timing signal to the microprocessor. The clock signal must have a duty cycle of 33% (high for one-third of the clocking period and low for two-thirds) to provide proper internal timing for the 8086/8088.
V_{CC}	This **power supply** input provides a +5.0 V, ±10 % signal to the microprocessor.
GND	The **ground** connection is the return for the power supply. Note that the 8086/8088 microprocessors have two pins labeled GND—both must be connected to ground for proper operation.
MN/$\overline{\text{MX}}$	The **minimum/maximum mode** pin selects either minimum mode or maximum mode operation for the microprocessor. If minimum mode is selected, the MN/MX pin must be connected directly to +5.0 V.
$\overline{\text{BHE}}$/S_7	The **bus high enable** pin is used in the 8086 to enable the most-significant data bus bits (D_{15}–D_8) during a read or a write operation. The state of S_7 is always a logic 1.

Minimum Mode Pins. Minimum mode operation of the 8086/8088 is obtained by connecting the MN/$\overline{\text{MX}}$ pin directly to +5.0 V. Do not connect this pin to +5.0 V through a pull-up resister or it will not function correctly.

IO/$\overline{\text{M}}$ or M/$\overline{\text{IO}}$	The IO/$\overline{\text{M}}$ (8088) or the M/$\overline{\text{IO}}$ (8086) pin selects memory or I/O. This pin indicates that the microprocessor address bus contains either a memory address or an I/O port address. This pin is at its high-impedance state during a hold acknowledge.
$\overline{\text{WR}}$	The **write line** is a strobe that indicates that the 8086/8088 is outputting data to a memory or I/O device. During the time that the $\overline{\text{WR}}$ is a logic 0, the data bus contains valid data for memory or I/O. This pin floats to a high-impedance during a hold acknowledge.
$\overline{\text{INTA}}$	The **interrupt acknowledge** signal is a response to the INTR input pin. The $\overline{\text{INTA}}$ pin is normally used to gate the interrupt vector number onto the data bus in response to an interrupt request.
ALE	**Address latch enable** shows that the 8086/8088address/data bus contains address information. This address can be a memory address or an I/O port number. Note that the ALE signal does not float during a hold acknowledge.
DT/$\overline{\text{R}}$	The **data transmit/receive** signal shows that the microprocessor data bus is transmitting (DT/$\overline{\text{R}}$ = 1) or receiving (DT/$\overline{\text{R}}$ = 0) data. This signal is used to enable external data bus buffers.
DEN	**Data bus enable** activates external data bus buffers.

PREPARED BY
SHAHADAT HUSSAIN PARVEZ

HOLD The **hold** input requests a direct memory access (DMA). If the HOLD signal is a logic 1, the microprocessor stops executing software and places its address, data, and control bus at the high-impedance state. If the HOLD pin is a logic 0, the microprocessor executes software normally.

HLDA **Hold acknowledge** indicates that the 8086/8088 microprocessors have entered the hold state.

Maximum Mode Pins. In order to achieve maximum mode for use with external coprocessors, connect the MN/$\overline{MX}$ pin to ground.

$\overline{S2}$, $\overline{S1}$, and $\overline{S0}$ The **status bits** indicate the function of the current bus cycle. These signals are normally decoded by the 8288 bus controller described later Table 2 shows the functions coded by these pins.

$\overline{RQ}$/GT1 and $\overline{RQ}$/GT0 The **request/grant** pins request direct memory accesses (DMA) during maximum mode operation. These lines are both bi-directional and are used to request and grant a DMA operation.

$\overline{LOCK}$ The **lock output** is used to lock peripherals off the system. This pin is activated by using the LOCK: prefix on any instruction.

QS_1 and QS_0 The **queue status** bits show the status of the internal instruction queue. These pins are provided for access by the numeric coprocessor (8087).

Table 3 Queue Status bits [Applicable for maximum mode]

QS_1	QS_0	Function
0	0	Queue is idle
0	1	First byte of opcode
1	0	Queue is empty
1	1	Subsequent byte of opcode

Input and output characteristics of 8086

Input Characteristics

Logic Level	Voltage	Current
0	0.8 V maximum	±10 µA maximum
1	2.0 V minimum	±10 µA maximum

Output Characteristics

Logic Level	Voltage	Current
0	0.45 V maximum	2.0 mA maximum
1	2.4 V minimum	–400 µA maximum

Mode of operation for 8086

MN/$\overline{MX}$

- The minimum mode is selected by applying logic 1 to the MN / MX input pin. This is a single microprocessor configuration.
- The maximum mode is selected by applying logic 0 to the MN / MX input pin. This is a multi micro processors configuration.

PREPARED BY
SHAHADAT HUSSAIN PARVEZ

8086 Block diagram

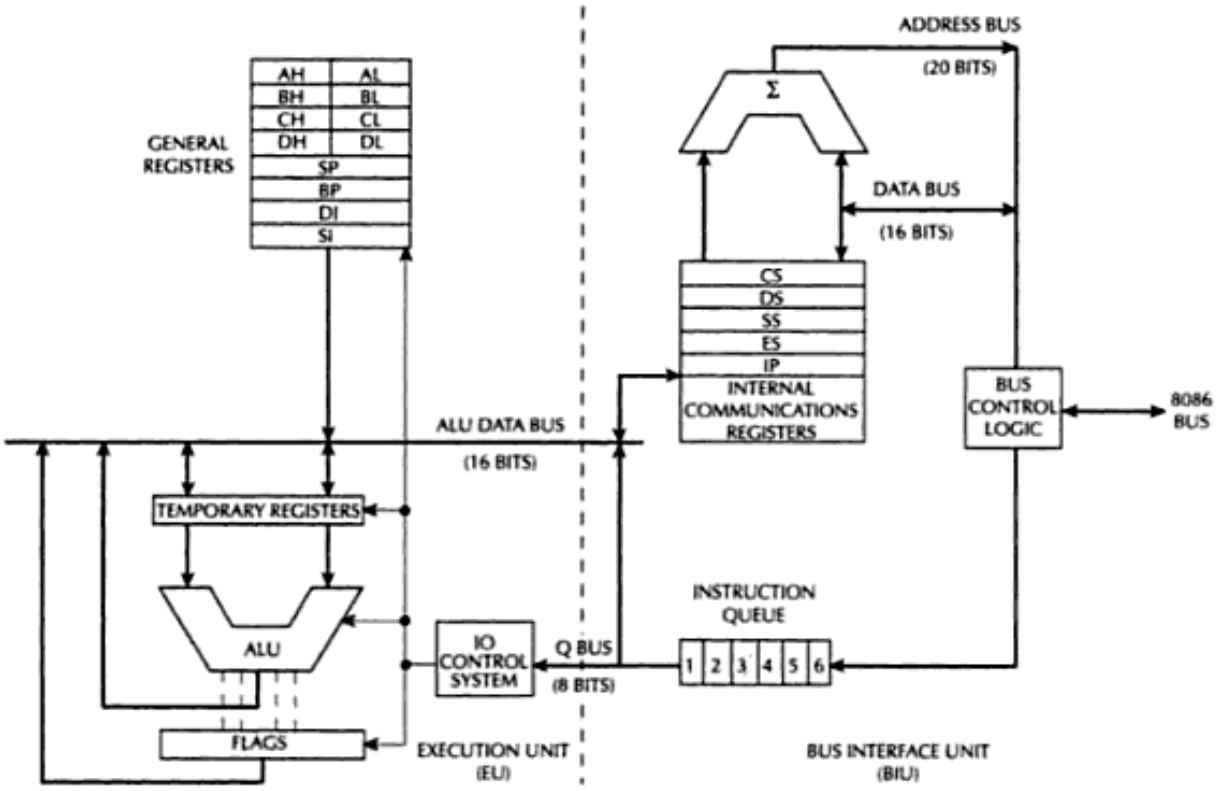


Figure 2 8086 Block diagram

The Execution unit contains

1. Control circuitry, Instruction decoder, and ALU
2. Flag register
3. General Purpose register

The BIU contains

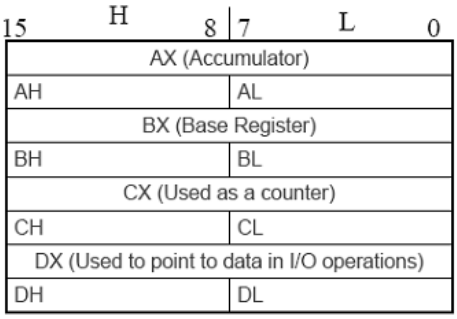
1. The queue
2. Segment registers
3. Instruction pointer

Registers of 8086

Category	Bits	Register Names
General	16	AX, BX, CX, DX
	8	AH, AL, BH, BL, CH, CL, DH, DL
Pointer	16	Stack Pointer (SP) Base Pointer (BP)
Index	16	Source Index (SI) Destination Index (DI)
Segment	16	Code Segment (CS) Data Segment (DS) Stack Segment (SS) Extra Segment (ES)
Instruction	16	Instruction Pointer (IP)
Flag	16	Flag Register (FR)

PREPARED BY
SHAHADAT HUSSAIN PARVEZ

General purpose registers



- **Data Registers** are normally used for storing temporary results that will be acted upon by subsequent instructions
- Each of the registers are 16 bits wide (AX, BX, CX, DX)
- General purpose registers can be accessed as either 16 or 8 bits. E.g. AH is upper half and AL is lower half of AX

Pointer and index registers

SP	Stack Pointer
BP	Base Pointer
SI	Source Index
DI	Destination Index
IP	Instruction Pointer

- The registers in this group are all 16 bits wide. Low and high bytes are not accessible. These registers are used as memory pointers.
- Stack pointer register is a 16-bit register, which holds the address from the start of the segment to the memory location, where a word was most recently stored on the stack.
- Base Pointer Register
- Instruction pointer is a 16-bit register used to hold the address of the next instruction to be executed.

Example: **MOV AH, [SI]**

Moves the byte stored in memory location whose address is contained in register SI to register AH

Segment Registers

BIU has 4 segment buses, i.e. CS, DS, SS& ES. It holds the addresses of instructions and data in memory, which are used by the processor to access memory locations. It also contains 1 pointer register IP, which holds the address of the next instruction to executed by the EU.

- **CS** − It stands for Code Segment. It is used for addressing a memory location in the code segment of the memory, where the executable program is stored.
- **DS** − It stands for Data Segment. It consists of data used by the program andis accessed in the data segment by an offset address or the content of other register that holds the offset address.
- **SS** − It stands for Stack Segment. It handles memory to store data and addresses during execution.
- **ES** − It stands for Extra Segment. ES is additional data segment, which is used by the string to hold the extra destination data.

Flag Register

It is a 16-bit register that behaves like a flip-flop, i.e. it changes its status according to the result stored in the accumulator. It has 9 flags and they are divided into 2 groups − Conditional Flags and Control Flags.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				O	D	I	T	S	Z		A		P		C

What Do the Letters Mean?

C	Carry		T	Trap
P	Parity		I	Interrupt
A	Auxillary Carry		D	Direction
Z	Zero			
S	Sign			
O	Overflow			

Conditional Flags

It represents the result of the last arithmetic or logical instruction executed. Following is the list of conditional flags −

- **Carry flag** − This flag indicates an overflow condition for arithmetic operations.
- **Auxiliary flag** − When an operation is performed at ALU, it results in a carry/barrow from lower nibble (i.e. D0 – D3) to upper nibble (i.e. D4 – D7), then this flag is set, i.e. carry given by D3 bit to D4 is AF flag. The processor uses this flag to perform binary to BCD conversion.
- **Parity flag** − This flag is used to indicate the parity of the result, i.e. when the lower order 8-bits of the result contains even number of 1's, then the Parity Flag is set. For odd number of 1's, the Parity Flag is reset.
- **Zero flag** − This flag is set to 1 when the result of arithmetic or logical operation is zero else it is set to 0.
- **Sign flag** − This flag holds the sign of the result, i.e. when the result of the operation is negative, then the sign flag is set to 1 else set to 0.
- **Overflow flag** − This flag represents the result when the system capacity is exceeded.

Control Flags

Control flags controls the operations of the execution unit. Following is the list of control flags −

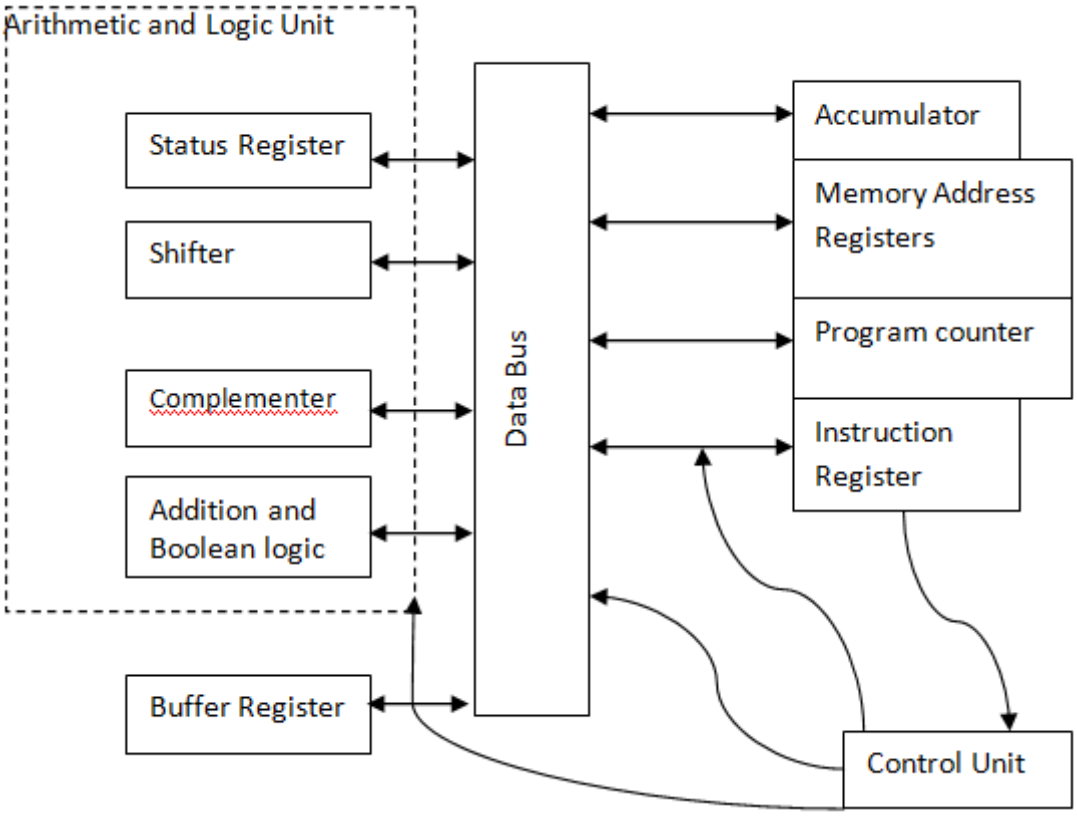
- **Trap flag** − It is used for single step control and allows the user to execute one instruction at a time for debugging. If it is set, then the program can be run in a single step mode.
- **Interrupt flag** − It is an interrupt enable/disable flag, i.e. used to allow/prohibit the interruption of a program. It is set to 1 for interrupt enabled condition and set to 0 for interrupt disabled condition.
- **Direction flag** − It is used in string operation. As the name suggests when it is set then string bytes are accessed from the higher memory address to the lower memory address and vice-a-versa.

Instruction queue

BIU contains the instruction queue. BIU gets upto 6 bytes of next instructions and stores them in the instruction queue. When EU executes instructions and is ready for its next instruction, then it simply reads the instruction from this instruction queue resulting in increased execution speed.

Fetching the next instruction while the current instruction executes is called **pipelining**.

ALU in 8086



1. What is the function of Execution Unit?
2. What is the function of BIU?
3. What is the function of ALU?
4. What are the registers in 8086?
5. What are the functions of flag register?
6. What are the functions of segments registers?
7. Why is queue important in microprocessor?
8. What is the difference between maximum mode and minimum mode?
9. What is the maximum amount of memory addressable by 8086 and why?
10. Why is 8086 a 16 bit microprocessor?

PREPARED BY
SHAHADAT HUSSAIN PARVEZ