Unit 3 8086 Microprocessor

Structure:

3.1 Introduction

In the previous unit you studied about the Evolution of Intel microprocessors and comparision of various Intel Microprocessors. In this unit you are going to study the architecture of 8086 microprocessor in detail. The 8086 was the first 16-bit Microprocessor to be introduced by Intel Corporation. The word 16-bit means that its arithmetic logical unit, internal registers, and most of its instructions are designed to work with 16-bit binary words. In this unit you will also study about machine language instructions along with the concepts of interpreters, compilers and debuggers and their role in 8086 microprocessor family and also the timing diagrams of instruction.

Objectives:

After studying this unit, you should be able to:

- explain the architecture of 8086 microprocessor
- list the important features of 8086
- discuss pin diagram of 8086
- explain the functions of interpreter, compiler and debugger
- explain the timing diagram of minimum mode memory read operation

3.2 Architecture of 8086 Microprocessor

8086 microprocessor is a, 16 bit Intel IC made by using HMOS technology. It consists of 2900 transistors. The term 16 bit means that the arithmetic logic unit, its internal registers and instructions are designed to work with 16 bit binary word. It has 16 bit data bus and 20 bit address bus.

The important features of 8086 microprocessor are:

- 8086 is a 16bit processor. It's ALU, internal registers work with 16 bit binary word.
- 8086 has a 16bit data bus. It can read or write data to a memory/port either 16bits or 8 bit at a time.
- 8086 has a 20bit address bus which means, it can address up to 2^{20} = 1MB memory location.
- Operating frequency range of 8086 is 6-10 MHz
- 8086 microprocessor employ parallel processing.
- It can support up to 64K I/O ports.
- It requires up to +5Volts power supply.
- Permissible range of the memory address is from 00000H to FFFFFH.
- 8086 operates in two modes i.e., minimum mode and maximum mode.
- Memory is byte addressable.
- It can prefetch up to 6 bytes of instructions and queue them so that execution speed is increased.

The pipelined architecture of 8086 microprocessor is as shown in figure 3.1.The term architecture as used in microprocessor describes the functional components that make up the micro processing unit and interaction between them.

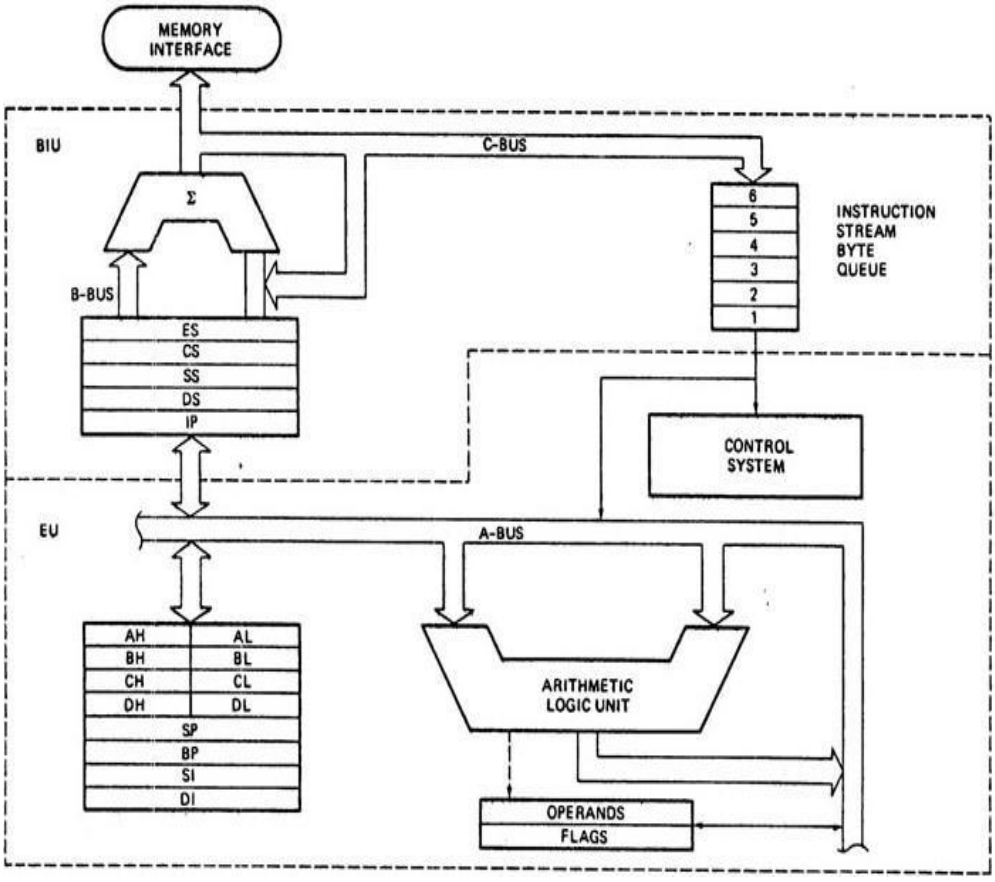


Figure 3.1: Block Diagram of 8086 microprocessor

The architecture is divided into two functional units as you can see in above figure:

1. Bus Interface Unit(BIU)
2. Execution Unit(EU)

1. Bus Interface Unit (BIU): The BIU handles all the data and address on the buses for the execution unit (EU). It performs all bus operations such as instruction fetching, reading and writing operands for memory and calculating the addresses of the memory operands. The instruction bytes are transferred to the instruction queue. Execution unit use the instruction queue to execute the instructions. Both BIU and EU work asynchronously to execute instructions by using Pipelining mechanism which means overlapping of instruction fetch and execute mechanism.

The pipelining results in efficient use of system bus and increases system performance.

The parts of BIU are:

i. Instruction Queue
ii. Segment Registers
iii. Instruction Pointer

i. Instruction Queue

The prefetched instruction bytes are stored in a first in first out (FIFO) group of registers called an ***instruction queue*** for the Execution Unit (EU). When the EU is ready for its next instruction, it simply reads the instruction from this instruction queue. This is much faster than sending out an address to the system memory and to send back the next instruction byte. Fetching the next instruction while the current instruction is executing is called ***pipelining***. 8086 uses pipelining to speed up the execution of the program. In 8086, a 6-bytes instruction queue is presented at the Bus Interface Unit (BIU). It is used to prefetch and store at the maximum of 6 bytes of instruction code from the memory.

ii. Segment Registers

The BIU contains four 16-bit segment registers. These are:

i) Extra Segment (ES) register,
ii) Code Segment (CS) registers,
iii) Data Segment (DS) registers, and
iv) Stack Segment (SS) registers.

The upper bits of the starting address of each of the segments are held by these segment registers. The part of the segment starting address stored in the segment register is known as ***segment base***. The description of segment registers is as follows:

i) Code Segment (CS) Register: It is a 16 bit register used for accessing instructions referenced by instruction pointer or we can say that it is used to address the memory location in the code segment, where the executable program is stored.

ii) Stack Segment (SS) Register: It is also a 16 bit register and is used to store the data used when the subprogram is executing. It contains the data pointed by Stack Pointer (SP) and base pointer (BP).

***iii) Data Segment (DS) Register*:** It is a 16 bit register containing the data used by the program.

iv) Extra Segment (ES) Register: It is a 16 bit register used to hold extra destination data during some string manipulation operations. It contains the data pointed by DI register.

iii. **Instruction Pointer (IP)*:*** It holds the 16 bit address of the next instruction to be executed.

Logical and Physical Address

Addresses within a segment can range from address 00000H to address 0FFFFH. An address within a segment is called an ***logical address***. A logical address gives the displacement from the address base of the segment to the desired location space. This "real" address is called the ***physical address***. The physical address is 20 bits long and corresponds to the actual address in binary code output by the BIU on the address bus lines. For selecting memory location, microprocessor will transfer 20 bit physical address on address lines. This physical address is generated by adding 20 bit base address with 16 bit effective address. The 16 bit register from where microprocessor will take 16 bit effective address is called ***memory pointer***. The important memory pointers of 8086 microprocessor are: Instruction Pointer (IP), Stack Pointer (SP),Base Register (BX), Base Pointer(BP), Source Index register(SI) and Destination Index register (DI).

The physical address (PA) which is a 20 bit actual address put on address bus in the 8086 is calculated by adding the Offset address(or effective address) with the contents of one of the segment registers left shifted by 4 bit positions. This (Segment value * 10H) is called 20 bit base address.

Physical Address (PA) = (Segment value * 10H) + Offset Value

The Physical Address is equal to the value in any Segment register (CS or DS or SS or ES) multiplied by 10H (or shifted one hexadecimal byte to the left and adding an extra 0 to the end of the hex number) plus the value in an Offset register called *offset value* or offset address or effective address (example, the value of IP) added to it. The figure 3.2 shows the generation of Physical address.

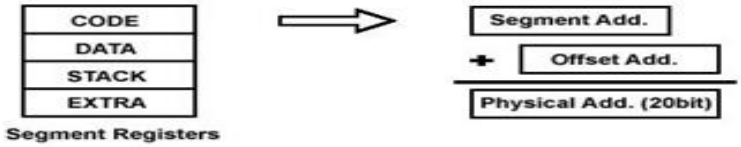


Figure 3.2: Generation of Physical Address

For example, consider the segment register to be a CS. if the CS register contains 1234H, and the IP register contains 5678H (i.e. 16 bit effective address), then the next instruction is fetched from physical address 179B8H, which is 1234H times 10H (12340H) plus 5678H.Therefore,

Physical Address (PA) = (1234H * 10H) + 5678H =179B8 H

i.e. if **CS:** 1234 H and IP: 5678H, Then PA is calculated as:

CS: 12340H ; the value of CS is shifted left and 0 is addedat the end.

IP: + 5678H ;16 bit Offset address or Effective Value or Effective address

_____

PA: 179B8 H ;20 bit physical address

_____

2. Execution Unit (EU)

It decodes the instructions fetched by BIU, generates control signals and executes the instruction**.** The main parts of Execution unit are:

i) Control system,
ii) Arithmetic and Logic Unit (ALU)
iii) Instruction decoder unit.
iv) Flag Register
v) General purpose registers and
vi) Pointers and index registers

(i) Control System: Control system performs various internal operations.

(ii) Arithmetic and logical unit: Arithmetic and logical unit performs different arithmetic operations like increment, decrement operations etc. and logical operations like AND, OR, NO T etc.

(iii) Instruction Decoder: This is used to decode the instructions that make up a program when they are being processed, and to determine in what actions must be taken in order to process them. These

decisions are normally taken by looking at the opcode of the instruction, together with the addressing mode used.

(iv) Flag Register: Flag register is a 16 bit register having 16 flip flops. Flag Register shows the condition or changes produced by the execution of an instruction and these flags get modified as and when mathematical or logical operations are performed. 8086 microprocessor uses only 9 flip-flops, so there are only nine flags in 8086 microprocessor. Figure 3.3 shows the diagram of a flag register.

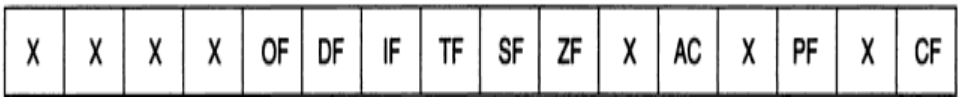


Figure 3.3: 8086 Flag Register

Note: X-Not used as a flag (Undefined)

The flags are categorized into two categories:

1. Conditional Flags or Status Flags
2. Control Flags

Conditional Flags: Depending upon the status of the status of the result obtained in ALU, the microprocessor stores the corresponding status bit (0/1) in these flip-flops. There are six status flags are CF, PF, AC, ZF, SF and OF and the description these flags is as follows:

- **Carry flag (CF):** This flag is set when the result of an unsigned arithmetic operation is too large to fit in the destination register. This happens when there is an end carry in an addition operation or there is an end borrows in a subtraction operation. A value of 1 = carry and 0 = no carry**..**
- **Auxiliary Carry flag (AC):** If an operation performed in ALU generates a carry/borrow from lower nibble (i.e. D0 – D3) to upper nibble (i.e. D4 – D7), the AF flag is set i.e. when carry generated from by D3 bit to D4, AF flag is set. This is not a general-purpose flag; it is used internally by the processor to perform Binary to BCD conversion.
- **Parity Flag (PF):** This flags reflects the number of 1s in the result of an operation. If the number of 1s is even its value = 1 and if the number of 1s is odd then its value = 0.
- **Zero Flag (ZF):** If the result of the arithmetic or logical operation is 0 then it is set to 1 else reset to 0.

- **Sign Flag (SF):** If the result of the operation is negative then this flag is set to one and if the result is positive then it is set to zero.
- **Overflow Flag (OF):** When the signed numbers are added or subtracted and there is a signed overflow then this flag is set else reset. If overflow flag is set then this indicates that the result has exceeded the capacity of machine.

Control Flags: These flags are not automatically set or reset. These flags are needed to be set or reset by the user to control certain operations. Description of these flags is as follows:

- **Trap Flag (TF):** When this flag is set then user can execute one instruction at a time and can do debugging.
- **Interrupt Flag (IF):** If this flag is set then interrupt (interruption of program) is enabled if reset then interrupt is disabled.
- **Direction Flag (DF):** If this flag is set, accessing of string bytes is from higher to lower memory address and if this flag is reset then accessing of string bytes is from lower to higher memory address.

General Purpose Registers

8086 microprocessor contains eight general purpose registers. They are, AL, BL, CL, DL, AH, BH, CH and DH. All these registers can store 8 bits (one byte) of data. A pairs of registers are also used to store 16 bits of data. The register pairs are: AL-AH, BL-BH, CL-CH and DL-DH and these pairs are called as AX, BX, CX and DX respectively.

AX (AL-AH) register: This register pair is used as 16 bit accumulator in 16 bit operation and is used in arithmetic and logical computations.

BX (BL-BH) register: This register pair is used as a memory pointer in data segment and is also used for based, based indexed and register indirect addressing modes (addressing modes will be discussed in unit 4)

CX (CL-CH): This register pair is known as counter register and is used in loop instructions.

DX (DL-DH): This register pair called Data Register DX is used in DIV instructions to hold higher word of the 32 bit operand and remainder after division. It is also used in multiplication operation to hold higher word of 32 bit result. It is also used to hold the 16 bit IO address.

Pointers and Index Registers

Stack Pointer Register: It is a 16-bit register pointing to program stack in stack segment.

Some other registers are also there in execution unit. These are

SI (source index): It is a 16-bit register. SI is used for indexed, based indexed and register indirect addressing (these addressing modes are explained in unit 4), as well as a source data address in string manipulation instructions**.** .

DI (destination index): It is a 16-bit register. DI is used for indexed, based indexed and register indirect addressing, as well as a destination data address in string manipulation instructions

BP (base pointer): It is a 16-bit register pointing to data in stack segment. BP register is usually used for based, based indexed or register indirect addressing.

Self Assessment Questions

1. 8086 microprocessor has _____ bit data bus and _____ bit address bus.
2. The prefetched instruction bytes are stored in a first in first out (FIFO) group of registers called an _____ for the Execution Unit.

3.3 Pin Diagram of 8086 Microprocessor

The figure 3.4 shows the pin diagram of 8086 microprocessor. The 8086 Microprocessor is a 16-bit CPU available in different clock rates (i.e. operating frequencies) and packaged in a 40 pin CERDIP (CERamic Dual In-line Package) or plastic package.

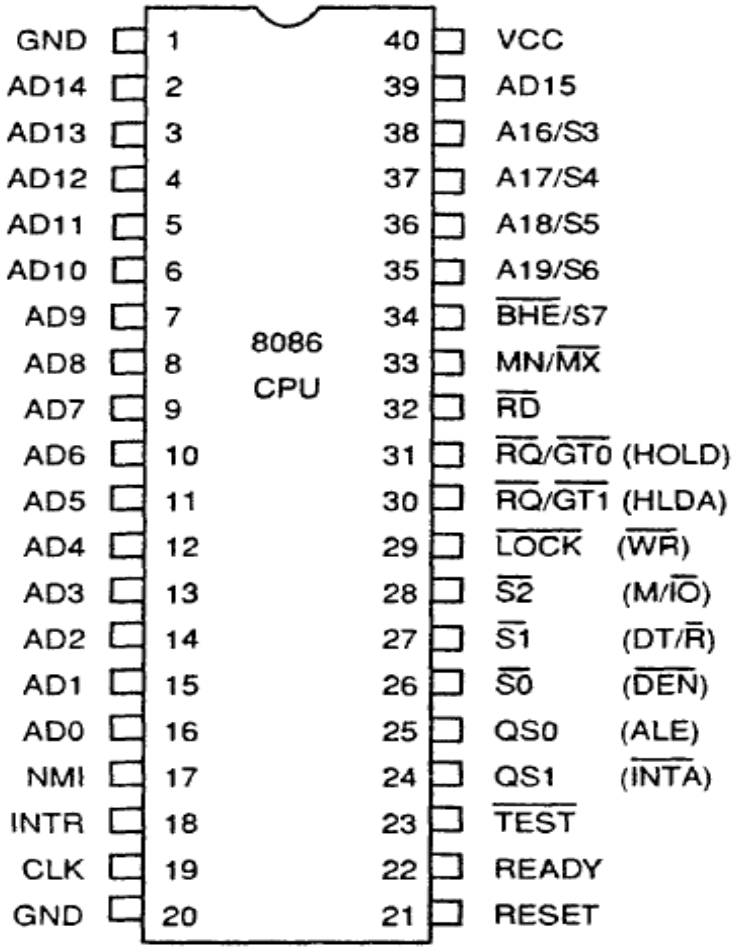


Figure 3.4: Pin Diagram of 8086

8086 microprocessor operates in two modes, the ***minimum mode*** (MN/$\overline{MX}$=1) and ***maximum mode*** (MN/$\overline{MX}$=0). In minimum mode the 8086 microprocessor works in a single processor environment. All control signals for memory and I/O are generated by the microprocessor. In maximum mode 8086 works in a multiprocessor environment. Control signals for memory and I/O are generated by an external BUS Controller. All control signals for memory and I/O are generated by the microprocessor 8086 and these signals can be divided into three categories. They are:

(i) The signals that have common functions in both the modes (minimum and maximum).

(ii) The signals that have specific function in minimum mode.

(iii) And the signals that have specific functions in maximum mode.

Pin Description of 8086 common to both the Modes:

AD_0-AD_{15}: These are time multiplexed address and data lines. During T1 state these lines carry address and during rest of the T states (T2, T3, TW and T4), these lines carry data.

T1, T2, T3 and T4 are four bus cycles whose description is given in section 3.5.

A19/S6, A18/S4, A17/S4, A16/S3: These lines are upper address lines and are multiplexed with status signals (S3-S6). These lines act as address lines during T states and shows status for remaining T states. These lines remain low during I/O operations. S4 and S3 indicates that which segment is being used by the processor. S5 keeps the value of interrupt enable flag. S6 signal indicates whether 8086 is bus master or other device is a bus master.

Table 3.1: Encoding of S4 and S3

S4	S3	Indications
0	0	alternate data
0	1	stack
1	0	code or none
1	1	data

$\overline{BHE}$/S7: Bus High Enable/Status

BHE stands for bus high enable signal and shows the data transfer over the higher order (D15-D8) data bus. When $\overline{BHE}$ is low then data is transferred over bus lines D15-D18 during T1 state of every machine cycle. The two signals are encoded as per the table 3.2.During rest of the T sates the this line send the status signal S7.The status signal S7 is used by numeric coprocessor to determine whether the processor is 8086 or 8088.

$\overline{RD}$: Read: It is an active low output signal. When this signal is low it indicates the processor is performing read (memory or I/O read) operation. It is activated during T2, T3, and TW (it indicates wait state) of any read machine cycle and in other T states it is high.

READY: This is an active high input signal and is used for slower peripherals.

When a peripheral device is ready to receive or transmit the data then it sends the high on READY signal to the processor.

INTR: Interrupt Request: This is an active high level triggered input signal. This signal is sampled during the last clock cycle of every instruction. It is sampled to determine the availability of the request and if there is any request then the processor enters the interrupt acknowledgement cycle.

$\overline{\text{TEST}}$**:** It is active low input signal. Execution will continue if $\overline{\text{TEST}}$ = 0 else processor will be in an idle state.

NMI: It is a Non-Maskable Interrupt. It is an edge triggered input and cannot be masked. And if there is a transition from low to high it indicates interrupt response at the end of the current instruction.

RESET: It is an input signal. This signal results in the termination of current activities of the system and results in restarting of the system.

CLK: Clock Input
The clock input provides the basic timing for processor operation and bus control activity.

Vcc:+5 volts power supply is connected to this pin.

GND: It is the GROUND pin.

MN / $\overline{MX}$: It indicates that in which mode processor will operate i.e., in minimum mode or maximum mode. If this pin is zero, then 8086 microprocessor operates in minimum mode else it operates in maximum mode.

Pin Description of 8086 in Minimum Mode*:*

1. M/I$\overline{O}$ (Status line): This is an output signal used to distinguish whether microprocessor is going to access memory or IO. If M/$\overline{IO}$ become zero then it means that it will be an IO operation otherwise a memory operation.

2. $\overline{\text{WR}}$ (Write): It is an active low output signal, if high indicates that the write operation is being performed either from I/O or from memory.

3. $\overline{INTA}$ (Interrupt Acknowledge): It is active low output signal. If this signal is low then this indicates the acceptance of interrupt signal by the processor.

4. ALE (Address latch enable): It is an active high output signal and provided by processor to latch the address into 8282/8283 address latch. It is a high pulse active during T1 of any machine cycle. It also shows that the address present on data or address bus is valid or not.

5. DT/ $\overline{R}$ (Data transmit/receive): it is data transmit and receive signal and shows the direction of the data flow through transceivers .If it is high then data will be transmitted else data will come to the microprocessor. This signal is used in minimum mode and is used to select the direction of the 8286 transceiver

6. $\overline{DEN}$ (Data Enable): It is an active low data enable signal and indicates the presence of valid data over address/data lines.

7. HOLD (hold request) and HLDA (hold acknowledge): It means hold and hold acknowledge. When HOLD is high then it means that bus is requested by some other external device. When HOLD signal is received by processor it releases the bus and issues an HLDA signal as an acknowledgement.

Pin Description of 8086 in Maximum Mode:

1. $\overline{S2}, \overline{S1}, \overline{S0}$:These signals are applied to the bus controller 8288 to generate different control signals the INTA signal in maximum mode. These status line are encoded, as shown in table 3.3, by the bus controller In the T1,T2 and T3 states, these signals are active and are passive during T3 and T4 state.

Table 3.2: S0,S1,S0 Encoding

S2	S1	S0	Operation
0	0	0	Interrupt acknowledge
0	0	1	Read I/O port
0	1	0	Write I/O port
0	1	1	Halt
1	0	0	Code Access
1	0	1	Read Memory
1	1	0	Write Memory
1	1	1	Passive

2. $\overline{RQ}/\overline{GT0}$ and $\overline{RQ}/\overline{GT1}$ (Request/ Grant): These pins are used by the other local bus master in maximum mode, to force the processor to release the local bus at the end of the processor current bus cycle. Each of the pin is bidirectional with $\overline{RQ}/\overline{GT0}$ having higher priority than $\overline{RQ}/\overline{GT1}$.

3. $\overline{LOCK}$: It is an active low output signal. This signal does not allow other devices to take control over the system bus when LOCK=0. The LOCK instruction activates this signal and it is active till the completion of next instruction.

4. QS1 and QS0: These are the queue status signals and gives the status of code-prefetch queue. These signal gives the status of 8086 queue to math coprocessor as these signals are interfaced with the signals of math coprocessor. The queue status signals are encoded in table 3.4

Table 3.3: Encoding of QS1 and QS2

QS1	QS0	Encoding
0	0	No operation
0	1	First byte of opcode from queue
1	0	Empty the queue
1	1	Subsequent byte from the queue

Self Assessment Questions

3. The _____ input provides the basic timing for processor operation and bus control activity.
4. _____ are time multiplexed address and data lines.
5. WR is an active high output signal. (True or False).

3.4 Machine Language Instructions

Machine code instructions are different for every processor or processor family. Machine language can be defined as a pattern of bits. So different patterns are there for different instructions. Different families of processors have their own instruction set. In a machine level instruction set all the instruction are of same length or of variable length. As the machine level instructions are organized into bit patterns their bit pattern arrangement depends on the processor architecture and sometimes on the instruction type also. An instruction has two fields one is **opcode** – that specifies the function to be performed and another is the **operand –** that specifies on

what the operation is to be performed. So the instructions contain opcode field, operand, these also contain addressing modes (you will study addressing modes in next unit 4), addressing offset and the value itself.

The program and data are stored in the main memory of a computer. These data and program stored are in binary form (0 and 1). The machine language instructions are fetched by the processor and then are executed. All the details of these machine language programs should be complete, correct and should be unambiguous as processor will execute the instructions in the manner it is written.

In 8086 microprocessor, assembly level language is used to write instructions and programs. The assembly language is one step higher than the machine language (0s and 1s) and is easy to understand and write as compared to machine language. Instructions written in assembly level language are converted into machine language instruction by a system software called ***Assembler.*** The format of assembly language instruction is shown in figure 3.5.

Label Field	**Opcode Field**	**Operand Field**	**Comment field**
NEXT	ADD	AL,07H	;ADD CORRECTION FACTOR

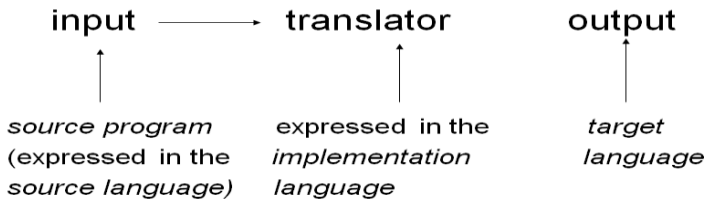
Figure 3.5: Format of assembly language instruction in 8086

Self Assessment Questions

6. _____ language can be defined as a pattern of bits.
7. An instruction has two fields one is _____ and another is _____.
8. In 8086 microprocessor we use _____ to give instructions and to write the programs for 8086.

3.5 Interpreter, Compiler and Debugger

The ***language translators*** are used to convert the source language (high level or assembly level language) into machine level language.



The input to the language translator is the source language and the output is the machine level language (language of 0s and 1s). For high level languages, we use compiler and interpreter as a translators. The programming language that we use in 8086 microprocessor is assembly level language. So the language translator called an ***assembler*** is used for translating assembly language into machine language. The Now let us see the brief description of these different language translators.

- ***Interpreter***

  An interpreter translates high level language into machine level language and the translation is done line by line that is one line at a time. The program has to be translated by interpreter at each and every time the program executes. In languages like BASIC, QBASIC and in VB the interpreters are used.

  The advantage of interpreter is that the programmer immediately knows if there are any error/sin the line of the code and can immediately correct it. The figure 3.6 shows the block diagram of an interpreter.

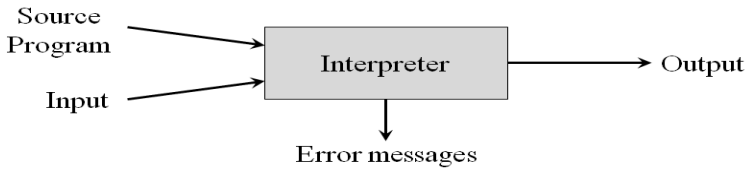


Figure 3.6: Interpreter

- ***Compiler***

  Compiler also translates high level language into machine level language but the complete source program is translated into machine level language at once and there is no need to recompile the program at each time the program is executed. The working of a compiler is shown in figure 3.7.

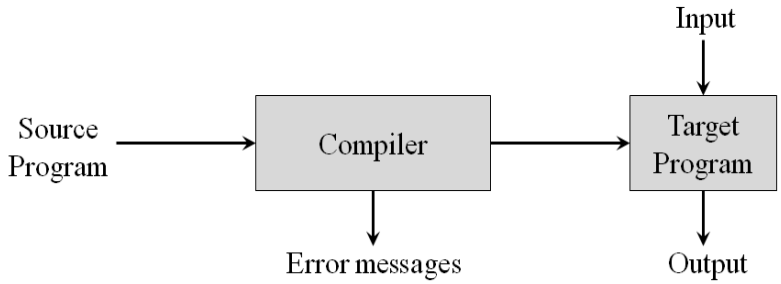


Figure 3.7: Working of Compiler

When any file is executed the compiler first checks for the syntax to be correct, then converts the high level language into machine language. The efficiency of the compiled code is more as it generates a complete machine language program which can then be executed. For high level languages compiler is used, so the target code does not depends on machine. In other words, the when the source code is once compiled then the code can be executed on any machine. So the code is machine independent.

- ***Assembler***

  It is also a type of translator that translates assembly level language into machine level language. The assembly level language consists of mnemonics and is written in the operand part of assembly language instruction e.g. ADD (for addition), SUB (for subtraction) etc. The assembly level language is also machine/processor dependent but it is easier to read and write assembly language than machine language. For 8086microprocessor we use assembly language program to write programs and assembler is used to convert 8086 microprocessor assembly language program into machine language.

- ***Debugger***

  Debuggers are the programs that enable the programmers to find and remove errors from the program. It enables the programmers to run the program step by step so that the programmer can find out the exact location of the error and the cause of the error and by having these information, the programmer can eliminate the errors. It also performs the function of loading the machine language program into system's memory.

Breakpoints can also be inserted in the program so that debugger can debug the program till that point only and can display the output till that point which helps user to check the program. It also allows checking the contents of the registers and memory. By this you can trace the program step by step which will help you in checking the program.

Debugger Commands:

1. **ASSEMBLE (A or a):** This instruction enables you to write the program from a specified address.
2. **DUMP (Dor d):** This instruction is used to check the contents of a particular memory location.
3. **ENTER(E or e):** It is used to enter the data into a particular memory location.
4. **GO (G or g):** It is used to execute the program.
5. **MOVE (M or m):** It is used to move the program from one location (source location) to other location (destination location).
6. **QUIT (Q or q):** It is used to exit debugger.
7. **REGISTER(R or r):** It is used to show the register's content.
8. **TRACE (T or t):** It is used to trace the program that is used to execute the program step by step.
9. **UNASSEMBLE:** It is used to un assemble the program and when this command is used you can see both the opcode and assembly language program.

Self Assessment Questions

9. Interpreter translates _____language into machine level language and the translation is done line by line.
10. Complier translates high level language into _____.
11. _____ enables the programmers to run the program step by step so that the programmer can find out the exact location of the error.
12. _____ instruction is used to exit debugger.

3.6 Instruction Execution Timing

In 8086 microprocessor there is a multiplexed address and data bus and is known as time multiplexed data and address bus.The reason for the

multiplexing of buses is that, there is an optimum utilization of the pins. It is possible to demultiplex the bus by using a few latches and transceivers, whenever required. There are four processor cycles for all the bus cycles. The bus cycles are referred as the T1, T2, T3 and T4 as shown in figure 3.8.

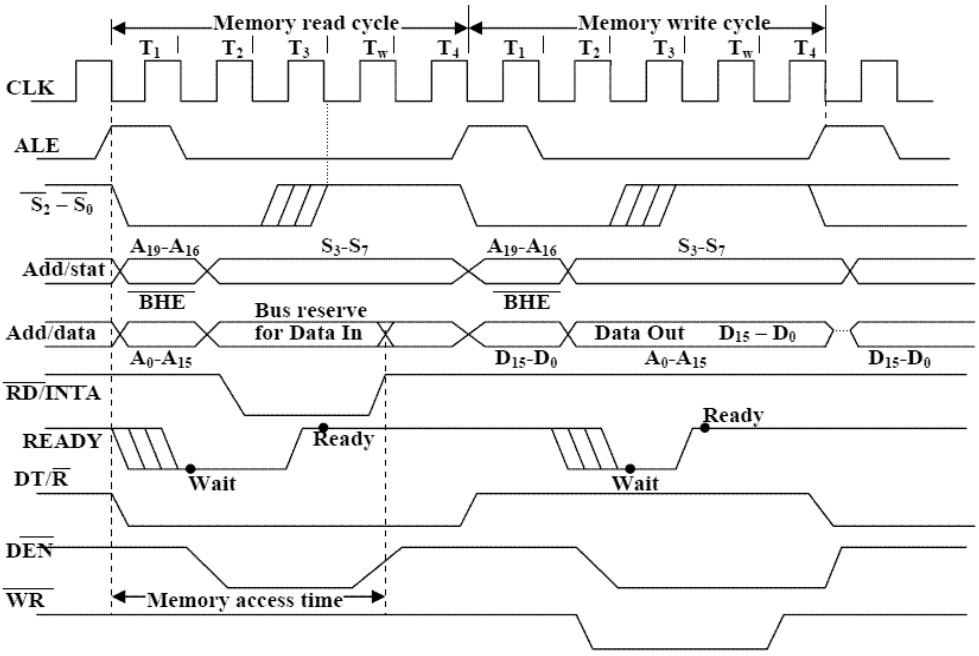


Figure 3.8: General bus operation cycle for 8086

The address is transmitted during T1 cycle and this address will be there on the bus for only one cycle T1.During T2 the bus is tristated for changing the direction of the bus for the data read cycle. During T3 and T4, the transfer of data takes place. Suppose if the device is slow and is not ready for data transfer then wait states T_w are inserted between T3 and T4.These states are also known as idle states T_i or inactive sates. These wait states are utilized by processor for internal use. During T1 the address latch enabled (ALE) signal is generated by processor (minimum mode) or by bus controller (maximum mode) depending upon the MN/$\overline{MX}$.

The negative edge of this ALE pulse is used to separate the address and the data or status information. In maximum mode, the status lines S0, S1 and S2 are used to indicate the type of operation. Status bits S3 to S7 are

multiplexed with higher order address bits and the $\overline{BHE}$ signal. Address is valid during T1 while status bits S3 to S7are valid during T2 through T4.

3.6.1 Minimum Mode Bus Operation Cycle

The timing diagram of minimum mode memory read operation is shown in figure 3.9.

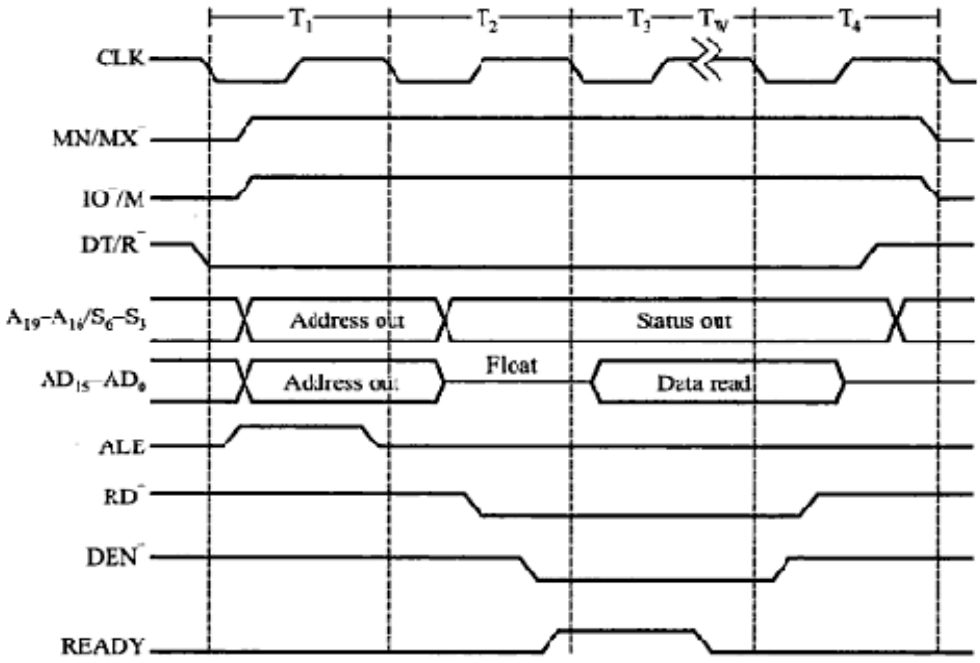


Figure 3.9: Timing Diagram for 8086 Memory Read Operation in Minimum Mode

As you can see in figure 3.8, the bus cycle starts during T1 and the valid address is latched together with setting minimum and maximum mode, input output, data transmit and receive mode. The read control signals are issued and data enable signal is asserted during T2. The data from the memory is read during T3.And all the bus signals are deactivated during T4.

The timing diagram of 8086 microprocessor memory write cycle is shown in the figure 3.10. It is similar to the read cycle timing diagram, the difference is that during this the DT/$\overline{R}$ line becomes high which shows the data transfer operation for the processor to memory. To

validate the data the $\overline{DEN}$ line becomes low. The $\overline{WR}$ line goes low showing a write operation.

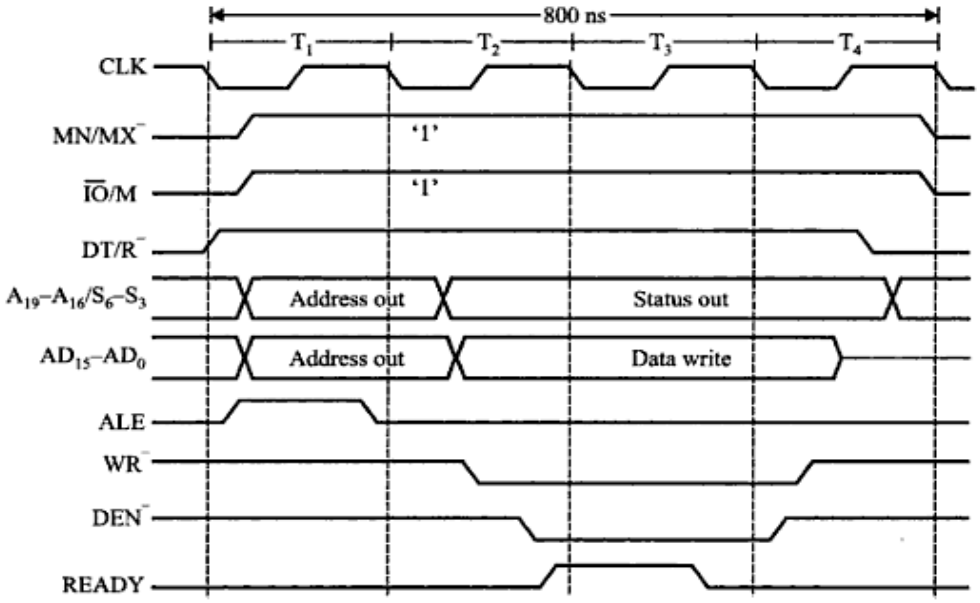


Figure 3.10: Timing Diagram for 8086 Memory Write Operation in Minimum Mode

3.6.2 Maximum Mode Bus Operation Cycle

In the maximum mode, the 8086 is operated by strapping the MN/$\overline{MX}$ pin to ground. In this mode, the processor derives the status signal S2, S1, S0. Another chip called bus controller derives the control signal using this status information. In the maximum mode, there may be more than one microprocessor in the system configuration. The control signal logic levels and timing diagram are same to that of read operation, except for data transmit and receive, memory read and write cycle. Here T-states correspond to the time during which $\overline{DEN}$ is low, write control goes low, DT/$\overline{R}$ is high and data output is available from the processor on data bus.

The figures 3.11 and 3.12 show the timing diagrams for 8086 maximum mode memory read operation and memory write operation respectively.

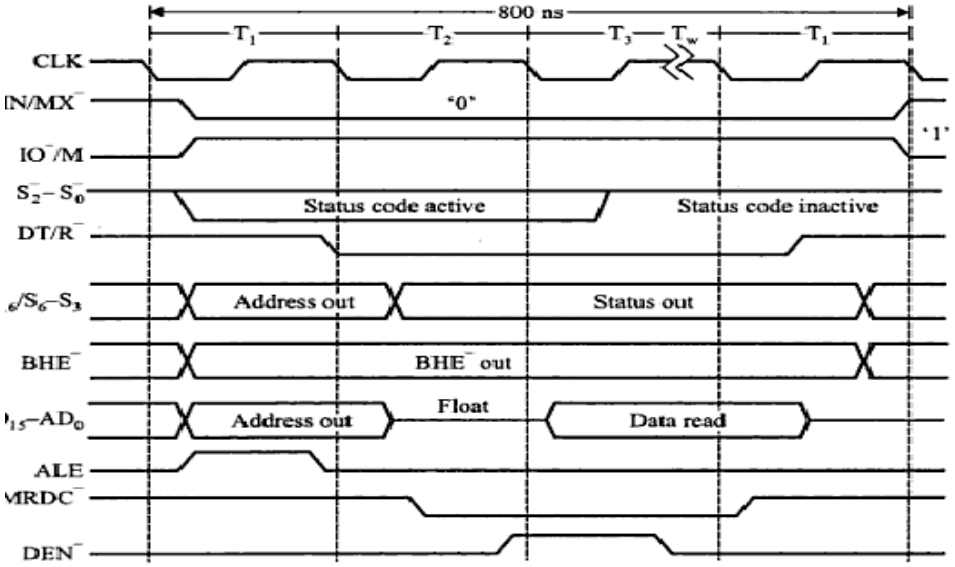


Figure 3.11: Maximum mode memory read cycle of 8086 microprocessor

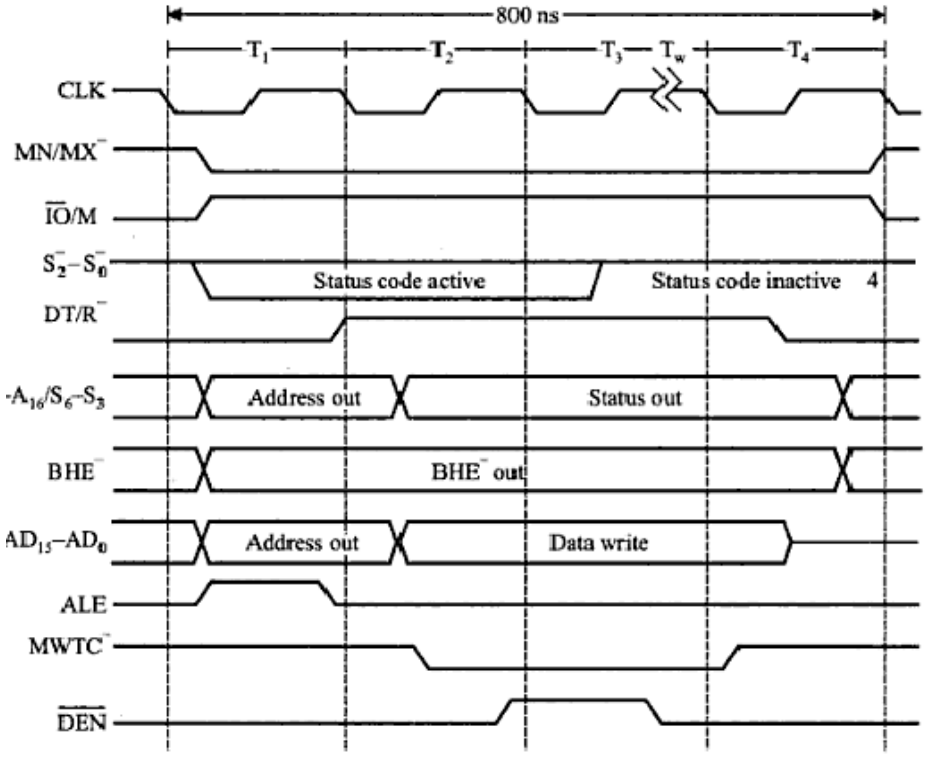


Figure 3.12: Maximum Mode memory write cycle of 8086 microprocessor

Self Assessment Questions

13. In 8086 microprocessor address and control bus are multiplexed. (True/False)
14. How many processor cycles are there for all the bus cycles?
15. During which cycle the address is transmitted?
16. In which mode the processor derives the status signals S2, S1 and S0?

3.7 Summary

- 8086 microprocessor is a 16 bit Intel IC made by using HMOS technology. It has 16 bit data bus and 20 bit address bus.
- 8086 operates in two modes that is minimum mode and maximum mode and requires up to +5Volts power supply.
- 8086 follows pipelined architecture. The architecture is divided into two functional units Bus Interface Unit (BIU) and execution unit (EU).

  The BIU handles all the data and address on the buses for the execution unit (EU). Execution Unit decodes instructions fetched by BIU generates control signals and executes the instruction.
- Flag register shows the condition or changes produced by the execution of an instruction.
- Stack pointer is a 16-bit register pointing to program stack in stack segment.
- Machine language can be defined as a pattern of bits and Machine code instructions are different for every processor or processor family.
- Interpreter translates high level language into machine level language and the translation is done line by line.
- Compiler translates high level language into machine level language but the complete source program is translated into machine level language at once by the compiler.
- Debuggers are the programs that enable the programmers to find and remove errors from the program.
- In 8086 microprocessor there is a multiplexed address and data bus and is known as time multiplexed data and address bus.

- There are four processor cycles for all the bus cycles and are referred as the T1, T2, T3 and T4.

3.8 Terminal Questions

1. Draw and explain the architecture of 8086 microprocessor.
2. Explain the pin diagram of 8086 microprocessor.
3. Differentiate between interpreter and compiler.
4. Write a short note on machine language instruction.
5. Explain instruction execution with timing diagram.

3.9 Answers

Self Assessment Questions

1. 16 bit, 20 bit
2. Instruction Queue
3. Clock
4. AD0-AD15
5. False
6. Machine
7. Opcode, Operand
8. Assembly level Language
9. High level
10. Machine level language
11. Debugger
12. QUIT
13. False
14. Four
15. T1 Cycle
16. Maximum mode

Terminal Questions

1. 8086 microprocessor is a 40 pin 16 bit Intel IC made by using HMOS technology. It has 16 bit data bus and 20 bit address bus. Refer section 3.2.
2. The 8086 microprocessor is a 16-bit CPU available in different clock rates and packaged in a 40 pin CERDIP. Refer to section 3.2.3.

3. Interpreter translates high level language into machine level language and the translation is done line by line.Compiler also translates high level language into machine level language but the complete source program is translated into machine level language at once. Refer to section 3.4 for more details.
4. Machine code instructions are different for every processor or processor family. Machine language can be defined as a pattern of bits and different pattern is there for different instructions. Refer to section 3.3.
5. There are four processor cycles for all the bus cycles. The bus cycles are referred as the T1, T2, T3 and T4. The address is transmitted during T1 cycle, Refer to section 3.5.