LECTURE NOTES

ON

MICROPROCESSOR & MICROCONTROLLER

SEMESTER-4TH

Prepared By

Er. Sabyasachi Patra

Asst. Prof. Electronics & Tele-Communication Engineering

Department Of Electronics & Tele-Comm. Engineering

C V RAMAN POLYTECHNIC

BBSR

CONTENTS

Chapter1

Microprocessor (Architecture and Programming-8 bit-8085)

LEC 2: MICROPROCESSOR 8085 BASICS - YouTube ←---------- follow these video series for more information

Introduction to Microprocessor and Microcomputer & distinguish between them.

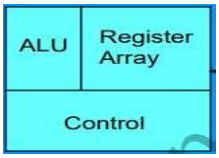
Microprocessor

The microprocessor is a programmable device that takes in numbers, performs on them arithmetic or logical operations according to the program stored in memory and then produces other numbers as a result.

Internally, the microprocessor is made up of 3 main units.

The Arithmetic/Logic Unit (ALU)

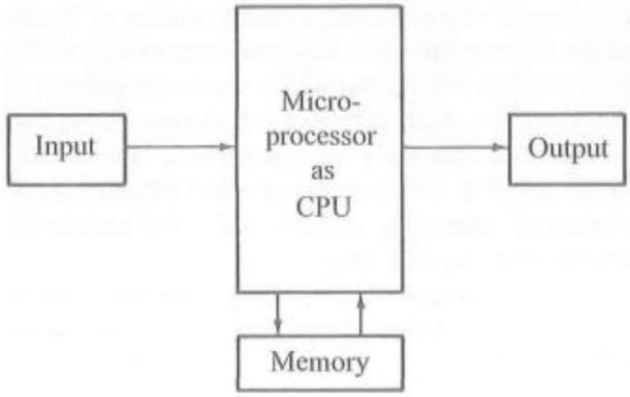
The Control Unit,



An array of registers for holding data while it is being manipulated.

Microcomputer

A microcomputer is a small, relatively inexpensive computer with a microprocessor as its central processing unit (CPU). It includes a microprocessor, memory and minimal input/output (I/O) circuitry



mounted on a single printed circuit board (PCB).

Memory stores information such as instructions and data in binary format (0 and 1). It provides this information to the microprocessor whenever it is needed.

Usually, there is a memory "sub-system" in a microprocessor-based system. This sub-system includes: – The registers inside the microprocessor – Read Only Memory (ROM) • used to store information that does not change.

Random Access Memory (RAM) (also known as Read/Write Memory). • used to store information supplied by the user. Such as pro www.noteshit.com grams and data.

Concept of Address bus, data bus, control bus & System Bus

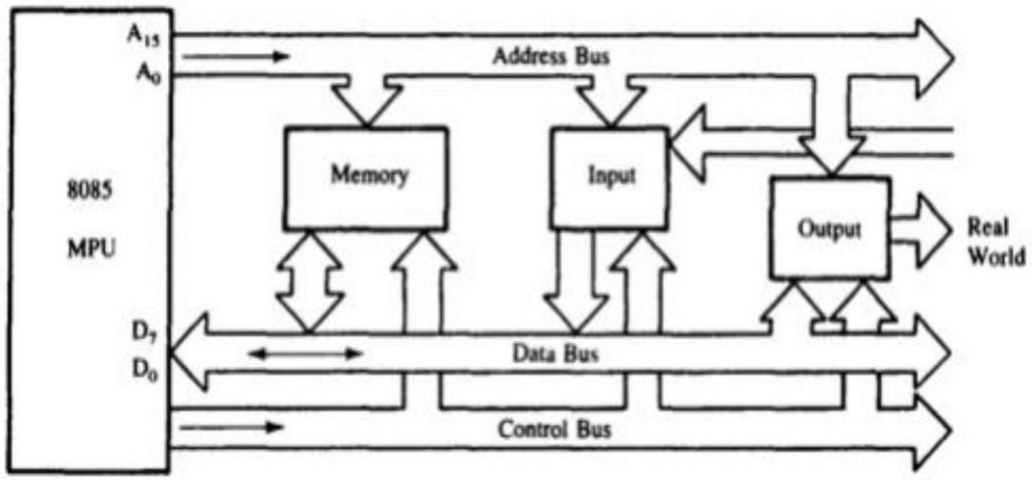
A bus is a group of wires/lines that carry similar information. 8085 MP uses a number of busses, collection of wires, which transmit binary numbers, one bit per wire. A typical microprocessor communicates with memory and other devices (input and output) using three busses: Address Bus, Data Bus and Control Bus.

Address Bus: The address bus is a group of 16 lines generally identified as A0 to A15. The address bus is unidirectional: bits flow in one direction-from the MPU to peripheral devices. The MPU uses the address bus to perform the first function: identifying a peripheral or a memory location.

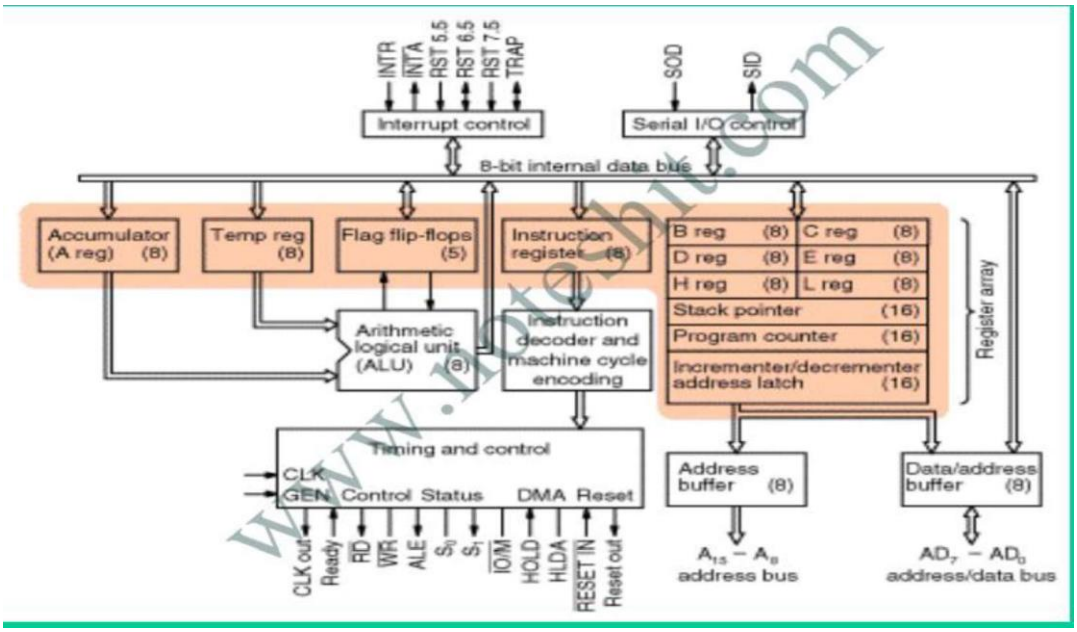
Data Bus: INTEL 8085 is a 8 bit microprocessor. The data bus is a group of eight lines used for data flow. 8 bit of data can be transmitted in parallel form. These lines are bi-directional - data flow in both directions between the MPU and memory and peripheral devices. The MPU uses the data bus for transferring binary information. The eight data lines enable the MPU to manipulate 8-bit data ranging from 00 to FF (28 = 256 numbers).

Control Bus: The control bus carries synchronization signals and providing timing signals. The MPU generates specific control signals for every operation it performs. These signals are used to identify a device type with which the MPU wants to communicate.

General Bus structure Block diagram.



Basic Architecture of 8085 (8 bit) Microprocessor



The 8085 microprocessor is an 8-bit processor available as a 40-pin IC package and uses +5 V for power. It can run at a maximum frequency of 3 MHz. Its data bus width is 8-bit and address bus width is 16-bit, thus it can address 2^{16} = 64 KB of memory.

It consists of the following:

Control Unit Generates signals within µP to carry out the instruction, which has been decoded. In reality causes certain connections between blocks of the µP to be opened or closed, so that data goes where it is required, and so that ALU operations occur.

Arithmetic Logic Unit The ALU performs the actual numerical and logic operation such as 'add', 'subtract', 'AND', 'OR', etc. Uses data from memory and from Accumulator to perform arithmetic. Always stores result of operation in Accumulator.

Registers The 8085 includes six registers, one accumulator, and one flag register. In addition, it has two 16-bit registers: the stack pointer and the program counter. The 8085 has six general- purpose registers to store 8-bit data; these are identified as B, C, D, E, H, and L as shown in the figure. They can be combined as register pairs - BC, DE, and HL - to perform some 16-bit operations. The programmer can use these registers to store or copy data into the registers by using data copy instructions.

Temporary Register: It is used to hold the data during the arithmetic and logical operations. Instruction Register: When an instruction is fetched from the memory, it is loaded in the instruction register.

Instruction Decoder: It gets the instruction from the instruction register and decodes the instruction. It identifies the instruction to be performed.

Serial I/O Control: It has two control signals named SID and SOD for serial data transmission.

Timing and Control unit: It has three control signals ALE, RD (Active low) and WR (Active low) and signals IO/M(Active low), S0 and S1. ALE is used for provide control signal to synchronize the components of microprocessor and timing for instruction to perform the operation. RD (Active low)

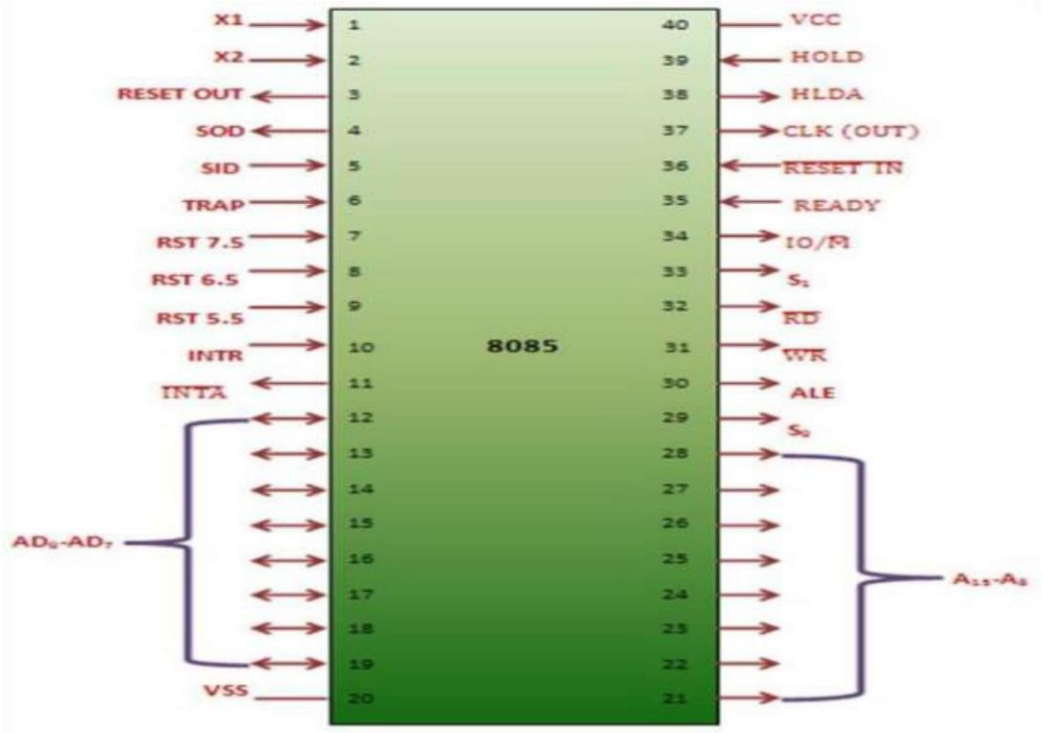
and WR (Active low) are used to indicate whether the operation is reading the data from memory or writing the data into memory respectively.IO/M(Active low) is used to indicate whether the operation is belongs to the memory or peripherals.

Signal Description (Pin diagram) of 8085 Microprocessor

The logic pin layout and signal groups of the 8085nmicroprocessor are shown in Fig.

All the signals are classified into six groups:

- Address bus
- Data bus
- Control & status signals
- Power supply and frequency signals
- Externally initiated signals
- Serial I/O signals



Address and Data Buses:

- A8 – A15 (output, 3-state): Most significant eight bits of memory addresses and the eight bits of the I/O addresses. These lines enter into tri-state high impedance state during HOLD and HALT modes.

• AD0 – AD7 (input/output, 3-state): Lower significant bits of memory addresses and the eight bits of the I/O addresses during first clock cycle. Behaves as data bus during third and fourth clock cycle. These lines enter into tri-state high impedance state during HOLD and HALT modes.

Control & Status Signals:

• ALE: Address latch enable -It occurs during the first clock cycle of a machine state and enables the address to get latched into the on chip latch of peripherals.

• RD : Read - indicates the selected memory or 1/0 device is to be read and that the Data Bus is available for the data transfer.

• WR : Write- ; indicates the data on the Data Bus is to be written into the selected memory or 1/0 location. Data is set up at the trailing edge of WR.

• IO/M -it is a status signal which distinguishes whether the address is for memory or I/O. when it goes high the address on the address bus is for an I/O device. When it goes low the address on the address bus is for a memory location.

S1 and S0 : Data Bus Status. Encoded status of the bus cycle shown in table .

S1	S0	
0	0	HALT
0	1	WRITE
1	0	READ
1	1	FETCH

Power Supply & Clock Frequency:

Vcc: +5 V power supply

• Vss: Ground reference

• X1, X2: A crystal having frequency of 6 MHz is connected at these two pins

• CLK: Clock output

Externally Initiated and Interrupt Signals:

• RESET IN : When the signal on this pin is low, the PC is set to 0, the buses are tri stated and the processor is reset.

• RESET OUT: This signal indicates that the processor is being reset. The signal can be used to reset other devices.

• READY: When this signal is low, the processor waits for an integral number of clock cycles until it goes high.

• HOLD: This signal indicates that a peripheral like DMA (direct memory access) controller is requesting the use of address and data bus.

• HLDA: This signal acknowledges the HOLD request.

• INTR: Interrupt request is a general-purpose interrupt.

• INTA : This is used to acknowledge an interrupt.

• RST 7.5, RST 6.5, RST 5.5 – restart interrupt: These are vectored interrupts and have highest priority than INTR interrupt.

• TRAP: This is a non-maskable interrupt and has the highest priority.

Serial I/O Signals:

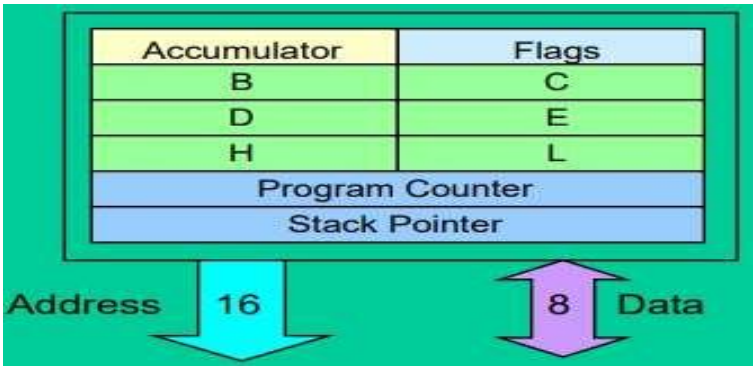
• SID: it is data line for serial input. The data on this line is loaded into the 7th bit of the accumulator when rim (read interrupt mask) instruction is executed.

SOD: it is data line for serial output. The 7th bit of the accumulator is output on sod line when sim instruction is executed.

Register Organizations, Distinguish between SPR & GPR, Timing & Control Module

Registers:-it is a collection of flip flops use to store a binary word. They are used by the microprocessor for the temporary storage and manipulation of data and instructions.

The various registers of 8085 can be broadly categorized in two sections, namely General Purpose Registers(GPR) and Special Purpose Registers(SPR) .



GPR

The 8085 has six general- purpose registers to store 8-bit data; these are identified as B, C, D, E, H, and L as shown in the figure.

They can be combined as register pairs - BC, DE, and HL - to perform some 16-bit operations. The programmer can use these registers to store or Register organization copy data into the registers by using data copy instructions.

SPR

The different SPRs and their functions are mentioned below.

Accumulator (A): It is an 8-bit register that is part of the arithmetic/logic unit (ALU). Used to store 8-bit data and to perform arithmetic and logical operations. The result of an operation is stored in the accumulator.

Flags: The ALU includes five flip-flops that are set or reset according to the result of an operation. The microprocessor uses the flags for testing the data conditions. They are Zero (Z), Carry (CY), Sign (S), Parity (P), and Auxiliary Carry (AC) flags.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S	Z		AC		P		CY

Sign Flag (S): After execution of any arithmetic and logical operation, if D7 of the result is 1, the sign flag is set. Otherwise it is reset. D7 is reserved for indicating the sign; the remaining is the magnitude of number. If D7 is 1, the number will be viewed as negative number. If D7 is 0, the number will be viewed as positive number.

Zero Flag (z): If the result of arithmetic and logical operation is zero, then zero flag is set otherwise it is reset.

Auxiliary Carry Flag (AC): If D3 generates any carry when doing any arithmetic and logical operation, this flag is set. Otherwise it is reset

Parity Flag (P): If the result of arithmetic and logical operation contains even number of 1's then this flag will be set and if it is odd number of 1's it will be reset.

Carry Flag (CY): If any arithmetic and logical operation result any carry then carry flag is set otherwise it is reset.

Program Counter (PC): This 16-bit register sequencing the execution of instructions. It is a memory pointer. Memory locations have 16-bit addresses, and that is why this is a 16-bit register. The function of the program counter is to point to the memory address of the next instruction to be executed. When an opcode is being fetched, the program counter is incremented by one to point to the next memory location.

Stack Pointer (SP): The stack pointer is also a 16-bit register used as a memory pointer. It points to a memory location in R/W memory, called the stack. The beginning of the stack is defined by loading a 16-bit address in the stack pointer.

Temporary Register: It is used to hold the data during the arithmetic and logical operations.
Instruction Register: When an instruction is fetched from the memory, it is loaded in the instruction register. Instruction Decoder:

Program Status Word(PSW): It is a combination of 8-bits where five bits indicates the 5 status flags & three bits are undefined. Psw and the accumulator treated as a 16 bit unit for stack operation.

Timing and Control unit:

It has three control signals ALE, RD (Active low) and WR (Active low) and three status signals IO/M(Active low), S0 and S1.

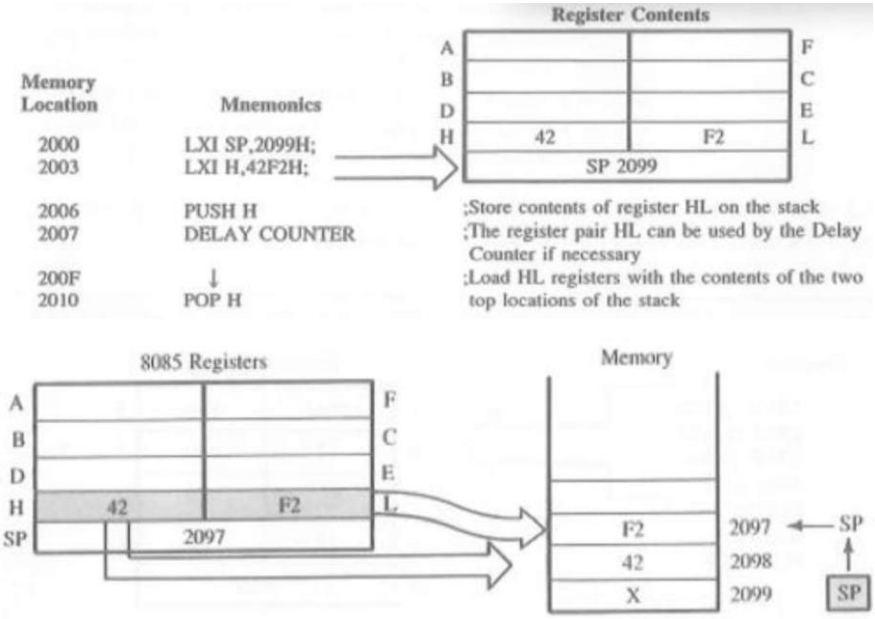
1. ALE is used for provide control signal to synchronize the components of microprocessor and timing for instruction to perform the operation.
2. RD (Active low) and WR (Active low) are used to indicate whether the operation is reading the data from memory or writing the data into memory respectively.
3. IO/M(Active low) is used to indicate whether the operation is belongs to the memory or peripherals.

Stack, Stack pointer & Stack top.

The **stack** is an area of R/W memory identified by the programmer for temporary storage of information. • The stack is a LIFO structure – Last In First Out. The programmer defines the bottom of the stack and the stack grows up into reducing address range. The programmer can use the stack to store data. And the microprocessor uses the stack to execute subroutines. The starting memory location of the stack is defined in the main memory location.

Stack pointer is a special purpose 16-bit register in the Microprocessor, which holds the address of the top of the stack.

The 8085 provides two instructions: PUSH and POP for storing information on the stack and retrieving it back.



Interrupts:-8085 Interrupts, Masking of Interrupt(SIM,RIM)

Interrupt is a process where an external device can get the attention of the microprocessor. The process starts from the I/O device .The process is asynchronous. An interrupt is considered to be an emergency signal. The Microprocessor should respond to it as soon as possible. When the Microprocessor receives an interrupt signal, it suspends the currently executing program and jumps to an Interrupt Service Routine (ISR) to respond to the incoming interrupt. Responding to an interrupt may be immediate or delayed depending on whether the interrupt is maskable or non-maskable and whether interrupts are being masked or not.

Interrupts can be classified into two types:

Software interrupts

Maskable (can be delayed) • Non-Maskable (can not be delayed)

The maskable interrupt process in the 8085 is controlled by a single flip flop inside the microprocessor. This Interrupt Enable flip flop is controlled using the two instructions "EI" and "DI" . The 8085 has a single Non-Maskable interrupt. The non-maskable interrupt is not affected by the value of the Interrupt Enable flip flop.

• hardware Interrupts

Vectored (the address of the service routine is hard-wired) • Non-vectored (the address of the service routine needs to be lied externally)

• There are two ways of redirecting the execution to the ISR depending on whether the interrupt is vectored or non-vectored. – The vector is already known to the Microprocessor – The device will have to supply the vector to the Microprocessor The 8085 Interrupts

The 8085 has 5 interrupt inputs.

– The INTR (input) • The INTR input is the only non-vectored interrupt. • INTR is maskable using the EI/DI instruction pair.it is an interrupt request signal. Among interrupts it has the lowest priority. An interrupt is used by i/o devices to transfer data to the microprocessor without wasting its time.

INTA (output)-it is an interrupt acknowledgement sent by the microprocessor after INTR is received.

RST 5.5, RST 6.5, RST 7.5(input) are all automatically vectored. • RST 5.5, RST 6.5, and RST 7.5 are all maskable. Signals are the restart interrupt, they causes an internal restart to be automatically inserted each of them of a programmable mask.

TRAP Trap interrupt is a non maskable restart interrupt. It is unaffected by any mask or Interrupt Enable. It has the highest priority of any interrupt. TRAP is also automatically vectored.

Order of priority-

TRAP→ RST 7.5----→ RST 6.5 ---→ RST 5.5 --→ INTR

When an interrupt is recognize the next instruction is executed from a fixed location in memory. A subroutine is executed which is called ISS(interrupt service subroutine).

EXTRA for information

8085 Functional Description

The 8085A is a complete 8 bit parallel central processor. It requires a single +5 volt supply. Its basic clock speed is 3 MHz thus improving on the present 8080's performance with higher system speed. Also it is designed to fit into a minimum system of three IC's: The CPU, a RAM/ IO, and a ROM or PROM/IO chip. The 8085A uses a multiplexed Data Bus. The address is split between the higher 8bit Address Bus and the lower 8bit Address/Data Bus. During the first cycle the address is sent out. The lower 8bits are latched into the peripherals by the Address Latch Enable (ALE). During the rest of the machine cycle the Data Bus is used for memory or l/O data. The 8085A provides RD, WR, and lO/Memory signals for bus control. An Interrupt Acknowledge signal (INTA) is also provided. Hold, Ready, and all Interrupts are synchronized. The 8085A also provides serial input data (SID) and serial output data (SOD) lines for simple serial interface. In addition to these features, the 8085A has three maskable, restart interrupts and one nonmaskable trap interrupt. The 8085A provides RD, WR and IO/M signals for Bus control. Status information is directly available from the 8085A. ALE serves as a status strobe. The status is partially encoded, and provides the user with advanced timing of the type of bus transfer being done. IO/M cycle status signal is provided directly also. Decoded So, S1 Carries the following status information: HALT, WRITE, READ, FETCH S1 can be interpreted as R/W in all bus transfers. In the 8085A the 8 LSB of address are multiplexed with the data instead of status. The ALE line is used as a strobe to enter the lower half of the address into the memory or peripheral address latch. This also frees extra pins for expanded interrupt capability.

Interrupt and Serial l/O

The8085A has5 interrupt inputs: INTR, RST5.5, RST6.5, RST 7.5, and TRAP. Each of the three RESTART inputs, 5.5, 6.5. 7.5, has a programmable mask. TRAP is also a RESTART interrupt except it is non- maskable. The three RESTART interrupts cause the internal execution of RST (saving the program counter in the stack and branching to the RESTART address) if the interrupts are enabled and if the interrupt mask is not set. The non-maskable TRAP causes the internal execution of a RST independent of the state of the interrupt enable or masks. The interrupts are arranged in a fixed priority that determines which interrupt is to be recognized if more than one is pending as follows: TRAP highest priority, RST 7.5, RST 6.5, RST 5.5, INTR lowest priority This priority scheme does not take into account the priority of a routine that was started by a higher priority interrupt. RST 5.5 can interrupt a RST 7.5 routine if the interrupts were re-enabled before the end of the RST 7.5 routine. The TRAP interrupt is useful for catastrophic errors such as power failure or bus error. The TRAP input is recognized just as any other interrupt but has the highest priority. It is not affected by any flag or mask. The TRAP input is both edge and level sensitive.

Chapter-2:

Instruction Set and Assembly Language Programming

An instruction is a command to the microprocessor to perform a given task on a specified data.

Each instruction has two parts: one is task to be performed, called the operation code (opcode), and the second is the data to be operated on, called the operand. The operand (or data) can be specified in various ways. It may include 8-bit (or 16-bit) data, an internal register, a memory location, or 8-bit (or 16-bit) address. In some instructions, the operand is implicit.

An instruction is a binary pattern designed inside a microprocessor to perform a specific function. The entire group of instructions, called the instruction set, determines what functions the microprocessor can perform. These instructions can be classified into the following five functional categories: data transfer (copy) operations, arithmetic operations, logical operations, branching operations, and machine-control operations.

Addressing data & Differentiate between one-byte, two-byte &three-byte instructions with examples.

Instruction format The 8085 instruction set is classified into the following three groups *according to word size*. In the 8085, "byte" and "word" are synonymous because it is an 8-bit microprocessor.

1. One-word or 1-byte instructions : One-Byte Instructions A 1-byte instruction includes the opcode and operand in the same byte. Operand(s) are internal register and are coded into the instruction.

These instructions are 1-byte instructions performing three different tasks. In the first instruction, both operand registers are specified. In the second instruction, the operand B is specified and the accumulator is assumed. Similarly, in the third instruction, the accumulator is assumed to be the implicit operand. These instructions are stored in 8-bit binary format in memory; each requires one memory location. MOV rd, rs rd <-- rs copies contents of rs into rd.

Task	Op code	Operand	Binary Code	Hex Code
Copy the contents of the accumulator in the register C.	MOV	C,A	0100 1111	4FH
Add the contents of register B to the contents of the accumulator.	ADD	B	1000 0000	80H
Invert (compliment) each bit in the accumulator.	CMA		0010 1111	2FH

2. Two-word or 2-byte instructions

In a two-byte instruction, the first byte specifies the operation code and the second byte specifies the operand. Source operand is a data byte immediately following the opcode. The instruction would require two memory locations to store in memory. MVI r,data [r <-- data]

Task	Opcode	Operand	Binary Code	Hex Code	
Load an 8-bit data byte in the accumulator.	MVI	A, Data	0011 1110	3E Data	First Byte Second Byte

3. Three-word or 3-byte instructions.

In a three-byte instruction, the first byte specifies the opcode, and the following two bytes pecify the 16-bit address, and the third byte is the high-order address. This instruction would require three memory locations to store in memory.

Three byte instructions - opcode + data byte + data byte

LXI rp, data16-bit , [rp is one of the pairs of registers BC, DE, HL used as 16-bit registers].

The two data bytes are 16-bit data in L H order of significance.

rp <-- data16

Example: LXI H, 0520H

Task	Opcode	Operand	Binary code	Hex Code	
Transfer the program sequence to the memory location 2085H.	JMP	2085H	1100 0011 1000 0101 0010 0000	C3 85 20	First byte Second Byte Third Byte

Addressing modes in instructions with suitable examples.

The microprocessor has different ways of specifying the data for the instruction. These are called "addressing modes". The 8085 has the following five types of addressing:

I. Immediate addressing :

Here, Data is present in the instruction. Load the immediate data to the destination provided.

Example: MVI R,data, MVI A, 9AH

II. Direct addressing:

Used to accept data from outside devices to store in the accumulator or send the data stored in the accumulator to the outside device. Example: IN 00H or OUT 01H

III. Register addressing :

Transfer a copy of a byte or word from source register to destination register.

Ex: MOV B, C

IV. Indirect addressing :

In register indirect addressing mode, the data to be operated is available inside a memory location and that memory location is indirectly specified by a register pair. [Indirect addressing uses the data in a register pair as a 16-bit address to identify the memory location being accessed. The HL register pair is always used in conjunction with the memory register "M". The BC and DE register pairs can be used to load data into the Accumultor using indirect addressing.]

Eg: LXI H, 4050H

MOV A,M

V. Implicit addressing:

In this addressing mode the instruction itself specifies the data to be operated upon.

Ex: CMA

[The instruction complements the content of the accumulator. No specific data or operand is mentioned in the instruction]

2.2 Addressing modes in instructions with suitable examples.

The microprocessor has different ways of specifying the data for the instruction. These are called "addressing modes". The 8085 has the following five types of addressing:

VI. **Immediate addressing** :

Here, Data is present in the instruction. Load the immediate data to the destination provided. Example: MVI R,data, MVI A, 9AH

VII. Direct addressing:

Used to accept data from outside devices to store in the accumulator or send the data stored in the accumulator to the outside device. Example: IN 00H or OUT 01H

VIII. Register addressing :

Transfer a copy of a byte or word from source register to destination register.

Ex: MOV B, C

IX. Indirect addressing :

In register indirect addressing mode, the data to be operated is available inside a memory location and that memory location is indirectly specified by a register pair.

[Indirect addressing uses the data in a register pair as a 16-bit address to identify the memory location being accessed. – The HL register pair is always used in conjunction with the memory register "M". – The BC and DE register pairs can be used to load data into the Accumultor using indirect addressing.]

Eg: LXI H, 4050H

MOV A,M

X. Implicit addressing:

In this addressing mode the instruction itself specifies the data to be operated upon.

Ex: CMA

[The instruction complements the content of the accumulator. No specific data or operand is mentioned in the instruction]

2.3 Instruction Set of 8085(Data Transfer, Arithmetic, Logical, Branching, Stack& I/O, Machine Control)

Data Transfer Operations –

These operations simply COPY the data from the source to the destination.

MOV, MVI, LDA, and STA – They transfer:

• Data between registers. • Data Byte to a register or memory location. • Data between a memory location and a register. • Data between an I\O Device and the accumulator.

The data in the source is not changed

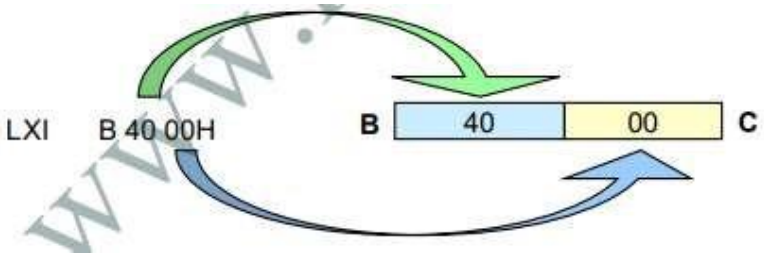
The **LXI** instruction

• The 8085 provides an instruction to place the 16-bit data into the register pair in one step.

• LXI Rp, (Load eXtended Immediate) –

The instruction LXI B 4000H

will place the 16-bit number 4000 into the register pair B, C. • The upper two digits are placed in the 1st register of the pair and the lower two digits in the 2nd .



The Memory "Register"

• Most of the instructions of the 8085 can use a memory location in place of a register. The memory location will become the "memory" register M.

• MOV M B – copy the data from register B into a memory location.

Which memory location?

• The memory location is identified by the contents of the HL register pair. – The 16-bit contents of the HL register pair are treated as a 16-bit address and used to identify the memory location .

ARITHMATIC INSTRUCTION

Addition (ADD, ADI):

– Any 8-bit number : ADI 40h

The contents of a register. ADD C

The contents of a memory location ADD M

• Can be added to the contents of the accumulator and the result is stored in the accumulator.

Subtraction (SUB, SUI):

Any 8-bit number SUI 45h

The contents of a register SUB C

The contents of a memory location SUB M

• Can be subtracted from the contents of the accumulator. The resu lt is stored in the accumulator.

Arithmetic Operations Related to Memory • These instructions perform an arithmetic operation using the contents of a memory location while they are still in memory. – ADD M • Add the contents of M to the Accumulator –

SUB M • Sub the contents of M from the Accumulator –

INR M / DCR M • Increment/decrement the contents of the memory location in place.

All of these use the contents of the HL register pair to identify the memory location being used.
Arithmetic Operations

Increment (INR) and Decrement (DCR): • The 8-bit contents of any memory location or any register can be directly incremented or decremented by 1. • No need to disturb the contents of the accumulator.

Logic Operations

• These instructions perform logic operations on the contents of the accumulator.

ANA, ANI, ORA, ORI, XRA and XRI

• Source: Accumulator and – An 8-bit number – The contents of a register – The contents of a memory location

• Destination: Accumulator

ANA R/M AND Accumulator With Reg/Mem

ANI # AND Accumulator With an 8-bit number

ORA R/M OR Accumulator With Reg/Mem

ORI # OR Accumulator With an 8-bit number

XRA R/M XOR Accumulator With Reg/Mem

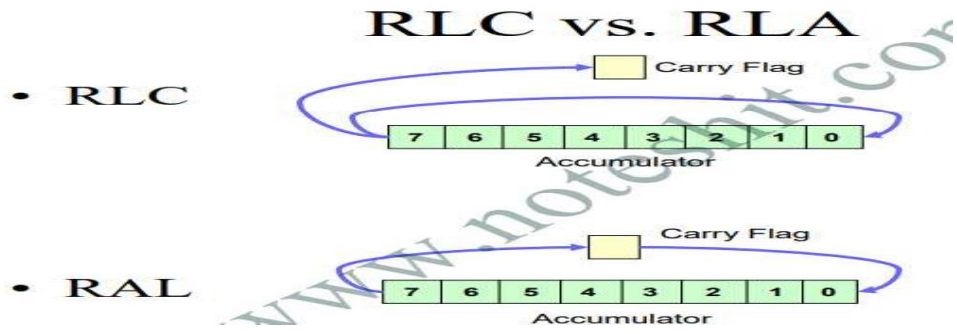
XRI # XOR Accumulator With an 8-bit number Logic Operations –

Complement: • 1's complement of the contents of the accumulator. CMA No operand

Rotate – Rotate the contents of the accumulator one position to the left or right.

RLC Rotate the accumulator left. Bit 7 goes to bit 0 AND the Carry flag.

RAL Rotate the accumulator left through the carry. Bit 7 goes to the carry and carry goes to bit 0.



RRC Rotate the accumulator right. Bit 0 goes to bit 7 AND the Carry flag.

RAR Rotate the accumulator right through the carry. Bit 0 goes to the carry and carry goes to bit 7.

Compare • Compare the contents of a register or memory location with the contents of the accumulator.

CMP R/M Compare the contents of the register or memory location to the contents of the accumulator.

CPI # Compare the 8-bit number to the contents of the accumulator. • The compare instruction sets the flags (Z, Cy, and S). • The compare is done using an internal subtraction that does not change the contents of the accumulator. (R / M / #)

Branch Operations • Two types:

Unconditional branch. • Go to a new location no matter what– JMP Address • Jump to the address specified (Go to).

Conditional Branch – Go to new location if a specified condition is met . • JZ Address (Jump on Zero) – Go to address specified if the Zero flag is set. • JNZ Address (Jump on NOT Zero) – Go to address specified if the Zero flag is not set. • JC Address (Jump on Carry) – Go to the address specified if the Carry flag is set. • JNC Address (Jump on No Carry) – Go to the address specified if the Carry flag is not set.

Machine Control – HLT • Stop executing the program. – NOP • No operation • Exactly as it says, do nothing. • Usually used for delay or to replace ins tructions during debugging.

Simple Assembly Language Programming of 8085

The programming of the problem is generally written in assembly language. The assembly language is written in mnemonics. The mnemonics are the initials or short form of the English word of the operation to be performed by the instruction

. Assembly language statements are written in standard format as given below:

Label Mnemonic Operand Comment

Label A label is a symbol or group of symbols used to represent an address of the location which is not specifically known at the time of program is written.

Mnemonic Short form of the operation to be performed.

Operand : Operand is the data on which the operation is performed. It can be a data, memory address, register or port address.

Comment The comment statement is started with the semicolon. It gives the idea of the program to the user. The program written in assembly language can be converted to machine language by hand.

For writing the program in machine language, the starting address, where the program is to be stored should be known. Now the op code of the instruction is to be written in first location (starting address)

and in the consecutive memory locations data /address of the operand is written. While storing the address in the memory locations, lower byte of the address is stored first then the upper byte.

Program1. Write assembly language program to Load the contents of memory locations 2100 H and 2101 H in B-register and C register respectively. The content of memory locations 2100 H and 2101H are 16 H and 19 H respectively.

Solution.

Label	Mnemonic	Operand	Comment
	LDA	2100H	; Loads the content of 2100H into accumulator.
	MOV B,	A	; moves the content of accumulator to B-register ($B \leftarrow A$).
	LDA	2101H	; Loads the content of 2100H into accumulator.
	MOV C,	A	; moves the content of accumulator to C-register ($C \leftarrow A$).
	HLT		; Stop processing.

Logic Operations (AND, OR, Complement 1's & 2's) & Masking of bits

Program 2.

Write an assembly language program to find the 2's complement of a hexadecimal number. The hexadecimal number 6A H is stored in memory location 2100H and the answer is to be stored in 2101 H.

Label	Mnemonic	Operand	Comment
	LDA	2100 H	; Loads the content of 2100 H into accumulator.
	CMA		; Complements the accumulator content (1's complement).
	INR A		; 1 is added to the accumulator content to get the 2's complement.
	STA	2101 H	; Loads the accumulator contents into memory location 2101 H.
	HLT		; Stop processing.

Program 3.

Write an assembly language program add two numbers 26h and 40h, then subtract number 3 from the sum. The final answer is to be stored in memory location 2100 H.

Label	Mnemonic	Operand	Comment
	MVI A,	26 H	; Loads the first number to accumulator.
	MVI B,	40 H	; Loads the second number to B-register.
	MVI C,	03 H	; Loads the third number to C-register.
	ADD B		; Adds the contents of B-register with the contents of accumulator and the answer is stored in A.
	SUB C		; Content of C gets subtracted from accumulator and difference is stored in A.
	STA	2100 H	; Answer is stored in 2100 H location.
	HLT		; Stop processing.

Program 4.

Write a program in assembly language to mask off the least significant 4 bits of a given hexadecimal number. The answer should be stored in memory location 2200 H. Let the given number is B3 H.

Solution.

The binary equivalent of B3 H is 1011 0011. The masking of the 4 least significant bits 0011 means to make 0011 to 0000. However, the four most significant bits should not be changed.

This can be done if the given number is ANDed with F0 H (1111 0000). In doing so when 4 most significant bits are ANDed with 1111 no change will be there, but 4 least significant bits will be 0000 as required.

Label	Mnemonic	Operand	Comment
	MVI A,	B3 H	; Loads the number B3H to accumulator.
	ANI	F0 H	; ANDs the accumulator with F0H and answer is loaded to accumulator.
	STA	2200 H	; Answer is stored in 2200 H location.
	HLT		; Stop processing.

Chapter-3:

TIMING DIAGRAMS

Define opcode, operand, T-State, Fetch cycle, Machine Cycle, Instruction cycle & discuss the concept of timing diagram.

Opcode & Operand

An instruction is a command to the microprocessor to perform a given task on a specified data. Each instruction has two parts:

The first part is the task or operation to be performed. This part is called the **"opcode"** (operation code).

The second part is the data to be operated on is called the **"operand".** The operand (or data) can be specified in various ways. It may include 8-bit (or 16-bit) data, an internal register, a memory location, or 8-bit (or 16-bit) address. In some instructions, the operand is implicit.

Machine Cycle

The time required by the microprocessor to complete an operation of accessing memory or input/output devices is called *machine cycle*.

Instruction cycle

Time required by the microprocessor to execute and fetch an entire instruction is called *instruction cycle*. It consists:

- **Fetch cycle –** The next instruction is fetched by the address stored in program counter (PC) and then stored in the instruction register.
- **Decode instruction –** Decoder interprets the encoded instruction from instruction register.
- **Reading effective address –** The address given in instruction is read from main memory and required data is fetched. The effective address depends on direct addressing mode or indirect addressing mode.
- **Execution cycle –** consists memory read (MR), memory write (MW), input output read (IOR) and input output write (IOW)

T-State

One time period of frequency of microprocessor is called *t-state*. A t-state is measured from the falling edge of one clock pulse to the falling edge of the next clock pulse.

TIMING DIAGRAM

It is the graphical representation of process in steps with respect to time. The timing diagram represents the clock cycle and duration, delay, content of address bus and data bus, type of operation ie. Read/write/status signals.

[Extra -Instruction flow(opcode)]

Timing Diagram is Representation of Various Control signals generated during Execution of an Instruction.

Following Buses and Control Signals must be shown in a Timing Diagram:

•Higher Order Address Bus.

•Lower Address/Data bus

•ALE

•RD

•WR

•IO/M

S_0 , S_1

The instruction code 0100 1111 (4FH) is stored in memory location 2005H. Illustrate the data flow and list the sequence of events when the instruction code is fetched by the MPU.

To fetch the instruction located in memory location 2005H, the following steps are performed:

1. The program counter places the 16-bit address 2005H of the memory location on the address bus (Figure 3.12).
2. The control unit sends the Memory Read control signal ($\overline{MEMR}$, active low) to enable the output buffer of the memory chip.
3. The instruction (4FH) stored in the memory location is placed on the data bus and transferred (copied) to the instruction decoder of the microprocessor.
4. The instruction is decoded and executed according to the binary pattern of the instruction.

Figure 3.12 shows how the 8085 MPU fetches the instruction using the address, the data, and the control buses. Figure 3.12 is similar to Figure 3.2, Memory Read operation, except that Figure 3.12 shows additional details.

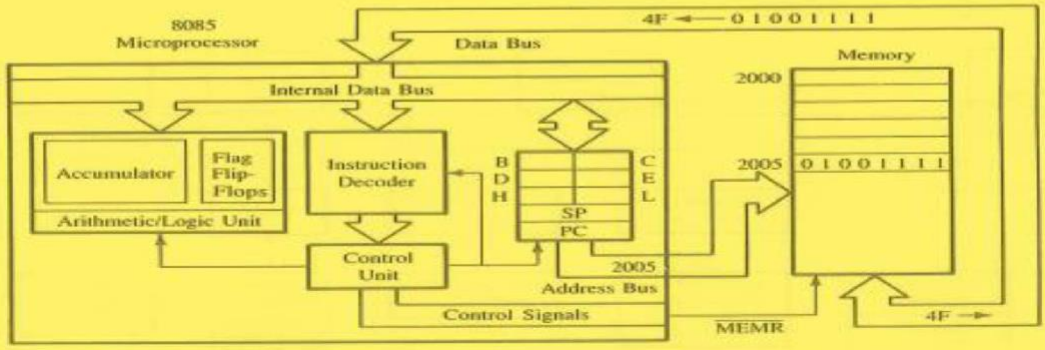


FIGURE 3.12
Instruction Fetch Operation

Draw timing diagram for memory read, memory write, I/O read, I/O write machine cycle.

Opcode fetch timing Operation:

The microprocessor requires instructions to perform any particular action. In order to perform these actions microprocessor utilizes opcode which is a part of an instruction which provides detail to microprocessor.

➢ During T1 state, microprocessor uses IO/M(bar), S0, S1 signals are used to instruct microprocessor to fetch opcode.

➢ Thus when IO/M(bar)=0, S0=S1= 1, it indicates opcode fetch operation.

➢ During this operation 8085 transmits 16-bit address and also uses ALE signal for address latching. ➢ At T2 state microprocessor uses read signal and make data ready from that memory location to read opcode from memory and at the same time program counter increments by 1 and points next instruction to be fetched.

➢ In this state microprocessor also checks READY input signal, if this pin is at low logic level ie. '0' then microprocessor adds wait state immediately between T2 and T3.

➢ At T3, microprocessor reads opcode and store it into instruction register to decode it further.

➢ During T4 microprocessor performs internal operation like decoding opcode and providing necessary actions.

➢ The opcode is decoded to know whether T5 or T6 states are required, if they are not required then the processor performs next operation.

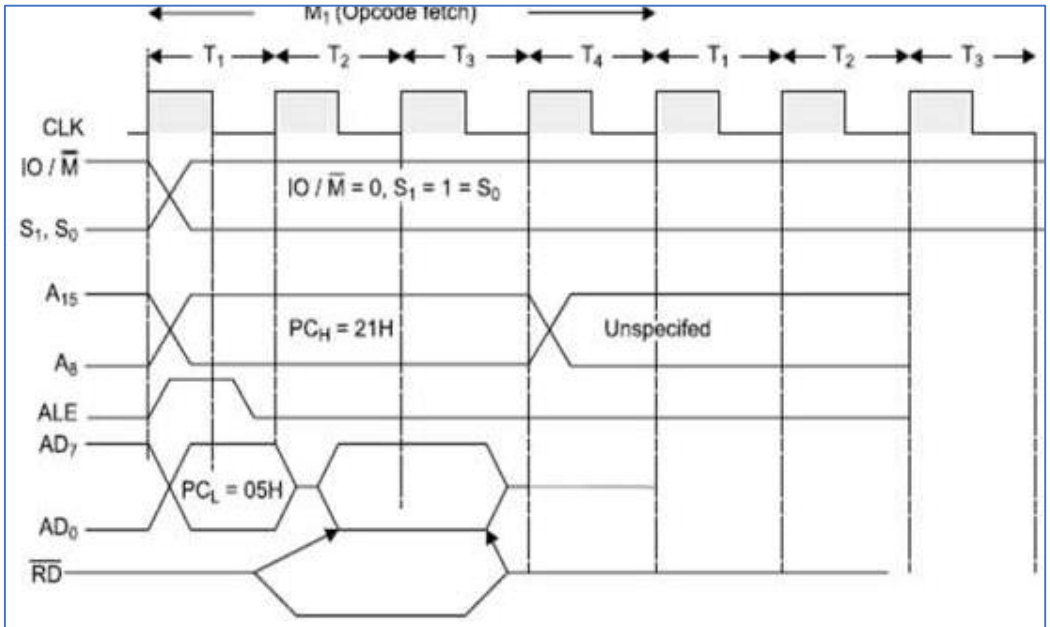


Figure 1. 5 Opcode Fetch Timing Diagram

MEMORY READ Timing Diagram

➢ It is used to fetch one byte from the memory.

➢ It requires 3 T-States.

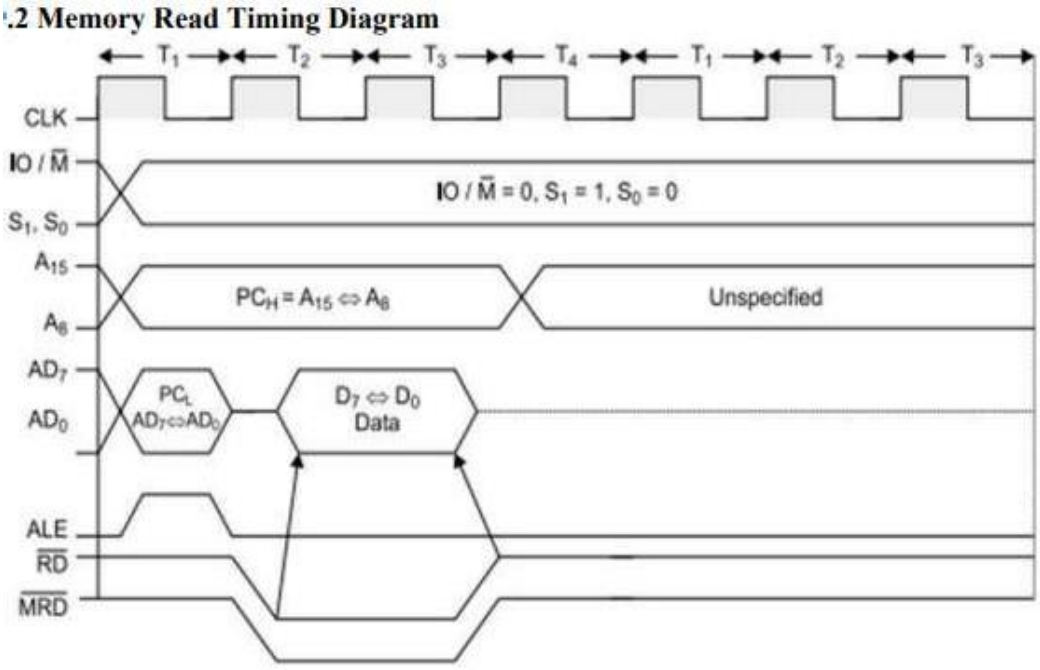
➢ It can be used to fetch operand or data from the memory.

➢ During T1, A8-A15 contains higher byte of address. At the same time ALE is high. Therefore Lower byte of address A0-A7 is selected from AD0-AD7.

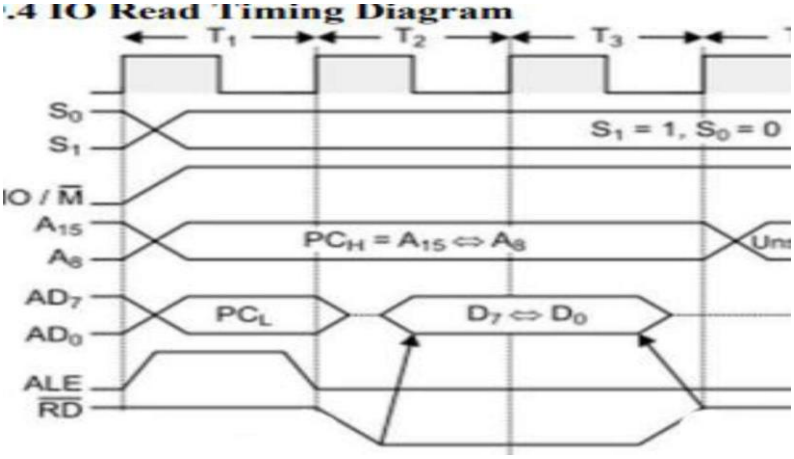
➢ Since it is memory ready operation, IO/M(bar) goes low.

➢ During T2 ALE goes low, RD(bar) goes low. Address is removed from AD0-AD7 and data D0-D7 appears on AD0-AD7.

➢ During T3, Data remains on AD0-AD7 till RD(bar) is at low signal.



.4 IO Read Timing Diagram



Memory Write Timing Diagram

➢ It is used to send one byte into memory.

➢ It requires 3 T-States.

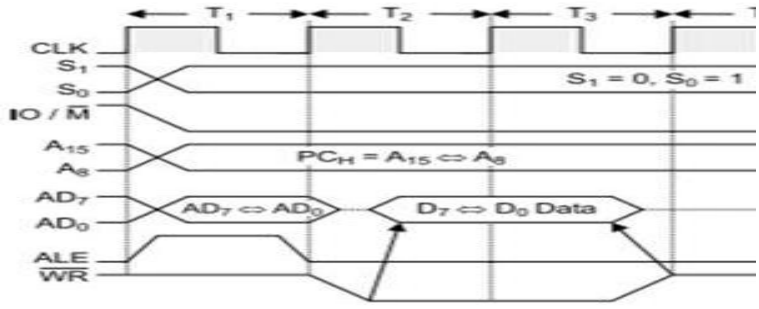
➢ During T1, ALE is high and contains lower address A0-A7 from AD0-AD7.

➢ A8-A15 contains higher byte of address. As it is memory operation, IO/M goes low.

➢ During T2, ALE goes low, WR goes low and Address is removed from AD0-AD7 and then data appears on AD0-AD7.

➢ Data remains on AD0-AD7 till WR is low.

.3 Memory write timing diagram



I/O Write Timing Diagram

It is used to writ one byte into IO device.

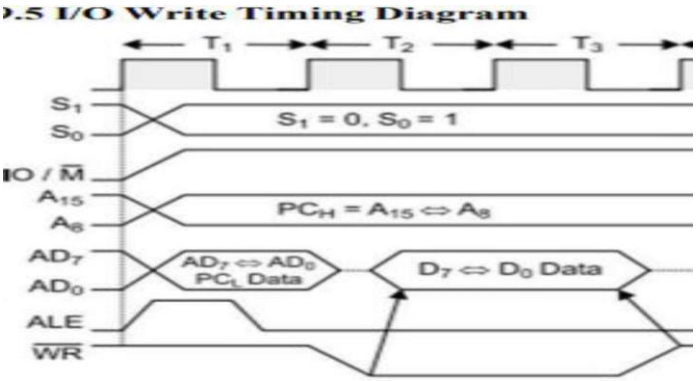
➢ It requires 3 T-States.

➢ During T1, the lower byte of address is duplicated into higher order address bus A8-A15.

➢ ALE is high and A0-A7 address is selected from AD0- AD7.

As it is an IO operation IO/M goes low.

➢ During T2, ALE goes low, WR goes low and data appears on AD0-AD7 to write data into IO device. ➢ During T3, Data remains on AD0-AD7 till WR is low.

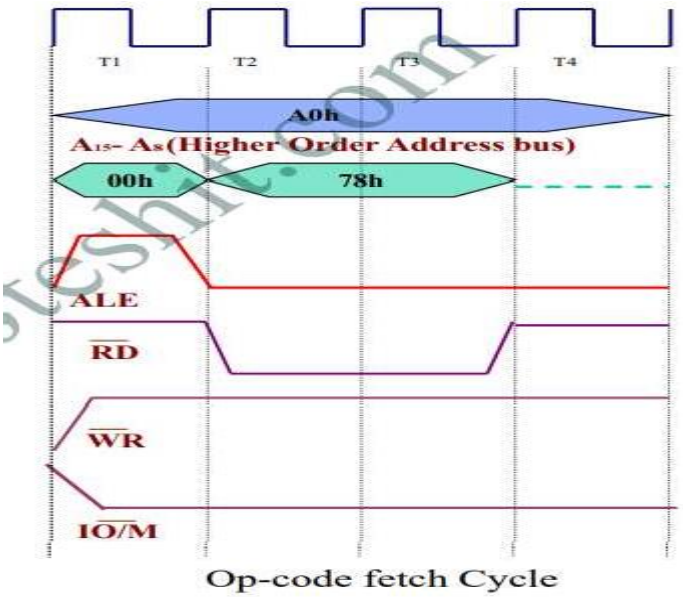


Draw a neat sketch for the timing diagram for 8085 instruction (MOV,MVI,LDA instruction).

MOV:

Instruction: A000h MOV A,B

Corresponding Coding: A000h 78



Op-code fetch Cycle

MVI

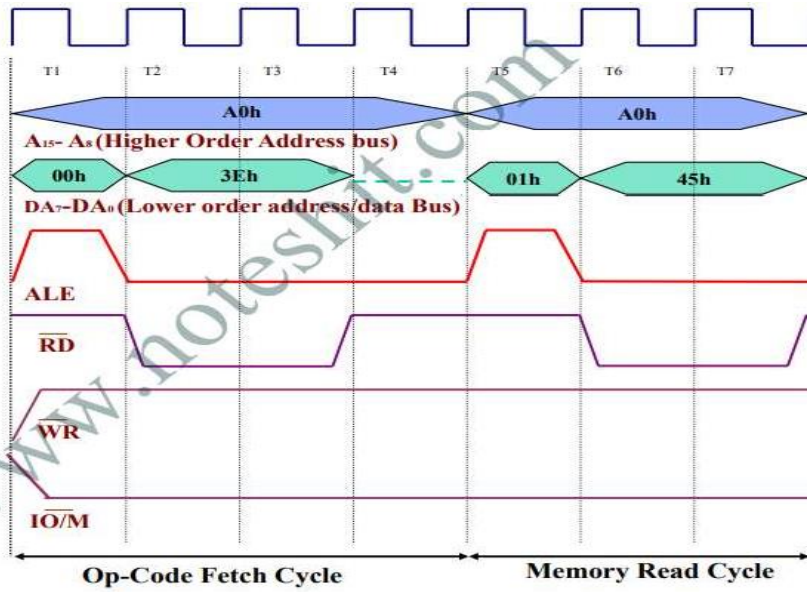
Instruction:

A000h MVI A,45h

Corresponding Coding:

A000h 3E

A001h 45



LDA

Instruction:

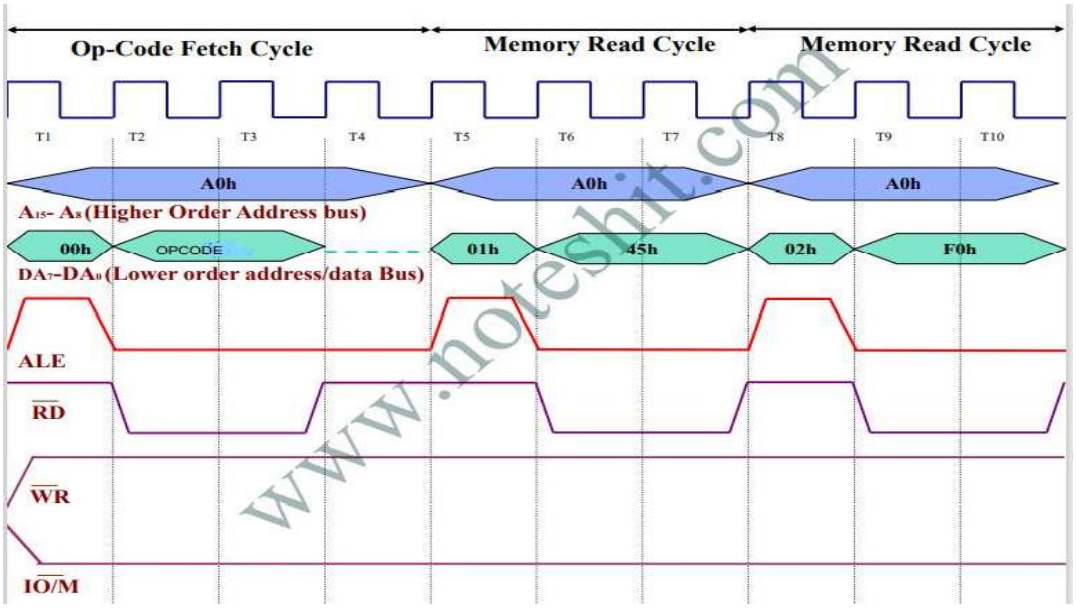
A000h LDA FO45h

Corresponding Coding:

A000h OPCODE

A001h 45

A002h F0



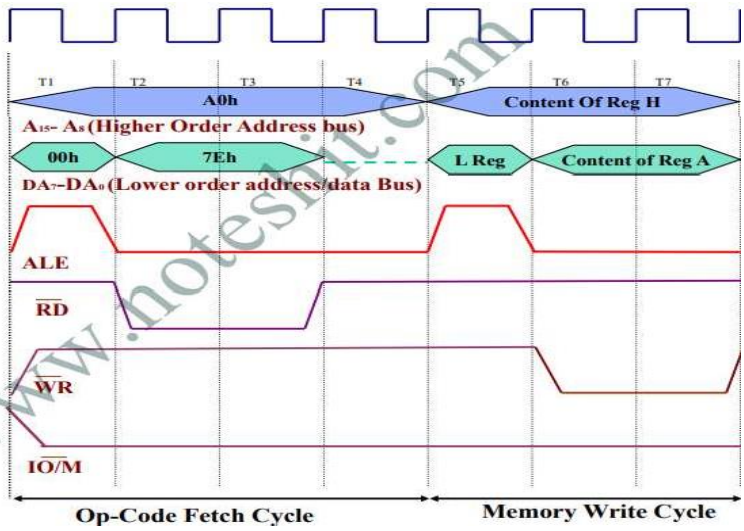
MOV M,A

Instruction:

A000h MOV M,A

Corresponding Coding:

A000h 77



Chapter-4

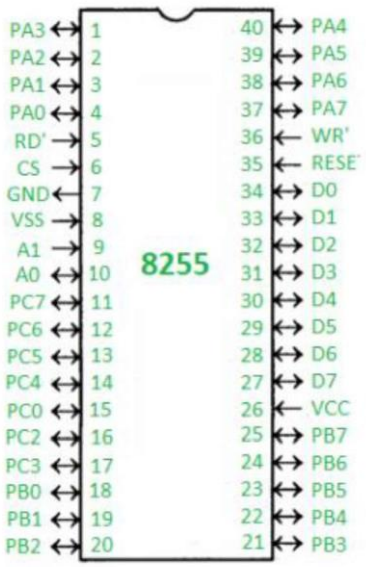
Microprocessor Based System Development Aids

Interfacing

Interfacing means using the processor for different task with a compatible device.

The interfacing circuit or chip can be programmed according requirement. For interfacing we use a chip called peripheral interface (PPI), intel 8255.

Pin configuration of Intel 8255



Data Bus Buffer

It is a tri-state 8-bit buffer, which is used to interface the microprocessor to the system data bus. Data is transmitted or received by the buffer as per the instructions by the CPU. Control words and status information is also transferred using this bus.

Read/Write Control Logic

This block is responsible for controlling the internal/external transfer of data/control/status word. It accepts the input from the CPU address and control buses, and in turn issues command to both the control groups.

CS

It stands for Chip Select. A LOW on this input selects the chip and enables the communication between the 8255A and the CPU. It is connected to the decoded address, and A_0 & A_1 are connected to the microprocessor address lines.

WR

It stands for write. This control signal enables the write operation. When this signal goes low, the microprocessor writes into a selected I/O port or control register.

RESET

This is an active high signal. It clears the control register and sets all ports in the input mode.

$\overline{RD}$-

It stands for Read. This control signal enables the Read operation. When the signal is low, the microprocessor reads the data from the selected I/O port of the 8255.

$\overline{WR}$-

When this pin goes low, the processor can write or store the data into the output port of PPI.

A_1, A_0

These are the address line and connected to the LSB of the address bus of processor.

RESET

When this is high, PPI is in reset condition and there is no data communication between ports.

Intel 8255 is used for parallel data transfer through data bus D_0-D_7

It has 3 ports Port A, Port B, Port C

Port C is divided into 2 sub group i.e., $PC_0 – PC_3$ (PC_{lower})

$PC_4 – PC_7$(PC_{upper})

All ports are of 8 bit. In total there are 4 ports which can be programmed according to the requirement as input and output port.

Block diagram of Intel 8255

The Intel 8255 is designed for use with Intel's 8 bit, 16 bit and higher capability of processor

It has 40 pins of 4 group.

8255 PPI chip is also called as parallel IO port chip.

All the port of 8255 can function independently as input or output by programming the controlled word register.

The block diagram of INTEL 8255 contains: a) Data bus buffer

b) Read/write control logic

c) Group A, and group B control

d) port A, port B, port C

a. **Data BUS Buffer:**

I. It is a 8 bit bidirectional data bus buffer.

II. It is used to interface the 8255 data bus with the system data bus.

III. It is outer pin are D_0- D7 and it is connected to system data bus.

IV. The direction of the data bus is to be decided by the read/ write control signal.

V. In read operation it transmits data to the system data bus. In write operation it receives data from the system data bus.

b. Read/write control logic:

a) This block function is to accept inputs from system control bus and address bus.

b) Here control signals like read, write chip select and address signals A_0, A_1 are used.

c) Among these 5 signals, Read and Write are connected to the address lines A_0 and A_1 of the system address lines.

d) The selection of 8255 is enabled or disabled by chip select. If it is '0' the 8255 is selected and if it is '1', the 8255 is rejected.

C**. Group A and Group B control**

a) 8255 is divided into 2 groups i.e., Group A and Group B control.

b) Group A consists of port A and Port C_{upper} and group B consists of port B and Port C_{lower}.

c) Each group consists of 12 pins.

d) The selections are done by mode operation.

Mode 0 = simple I/O (Port A, B,C)

Mode 1 = Strobed or handshake I/O (Port A, B)

Mode 2 = Bideirectional port(Port C)

d. Port A, Port B, and Port C

1. The ports of 8255 PPI and Port A, Port B, Port C each port consists of an 8-bit data input buffer.

2. The function of port A, port Band Port C are decided by control bit pattern of CWR.

3. The port C divided into two groups PC_{upper} and PC_{lower}.

4. Port C pin can be used as simple I/O handshake signal and stauts signal.

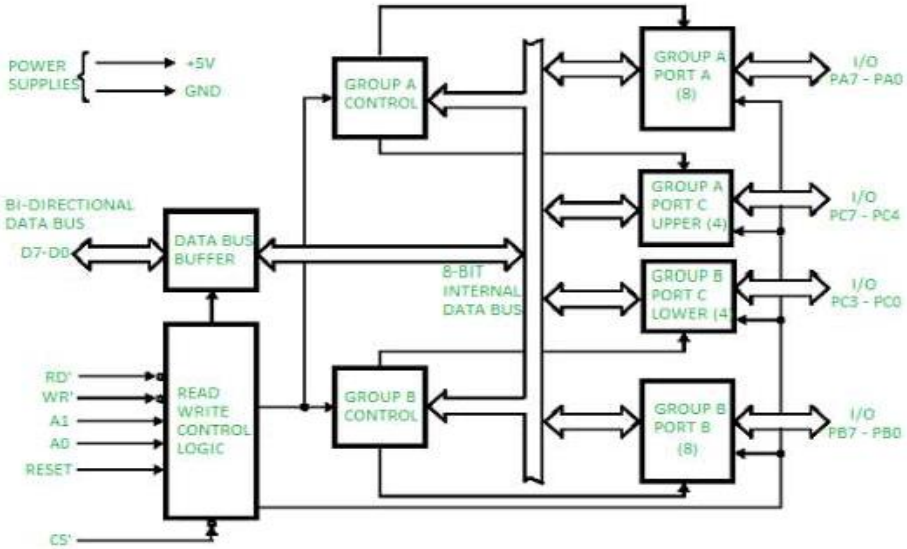
5. It is used for coordinate between port A port B.

Truth Table of INTEL 8255

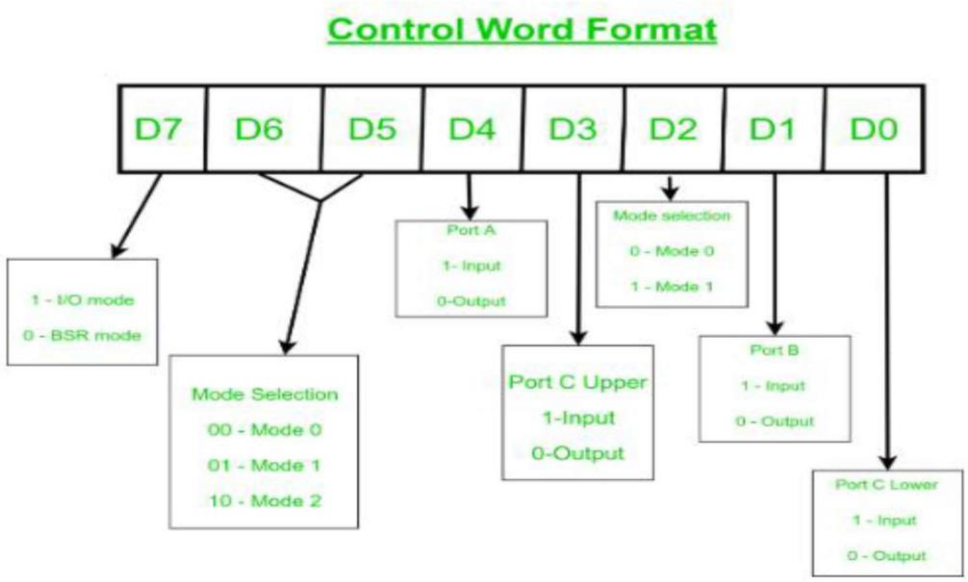
A_1	A_0	Port/Register selection
0	0	Port A

0	1	Port B
1	0	Port C
1	1	CWR

Block Diagram of Intel 8255



Control Word of INTEL 8255



Q. Make a control word when the ports of INTEL are defined as follow: -

Port A = input port

Port B = output port

Mode of port A = mode 0

Mode of port B = mode 0

Port C_{uuper} = input port

Port C_{lower} = Output port

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	**0**	**0**	**1**	**1**	**0**	**0**	**0**

Ans= 98_H

Q. Make a control word of Intel 8255 for mode 0 operation and port B, Port C_{upper} port C_{lower} all are output port.

Direct Memory Access (DMA)

In a computer system data communication take place between I/O device and CPU and memory.

The technology involved with CPU and memory is same i.e., semiconductor technology.

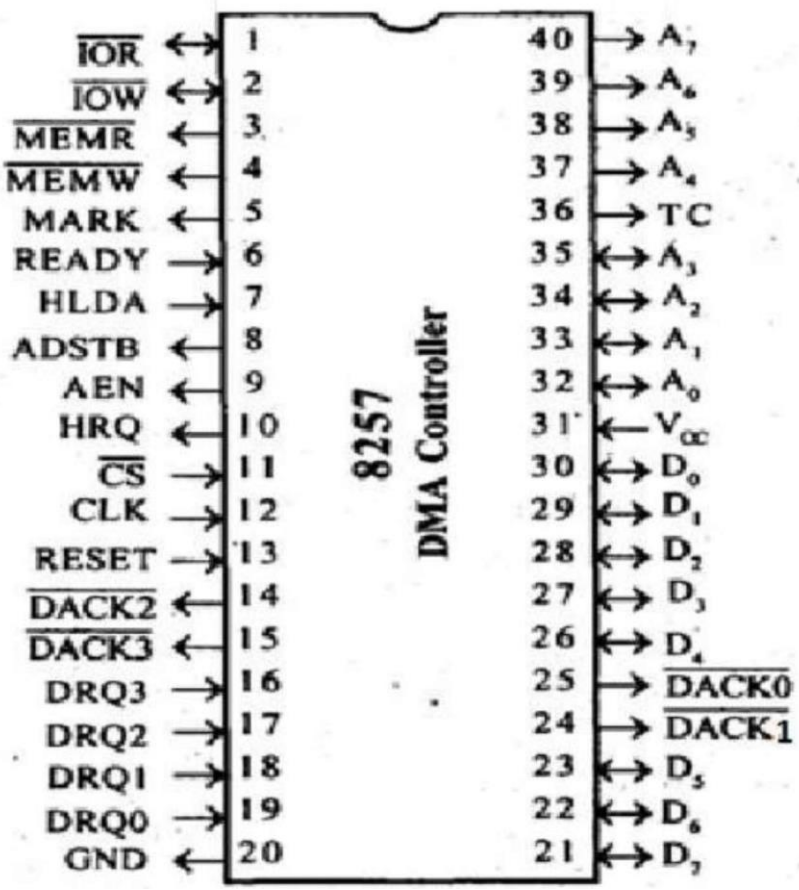
So, the speed between CPU and memory matches.

The I/O devices don't use semiconductor technology. Hence the speed of memory and I/O devices the DMA technology is used.

(The chip used for this technology is INTEL 8257)

The CPU does not interfere in DMA technique but it has to participate in the process through buses.

Pin configuration of INTEL 8257



1. DMA channel

There are 4 channels in INTEL 8257: channel 0

Channel 1

Channel 2

Channel 3

These channels are connected to different I/O devices. Each channel has 2 registers

- DMA address Register
- Terminal Count or TC register

2. DMA request (DRQ_0- DRQ_3)

DMA request starts from DRQ_0 and ends with DRQ_3. That means at a time 4 peripheral devices can

send DMA request to INTEL 8257.

DRQ_0 has lowest priority that means that line held high until it receives an acknowledgement from 8257.

3. DMA Acknowledgement:

A peripheral device when gives a DMA request, the DMA controller or 8257 sends acknowledgement signal through DACK signal. As there are 4 DRQ lines so there are 4 DACK line.

4. Data Bus Buffer:

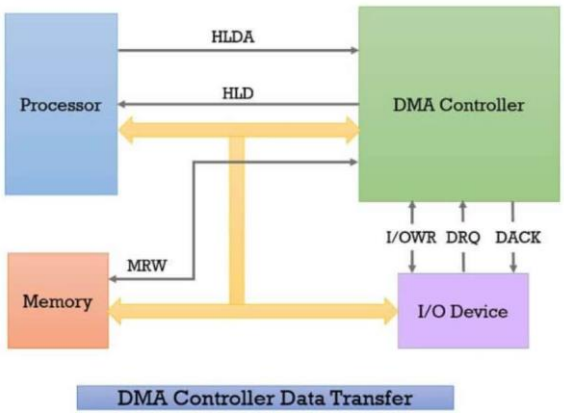
This is a 8 bit bidirectional buffer used to interface the 8257 with the processor bus.

When the CPU has the control over the bus then that is called "slave mode".

IOW means the pin is used to write the data into the output device.

IOR means the pin is used to read the data into the output device.

DMA Operation



An I/O device sends request for data transfer through DRQ lines (DMA request).

When 8257 receives DRQ from an I/O device it issues a HOLD request line to the CPU.

When the processor gets HRQ request, it acknowledges through HLDA line.

After receiving HLDA for processor, Intel sends $\overline{DACK}$ to the I/O devices.

Now, the address is sent to the address bus and data is sent through data bus.

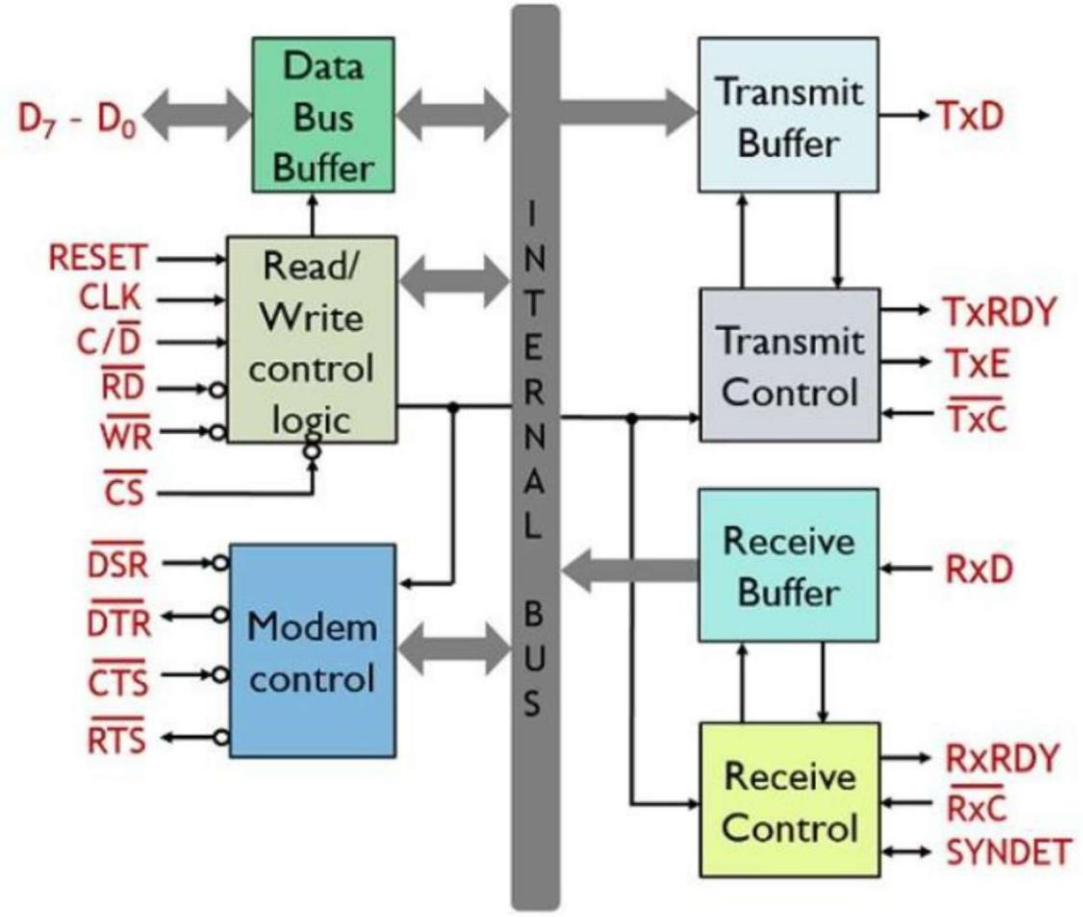
Ready pin is used to show that I/O devices are ready for data transfer. The data transfer continue till the ready pin remains high.

$\overline{MEMR}, \overline{MEMW}, \overline{IOR}, \overline{IOW}$ are the control signals used for data flow between memory and I/O devices.

Universal Synchronous Asynchronous receiver and Transmitter (USART), INTEL 8051

8251 is programmed to operate in any of the serial communication with microprocessor.

This chip converts parallel data into serial stream of bits and converts it into parallel byte for the microprocessor.



Architecture of 8251 USART

$\overline{DSR}$- Data set Ready

$\overline{DTR}$- Data terminal Ready

$\overline{CTS}$- Clear to send

$\overline{RTS}$- Request to send

SYNDET **–** Synchronous Detect

BRKDET – Break Detect

TXD **–** Transmit Data output

TXC – Transmitter clock input

TXRDY- Transmitter Ready

TXEMPTY – Transmitter Empty/ TxE

RXD – Receiver data input

Operation:

The data bus buffer interface the internal bus with the system bus (processor bus)

The read/ write control logic controls the operation of the peripheral depending on the operation of the processor. This unit is also responsible to select one of the two internal addresses like control address and data address.

The modem control unit transmits the data byte received by the data bus buffer from processor for serial communication. This decides the transmission rate by that signal $\overline{\overline{TX}_C}$.

The receiver control unit decides the receiver frequency control by $\overline{RX}_C$. This unit generates RxRDY signal that may be used by the processor for the data communication.

Pin configuration of INTEL 8251:

D_0-D_7 It is a 8 bit data bus which is connected to the data bus of processor.

C/$\overline{D}$- It is a control or data address. If it is high, then the address is control address and if it is low, then it is for data address.

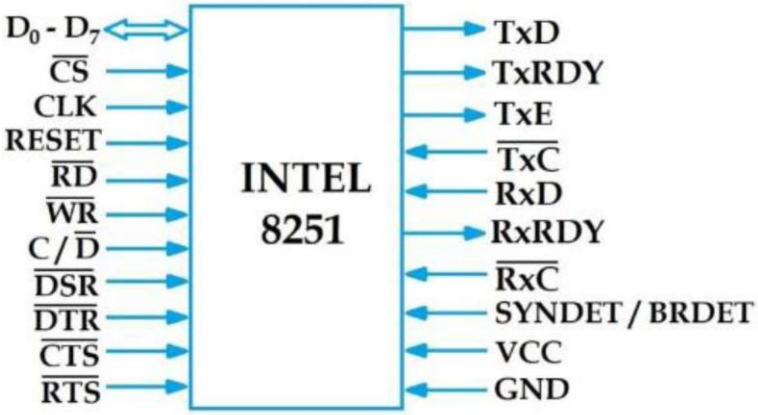
$\overline{RD}$- When it is low, then CPU reads data from the internal resisters of 8251.

$\overline{WR}$- When it is low, then CPU writes data onto 825.

CLK- It is used to generate internal device time; it should be at least 30 times greater than transmission and reception frequency.

RESET- When this is high, the 8251 is ideal.

$\overline{TXc}$- It is a transmitter clock input. It controls the rate(speed) of data transmission.



TXD- It is a transmitter data output. It carries serial stream of transmitted data.

RXRDY-It is a receiver ready signal. It gives the signal to know whether the receiver is ready to transfer the data or not.

$\overline{DSR}$- It is a data set Ready signal. When it is low the internal data bus is set.

$\overline{RTS}$ – It is request to send Data.

$\overline{CTS}$- It is clear to send Data.

R_XC- It is the receiver clk signal, that controls data rate of receiver.

TXE- It is the transmitter Empty signal. If it is high then the transmitter has has already sent the data and there is no data left.

$\overline{DTR}$- It is the Data Terminal Ready signal.

Analog to Digital Conversion

The A/D converter is used to convert analog signal to digital signal.

The most important method to convert from analog to digital is successive approximation method.

In this method the unknown analog voltage V_{in} is convert to digital by comparing it with the reference voltage V_r.

For an 'n' bit digital input, the comparison with the reference voltage is done 'n' times.

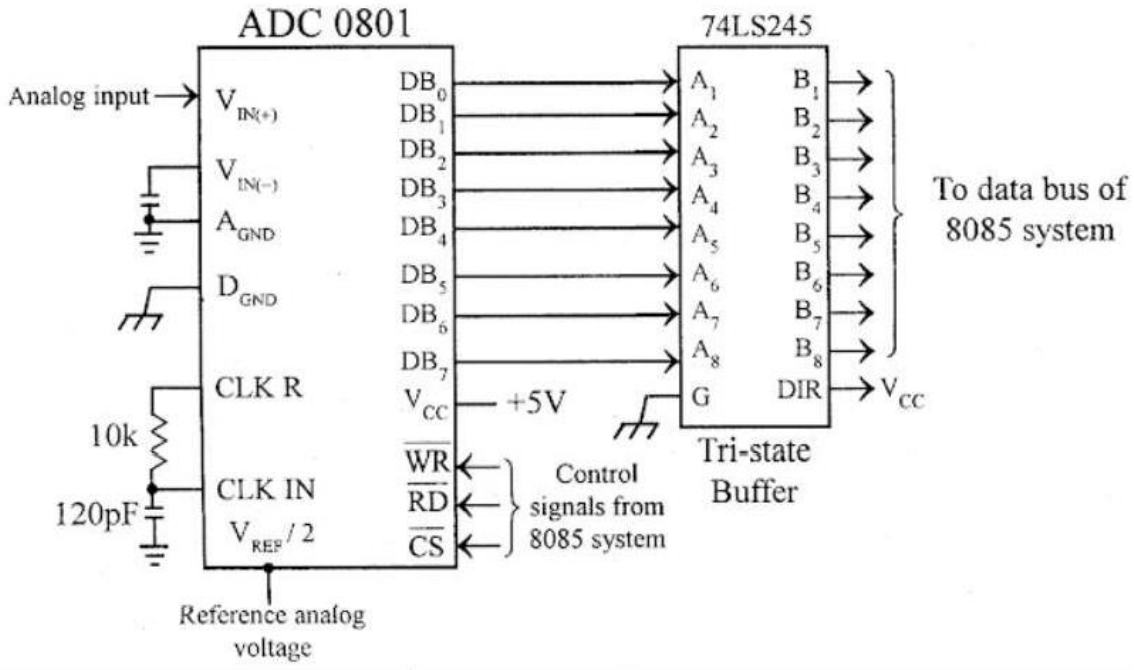
If $V_{in} > V_r$ or $V_{in} >$ fraction of V_r then the digital bit is set to 1.

If $V_{in} < V_r$ then the digital bit is set too.

Thus, the formula for conversion is,

$$\sum_{n=1}^{n} [b_i 2^{-i}] V_r$$

b_i = constant



Interfacing of Analog to Digital Converter with Intel 8085

Chapter-5

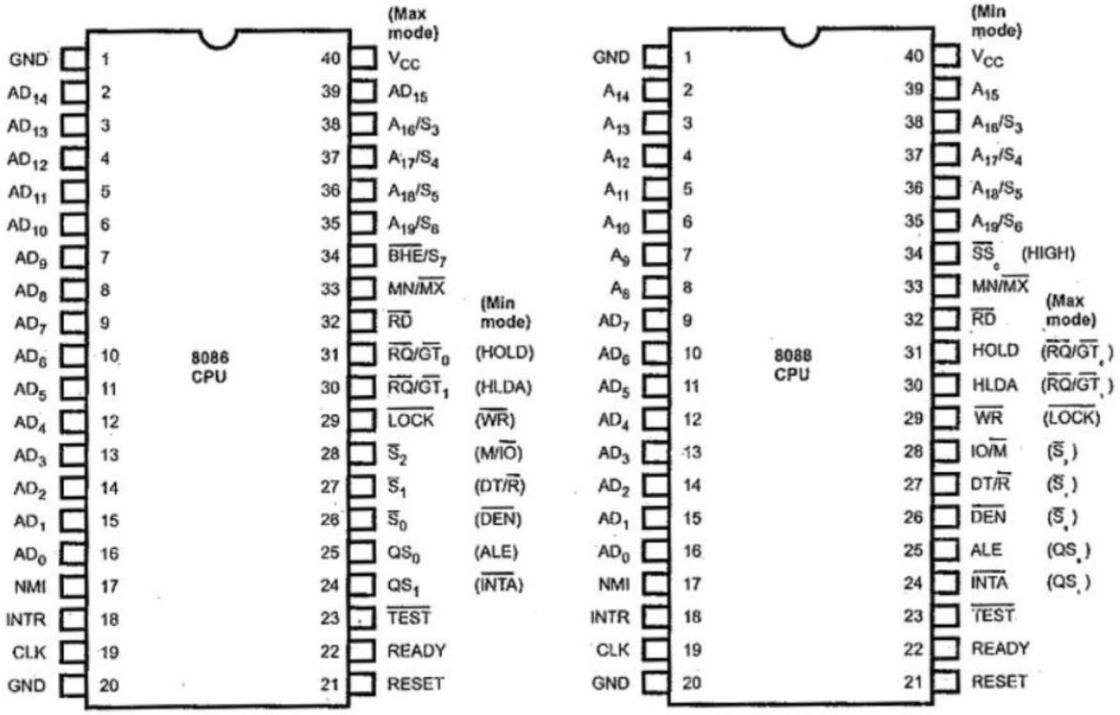
Microprocessor (Architecture and Programming-16 bit-8086)

Special features of Intel 8086:

- ✓ It is a 16-bit microprocessor.
- ✓ It is a 40 pin DIP.
- ✓ It has three clock pulses like 5 MHz, 8 MHz, and 10 GHz.
- ✓ It has 20-line address bus i.e., A_0-A_{19}.
- ✓ It has a 16-bit wide data bus i.e., D_0-D_7.

Pin Configuration of Intel 8086:

1. Minimum Mode Configuration of 8086
2. Maximum Mode Configuration of 8086



AD_0-AD_7 –

- ✓ These are the multiplexed address and data line.

- ✓ The address bus is of 20 bits wide i.e., A_0-A_{19}.
- ✓ It can be grouped as lower order address line(A_o-A_{15}) & higher order address line(A_{16}-A_{19}).
- ✓ Data bus is 16 bits wide i.e., D_0-D_{15} to make AD_0-AD_{15}.
- ✓ These are bidirectional.

NMI:

It stands for non-maskable interrupt.

It is an interrupt request line i.e., unavoidable by the processor.

INTR:

This is also an interrupt request signal that can be avoided by the processor.

RESET:

When this pin is high the processor is in reset condition.

Ready: When it is high the data transfer between processor and peripheral takes place.

CLK:

It provides clock frequency for the processor.

V_{cc}:

It is +5-volt supply for the processor.

GND

Ground connection or negative polarity.

$\overline{TEST}$-

When it is low, microprocessor continues execution, otherwise it waits in an idle state.

***$\overline{BHE}$*/S7**

- ✓ Bus High Enable with the status sign.
- ✓ It is used to enable the MSB of the data bus(D_8-D_{15}) to be transferred with the bus and it is multiplexed with the status signal S_7.

MN/*$\overline{MX}$*

- As we know, INTEL 8086 can operate in two modes i.e., minimum mode and maximum mode.
- When **MN/$\overline{MX}$** is high, the 8086 is operated in minimum mode.
- When **MN/$\overline{MX}$** goes low the 8086 is in maximum mode.

Pin no 24-31 with minimum mode

$\overline{INTA}$- It is an interrupt acknowledgement signal.

ALE- This stand for address latch enable. It is used to latch the address line into 8282 latch.

$\overline{DEN}$- It stands for Data Enable. When it goes low the data from data bus buffer goes to peripherals.

DT/$\overline{R}$-

It stands for Data transmit &receive.

When it goes high the data is transmitted and when it goes low the data is received from peripheral.

M/$\overline{IO}$ –

When it is high, the data is for memory and when it is low, the data is for IO device.

$\overline{WR}$-

When it goes low, the processor writes the data into the memory.

HLDA-

It stands for Hold Acknowledgement.

HOLD- It is HOLD request line**.**

Pin no 24-31 with minimum mode

QS_1, QS_0

QS_1	**QS_0**	**Instruction**
0	0	No operation
0	1	1st byte opcode from queue
1	0	Empty the queue

1	1	Subsequent byte from the queue

$\overline{S_0 S_1 S_2}$-

These are status signal connected to the bus controller.

$\overline{LOCK}$-

It is an active low pin or signal when it goes low all the interrupt are masked and no HOLD request will be received.

$\overline{RQ/GT_1}$,$\overline{RQ/GT_0}$ –

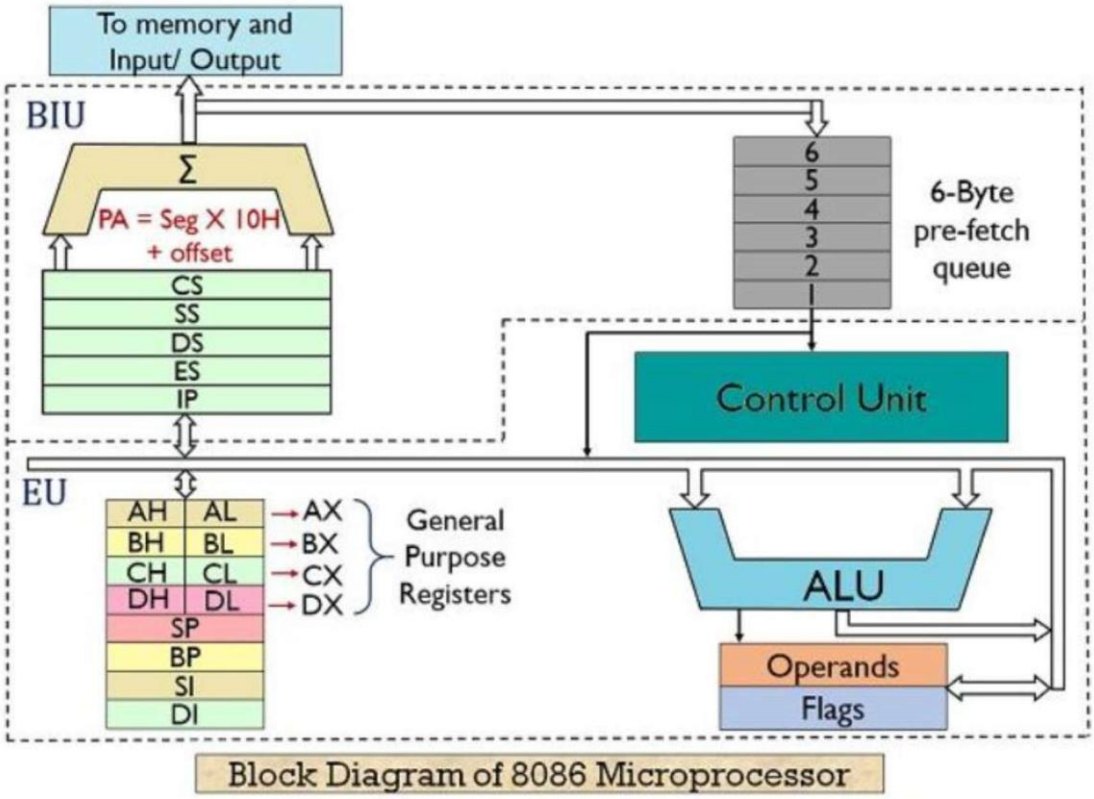
These are bidirectional line and are request and Grant command.

Priority of $\overline{RQ_0}$is more than $\overline{RQ_1}$.

Advantages of INTEL 8086 over INTEL 8085

- 8086 can operate in two modes
  a- Minimum mode
  b- Maximum mode
- Intel 8086 has pipe line architecture.
- 8086 has segmented memory.

Block Diagram of Intel 8086:



Block Diagram of 8086 Microprocessor

GPR:

Intel 8086 microprocessor is divided into two units

1- Bus Interface Unit
2- Execution Unit

BIU:

It contains segment registers i.e., SS, CS, ES, and DS

SS- Stack Segment

CS- Code Segment

ES- Extra Segment

DS- Data Segment

IP- Instruction pointer

- It also contains instruction queue and address conversion mechanism for physical address calculation.
- It is responsible for all external bus operation.
- It sends address by address bus , fetches the instruction, reads the data from memory and I/O devices, also write the data into the memory and I/O device.
- Responsible for generating 20 bit address and in communication the data with the data bus.

EU:

- ✓ EU consists of ALU, GPRs, instruction decoder, timing and control unit.
- ✓ This unit decodes the instruction opcode from BIU and executes it.
- ✓ The 16-bit ALU does all the arithmetic and logical operation.
- ✓ The 16 bit flag register show the result of execution.
- ✓ The timing and control unit provides the necessary control signal fir the execution.

Pipelining:

AS the BIU and EU work independently, so the operation work becomes faster, this is called pipelining. In this method, fetching and execution are done simultaneously.

Register Organisation of INTEL 8086

1. General purpose Register
2. Segment Register
3. Pointer and Index register
4. FLAG Register
5. Instruction Register

GPR:

There are four GPR in INTEL 8086 such as,

AX, BX, CX and DX

All these are 16 bit registers.

These 16 bit registers can be divided into lower and higher order group like:

AX- AH &AL

BX- BH & BL

CX- CH & CL

DX- DH & DL

H- Higher order

L- Lower Order

Segment Registers:

- ✓ Segment means logical division of memory.
- ✓ The memory connected to intel 8086 is segment into four groups, i.e., code segment, data segment, extra segment and stack segment.
- ✓ So, the register which relate to this segment have also the same name like

a- Code segment Register (CSR)
It holds the instruction code of a program.

b- Data Segment Register (DSR)
It holds the constant and variable data of the program.

c- Extra Segment Register (ESR)
It holds the destination address of data.

d- Stack Segment Register (SSR)
It holds the address and data of subroutine.

Pointer and index register:

SP
BP
SI
DI

Stack pointer holds the address of stack top. All the register are used for memory address computation.

Instruction pointer:

It does the same work as program counter in Intel 8085.

Flag Register:

X	X	X	X	O	D	I	T	S	Z	X	Ac	X	P	X	Cy

Cy to S is same as Intel 8085

T- Trap

If it is high the processor enter single step execution.

I-Interrupt

If it is high, the CPO recognizes the maskable interrupt but if it is low the interrupt are avoided.

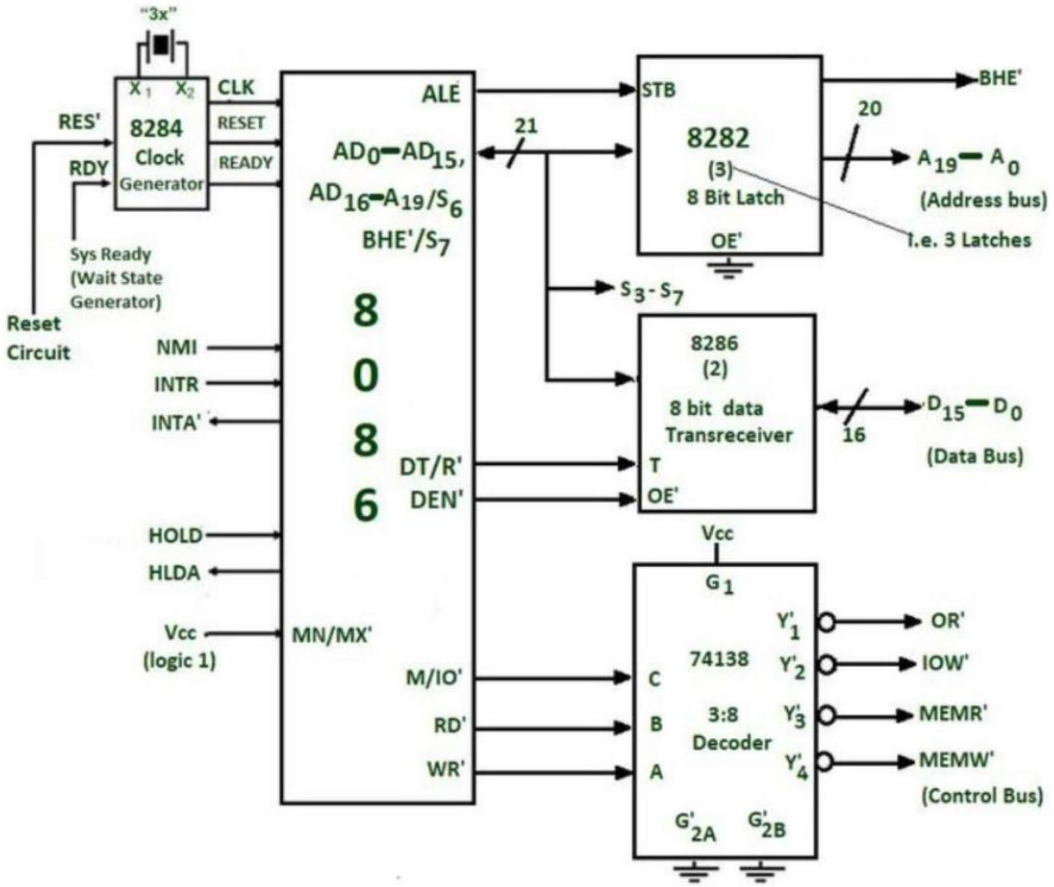
D- Direction

If it is high, string manipulation is carried out.

O-Overflow

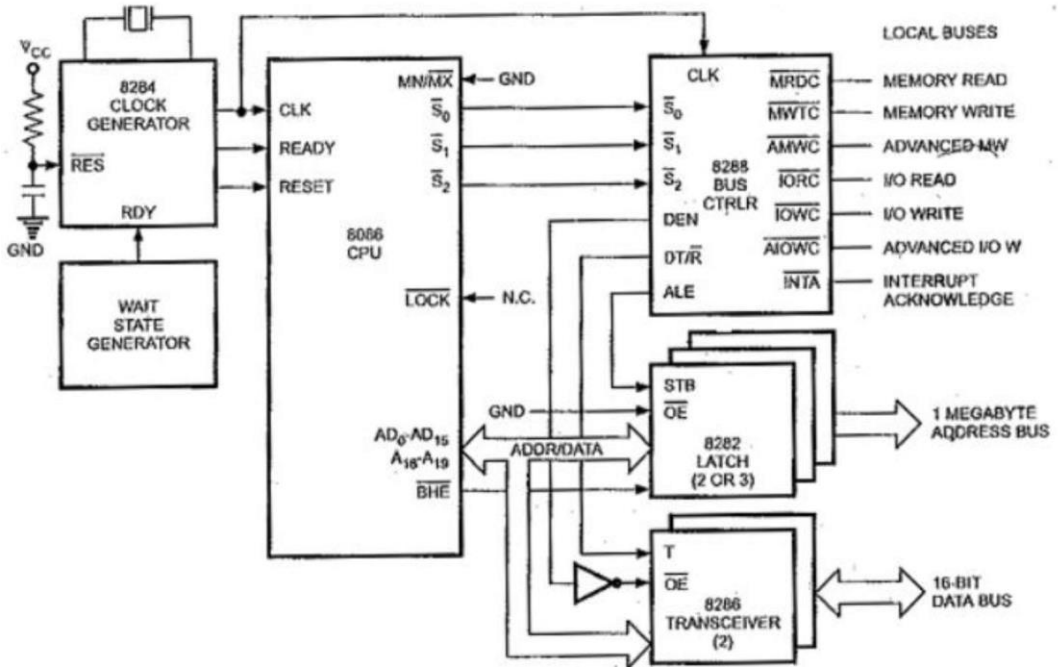
If there is a bit overflow condition, in case of computation, the overflow flag is set otherwise reset.

Intel 8086 in minimum mode: -



- ✓ 8086 is operated in minimum mode by strapping its MN/$\overline{MX}$ to logic 1
- ✓ In this mode a single processor is used. So, all the control signal are given by processor itself.
- ✓ Some extra components are latches, transceivers, clock generator, I/O device, and memory.
- ✓ The latches are D- flip flop and used for separating the data and address signal.
- ✓ The transceivers are to amplify the data and is connected by two signals DT/$\bar{R}$ and $\overline{DEN}$.
- ✓ $\overline{DEN}$ informs whether the data in the data bus is available or not.

Intel 8086 in maximum mode: -



- ✓ 8086 is operated in maximum mode by strapping MN/$\overline{MX}$ to logic 0.
- ✓ In maximum mode, the architecture contains same blocks as in minimum mode with an extra bus controller Intel 8288.
- ✓ As in maximum mode, these are more than one processor connected in cascade, so to have control over the buses, the bus controller is used.
- ✓ Here the control signals $\overline{DEN}$, DT/$\overline{R}$ are issued by Intel 8288.
- ✓ In this mode, the processor gives the status signals $\overline{S_o}$,$\overline{S_1}$, $\overline{S_2}$ to the bus controller.

Addressing modes of intel 8086

The way in which an operand is specified for an instruction, is called the addressing mode.

Intel 8086 has 8 addressing modes,

1. Register addressing mode
2. Direct addressing mode
3. Register addressing mode
4. Immediate addressing mode
5. Indexed addressing mode
6. Based addressing mode
7. Based indexed addressing mode
8. Based indexed with displacement addressing mode

Register addressing:

In register addressing the operand is one of the 16 bit or 8 bit general purpose register.

Ex: MOV AX, CX

ADD AL, BL

Immediate addressing:

Here the operand is specified in the instruction itself.

Ex: MOV AL, 35H

MOV BX, 0301H

ADD AX, 4836H

Direct addressing:

In direct addressing mode, the operand's offset is given in the instruction as an 8 bit or 16-bit displacement element.

Ex: ADD AL, [0301 ----- Displacement

This instruction adds the content of the offset address 0301 to AL. The operand is placed at the given offset o301 within the data segment DS.

Register Indirect addressing:

The operand's offset is placed in any one of the registers BX, BP, SI, or DI are specified in the instruction.

Indexed addressing:

The operand's offset is the sum of the content of an index register SI or DI and an 8 bit or 16-bit displacement.

Offset= [SI or DI+ 8 bit or 16-bit displacement]

Ex: MOV AX, [SI+05]; 8-bit displacement

MOV AX, [SI+1250]; 16-bit displacement

Based addressing:

The operand's offset is the sum of the 8 bit or 16-bit displacement and the contents of the base register BX or BP. BX is used as a base register for data segment and BP is used as a base register for stack segment.

Offset = [BX + 8 bit or 16-bit displacement]

Ex: MOV AL, [BX+05] 8-bit displacement

Suppose, the register BX contains 0301, the offset will be 0301+05=0306

The content of the memory location 0306 will move to AL.

MOV AL, [BX+1346] is an example of 16-bit displacement.

Based indexed addressing:

The operand's offset is the sum of the content of a base register BX or BP and an index register SI or DI.

Offset = [BX or BP] + [SI or DI]

Ex: ADD AX, [BX+SI]

MOV CX, [BX+SI]

Based indexed with displacement

In this mode of addressing, the operand's offset is given by

Offset = [BX or BP] + [SI or DI] 8bit or 16-bit displacement.

Ex: MOV AX, [BX + SI + 05]

MOV AX, [BX+ SI + 1247H] 16-bit displacement

Classification of instructions of INTEL 8086:

Instructions are classified as per their basic function perform. These are categorized as: -

1. Data transfer instruction
2. Arithmetic instruction
3. Logical instruction
4. Branch instruction
5. Loop instruction
6. String instruction

References:

1. **Microprocessor Architecture, programming and programming with application the 8085, Ramesh Gaonkar, Pearson Prentice Hall, 2002.**

2. **N. Senthil Kumar, M. Saravanan, and S. Jeevananthan. 2011. Microprocessors and Microcontrollers. Oxford University Press, Inc., USA.**

3. **Fundamentals of microprocessor and microcontroller, B Ram, 2008**