Lecture-1

An Overview of Microprocessor

The first question that comes in one’s mind is "What is a microprocessor?”. Let us start with a more familiar term computer. A digital computer is an electronic machine capable of quickly performing a wide variety of tasks. They can be used to compile, correlate, sort, merge and store data as well as perform complex calculations at much faster rate than human being by means of stored instructions.

A digital computer is different from a general purpose calculator in a sense that digital computer is capable of operating according to the instructions that are stored within the computer whereas a calculator must be given instructions on a step by step basis to perform calculations. By this definition a programmable calculator can be considered a computer.

Historically, digital computers have been categorized according to the size using the words large, medium, minicomputer and microcomputer. In the early years of development, the emphasis was on large and more powerful computers. Large and medium sized computers were designed to solve complex scientific and engineering problems. In early stage of development these computers were accessible and affordable only to large corporations, big universities and government agencies. Later on, minicomputers were made available for use in office, small collage, medium size business organization, small factory etc. As the technology has advanced from SSI to VLSI & SLSI, the face of the computer has changed gradually and it became possible to build the entire central processing unit (CPU) on a single-chip known as microprocessor.

A control processing unit (CPU) with its related timing functions on a single chip known as microprocessor. A microprocessor combined with memory and input/output devices forms a microcomputer.

The microcomputer is making an impact on every activity of mankind. It is being used in almost all control applications. For example analytical and scientific instruments, data communication, character recognition, musical instruments, household items, defence equipments, medical equipments etc.

Microcomputers or, in general, computers communicate and operate in binary numbers ‘0’ and ‘1’ also known as bits. A bit is the abbreviation for the term **b**inary dig**it**. The bit size of a microprocessor refers to the number of bit which can be processed simultaneously by the arithmetic circuit of the microprocessor. A number of bits taken as a group in this manner is called word. For example, the first commercial microprocessor the Intel 4004 which was introduced in 1971 is a 4-bit machine and is said to process a 4-bit word. A 4-bit word is commonly known as nibble and an 8-bit word is commonly known as byte. Intel 8085A is an 8-bit microprocessor whereas Intel 8086 is a 16-bit microprocessor. It should be noted that a processor can perform calculations involving more than its bit size but through program and takes more time to complete the operation. For example, multi-byte data can be added byte by byte in 8085A processor which is an 8-bit processor. The short word length requires few circuitry and interconnection in the CPU.

Microcomputers:

In a very general sense, a microcomputer is best regarded as a system incorporating a CPU and associated hardware whose purpose is to manipulate data in some fashion. This is exactly what any digital circuit designed using SSI's and MSI's does. Therefore, microcomputer should be regard as a general purpose logic device. In contrast to standard SSI's and MSI's where the manufacturer decides what the device will do, with microcomputer it is the user who decides what the device should do by asking it to execute a proper set of instructions. A microcomputer, from this point of view is merely an assembly of devices whose sole task is to ensure that the instruction desired are indeed carried out properly and to allow the microprocessor to communicate with the real world, i.e. the user environment. The power of the microcomputer lies in the fact that if the application changes, the same system can be used by appropriately modifying the instructions to be executed and, if necessary, some changes in the hardware. In contrast, a digit circuit designed using SSI's and MSI's for some application will need to be completely redesigned if the application changes significantly.

The objective of a microcomputer (μc) is to manipulate data in a certain fashion specified by the system designer. A typical microcomputer achieves their objective by getting its CPU to execute a number of instructions in the proper sequence. This sequence of instruction comprises the program that is executed by the microcomputer. A microcomputer which does nothing other than manipulate data present within itself, will not be of much use to anyone. In order to do something meaningful, data being manipulated should depend on inputs provided to the microprocessor by the user. Similarly, the data manipulations being carried out by the microprocessor would be completely meaningless unless the results of these manipulations affect things outside the microcomputer itself. Therefore, the µc should provide outputs which in some way depend on its inputs, the way inputs and outputs are related is decided by the program that gets executed.

Therefore, a microcomputer is an assembly of devices including a CPU, which manipulate data depending on one or more inputs and according to a program, in order to generate one or more output.

Microcontrollers

A µP does not have enough memory for program and data storage, neither does it has any input and output devices. Thus when a µP is used to design a system, several other chips, such as memory chips and input/output ports, are also used to make up a complete microcomputer system. For many applications, these extra chips imply additional cost and increased size of the product and may not be suitable for the application. For example, when used inside a toy, a designer would like to minimize the size and cost of the electronic equipment inside the toy. Therefore, in such applications a microcontroller is used more often than a microprocessor.

A microcontroller is a chip consisting of a microprocessor, memory and input/output ports.

Evolution of the Microprocessors

The first µP was introduced in 1971 by Intel Corporation. This was the Intel 4004, a processor on a single chip. It had the capability of performing simple arithmetic and logical operations. For example, addition, subtraction, comparison, logical AND and OR operations. It also had a control unit which could perform various control functions like fetching an instruction from the memory, decoding it and generating control signals to execute it. It was a 4 bit µP operating on 4 bits of data at a time. The processor was the central component in the chip set, which was called the MCS-4. The other components in the set were a 4001 ROM, 4002 ROM and a 4003 shift register.

Shortly after the 4004 appeared in the commercial market place, three other general purpose microprocessors were introduced. These devices were the Rockwell International 4-bit PPS-4, the Intel 8-bit 8008 and the National Semiconductor 16-bit IMP-16. Other companies had also contributed in the development of µP.

The first 8 bit µP, which would perform arithmetic and logic operations on 8 bit words, was introduced in 1973, by Intel. This was 8008 that was followed by an improved version- the 8080 from the same company. The µPs introduced between 1971and 1972 were the first generation systems. They were designed using the PMOS technology. This technology provided low cost, slow speed and low output currents and was compatible with TTL.

After 1973, the second generation µPs such as Motorola 6800 and 6809, Intel 8085 and Zilog Z80 evolved. These µPs were fabricated using NMOS technology. The NMOS process offered faster speed and higher density than PMOS and was TTL compatible.

The distinction between the 1st & 2nd generation devices was primarily the use of new a semiconductor technology to fabricate the chips. This new technology resulted in a significant increase in instruction execution speed & higher chip densities.

After 1978, the 3rd generation microprocessors were introduced. Typical µPs were Intel 8086/ 80186/ 80286 and Motorola 68000/ 68010. These µPs were designed using HMOS technology. HMOS provides the following advantages over NMOS.

1) Speed power produced (SSP) of HMOS is 4 times better than that of NMOS. That is for NMOS, SSP is 4 picojoules (PJ) and for HMOS, SSP is 1 picojoules (PJ).

   Speed power product = speed * power
   = nanoseconds * mill watt
   = picojoules

2) Circuit densities provided by HMOS are approximately twice those of NMOS. That is for NMOS, it is 4128 µm^2/gate and for HMOS it is 1052.5 µm^2/gate.

Later, Intel introduced a high speed version of the 8085A called 8085AH using HMOS technology to fabricate the 8085A.

The third generation introduced in 1978 was typically separated by the Intel 8086 iAPX8086, iAPX80186, iAPX80286, Zilog 8000, and the Motorola 68000 which are 16- bit µPs with minicomputer like performance. One of the most popular 16-bit µP introduced by Intel was 8088. The 8088 has the same introduction set as the 8086. However, it has only an 8 bit data bus. The 8088 is the µP used in the

IBM PC and its clones. A precursor to these microprocessors was the 16-bit Texas Instruments 9900 microprocessor introduced in 1976.

Table: Evaluation of major µP characteristics from Intel

	4004	**8008**	**8085A**	**8086**	**80386**
Year of Introduction	1971	1971	1977	1978	1985
Data Bus	4-bit	8-bit	8-bit	16-bit	32-bit
Technology	PMOS	PMOS	NMOS	HMOS	CHMOS
Word size data/ instr.	4/8	8/8	8/8	16/16	32/32
Address capacity	4K	16K	64K	1M	4G
Clock kHz/phase	740/2	800/2	6250/2	8000/2	16000/2
Addition time	10.8 µs	20 µs	1.3 µs	0.375 µs	0.125 µs
ALU/General Purpose Reg.	1/16	1/6	1/6	1/8	1/8
Stack size	3x12	7x14	RWM	RWM	RWM
Voltages	15-10,5*	-9.5v	+5V	+5V	+5V
Package size	16pin	18pin	40pin	40pin	132pin
Instructions	45	48	74	133	135
Transistors	2,300	2,000	6,200	29,000	2,75,000
Chip size(mil)	117x159	125x170	164x222	225x230	390x390

In 1980, the fourth generation µPs were evolved. Intel introduced the first commercial 32 bit microprocessor, Intel 432. Since 1985, more 32-bit µPs have been introduced. These include

Intel iAPX80386, Intel 80486, Motorola MC68020/68030/68040, National semiconductor NS 32032. These processors were fabricated using the low power version of HMOS technology called HCMOS, and they include an on-chip RAM called the cache memory to speed up program execution. The characteristics for few microprocessors introduced by Intel are given in the Table.

This shows that power of microprocessors has increased tremendously with advancement in integrated circuit technology & microprocessor systems architecture. Very large scale integration, VLSI, allows extremely complex system consisting of as many as a million of transistors on a single chip to be realized.

The performance offered by a 32-bit µP is more comparable to that of supercomputers such as VAX 11. Extensive research is being carried out for implementation of more on chip functions and for improvement of the speed of the memory and I/O devices.

Applications of Microprocessors:

The application of microprocessors is increasing day by day. Some of the applications are:

1) Analytical scientific instruments

2) Smart terminals

3) Stacker crane controls

4) Conveyor controls

5) Word processor

6) Point of scale systems

7) Standalone electronics cash system

8) Electronic games

9) Vending and dispensing machines
10) Market scales
11) Traffic light controls
12) Home heating and lighting controls
13) Security & fire alarm system
14) Home appliances
15) Computer aided instruction
16) On line control of lab instrumentation
17) Desktop computers
18) Check processor
19) Payroll system
20) Inventory control
21) Automatic type setting
22) Compact business machines
23) Medical instrumentation
24) Automobile diagnostics
25) Data communication processing
26) Optical character recognition
27) I/O terminal for computers.

Lecture 2

Microcomputer Organization:

As discussed in previous lecture microprocessor is a central processing unit (CPU) with its related timing functions on a single chip. A microprocessor combined with memory and input/output devices forms a microcomputer. Therefore, the basic components of a microcomputer are:

1) CPU
2) Program memory
3) Data memory
4) Output ports
5) Input ports
6) Clock generator.

These components are shown in fig.1.1 below:

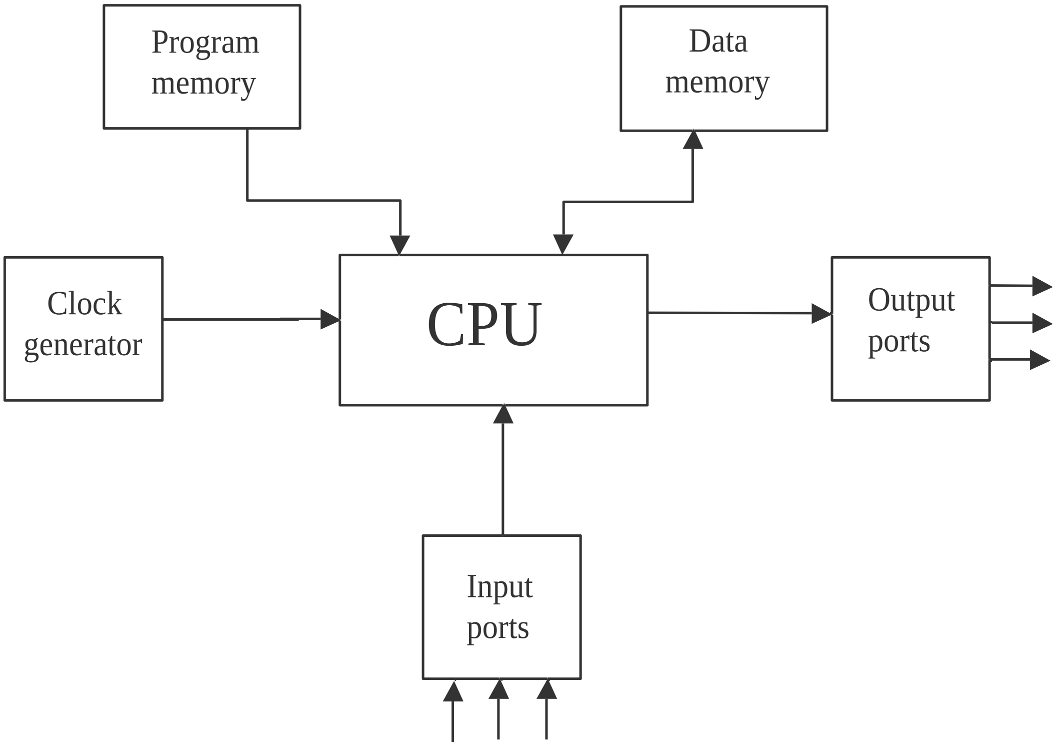


Fig.1.1 Basic Components of Microcomputer

Central Processing Unit:

The CPU consists of ALU (Arithmetic and Logic Unit), register unit and control unit. The CPU fetches the stored instructions from the program memory, data word from data memory or from an input device and after processing the data stores the result in data memory or sends it to an output device.

a) ALU (Arithmetic and Logic Unit)

This unit performs computing functions on m-bit data where ‘m’ is the bit size of the processor. These functions are arithmetic operations such as addition, subtraction and logical operation such as AND, OR, XOR, rotate, compare etc. Results are stored either in registers or in memory or sent to output devices.

b) Register Unit:

It contains various 8-bit or 16-bit registers. These registers are used primarily to store data temporarily during the execution of a program. Some of the registers are accessible to the user through instructions. It means there contents can be read and/or changed through instructions. Some of the registers are not accessible to user but they are used by the processor for the execution of an instruction. 8085A microprocessor contains 8-bit registers such as Accumulator (Reg. A), B, C, D, E, H, L etc and16-bit registers such as Program Counter (PC), Stack Pointer (SP).

c) Control Unit:

It provides necessary timing & control signals required for the operation of microcomputer. It controls the flow of data between the microprocessor and peripherals (input, output & memory). The control unit gets a clock signal which determines the speed of the microprocessor.

In all, the CPU has the following basic functions:

a. It fetches an instructions word stored in memory.
b. It decodes the instruction to determine what the instruction is telling it to do.
c. It executes the instruction. Executing the instruction may include same of the following major tasks:
    1. Transfer of data from one register to another register in the CPU itself.
    2. Transfer of data between a CPU register & specified memory location or input/output device.
    3. Performing arithmetic and logical operations on data from a specific memory location or a designated CPU register.
    4. Directing the CPU to change the sequence of fetching instructions, if processing the data created a specific condition.
    5. Performing housekeeping function within the CPU itself in order to establish desired condition at certain registers.
d. It looks for control signal such as interrupts and provides appropriate responses.
e. It provides status, control, and timing signals that the memory and input/output section can use.

Memory:

It stores both the instructions to be executed (i.e. program) and the data involved. It usually contains ROM (Read Only Memory) and RWM (Read Write Memory). The ROM can only read and cannot be written into and is non volatile that is, it retains its contents when the power is turned off. A ROM is typically used to store instructions and data that do not change. For example, it stores the monitor program of a microcomputer.

One can either read from or write into a RWM in memory read operation or memory write operation respectively. The RWM is volatile, that is it does not retain its contents when the power is turned off. It is used to store user programmes & data which are temporary might change during the course of executing a program.

Both ROM & RWM are RAMs (Random Access Memory). During a memory read operation, the content of the addressed location is not destroyed. During a write operation, the original content of the addressed location is destroyed.

Program Memory:

The basic task of a microcomputer system is to ensure that its CPU executes the desired instructions sequence i.e., the program properly. The instructions sequence is stored in the program memory. On initialization- usually on power up or manual reset the processor starts executing the instructions from a predetermined location in program memory. The first instruction of the program should, therefore, be in this location. In typical processor based system, the program to be executed is fixed one which does not change.

Therefore these programmes are stored in non-volatile memory such as ROM, or PROM, EPROM, EEPROM.

In the trainer kit, ROM contains only the monitor program which is an application program for the trainer system. It allows the user to interact with microprocessor to enter user program and execute it. The user program is not stored in ROM because it needs not to be stored permanently. The user program is stored in RWM or RAM.

Data Memory:

A microcomputer manipulates data according to the algorithm given by the instruction in the program in the program memory. These instructions may require intermediate results to be stored. The functional block in the µc used for this storage is the data memory. Microprocessors also have a small amount of memory in the form of internal registers which can also be used if available for such storage. External data memory is needed if the storage requirement is more.

Apart from intermediate storage, the data memory may also be used to provide input data needed by the program and to store some of the results of the program. Data memory is used for all storage purposes other than storage of program. Therefore, they must have Read-Write capability called Read-Write Memory (RWM).

Both ROM & RWM are arranged into words, each of which has a unique address. The address of a word in memory is different than its contents. To refer the contents of a memory location, its address is placed in parentheses. Therefore, **X** is an address and (**X**) is the content of that address **X**.

The address decoder takes an address and from the control unit and selects the proper memory location. Finding the correct memory location and obtaining its content takes certain amount of time, this times is the access time of the memory. The access time affects the speed of the computer since the computer must obtain the instruction and data from the memory. Computer memories are usually RAM so that all memory location have the same access time. The computer must wait whenever it uses its memory, typical memory access time range from few nano-secs to several μsecs. Memory sections often subdivided into units called pages. The entire memory section may involve million of words, whereas a page contains between 256 & 4k warts. The computer may access a memory location by first accessing a particular page and then accessing a location (or line number) on that page. The advantage of paging is that the computer can reach several locations on the same page with just the address on the page. The process is like describing street address by first specifying a street and then listing the house numbers. The control section transfers data to or from memory as follows:

1. The control section reads an address to the memory.
2. The control section sends a read and write signal to the memory to indicate, the direction of the transform.
3. The control section waits until transfer has been completed. This delay precedes the actual data’s transfer in the input case and follows it in the output case.

Input/Output Ports:

The input & output ports provide the microcomputer the capability to communicate with the outside world. The input ports allow data to pass from the outside world to the µc data which will be used in the data manipulation being done by the microcomputer to send data to output devices

The user can enter instruction (i.e. program) and data in memory through input devices such as keyboard, or simple switches, CRT, disk devices, tape or card readers. Computers are also used to measure and control physical quantities like temperature, pressure, speed etc. For these purposes, transducers are used to convent physical quantise into proportional electrical signals A/D computers are used to convert electrical signals into digital signals which are sent to the compute.

The computer sends the results of the computation to the output devices e.g. LED, CRT, D/A converters, printers etc. These I/O devices allow the computer to communicate with the outside world. I/O devices are called peripherals.

Clock Generator:

Operations inside the microprocessor as well as in other parts of the microcomputer are usually synchronous by nature. This is done so that events in different parts of the system can proceed in a systematic fashion. The clock needed to perform this synchronous operation is provided by the clock generator. The clock generator generates the appropriate clock periods during which instruction executions are carried out by the microprocessor.

Some of the microprocessors have an internal clock generator circuit to generate a clock signal. These microprocessors require an external crystal or RC network to be connected at the appropriate pins for deciding the operating frequency (e.g. 8085A). Some microprocessors require an external clock generator (e.g. 8086). These microprocessors also provide an output clock signal which can be used by other devices in the microcomputer system for their own timing and synchronizing.

Lecture-3

Bussed Architecture

The basic components of a microcomputer, as discussed earlier, are:

1) CPU
2) Program memory
3) Data memory
4) Output ports
5) Input ports
6) Clock generator.

The clock generator generates the appropriate clock pulses for the synchronized operation of different components of microcomputer. The clock generator is on-chip in Intel 8085A microprocessor. Now, the question comes, how the microprocessor is connected to other components - memory and I/O ports. One possibility is that all the memory chips and ports are connected separately to CPU as shown in fig.1.2.

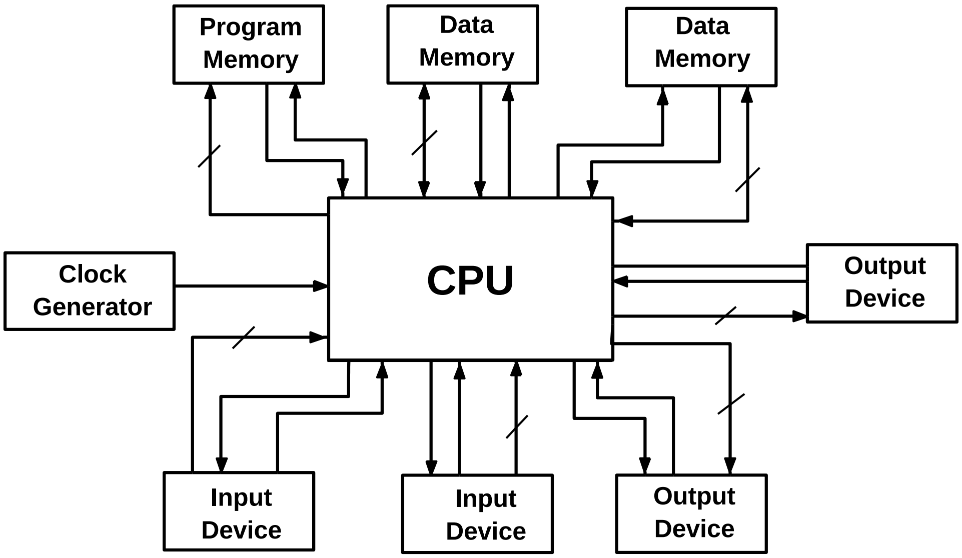


Fig.1.2 Separate Connection of Memory and I/O Devices with CPU

In this case, large no of address lines, signal lines, input and output lines are required from the CPU. The size of the CPU increases much if all the components are to be simultaneously controlled. The capability of expanding the system by adding more components will be limited. If a CPU is fabricated on a chip such that it provides interface to N devices, adding one more device will be impossible. The system becomes too complex, therefore, bussed architecture is used to connect component to the microprocessor.

The Bussed Architecture for Microprocessor:

The first question is what is a 'Bus'? Bus is a group of parallel lines that connect two or more devices. It carries information in bits. Whenever processor (CPU) needs to access any memory or I/O device of the microcomputer system, it does so by setting up signals on the address bus to identify the appropriate circuit. Data may be transferred by means of data bus, in required direction between the device and the processor. Signals on the control bus serve a number of purposes such as control the transfer of data direction.

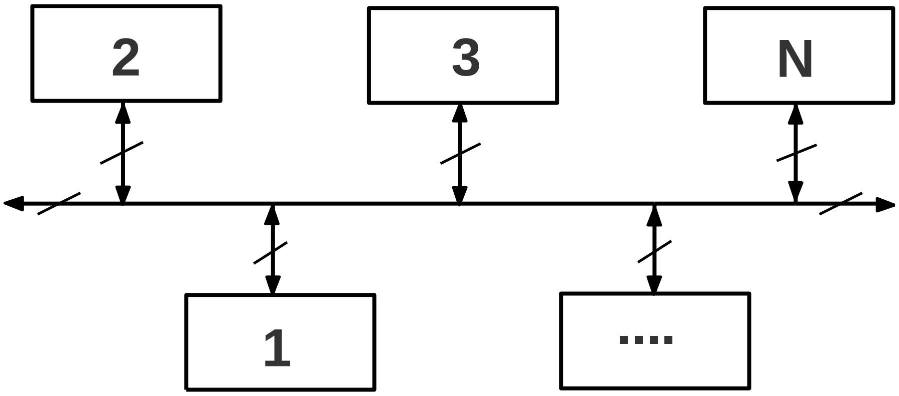


Fig.1.3 Data Bus Shared by 'N' Devices

Consider a situation where there are N devices connected to one single data bus as shown in fig.1.3. Some of them are input and some are output devices. Let us consider, device ‘1’ wants to transfer data to device ‘2’ using this line. This transfer of data can be performed provided:

a. Device ‘1’ knows when to output data such that device ‘2’ is in a position to receive the data. This can be easily ensured if device ‘2’ has some means of signally device ‘1’ to output data.
b. Other than device ‘1’, no device outputs data on that signal line during this period. Device ‘1’ should be the only device driving the signal line at this time.
c. Other than device ‘2’, no device should accept the data from the data bus.

If the above conditions can be ensured, the same signal line can be shared by all N devices to transfer data between any two of the devices. The signal lines that are shared by a number of devices are referred to as the bus.

Normal gates are not suited for driving the bus in a bussed architecture. This is because such a gate will always be outputting either logic ‘1’ or logic ‘0’ it will not be possible to satisfy condition ‘b’. Therefore, tri-state buffers are used for driving a bus.

The Microcomputer Bus:

The microcomputer contains three buses which carry all the address, data and control information involved in program execution. These buses connect the microprocessor to other elements - memory and I/O devices so that transfer of information between the microprocessor & any of the elements can take place.

Address Bus:

In a microcomputer system, it is the CPU, which is the heart of the system, decides what action is to be taken in the system. Therefore, processor always selects the device for data transfer by putting the address of the device on the address bus. On address bus, information (address) flow takes place only in one direction, i.e., from the microprocessor to the memory or I/O devices. Therefore, this is called unidirectional address bus. The processor uses the address bus to identify an I/O device or memory. In the case of memory, this address also identifies the particular memory location inside the memory.

In 8085A processor, this bus is typically 16 bit long (A_0 to A_{15}). The CPU can generate 2^{16} or 65,536 different addresses on this bus. A memory location or an I/O device can be represented by each one of these addresses.

When the microprocessor wants to transfer information between itself and a certain memory location on I/O devices, it generates the 16-bit address (in 8085A processor) from an internal register on its 16 address pins, which then appear on the address bus. These address bits are decoded to determine the desired memory location on I/O devices. The decoding process normally requires logic circuit (decoders) in the microcomputer system. The logic circuit decodes the address to decide which I/O device or memory location is required to be involved in any data transfer operation. If this decoding uniquely identifies only one port or memory location, then only data transfer takes place.

Data Bus:

A set of data lines (8 in 8085A processor)) referred to as the data bus is shared by number of devices to transfer data between microprocessor and peripherals. Care must be taken that at a time only one device should output data on the data bus, the other devices which can output data meet be in high-Z condition. The data can flow in both directions, i.e., to or from the microprocessor. Therefore, this is called bidirectional data bus (BDB). In some microprocessors, the data pins are also used to send other information such as address bits in addition to data. This means that the data pins are time shared or multiplexed. In Intel 8085A microprocessor lower 8-bits of the address (A_7-A_0) are time-multiplexed with the 8-bit data (D_7-D_0) and, therefore, this bus is called AD bus (AD_7-AD_0).

Control Bus:

The control bus is comprised of various single lines that carry control signals. These signals are used to synchronize the operation of the individual microcomputer elements. The microprocessor uses these signals for every operation it performs, like reading or writing a memory location or I/O device. These signals are also used to identify a memory location or an I/O device, e.g., $\overline{RD}$, $\overline{WR}$, IO/$\overline{M}$. Some of the signals of the control bus are issued by the processor and some of the signals are received by the processor. Therefore, the control bus is called bidirectional control bus (BCB). The difference between BDB and BCB is that in BDB all data lines are either in input mode or in output mode whereas in BCB the direction of signal flow on a line is fixed.

The bussed architecture of microprocessor is shown in fig.1.4

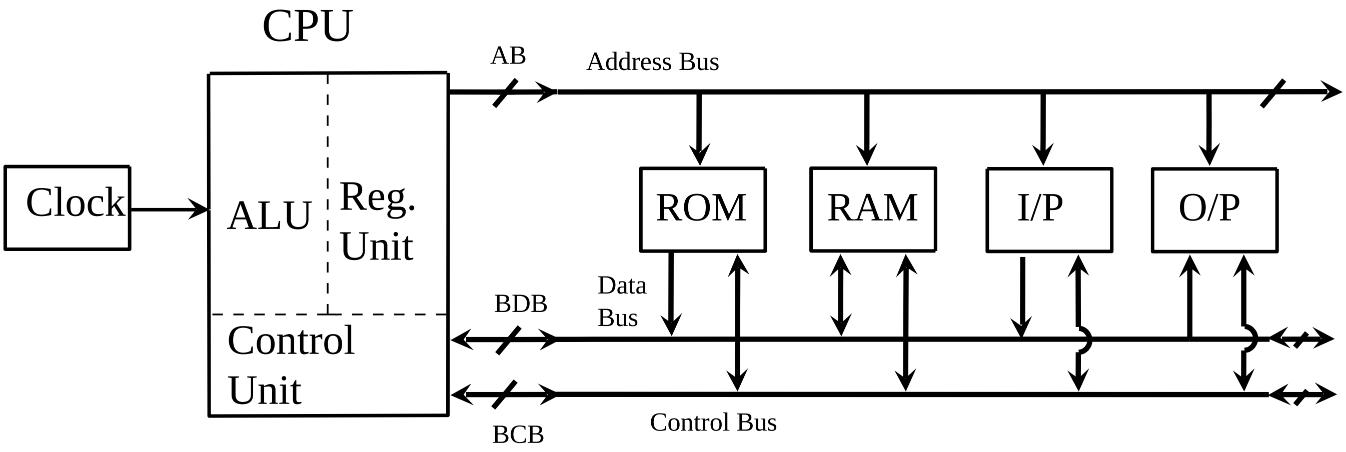


Fig.1.4 Bussed Architecture of Microprocessor

A microprocessor performs the function of the central processing unit. The microprocessor in combination with memory, I/O & a clock is essentially a microcomputer.

Computer Language:

Each machine has its own set of instructions based on the design of its microprocessor. To communicate with the computer one must give instruction in binary language or machine language the form in which it is stored in memory, i.e, as patterns of 1s & 0s. Since it is difficult for most users to write programs in machine language, computers manufactured have developed English like words to represent the binary instructions of a microprocessor. e.g. ADD, SUB or JMP etc. Users can write programs, called assembly language programs (ALP), using these words called mnemonics. However, since the microprocessor can only executes the bit patterns of machine language instructions, the assembly language program must be converted to machine codes. This conversion can be carried out by hand, but this procedure is also time consumes and error prone. Special programs are available for each type of microprocessor that converts their assembly language programs to the equivalent machine codes. These programs are called assemblers and are run either on a microcomputer or minicomputer.

Because an assembly language is specific to a given machine, programs written in assembly language are not transferable from one machine to another. To circumvent this limitation, such general purpose languages as BASIC, FORTRAN, PASCAL, PL/M, C, have been devised, a program written in these languages are called high level languages (HLL). The programmes written in HLL are converted to machine language by another program called compiler or interpreter.

High–level languages do have same limitations in processor applications. The machine codes produced by compiler may be less efficient than that of the optimum equivalent ALP, increased memory requirement may not be important in view of cheap memory chips but increased executions time may be unacceptable in time critical applications. It is then desirable to write time critical parts of the program in assembly language. Further many peripheral device dependent operations may have to be programmed in ALP as such operation is often not supported by high level language.

Lecture-4

MEMORY:

It is a storage device. It stores program instructions, data and the results. There are two kind of memories; semiconductor memories & magnetic memories. Semiconductor memories are faster, smaller, and lighter and consume less power. Semiconductor memories are used as the main memory of a computer. Magnetic memories are slow but they are cheaper than semiconductor memories. Magnetic memories are used as the secondary memories of a computer for bulk storage of data and information’s. With the development in technology, semiconductor memories are used everywhere.

If a memory stores N- words of information each word being of m bits, we say it is a N x m memory. E.g. 8x4 memory means there are 8 words and each word containing 4- bit of information (called nibble). 8 words are stored at 8-memory locations and these memory locations are clearly identified by 8 unique addresses.

Table: Formulation of Memory Address

A_2	A_1	A_0	Decimal Equivalent	Memory Location	Contents of the memory location
0	0	0	0	0	M(0)
0	0	1	1	1	M(1)
0	1	0	2	2	M(2)
0	1	1	3	2	M(3)
1	0	0	4	3	M(4)
1	0	1	5	5	M(5)
1	1	0	6	6	M(6)
1	1	1	7	7	M(7)

Addresses are formulated by bit combination available in wires known as address lines. To identify 8-memory locations, 3 address lines designated $A_2A_1A_0$ are required. The memory locations identification and the corresponding contents stored are shown in table. M(0) is the content of memory location ‘0’ and it has 4 bits here. M(1) is the content of memory location ‘1’ and so on. It can also be represented as shown in fig.2.1.

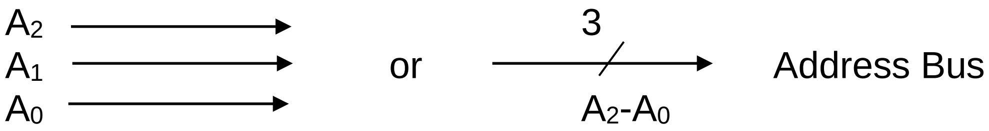


Fig.2.1 Representation of Address Bus

The three address lines A_2 A_1 A_0 together is known as address bus. It is a unidirectional bus. The microprocessor always sends the addresses.

In general, an Nxm memory shall have ‘k’ address lines designated A_{k-1} A_{k-2} A_{k-3} ……… $A_2A_1A_0$ such that ‘k’ is the smallest integer satisfying the inequality $2^k \geq N$. e.g. 200x8 memory shall have 200 memory locations. Each location contains 8 bit of information. To identify 200 memory locations we require a minimum of 8 (=k) lines designated A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0. However k = 8 address lines can identify a total of 256 memory location starting from $(0000\ 0000)_2$ to $(1111\ 1111)_2$ or 00_H to FF_H. But we are using only 200 memory locations and rests of the locations are redundant. The 200 memory locations shall be identified starting from $(00000000)_2$ to $(11000111)_2$. The other combinations $(11001000)_2$ to $(11111111)_2$ are not used in this memory & are redundant addresses. Since it is too tiring & boring to use binary numbers for identifying the addresses we normally make use of hexadecimal number notation. E.g. 200 memory locations are identified starting from 00_H to $C7_H$ and $C8_H$ to FF_H are redundant memory locations. Using 10 address lines designated as $A_9A_8A_7$……..$A_2A_1A_0$, one can directly address 2^{10} = 1024 memory locations. This is known as 1k memory locations.

The capacity of a memory is specified terms of the maximum number of words the memory can store. In general, if the memory has k-bit address and each word is of length m, then the memory has a capacity of 2^k x m bits, organized as 2^k words each of m-bits. If k=10, then the memory can store 1024 words or 1k words. Intel 808A microprocessor has 16-address lines. Therefore it can address directly 2^{16} memory locations.

$$\begin{aligned} 2^{16} \text{ memory locations} &= 2^8 \text{ x } 2^{10} \text{ memory locations} \\ &= 64\text{k memory locations} \\ &= 65536 \text{ memory locations.} \end{aligned}$$

Thus, 8-bit microprocessor provides a maximum of 2^{16} or 64k memory addresses ranging from 0000 to $FFFF_H$.

Development of Memory:

Let us see how semiconductor memories are developed. The smallest unit of information a digital system can store is a binary digit which has a logic value of ‘0’ or ‘1’. A bit of data is stored in an electronic device called a flip–flop or a 1-bit register. A flip – flop is a general memory and has two stable states in which it can remain indefinitely as long as the operating power is not interrupted. The output can be changed only if the input signals allow for it. A very simple type of flip–flop is D- type flip-flop as shown in fig.2.2.

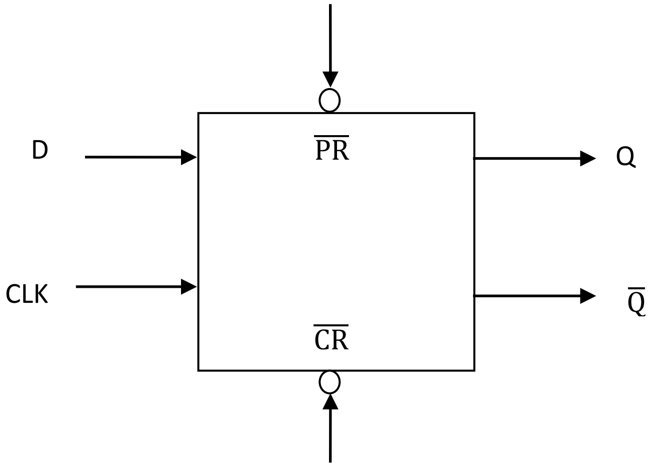


Fig.2.2 D Flip Flop

It has a single data input D and two outputs, Q and $\overline{Q}$. Output Q represents the state of flip- flop; $\overline{Q}$ represents the complement of the flip-flop's state. The logic value at a flip-flop's D input when a clock signal (CLK) occurs is stored in the flip- flop. If the stored value is equal to 1 (Q = 1) the flip–flop is set. If the stored value is equal to 0 (Q = 0) the flip–flop is clear.

The logical operation of a D type flip–flop is expressed by the characteristic equation $Q_{n+1} = D_n$. This equation indicates that the output of a D- type flip – flop, after the occurrence of a clock pulse, Q_{n+1} is equal to the logic value of the D–input before the occurrence of the clock pulse D_n. But D–type flip-flop differs with regard to the precise time at which the clock pulse causes the input data to be accepted, the output to change in accordance with the input, and the output to be held or latched.

Two clock pulse or strobes are shown in fig.2.3

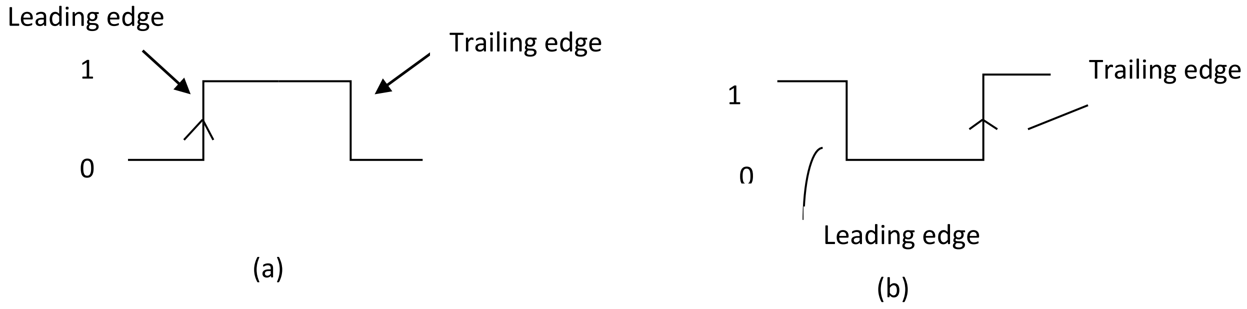


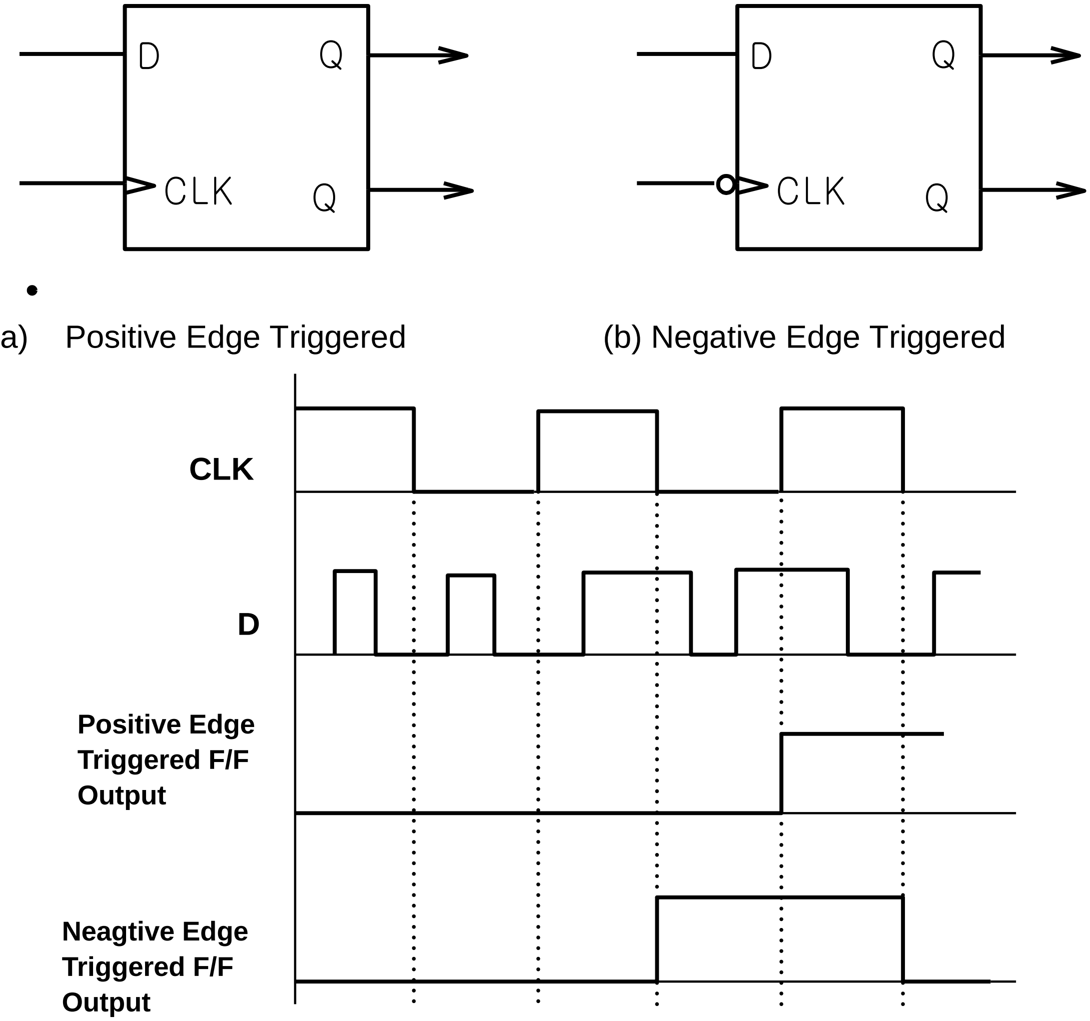
Fig.2.3 Rising Edge Active Clock Pulses

Positive clock pulse: This signal is logic 0 in its quiescent state, makes a transition to logic 1 remains at logic 1 momentarily, and then returns to logic 0. The leading edge of the pulse is a 0 to 1 or positive transition and the trailing edge is a 1 to 0 or negative transition.

Negative clock pulse: The quiescent value of this signal is logic 1 and it makes a momentary negative transition to logic 0 followed by a positive transition back to logic 1. A positive transition is also referred to as a rising edge, and a negative transition is also referred as the falling edge.

An edge triggered D–type flip-flop latches the logic value at the D input during the clock pulse’s transition from one logic value to the other. The sensitivity of the flip-flop to the transition (edge) of the clock is indicated on the flip-flop logic symbol by a dynamic indicator, a triangle ‘>’ at the clock input. Positive edge triggered flip–flops latch on the positive transition of the clock. Negative edge triggered Flip-flops latch on the negative transition of the clock.

If the clock pulse of fig (a) is applied to the positive edge pulse triggered flip-flop, the data is latched at the leading edge of the pulse. If the clock pulse of fig (b) is applied to a positive edge triggered flip-flop, the data is latched at the trailing edge of the pulse. This is shown in fig.2.4. Note that in the edge triggered flip-flop, the input is accepted, and the output changes and is latched during a single clock transition.

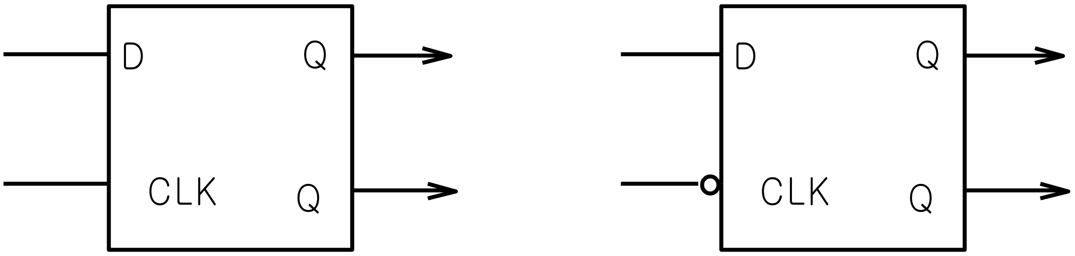

D
Q
CLK
Q
D
Q
CLK
Q
CLK
D
Positive Edge Triggered F/F Output
Neagtive Edge Triggered F/F Output

(a) Positive Edge Triggered (b) Negative Edge Triggered

Fig.2.4 Edge Triggered D Flip-Flops and Outputs

A level triggered flip-flop usually referred to simply as a latch. It has a clock input that is sensitive to the level of the clock signal. The output of a positive level triggered D-flip-flop follows the D input when the clock signal is high. When the edge makes a transition from 1 to

0, the data present at the D input is latched. The output of a negative level triggered D flip-flop follows the input when the clock is logic ‘0’ and latches the input on a 0 to 1 transition. Thus for a level triggered flip-flop, the output follows the input, when the clock is at the trigger level. This is shown in fig.2.5. During this condition the flip-flop is referred to as being transparent. The input data is latched on the transition from the trigger level to the quiescent level.



- Positive Edge Triggered (b) Negative Edge Triggered

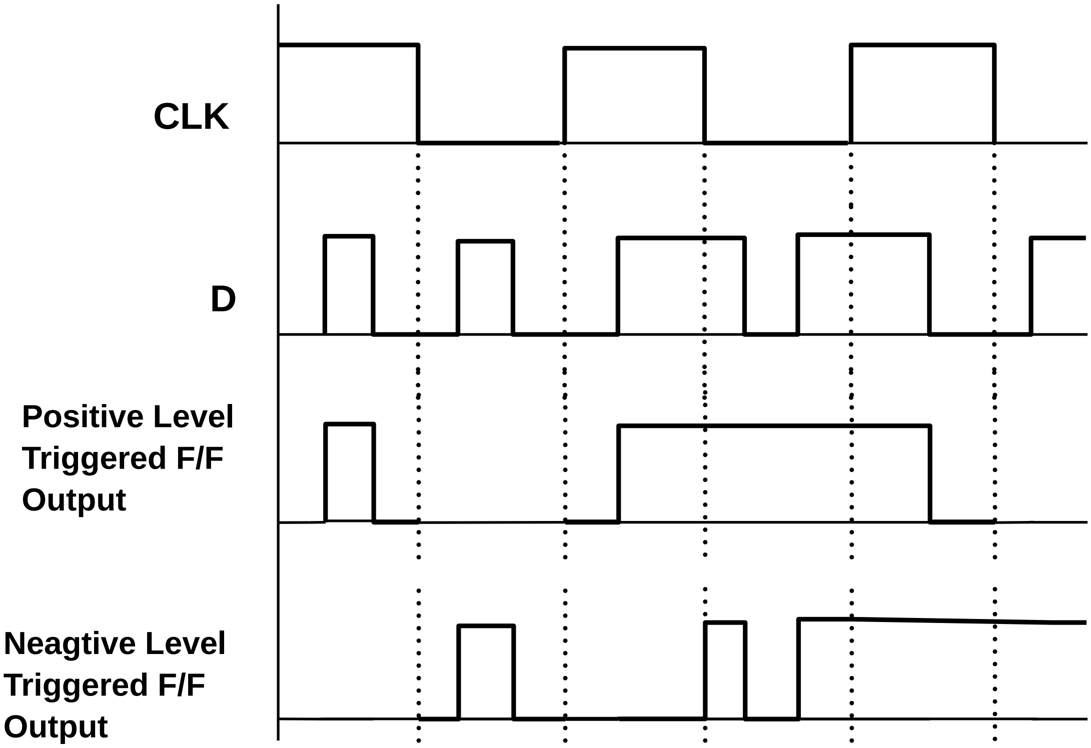

CLK
D
Positive Level Triggered F/F Output
Neagtive Level Triggered F/F Output

Fig.2.5 Level Triggered D Flip-Flops and Outputs

The D flip-flop shows two additional inputs common to most ICs $\overline{PR}$ (Preset) & $\overline{CR}$ (Clear). Both the inputs are active low signals. Preset & clear are asynchronous input; they affect the state of the flip-flop independent of the clock's level or transition. Thus preset & clear have override influence on clock & synchronous input. Logic '0' at the clear input clears it for proper operation. Preset & clear inputs are not strobed simultaneously.

Examples of latches:

Typical examples of transparent latches are 74LS373 & the Intel 8282 shown in fig.2.6(a) & (b). Both are functionally similar; however, they are not pin compatible. These octal latches are suitable to latch 8-bit data.

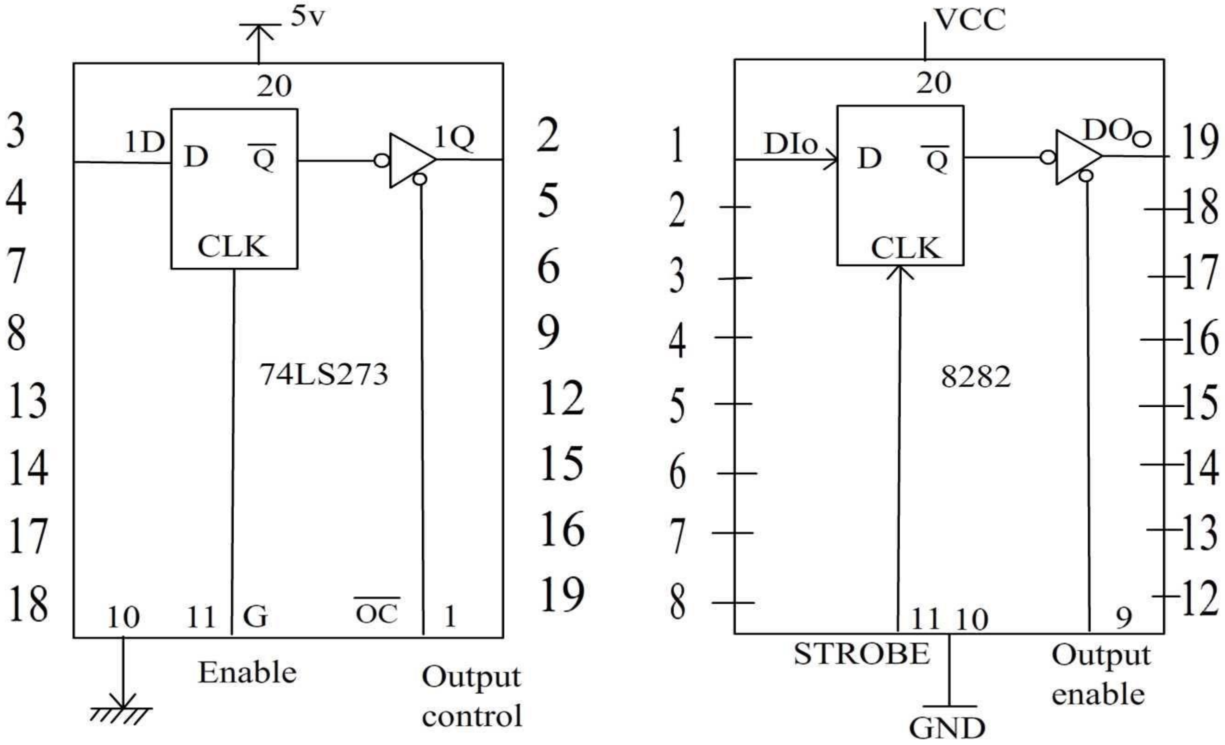


Fig.2.6 Schematic Diagram of (a) 74LS273 Latch (b) Intel 8282 Latch

Function table:

Output control	Enable G	Input	Output
L	H	H	H
L	H	L	L
L	L	X	Q
H	X	X	High-Z

Output enable	STB	Input DI	Output
L	H	H	H
L	H	L	L
L	L	X	Data latch
H	X	X	L

These devices include eight D latches with tri-state buffers. They require two input signals; enable (G) & the output control $\overline{\text{OE}}$ for the 74LS373 which are synchronous to the strobe (STB) & the output control $\overline{\text{OE}}$ for the 8282. The enable is an active high signal connected to the clock signal input of the flip-flop. When this signal goes low, data are latched from the data bus. When the output control is low (active) the data latched is accessible to the display devices.

Lecture-10

Intel 8085 Microprocessor

The 8085A is an 8- bit microprocessor suitable for a wide range of application. It is a 40-pin DIP (Dual in package) chip, based on NMOS technology. It contains approximately 6200 transistors on a 164 x 222 mil chip. The pin configuration is shown in fig.3.1.

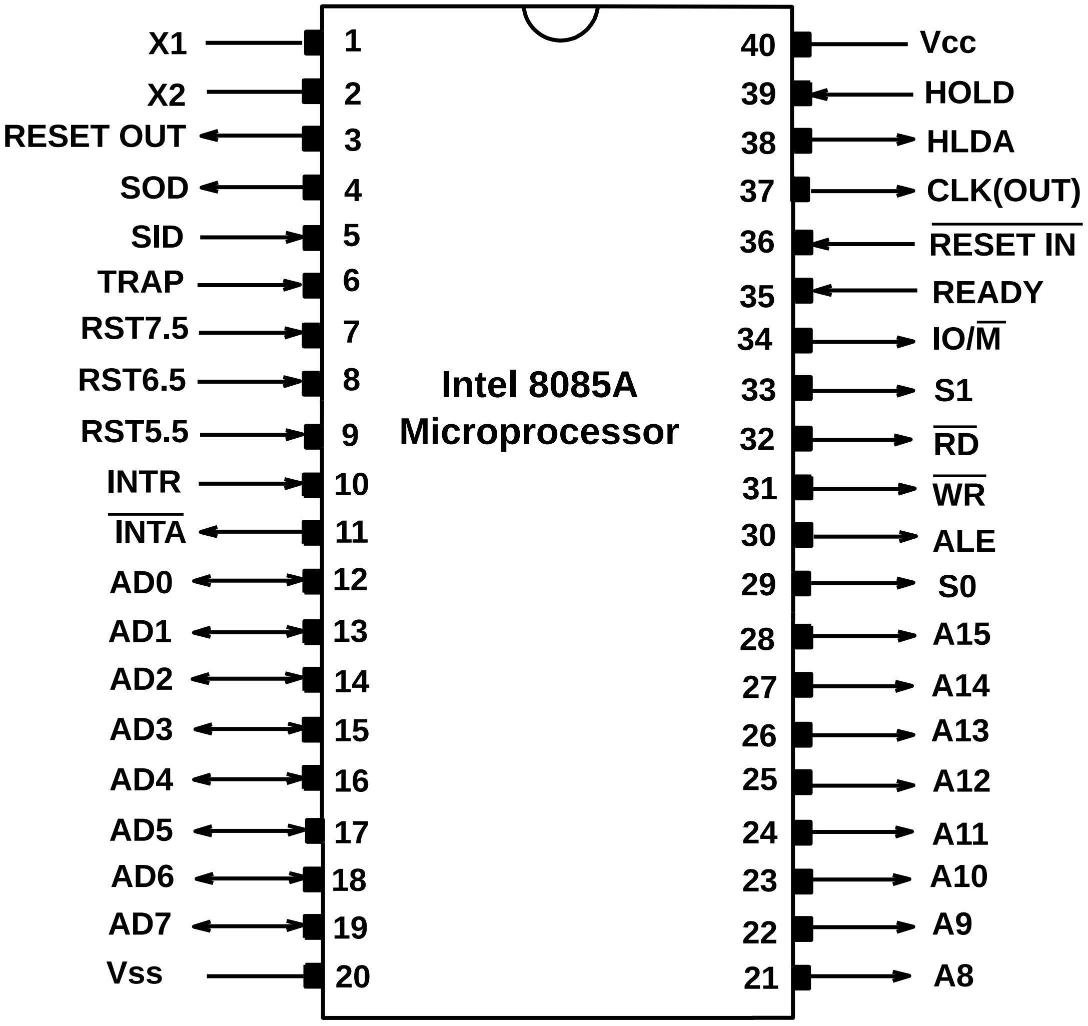


Fig.3.1 Pin Configuration of Intel 8085A Microprocessor

It requires a single +5V supply between V_{cc} at pin no. 40 and V_{ss} at pin no. 20.

Pin Configuration of Intel 8085A Microprocessor:

A_{15} – A_8 at pin no. 28 to pin no. 21:

The microprocessor can address directly 2^{16} memory locations or 65536 memory locations or 64k memory locations using 16-address lines (A_{15}-A_0). Pin no. 28 to pin no. 21 give us the higher order 8-bits of the address (A_{15}-A_8). These address lines are uni-directional, tri-state address lines. These address lines become tri-stated under three conditions namely:

(a) During DMA (direct memory access) operation.

(b) When a HALT instruction is executed.

(c) When microprocessor is being RESET.

AD_7–AD_0 at pin no. 19 to pin no. 12:

Pin no. 19 to pin no.12 marked AD_7–AD_0 are used for dual purpose. It is time multiplexed lower 8-bit address bus (A_7-A_0) and 8-bit data bus (D_7-D_0). Because at the time when this chip was developed, the practical limit on the numbers of pins was 40. The only solution was to multiplex part of the address bus with the data bus.

Before discussing, the functions of different pins, it is better to know few more points about the processor. The microprocessor, being a logic circuit, shall move from one state to the other state during it operation. There are ten (10) different possible states for the processor and the processor will be in one of these states as long as the power is ON. These states are:

1. RESET STATE: (T_{RESET}): Whenever microprocessor is reset, it enters in reset state. The microprocessor can be in T_{RESET} state for an integral multiple of clock cycle.
2. WAIT STATE: It can be in this state for an integral number of clock cycles, the duration being determined by an external content signal input marked READY.
3. HOLD STATE: (T_{HOLD}): As long as HOLD signal is active, microprocessor is in HOLD state.
4. HALT STATE: (T_{HALT}): Microprocessor enters in this state when an HALT instruction is executed by the processor. It remains in this state till such time when an external signal dictated by the user asked the microprocessor to perform further duties.
5. The other states the microprocessor can be in are marked T_1, T_2, $T_{3,}$ $T_{4,}$ T_5 & T_6 state. Each of these states is of one clock period duration. During each of these predetermined timing slots microprocessor performs very well defined activities.

Pin no.19 to pin no.12 are used by the microprocessor to send lower order 8-bits of the memory address during T_1 timing plot of a machine cycle. Therefore, the same 8-pins are utilized as bi-directional data bus for data transfer operation in the subsequent timing plots T_2 & T_3. Hence, these pins are designated as $AD_{7\,–}\,AD_{0.}$

These 8 lines are also tri-state lines. They will be tri-stated during $T_{4,}$ T_5 & T_6 states. They will also be restated during DMA operation, during RESET operation & when a HALT instruction is executed. These lines will also be tri-stated for a very short duration of time (few neon seconds) between T_1 &T_2 states.

ALE at Pin No 30:

The 8085A uses a time multiplexed address-data bus. This is due to limited number of pins on the 8085A. Low-order 8-bits of the address appear on the AD bus during the first clock cycle i.e., T_1 state of a machine cycle. It then becomes the data bus during the second and third clock cycles i.e., T_2 and T_3 states.

ALE stands for address latch enable. It is used to distinguish whether the AD_7 – AD_0 bus contains address bits A_7 – A_0 or data bits D_7- D_0. It is a single pulse issued during every T_1 state of the microprocessor as shown in fig.3.2.

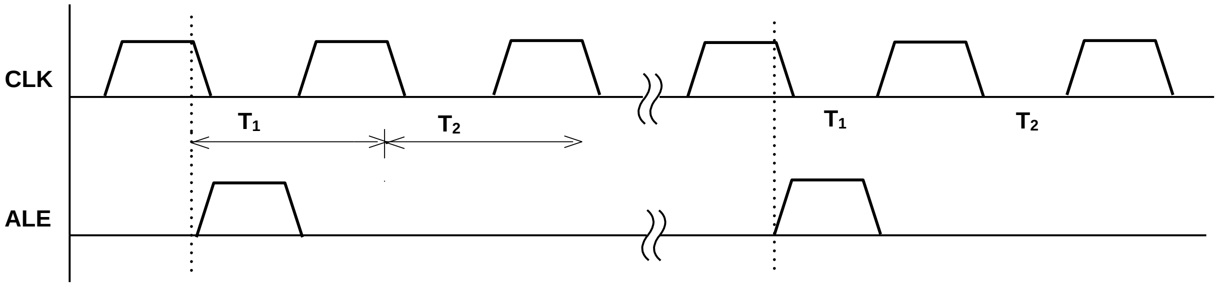


Fig.3.2 ALE Signal Issued in Every T_1 State

Since the lower 8-bits of the address information A_7 to A_0 is available at pin no.19 to pin no.12 only during T_1 period, therefore, ALE pulse can be can be used to latch address A_7 to A_0 in an external latch. ALE output is high during first half of the T_1 period and its falling edge can be used to latch the address bits A_7 to A_0 in an external latch e.g. 74LS373 register latch.

Fig.3.3a shows a schematic that uses a latch and the ALE signal to de multiplex the bus. The bus AD_7-AD_0 is connected as the input to the latch 74LS373. The ALE signal is connected to the enable (G) pin of the latch, and the output control (OC) signal of the latch is grounded. When ALE goes high during the T_1 state of a machine cycle, the latch is transparent and the output of the latch changes according to the input. The CPU is putting lower-order bits of address during this time. When the ALE goes LOW, the address bits get latched on the output and remain so until the next ALE signal.

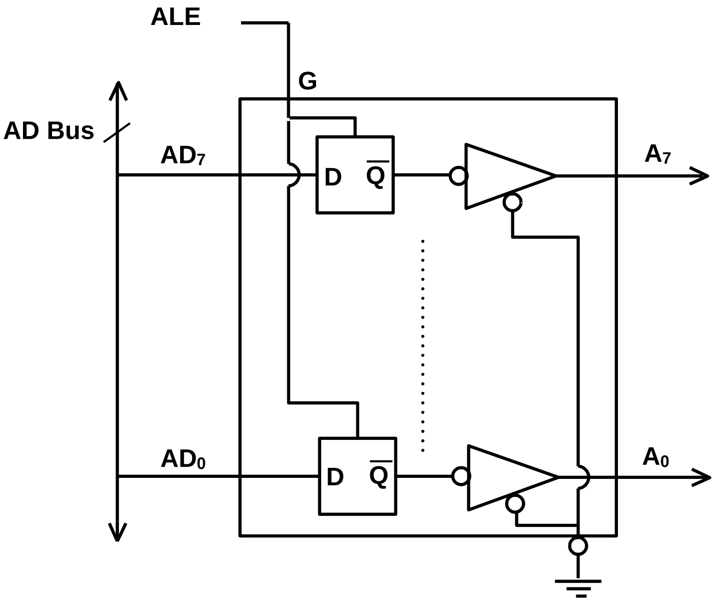


Fig.3.3a Latching of Lower Order Address in External Latch

Once saved in an external latch the lower order address A_7 to A_0 shall be available at the output of the register latch for the subsequent states T_2, $T_{3,}$ $T_{4,}$ T_5 & T_6, while pin no. 19 to pin no.12 can then be utilized by the microprocessor for bi-directional operation. The falling edge of the ALE can also be used to store status information being output by the 8085A during each machine cycle. The ALE output is never tri-stated in the 8085A. The manner of utilization of pins 19 to 12 is known as time multiplexed mode of operation.

The de-multiplexing of AD bus by latching lower byte of 16-bit address in external 8-bit latch 74LS373 is shown in fig.3.3b. Once the lower byte address is latched, the AD bus is available for bi-directional data transfer. The 8-bit higher order address issued by microprocessor in every T1 state along with latched lower byte address constitutes unidirectional 16-bit address bus. The control signals put together constitutes bi-directional control bus, where some of the signals are always input and some are always output. The three buses, address bus, data bus and control bus together constitutes system bus.

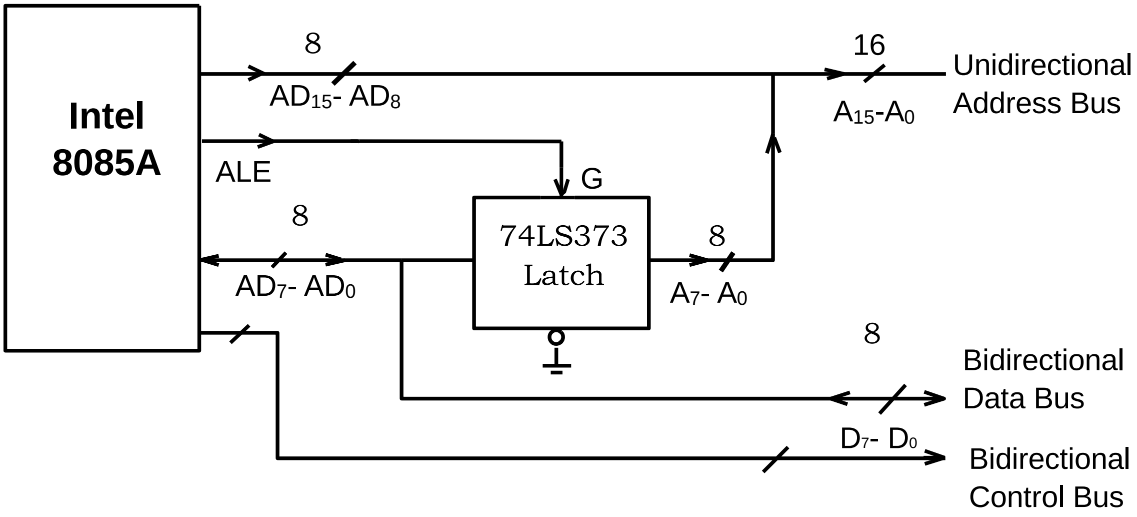


Fig.3.3b De-multiplexing of AD bus to Generate System Bus

The fact that ALE is required is a direct consequence of having a multiplexed data/address bus. This is unlike the Intel 8080 microprocessor which is similar to the 8085A but where these buses are not multiplexed. Some of the peripheral chips 8155/ 8156/ 8355/ 8755A have internal multiplexing facility, therefore, ALE input pin of these peripheral chips is connected to ALE output pin of the 8085 A, thus allowing a direct interface with the 8085 A. Thus IC chips internally de-multiplex the AD bus using the ALE signal. Since a majority of peripheral devices do not have the internal multiplexing facility, there is external hardware necessity for it.

$\overline{RD}$ & $\overline{WR}$ Control signals at pin no 32 and at pin no 31:

The BDB at pin no 19 to 12 are used for bi-directional data transfer operation during T_2 & T_3 states. When the BDB is inputting the information from the external world into the microprocessor, we say that μp is in READ mode and operation is READ operation. When the μp is outputting 8-bit of information to the external world through BDB we say μp is in WRITE mode and operation is WRITE operation.

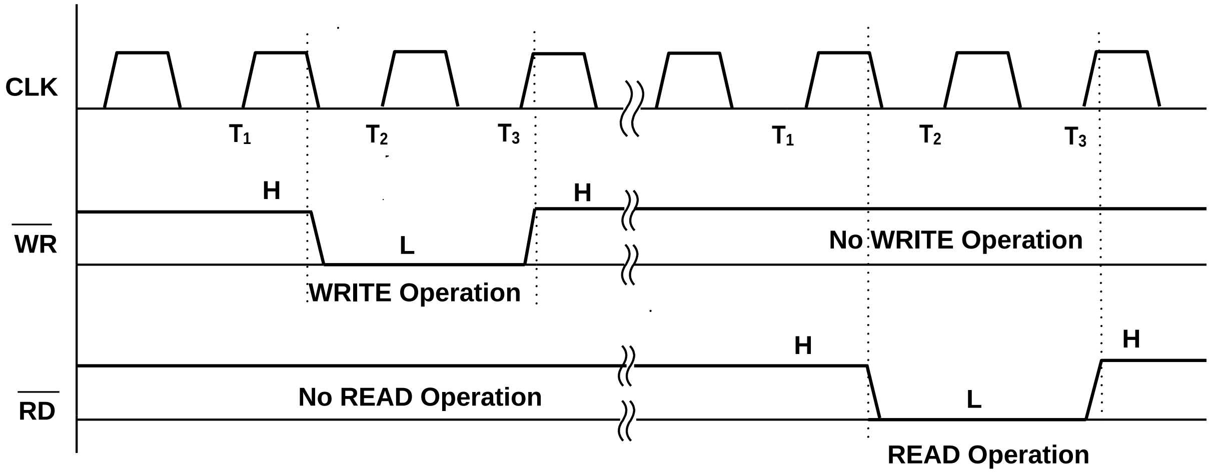


Fig.3.4 Read and Write Signals Issued During T_2-T_3 State

To tell the external world that μp is in WRITE mode, μp issues a control signal output $\overline{WR}$ at pin no. 31. It is normally HIGH & active LOW. It goes LOW during the beginning of T_2 state and goes HIGH again during middle of T_3 state of the microprocessor. This is shown is fig.3.4.

During WRITE operation, μp first send the desired address on the address bus during T_1 state, thereafter it places the desired data on BDB which is now in input mode and then issues a control signal, $\overline{WR}$. A low level on $\overline{WR}$ indicates that the data on data bus is to be written in to the selected memory location or I/O device. Data is setup at the trailing edge. It is for the user to take appropriate action externally by the interfacing circuitry so that the data so placed goes to the appropriate device.

Similarly to tell the external world the microprocessor is in input mode for READ operation, it issues a control signal $\overline{RD}$ which is normally HIGH and active LOW. $\overline{RD}$ signal goes LOW during T_2 state and goes HIGH again during T_3 state similar to $\overline{WR}$ signal. A LOW level an $\overline{RD}$ indicates the selected memory or I/O device is to be read and the data bus is available for the data transfer. It is for the user to keep the appropriate 8-bit data either from the memory or I/O device during this period.

Both $\overline{RD}$ and $\overline{WR}$ are never made LOW at the same time. Both the signals are tri-stated during HOLD, HALT and RESET states.

IO/$\overline{M}$ at pin no 34:

IO/$\overline{M}$ is an output tri-state control signal. It is active both ways (HIGH as well as LOW). Whenever the address issued by the μp on the address lines refers to the memory then the μp makes IO/$\overline{M}$ LOW throughout T_1,T_2,T_3,T_4,T_5 & T_6 states of the machine cycle to indicate the external world that the address so sent belongs to the memory and data on the BDB refers to the memory.

Whenever the address on the address lines refers to an I/O device the μp makes IO/$\overline{M}$ control signal output HIGH to tell the external world that the address on the address bus refers to and I/O device and the data on the BDB refers to an I/O device.

Note that $IO/\overline{M}$ signal is LOW or HIGH as the case may be throughout six timing slots T_1,T_2,T_3,T_4,T_5 & T_6 states. It is for the user to make use of this feature to develop proper interfacing circuitry i.e., to generate the chip selected signals. In other words, a LOW $IO/\overline{M}$ signal enables the memory chips and a HIGH $IO/\overline{M}$ signal enables the I/O device.

Lecture-11

READY at PIN NO 35:

This is a control signal input. There are many peripheral devices which are slow in operation compared to the microprocessor speed. Eg., the access time of a memory interfaced with a μp may be much larger than the clock period of the μp. Thus, there is a need for telling the μp that the device so addressed by the μp is not ready for data transfer operation. The device, selected should have the ability to generate a control signal output READY which shall be LOW if the device is not ready for data transfer operation and goes HIGH when the device is READY for data transfer operation. This idea is summarized in fig.3.5 considering memory as the external device.

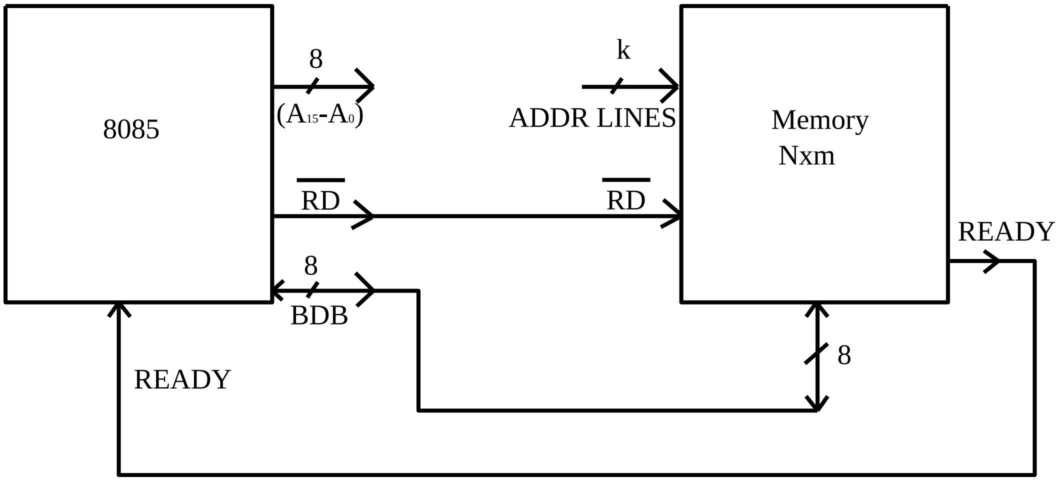


Fig.3.5 Interfacing Slow Speed Memory Using READY Signal

The μp sent the address during T_1 state of the microprocessor to address the memory and then issues appropriate $\overline{RD}$ or $\overline{WR}$ signal during T_2 state either to read the memory or write into the memory. Having issued the appropriate $\overline{RD}$ or $\overline{WR}$ signal during T_2 state of the μp, it then samples and monitors READY control signal input in the middle of T_2 state. If the READY control signal is found LOW the microprocessor knows that the device addressed is not ready for data transfer operation and therefore goes to WAIT state (T_{WAIT}). Once in WAIT state, the μp does not do any other work except monitoring control signal. The outputs remain unchanged during the wait period and they remain what they were at the end of the T_2 clock cycle. Its internal status also remains unchanged during this time. This feature provides a slow peripheral device more time to respond that what it would normally have. As long as READY signal is LOW, μp remains in WAIT state. When the ready signal goes high μp realized that that the device addressed is ready for data transfer operation and comes out of the WAIT state and goes into T_3 state. The partial state transition diagram describing the above process is shown in fig.3.6.

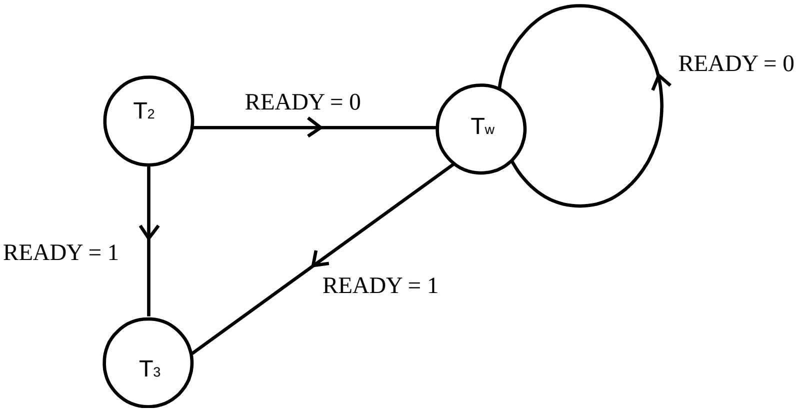


Fig.3.6 Partial State Transition Diagram Showing WAIT State

The appropriate control signal output $\overline{WR}$ or $\overline{RD}$ shall remain LOW throughout the T_{WAIT} state and goes HIGH when the μp comes out of the T_{WAIT} state and goes to T_3 state. The corresponding timing signals are shown below in fig.3.7.

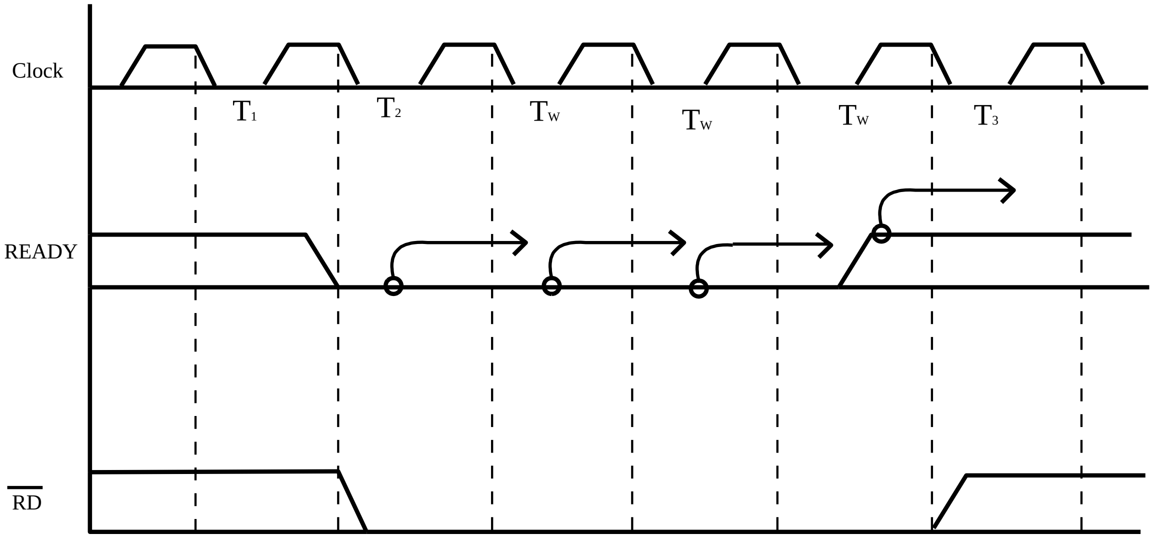


Fig.3.7 Extension of Read Control Signal During WAIT State

$\overline{\text{RESET IN}}$ at pin no.36

It is an input control signal normally HIGH and active LOW. It is used to RESET the microprocessor to its initial state. If RESET signal is held low for 600nsec (3 clock periods), this input forces the processor to do the following:

(a) Program Counter (PC) is reset to $(0000)_H$.

(b) Instruction register (IR) is cleared i.e, ongoing instruction execution is discontinued.

(c) All interrupts except TRAP are disabled with RST7.5, RST6.5 and RST6.5 also masked.

(d) Serial Output Data line (SOD) is forced to 0.

(e) During RESET IN LOW data, address and control bus are floated i.e., these buses are tri-stated

(f) Because of the asynchronous nature of the RESET, the internal registers of he microprocessor and flags are altered with unpredictable results.

(g) The CPU remains in RESET state until the $\overline{\text{RESET IN}}$ goes high.

During power-on the microprocessor must be RESET. The necessary circuitry for resetting the μp is shown in figure.

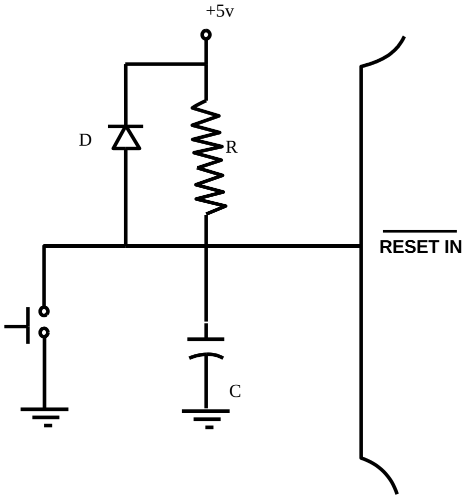


Fig.3.8 Power ON Reset Circuit

Initially, the capacitor is discharged. Therefore, to start with when the power is first put ON, $\overline{\text{RESET IN}}$ control signal becomes LOW. Therefore, the μp will be RESET to its initial state. Upon power ON the $\overline{\text{RESET IN}}$ must remain LOW for at least 10ms after the minimum V_{cc} has been reached. Depending upon the time constant RC the voltage across the capacitor exponentially increases to 5V. When the voltage across the capacitor reaches to 2.4V, $\overline{\text{RESET IN}}$ control signal goes to logical 1; μp comes out of the RESET state and straight away goes to T_1 state. The time duration to reach around 2.4V from the instant of switching the supply is around 4 to 5 clock cycles. If this duration is not maintained the resetting action of the ▯*p* is not guaranteed and therefore RC combination should be selected accordingly.

The diode D is provided for the discharge path for the capacitor when the power supply is finally put OFF. A push button switch is also provided to manually reset the μp as and when necessary. The partial state diagram when $\overline{\text{RESET IN}}$ control signal is active as shown in fig.-8.

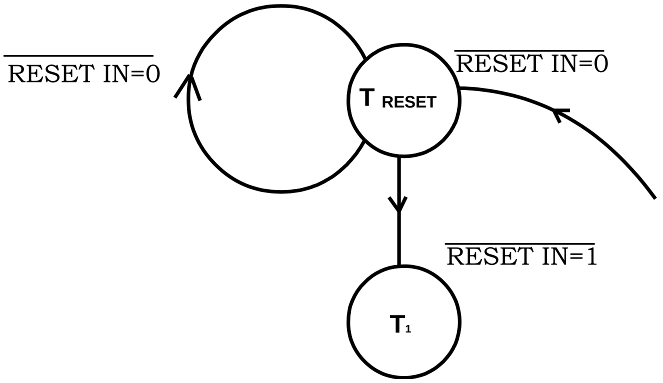

$\overline{\text{RESET IN}}$=0
T_{RESET}
$\overline{\text{RESET IN}}$=0
$\overline{\text{RESET IN}}$=1
T_1

Fig.3.9 Partial State Transition Diagram Showing Reset State

RESET OUT at Pin no.3:

It is normally low signal output. It indicates 8085A is being RESET. When $\overline{\text{RESET IN}}$ control signal at pin no. 36 is LOW, RESET OUT at pin no.3 goes HIGH. It remains HIGH as long as $\overline{\text{RESET IN}}$ is active and LOW. RESET OUT control signal is provided for the user to use it to RESET all the peripheral devices to their initial states. The signal is synchronized with the system CLK and it remains high for an integral number of clock periods. After the $\overline{\text{RESET IN}}$ goes HIGH, RESET OUT goes LOW, the processor enters in the T_1 state and normal operation begins.

X_1, X_2 terminal at pin nos.1 & 2 and CLK(OUT) at pin no.37:

The 8085A μp has an on-chip oscillator with all the required circuitry except for the crystal, LC or RC network that controls the operating frequency. The internal circuitry of on-chip oscillator is shown in fig.3.10.

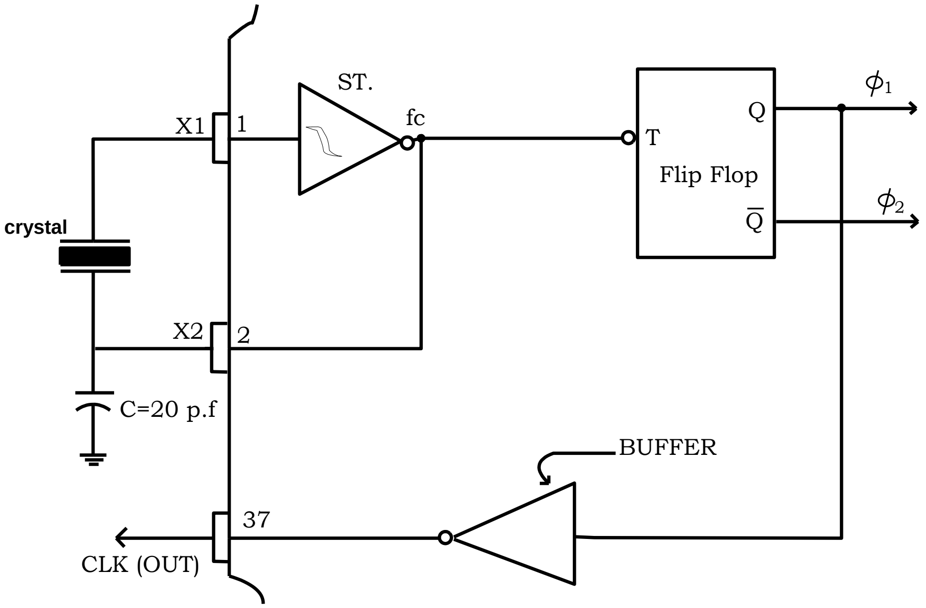


Fig.3.10 Schematic Diagram of Internal Clock Generator Circuit

A crystal is connected across X_1 and X_2 to provide a crystal frequency of $f_{crystal}$ MHz. Because of the internal T – F/F the operating frequency of $\emptyset_1$ clock is $f_{crystal}/2$ (half of crystal frequency). The $\emptyset_1$ clock is also buffered and sent out through pin no. 37 to tell the outside world as CLK(OUT) signal. CLK(OUT) signal is used for synchronizing the peripheral devices with the μp operation. The data sheet of 8085 puts a lower limit to the operating frequency at 500 kHz. The maximum operating frequency for 8085A is 3.125 MHz. The maximum operating frequency for 8085A is 5 MHz. Therefore, while using 8085A μp, a crystal having a frequency from 1.0 MHz to 6.25 MHz can be used; 10 MHz crystal has to be used for of 8085 A-2 μp. The 20 pf capacitor is necessary between X_2 and ground if the crystal frequency is less than 4 MHz to provide oscillation. It is not necessary for higher frequency. The data sheet also mentions a 15 pf stray capacitance across X_1 and X_2 internally.

A crystal is to be used across X_1 and X_2 to get stable frequency of oscillation. This ensures a fixed clock period, needed for accurate time delays. If stable frequency of oscillation is not required, a parallel resonant LC network may be used as the frequency determining network provided its frequency tolerance of approximately ±10% is acceptable. The frequency of oscillation is given by,

$$\mathbf{f = \frac{1}{2\pi\sqrt{L(C_{ext}+C_{in})}}}$$

To minimize variations in frequency, it is recommended that the value of C_{ext} must be at least twice that of C_{in} i.e., 30 pf. The use of LC circuit is not recommended for frequencies higher than approximately 5 MHz.

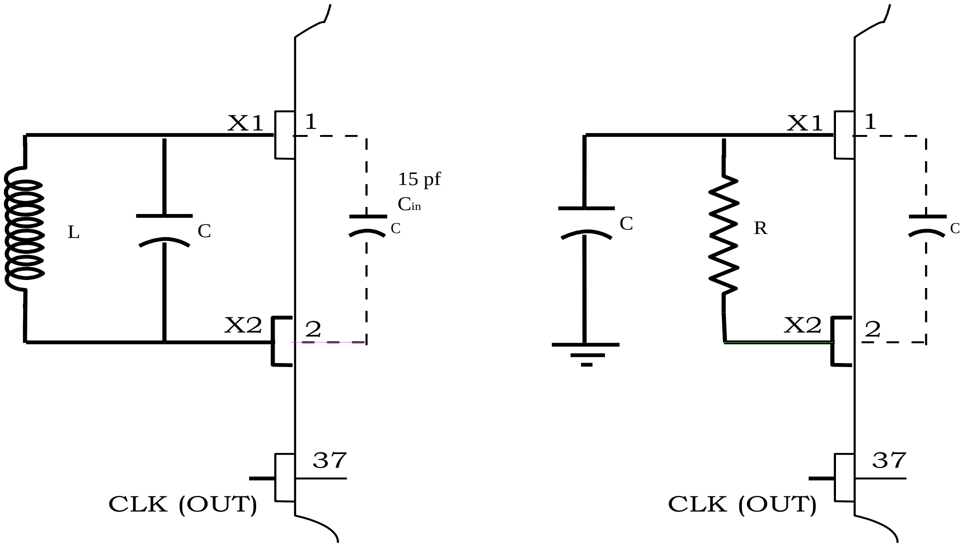
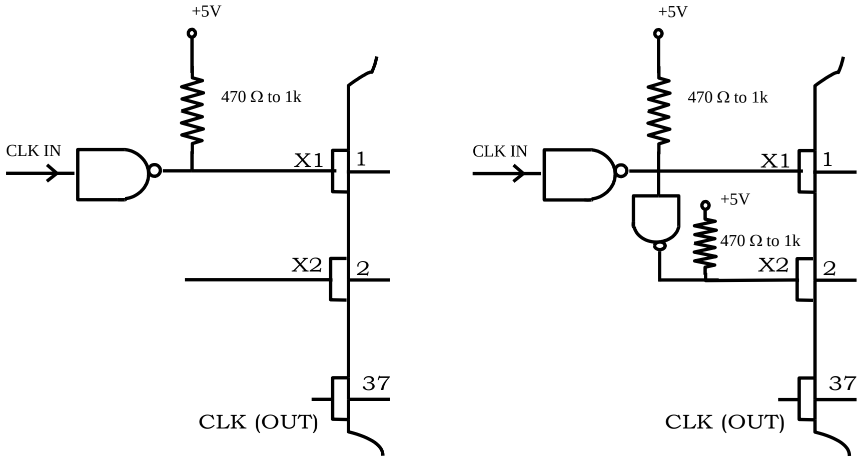


Fig.3.11 Use of (a) LC Oscillator (b) RC Oscillator as Clock Generator

An RC circuit may be used as the frequency determining network for 8085 AH, if maintaining a precise clock frequency is of no importance. Variations in the ON chip timing generation can cause a wide variation in frequency when using the RC network. Its advantage is its low component cost. The driving frequency generated by the circuit shown is approximately 3 MHz. It is not recommended that frequencies higher or lower than this are to be attempted.

If we have an external clock whose frequency is same and can be varied from 1MHz to 6.25 MHz, the external clock may be connected to X_1 and X_2 is left open as shown in Fig.3.12.



(a) Upto 6.25 MHz **(b) 6 to 12 MHz**

Fig.3.12 Connection of External Clock

If the driving frequency is 6 MHz to 12 MHz, stability of the clock generator will be improved by driving X_1 and X_2 with a push-pull source. To prevent self oscillations of the 8085, X_2 should not be coupled back to X_1 through the driving circuit. In last two cases, pull up resistors are required to assure that the high level voltage of the input is at least 4V and maximum low level voltage of 0.8V.

SID & SOD at PIN NO 5 & 4

SID stands for SERIAL INPUT DATA and SOD stands for SERIAL OUTPUT DATA. These two pins are specially provided in 8085 μp for communicating with serial devices, like CRT, TTY, and Printers etc. Microprocessor as and when needed uses SID and SOD lines for transfer of data bit by bit along the same lines. The data on SID line is loaded into accumulator bit 7 whenever RIM instruction is executed. The output serial line SOD is set or reset as specified by the SIM instruction.

Lecture-12

INTERRUPT CONTROL SIGNALS:

TRAP at pin no. 6, RST7.5 at pin no. 7, RST6.5 at pin no. 8, RST 5.5 at pin no. 9, and INTR at pin no. 10 are interrupt control signals input provided for interrupting the μp while it is executing the programme. RST stands for RESTART. These interrupt control signal input can be broadly divided in to two categories:

(a) Non – maskable interrupts

(b) Maskable interrupts

Non–maskable control signal inputs are those control signal inputs which can interrupt the μp programming execution once the power is ON. The maskable interrupts are those control signal inputs which can be individually disabled or enabled as and when necessary.

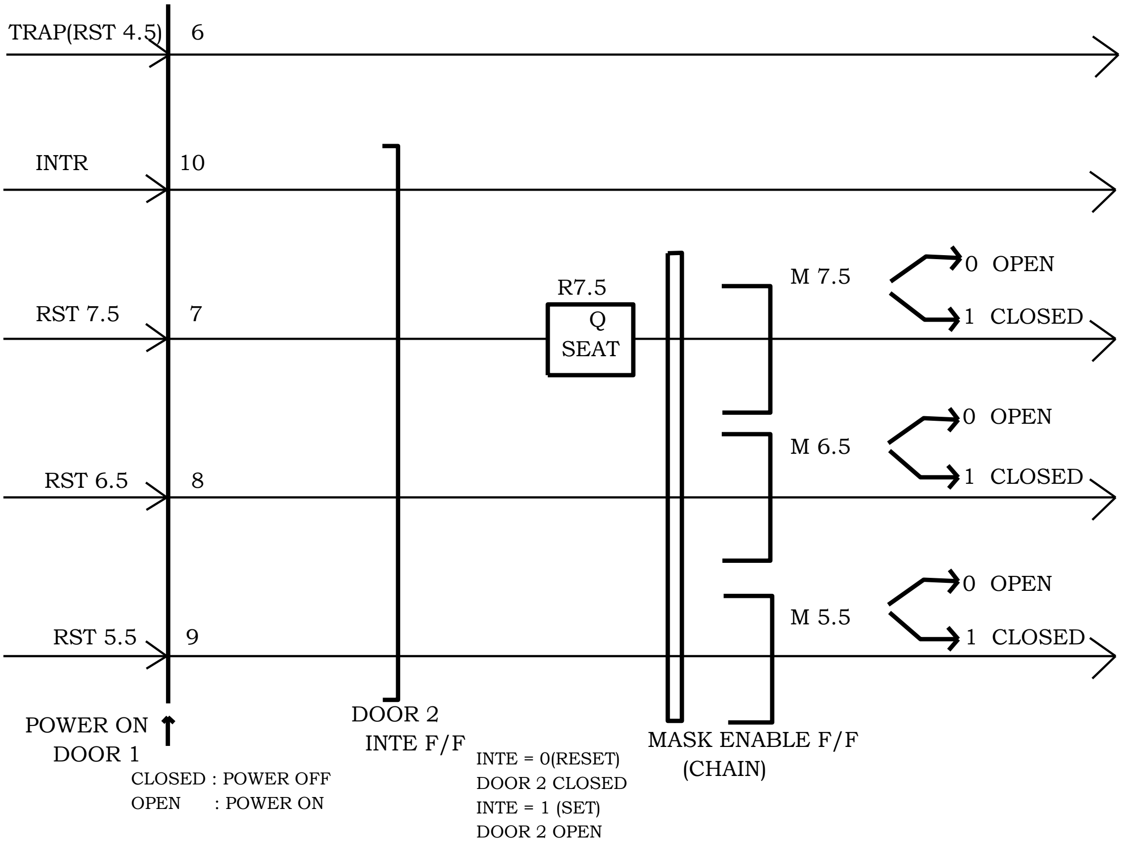


Fig.3.13 Schematic Diagram of Interrupt Section of 8085A

The way these interrupt control signal input interrupt the μp can be pictorially represented as shown in fig.3.13

TRAP at pin no.6:

TRAP control signal input of Intel 8085A processors is a non maskable (NMI) RESTART vectored interrupt. When the power is ON, it is enabled and no enable interrupt command is required. TRAP has the highest priority of any interrupt. It is both rising edge and level sensitive interrupt i.e. it becomes active at the Lo-Hi edge but must stay high until it is sampled and recognized. Whenever this interrupt is recognized, it forces the 8085A to perform a CALL 0024H instruction, means when the current instruction execution is over, the program counter (PC) is loaded with 0024H so that the CPU starts executing the program from 0024H.

INTR at pin no.10:

This is the lowest priority interrupt request in the 8085A processor and is used as a general purpose interrupt. An input of INTR=1 implies some external device has put up an interrupt and wants the CPU to execute an appropriate service routine. The 8085A monitors the status of the INTR line by sampling it in the last but one clock cycle of each instruction and during HOLD & HALT states. If the interrupt structure of the 8085A is enabled when INTR is sampled high, the PC will not be incremented and an interrupt acknowledge signal ($\overline{\text{INTA}} = 0$) will be issued by the μp in response to INTR. It is now the responsibility of the interrupting device to issue a RESTART or CALL instruction so that the 8085A can jump to the proper interrupt service subroutine.

The INTR is enabled by executing an EI instruction and is disabled by executing a DI instruction. Disabled means INTR will not be acknowledged. It is also disabled by RESET and immediately after an interrupt is acknowledged.

$\overline{\text{INTA}}$ at pin no.11:

$\overline{\text{INTA}}$ is an Interrupt acknowledge control signal output. This is an active LOW control signal output. When the μp acknowledges any interrupt than instead of $\overline{\text{RD}}$ signal it issues $\overline{\text{INTA}}$ signal to tell the external world that processor is now processing an interrupt acknowledge machine cycle. Basically, it replaces $\overline{\text{RD}}$ control signal output during $\overline{\text{INTA}}$ machine cycle. $\overline{\text{INTA}}$ is normally HIGH and becomes active LOW during T_2 timing slot of the μp and goes HIGH again during T_3 state of the μp just like $\overline{\text{RD}}$ signal. During this period $\overline{\text{RD}}$ signal is HIGH. It can be used to activate an 8259A interrupt controller chip or some other interrupt port.

RST5.5, RST6.5 & RST7.5:

These are 8085A's maskable vectored interrupt inputs. They operate exactly like INTR except for the following:

1. The RESTART instruction is automatically inserted by internal logic. It does not have to be provided from outside. These instructions are:

RST 5.5	identical to	CALL 0020 H

RST 6.5	identical to	CALL 0034 H
RST 7.5	identical to	CALL 0030 H

Restart instructions are special 1-byte unconditional CALL instructions. The address for any RST can be calculated multiplying the RST number with 8 and converting it into hexa. E.g. for RST5.5 the address is 5.5x8 = 44D = 002C H

2. RST7.5 is an edge (Low to High) sensitive interrupt unlike RST6.5, RST5.5 and INTR which are level (High) triggered.
3. These three interrupts can be individually masked or unmasked using SIM instructions.
4. They have higher priority than INTR. Among them RST7.5 has the highest priority and RST5.5 has the lowest priority.

Like the INTR, whenever any of these interrupts is recognized it disables all the interrupts. These interrupts can be enabled/disabled using EI/DI instruction.

Status Signals S_1 (33) and S_0 (29):

These two status signals along with $IO/\overline{M}$ signal output identify the type of the machine cycle being executed by the 8085A. These signals are issued (or become valid) at the beginning of the machine cycle and remain stable throughout the machine cycle. The falling edge of ALE may be used to catch the state of these lines, if required. The seven types of machine cycles are:

1. Opcode Fetch Machine Cycle (OFMC)
2. Memory Read Machine Cycle (MRMC)
3. Memory Write Machine Cycle (MWRMC)

4. I/O Read Machine Cycle (IORMC)
5. I/O Write Machine Cycle (IOWRMC)
6. Interrupt Acknowledge Machine Cycle (INTAMC)
7. Bus Idle Machine Cycle. (BIMC)

The truth table indicating the status of different signals to identify a machine cycle is shown below:

Machine Cycle	Status signal			Control Signal		
	$IO/\overline{M}$	S1	S0	$\overline{RD}$	$\overline{WR}$	$\overline{INTA}$
OFMC	0	1	1	0	1	1
MRMC	0	1	0	0	1	1
MWRMC	0	0	1	1	0	1
IORMC	1	1	0	0	1	1
IOWRMC	1	0	1	1	0	1
INTAMC	1	1	1	1	1	0
BIMC						
(a) DAD	0	1	0	1	1	1
(b) HALT state	T.S.	0	0	T.S.	T.S.	1
(c) HOLD state	T.S.	X	X	T.S.	T.S.	1
(d) RESET state	T.S.	X	X	T.S.	T.S.	1

*T.S. is tri-state.

These control signal are normally HIGH and becomes active LOW during T_2 state and goes back to HIGH during T_3 state. In between T_2 & T_3 states any no of WAIT states T_{wait} can be inserted.

HOLD at pin no 39 and HLDA at pin no 38:

HOLD (Hold) is a control signal input and HLDA (Hold acknowledge) is a control signal output. These two signals are used for hand shaked control during DMA operation (Direct Memory Access). The use of these control signals are depicted in the function diagram of fig.3.14.

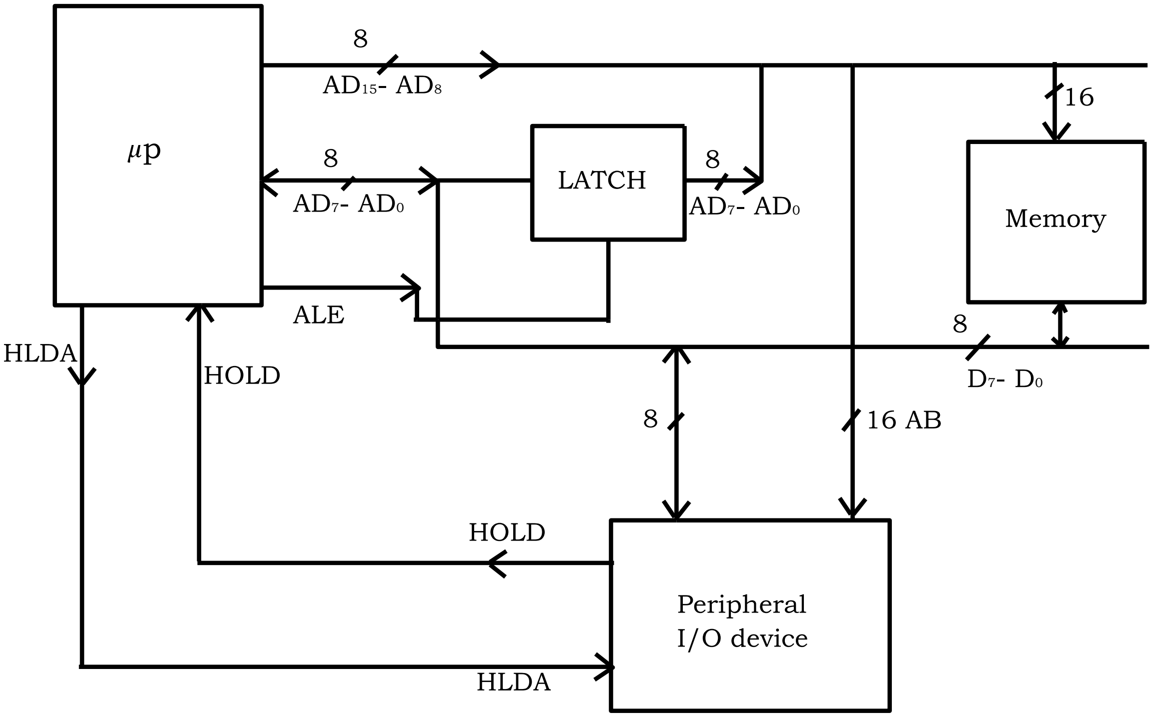


Fig.3.14 Interconnection of Peripheral Device for DMA

These two signals are used where there is more than one CPU like devices sharing the same system bus. The device asking for DMA makes the HOLD signal input HIGH. The processor which continuously monitors the HOLD signal input during each machine cycle recognizes that an external device is requesting for a DMA (i.e., the control of address bus, data bus, $\overline{RD}$, $\overline{WR}$ and IO/$\overline{M}$) completes its current machine cycle in, thereafter, tri states the address bus, data bus and $\overline{\text{RD}}$, $\overline{\text{WR}}$ and IO/$\overline{\text{M}}$ control signals and then enters into a HOLD state T_{HOLD}. Also, while entering the HOLD state the μp issues the HOLD acknowledge signal HLDA high at pin no. 38. The external device asking for DMA monitors the HLDA control signal continuously and knows that the μp has gone into HOLD state when HLDA is found HIGH. Therefore, data bus, address bus, $\overline{\text{RD}}$, $\overline{\text{WR}}$ and IO/$\overline{\text{M}}$ control signals which are tri-stated with respect to the μp are in the exclusive control of the external device asked for DMA. The DMA operation between the external device and memory continues and data transfer takes place between external device and processor memory.

Internal processing not requiring the use of the system buses may continue Note that all instructions are in program memory and, therefore, to read the instructions from the memory system bus is required which is not available to μp and therefore, μp is forced to stop the execution at the next instruction.

The μp while in HOLD state continues to monitor the HOLD control signal input. As long as it is HIGH it remains in HOLD state. The external device performing the DMA operation, after completing its operation makes HOLD signal LOW to tell the μρ that the DMA is over. The processor which is continuously monitoring HOLD signal in HOLD state recognizes the above fact and comes out of HOLD state and continues the operation from where it has gone into hold state.

HLDA goes LOW after HOLD goes LOW; the 8085A takes the control of bus one half a clock cycle after HLDA returns to ‘0’.

The system configuration indicating address lines, data lines and all control lines including “Power ON’ reset circuit, clock generator is shown in fig.3.15.

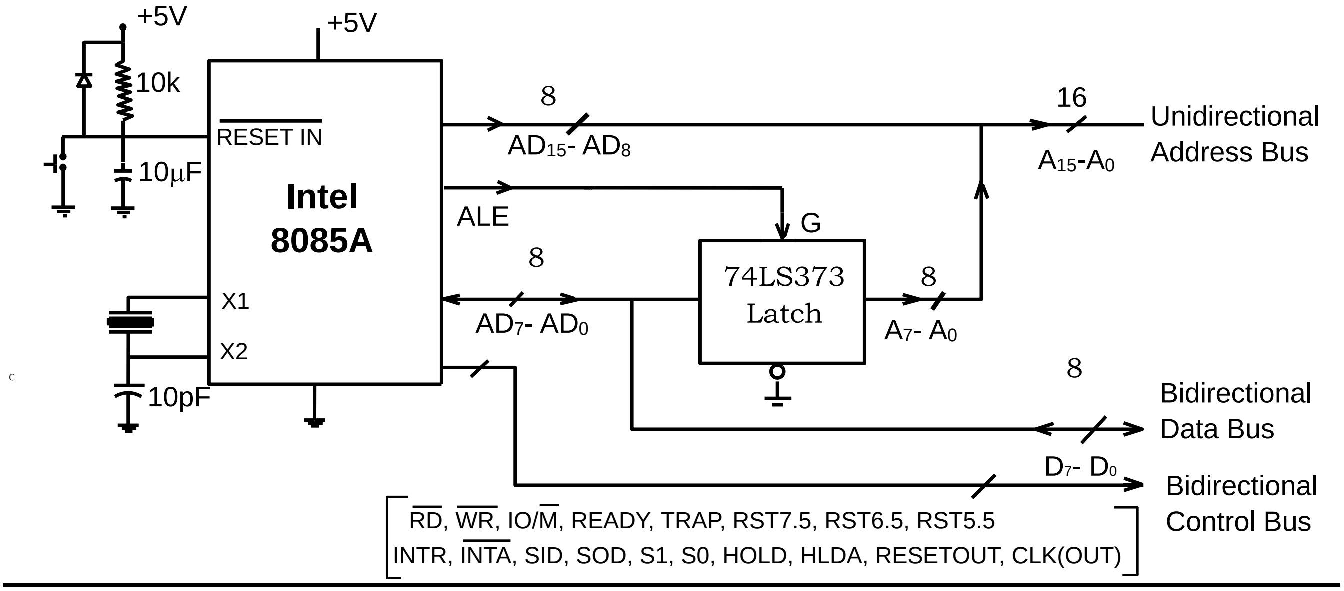


Fig.3.15 System Configuration of Microprocessor