Computer Science

Paper 2

Tejas Shah

Contents

Introduction to Microprocessors

Microprocessors

A microprocessor is a multipurpose, programmable, clock driven, register based electronic device that reads binary instructions from a storage device called memory, accepts binary data as input, processes data according to those instructions, and provides results as output.

A typical programmable machine can be represented with three components: microprocessor, memory and I/O. These three components work together or interact with each other to perform a given task.

The physical components are called as hardware. A set of instructions written for the microprocessor to perform a task is called as a program and a group of programs is called software.

The microprocessor operates in binary digits, 0 and 1, also known as bits. These digits are represented in terms of electrical voltages in the machine. The digits 0 and 1 are also synonymous with low and high, respectively. Each microprocessor recognized and processes a group of bits called the word. Microprocessors are classified according to their word length. For example, a processor with an 8-bit word is known as 8-bit microprocessor.

Microprocessor as programmable device

The microprocessor can be instructed to perform different tasks within its capacity. It can be used to perform various sophisticated computing functions, as well as simple tasks such as turning devices on and off. A programmer can select appropriate instructions and ask the microprocessor to perform various tasks on a given set of data. These instructions are entered or stored in storage (memory) which can be read by the microprocessor.

Memory

Memory is like the pages of a notebook with space for a fixed number of binary numbers on each line. However, these pages are generally made of semiconductor material. Typically, each line is an 8 bit (1 byte) register that can store eight binary digits, and several of these registers are arranged in a sequence. These registers are always grouped in powers of two. The memory is usually measured in the number of bytes it can hold. There are two types of memory:

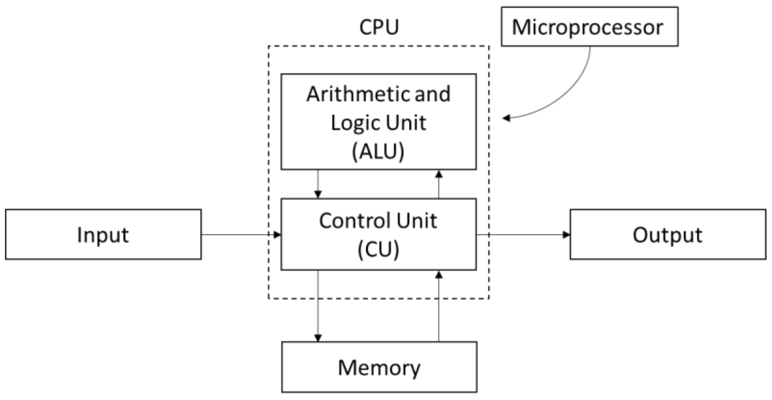
- ROM: Read Only Memory.
- RAM: Random Access Memory.

The user writes the necessary instructions and data in memory through an input device, and asks the microprocessor to perform the given task and find the answer. The answer is generally displayed at an output device or stored in memory.

Input / Output

The user can enter instructions and data into memory through devices such as a keyboard or simple switches. These devices are called input devices. The microprocessor reads the instructions from the memory and processes the data according to those instructions. The result can be displayed by a device such as seven segment LED (light emitting diode) display or printed by a printer. These devices are called output devices.

Microprocessor as a CPU



We can also view the microprocessor as a primary component of a computer. Traditionally, the computer is represented in block diagram as shown in the figure. The block diagram shows that the computer has four components: memory, input, output and the central processing unit (CPU). The CPU consists of the arithmetic and logic unit (ALU) and the control unit (CU). In the late 1960s, the CPU was designed with discrete components on various boards. With the advent of integrated circuit technology, it became possible to build the CPU on a single chip. This single chip came to be known as the microprocessor.

Evolution and Generations of microprocessors

Microprocessors are categorized into five generations: first, second, third, fourth, and fifth generations.

First generation

The microprocessors that were introduced in 1971 to 1972 were referred to as the first generation systems. First-generation microprocessors processed their instructions serially—they fetched the instruction, decoded it and then executed it. When an instruction was completed, the microprocessor updated the instruction pointer and fetched the next instruction, performing this sequential drill for each instruction in turn. Examples of microprocessor of this generation are Intel's 4 bit microprocessors 4004 and 4040.

Second generation

Motorola's MC68000 and Intel's 8080/8085 are examples of microprocessors of the second generation. The microprocessors of this generation were mostly **8-bit and introduced pipelined instruction processing**. As the first instruction is processed in the execution unit, the second instruction is decoded and the third instruction is fetched.

The distinction between the first and second generation devices was primarily the use of newer semiconductor technology to fabricate the chips. This new technology resulted in a five-fold increase in instruction, execution, speed, and higher chip densities.

Third generation

The third generation, introduced in 1978, was represented by Intel's 8086 and the Zilog Z8000, which were 16-bit processors with minicomputer-like performance. These microprocessors contain almost 250,000 transistors.

Intel 80186 and 80286 also belong to this generation.

Fourth generation

In 1980, the fourth-generation microprocessors evolved. Intel introduced 32-bit microprocessors 80386/80486, and the Motorola has introduced 68020/68030/68040 processors. These processors included an **on-chip RAM called the cache memory** to speed up program execution. Microprocessors of this generation had few million transistors.

Fifth generation

Microprocessors in their fifth generation contain more than 10 million transistors. Examples of fifth generation microprocessor are Intel's Pentium series of processors.

Microprocessor Systems

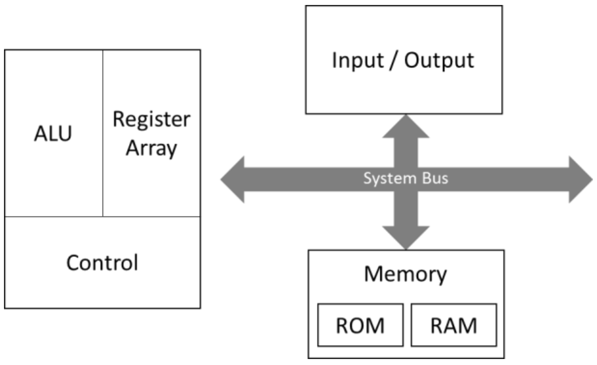
Role of a microprocessor

The microprocessor performs the following functions:

- Fetch, decode (interpret) and execute program instructions in a proper order.
- Transfer data to and from memory and to and from the input / output devices.
- Respond to external interrupts.
- Provide overall timing and control signals for the entire system.

Organization of a microprocessor based system

The next figure shows a simplified structure of a microprocessor based system. The three components of the system – microprocessor, memory and input / output – are organized around a common communication path called a bus. The system bus is nothing but a group of wires to carry bits (data).



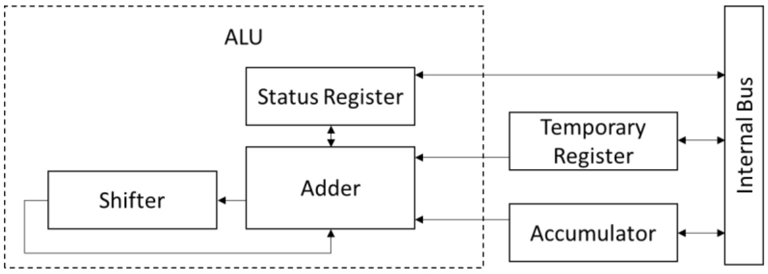
The microprocessor itself can be divided into three segments:

- Arithmetic and logic unit.
- Control unit.
- Register array.

Microprocessor

Arithmetic and logic unit

This is the area of the microprocessor where various computing functions are performed on the data. Arithmetic operations include operations such as addition and subtraction while logic operations include operations like AND, OR and XOR.



A simple ALU contains following blocks:

- Adder: It performs arithmetic operations like addition, subtraction, increment, decrement, etc. The result of operation is stored into accumulator.
- Shifter: It performs logical operations like rotate left, rotate right, etc. The result of operation is again stored into accumulator.
- Status Register: Also known as flag register. It contains a no. of flags either to indicate conditions arising after last ALU operation or to control certain operations.

Control unit

The control unit provides the necessary timing and control signals to all the operations of the microprocessor. It controls the flow of data between the microprocessor, the memory and the peripheral devices.

The timing and control unit is probably one of the most complex sections of a microprocessor. It affects and sequences all the events within the CPU and the entire microcomputer. Each program instruction can be divided into fetch, decode and execute stage. Each of these stages can be further subdivided into a series of steps that might be referred to as a micro program. The micro program for each instruction resides in the instruction decoding section and is executed by the control and timing unit of the microprocessor.

Register array

This area of the microprocessor consists of various registers identified by letters B, C, D, E, H and L. These registers are primarily used to store data temporarily during execution.

Memory

Memory stores such binary information as instructions and data, and provides that information to the microprocessor whenever necessary. To execute programs, the microprocessor reads instructions and data from memory and performs the computing operations in its ALU section. Results are either transferred to the output section for display or stored in memory for later use. The memory block shown in figure has two sections:

- Read-Only memory (ROM)
- Read/Write memory (R/WM), popularly known as Random-Access memory (RAM).

The ROM is used to store programs that do not need alterations. The monitor program of a single-board microcomputer is generally stored in the ROM.

The Read/Write memory (R/WM) is also known as user memory. It is used to store user programs and data. The information stored in this memory can be easily read and altered.

Input/Output

The third component of a microprocessor-based system is I/O (input/output); it communicates with the outside world. I/O includes two types of devices: input and output; these I/O devices are also known as peripherals. The input devices such as a keyboard, switches, and an analog-to-digital (A/D) converter transfer binary information (data and instructions) from the outside world to the microprocessor. The output devices transfer data from the microprocessor to the outside world. They include devices such as light emitting diodes (LEDs), or video screen, a printer, and digital-to-analog (D/A) converter.

System Bus

The system bus is a communication path between the microprocessor and peripherals; it is nothing but a group of wires to carry bits. All peripherals (and memory) share the same bus. However, the microprocessor communicates with only one peripheral at a time.

Generic microprocessor

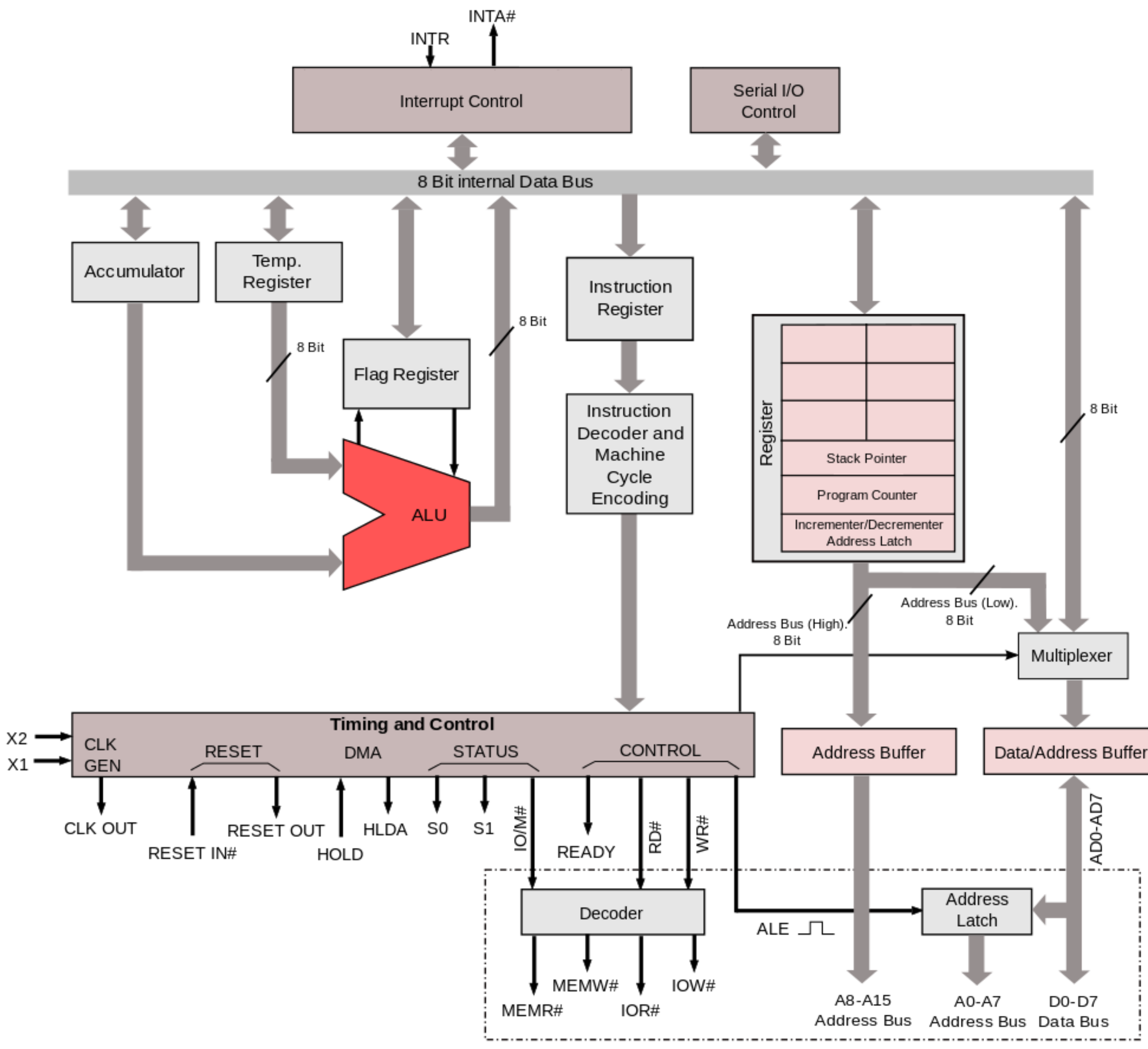
A microprocessor consists of the following key elements:

- Arithmetic and logic unit: Performs arithmetic and logical operations. The result operation is stored in accumulator. The flag register (status register) is set according to the result of the operation.
- Accumulator: The accumulator is used to store data and perform arithmetic and logical operations. The result of an operation is stored in the accumulator.
- Registers
  - Flag register: The flag register consists of bits that indicate the nature of the data stored in the accumulator.
  - Temporary registers: These registers are used internally by the microprocessor and are not available to the programmer.
  - General purpose registers: These registers are available to the programmer to use in his/her program.

- Program counter: This register is used to store the address from where the next instruction should be fetched.
- Stack Pointer: Used to store the address of the top of the stack.

- Instruction register and decoder: The instruction register stores the instruction which is then decoded by the decoder.
- Timing and control unit: Provides overall timing, coordinates the execution of instructions and controls the interfacing with external peripheral devices.
- Interrupt control unit.
- Serial I/O control unit.

The figure below shows the block diagram of a **generic microprocessor**.



8085 Operations and Architecture

The microprocessor has a set of instructions that are designed in the microprocessor to manipulate data and communicate with peripherals. This process of data manipulation and communication is determined by architecture (logic design) of the microprocessor.

The various operations of the microprocessor can be classified into three categories:

- Microprocessor initiated instructions.
- Internal operations.
- Peripheral (externally initiated) operations.

In this section we will look at how the internal structure of the 8085 microprocessor allows it to perform is tasks.

Microprocessor initiated operations and 8085 bus organization

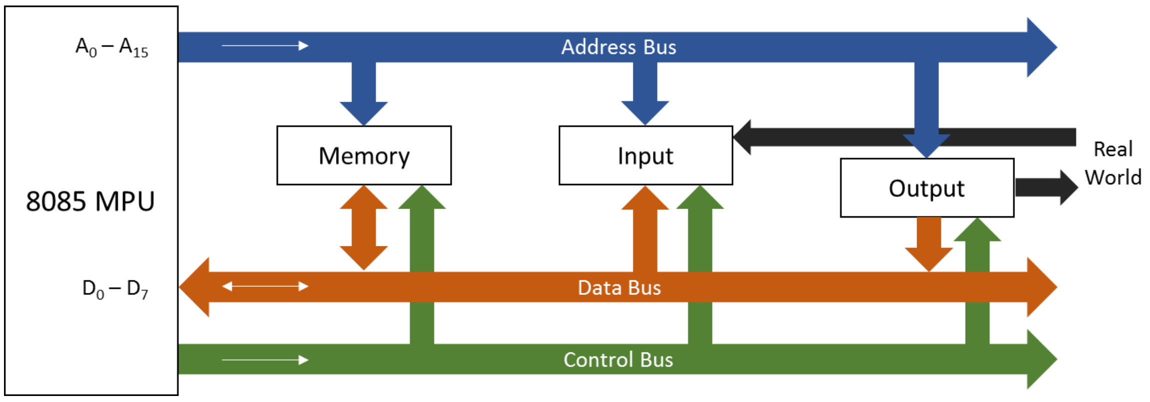
The microprocessor performs primarily four operations:

1. Memory read: Read data or instructions from memory.
2. Memory write: Write data or instructions into memory.
3. I/O read: Accept data from input devices.
4. I/O write: Send data to output devices.

All these operations include communication between the microprocessor and peripheral devices and memory. To communicate with a peripheral device or memory, the microprocessor needs to perform the following steps:

1. Identify the peripheral device or a memory location using its address.
2. Transfer binary information (data or instruction).
3. Provide timing and synchronization signals.

The 8085 microprocessor performs these steps using three sets of communication lines called buses: address bus, data bus and control bus. These buses are together known as the system bus.



Address bus

8085 has 16 address lines identified as A_0 to A_{15} and is capable of addressing 2^{16} = 65536 address (64K). However not every microprocessor system has 64K memory.

The address bus is unidirectional – bits flow the microprocessor to peripheral devices of memory. The microprocessor uses the address bus to perform the first function – identify the peripheral device or a memory location.

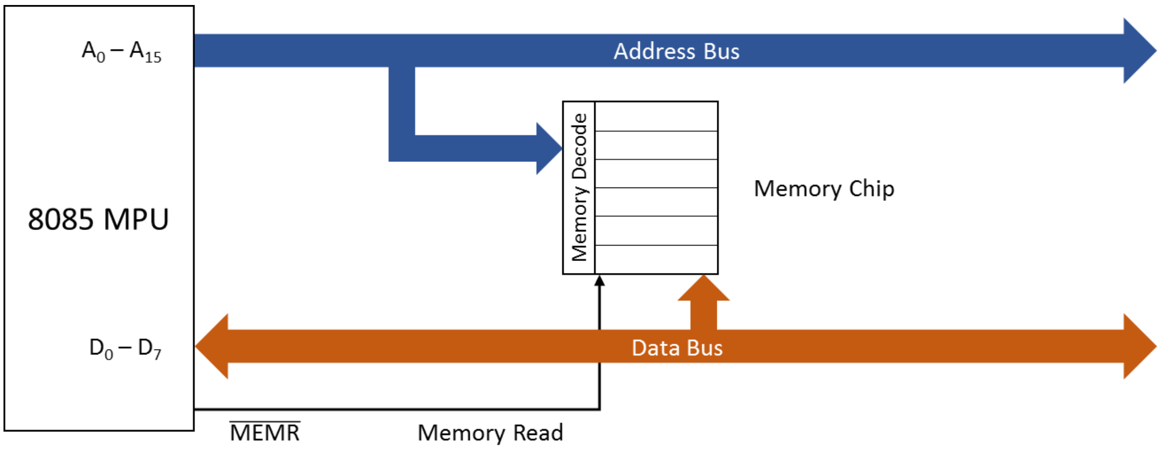
Data bus

The data bus is a group of 8 lines used for data flow. These lines are generally identified as D_0 to D_7. These lines are bidirectional – data flows in both directions between the microprocessor and the peripheral device or memory. The microprocessor uses the data bus to perform the second function – transfer binary information.

Control bus

The control bus consists of individual lines that provide a pulse to indicate a microprocessor operation. These signals are used to identify a device with which the microprocessor intends to communicate. The microprocessor generated specific control signals for every operation it performs.

Steps involved in reading information from memory



The microprocessor performs the following steps to read an instruction (or data) from a memory location:

1. The microprocessor places a 16 bit address on the address bus.
2. This address is decoded by an external logic circuit (similar to a de-multiplexer) and one memory address is identified.
3. The microprocessor then sends a pulse called Memory Read as a control signal. This pulse activates the memory chip and the contents of the memory address are placed on to the data bus and brought inside the microprocessor.

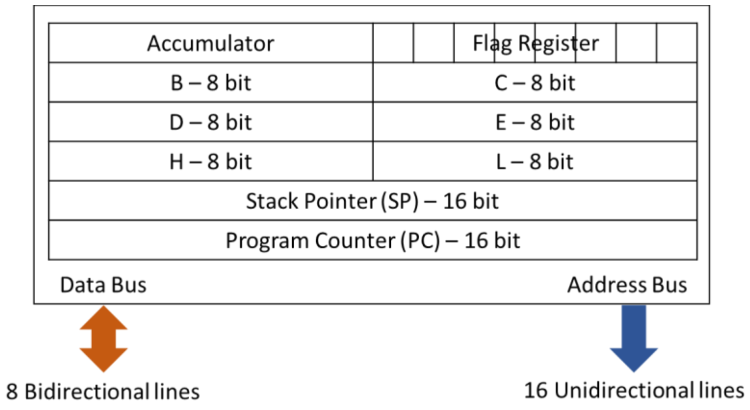
The steps involved to read a byte from an input device are similar. However, the microprocessor uses another control signal to indicate whether it wants to read data from the memory of from an input device.

Internal data operations and 8085 registers

The internal architecture of 8085 determines how and what operations can be performed with the data. These operations include:

- Store 8 bit data.
- Perform arithmetic and logical operations.
- Test for conditions.
- Sequence the execution of instructions.
- Store data temporarily during execution.

To perform these operations, the microprocessor requires registers, ALU, CU and internal buses. The next figure gives a simplified representation of 8085 internal architecture.



Registers
8085 has 6 general purpose registers to perform store 8 bit data during execution. These registers are identified as B, C, D, E, H and L. They can be combined as register pairs – BC, DE and HL – to perform some 16 bit operations.

Accumulator
The accumulator is an 8 bit register which is a part of the ALU. This register is used to store 8 bit data and perform arithmetic and logical operations. The result of an operation is stored in the accumulator. The accumulator is also identified as register A.

Flags
The ALU included 5 flip-flops that are set (1) or reset (0) according to the result of an operation. The microprocessor uses these to test for conditions.

S	Z	-	AC	-	P	-	CY

The five flag registers include:

- S (Sign flag): This flag is set to 1 if the number in the accumulator is negative.
- Z (Zero flag): The zero flag is set to 1 if the ALU operation results in 0. The flag is modified by results in the accumulator as well as the other registers.
- P (Parity flag): This flag is used to indicate the parity of the result in the accumulator. It is set to 1 if the result has an even number of 1s.
- CY (Carry flag): The carry flag is set if an arithmetic operation results in a carry.
- AC (Auxiliary Carry flag): This flag is used internally for BCD operations and not available to the programmer to change execution sequence of program. It is set to 1 if an arithmetic operation results in a carry from bit 3 to bit 4.

Program counter
The program counter is a 16 bit register. It helps the microprocessor to sequence the execution of instructions. This register is a memory pointer. Since memory address are 16 bits, this register is also 16 bit. The function of the program counter is to point to the memory address from which the next instruction is to be fetched. When a byte is being fetched, the program counter is incremented by 1 to point to the next memory location.

Stack pointer
The stack pointer is also a 16 bit register which is used a memory pointer. The stack pointer points to a memory location in the RAM called the stack. The stack is used to temporarily store data.

Peripheral or externally initiated operations
External devices (or signals) can initiate the following operations: reset, interrupt, ready and hold. Individual pins are assigned for each of these on the microprocessor.

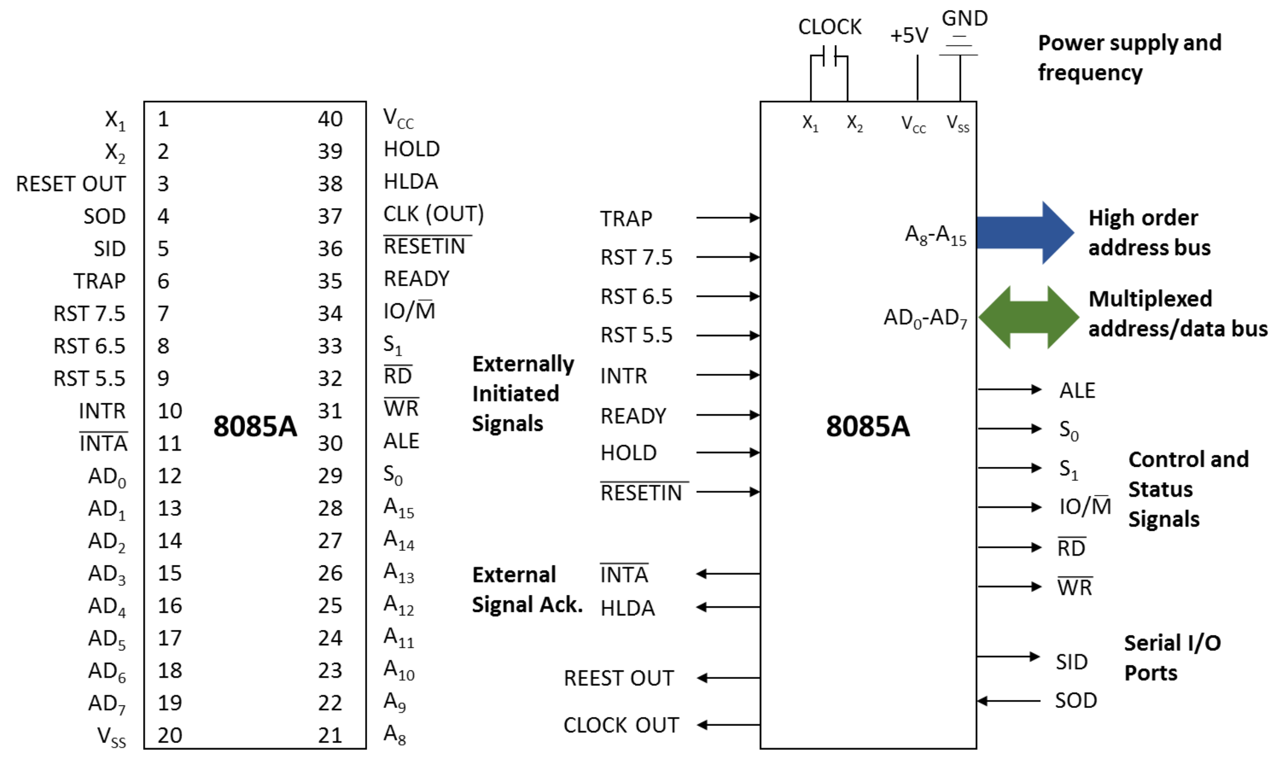
Reset
When the reset key is activated, all internal operations are suspended and PC is cleared (set to 0000H). Now the program execution can again being at the zero memory address.

Interrupt
The microprocessor can be interrupted from the normal execution of instructions and asked to execute some other instructions called a service routine. The microprocessor resumes its operations after completing the service routine.

Ready
When this signal is low, the microprocessor enters in to a Wait state. This signal is used to synchronize slower peripherals with the microprocessor.

Hold
When the HOLD pin is activated by an external signal, the microprocessor relinquishes control of buses and allows the external peripheral to use them. (e.g. used for direct memory access).

8085 pinout and signals



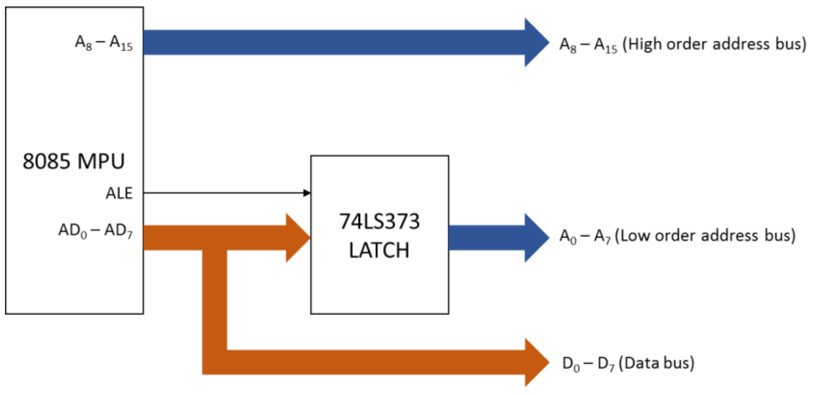
8085 microprocessor has 40 pins, requires a +5 V single power supply and can operate with a 3 MHz single phase clock. The 8085A-2 version can operate at a maximum frequency of 5 MHz. The above diagram shows the pinout of the 8085 microprocessor. All 8085 signals can be classified into 6 groups: address bus, data bus, control and status signals, power supply and frequency signals, externally initiated signals and serial I/O ports.

Address bus

The 8085 has 8 signal lines, A_{15}-A_8 which are unidirectional and used as the high order address bus.

Multiplexed address and data bus

The signal lines AD_7-AD_0 are bidirectional. They serve a dual purpose. They are used as the low order address bus as well as the data bus. In executing an instruction, during the earlier part of the cycle, these lines are used as the low order address bus. During the later part of the cycle, these lines are used as the data bus. The low order address bus can be separated by using a latch.


A8 – A15
8085 MPU
ALE
AD0 – AD7
74LS373 LATCH
A8 – A15 (High order address bus)
A0 – A7 (Low order address bus)
D0 – D7 (Data bus)

The above figure shows how the low order address bits are separated using a latch like 74LS373. The latch consists of 8 D flip-flops which are activated using the ALE signal. The input given to each flip-flop is latched and is available as the output of the flip-flop.

Control and Status Signals

This group includes 2 control signals ($\overline{RD}$ and $\overline{WR}$), 3 status signals ($IO/\overline{M}$, S_1 and S_0) to identify the nature of the operation and 1 special signal (ALE) to indicate the beginning of the operation.

ALE (Address Latch Enable): This positive going pulse is generated every time 8085 begins an operation. It indicates that the bits of AD_7-AD_0 are address bits. This signal is primarily used to latch the low order address from the multiplexed bus and generate a separate set of eight address lines A_7-A_0.

$\overline{RD}$ (Read): This signal indicates that data has to be read from the selected I/O device or memory and data is available on the data bus. It is active low.

$\overline{WR}$ (Write): This signal indicates that data on the data bus has to be written to the selected I/O device or memory. It is active low.

IO/$\overline{M}$ (Input / Output or Memory): This signal is used to differentiate between I/O and memory operations. When it is high, it indicates an I/O operation and when it is low it indicates a memory operation.

S_1 and S_0 (Status): Similar to the IO/$\overline{M}$ signal, these can be used to identify various operations. But they are rarely used in small systems.

Machine Cycle	**Status Signals**			**Control Signals**
	IO/$\overline{M}$	**S_1**	**S_0**	
Opcode Fetch	0	1	1	$\overline{RD}$ = 0
Memory Read	0	1	0	$\overline{RD}$ = 0
Memory Write	0	0	1	$\overline{WR}$ = 0
I/O Read	1	1	0	$\overline{RD}$ = 0
I/O Write	1	0	1	$\overline{WR}$ = 0
Interrupt Ack.	1	1	1	$\overline{INTA}$ = 0
Halt	Z	0	0	$\overline{RD}$ = Z $\overline{WR}$ = Z $\overline{INTA}$ = 1

Power supply and clock frequency

The power supply and frequency signals are as follows:

- V_{CC}: +5 V power supply.
- V_{SS}: Ground reference.
- X_1 and X_2: A crystal (or an oscillator) is connected between these two pins. The frequency is internally divided by 2. Therefore to operate the system at 3 MHz, the crystal should have a frequency of 6 MHz.
- CLK OUT (Clock Output): This signal can be used as the system clock for other devices.

Externally initiated signals

The 8085 has 5 interrupt signals that can be used to interrupt a program execution. The microprocessor acknowledges an interrupt request by the $\overline{INTA}$ (Interrupt Acknowledge) signal. Three pins, RESET, HOLD and READY accept the externally initiated signals as inputs. To respond to HOLD request, the 8085 has one signal called HLDA (Hold Acknowledge).

- INTR (Interrupt Request): This is used as a general purpose interrupt.
- $\overline{INTA}$ (Interrupt Acknowledge): This is used to acknowledge an interrupt.
- RST 7.5, RST 6.5 and RST 5.5 (Restart Interrupts): These are vectored interrupts that transfer the program control to specific memory locations. They have higher priority that the INTR interrupt. Among these three the priority order is 7.5, 6.5 and 5.5.
- TRAP: This is a non-maskable interrupt and has the highest priority.
- HOLD: This signal indicates that a peripheral (like a Direct Memory Access controller) is requesting the use of the address and data bus.
- HLDA (Hold Acknowledge): This signal is generated by the microprocessor to acknowledge the HOLD request.
- READY: This signal is used to delay the microprocessor's read or write cycle until a slow responding peripheral is ready to send or accept data. When this signal goes low, the microprocessor waits for an integral number of clock cycle until this signal goes high.

- $\overline{\text{RESET IN}}$: When the signal on this pin goes low, the program counter is set to 0, the buses are tri-stated and the microprocessor is rest.
- RESET OUT: This signal indicates that the microprocessor is being reset. The signal can be used to reset other devices.

Serial I/O Ports

The 8085 has two signals to implement serial transmissions: SID (Serial Input Data) and SOD (Serial Output Data).

8085 machine cycles and bus timings

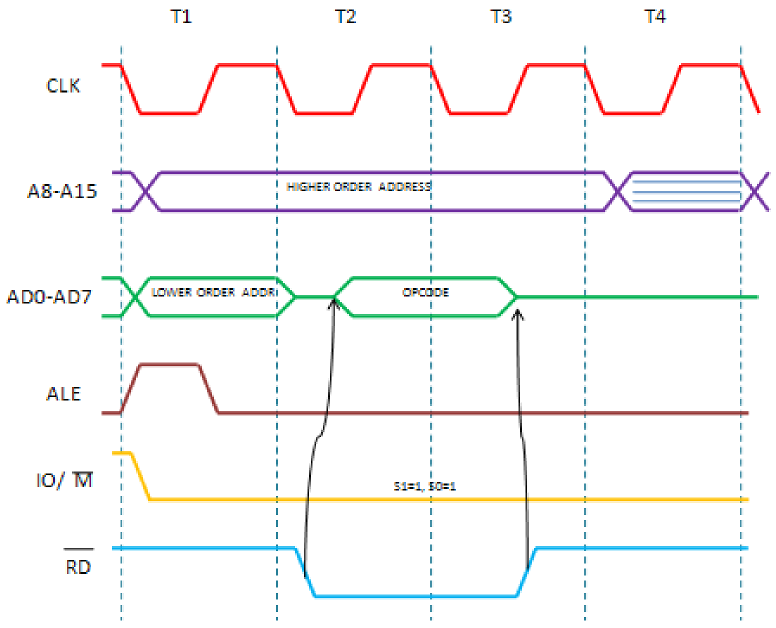
The 8085 is designed to execute 74 different instruction types. Each instruction has two parts: operation code, known as the opcode and the operand. The opcode is a command such as ADD and the operand is an object to be operated on such as a byte of data or the contents of a register.

Some instructions are 1 byte instructions and others are multi-byte instructions. To execute an instruction the 8085 needs to perform various operations like memory read/write and I/O read/write. There is no direct relationship between the number of bytes of an instruction and the number of operations the 8085 has to perform.

All instructions are divided into a few basic machine cycles and these machines cycles are divided into precise system clock periods. The machine cycles are nothing by the micro programs stored in the microprocessor which need to be executed to process an instruction. The different machine cycles of 8085 are: opcode fetch, memory read, memory write, I/O read, I/O write, interrupt acknowledge, halt, hold and reset.

The microprocessor may need to execute one more machine cycles to execute one instruction. This group of machine cycles constitutes one instruction cycle. **Each machine cycle is further divided into T-states. T-states are always of a given time interval that is determined by the clock frequency**. A machine cycle generally is made up of 3-4 T-states.

Opcode fetch machine cycle

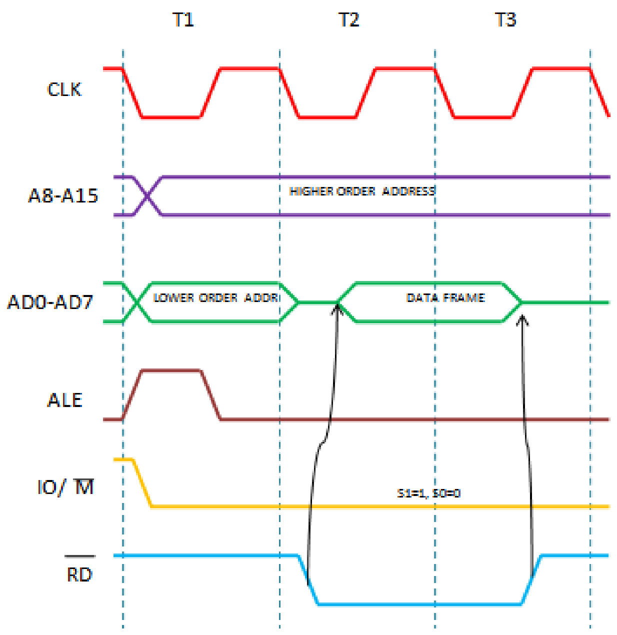


Opcode fetch cycle is part of any instruction execution. In this machine cycle 8085 fetches opcode of instruction. The following are the sequence of actions that are performed by 8085 to fetch an opcode from memory. This machine cycle consists of 4 T-states.

1. 8085 places 16-bit address from PC on to the address bus (of course lower order address bus is multiplexed with data bus) and issues ALE pulse in first T-state (T1). This is used to de-multiplex the address and data bus. It also issues IO/$\overline{M}$ signal to 0. This indicates that processor is performing memory related operation.
2. In second T-state (T2) processor issues $\overline{RD}$ control signal to memory. This enables memory to put data present at the address location given in previous T-state on to data bus. $\overline{RD}$ control signal is active for two clock pulses.

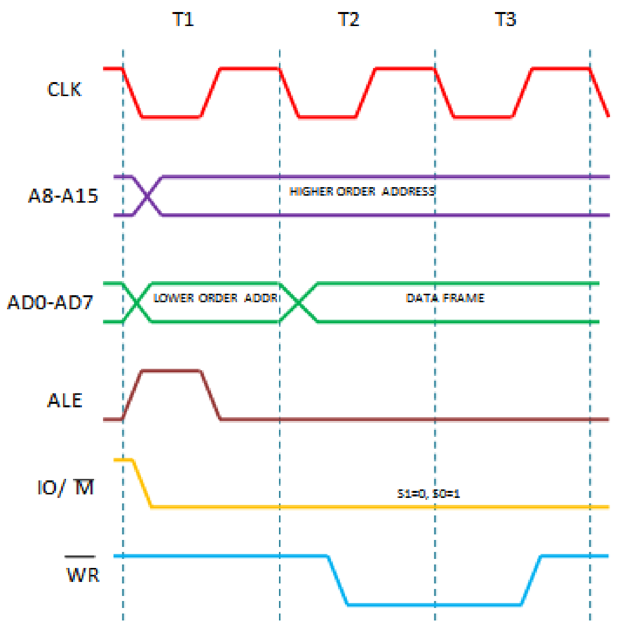
3. In T3 state memory places opcode on Data bus. Processor reads opcode present on data bus and de-asserts $\overline{RD}$ signal. Thus data bus goes into high impedance state.
4. In T4 state processor decodes instruction and necessary actions are performed.

Memory read machine cycle



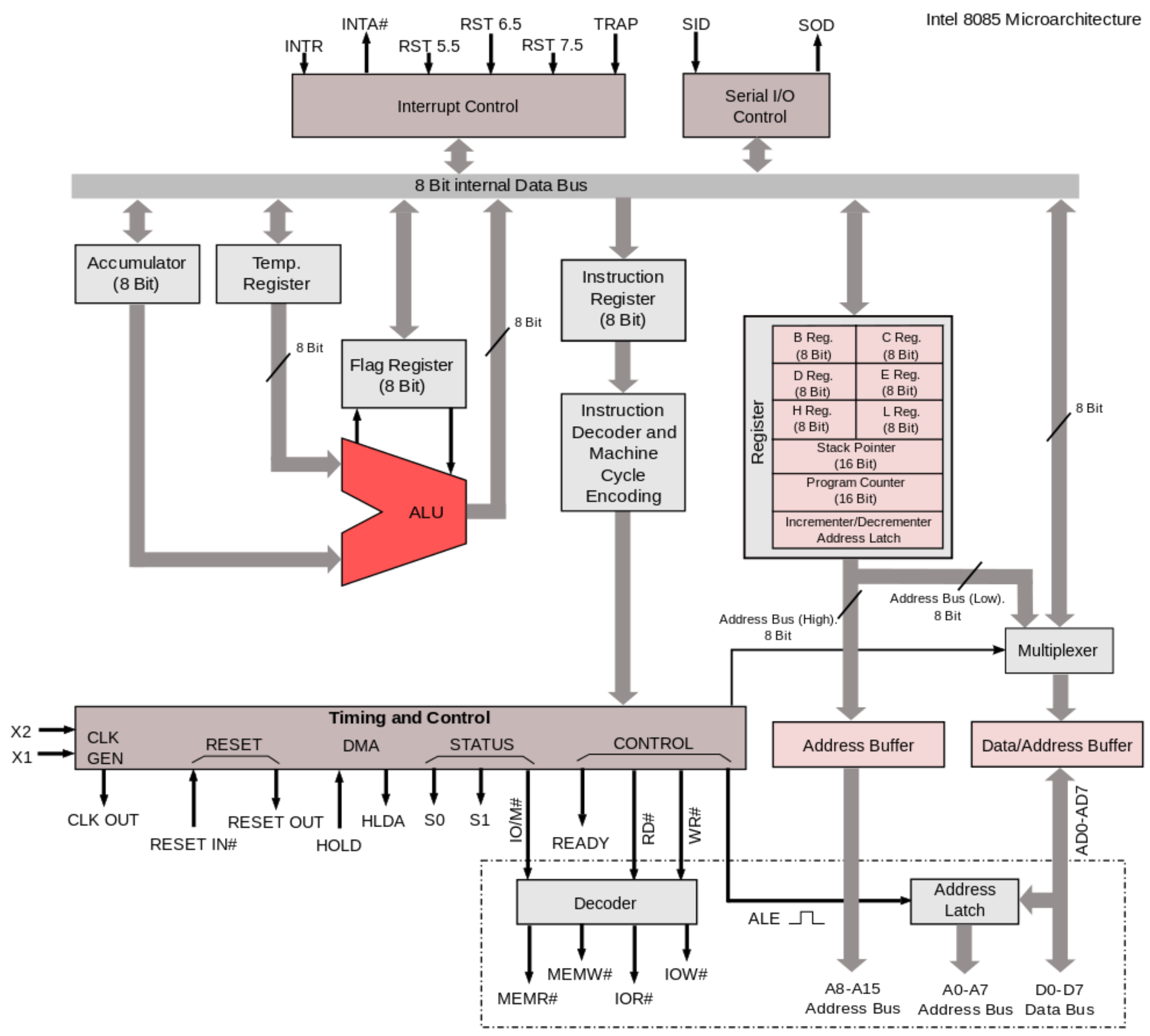
1. In the first T-state (T1) 8085 places address on address bus and issues ALE signal. And also IO/$\overline{M}$ signal is made low, since it is memory related operation.
2. In the second T-state (T2), processor issues $\overline{RD}$ control signal to memory. In response to this memory places data on data bus.
3. In the third T-state (T3), processor reads data from data bus, and de-asserts $\overline{RD}$ signal.

Memory write machine cycle



1. In first T-state (T1), 8085 processor places 16- bit address on address bus and issues ALE signal. And also it makes IO/$\overline{M}$ signal to low, indicating it is memory related operation.
2. In second T-state (T2), processor places data to be written on data bus and asserts $\overline{WR}$ signal to the memory.
3. In the third T-state (T3), memory stores the data and processor de-asserts $\overline{WR}$ signal.

Functional block diagram of 8085



Interfacing I/O devices with 8085

Memory mapped I/O

In memory mapped I/O devices are interfaced using address from memory space. That means IO device address are part of addresses given to memory locations.8085 uses 16-bit address to memory interfacing. So any address between 0000_H-$FFFF_H$ can be given to each peripheral. But the addresses given to peripheral can't be used for memory. Memory control signals are used as read and write control signals for peripherals. And all the operations that can be performed on memory can also be performed on peripherals. There is no need of using I/O instructions such as IN and OUT.

Peripheral mapped or I/O mapped I/O

In this method separate address space is given to IO devices. Each IO device is given an 8 bit address. Hence maximum 256 devices can be interfaced to the processor. The address range for the IO devices is 00_H-FF_H. I/O control signals are used to perform read, write operations. For reading data from I/O device or writing data to I/O device IN, OUT instructions needs to be used. Arithmetic and logical operations can't be performed directly on I/O devices as in memory mapped I/O. During I/O mapped I/O the address on the low address bits is duplicated on the high address bits of the address bus.

Interrupts

An interrupt is a signal or condition that causes processor to stop its normal execution flow and makes it to jump to some other location for processing the interrupt.

The interrupt I/O is a process of data transfer whereby an external device can inform the processor that it is ready for communication and it requests attention. The process is initiated by an external device and is asynchronous, meaning that it can be initiated at any time without any reference to the system clock. However, the response to the interrupt is directed or controlled by the microprocessor.

The interrupts in 8085 are classified in two categories: maskable interrupt and nonmaskable interrupt. The 8085 includes 4 maskable interrupts and 1 nonmaskable interrupt. Among the four maskable interrupts one is non-vectored and requires external hardware to supply a Call location to restart the execution. The other three are vectored to specific location.

The microprocessor can ignore or delay a maskable interrupt request if it is performing some critical task. However, it has to respond to a nonmaskable interrupt immediately. In many ways, the maskable interrupt is like a telephone, which can be kept off the hook if one is not interested in receiving any message. The nonmaskable interrupt is like a smoke detector, which should be attended to it set off.

Interrupt Name	Maskable	Masking Method	Vectored	Memory	Triggering Method
INTR	Yes	DI / EI	No	No	Level Sensitive
RST 5.5 / RST 6.5	Yes	DI / EI SIM	Yes	No	Level Sensitive
RST 7.5	Yes	DI / EI SIM	Yes	Yes	Edge Sensitive
TRAP	No	None	Yes	No	Level & Edge Sensitive

The interrupt process allows the microprocessor to respond to the external requests for attention on a demand basis and leaves the microprocessor free to perform other tasks.

Non-vectored Interrupt (INTR)

The 8085 interrupt process is controlled by the Interrupt Enable flip-flop, which is internal to the processor and can be set or reset by using software instructions. If the flip-flop is enabled and the input to the interrupt signal INTR (pin 10) goes high, the microprocessor is interrupted. This is a maskable interrupt and can be disabled. The 8085 has a nonmaskable and three additional vectored interrupt signals as well. The best way to describe the 8085 interrupt process is to compare it to a telephone with a blinking light instead of a ring.

Assume that you are reading an interesting novel at your desk, where there is a telephone. For you to receive and respond to a telephone call, the following steps should occur:

1. The telephone system should be enabled, meaning that the receiver should be on the hook.
2. You should glance at the light at certain intervals to check whether someone is calling.
3. If you see a blinking light, you should pick up the receiver, say hello, and wait for a response. Once you pick up the phone, the line is busy, and no more calls can be received until you replace the receiver.
4. Assuming that the caller is your roommate, the request may be: It is going to rain today. Will you please shut all the windows in my room?
5. You insert a bookmark on the page you are reading.
6. You replace the receiver on the hook.
7. You shut your roommate's windows.
8. You go back to your book, find your mark, and start reading again.

Steps 6 and 7 may be interchanged, depending on the urgency of the request. If the request is critical and you do not want to be interrupted while attending to the request, you are likely to attend to the request first, then put the receiver back on the hook. The 8085 interrupt process can be described in terms of those eight steps.

Step 1: The interrupt process should be enabled by writing the instruction EI in the main program. This is similar to keeping the phone receiver on the hook. The instruction EI sets the Interrupt Enable flip-flop. The instruction DI resets the flip-flop and disables the interrupt process.

Instruction EI (Enable Interrupt)

- This is a 1-byte instruction.
- The instruction sets the Interrupt Enable flip-flop and enables the interrupt process.
- System reset or an interrupt disables the interrupt process.

Instruction DI (Disable Interrupt)

- This is a 1-byte instruction.
- The instruction resets the Interrupt Enable flip-flop and disables the interrupt.
- It should be included in a program segment where an interrupt from an outside source cannot be tolerated.

Step 2: When the microprocessor is executing a program, it checks the INTR line during the execution of each instruction. (**The pin is checked one clock period before the last T-state of an instruction cycle**).

Step 3: If the line INTR is high and the interrupt is enabled, the microprocessor completes the current instruction, disables the Interrupt Enable flip-flop and sends a signal called $\overline{\text{INTA}}$ – Interrupt Acknowledge (active low). **The processor cannot accept any interrupt requests until the interrupt flip-flop is enabled again**.

Step 4: The signal $\overline{\text{INTA}}$ is used to insert a restart (RST) instruction (or a Call instruction) through external hardware. The RST instruction is a 1-byte call instruction (explained below) that transfers the program control to a specific memory location on page 00H and restarts the execution at that memory location after executing Step 5.

Step 5: When the microprocessor receives an RST instruction (or a Call instruction), it saves the memory address of the next instruction on the stack. This is similar to inserting a bookmark. The program is transferred to the CALL location.

Step 6: Assuming that the task to be performed is written as a subroutine at the specified location, the processor performs the task. This subroutine is known as a service routine.

Step 7: **The service routine should include the instruction EI to enable the interrupt again**. This is similar to putting the receiver back on the hook.

Step 8: At the end of the subroutine, the RET instruction retrieves the memory address where the program was interrupted and continues the execution. This is similar to finding the page where you were interrupted by the phone call and continuing to read.

RST (Restart) Instructions

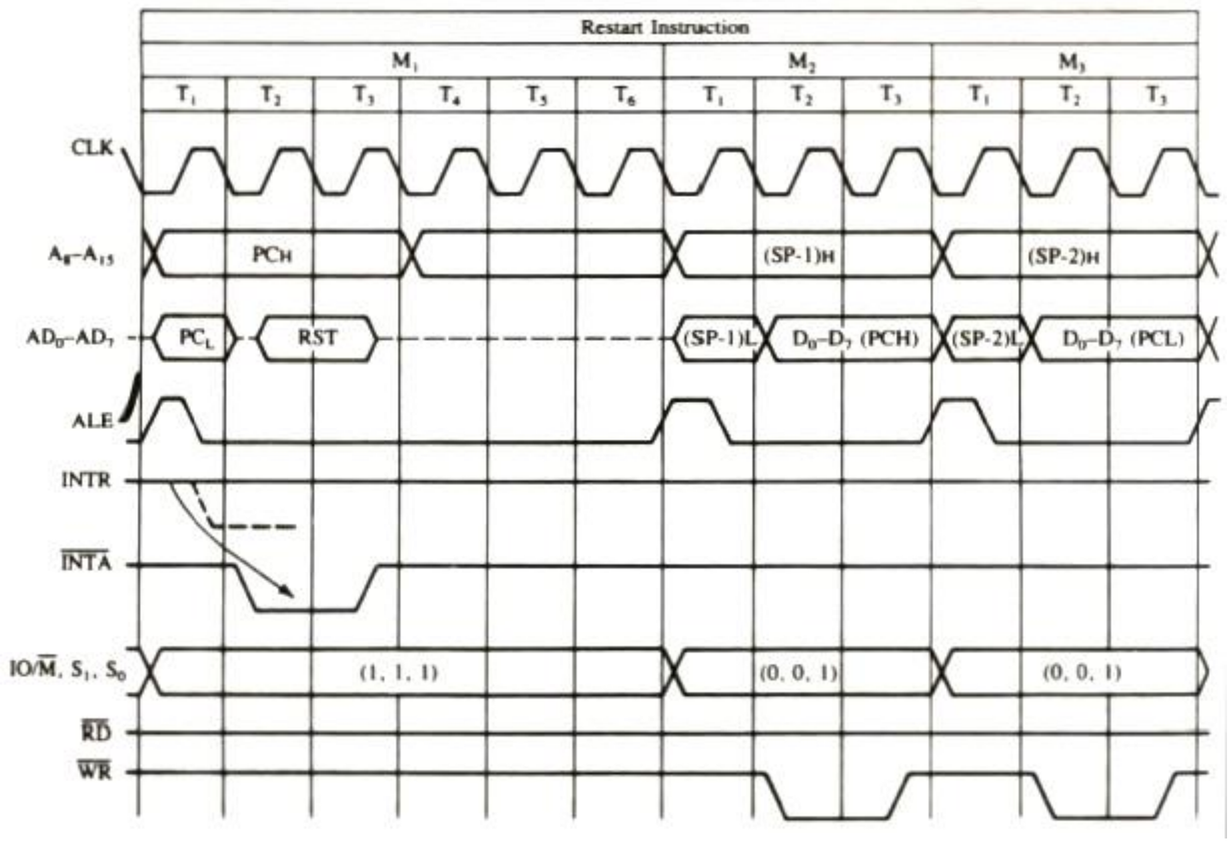
The 8085 instruction set includes eight RST (Restart) instructions. These are 1-byte Call instructions that transfer the program execution to a specific location on page 00H, as listed in table below. The RST instructions are executed in a similar way to that of Call instructions. The address in the program counter (meaning the address of the next instruction to an RST instruction) is stored on the stack before the program execution is transferred to the RST call location. When the processor encounters a Return instruction in the subroutine associated with the RST instruction, the program returns to the address that was stored on the stack. In case of a hardware interrupt, we will use an RST instruction to restart the program execution.

Mnemonics	Hex Code	Call location
RST 0	C7	0000
RST 1	CF	0008
RST 2	D7	0010
RST 3	DF	0018
RST 4	E7	0020
RST 5	EF	0028
RST 6	F7	0030
RST 7	FF	0038

The Step 4 in the interrupt process is executed by using external hardware and the signal $\overline{\text{INTA}}$ (Interrupt Acknowledge). In response to the INTR (Interrupt Request) high signal, the 8085 sends the $\overline{\text{INTA}}$ (Interrupt Acknowledge) low signal and the RST instruction is placed on the data bus during M_1. During M_1 the program counter holds the memory address of the next instruction, which should be stored on the stack so that the program can continue after the service routine. During M_2, the address of/the stack pointer minus one (SP - 1) location is placed on the address bus, and the high/order address of the program counter is stored on the stack. During M_3, the low-order address of the program counter is stored in the next location (SP - 2) of the stack.

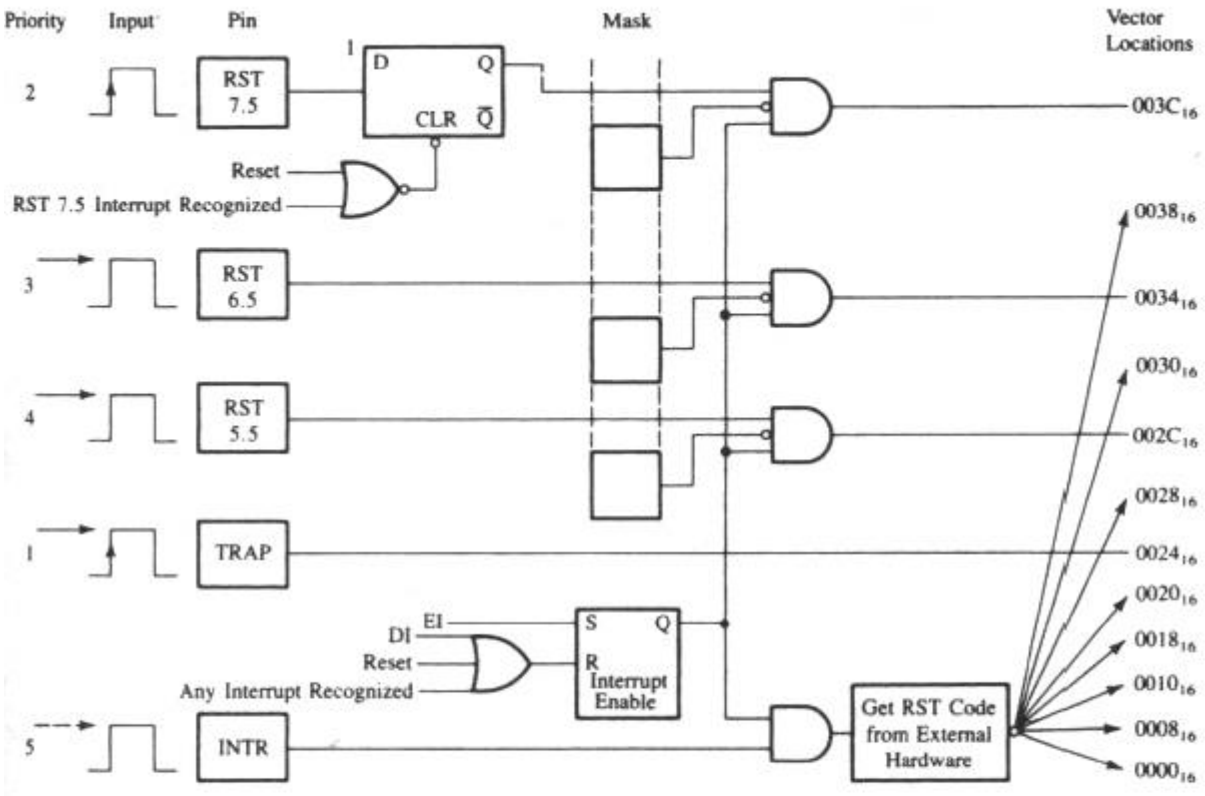
The machine cycle M_1 of the Interrupt Acknowledge is identical with the opcode fetch cycle, with two exceptions. The $\overline{\text{INTA}}$ signal is sent out instead of the RD signal, and the status lines (IO/$\overline{\text{M}}$, S_0 and S_1) are 1 1 1 instead of 0 1 1. During M_1 the RST is decoded and a 1-byte Call instruction to call location 0028H for the specific RST is executed. The machine cycles M_2 and M_3 are Memory Write cycles that store the contents of the program counter on the stack, and then a new instruction cycle begins.

In this next instruction cycle, the program is transferred to call location. The service routine is written somewhere else in memory, and the Jump instruction is written at the call location to specify the address of the service routine.



Vectored Interrupts

The 8085 has five interrupt inputs (Figure 12.5). One is called INTR (discussed in the previous section), three are called RST 5.5, 6.5, and 7.5, respectively, and the fifth is called TRAP, a nonmaskable interrupt. These last four (RSTs and TRAP) are automatically vectored (transferred) to specific locations on memory page 00H without any external hardware. They do not require the INTA signal or an input port; the necessary hardware is already implemented inside the 8085. These interrupts and their call locations are as follows:



The TRAP has the highest priority, followed by RST 7.5, 6.5, 5.5, and INTR, in that order.

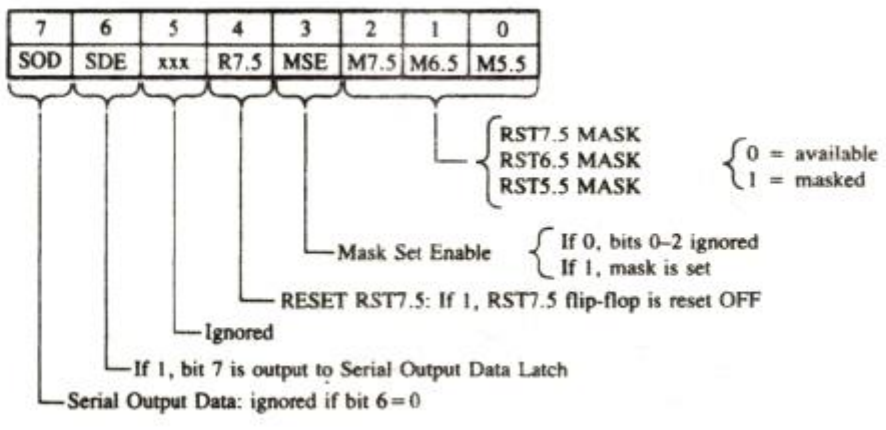
TRAP

TRAP, a nonmaskable interrupt known as NMI, is analogous to the smoke detector described earlier. It has the highest priority among the interrupt signals, it need not be enabled, and it cannot be disabled. It is level- and edge-sensitive, meaning that the input should go high and stay high to be acknowledged. It cannot be acknowledged again until it makes a transition from high-to-low-to-high. TRAP is generally used for such critical events as power failure and emergency shut-off.

RST 7.5, 6.5, and 5.5

These maskable interrupts are enabled under program control with I/O instructions: EI (Enable Interrupt) described earlier, and SIM (Set Interrupt Mask) described below.

SIM Instruction


7 6 5 4 3 2 1 0
SOD SDE xxx R7.5 MSE M7.5 M6.5 M5.5
RST7.5 MASK
RST6.5 MASK
RST5.5 MASK
0 = available
1 = masked
Mask Set Enable
If 0, bits 0–2 ignored
If 1, mask is set
RESET RST7.5: If 1, RST7.5 flip-flop is reset OFF
Ignored
If 1, bit 7 is output to Serial Output Data Latch
Serial Output Data: ignored if bit 6 = 0

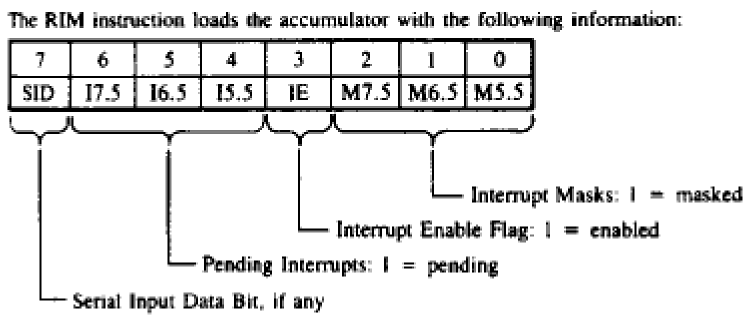
SIM: Set Interrupt Mask. This is a 1-byte instruction and can be used for three different functions:

- One function is to set mask for RST 7.5, 6.5, and 5.5 interrupts. This instruction reads the content of the accumulator and enables or disables the interrupts according to the content of the accumulator. Bit D3 is a control bit and should = 1 for bits D_0, D_1 and D_2 to be effective. Logic 0 on D_0, D_1, and D_2 will enable the corresponding interrupts, and logic 1 will disable the interrupts.
- The second function is to reset RST 7.5 flip-flop. Bit D_4 is additional control for RST 7.5. If D_4 = 1, RST 7.5 is reset. This is used to override (or ignore) RST 7.5 without servicing it.
- The third function is to implement serial I/O. Bits D_7 and D_6 of the accumulator are used for serial I/O and do not affect the interrupts. Bit D_6 = 1 enables the serial I/O and bit D_7 is used to transmit (output) bits.

Pending Interrupts

Because there are several interrupt lines, when one interrupt request is being served, other interrupt requests may occur and remain pending. The 8085 has an additional instruction called RIM (Read Interrupt Mask) to sense these pending interrupts.

RIM Instruction


The RIM instruction loads the accumulator with the following information:
7 6 5 4 3 2 1 0
SID I7.5 I6.5 I5.5 IE M7.5 M6.5 M5.5
Interrupt Masks: 1 = masked
Interrupt Enable Flag: 1 = enabled
Pending Interrupts: 1 = pending
Serial Input Data Bit, if any

RIM: Read Interrupt Mask. This is a 1-byte instruction that can be used for the following functions.

- To read interrupt masks. This instruction loads the accumulator with 8 bits indicating the current status of the interrupt masks
- To identify pending interrupts. Bits D_4, D_5, and D_6 identify the pending interrupts.
- To receive serial data. Bit D_7 is used to receive serial data.

Stack

The stack is a group of memory locations in the R/W memory that is used for temporary storage of binary information during the execution of a program. The starting memory location of the stack is defined in the main program, and space is reserved, usually at the high end of the memory map. The method of information storage resembles a stack of books. The contents of each memory location are, in a sense, "stacked"—one memory location above another - and information is retrieved starting from the top. Hence, this particular group of memory locations is called the stack.

Stack in 8085

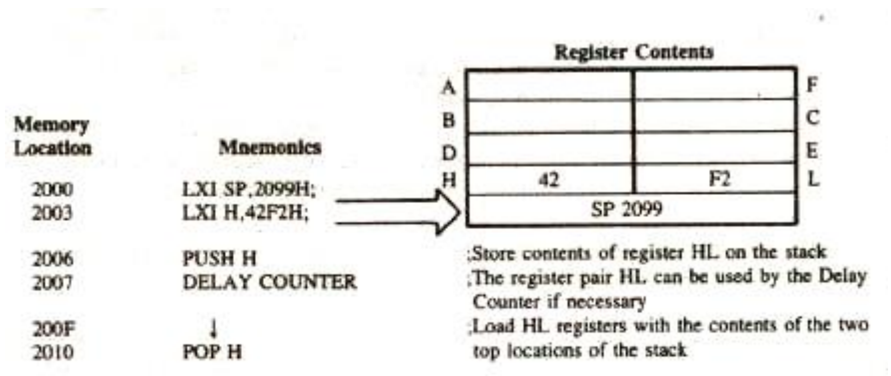
The stack in an 8085 microcomputer system can be described as a set of memory locations in the R/W memory, specified by a programmer in a main program. These memory locations are used to store binary information (bytes) temporarily during the execution of a program.

The beginning of the stack is defined in the program by using the instruction LXI SP, which loads a 16-bit memory address in the stack pointer register of the microprocessor. **Once the stack location is defined, storing of data bytes begins at the memory address that is one less than the address in the stack pointer register**. For example, if the stack pointer register is loaded with the memory address 2099H (LXI SP,2099H), the storing of data bytes begins at 2098H and continues in reversed numerical order (decreasing memory addresses such as 2098H, 2097H, etc.). Therefore, as a general practice**, the stack is initialized at the highest available memory location to prevent the program from being destroyed by the stack information**. The size of the stack is limited only by the available memory.

Data bytes in the register pairs of the microprocessor can be stored on the stack (**two at a time**) in reverse order (decreasing memory address) by using the instruction PUSH. Data bytes can be transferred from the stack to respective registers by using the instruction POP. The stack pointer register tracks the storage and retrieval of the information.

Because two data bytes are being stored at a time, the 16-bit memory address in the stack pointer register is decremented by two; when data bytes are retrieved, the address is incremented by two. An address in the stack pointer register indicates that the next two memory locations (in descending numerical order) can be used for storage.

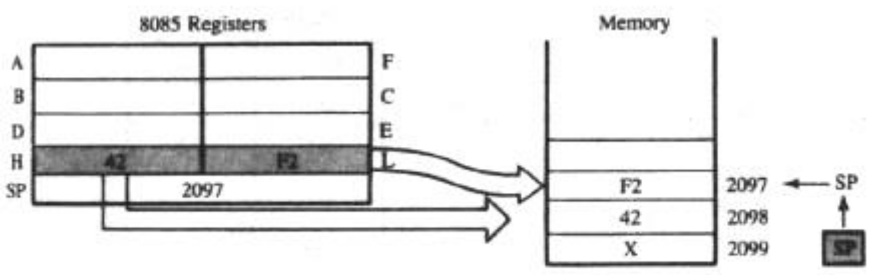
The stack is shared by the programmer and the microprocessor. The programmer can store and retrieve the contents of a register pair by using PUSH and POP instructions. Similarly, the microprocessor automatically stores the contents of the program counter when a subroutine is called.



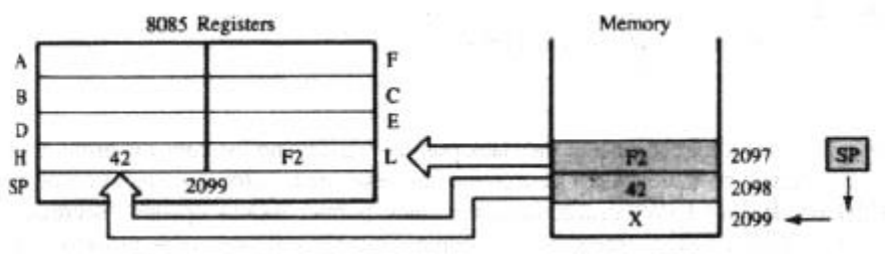
```
Memory
Location    Mnemonics
 2000       LXI SP,2099H;
 2003       LXI H,42F2H;

 2006       PUSH H              ;Store contents of register HL on the stack
 2007       DELAY COUNTER       ;The register pair HL can be used by the Delay
                                 Counter if necessary
 200F           ↓               ;Load HL registers with the contents of the two
 2010       POP H                top locations of the stack
```

Contents of stack after PUSH instruction:



Contents of stack after POP instruction:



8085 instructions for working with stack

The following instructions of 8085 involve the use of stack or the stack pointer:

- LXI SP
- PUSH rp
- POP rp
- CALL
- RET
- SPHL
- DAD SP

Subroutine

A subroutine is a group of instructions that performs a subtask (e.g., time delay or arithmetic operation) of repeated occurrence. The subroutine is written as a separate unit, apart from the main program, and the microprocessor transfers the program execution from the main program to the subroutine whenever it is called to perform the task. After the completion of the subroutine task, the microprocessor returns to the main program. The subroutine technique eliminates the need to write a subtask repeatedly; thus, it uses memory more efficiently.

Before implementing the subroutine technique, the stack must be defined; the stack is used to store the memory address of the instruction in the main program that follows the subroutine call.

Subroutine in 8085

The 8085 microprocessor has two instructions to implement subroutines: CALL (call a subroutine), and RET (return to main program from a subroutine). The CALL instruction is used in the main program to call a subroutine, and the RET instruction is used at the end of the subroutine to return to the main program. When a subroutine is called, the contents of the program counter, which is the address of the instruction following the CALL instruction, is stored on the stack and the program execution is transferred to the subroutine address. When the RET instruction is executed at the end of the subroutine, the memory address stored on the stack is retrieved, and the sequence of execution is resumed in the main program.

Restart, Conditional Call and Return instructions

Restart (RST) instructions

RST instructions are 1-byte Call instructions that transfer the program execution to a specific location on page 00H. They are executed the same way as Call instructions. When an RST instruction is executed, the 8085 stores the contents of the program counter (the address of the next instruction) on the top of the stack and transfers the program to the Restart location. These instructions are generally used in conjunction with the interrupt process.

Conditional Call and Return Instructions

The conditional Call and Return instructions are based on four data conditions (flags): Carry, Zero, Sign, and Parity. The conditions are tested by checking the respective flags. In case of a conditional Call instruction, the program is transferred to the subroutine if the condition is met; otherwise, the main program is continued. In case of a conditional Return instruction, the sequence returns to the main program if the condition is met; otherwise, the sequence in the subroutine is continued. If the Call instruction in the main program is conditional, the Return instruction in the subroutine can be conditional or unconditional. The conditional Call and Return instructions are listed for reference.

- CC: Call subroutine if Carry flag is set (CY = 1)
- CNC: Call subroutine if Carry flag is reset (CY = 0)
- CZ: Call subroutine if Zero flag is set (Z = 1)
- CNZ: Call subroutine if Zero flag is reset (Z = 0)
- CM: Call subroutine if Sign flag is set (S = 1, negative number)
- CP: Call subroutine if Sign flag is reset (S = 0, positive number)
- CPE: Call subroutine if Parity flag is set (P = 1, even parity)
- CPO: Call subroutine if Parity flag is reset (P = 0, odd parity)

- RC: Return if Carry flag is set (CY = 1)
- RNC: Return if Carry flag is reset (CY = 0)
- RZ: Return if Zero flag is set (Z = 1)
- RNZ: Return if Zero flag is reset (Z = 0)
- RM: Return if Sign flag is set (S = 1, negative number)
- RP: Return if Sign flag is reset (S = 0, positive number)
- RPE: Return if Parity flag is set (P = 1, even parity)
- RPO: Return if Parity flag is reset (P = 0, odd parity)

Advanced Microprocessors

The Microprocessor Age

The world's first microprocessor, the Intel 4004, was a **4-bit microprocessor**–programmable controller on a chip. It addressed a mere 4096, 4-bit-wide memory locations. The 4004 instruction set contained only **45 instructions**. The main problems with this early microprocessor were its speed, word width, and memory size. The evolution of the 4-bit microprocessor ended when Intel released the 4040, an updated version of the earlier 4004. The 4040 operated at a higher speed, although it lacked improvements in word width and memory size.

Later in 1971, realizing that the microprocessor was a commercially viable product, Intel Corporation released the 8008—an extended **8-bit version** of the 4004 microprocessor. The 8008 addressed an expanded memory size (**16K bytes**) and contained additional instructions (**a total of 48**) that provided an opportunity for its application in more advanced systems.

Intel introduced the 8080 microprocessor in 1973. It was the first modem 8-bit microprocessor. 8080 could address 4 times more memory (**64K bytes**), execute additional instructions and was almost 10 times faster than the 8008.

In 1977, Intel Corporation introduced an updated version of the 8080—the 8085. The 8085 was to be the last 8-bit, general-purpose microprocessor developed by Intel. Although only slightly more advanced than an 8080 microprocessor, the 8085 executed software at an even higher speed.

The Modern Microprocessor

8086 and 8088

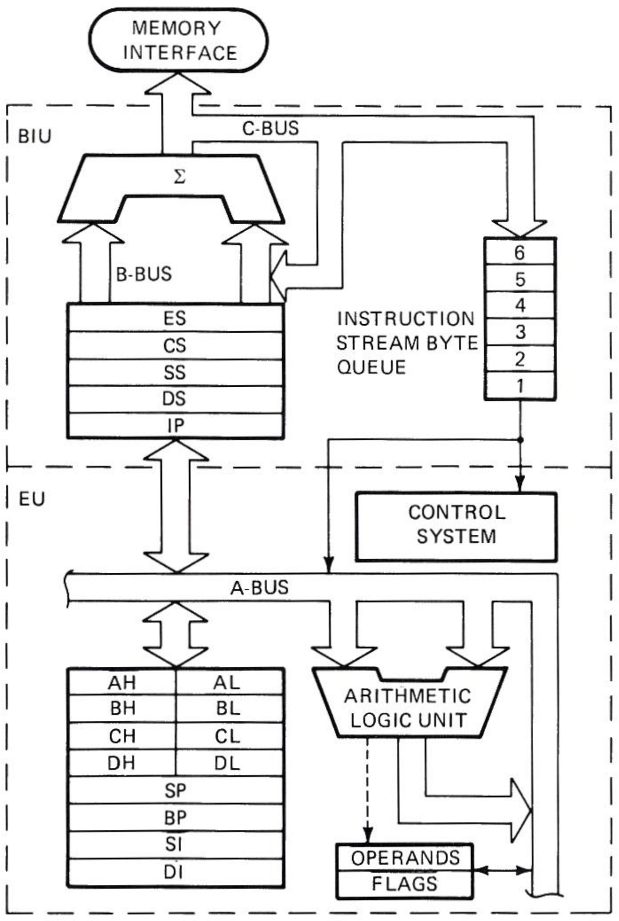
Intel 8086 microprocessor is a first member of x86 family of processors.

- 8086 was "source-code compatible" with Intel 8080 and Intel 8085 processors which meant that existing programs could be recompiled without any change to run on 8086.
- The 8086 has complete 16-bit architecture – 16-bit internal registers, 16-bit data bus, and 20-bit address bus (1 MB of physical memory). Because the processor has 16-bit index registers and memory pointers, it can effectively address only 64 KB of memory. To address memory beyond 64 KB the CPU uses segment registers – these registers specify memory locations for code, stack, data and extra data's 64 KB segments. The segments can be positioned anywhere in memory, and, if necessary, user programs can change their position.
- The 8086 executed instructions in as little as 400 ns (2.5 MIPs, or 2.5 millions of instructions per second).
- 8086 had a 6-byte instruction cache or queue that prefetched a few instructions before they were executed. The queue sped the operation of many sequences of instructions and proved to be the basis for the much larger instruction caches found in modem microprocessors.
- Intel 8086 instruction set includes a few very powerful string instructions and multiply and divide instructions. Considering all possible variations, 8086 supported over 20000 instructions (8085 supported only 246).
- The 8086 microprocessor provides support for Intel 8087 numeric co-processor. The numeric co-processor helped to speed up mathematical calculations involving decimal numbers.
- The 8086 was available in 40-pin DIP and supported clock speeds of up to 10 MHz.

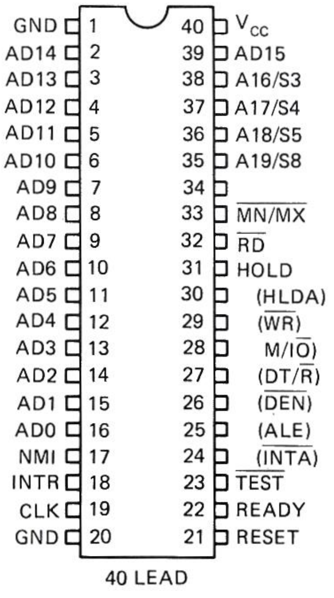
Intel 8088 microprocessor was released in 1979, (one year after the Intel 8086). The only difference was that external data bus width was reduced from 16 bits in 8086 to 8 bits in 8088. The 8088 CPU uses two consecutive bus cycles to read or write 16 bit data instead of one bus cycle for the 8086, which makes the 8088 processor to run slower. On the plus side hardware changes to the 8088 CPU made it compatible with 8080/8085 support chips.

8086 Block diagram and Pin diagram

The 8086 CPU is organized as two separate processors, called the Bus Interface Unit (BIU) and the Execution Unit (EU). The BIU provides various functions, including generation of the memory and I/O addresses for the transfer of data between outside the CPU, and the EU. The EU receives program instruction codes and data from the BIU, executes these instructions, and store the results in the general registers. By passing the data back to the BIU, data can also be stored in a memory location or written to an output device. Note that the EU has no connection to the system buses. It receives and outputs all its data through the BIU.



8086 Functional Block Diagram



8086 Pin Diagram

Bus Interface Unit

The BIU fetches instructions, reads data from memory and ports, and writes data to memory and I/O ports. The EU executes instructions that have already been fetched by the BIU. The BIU and EU function independently. The BIU interfaces the 8086 to the outside world. The BIU provides all external bus operations. The BIU contains **segment registers, instruction pointer, instruction queue, and address generation bus control circuitry** to provide functions such as fetching and queuing of instructions, and bus control.

Segment registers and instruction pointer: The BIU has four 16-bit segment registers. These are the Code Segment (CS) register, the Data Segment (DS) register, the Stack Segment (SS) register, and the Extra Segment (ES) register. The 8086's one-megabyte memory is divided into segments of up to 64K bytes each. The 8086 can directly address four segments (256K byte within the 1 Mbytes memory) at a particular time. Programs obtain access to code and data in the segments by changing the segment register contents to point to the desired segments. All program instructions must be located in main memory pointed to by the 16-bit CS register with a 16-bit offset in the segment contained in the 16-bit instruction pointer (IP).

Instruction queue: The BIU's instruction queue is a First-In First-out (FIFO) group of registers in which up to six bytes of instruction code are perfected from memory ahead of time. This is done in order to speed up program execution by overlapping instruction fetch with execution. This mechanism is known as pipelining.

Address generation circuitry: The BIU contains a dedicated adder, which is used to produce the 20-bit address. The bus control logic of the BIU generates all the bus control signals such as read and write signals for memory and I/O.

Execution Unit

The EU decodes and executes instructions. A decoder in the EU control system translates instructions. The EU has a 16-bit ALU for performing arithmetic and logic operations.

General purpose registers:

The EU has eight 16-bit general registers. These are AX, BX, CX, DX, SP, BP, SI, and DI. The 16-bit registers AX, BX, CS, and DX can be used as two 8-bit registers (AH, AL, BH, BL, CH, CL, DH, DL).

The general-purpose registers AX, BX, CX, and DX are named after special functions carried out by each one of them.

- AX is called the 16-bit accumulator while the AL is the 8-bit accumulator. The use of accumulator registers is assumed by some instructions. Multiplication and division instructions also use AX or AL. The AL register is the same as the 8085 A register.
- BX register is called the base register. This is the only general-purpose register, the contents of which can be used for addressing 8086 memory. The BX register is similar to 8085 HL register. In other words, 8086 BH and BL are equivalent to 8085 H and L registers, respectively.
- The CX register is known as the counter register. This is because some instructions such as shift rotate, and loop instructions use the contents of CX as a counter.
- The data register DX is used to hold high 16-bit result (data) in 16x16 multiplication or high 16-bit dividend (data) before a 32/16 division and the 16-bit remainder after the division.
- The two pointer registers, SP (stack pointer) and BP (base pointer), are used to access data in stack segment. The SP is used as an offset from the current SS during execution of instructions that involve stack segment in external memory. The SP contents are automatically updated (incremented or decremented) due to execution of POP or PUSH instruction.
- The base pointer contains an offset address in the current SS. This offset is used by the instructions utilizing the based addressing mode.

Flags:

The 8086 have six one-bit flags.

- AF (Auxiliary carry flag) is used by BCD bit) into the high nibble or a borrow from the high nibble into the low nibble of the low-order 8-bit of a 16-bit number.
- CF (Carry Flag) is set if there is a carry from addition or borrow from subtraction.
- OF (Overflow Flag) is set if there is an arithmetic overflow, that is, if the size of the result exceeds the capacity of the destination location. An interrupt on overflow instructions is available which will generate an interrupt in this situation.
- SF (Sign Flag) is set if the most significant bit of the result is one (Negative) and is cleared to zero for non-negative result.
- PF (Parity Flag) is set if the result has even parity; PF is zero for odd parity of the result.
- ZF (Zero Flag) is set if the result is zero; ZF is zero for non-zero result.

The 8086 also has three control bits in the flag register which can be set or reset by the programmer:

- Setting DF (Direction Flag) to one causes string instructions to auto decrement and clearing DF to zero causes string instructions to auto increment.
- Setting IF (Interrupt Flag) to one causes the 8086 to recognize external mask able interrupts; clearing IF to zero disables these interrupts.
- Setting TF (Trace Flag) to one places the 8086 in the single-step mode. In this mode, the 8086 generate an internal interrupt after execution of each instruction. The user can write a service routine at the interrupt address vector to display the desired registers and memory locations. The user can thus debug a program.

80186

Intel 80186 microprocessor, sometimes called i186, is an enhanced version of Intel 8086 16-bit processor. It was completely object code compatible with the 8086. The 80186 integrated many system components (DMA controller, clock generator, interrupt controller, timers etc.) into one chip and added 7 new instructions. With the exception of integrated components, the Intel 80186 microprocessor is not very different from the 8086, and, because of this, the 80186 may be considered as an embedded version of 8086.

80286

The 80286 was introduced in 1982 and was a 16-bit microprocessor.

- The 80286 microprocessor was almost identical to the 8086 and 8088, except that it had **24 address lines** and could address 16 MB of memory.
- The instruction set of the 80286 was almost identical to the 8086 and 8088, except for a few additional instructions that managed the extra 15M bytes of memory.
- The clock speed of the 80286 was increased, so it executed some instructions in as little as 250 ns (4.0 MIPs) with the original release 8.0 MHz version. Some changes also occurred to the internal execution of the instructions, which led to an eightfold increase in speed for many instructions when compared to 8086/8088 instructions.
- The major new feature of the 80286 microprocessor was **protected mode**. In the protected mode it was possible to protect memory and other system resources from user programs - this feature was necessary for real program multitasking.
- Models of 80286 that supported clock speeds of 12.5 MHz, 20 MHz and 25 MHz were released later.
- The 80286 has 4 functional units:
    - Bus Interface Unit
    - Instruction Unit
    - Execution Unit
    - Addressing Unit
- It supported external numeric co-processor 80287.

80386

The Intel 80386 also known as the Intel386 or i386, was a 32-bit microprocessor introduced by Intel in 1985. The first versions had 275,000 transistors. While the 80386 represented a major overhaul of the 16-bit 8086–80286 architecture, it was backwards compatible with previous generations of 80x86 processors.

- The 80386 contained a **32-bit data bus and a 32-bit memory address**. Through these 32-bit buses, the 80386 addressed up to 4 GB of memory.
- Major new feature in the i386 CPU was 80386 protected mode - this mode fixed many shortcomings that existed in the 80286 protected mode.
- Another new mode in the 80386 CPU was 8086 **virtual mode**. In this mode the CPU could run old 8086 applications while providing necessary protection of memory and other resources.
- There were a few different versions of the 80386 CPUs:
    - 80386SX – low cost version of the 80386. This processor had 16 bit external data bus and 24-bit external address bus.
    - 80386SL – low-power microprocessor with power management features, with 16-bit external data bus and 25-bit external address bus.
- The 80386 has 6 functional units:
    - Bus Interface Unit
    - Instruction Unit
    - Execution Unit
    - Addressing Unit
    - Decoder Unit
    - Pre-fetch Unit
- The initial version of 80386 supported clock speed of 16 MHz. Later 80386 versions supported clock speeds of 33 MHz.
- It supported external numeric co-processor 80387.

80486

The successor to the 80386 processor, Intel 80486 (i486) included many changes to its microarchitecture that resulted in significant performance improvements.

- **An 8 KB unified level 1 cache** (on chip cache) for code and data was added to the CPU. In later versions of the 80486 the size of level 1 cache was increased to 16 KB.
- Execution time of instructions was significantly reduced. Many load, store and arithmetic instructions executed in just one cycle (assuming that the data was already in the cache).
- Intel 486 featured much faster bus transfers – 1 CPU cycle as opposed to two or more CPU cycles for the 80386 bus.
- **Floating-point unit was integrated into 80486DX CPUs**. This eliminated delay in communications between the CPU and FPU. Furthermore, all floating-point instructions were optimized – they required fewer number of CPU cycles to execute.
- Clock-doubling and clock-tripling technology was introduced in faster versions of Intel 80486 CPU. These i486 processors could run in existing motherboards with 20 – 33 MHz bus frequency, while running internally at two or three times of bus frequency. 80486SX2 and 80486DX2 were clock-doubled version, and 80486DX4 was a clock-tripled version.
- Power management features and System Management Mode (SMM) became a standard feature of the processor.
- Intel 80486 microprocessor was produced at speeds up to 100 MHz.

Pentium

The Pentium, introduced in 1993, was similar to the 80386 and 80486 microprocessors. This microprocessor was originally labeled the P5 or 80586, but Intel decided not to use a number because it appeared to be impossible to copyright a number.

- The two introductory versions of the Pentium operated with a clocking frequency of 60 MHz and 66 MHz.
- The level 1 cache size was increased to 16 KB – 8 KB instruction cache and an 8KB data cache.
- The data bus width was increased from the 32 bits found in the 80386 and 80486 to a full **64 bits**.
- The most ingenious feature of the Pentium is its dual integer processors. The Pentium executes two instructions, which are not dependent on each other, simultaneously because it contains **two independent internal integer processors called superscaler technology**. This allows the Pentium to often execute two instructions per clocking period. To support this Pentiums has two instruction pipelines – U and V.
- Another feature that enhances performance is a **jump/branch prediction technology** that speeds the execution of program loops. The branch prediction logic helps the microprocessor make an educated guess about where the next instruction following a conditional instruction will be.
- As with the 80486, the Pentium also employs an internal floating-point coprocessor to handle floating-point data, albeit at a five times speed improvement.

x86 Family Summary

Processor	**Clock Speed(s)**	**Intro Date(s)**	**Transistors**	**Bus Width**	**Addressable Memory**	**Cache**	**Typical Use**
Intel® Pentium® Processor	66 MHz 60 MHz	Mar-93	3.1 million	64	4 GB	8 KB L1 Cache	Desktops
Intel486™ SL Processor	33 MHz 25 MHz 20 MHz	Nov-92	1.4 million	32	4 MB	8 KB	First CPU specifically designed for Notebook PCs
IntelDX4™ Processor	100 MHz 75 MHz	Mar-94	1.6 million	32	4 GB	16 KB	High-performance, entry-level desktops and value notebooks
IntelDX2™ Processor	66 MHz 50 MHz 40 MHz	Mar-92	1.2 million	32	4 GB	8 KB	High-performance, low-cost desktops
Intel486™ SX Processor	33 MHz 25 MHz	Sept-91	1.2 million	32	4 GB	8 KB	Low-cost, entry-level desktops

Processor	Clock Speed(s)	Intro Date(s)	Transistors	Bus Width	Addressable Memory	Cache	Typical Use
	20 MHz 16 MHz						
Intel386™ SL Processor	25 MHz 20 MHz	Oct-90	855,000	16	4 GB	None	First CPU designed specifically for portables
Intel486™ DX Processor	50 MHz 33 MHz 25 MHz	Apr-89	1.2 million	32	4 GB	8 KB	Desktops and servers.
Intel386™ SX Processor	33 MHz 25 MHz 20 MHz 16 MHz	Jun-88	275,000	16	16 MB	None	Entry-level desktop and portable computing
Intel386™ DX Processor	33 MHz 25 MHz 20 MHz 16 MHz	Oct-85	275,000	32	4 GB	None	Desktops
80286	12 MHz 10 MHz 6 MHz	Feb-82	134,000	16	16 MB	None	Desktops (standard CPU for all IBM PCs clones at the time)
8088	8 MHz 4.77 MHz	Jun-79	29,000	8	64 KB	None	Desktops (standard CPU for all IBM PCs and PC clones at the time)
8086	10 MHz 8 MHz 4.77 MHz	Jun-78	29,000	16	1 MB	None	Portable computing
8085	2 MHz	Mar-76	6,500	8	64 KB	None	Portable computing
8080	2 MHz	Apr-74	6,000	8	64 KB	None	Traffic light controller, Altair computer (first PC).
8008	200 KHz	Apr-72	3,500	8	16 KB	None	Dumb terminals, general calculators, bottling machines, data/character manipulation
4004	108 KHz	Nov-71	2,300	4	640 Bytes	None	Calculator, arithmetic manipulation

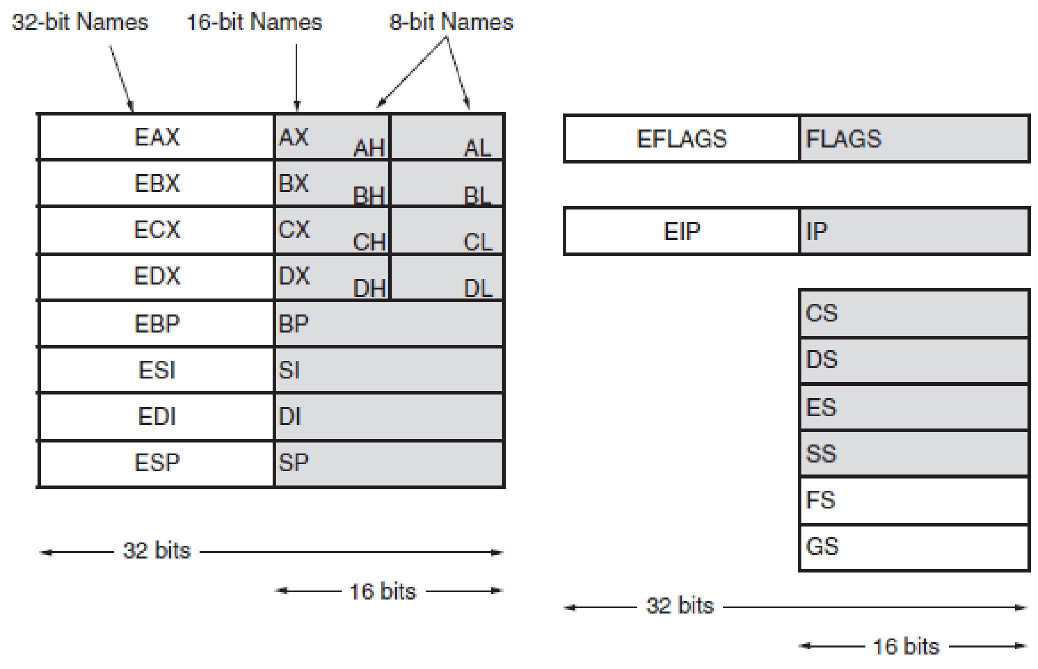
Programming model for x86 microprocessors

Figure below illustrates the programming model of the 8086 through the Pentium Pro microprocessor. The earlier 8086, 8088, and 80286 contain 16-bit internal architectures, a subset of the registers shown in figure. The 80386 through the Pentium microprocessors contain full 32-bit internal architectures. The architectures of the earlier 8086 through the 80286 are fully upward-compatible to the 80386 through Pentium. The shaded areas in this illustration represent registers that are found in early versions of the 8086, 8088, or 80286 microprocessors and are provided on the 80386–Pentium microprocessors for compatibility to the early versions.

The programming model contains 8-, 16-, and 32-bit registers. The 8-bit registers are AH, AL, BH, BL, CH, CL, DH, and DL and are referred to when an instruction is formed using these two-letter designations. The16-bit registers are AX, BX, CX, DX, SP, BP, DI, SI, IP, FLAGS, CS, DS, ES, SS, FS, and GS. Note that the first 4 16 registers contain a pair of 8-bit registers. The extended 32-bit registers are EAX, EBX, ECX, EDX, ESP, EBP, EDI, ESI, EIP, and EFLAGS. These 32-bit extended registers, and 16-bit registers FS and GS, are available only in the 80386 and above.

The registers of the x86 family can be grouped together in 3 sets:

1. Multipurpose registers.
2. Special purpose registers.
3. Segment registers.



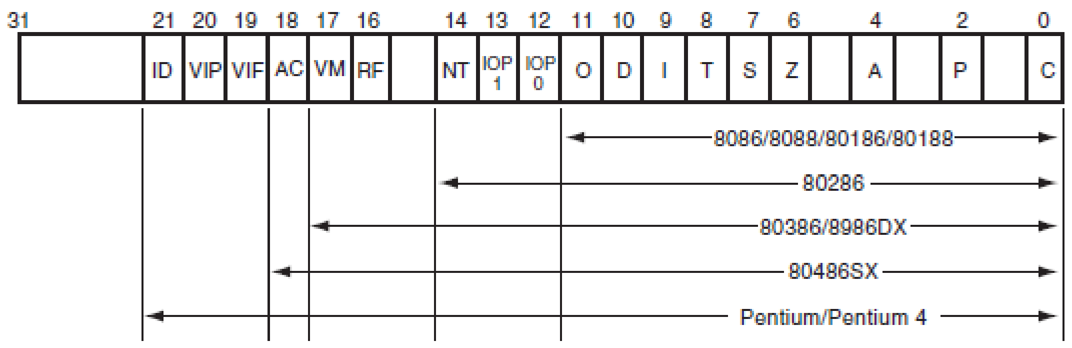
Multipurpose registers

The multipurpose registers include EAX, EBX, ECX, EDX, EBP, EDI, and ESI. These registers hold various data sizes (bytes, words, or doublewords) and are used for almost any purpose, as dictated by a program.

Special purpose registers

The special-purpose registers include EIP, ESP, and EFLAGS.

- EIP (Instruction pointer): EIP addresses the next instruction in a section of memory defined as code segment.
- ESP (Stack pointer): ESP addresses an area of memory called the stack.
- EFLAGS (Flag register): EFLAGS indicate the condition of the microprocessor and control its operation. The first 8 bits are the same as in 8085. The other bits were introduced in later generations of microprocessors to support enhanced features.



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

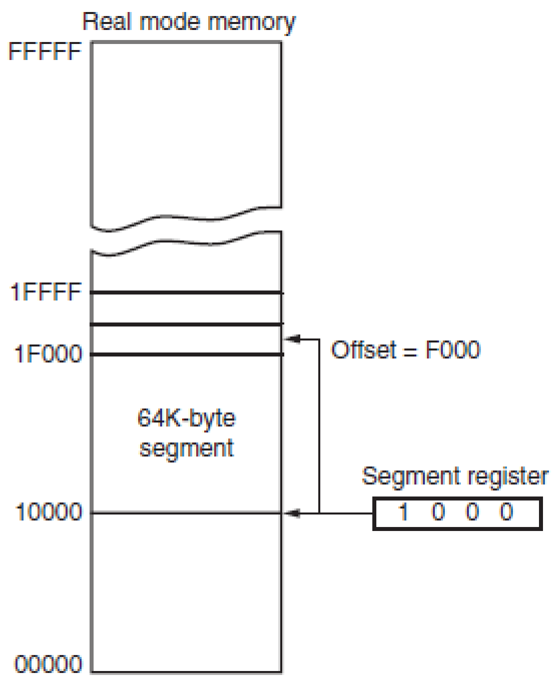
IOP: I/O privilege level
NT: Nested task
RF: Resume
VM: Virtual mode
AC: Alignment check
VIF: Virtual interrupt
VIP: Virtual interrupt pending
ID: Identification

Segment registers

Segment registers include CS, DS, ES, SS, FS, and GS. All segment registers are 16 bits. These registers generate memory addresses when combined with other registers in the microprocessor. There are either four or six segment registers in various versions of the microprocessor. A segment register functions differently in the real mode when compared to the protected mode operation of the microprocessor.

Segments and Offsets

A combination of a segment address and an offset address accesses a memory location in the real mode. The segment address, located within one of the segment registers, defines the beginning address of any 64K-byte memory segment. The offset address selects any location within the 64K byte memory segment. Segments in the real mode always have a length of 64K bytes. Figure shows how the segment plus offset addressing scheme selects a memory location.



This illustration shows a memory segment that begins at location 10000H and ends at location 1FFFFH—64K bytes in length. It also shows how an offset address of F000H selects location 1F000H in the memory system. Note that the offset is the distance above the start of the segment. The segment register in contains 1000H, yet it addresses a starting segment at location 10000H. In the real mode, each segment register is internally appended with a 0H on its rightmost end. This forms a 20-bit memory address, allowing it to access the start of a segment. The microprocessor must generate a 20-bit memory address to access a location within the first 1M of memory.

Default 32-bit segments and offset registers:

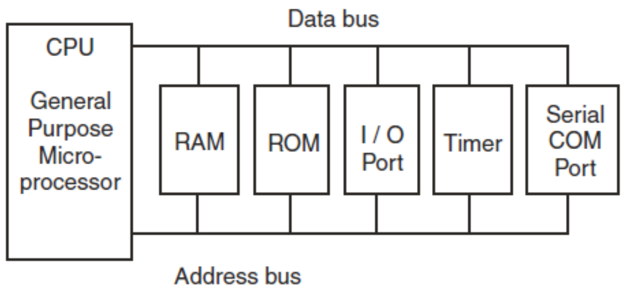
Segment	*Offset*	*Special Purpose*
CS	EIP	Instruction address
SS	ESP or EBP	Stack address
DS	EAX, EBX, ECX, EDX, ESI, EDI, an 8- or 32-bit number	Data address
ES	EDI for string instructions	String destination address
FS	No default	General address
GS	No default	General address

8051 Microcontroller

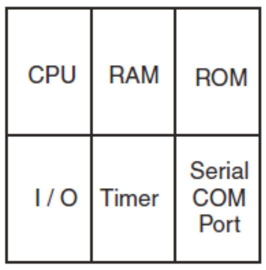
A microcontroller is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals. Program memory in the form of ROM is also often included on chip, as well as a small amount of RAM. Microcontrollers are designed for embedded applications, in contrast to the microprocessors used in personal computers or other general purpose applications.

Microcontrollers are used in automatically controlled products and devices, such as automobile engine control systems, implantable medical devices, remote controls, office machines, appliances, power tools, toys and other embedded systems. By reducing the size and cost compared to a design that uses a separate microprocessor, memory, and input/output devices, microcontrollers make it economical to digitally control even more devices and processes.

Microcontrollers vs Microprocessors



(a) General-purpose Microprocessor System



(b) Microcontroller

A microprocessor requires an external memory for program/data storage. Instruction execution requires movement of data from the external memory to the microprocessor or vice versa. Usually, microprocessors have good computing power and they have higher clock speed to facilitate faster computation.

A microcontroller has required on-chip memory with associated peripherals. A microcontroller can be thought of a microprocessor with inbuilt peripherals. A microcontroller does not require much additional interfacing ICs for operation and it functions as a standalone system. The operation of a microcontroller is multipurpose, just like a Swiss knife. Microcontrollers are also called embedded controllers. A microcontroller clock speed is limited only to a few tens of MHz. Microcontrollers are numerous and many of them are application specific.

8051 Family of Microcontrollers

The Intel 8051 microcontroller is one of the most popular general purpose microcontrollers in use today. The success of the Intel 8051 spawned a number of clones which are collectively referred to as the MCS-51 family of microcontrollers.

Some of the microcontrollers of 8051 family are given as follows:

Device	On-chip data memory (bytes)	On-chip program memory (bytes)	16-bit timer/counter	No. of vectored interrupts
8031	128	None	2	5
8032	256	None	2	6
8051	128	4K ROM	2	5
8052	256	8K ROM	3	6
8751	128	4K EPROM	2	5
8752	256	8K EPROM	3	6
AT89C51	128	4K Flash memory	2	5
AT89C52	256	8K Flash memory	3	6

The Intel 8051 is an 8-bit microcontroller which means that most available operations are limited to 8 bits. The MCS-51 family of microcontroller has been optimized for 8-bit math and single bit Boolean operations.

The 8051's predecessor and Intel's first microprocessor, the 8048, was used in the keyboard of the first IBM PC, where it converted key presses into the serial data stream which is sent to the main unit of the computer. The 8048 and derivatives are still used today for basic model keyboards.

The 8031 was a cut down version of the original Intel 8051 that did not contain any internal program memory (ROM). To use this chip, external ROM had to be added containing the program that the 8031 would fetch and execute.

The 8052 was an enhanced version of the original 8051 that featured 256 bytes of internal RAM instead of 128 bytes, 8 KB of ROM instead of 4 KB, and a third 16-bit timer. The 8032 had these same features except for the internal ROM program memory. 8052 AH, a variant of 8052 that supports programming in BASIC language is also available.

The 8751 and 8752 are similar to 8051 and 8052 respectively but these microcontrollers have 4 KB of EPROM which makes them user reprogrammable.

8048 family has two other members – 8049 which supports 2KB of ROM and 8050 which supports 4KB of ROM.

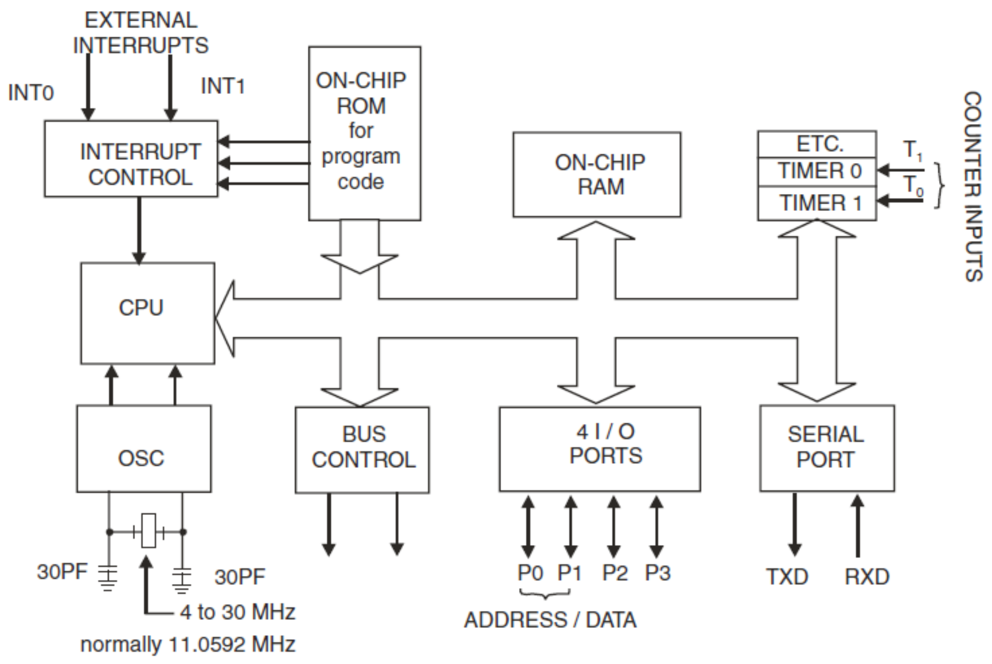
Intel discontinued its MCS-51 product line in March 2007, however there are plenty of enhanced 8051 products available from other vendors.

Features of 8051

Some of the features that have made the 8051 popular are:

- 4 KB on chip program memory (ROM).
- 128 bytes on chip data memory (RAM).
- Four 8-bit register banks.
- Bit as well as byte addressable RAM area of 16 bytes.
- Special Function Registers (SFRs) of 128 bytes.
- 8-bit data bus and 16-bit address bus.
- Two 16 bit timers/counters (T0 and T1).
- 3 internal and 2 external vectored interrupts.
- 32 I/O pins arranged as four 8-bit ports (P0 - P3)
- 16-bit program counter and data pointer.
- One full duplex serial I/O.
- 1 Microsecond instruction cycle with 12 MHz Crystal.

8051 Architecture and Programming Model



Input/Output Ports

All 8051 microcontrollers have 4 I/O ports each comprising 8 bits which can be configured as inputs or outputs. Accordingly, in total of 32 input/output pins enabling the microcontroller to be connected to peripheral devices are available for use.

Whether it is to be configured as an input (1) or an output (0), depends on its logic state. In order to configure a microcontroller pin as an output, it is necessary to apply a logic zero (0) to appropriate I/O port bit. In this case, voltage level on appropriate pin will be 0.

Similarly, in order to configure a microcontroller pin as an input, it is necessary to apply a logic one (1) to appropriate port. In this case, voltage level on appropriate pin will be 5V (as is the case with any TTL input).

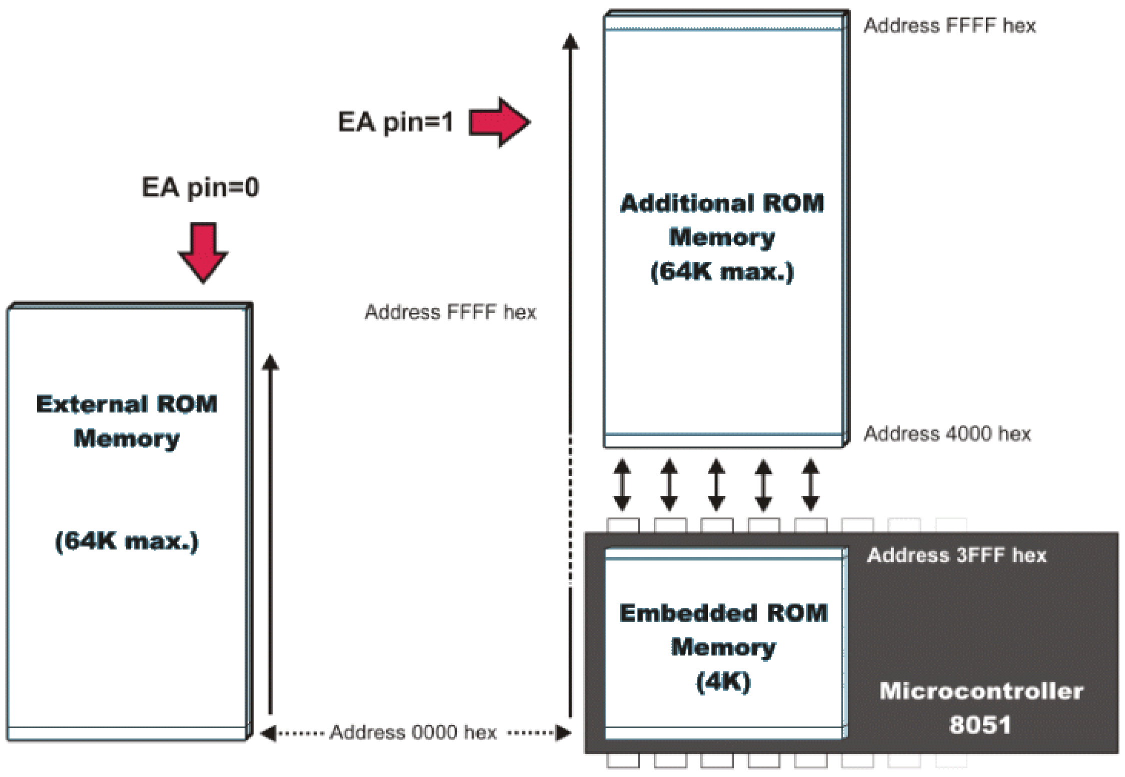
- Port 0: If external memory is used then the lower address byte (addresses A0-A7) is available on P0 port. Otherwise, all bits of this port are configured as inputs/outputs.
- Port 1: P1 is a true I/O port, because it doesn't have any alternative functions.
- Port 2: P2 acts similarly to P0 when external memory is used. The higher address byte (address A8-A15) is available on P2. When no memory is added, this port can be used as a general input/output port.
- Port 3: All port pins can be used as general I/O, but they also have an alternative function (serial I/O, interrupt, counter). In order to use these alternative functions, a logic one (1) must be applied to appropriate bit of the P3 register.

Memory Organization

Program Memory

The first models of the 8051 microcontroller family did not have internal program memory. It was added as an external separate chip. These models are recognizable by their label beginning with 803 (for example 8031 or 8032). All later models have a few Kbyte ROM embedded. Even though such an amount of memory is sufficient for writing most of the programs, there are situations when it is necessary to use additional memory as well.

How the microcontroller handle external memory depends on the EA pin logic state:

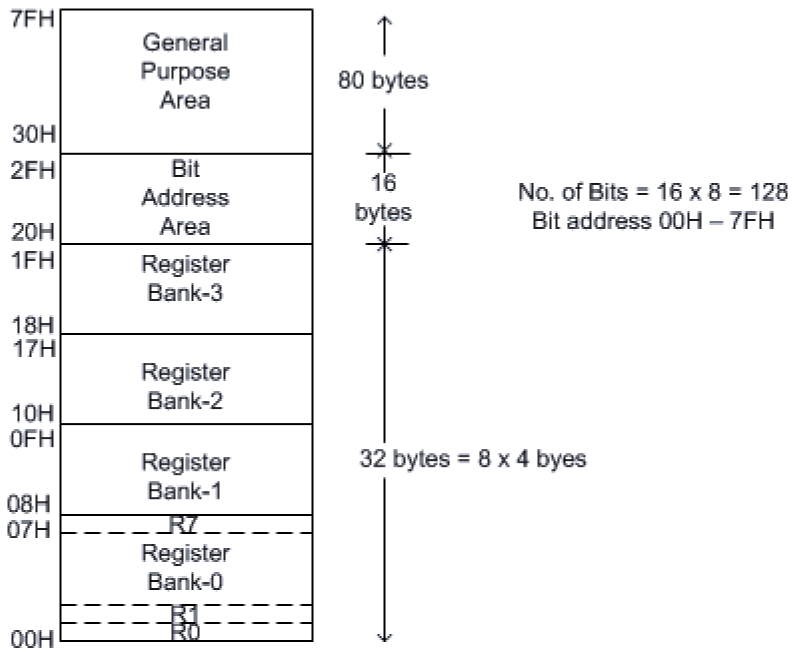


- **EA=0**: In this case, the microcontroller completely ignores internal program memory and executes only the program stored in external memory.

- **EA=1**: In this case, the microcontroller executes first the program from built-in ROM, then the program stored in external memory.

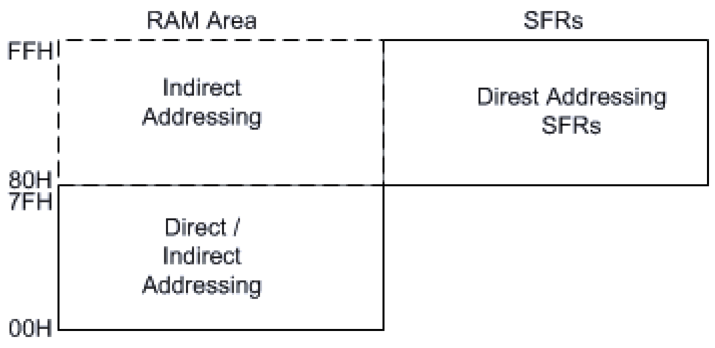
Data Memory

Data Memory is used for temporarily storing data and intermediate results created and used during the operation of the microcontroller. Besides, RAM memory built in the 8051 family includes many registers such as hardware counters and timers, input/output ports, serial data buffers etc. The previous models had 256 RAM locations, while for the later models this number was incremented by additional 128 registers. However, the first 256 memory locations (addresses 00H – FFH) are the heart of memory common to all the models belonging to the 8051 family. Locations available to the user occupy memory space with addresses 00H to 7FH, i.e. first 128 registers. This part of RAM is divided in several blocks.



- The first block consists of 4 banks each including 8 registers denoted by R0-R7. Prior to accessing any of these registers, it is necessary to select the bank containing it.
- The next memory block (address 20H – 2FH) is bit- addressable, which means that each bit has its own address (00H – 7FH). Since there are 16 such registers, this block contains in total of 128 bits with separate addresses (address of bit 0 of 20H byte is 0, while address of bit 7 of 2FH byte is 7F).
- The third group of registers occupy addresses 2FH – 7FH, i.e. 80 locations, and does not have any special functions or features.

Internal Data Memory and SFR



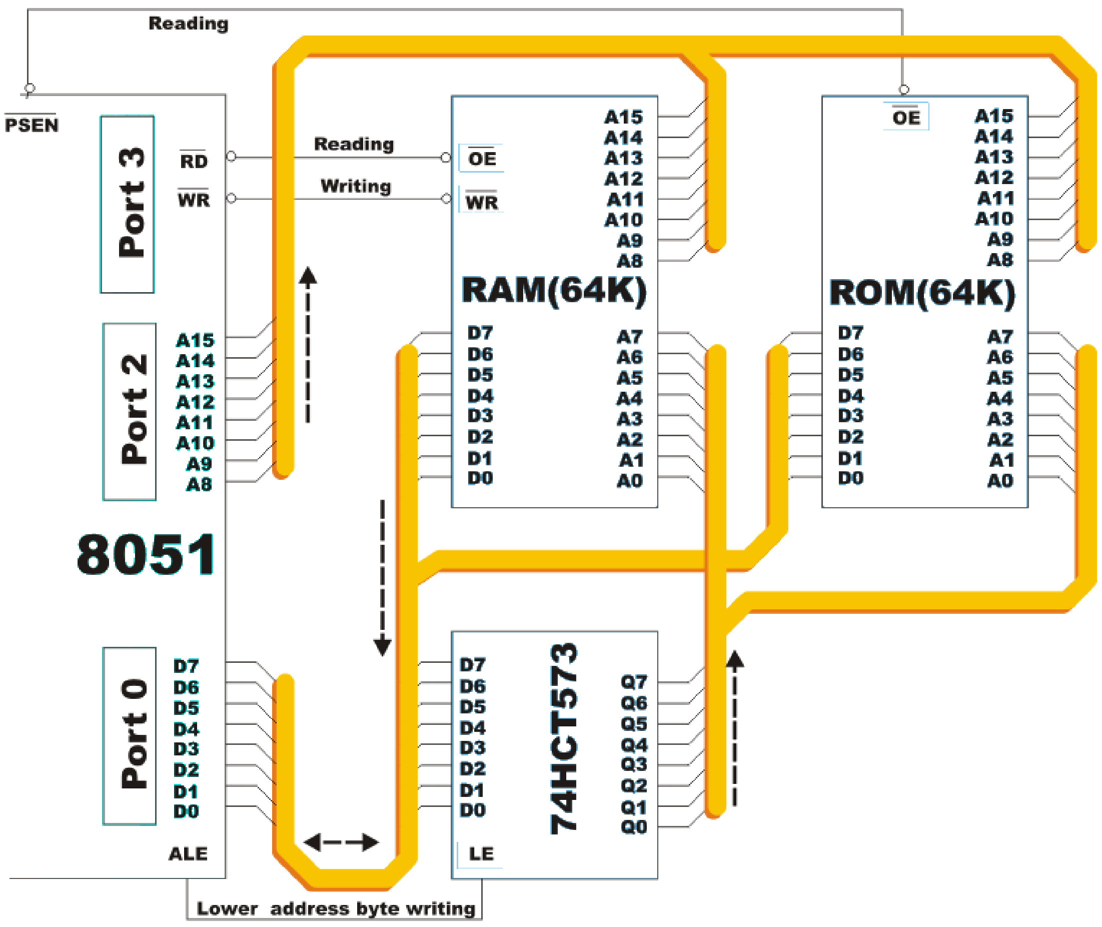
The special function registers (SFRs) are mapped in the upper 128 bytes of internal data memory address. Hence there is an address overlap between the upper 128 bytes of data RAM and SFRs. Please note that the upper 128 bytes of data RAM are present only in the 8052 family. The lower128 bytes of RAM (00H – 7FH) can be accessed both by direct or indirect addressing while the upper 128 bytes of RAM (80H – FFH) are accessed by indirect addressing. The

SFRs (80H – FFH) are accessed by direct addressing only. This feature distinguishes the upper 128 bytes of memory from the SFRs.

Memory Expansion

In case memory (RAM or ROM) built in the microcontroller is not sufficient, it is possible to add two external memory chips with capacity of 64Kb each. P2 and P3 I/O ports are used for their addressing and data transmission.

From the user's point of view, everything works quite simply when properly connected because most operations are performed by the microcontroller itself. The 8051 microcontroller has two pins for data read $\overline{\mathrm{RD}}$ (P3.7) and $\overline{\mathrm{PSEN}}$. The first one is used for reading data from external data memory (RAM), while the other is used for reading data from external program memory (ROM). Both pins are active low.



Special Function Registers (SFR)

Special Function Registers (SFRs) are a sort of control table used for running and monitoring the operation of the microcontroller. Each of these registers as well as each bit they include, has its name, address in the scope of RAM and precisely defined purpose such as timer control, interrupt control, serial communication control etc. Even though there are 128 memory locations intended to be occupied by them, the basic core, shared by all types of 8051 microcontrollers, has only 21 such registers. Rest of locations are intentionally left unoccupied in order to enable the manufacturers to further develop microcontrollers keeping them compatible with the previous versions. It also enables programs written a long time ago for microcontrollers which are out of production now to be used today.

A Register (Accumulator)

A register is a general-purpose register used for storing intermediate results obtained during operation. Prior to executing an instruction upon any number or operand it is necessary to store it in the accumulator first. All results obtained from arithmetical operations performed by the ALU are stored in the accumulator. Data to be moved from one register to another must go through the accumulator. In other words, the A register is the most commonly used register.

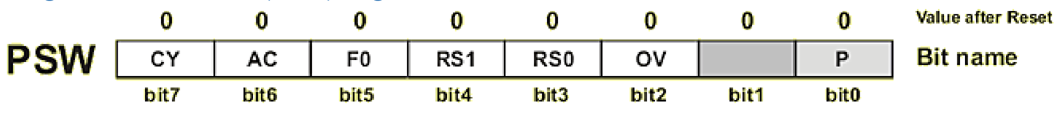
B register

Multiplication and division can be performed only upon numbers stored in the A and B registers. All other instructions in the program can use this register as a spare accumulator (A).

R Registers (R0-R7)

This is a common name for 8 general-purpose registers (R0, R1, R2 ...R7). Even though they are not true SFRs, they deserve to be discussed here because of their purpose. They occupy 4 banks within RAM. Similar to the accumulator, they are used for temporary storing variables and intermediate results during operation. Which one of these banks is to be active depends on two bits of the PSW Register. Active bank is a bank the registers of which are currently used.

Program Status Word (PSW) Register



	0	0	0	0	0	0	0	0	Value after Reset
PSW	CY	AC	F0	RS1	RS0	OV	▨	P	Bit name
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	

PSW register is one of the most important SFRs. It contains several status bits that reflect the current state of the CPU. PSW contains carry bit (CY), auxiliary carry (AC), user-definable status flag (F0), two register bank select bits (RS1 and RS0), overflow flag (OV), and parity bit.

Data Pointer Register (DPTR)

DPTR register is not a true one because it doesn't physically exist. It consists of two separate registers: DPH (Data Pointer High) and (Data Pointer Low). For this reason it may be treated as a 16-bit register or as two independent 8-bit registers. Their 16 bits are primarily used for external memory addressing. Besides, the DPTR Register is usually used for storing data and intermediate results.

Stack Pointer (SP) Register

A value stored in the Stack Pointer points to the first free stack address and permits stack availability. Stack pushes increment the value in the Stack Pointer by 1. Likewise, stack pops decrement its value by 1. Upon any reset and power-on, the value 7 is stored in the Stack Pointer, which means that the space of RAM reserved for the stack starts at this location. If another value is written to this register, the entire Stack is moved to the new memory location.

P0, P1, P2, P3 – Input/Output Registers

If neither external memory nor serial communication system are used then 4 ports with a total of 32 input/output pins are available for connection to peripheral environment. Each bit within these ports affects the state and performance of appropriate pin of the microcontroller.

Networking Technology

What is a Computer Network?

A network is any collection of independent computers that communicate with one another over a shared network medium. A computer network is a collection of two or more connected computers. When these computers are joined in a network, people can share files and peripherals such as modems, printers, tape backup drives, or CD-ROM drives. When networks at multiple locations are connected using services available from phone companies, people can send e-mail, share links to the global Internet, or conduct video conferences in real time with other remote users. As companies rely on applications like electronic mail and database management for core business operations, computer networking becomes increasingly more important.

Every network includes:

- At least two computers Server and Client workstation.
- Networking Interface Card's (NIC)
- A connection medium, usually a wire or cable, although wireless communication between networked computers and peripherals is also possible.
- Network Operating system software, such as Microsoft Windows Server, Novell NetWare, UNIX and Linux.

Types of Network

Interprocessor distance	Processors located in same	Example
1 m	Square meter	Personal area network
10 m	Room	
100 m	Building	Local area network
1 km	Campus	
10 km	City	Metropolitan area network
100 km	Country	
1000 km	Continent	Wide area network
10,000 km	Planet	The Internet

PAN

PANs (Personal Area Networks) let devices communicate over the range of a person. A common example is short range wireless network called Bluetooth.

LAN

LANs are networks usually confined to a geographic area, such as a single building or a college campus. LANs can be small, linking as few as three computers, but often link hundreds of computers. The development of standard networking protocols and media has resulted in worldwide proliferation of LANs throughout business and educational organizations.

MAN

The term metropolitan area network (MAN) is not used often anymore; it refers to a network that exists within a single city or metropolitan area. If we had two different buildings within a city that were connected together, it would be considered a MAN. The best-known examples of MANs are the cable television networks available in many cities.

WAN

Wide area networking combines multiple LANs that are geographically separate. This is accomplished by connecting the different LANs using services such as dedicated leased lines, dial-up phone lines and satellite links.

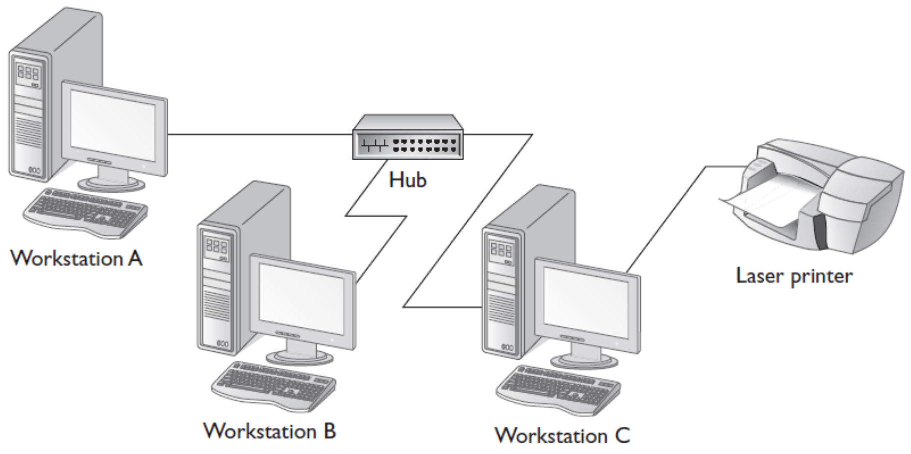
Categories of Network

Organizations of different sizes, structures, and budgets need different types of networks. A local newspaper company has needs for its network that would be different from the needs of a multinational company. Networks can be divided into one of two categories: peer-to-peer or server-based networks.

Peer-to-peer Network

A peer-to-peer network has no dedicated servers; instead, a number of workstations are connected together for the purpose of sharing information or devices. When there is no dedicated server, all workstations are considered equal; any one of them can participate as the client or the server. Peer-to-peer networks are designed to satisfy the networking needs of home networks or of small companies that do not want to spend a lot of money on a dedicated server but still want to have the capability to share information or devices. For example, a small accounting firm with three employees that needs to access customer data from any of the three systems or print to one printer from any of the three systems may not want to spend a lot of money on a dedicated server. A small peer-to-peer network will allow these three computers to share the printer and the customer information with one another.

Most of the modern operating systems such as Windows already have built-in peer-to-peer networking capabilities, which is why building a peer-to-peer network would be a “cheap” network solution. The disadvantage of a peer-to-peer network is the lack of centralized administration – with peer-to-peer networks, you need to build user accounts and configure security on each system.

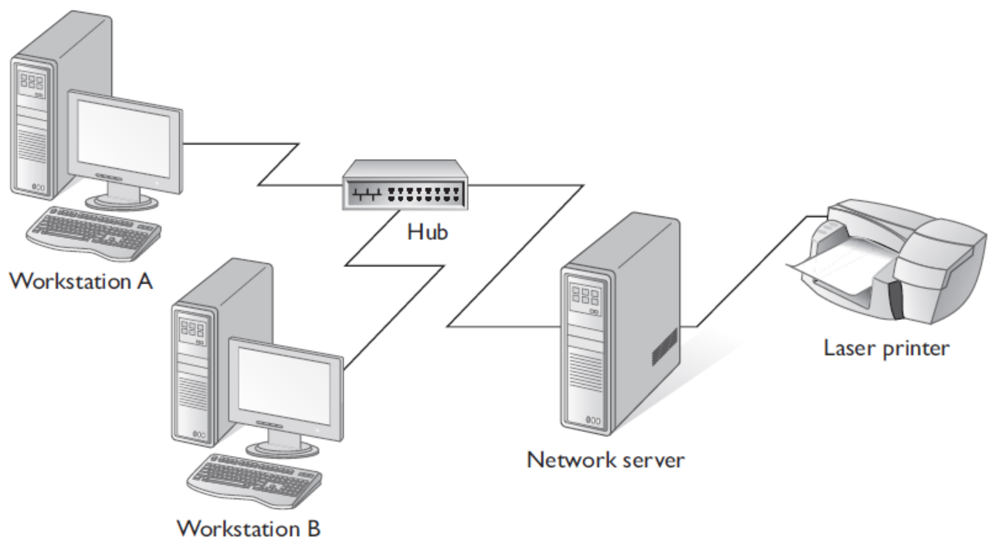


Peer-to-peer networks are typically designed for fewer than 10 systems.

Server based Network

A big disadvantage of peer-to-peer networking is that it cannot be administered from a single place. With peer-to-peer networking, user accounts typically are created on all the systems, and data files are stored throughout all the systems. This leads to a more complicated environment and makes the job of a network administrator harder. Usually after four or five systems have been networked, the need for a dedicated server to store all of the user accounts and data files becomes apparent – this is a server-based network.

The advantage of a server-based network is that the data files that will be used by all of the users are stored on the one server. This provides a central point to set up permissions on the data files, and it will gives a central point from which to back up all of the data in case data loss should occur. With a server-based network, the network server stores a list of users who may use network resources and usually holds the resources as well.



The server in a server-based network may provide a number of different services. The services it will offer to the network usually are decided by the server's role.

There are a number of different roles that a server could play on a network:

- File and print servers
- Application servers
- Web servers
- Directory servers

File and print servers control and share printers and files among clients on the network. File and print servers were the original reason to have a network; a large number of users needed access to the same files, so the files were placed on a server, and all clients were connected to the server when they needed to work with the files. File servers often have the following characteristics:

- Large amounts of memory
- Fast hard disks
- High-capacity tape drives
- Multiple CPUs
- Fast I/O buses
- Fast network adapters
- Redundant power supplies
- Hot-swappable hard disks

File and print servers also check the access control list (ACL) of each resource before allowing a user to access a file or use a printer. If the user or a group to which the user belongs is not listed in the ACL the user is not allowed to use the resource, and an "access denied" message appears on the user's screen.

Application server are servers that run some form of special program on the server. A good example of an application server is a server that runs the company's e-mail server. The e-mail server software is special software that can be run on a server operating system. Another example of software that would run on an application server is a database server product such as Microsoft SQL Server. A database server is a server that holds the company's core business data and typically gives this data to custom applications that run on the workstations.

Web server are servers that run the Hypertext Transfer Protocol (HTTP) and are designed to publish information on the Internet or the corporate intranet. Web servers are popular in today's businesses because they host web applications (web sites) for the organization. These web applications could be designed for internal use, or they could be used to publish information to the rest of the world on the Internet. Examples of web server software are

Microsoft's Internet Information Services that runs on Windows or Apache web server software that runs on UNIX/Linux and Windows.

Directory server hold a list of the user accounts that are allowed to log on to the network. This list of user accounts is stored in a database (known as the directory database) and can store information about these user accounts such as address, city, phone number, and fax number.

In a server-based network environment, the centralized administration comes from the fact that the directory server stores all user accounts in its directory database. When a user sits at a client machine to log on to the network, the logon request is sent to this directory server. If the username arid password exist in the directory database, the client is allowed to access network resources.

Network Protocols (Software Layer)

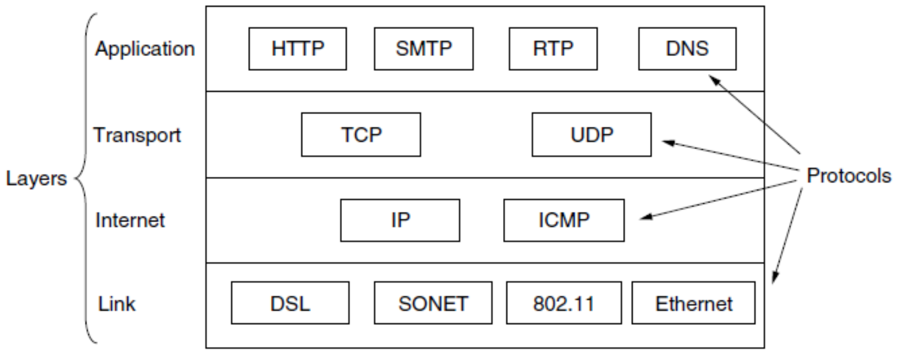
To reduce their design complexity, most networks are organized as a stack of layers or levels, each one built upon the one below it. The purpose of each layer is to offer certain services to the higher layers while shielding those layers from the details of how the offered services are actually implemented. The number of layers, the name of each layer, the contents of each layer, and the function of each layer differ from network to network.

This concept is actually a familiar one and is used throughout computer science, where it is variously known as information hiding, abstract data types, data encapsulation, and object-oriented programming.

When layer *n* on one machine carries on a conversation with layer *n* on another machine, the rules and conventions used in this conversation are collectively known as the layer n **protocol**. Basically, a protocol is an agreement between the communicating parties on how communication is to proceed.

TCP/IP

TCP/IP reference model is used by the worldwide Internet. It is based on the architecture designed for ARPANET (a research network sponsored by the US Department of Defense). TCP/IP consists of 4 layers. From bottom to top these are – Link Layer, Internet Layer, Transport Layer and Application Layer.



Link Layer

The data link layer provides node-to-node data transfer – a link between two directly connected nodes. It detects and possibly corrects errors that may occur in the physical layer. It defines the protocol to establish and terminate a connection between two physically connected devices.

IEEE 802 divides the data link layer into two sublayers:

- Medium access control (MAC) layer – responsible for controlling how devices in a network gain access to a medium and permission to transmit data.
- Logical link control (LLC) layer – responsible for identifying and encapsulating network layer protocols, and controls error checking and frame synchronization.

Internet Layer

The internet (network) layer provides the functional and procedural means of transferring variable length data sequences (called datagrams) from one node to another connected in "different networks". A network is a medium to which many nodes can be connected, on which every node has an address and which permits nodes connected to it to transfer messages to other nodes connected to it by merely providing the content of a message and the address of the destination node and letting the network find the way to deliver the message to the destination node, possibly routing it through intermediate nodes.

If the message is too large to be transmitted from one node to another on the data link layer between those nodes, the network layer implements message delivery by splitting the message into several fragments at one node, sending the fragments independently, and reassembling the fragments at another node. It may, but does not need to, report delivery errors. Message delivery at the network layer is not necessarily guaranteed to be reliable; a network layer protocol may provide reliable message delivery, but it need not do so.

The internet layer defines an official packet format and protocol called IP (Internet Protocol), plus a companion protocol called ICMP (Internet Control Message Protocol) that helps it function. The job of the internet layer is to deliver IP packets where they are supposed to go. A unique IP address is assigned to each device that connects to the network. A typical IP address looks like 192.168.200.101. It consists of a string of four, 3 digit numbers separated by a period. The 3 digit numbers can range from 0 to 255. IP does not guarantee message delivery.

Transport Layer

The transport layer provides the functional and procedural means of transferring variable-length data sequences from a source to a destination host, while maintaining the quality of service.

Two end-to-end transport protocols have been defined here:

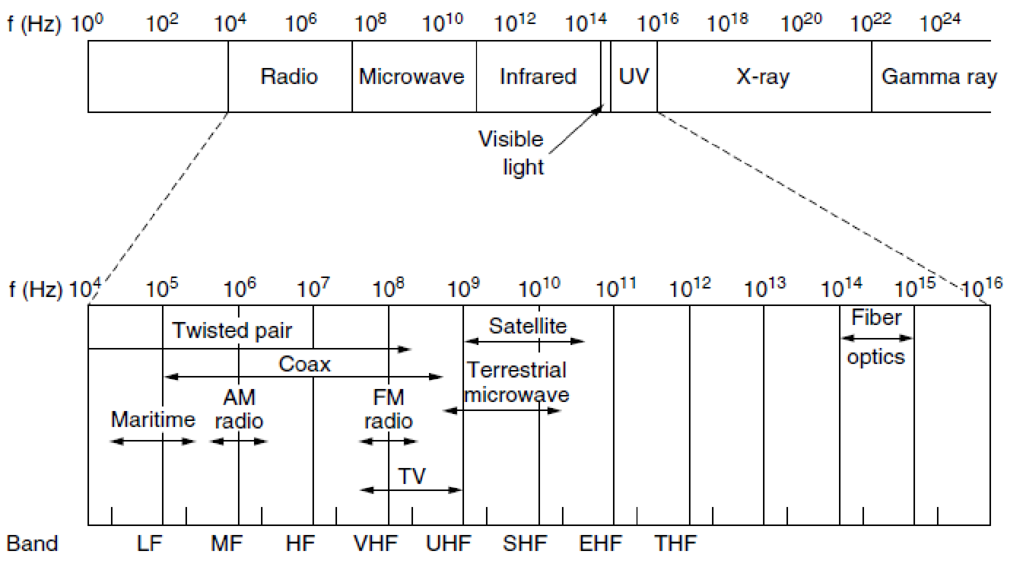
- TCP (Transmission Control Protocol): TCP guarantees that the recipient will receive the packets in order by numbering them. The recipient sends messages back to the sender saying it received the messages. If the sender does not get a correct response, it will resend the packets to ensure the recipient received them. Packets are also checked for errors. TCP ensures reliability. Packets sent with TCP are tracked so no data is lost or corrupted in transit. This is why file downloads do not become corrupted even if there are network hiccups.
- UDP (User Datagram Protocol): A datagram is the same thing as a packet of information. The UDP protocol works similarly to TCP, but it does not implement error-checking. When using UDP, packets are just sent to the recipient. There is no guarantee that the receiver will get all the packets and there is no way to ask for a missed packet. This makes the data transfer faster but at the cost of reliability. UDP is used when speed is desirable and error correction is not necessary. For example, UDP is frequently used for live broadcasts and online games.

Application Layer

The application layer is closest to the user. The user interacts with the application layer through a software application. The application layer contains higher-level protocols like FTP (File Transfer Protocol), HTTP (Hypertext Transfer Protocol), STMP (Simple Mail Transfer Protocol), DHCP (Dynamic Host Control Protocol) and DNS (Domain Name Service).

Transmission Media

Transmission media make it possible to transmit the electronic signals from one computer to another. Computer signals express data value in form of binary impulses (0/1). These signals are transmitted between devices on the network using some form of transmission media such as cables or radio waves till they reach the desired computer. The figure below illustrates the range of electromagnetic spectrum and their associated frequencies. Different parts of the spectrum are used by different forms of communication.


f (Hz) 10^0 10^2 10^4 10^6 10^8 10^10 10^12 10^14 10^16 10^18 10^20 10^22 10^24
Radio
Microwave
Infrared
UV
X-ray
Gamma ray
Visible light
f (Hz) 10^4 10^5 10^6 10^7 10^8 10^9 10^10 10^11 10^12 10^13 10^14 10^15 10^16
Twisted pair
Coax
Satellite
Terrestrial microwave
Fiber optics
Maritime
AM radio
FM radio
TV
Band LF MF HF VHF UHF SHF EHF THF

Transmission Media Characteristics

A transmission media has certain characteristics that makes it suitable for a specific type of service. These include:

- Cost: One of the main factors in the purchase decision of any networking component is the cost. The network designer needs to balance between the cost and features of a transmission media.
- Installation Requirements: It involves the ease/complexity to install and maintain the network as well as the actual physical layout of the network.
- Bandwidth: Bandwidth is the measure of the capacity of a medium to transmit data. Data transmission rates are measured in bps (bits per second). Bandwidth depends on the type of cable and the length of the cable.
- Band usage: There are two techniques that may be used to transmit signal along a channel:
  - Baseband: Information is sent as digital signals through the media as a single channel that uses the entire bandwidth of the media. The signal is delivered as a pulse of electricity or light, depending on the type of cabling being used. Baseband communication is also bidirectional, which means that the same channel can be used to send and receive signals.
  - Broadband: Information is sent in the form of an analog signal, which flows as electromagnetic waves or optical waves. Each transmission is assigned to a portion of the bandwidth, so unlike with baseband communication, it is possible to have multiple transmissions at the same time, with each transmission being assigned its own channel or frequency. Broadband communication is unidirectional, so in order to send and receive, two pathways will need to be used.
- Attenuation: It is a measure of how much the signal weakens as it travels through a medium. It decides the maximum length of the cable. When the signal drops below a certain limit it becomes difficult to separate the signal from noise. Repeaters can be used to regenerate the signal and deal with attenuation.
- EMI – Electromagnetic Interference: EMI refers to electromagnetic noise that distorts the signal in a medium. Cross talk is a special type of EMI caused by the adjacent wire. It occurs when the signal from one wire is picked up by another wire.

Wired Media

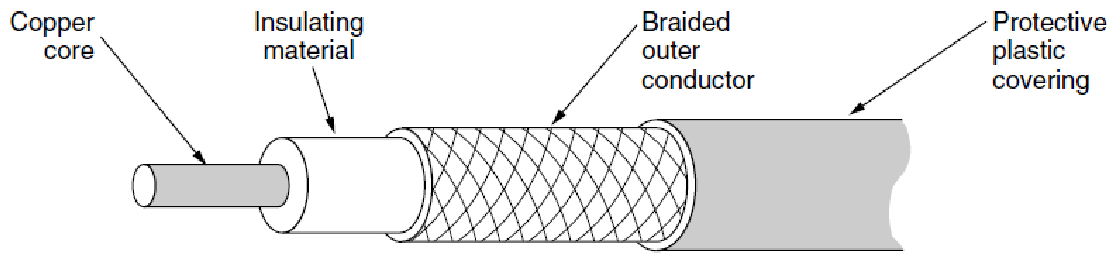
Cabling is the medium for transmission of data between hosts and LANs. LANs can be connected together using a variety of cable types. Each cable type has its own advantages and disadvantages.

Coaxial Cable

Coaxial, or coax, cable looks like the cable used to bring the cable TV signal to your television. One strand (a solid-core copper wire) runs down the middle of the cable. Around that strand is a layer of insulation, and covering that insulation is braided wire and metal foil, which shields against electromagnetic interference. A final layer of insulation covers the braided wire. Because of the layers of insulation, coaxial cable is more resistant to outside interference than other cabling, such as unshielded twisted-pair (UTP) cable. There are two types of coax cabling: thinnet (thin coaxial cable about $\sim {}^{1}/_{4}$ inch thick) and thicknet (thick coaxial cable about $\sim {}^{1}/_{2}$ inch thick). The two differ in thickness and maximum cable distance that the signal can travel.

Thinnet is used for short distance communication (less than 185 m). It connects directly to a workstations network adapter card using a BNC (British Naval Connector).

Maximum cable length of thicknet is 500 m. Due to its thickness it is harder to work with thicknet cable. Connection from the transceiver to the network adapter is made using AUI (adapter unit interface) port connector.



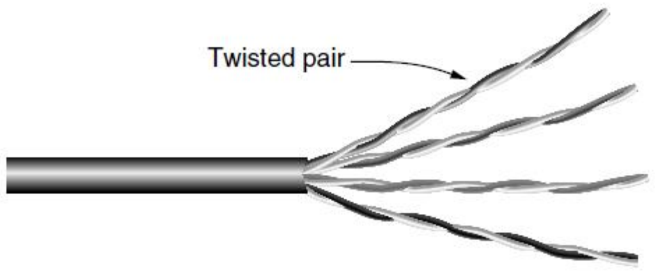
Twisted Pair Cable (UTP and STP)

Twisted pair cable is more widely used today. It has four pairs of wires that are twisted to help reduce crosstalk or interference from outside electrical devices. There are two forms of twisted pair cables – unshielded twisted pair (UTP) and shielded twisted pair (STP). Data transfer rates of up to 1 Gbps are possible with CAT 5e and up to 10 Gbps are possible with CAT 6 twisted pair cable.

UTP

The typical twisted-pair cable for network use contains four pairs of wires. Each member of the pair of wires contained in the cable is twisted around the other. The twists in the wires help shield against electromagnetic interference. The maximum distance of UTP is 100 meters. UTP cable uses small plastic connectors designated as registered jack 45, or most often referred to as RJ-45. RJ-45 is similar to the phone connectors, except that instead of four wires, as found in the home system, the network RJ-45 connector contains eight contacts, one for each wire in a UTP cable.

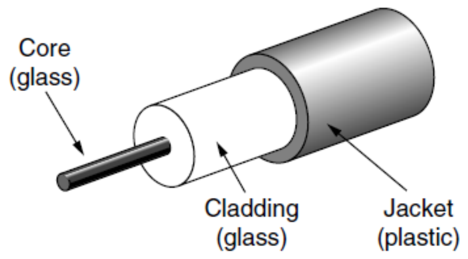
UTP cable is easier to install than coaxial because you can pull it around corners more easily due to its flexibility and small size. Twisted-pair cable is more susceptible to interference than coaxial, however, and should not be used in environments containing large electrical or electronic devices.



STP

Shielded twisted-pair (STP) cable is very similar to UTP cabling, but it differs from UTP in that it uses a layer of insulation within the protective jacket, which helps maintain the quality of the signal.

Fiber Optic Cable



The third type of cabling that we want to discuss is fiber-optic cabling. Fiber-optic cabling is unlike coax and twisted-pair, because both of those types have a copper wire that carries the electrical signal. Fiber-optic cables use optical fibers that carry digital data signals in the form of modulated pulses of light. An optical fiber consists of an extremely thin cylinder of glass, called the core, surrounded by a concentric layer of glass, known as the cladding. There are two fibers per cable—one to transmit and one to receive. The core also can be an optical-quality clear plastic, and the cladding can he made up of gel that reflects signals back into the fiber to reduce signal loss. There are two types of fiber-optic cables: single-mode fiber (SMF) and multimode fiber (MMF).

- SMF uses a single ray of light, known as a mode, to carry the transmission over long distances.
- MMF uses multiple rays of light (modes) simultaneously, with each ray of light running at a different reflection angle to carry the transmission over short distances.

Advantages

- Fiber optic cable supports up to 1000 stations and can carry the signal up to and beyond 2 kilometers.
- Fiber optic cables are also highly secure from outside interference, such as radio transmitters, arc welders, fluorescent lights, and other sources of electrical noise.
- Fiber optic does not generate electromagnetic radiation.
- Fiber optic cables are thinner and lighter than copper cables.
- Fiber optic cable can support data transfer speeds of up to 10 Gbps (10000 Mbps).
- Fiber optic transmission is more secure. Taps can be easily detected.

Disadvantages

- Fiber optic cable is by far the most expensive of these cabling methods, and a small network is unlikely to need these features.
- Specialized equipment is required to install and maintain fiber optic network.

Wireless Media

Wireless networks use high frequency radio signals, infrared beams or lasers to communicate between workstations and network devices. Each device on the network has a transceiver to send the receive data. For long distance wireless communications make use of microwaves.

Radio Waves

Radio waves can be generated easily and travel a long distance. Radio frequency ranges from 3 KHz to 1 GHz. They can travel in all directions and can easily penetrate through obstacles. They have low bandwidth for data communications.

Microwaves

Microwave communication can take place between two earth based stations or between an earth based station and a satellite. Microwaves travel in a straight line. A parabolic antenna is mounted at a height and transmits a narrow beam of signal to the receiving antenna. Microwave communication is expensive as compared to wired communication. Microwaves provides large bandwidth for data transfer. Telephone relay towers make use of microwave communication.

Infrared and Millimeter Waves

Unguided infrared and millimeter waves are widely used for short range communication (100 feet). Infrared communication is used in remote controls of television. It is relatively directional, cheap and easy to generate. Major drawback of these waves is that they cannot pass through solid objects. This also means that an infrared system in one room will not interfere with infrared system in any other room. Bandwidths of up to 10 Mbps can be supported by infrared transmissions.

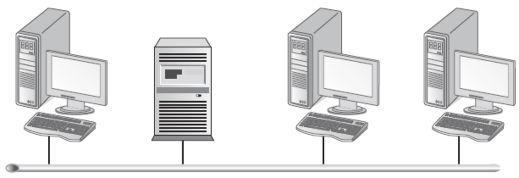
Network Topologies

A network topology is the physical layout of computers, cables and other components of the network. There are a number of different network topologies and a network may be build using multiple topologies.

Bus Topology

A bus topology uses one cable as a main trunk to connect all of the systems together. A bus topology is very easy to set up and requires no additional hardware such as a hub. The cable is also called a trunk, a backbone, or a segment.

With a bus topology, when a computer sends out a signal, the signal travels the cable length in both directions from the sending computer. When the signal reaches the end of the cable length, it bounces back and returns in the direction it came from. This is known as signal bounce. Signal bounce is a problem, because if another signal is sent on the cable length at the same time, the two signals will collide and be destroyed and then must be retransmitted. For this reason, at each end of the cable there is a terminator. The terminator is designed to absorb the signal when the signal reaches the end, preventing signal bounce. If there is no termination, the entire network fails because of signal bounce, which also means that if there is ever a break in the cable, you will have unterminated ends and the entire network will go down.



A bus is a passive topology, which means that the workstations on the bus are not responsible for regenerating the signal as it passes by them. Since the workstations do not play an active role, the workstations are not a requirement of a functioning bus, which means that if a workstation fails, the bus does not fail. But if there is an unterminated end in the bus, the entire network will fail.

Advantages of a Bus Topology

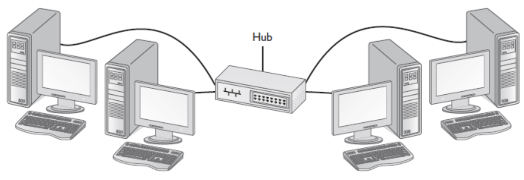
- One advantage of a bus topology is cost.
- A bus topology uses less cable than a star topology or a mesh topology. There is no need to purchase any additional devices such as hubs.
- Another advantage of a bus topology is the ease of installation. With a bus topology, one can simply connect the workstation to the cable segment or backbone.
- Most importantly, if a computer fails, the network stays functional.

Disadvantages of a Bus Topology

- It has a single point of failure. The entire network shuts down if there is a break in the main cable.
- The other disadvantage of a bus topology is the difficulty in troubleshooting it. When the network goes down, it is usually due to a break in the cable segment. With a large network, this problem can be tough to isolate.
- The bus topology is not very scalable. Its performance degrades as additional computers are added.

Star Topology

In a star topology, all computers are connected through one central device known as a hub or a switch. Each workstation has a cable that goes from the network card to the hub device. One of the major benefits of a star topology is that a break in the cable causes only the workstation that is connected to the cable to go down, not the entire network, as with a bus topology. Star topologies are very popular topologies in today's networking environments.



Advantages of a Star Topology

- One advantage of a star topology is scalability and ease of adding another system to the network. To add another workstation to the network with a star topology, one simply needs to connect that system to an unused port on the hub.

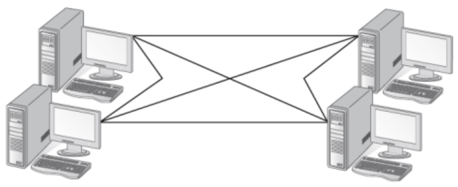
- Another benefit is the fact that if there is a break in the cable it affects only the system that is connected to that cable.
- Star topology is easier to administer and easy to monitor network traffic.
- It is also easy to add or change configurations because all of the connections come to a central point.

Disadvantages of a Star Topology

- If the hub fails in a star topology, the entire network comes down, so it has a central point of failure. But this is a much easier problem to troubleshoot than trying to find a cable break with a bus topology.
- Another disadvantage of a star topology is cost. To connect each workstation to the network a hub with an available port will be required and in addition a cable will have to go from the workstation to the hub. Today, the cost is increasingly less of a disadvantage because of the low prices of devices such as hubs and switches.

Mesh Topology

A mesh topology is not very common in computer networking today. In a mesh, every node has a dedicated point to point link to every other node in the network. This means that each node carries traffic only between the two nodes it connects.



Advantages of a Mesh Topology

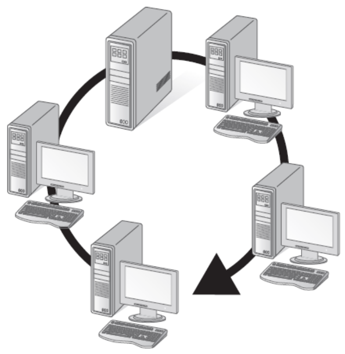
- The biggest advantage of a mesh topology is fault tolerance, meaning that, if there is a break in a cable segment, traffic can be rerouted through a different pathway because there are multiple pathways to send data from one system to another. This fault tolerance means that it is almost impossible for the network to go down due to a cable fault.
- Network congestion is generally not a problem because of the dedicated links between nodes.

Disadvantages of a Mesh Topology

- A disadvantage of a mesh topology is the cost of the additional cabling and network interfaces to create the multiple pathways between each system.
- A mesh topology is very hard to install and administer because of the numerous connections.

Ring Topology

In a ring topology, all computers are connected via a cable that loops in a ring or circle. A ring topology is a circle that has no start and no end. Because there are no ends, terminators are not necessary in a ring topology. Signals travel in one direction on a ring while they are passed from one computer to the next, with each computer regenerating the signal so that it may travel the distance required.



Advantages of Ring Topology

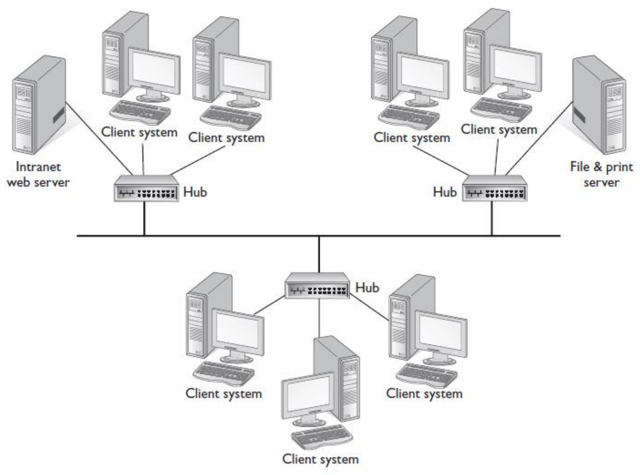
- A major advantage of a ring topology is that signal degeneration is low because each workstation is responsible for regenerating or boosting the signal. Because each workstation in a ring topology regenerates the signal, the signal is stronger when it reaches its destination and seldom needs to be retransmitted.

Disadvantages of a Ring Topology

- The biggest problem with ring topologies is that if one computer fails or the cable link is broken, the entire network could go down.
- Isolating a problem can be difficult in some ring configurations.
- Another disadvantage is that if you make a cabling change to the network or move a workstation, the brief disconnection can interrupt or bring down the entire network.

Hybrid Topology (Tree Topology)

It is typical for networks to implement a mixture of topologies to form a hybrid topology. For example, a very popular hybrid topology is a star-bus topology, in which a number of star topologies are connected by a central bus.



Wireless Topology

A wireless topology is one in which few cables are used to connect systems. The network is made of transmitters that broadcast the packets using radio frequencies. The transmitters (a.k.a. wireless access points) are connected to the network through a hub or a switch. Clients connect wirelessly to the access points to access the network. The main advantage of wireless topology is the lack of cabling. The main disadvantage is signal interference.

Access Methods

An access method determines how a host will place data on the wire. Does the host need to wait for its turn or can it place data on the wire whenever it wants? The answer to this is determined by three major access methods – CSMA/CD, Token passing and Polling.

Contention

The simplest method of access is contention. In this all systems share a common transmission channel. The message sent it broadcasted across the network. All systems can listen to the transmission but only the target system will respond to the message while all other systems will ignore the message. A system transmits message without checking the availability of the channel. This can lead to message from one system overlapping with another resulting in a collision.

(CSMA/CD)

Carrier sense multiple access with collision detection (CSMA/CD) is one of the most popular access methods in use today. With CSMA/CD, every host has equal access to the wire (MA) and can place data on the wire when the wire is free from traffic. If a host wishes to place data on the wire, it will “sense” the wire and determine whether there is a signal already on the wire (CS). If there is, the host will wait to transmit the data; if the wire is free, the host will send the data.

The problem with the process just described is that, if there are two systems on the wire that “sense” the wire at the same time to see if the wire is free, they will both send data out at the same time if the wire is free. When the two pieces of data are sent out on the wire at the same time, they will collide with one another, and the data will be destroyed. If the data is destroyed in transit, the data will need to be retransmitted. Consequently, after a collision, each host will wait a variable length of time before retransmitting the data (they don’t want the data to collide again), thereby preventing a collision the second time. When a system determines that the data has collided and then retransmits the data, it is known as collision detection (CD).

Token Passing

With both contention and CSMA/CD, the possibility of collisions is always there, and the more hosts that are placed on the wire, the greater the chances of collisions, because you have more systems “waiting” for the wire to become free so that they can send their data.

Token passing takes a totally different approach to deciding on how a system can place data on the wire. With token passing, there is an empty packet running around on the wire – the “token.” In order to place data on the wire, you need to wait for the token; once you have the token and it is free of data, you can place your data on the wire. Since there is only one token and a host needs to have the token to “talk,” it is impossible to have collisions in a token-passing environment.

For example, if Workstation 1 wants to send data on the wire, the workstation would wait for the token, which is circling the network millions of times per second. Once the token has reached Workstation 1, the workstation would take the token off the network, fill it with data, mark the token as being used so that no other systems try to fill the token with data, and then place the token back on the wire heading for the destination host. All systems will look at the data, but they will not process it, since it is not destined for them. However, the system that is the intended destination will read the data and send the token back to the sender as a confirmation. Once the token has reached the original sender, the token is un-flagged as being used (marked as free) and released as an empty token onto the network.

Polling

A polling network uses polling to control access to the node. Each system or node is given exclusive access to the network in a predetermined order. Permission to transmit on the network is passed from system to system using a special message called a poll. Polling may be centralized (called as hub polling) or decentralized (distributed polling). In hub polling, the polling order is maintained by a single central hub. When a system finishes its turn of transmitting data, it sends a message to the hub which then forwards the poll to the next station in the sequence. In a decentralized polling scheme, each system knows its successor in the polling sequence and sends the poll directly to that station.

Network Architectures

Ethernet

Ethernet is the most popular LAN architecture in the world and connects almost 80% of LAN devices. It is defined by IEEE (Institute for Electrical and Electronics Engineers) as 802.3 standard. Ethernet makes use of CSMA/CD as the access method. It is implemented on NICs and is part of the link layer of a network.

10Base2

- Typically implemented as a bus topology, but it could be a mix of star and bus topology.
- Uses of thinnet cable and BNC connectors.
- Baseband communication.
- Data transfer rate is 10 Mbps.
- Maximum distance of 185 meters per network segment.
- A maximum of 30 hosts can be connected per segment.
- There must be 0.5 meters minimum distance between hosts.

10Base5

- Uses of thicknet (co-axial) cable and AUI connectors. The thicker cable allows the signal to travel farther than it is possible with thinnet cable.

- Baseband communication.
- Data transfer rate is 10 Mbps.
- Maximum distance of 500 meters per network segment.
- A maximum of 100 hosts can be connected per segment.
- There must be 2.5 meters minimum distance between hosts.

10BaseT

- Uses of CAT 3 UTP cable or higher and RJ-45 connector.
- Baseband communication.
- Hub and NIC are used to connect devices.
- Device can be 100 m from the hub.
- Data transfer rate is 10 Mbps.

10BaseFL

- Uses fiber optic cable as the backbone. This allows the network to reach greater distances.
- Data transfer rate is 10 Mbps.

Fast Ethernet (100BaseTX and 100BaseFX)

- Implemented using star topology.
- Baseband communication.
- Data transfer rate is 100 Mbps.
- Uses two pairs of CAT 5 UTP cable for TX and two stands of fiber for FX.

Gigabit Ethernet

- It is becoming the de facto standard for network architectures today.
- Can be implemented using co-axial, twisted pair and fiber optic cables.
- Data transfer rate is 1000 Mbps (1 Gbps).

Token Ring

A big competitor to Ethernet in the past was Token Ring, which runs at 4 Mbps or 16 Mbps. Token Ring is a network architecture that uses a star ring topology and can use many forms of cables. Token Ring uses the token-passing access method. It is impossible to have collisions in a token passing environment. Token ring is defined as IEEE 802.5 standard.

Network Devices

Network Interface Cards

Network interface cards, commonly referred to as NICs are used to connect a PC to a network. The NIC provides a physical connection between the networking cable and the computer's internal bus. The larger the number of bits that can be transferred to the NIC, the faster the NIC can transfer data to the network cable.

Three types of NICs available are – Ethernet card, LocalTalk connectors and Token Ring cards.

Modem

A modem is a device that makes it possible for computers to communicate over telephone lines. The word modem comes from Modulate and Demodulate. Because standard telephone lines use analog signals, and computers digital signals, a sending modem must modulate its digital signals into analog signals. The computers modem on the receiving end must then demodulate the analog signals into digital signals.

Modems can be external, connected to the computers serial port by an RS-232 cable or internal in one of the computers expansion slots. Modems connect to the phone line using standard telephone RJ-11 connectors.

Repeaters

A signal weakens as it travels through a medium. This is known as attenuation. A repeater is a physical layer device that is installed to amplify the signal so that the signal can be transmitted over a long distance.

The kind of amplification done by the repeater is different from the regular amplification by amplifiers. The regular amplifies everything fed into it. That means, if the input signal has noise induced into it, both the desired signal and noise signal are together amplified. But, in the case of a repeater, it regenerates the input signal, and amplifies only the desirable signal. Hence, the noise component of the signal is eliminated.

Hubs

A hub works in the physical layer of the OSI model. It is a non-intelligent device, and has no decision making capability. A hub basically takes the input data from one of the ports and broadcast the information to all the other ports connected to the network.

Consider a 4 port network. There are 4 computers connected to the 4 ports. Suppose, if Computer A wants to send some data to Computer B using a hub, then, Computer A broadcasts the data on the network, and Computer B, being connected to the network, has access to the data. But, in this case all the other ports connected to the network has access to the data that is being transmitted by Computer A. This happens because, the hub works in the Physical Layer and hence it does not know about the MAC addresses of the ports connected to the network. So, there is a lack of security in the Hub.

Hubs can be active or passive. An active hub strengthens and regenerates the incoming signals before sending the data on to its destination. Passive hubs do nothing with the signal.

Switch

A switch is an intelligent device that works in the data link layer. The term intelligent refers to the decision making capacity of the Switch. Since it works in the Data link layer, it has knowledge of the MAC addresses of the ports in the network.

Consider a 4 port network. If data has to be sent from Computer A to Computer B, then, the data is transferred to the Computer B only, and not to any other computers connected on the network. Hence, it establishes a link between the sender and the receiver based on the MAC addresses. This also means that when data is being sent from A to B, Computer C can establish a link with Computer D and communication can take place between them. So, simultaneous data transfer is possible in a switch. Hub divides bandwidth, but a switch does not.

Also a switch is a secure device, because it sends information only to the desired destinations, and also certain security features such as firewalls can be implemented in the Switches.

Bridge

A bridge is used to join two network segments together, it allows computers on either segment to access resources on the other. They can also be used to divide large networks into smaller segments. Bridges have all the features of repeaters, but can have more nodes, and since the network is divided, there is fewer computers competing for resources on each segment thus improving network performance.

Bridges can also connect networks that run at different speeds, different topologies, or different protocols. But they cannot, join an Ethernet segment with a Token Ring segment, because these use different networking standards. Bridges operate at both the Physical Layer and the MAC sublayer of the Data Link layer. Bridges read the MAC header of each frame to determine on which side of the bridge the destination device is located, the bridge then repeats the transmission to the segment where the device is located.

Router

Routers are networking devices used to extend or segment networks by forwarding packets from one logical network to another. Routers are most often used in large internetworks that use the TCP/IP protocol suite and for connecting TCP/IP hosts and local area networks (LANs) to the Internet using dedicated leased lines.

Routers work at the network layer (layer 3 of the Open Systems Interconnection OSI reference model for networking) to move packets between networks using their logical addresses (which, in the case of TCP/IP, are the IP addresses of destination hosts on the network). Because routers operate at a higher level than bridges do, they have better packet-routing and filtering capabilities and greater processing power, which results in routers costing more than bridges.

APPENDIX

8085 Instruction Set

DATA TRANSFER GROUP

Move

MOV	A,A	7F
	A,B	78
	A,C	79
	A,D	7A
	A,E	7B
	A,H	7C
	A,L	7D
	A,M	7E
MOV	B,A	47
	B,B	40
	B,C	41
	B,D	42
	B,E	43
	B,H	44
	B,L	45
	B,M	46
MOV	C,A	4F
	C,B	48
	C,C	49
	C,D	4A
	C,E	4B
	C,H	4C
	C,L	4D
	C,M	4E
MOV	D,A	57
	D,B	50
	D,C	51
	D,D	52
	D,E	53
	D,H	54
	D,L	55
	D,M	56

Move (cont)

MOV	E,A	5F
	E,B	58
	E,C	59
	E,D	5A
	E,E	5B
	E,H	5C
	E,L	5D
	E,M	5E
MOV	H,A	67
	H,B	60
	H,C	61
	H,D	62
	H,E	63
	H,H	64
	H,L	65
	H,M	66
MOV	L,A	6F
	L,B	68
	L,C	69
	L,D	6A
	L,E	6B
	L,H	6C
	L,L	6D
	L,M	6E
MOV	M,A	77
	M,B	70
	M,C	71
	M,D	72
	M,E	73
	M,H	74
	M,L	75
XCHG		EB

Move Immediate

MVI	A, byte	3E
	B, byte	06
	C, byte	0E
	D, byte	16
	E, byte	1E
	H, byte	26
	L, byte	2E
	M, byte	36

Load Immediate

LXI	B, dble	01
	D, dble	11
	H, dble	21
	SP, dble	31

Load/Store

LDAX B	0A
LDAX D	1A
LHLD adr	2A
LDA adr	3A
STAX B	02
STAX D	12
SHLD adr	22
STA adr	32

byte = constant, or logical/arithmetic expression that evaluates to an 8-bit data quantity. (Second byte of 2-byte instructions).

dble = constant, or logical/arithmetic expression that evaluates to a 16-bit data quantity. (Second and Third bytes of 3-byte instructions).

adr = 16-bit address (Second and Third bytes of 3-byte instructions).

* = all flags (C, Z, S, P, AC) affected.

** = all flags except CARRY affected; (exception: INX and DCX affect no flags).

† = only CARRY affected.

All mnemonics copyright ©Intel Corporation 1976.

ARITHMETIC AND LOGICAL GROUP

Add*

ADD	A	87
	B	80
	C	81
	D	82
	E	83
	H	84
	L	85
	M	86
ADC	A	8F
	B	88
	C	89
	D	8A
	E	8B
	H	8C
	L	8D
	M	8E

Subtract*

SUB	A	97
	B	90
	C	91
	D	92
	E	93
	H	94
	L	95
	M	96
SBB	A	9F
	B	98
	C	99
	D	9A
	E	9B
	H	9C
	L	9D
	M	9E

Double Add †

DAD	B	09
	D	19
	H	29
	SP	39

Increment**

INR	A	3C
	B	04
	C	0C
	D	14
	E	1C
	H	24
	L	2C
	M	34
INX	B	03
	D	13
	H	23
	SP	33

Decrement**

DCR	A	3D
	B	05
	C	0D
	D	15
	E	1D
	H	25
	L	2D
	M	35
DCX	B	0B
	D	1B
	H	2B
	SP	3B

Specials

DAA*	27
CMA	2F
STC†	37
CMC†	3F

Rotate †

RLC	07
RRC	0F
RAL	17
RAR	1F

Logical*

ANA	A	A7
	B	A0
	C	A1
	D	A2
	E	A3
	H	A4
	L	A5
	M	A6
XRA	A	AF
	B	A8
	C	A9
	D	AA
	E	AB
	H	AC
	L	AD
	M	AE
ORA	A	B7
	B	B0
	C	B1
	D	B2
	E	B3
	H	B4
	L	B5
	M	B6
CMP	A	BF
	B	B8
	C	B9
	D	BA
	E	BB
	H	BC
	L	BD
	M	BE

Arith & Logical Immediate

ADI byte	C6
ACI byte	CE
SUI byte	D6
SBI byte	DE
ANI byte	E6
XRI byte	EE
ORI byte	F6
CPI byte	FE

BRANCH CONTROL GROUP

Jump

JMP adr	C3
JNZ adr	C2
JZ adr	CA
JNC adr	D2
JC adr	DA
JPO adr	E2
JPE adr	EA
JP adr	F2
JM adr	FA
PCHL	E9

Call

CALL adr	CD
CNZ adr	C4
CZ adr	CC
CNC adr	D4
CC adr	DC
CPO adr	E4
CPE adr	EC
CP adr	F4
CM adr	FC

Return

RET	C9
RNZ	C0
RZ	C8
RNC	D0
RC	D8
RPO	E0
RPE	E8
RP	F0
RM	F8

Restart

RST	0	C7
	1	CF
	2	D7
	3	DF
	4	E7
	5	EF
	6	F7
	7	FF

I/O AND MACHINE CONTROL

Stack Ops

PUSH	B	C5
	D	D5
	H	E5
	PSW	F5
POP	B	C1
	D	D1
	H	E1
	PSW*	F1
XTHL		E3
SPHL		F9

Input/Output

OUT byte	D3
IN byte	DB

Control

DI	F3
EI	FB
NOP	00
HLT	76

New Instructions (8085 Only)

RIM	20
SIM	30

ASSEMBLER REFERENCE

Operators

( )
NUL
LOW, HIGH
*, /, MOD, SHL, SHR
+, —
NOT
AND
OR, XOR

ASSEMBLER REFERENCE (Cont.)

Pseudo Instruction

General:
ORG
END
EQU
SET
DS
DB
DW

Macros:
MACRO
ENDM
LOCAL
REPT
IRP
IRPC
EXITM

Relocation:

ASEG	NAME
DSEG	STKLN
CSEG	STACK
PUBLIC	MEMORY
EXTRN	

Conditional Assembly:
IF
ELSE
ENDIF

Constant Definition

0BDH, 1AH	Hex
105D, 105	Decimal
72O, 72Q	Octal
11011B, 00110B	Binary
'TEST', 'A' 'B'	ASCII