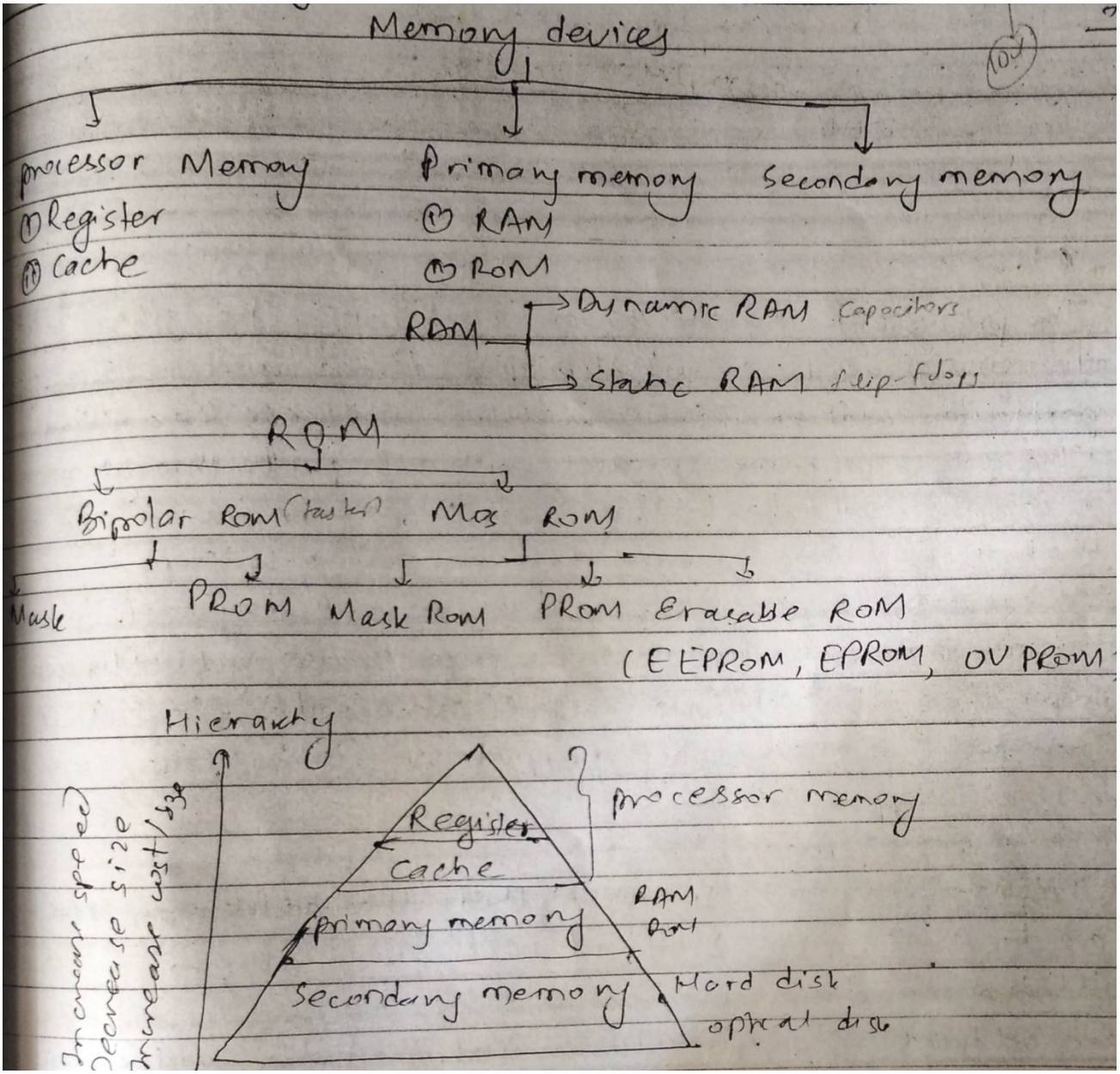
Memory Device Classification



For detail: Manual Page 96 to 101

Address decoding:

Microprocessor is connected with memory and I/O devices via common address and data bus. Only one device can send data at a time and other devices can only receive that data. If more than one device sends data at the same time, the data gets garbled. In order to avoid this situation, ensuring that the proper device gets addressed at proper time, the technique called address decoding is used.

The process of decoding or identifying the given address of memory unit or input/output device and activating particular unit or device is called address decoding.

Depending upon the no. of address lines used to generate chip select signal for the device, the address decoding is classified as:

1. **I/O mapped I/O**

In this method, a device is identified with an 8 bit address and operated by I/O related functions IN and OUT for that IO/$\overline{M}$ =1. Since only 8bit address is used, at most 256 devices can be identified uniquely. Generally low order address bits A_0-A_7 are used and upper bits A_8-A_{15} are considered don't care. Usually I/O mapped I/O is used to map devices like 8255A, 8251A etc.

2. **Memory mapped I/O**

In this method , a device is identified with 16 bit address and enabled memory related functions such as STA , LDA for which IO/$\overline{M}$ =0, here chip select signal of each device is derived from 16 bit address lines thus total addressing capability is 64K bytes . Usually memory mapped I/O is used to map memories like RAM, ROM etc.

Depending on the address that are allocated to the device the address decoding are categorized in the following two groups.

1. **Unique (absolute) Address Decoding:**

If all the address lines on that mapping mode are used for address decoding then that decoding is called unique address decoding. It means all 8-lines in I/O mapped I/O and all 16 lines in memory mapped I/O are used to derive $\overline{CS}$ signal. It is expensive and complicated but fault proof in all cases.

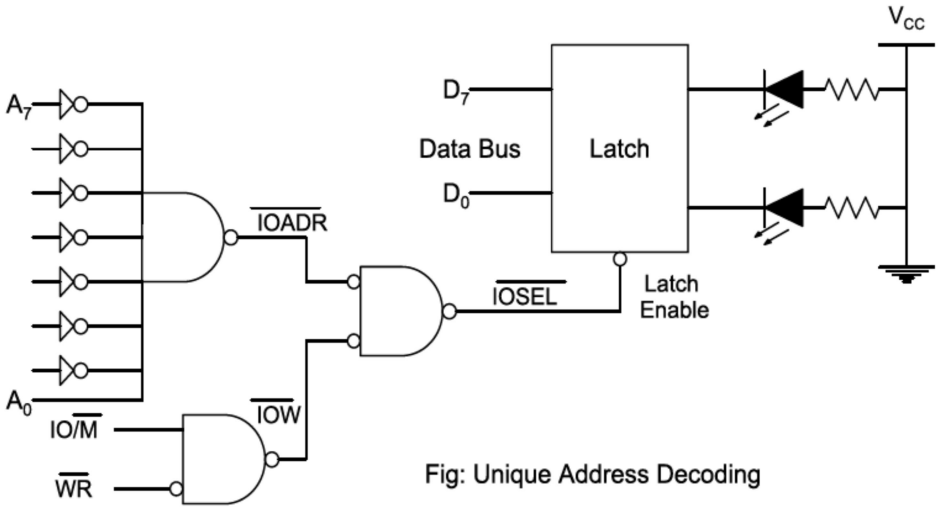


Fig: Unique Address Decoding

- If A_0 is high and A_1- A_7 are low and if IOW becomes low, the latch gets enabled.
- The data to the LED can be transferred in only one case and hence the device has unique address of 01H.

2. **Non Unique (partial) Address decoding:**

If all the address lines available on that mode are not used in address decoding then that decoding is called non unique address decoding. Though it is cheaper there may be a chance of address conflict.

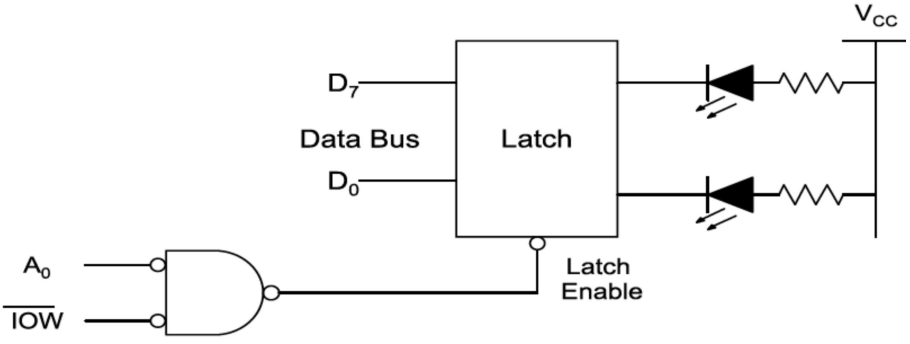
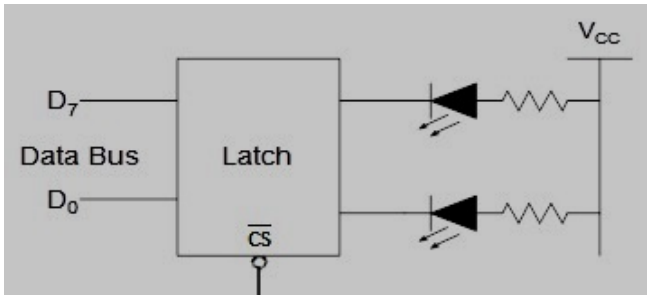


Fig: Non unique Address Decoding

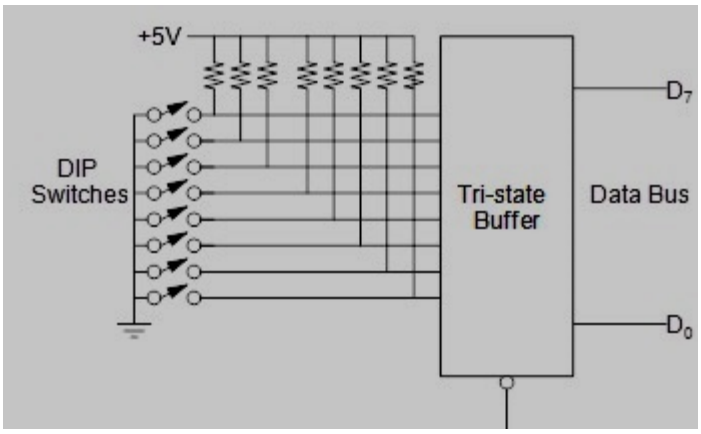
- If A_0 is low and $\overline{IOW}$ is low, then latch gets enabled.
- Here A_1-A_7 is neglected that is any even address can enable the latch.

For address decoding

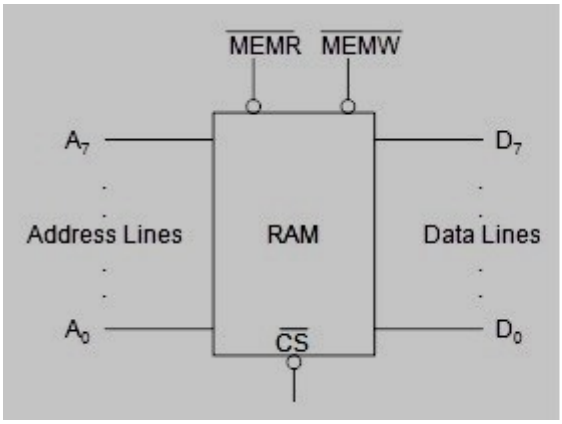
Output Port



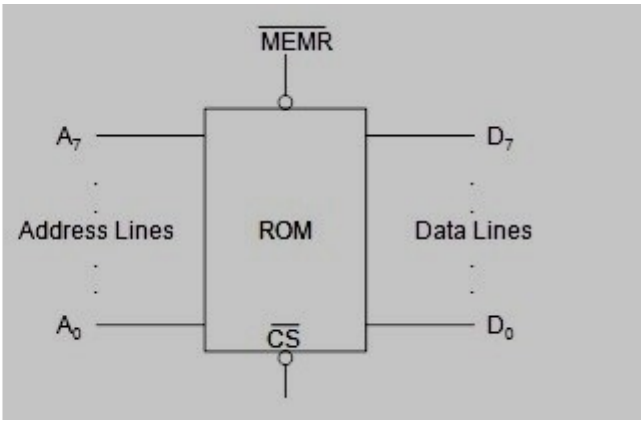
Input Port



RAM



ROM

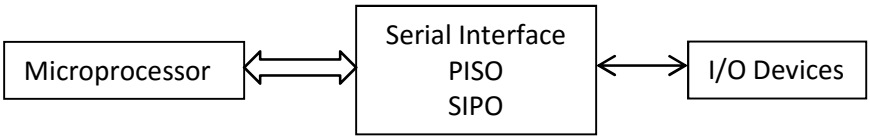


Examples….

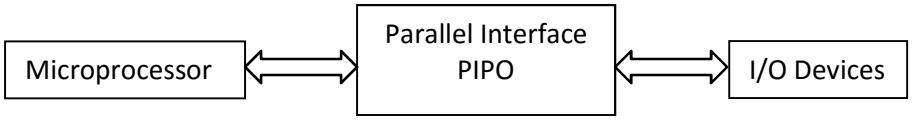
Input/output Interface

Input/output interface is a circuitry that connects the input/output devices to the microcomputer for interfacing of typical microcomputer to common peripherals.

a) Serial Interface



b) Parallel Interface



Data Transfer Methods

1. Serial Data Transfer

Two types of serial data transfer

- **Synchronous serial data transfer**
  - Data is transmitted or received based on a clock signal i. e. synchronously.
  - The transmitting device sends a data bit at each clock pulse.
  - Usually one or more SYNC characters are used to indicate the start of each synchronous data stream.
  - SYNC characters for each frame of data.
  - Transmitting device sends data continuously to the receiving device. If the data is not ready to be transmitted, the transmitter will send SYNC character until the data is available.
  - The receiving device waits for data, when it finds the SYNC characters then starts interpreting the data.

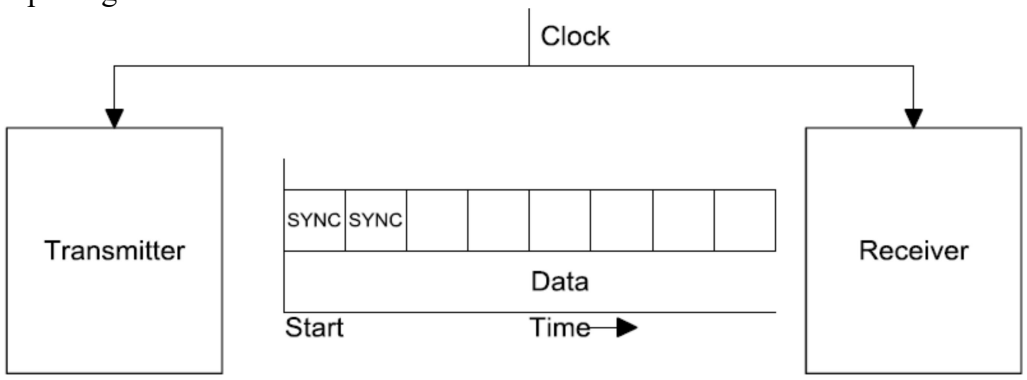


Fig: Synchronous Serial Transmission Format

- **Asynchronous serial data transfer**
  - The receiving device does not need to be synchronized with the transmitting device.
  - Transmitting device send data units when it is ready to send data.
  - Each data unit must contain start and stop bits for indicating beginning and the end of data unit. And also one parity bit to identify odd or even parity data.
  - For e.g. To send ASCII character (7 bit)

    We need: 1 start bit: beginning of data
    1 stop bit: End of data
    1 Parity bit: even or odd parity
    7 or 8 bit character: actual data transferred

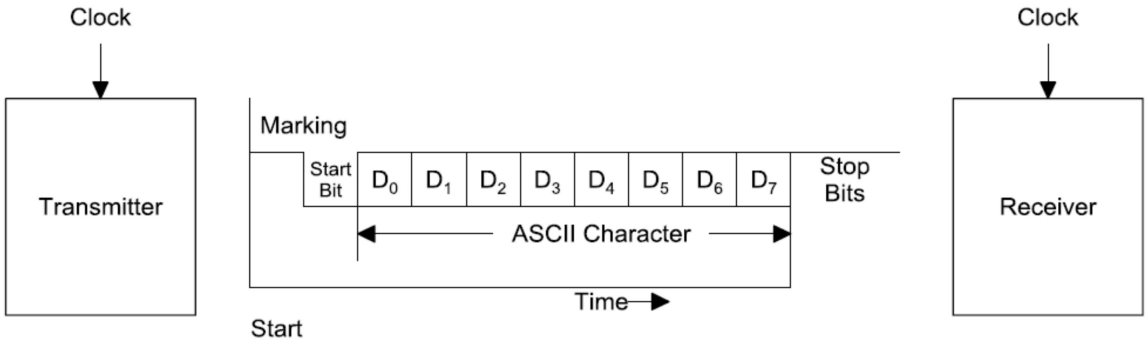


Fig: Asynchronous Serial Transmission Format

2. Parallel Data Transfer

Four methods of parallel data transfer

- **Simple Input/Output**

For simple I/O, the buffer switch and latch switches are always connected to the input and output ports. The devices are always ready to send or receive data.

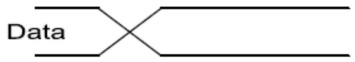


Fig: Simple I /O

- **Simple Strobe Input/output**

In many applications, valid data is present on an external device only at a certain time and must be read in at that time. Here a strobe pulse is supplied to indicate the time at which data is being transmitted. For an example, we can discuss the ASCII encoded keyboard. When a key is pressed, circuitry on keyboard sends out ASCII code for pressed key on eight parallel data lines and then sends out a strobe signal on another line to indicate that valid data is present on eight data lines.

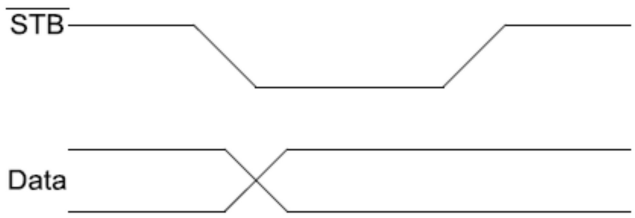


Fig: Simple Strobe I/O

- **Single Handshake Input/output**

- The peripheral outputs some data and send $\overline{STB}$ signal to MP. "here is the data for you."
- MP detects asserted $\overline{STB}$ signal, reads the data and sends an acknowledge signal (ACK) to indicate data has been read and peripheral can send next data. "I got that one, send me another."
- MP sends or receives data when peripheral is ready.

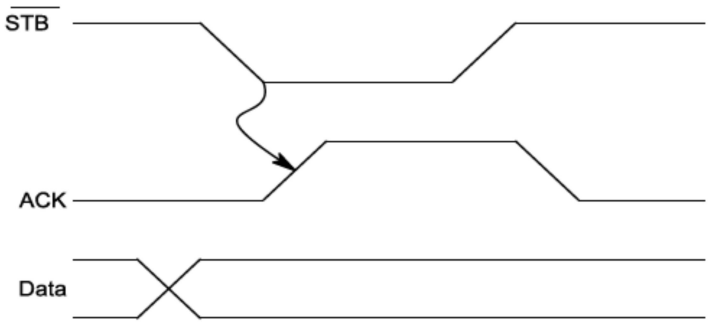


Fig: Single Handshaking

- **Double Handshake Input/output**

- The peripheral asserts its $\overline{STB}$ line low to ask MP "Are you ready?"
- The MP raises its ACK line high to say " I am ready".
- Peripheral then sends data and raises its $\overline{STB}$ line low to say "Here is some valid data for you."
- MP then reads the data and drops its ACK line to say, "I have the data, thank you, and I await your request to send the next byte of data."

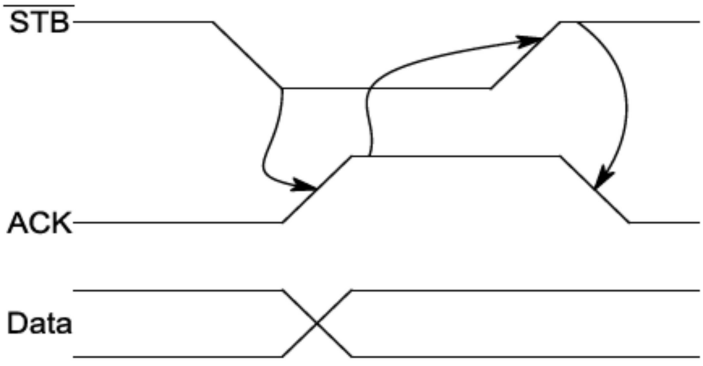


Fig: Double Handshaking

Programmable Peripheral Interface (PPI)

Programmable peripheral interface is an intermediate device between microprocessor and I/O devices. Since microprocessor don't have inbuilt I/O ports, PPI gives the facility of those ports so that microprocessor can communicate with I/O devices.

One of the popular PPI is 8255A. The common features of 8255A are:

- The INTEL 8255 is a device used to parallel data transfer between processor and slow peripheral devices like ADC, DAC, keyboard, 7-segment display, LCD, etc.
- The 8255 has three ports: Port-A, Port-B and Port-C.
- Port-A can be programmed to work in any one of the three operating modes mode-0, mode-1 and mode-2 as input or output port.
- Port-B can be programmed to work either in mode-0 or mode-1 as input or output port.
- Port-C (8-pins) has different assignments depending on the mode of port-A and port-B.
- If port-A and B are programmed in mode-0, then the port-C can perform any one of the following functions.
  - As 8-bit parallel port in mode-0 for input or output.
  - As two numbers of 4-bit parallel ports in mode-0 for input or output.
  - The individual pins of port-C can be set or reset for various control applications.
- If port-A is programmed in mode- 1/mode-2 and port-B is programmed in mode-1 then some of the pins of port-C are used for handshake signals and the remaining pins can be used as input/ output lines or individually set/reset for control application.

Key Features of Mode-0,Mode-1 and Mode-2

- *Mode 0: Ports A and B operate as either inputs or outputs and Port C is divided into two 4-bit groups either of which can be operated as inputs or outputs*
- *Mode 1: Same as Mode 0 but Port C is used for handshaking and control*
- *Mode 2: Port A is bidirectional (both input and output) and Port C is used for handshaking. Port B is not used.*

Serial Interface Standards

RS -232

- Serial transmission of data is used as an efficient means for transmitting digital information across long distances, the existing communication lines usually the telephone lines can be used to transfer information which saves a lot of hardware.
- RS-232C is an interface developed to standardize the interface between data terminal equipment (DTE) and data communication equipment (DCE) employing serial binary data exchange. Modem and other devices used to send serial data are called data communication equipment (DCE). The computers or terminals that are sending or receiving the data are called data terminal.
- Equipment (DTE) RS- 232C is the interface standard developed by electronic industries Association (EIA) in response to the need for the signal and handshake standards between the DTE and DCE.
- It uses 25 pins (DB – 25P) or 9 Pins (DE – 9P) standard where 9 pin standard does not use all signals i. e. data, control, timing and ground.
- It describes the voltage levels, impendence levels, rise and fall times, maximum bit rate and maximum capacitance for all signal lines.
- It also specifies that DTE connector should be male and DCE connector should be female.
- It can send 20kBd for a distance of 50 ft.
- The voltage level for RS-232 are:

- A logic high or 1 , 3V to -15V
- A logic low or 0, +3v to +15v
- Normally ±12V voltage levels are used

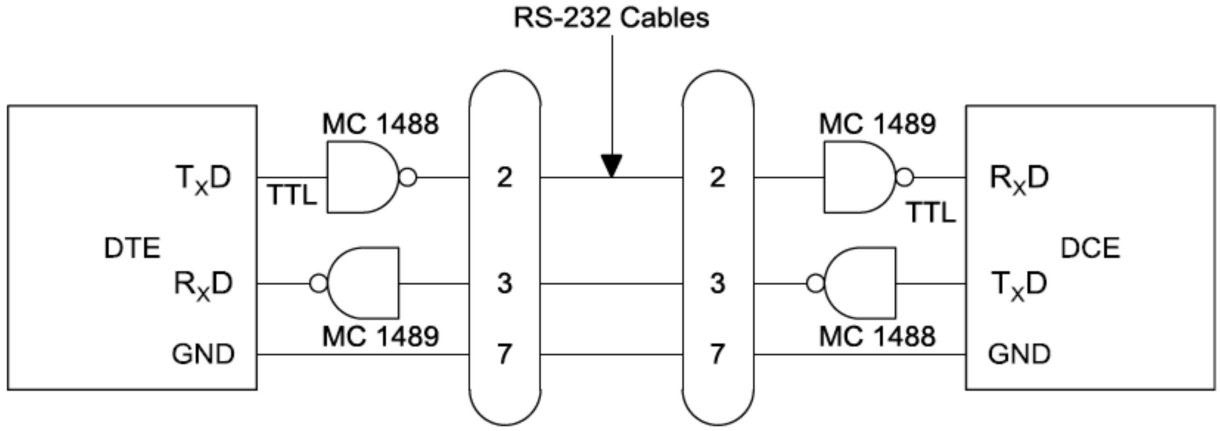


Fig: Connection of DTE and DCE through RS-232C Interface

- Mc1488 converts logic 1 to -9V
  logic 0 to +9v
- Mc1485 converts RS – 232 to TTL
- Signal levels of RS-232 are not compatible with that of the DTE and DCE which are TTL signals for that line driver such as M 1488 and line receiver MC1485 are used.

RS 423A and RS 422A serial standards

RS 423A

- A major problem with RS-232C is that it can only transmit data reliably for about 50 ft at its maximum rate of 20Kbd. If longer lines are used the transmission rate has to be drastically reduced due to open signal lines with a common signal ground.
- Another EIA standard which is improvement over RS-232C is RS-423A.
- This standard specifies a low impendence single ended signal which can be sent over 50Ω coaxial cable and partially terminated at the receiving end to prevent reflection.
- RS-423 standard defines a unidirectional interface between one transmitter and many receivers.
- Voltage level: High 4 -6V negative
  Low 4-6V positive
- Transmission rate: 100 Kbd over 40 ft
  1 Kbd over 4000 ft

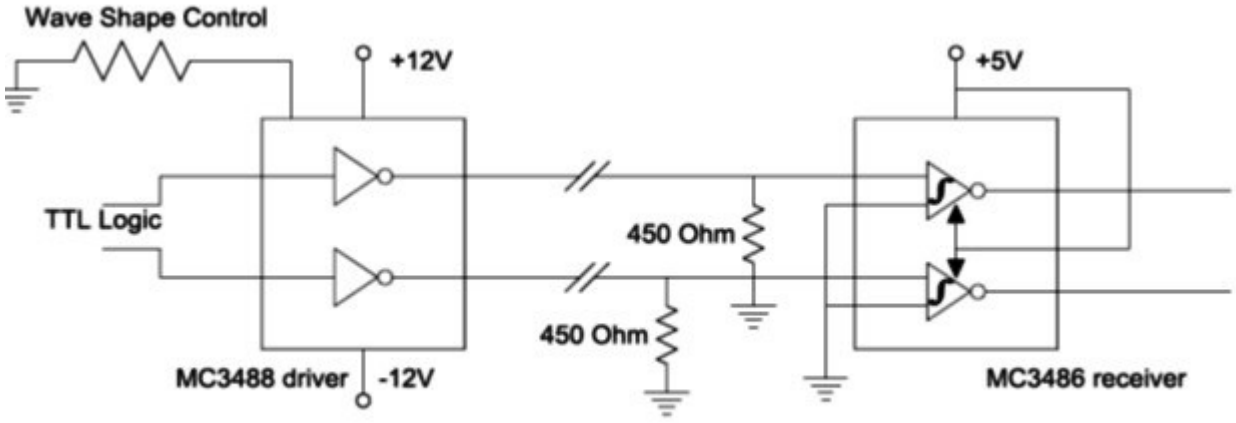


Fig: MC3488 driver and MC3489 receiver used for RS-423A Interface

RS 422A

- A newer standard for serial data transfer.
- It specifies that can signal will be send differentially over two adjacent wires in a ribbon cable or a twisted pair of wires uses differential amplifier to reject noise.

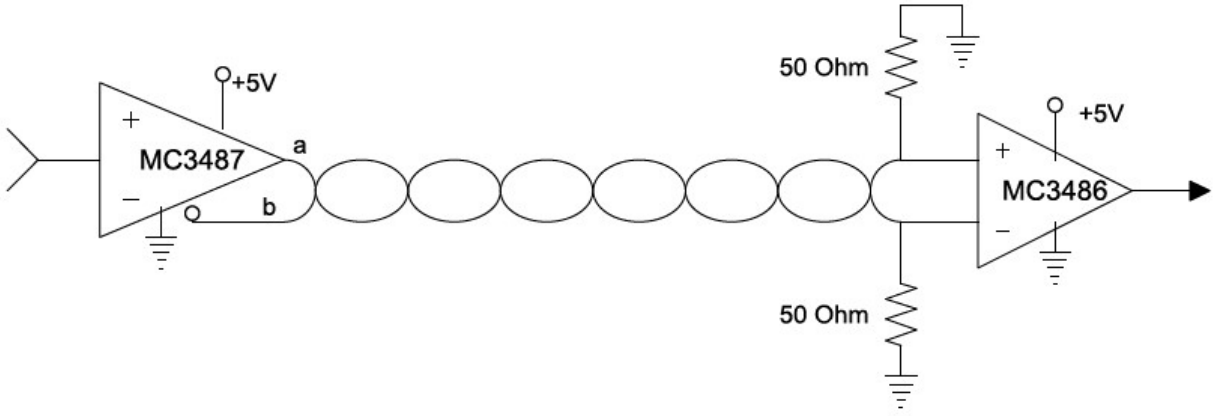


Fig: MC3487 driver and MC3486 receiver used for RS-422A Interface

- Logic high is transmitted by making 'b' line more positive than 'a' line.
- Logic low is transmitted by making 'a' line more positive than 'b' line.
- The voltage difference between the two lines must be greater than 0.4V but less than 12V.
- The mc3487 driver provides a differential voltage of about 2V.
- The center or common mode voltage on the lines must be between -7v and +7v
- Transmission rate is 10000 KBd for 40 FT
- 100 KBD for 4000 ft.
- The high data transfer is because of differential line functions as a fully terminated transmission line.
- Mc3486 receiver only responds to the differential voltage eliminating noise.

Universal Serial Bus (USB)

A new option for I/O interfacing is the Universal Serial Bus (USB), a project of a group that includes Intel and Microsoft. A single USB port can have up to 127 devices communicating at either 1.5 Megabits/second or 12 Megabits/second over a 4-wire cable. The USB standard also describes both the hardware interface and software protocols. Newer PCs may have a USB port built-in, but because it's so new, most existing computers can't use it without added hardware and software drivers.

- USB is an industry standard developed in the mid-1990s that defines the cables, connectors and protocols used for connection, communication and power supply between computers and electronic devices.
- USB was designed to standardize the connection of computer peripherals, such as keyboards, pointing devices, digital cameras, printers, portable media players, disk drives and network adapters to personal computers, both to communicate and to supply electric power. It has become commonplace on other devices, such as smart phones, PDAs and video game consoles. USB has effectively replaced a variety of earlier interfaces, such as serial and parallel ports, as well as separate power chargers for portable devices.

Introduction to Direct Memory Access (DMA) & DMA Controllers

It is a process where data is transferred between two peripherals directly without the involvement of the microprocessor. In situation in which the microprocessor controlled data transfer is too slow, DMA is generally used.
This process employs the HOLD pin on the microprocessor. The external DMA controller sends a signal on the HOLD pin to the microprocessor. The microprocessor completes the current operation and sends a signal on HLDA and stops using the buses. Once the DMA controller is done, it turns off the HOLD signal and the microprocessor takes back control of the buses.

Basic DMA operation

- The direct memory access (DMA) technique provides direct access to the memory while the microprocessor is temporarily disabled.
- A DMA controller temporarily borrows the address bus, data bus, and control bus from the microprocessor and transfers the data bytes directly between an I/O port and a series of memory locations.
- The DMA transfer is also used to do high-speed memory-to memory transfers.
- Two control signals are used to request and acknowledge a DMA transfer in the microprocessor-based system.
- The HOLD signal is a bus request signal which asks the microprocessor to release control of the buses after the current bus cycle.
- The HLDA signal is a bus grant signal which indicates that the microprocessor has indeed released control of its buses by placing the buses at their high-impedance states.

PROGRAMMABLE DMA CONTROLLER - INTEL 8257

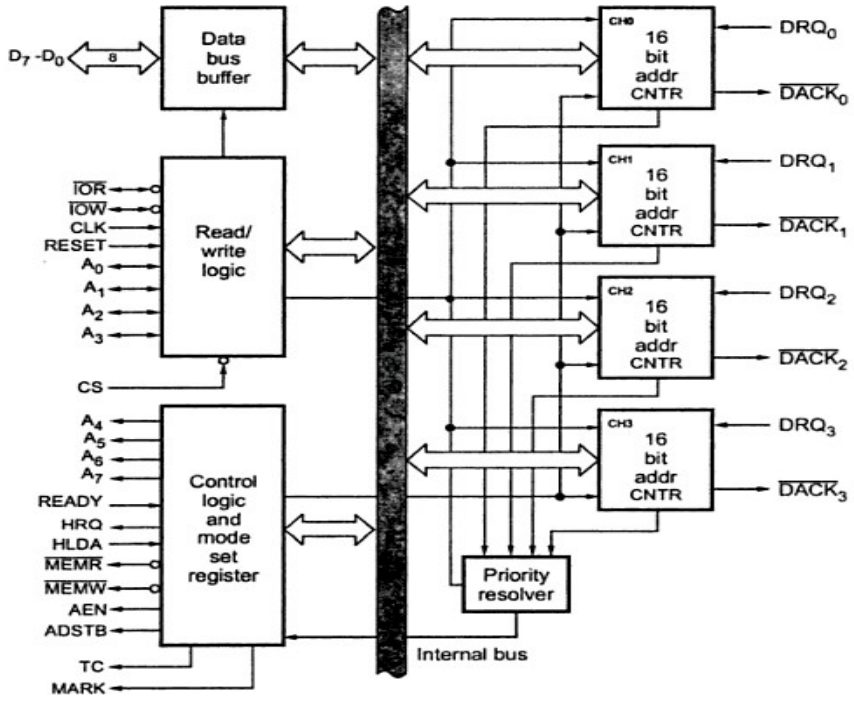


Fig: Functional block diagram of DMA Controller

It is a device to transfer the data directly between IO device and memory without through the CPU. So it performs a high-speed data transfer between memory and I/O device.

The features of 8257 is,

- The 8257 has four channels and so it can be used to provide DMA to four I/O devices.
- Each channel can be independently programmable to transfer up to 64kb of data by DMA.
- Each channel can be independently perform read transfer, write transfer and verify transfer.

Operation of 8257 DMA Controller

- Each channel of 8257 has two programmable 16-bit registers named as address register and count register.
- Address register is used to store the starting address of memory location for DMA data transfer.
- The address in the address register is automatically incremented after every read/write/verify transfer.
- The count register is used to count the number of byte or word transferred by DMA.
- In read transfer the data is transferred from memory to I/O device.
- In write transfer the data is transferred from I/O device to memory.
- Verification operations generate the DMA addresses without generating the DMA memory and I/O control signals.