

UNIT-3

NEED FOR HARDWARE INTERFACING - VARIOUS TYPES OF INTERFACING DEVICES - FUNCTIONS OF EACH OF THE ABOVE INTERFACING DEVICES.

i) Need for hardware Interfacing

Hardware interfacing: The interconnection of the peripheral devices, memory with in a microcomputer system is called interfacing.

Need for Hardware interfacing:

- * An interface is a shared boundary between two (or) more devices involving sharing of information.
 - * Suitable memory and I/O devices have to be selected to interface with processor for compatibility.
 - * If any peripheral device is not compatible, then additional logic circuitry has to be designed which is called interface circuit (or) interface.
- The different principles (or) methods of interfacing with peripherals or Input/Output (I/O) devices are,

i.) I/O mapped: In this method, I/O devices are considered as different devices with respective addressing. In order to interfacing the I/O devices, all the available processor address lines are not used.

ii.) Memory mapped: In this method, the devices are considered as memory locations with respective

addressing. In order to interface the memory devices, all the available processor address lines are used for address decoding.

Various types of Interfacing devices

The various types of interfacing devices are,

1. Keyboard interfacing

i.) Simple keyboard

ii.) Matrix keyboard

2. Display device interfacing

i.) LED

ii.) LCD

3. Converter device interfacing

i.) Analog to Digital converter (ADC)

ii.) Digital to Analog converter (DAC)

4. Sensor control device interfacing

i.) Temperature Sensors (LM35, AD590)

ii.) Pressure sensors

iii.) Light sensors etc

5. Motor control device interfacing

i.) Relays

ii.) opto isolator and opto couplers

iii.) Stepper motor and DC motors

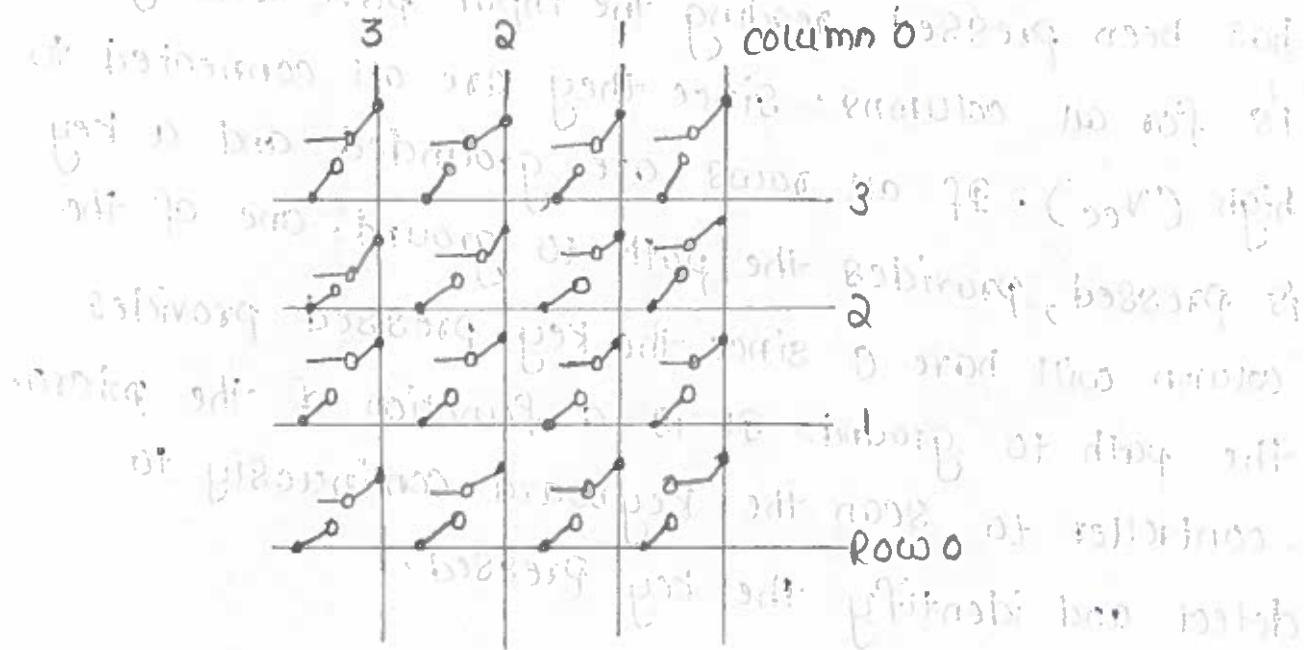
6. Memory device interfacing

i.) ROM

ii.) RAM etc.

(2)

7. 8255 programmable peripheral interfacing device
8. DS12887 real time clock device, etc.
- A keyboard is an external peripheral device that can be interfaced to microcontroller system. Generally, keyboard is used to provide input to the microcontroller system. There are various keyboard configurations, but the basic form is 4×4 matrix which is shown in figure.



Figure

The above figure shows sixteen keys in 4 rows and 4 columns. The connection between row and column is possible only when the switch is closed i.e; if the switch is kept open, then there won't be any connection between that particular row and column.

Initially all the keys are opened, no connection between rows and columns. Whenever a key is pressed, the particular switch is closed forming a short circuit between row and column. If the output line of shorted column is low, then its corresponding row line will become low.

Key press and detection method mechanism:

There are two methods to identify whether any key is pressed or not and which key has been pressed. The rows are connected to an output port and the columns are to the input port. If no key has been pressed, reading the input port will yield 1's for all columns. Since they are all connected to high (V_{cc}). If all rows are grounded and a key is pressed, provides the path to ground one of the column will have 0 since the key pressed provides the path to ground. It is a function of the micro-controller to scan the keyboard continuously to detect and identify the key pressed.

Method 1

This method is used to know, whether any key is pressed or not.

Step(i): Make all the column lines low by sending low signal.

Step(ii): Check whether status of all the return lines is high (or) not

(3)

If it is high then \rightarrow no key is pressed

If it is low then \rightarrow key is pressed.

Method 2

To identify, which key is pressed

Step(i): Make any one column line zero.

Step(ii): The zero from any return line indicates that the key is pressed from the corresponding row and selected column.

If the status of all return lines is high, then it

indicates that the key is pressed from that column.

Step(iii): If all the return lines in step (ii) are high then repeat step (i) and continue with step (ii) again.

Interfacing of 4x4 matrix keyboard with 8051

The interfacing of matrix keyboard using 4x4 matrix is shown in figure.

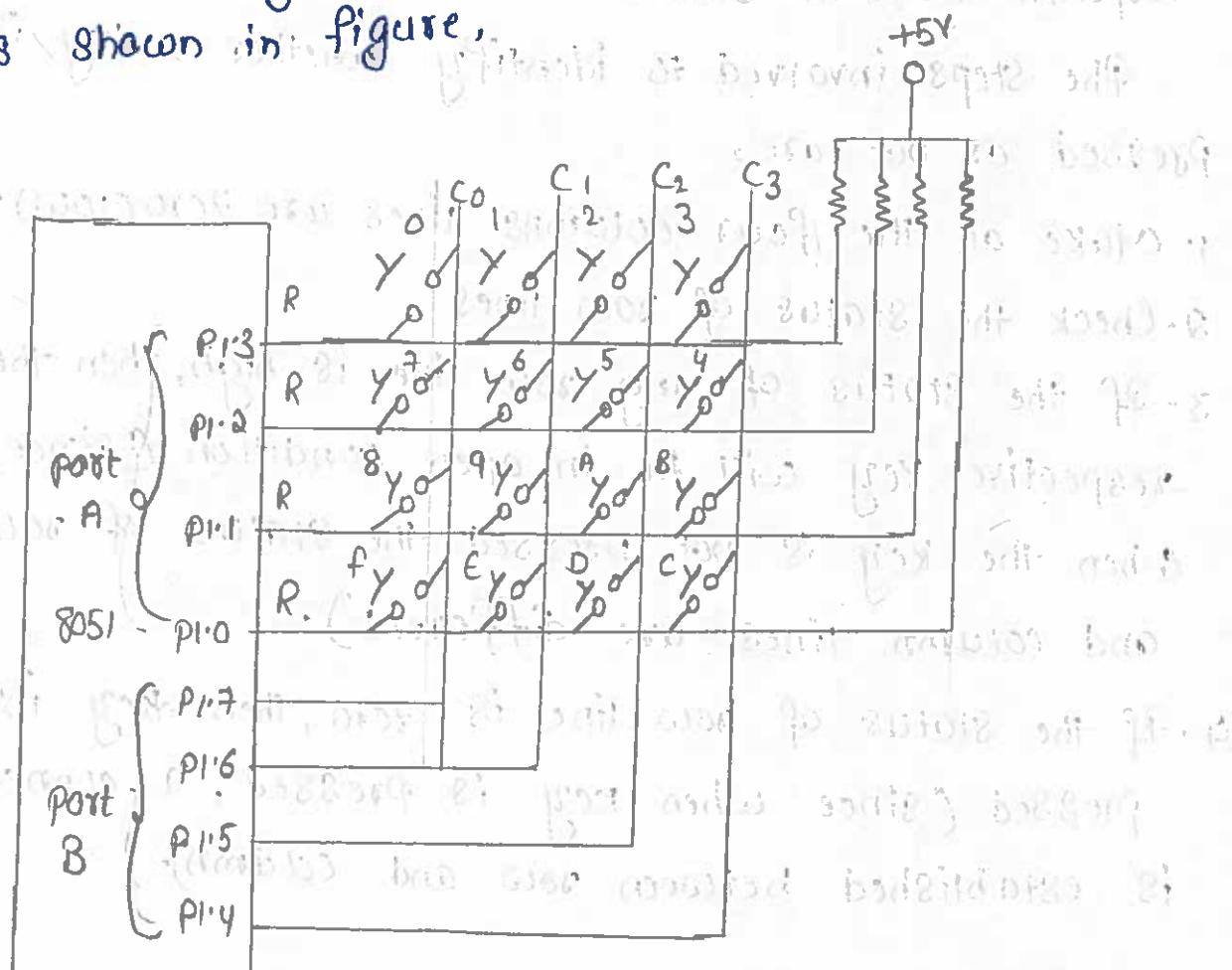


Fig: Interfacing of 4x4 matrix keyboard.

In the interfacing of 4x4 matrix keyboard with the two ports (i.e., port A or port B) of 8051, port A is referred as input port and port B as the output port. In 4x4 matrix keyboard, there are 16 keys, which are arranged along 4 rows and 4 columns.

The four horizontal lines (Rows - R₀, R₁, R₂, R₃) are connected to input port A and the lines are called return lines. The four vertical lines (C₀, C₁, C₂, C₃) are connected to output port 'B' and the lines are called scan lines.

When a key is pressed (closed), a connection is established between row and column and they will be in the same state (i.e. both will be either low (or) high). When a key is opened, there is no connection between row and column and they remains in their respective default states.

The steps involved to identify whether a key is pressed or not are,

1. Make all the four column lines zero (low).
2. Check the status of row lines
3. If the status of any row line is high, then the respective key will be in open condition (since, when the key is not pressed, the status of row and column lines are different).
4. If the status of row line is zero, then key is pressed (since when key is pressed, a connection is established between row and column).

functions of pins of LCD

Liquid Crystal Display (LCD) has 14 pins performs specific task that are shown in table.

Pin	Symbol	Mode of operation	Description
1.	V _{SS}		Ground
2.	V _{CC}		+5V power supply
3.	V _{EE}		Power supply to control LCD contrast.
4.	R _S	Input	If R _S =0, then command code register is selected. This allows the user to send a command such as clear display etc. If R _S =1, then the data register is selected. This allows the user to send data to be displayed on LCD.
5.	R/W	Input	If R/W=0, then write operation is performed. If R/W=1, then read operation is performed.
6.	E	Input/Output	Enable pin is used by LCD to latch information presented to its data pins. When data is supplied to the data pins, a high to low pulse must be applied to this pin.

7-14

DB0 - DB7

Input / Output

It is an 8-bit data bus.
These are used for information to the LCD.
(or) read the contents of the LCD's internal registers.
To display letters and numbers, we send ASCII codes for the letters A-Z, a-z, and 0-9 to these pins.

(5)

The interfacing diagram shows the connection of each LCD pin with port pins of microcontroller.

The functions of various pins of LCD in interfacing are as follows:

1. V_{CC} is connected to power supply (5V)

2. V_{SS} is grounded

3. V_{EE} is used to control contrast

4. The data pins D₀-D₇ of LCD are connected to respective pins of port 1

5. The EN (Enable) pin acknowledges LCD that microcontroller is about to send data. Microcontroller first makes EN line high so that it acknowledges LCD and then defines R_S and RW to put data on bus.

When EN=0, it indicates that the data transmitted and LCD will display by respective data.

6. R_S (Register Select)

When R_S=0, this line will select the command register and treats the data as instruction or command to LCD.

When R_S=1, it selects the data register and data will be in text that is to be displayed on LCD.

7. RW (Read/Write)

When RW=0, the data will be written on LCD

When RW=1, the data will be read from LCD.

Interfacing of LCD to 8051 Microcontroller

Generally, LCD's are available in 16x2 (or) 20x2 size i.e; 2 lines of 16 (or) 20 characters in each line with controller. The LCD controller is responsible for communication with microcontroller unit. The LCD is interfaced with microcontroller so that the CPU can display the output on LCD. The interfacing of LCD with microcontroller is as shown in figure.

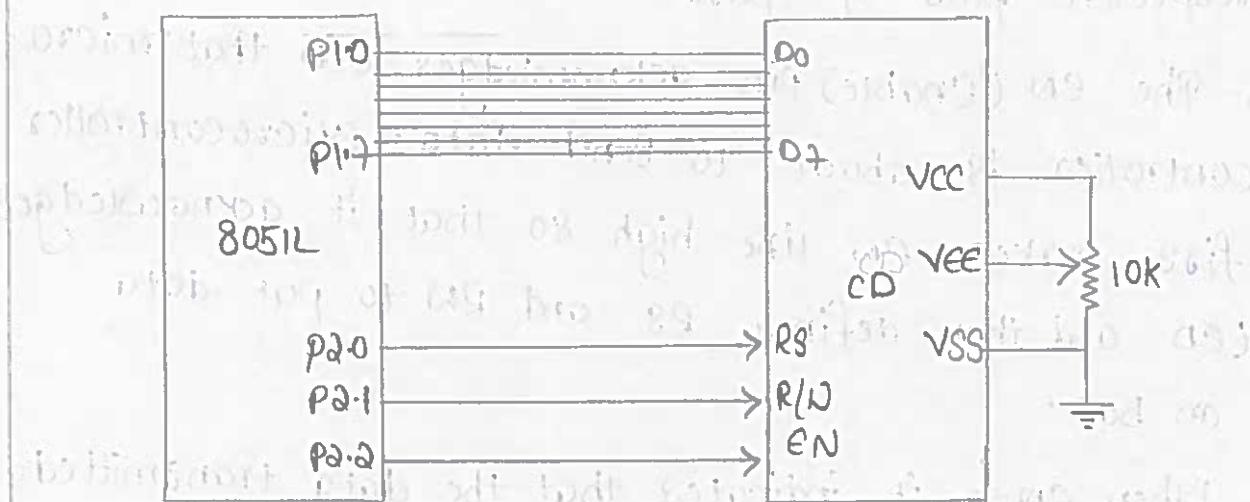


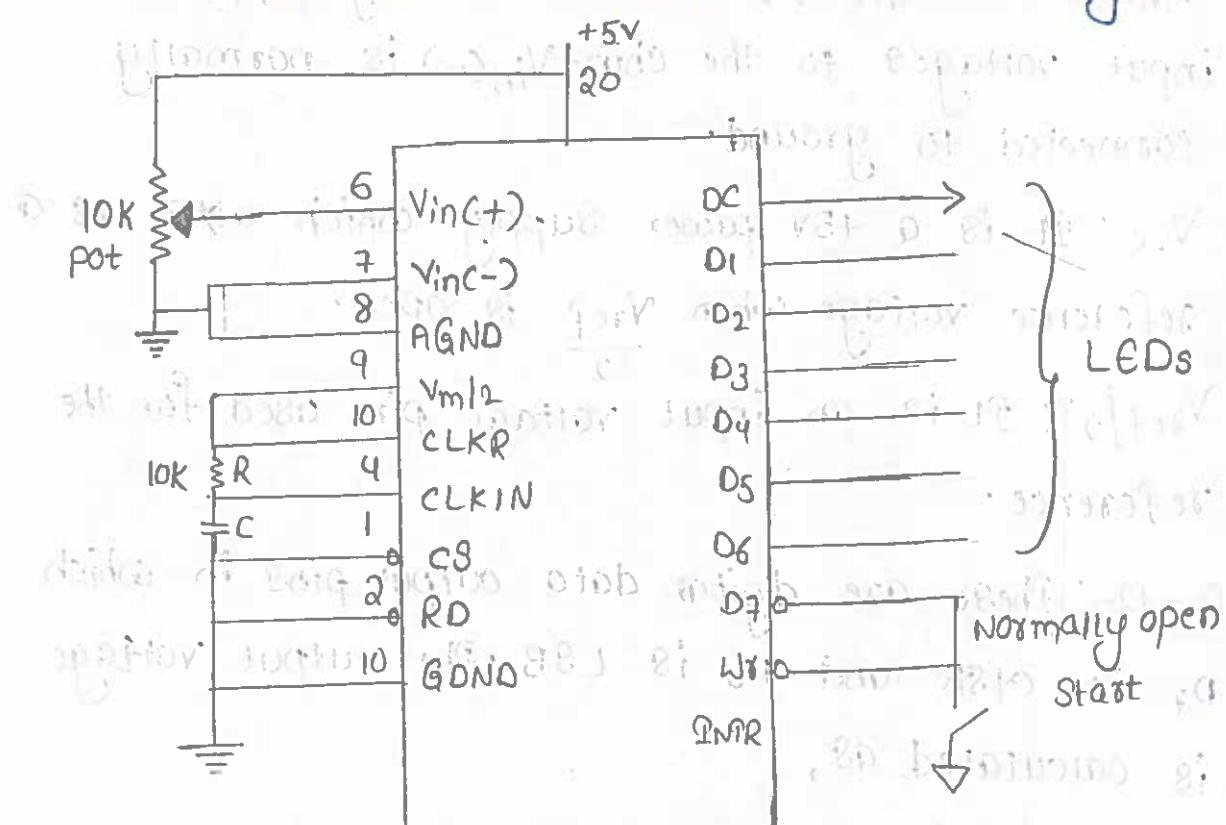
Figure : Interfacing of LCD with microcontroller.

Reasons for popularity of LCD's.

LCD's stands for liquid crystal Display. They are popular due to the following reasons, LCD's overcome the disadvantages in the conventional LED displays. Their cost is less compared to LED's. Unlike LEDs, LCD's have the ability to display numbers, characters and graphics.

Unlike in LEDs, LCD's include a refreshing controller that eliminates the refreshing task of CPU.

PIN NO.	NAME	FUNCTION
1.	VSS	Ground
2.	VCC	Supply voltage for logic
3.	VEE	Supply voltage for LCD contrast
4.	RS	Register Select
5.	R/W	Read / Write
6.	E	Chip Enable
7.	DO	Data Bit 0
8.	D1	Data Bit 1
9.	D2	Data Bit 2
10.	D3	Data Bit 3
11.	D4	Data Bit 4
12.	D5	Data Bit 5
13.	D6	Data Bit 6
14.	D7	Data Bit 7
15.	A(LED+)	Anode for LED Backlight
16.	K(LED-)	Cathode for LED Backlight



Figure

unit - 3, 11/44

The function of ADC 0804 pins is as follows.

CS: Chip select is an active low input signal which is used to activate ADC.

RD: Read is an active low input signal that is used to read converted digital data from ADC. It is also called as Output Enable (OE) pin as the output appears at high-to-low transition on RD when CS=0.

WR: It is an active low input which is used to inform ADC to start the conversion.

CLK IN and CLK R: These are the input pins to which capacitor ($C = 150 \text{ pF}$) and resistance ($R = 10k\Omega$) are connected to access internal clock generator.

Hence, the clock frequency is given by,

$$f = \frac{1}{1 \cdot IRC}$$

QNTR: It is an active low output pin which indicates the completion of conversion process.

$V_{in}(+)$ and $V_{in}(-)$: These are analog differential input voltages to the chip. $V_{in}(-)$ is normally connected to ground.

Vcc: It is a +5V power supply which acts as a reference voltage when $\underline{V_{ref}}$ is open.

$V_{ref}/2$: It is an input voltage pin used for the reference.

D₀-D₇: These are digital data output pins in which D₇ is MSB and D₀ is LSB. The output voltage is calculated as,

$$D_{out} = \frac{V_{in}}{\text{Step size}}$$

where,

D_{out} = Digital data output (in decimal)

V_{in} = Analog input voltage.

Step size = Resolution = $(2 \times V_{ref}/2)/256$ (smallest change).

-AGND and DGND: Analog ground and digital ground are input pins which are connected to ground of analog V_{in} and V_{ccP} respectively.

Hardware interfacing of ADC chip.

The typical hardware interfacing of ADC 08XX with microprocessor / microcontroller system is as shown in figure,

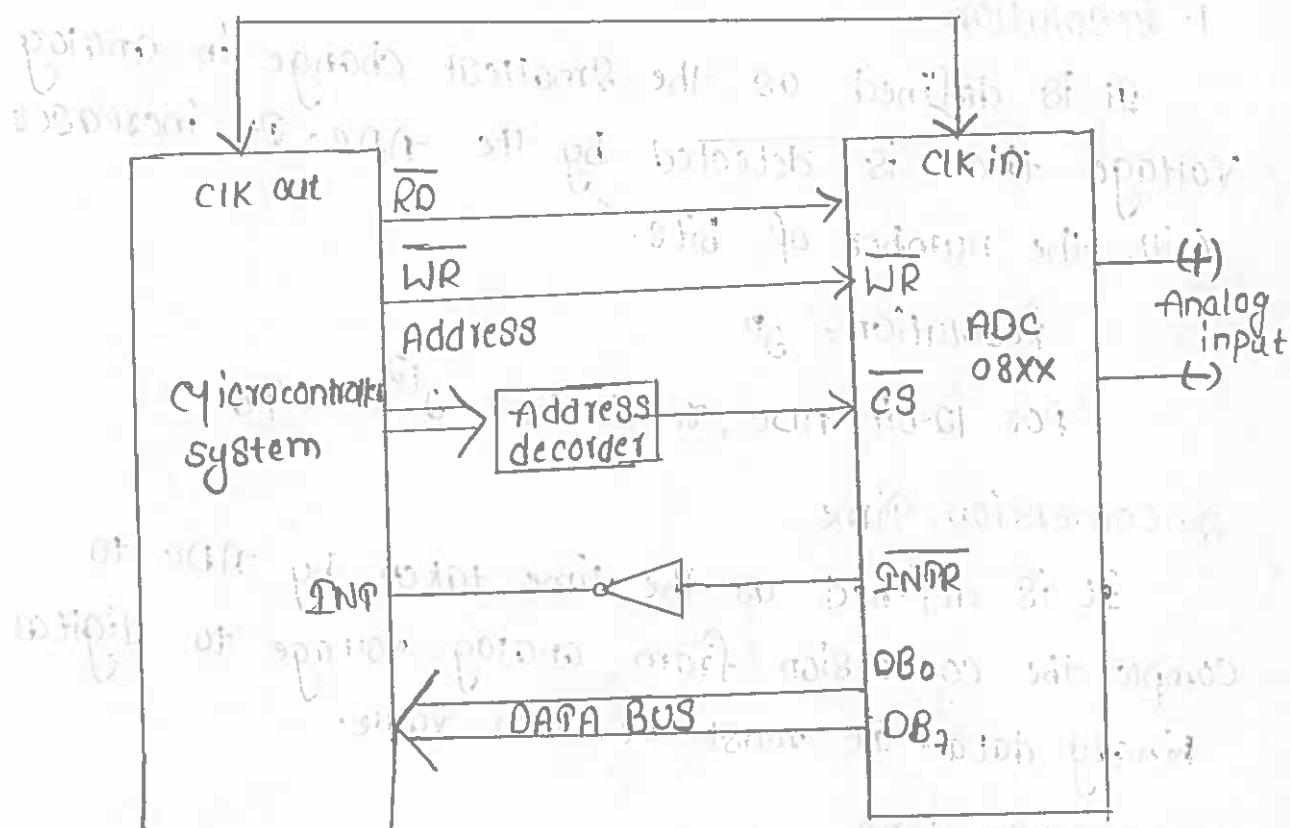


Figure: Hardware interfacing of ADC chip.

A clock input is applied externally, i.e. from microcontroller to ADC. ADC is in reset state when $\overline{CS} = 0$ and $\overline{WR} = 0$. \overline{CS} is made low by applying address from address decoder of microcontroller. When \overline{INTR} is high, conversion process begins. When \overline{INTR} is low, \overline{INT} pin of microcontroller becomes high which starts to read digital output through DB_0 to DB_7 data signals. During this process \overline{CS} and \overline{RD} signals are made low.

Choice of Selecting ADC chip.

The selection of ADC chip is based on its performance criteria. i.e; from the following parameters. They are,

1. Resolution

It is defined as the smallest change in analog voltage that is detected by the ADC. It increases with the number of bits.

$$\text{Resolution} = 2^n$$

$$\text{for 12-bit ADC, resolution} = 2^{12} = 4096$$

2. Conversion Time

It is defined as the time taken by ADC to complete the conversion from analog voltage to digital binary data. It must be low value.

3. Linearity Error

It is defined as the deviation from straight line drawn through zero and full scale. The accuracy in linearity must be high.

4. Sampling Rate Speed

It is defined as the number of times per second the analog signal is converted into digital signal. It must be twice the highest frequency in analog signal.

Clock source for ADC0804

The conversion speed of ADC depends on the speed of the clock signal.

The available clock sources for ADC0804 are,

1. self clocking
2. clock from crystal of 8051 microcontroller.

1. ADC 0804 with self-clocking

Figure (1) shows the 8051 connection to ADC0804 with self clocking.

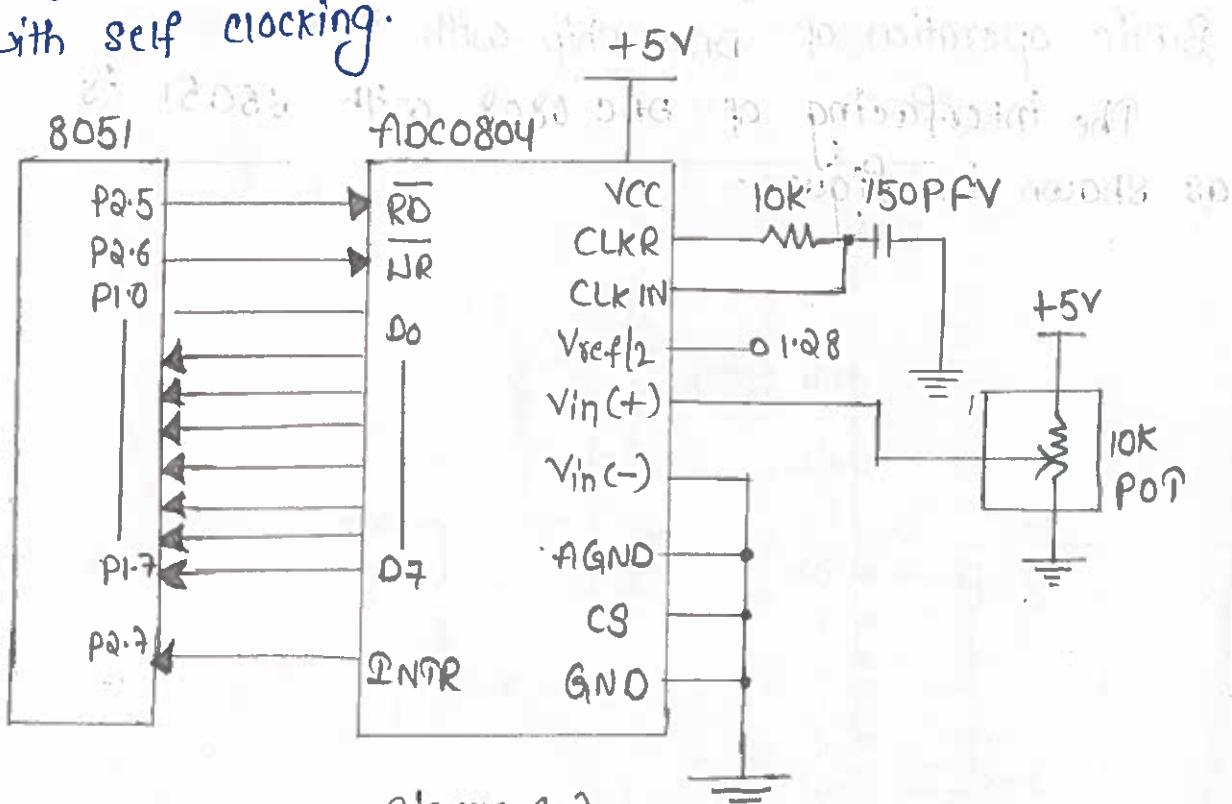
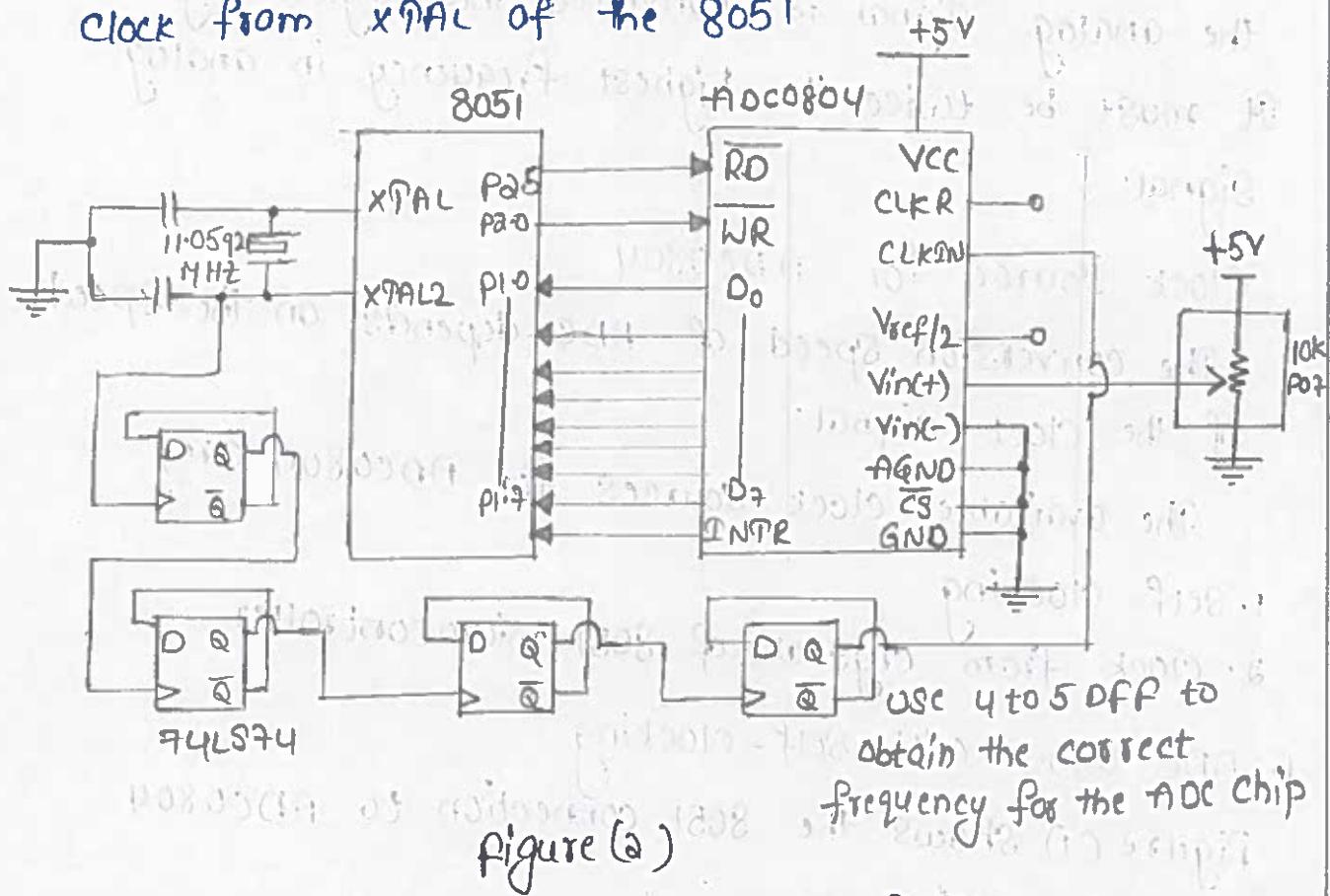


Figure (1)

Here the clock input (CLK IN) is taken from the CLK R, thus the ADC has internal clock generator, where both CLK IN and CLK R are connected to capacitor and resistor network.

Q. ADC 0804 with clock from Crystal.

fig a shows the connection of ADC 0804 with clock from X7PAL of the 8051



Basic operation of DAC chip with 8051.

The interfacing of DAC 0808 with 8051 is as shown in figure.

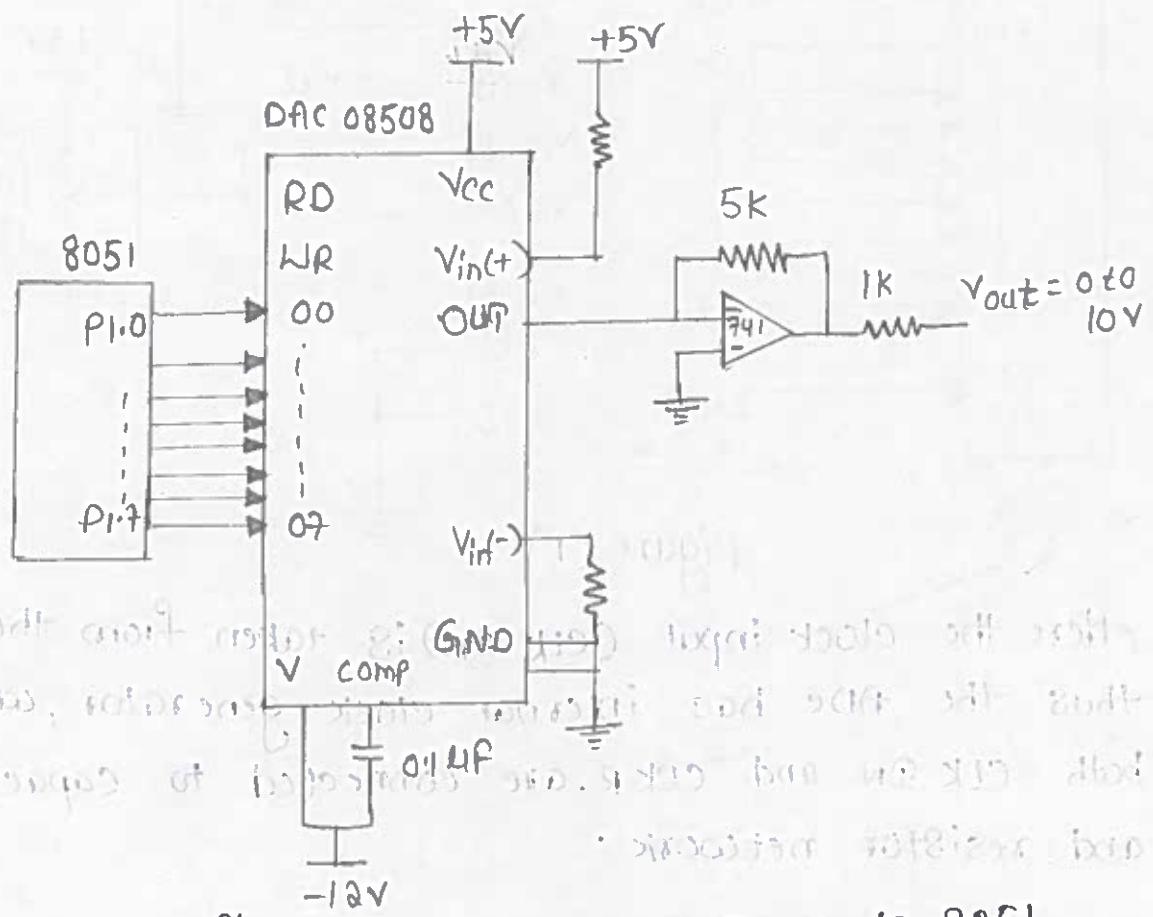


fig: DAC 0808 Interfaced to 8051

(9)

The port P1 of 8051 is interfaced with the digital inputs of DAC. The output of the DA is current which is converted into voltage using an op-amp.

This analog output current depends on I_{ref} following through V_{ref} and status of $D_0 - D_7$ bits. Hence, the output voltage is obtained as,

$$I_{out} = I_{ref} \left[\frac{D_7}{2} + \frac{D_6}{4} + \frac{D_5}{8} + \frac{D_4}{16} + \frac{D_3}{32} + \frac{D_2}{64} + \frac{D_1}{128} + \frac{D_0}{256} \right]$$

Where,

$$D_7 = L8B$$

The analog output voltage can be calculated as,

$$\text{Resolution of DAC} \times (D_7 - D_0)_{BCD}$$

External data memory (RAM) interface

Fig Q.71 shows the circuit diagram for connecting external data memory. The multiplexed address/data bus provided by port 0 is demultiplexed by external latch and ALE signal. Port 2 gives the higher order address bus. The RD and WR signals from 8051 selects the memory read and memory write operation respectively.

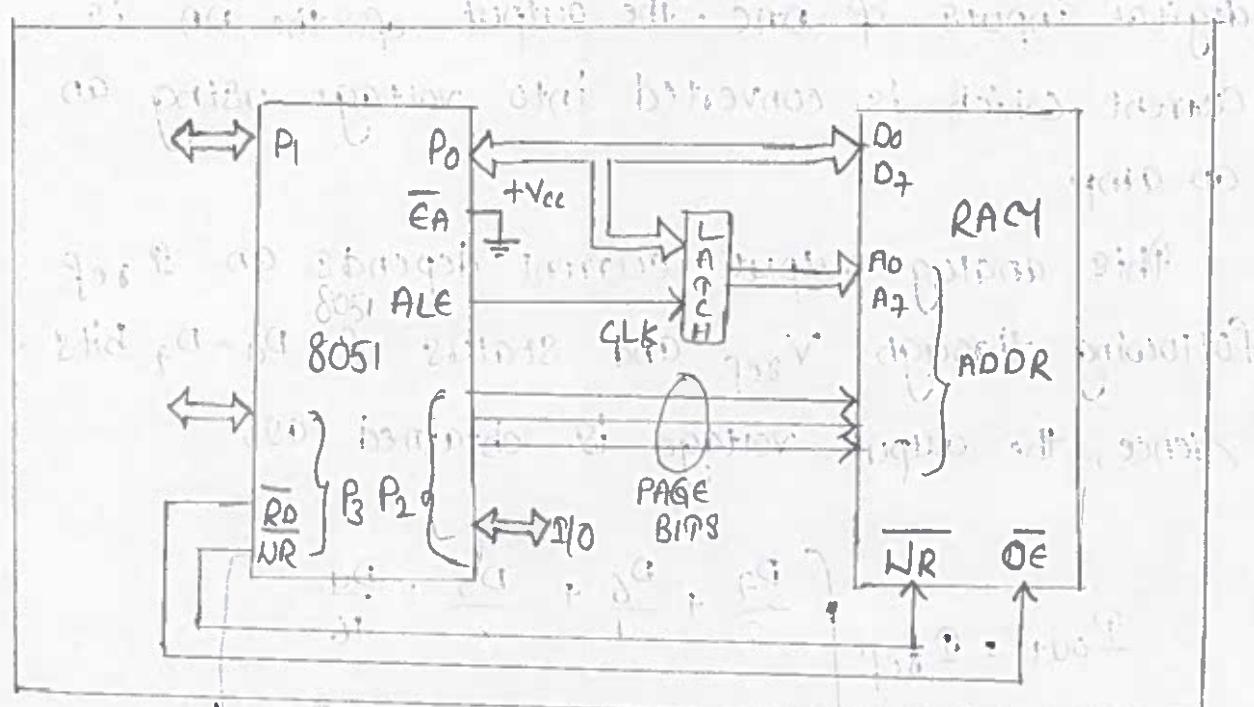


fig Q.7.1 Accessing External data memory.

Interfacing of 8051 with external ROM.

The interfacing of 8051 with External ROM is shown in figure.

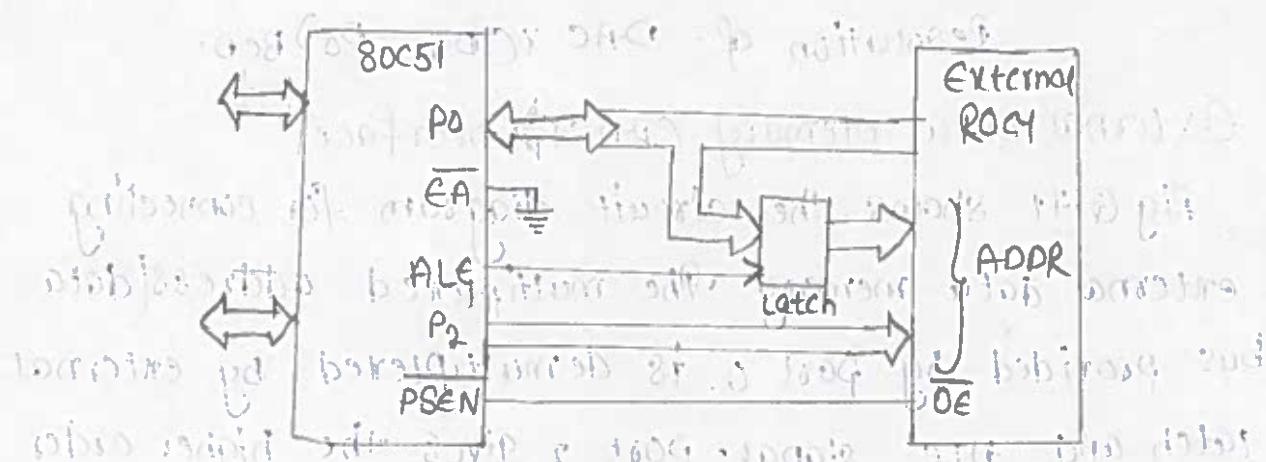


figure.

In the figure, the EA pin is connected with Vcc in order to indicate that the program code is stored in on-chip ROM. When EA is connected to ground, the program code is stored in external ROM. Port 0 & port 2 provides the 16 bit address to access external memory. P0 provides lower 8-bit address to A0-A7,

while P_2 provides upper 8-bit address to $A_{8-A_{15}}^{10}$
 P_0 also provides 8-bit data bus D_0-D_7 ($P_{00}-P_{0\cdot7}$)
When $ALE = 0$, 8051 uses P_0 for data path otherwise
i.e., $ALE \neq 1$, it uses P_0 for address path. 74LS373
latch is used to (access external) extract address
(or data) address, demultiplexing for P_0 pins. Program
store enable ($PSEN$) signal is used to access external
ROM containing program code by connecting it to
 OE pin.

On-chip ROM contains system boot code, while
external or off-chip ROM contains user's program.
The external ROM is considered only when the on-chip
program reaches the end of the on-chip ROM.

eight or ninth kid & upper evidence of sides
(6-09-009) 84-88 and this kid, evidence seen of
milkings from this with the 1800-1808 & 1814-1816
80-82-83-84-85-86-87-88-89-90-91-92-93
consists mostly (active season) of 1800 & 1801
1802-1803 of old *Calomyscus* scrubbo (otes 12)
various stages of hair in large cases which make
of the following of the more common kinds
which were found except evidence from girls and
young children most of the time in
which will make plain differences of these various
1804 girls-1805 left to be all others except

1805-1806-1807-1808-1809-1810-1811-1812-1813

1814-1815-1816-1817-1818-1819-1820-1821-1822-1823

1824-1825-1826-1827-1828-1829-1830-1831-1832-1833

1834-1835-1836-1837-1838-1839-1840-1841-1842-1843

1844-1845-1846-1847-1848-1849-1850-1851-1852-1853

1854-1855-1856-1857-1858-1859-1860-1861-1862-1863

1864-1865-1866-1867-1868-1869-1870-1871-1872-1873

1874-1875-1876-1877-1878-1879-1880-1881-1882-1883

1884-1885-1886-1887-1888-1889-1890-1891-1892-1893

1894-1895-1896-1897-1898-1899-1900-1901-1902-1903

1904-1905-1906-1907-1908-1909-1910-1911-1912-1913

1914-1915-1916-1917-1918-1919-1920-1921-1922-1923

1924-1925-1926-1927-1928-1929-1930-1931-1932-1933

1934-1935-1936-1937-1938-1939-1940-1941-1942-1943

1944-1945-1946-1947-1948-1949-1950-1951-1952-1953

1954-1955-1956-1957-1958-1959-1960-1961-1962-1963

1964-1965-1966-1967-1968-1969-1970-1971-1972-1973

1974-1975-1976-1977-1978-1979-1980-1981-1982-1983

1984-1985-1986-1987-1988-1989-1990-1991-1992-1993

1994-1995-1996-1997-1998-1999-2000-2001-2002-2003

2004-2005-2006-2007-2008-2009-2010-2011-2012-2013

2014-2015-2016-2017-2018-2019-2020-2021-2022-2023

2024-2025-2026-2027-2028-2029-2030-2031-2032-2033

2034-2035-2036-2037-2038-2039-2040-2041-2042-2043

2044-2045-2046-2047-2048-2049-2050-2051-2052-2053

2054-2055-2056-2057-2058-2059-2060-2061-2062-2063

2064-2065-2066-2067-2068-2069-2070-2071-2072-2073

2074-2075-2076-2077-2078-2079-2080-2081-2082-2083

3rd Unit Second half

(11)

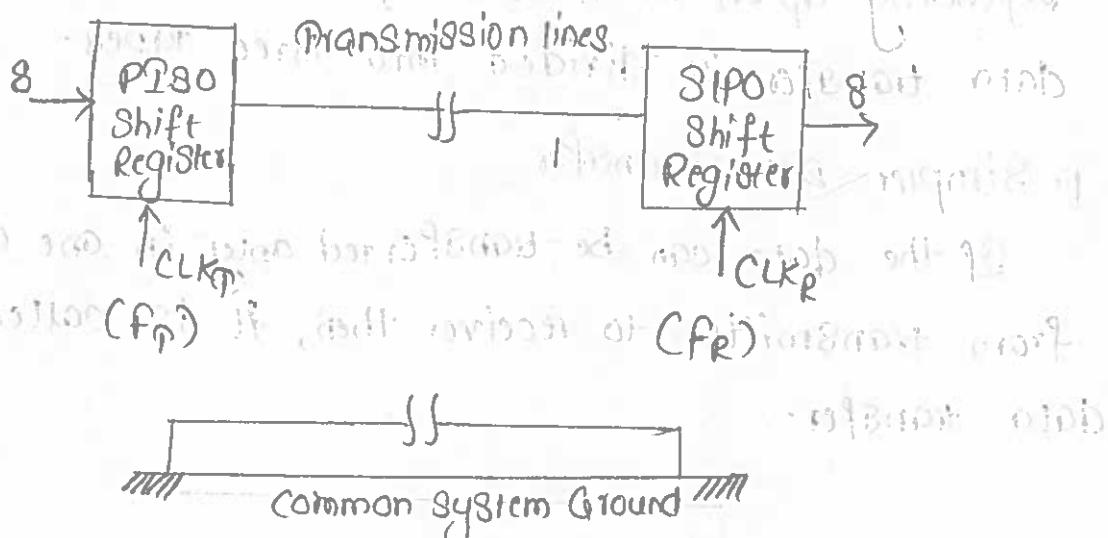
Types of Serial Data Transfer

There are two types of serial data transfer. They are,

1. Synchronous Serial Data Transmission

The synchronous serial data transmission mode is

shown in figure.



If operating clock frequency of transmitter and Receiver is exactly same ($f_P = f_R$) then it is called synchronous serial data transfer. If there is slight difference in frequency then there will be error in data transmission.

Example

Data transfer b/w processor and CRT terminal, Keyboard printer.

2. Asynchronous Serial Data Transmission.

In this type of data transfer, there is slight difference in the frequency of transmitter and receiver which can be adjusted by using stop bits after each character is transferred.

Example

Data Transfer between processor and monitor,
audio cassettes.

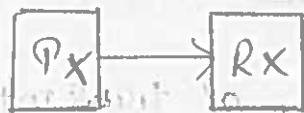
Classification of Serial Data Transfer

The data in serial mode is transferred from transmitter to receiver in bits. The rate of transmission in serial mode is BAUD i.e Bits per second.

Depending upon the direction of data transfer, serial data transfer is divided into three types.

1. Simplex Data Transfer

If the data can be transferred only in one direction from transmitter to receiver then, it is called simplex data transfer.



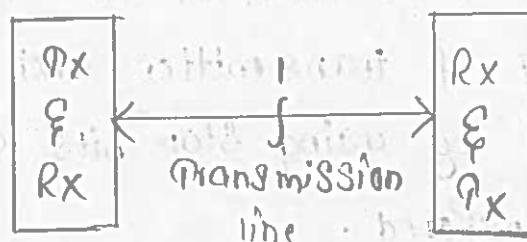
Example

Data transfer from CPU to printer

2. Half Duplex data transfer

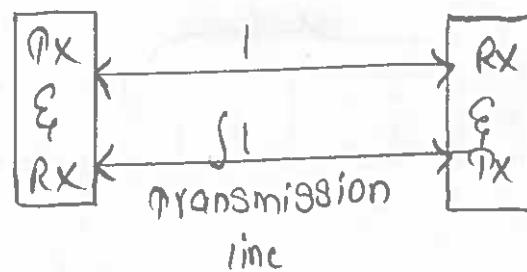
If the data can be transferred in both the directions but there is only one transmission line.

Then, such type of data transmission is called half duplex data transfer.



3. Full Duplex Data Transfer

If the data can be transferred in both direction i.e simultaneously, so there are two different transmission lines.



Example:

Telephone lines.

Synchronous and asynchronous formats and discuss the differences.

Synchronous Data Format

Synchronous communication uses the following data format:

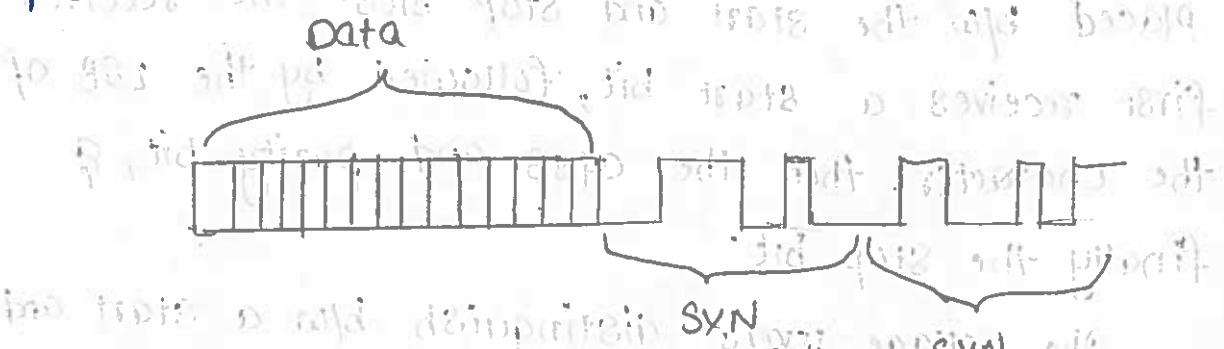


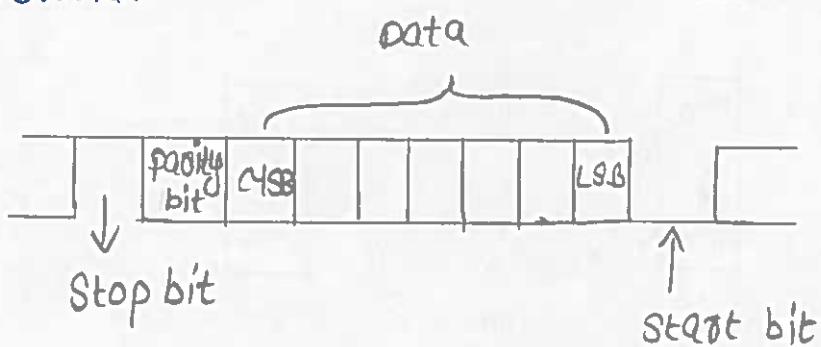
figure (1): Synchronous Data format.

Synchronous transmission begins with the transmission of the synchronization characters. The receiver is always equipped with a known as synchronization pattern. Thus, when the synchronization bit pattern is received by the receiver, the bit pattern is compared with synchronization pattern. If these two patterns are identical, the character on the

data line is read by the receiver.

Asynchronous Data Format

Asynchronous communication uses the following data format-



Fig(a) Asynchronous Data format.

The synchronization bits used in asynchronous communication are Start and Stop. The number of the Stop bits used depends on the communication scheme. For example, 1, 1½ or 2 stop bits can be used as per requirement. The character (data) is placed b/w the start and stop bits. The receiver first receives a start bit, followed by the LSB of the character, then the MSB and parity bit, & finally the stop bit.

The voltage levels distinguish b/w a start and stop bit and, the duration of each bit is called a bit time.

Difference between synchronous and asynchronous

13

Synchronous serial Transmission

- 1) In this transmission system, if there is a slight difference in frequency then there will be errors in data transmission.

- 2) This type of data transfer is used when the processor and input/output device for data transfer is not checked.

- 3) Here, the readiness of input/output device for data transfer is not checked.

- 4) The input/output device is always ready for data transfer when all IN or OUT instruction is issued by the processor as they have same speed.

Example

Data transfer between processor and CRT terminal, keyboard, printer, etc.

Asynchronous serial Transmission.

- 1) In this type of data transfer, if there is a slight difference in the frequency of transmitter and receiver it can be adjusted by using stop bits after each character transferred.

- 2) This type of data transfer is used when the processor and input/output devices do not match in speed.

- 3) Here, the readiness of input/output device for data transfer is checked by the CPU before transferring data.

- 4) The processor informs the input/output device to set ready for data transfer and then regularly examines the status of input/output device till the input/output device get ready for data transmission.

Example:

Data transfer between processor and modem, audio cassettes, etc.

sequence of operation for communicating with an I²C slave device.

Inter Integrated circuit (I²C) :-

The on board communication interface, I²C, is a synchronous bi-directional half duplex two wire serial interface bus. The connection between an embedded system and external devices is made easy using the I²C bus. The two major bus lines of I²C are, the SCL bus line generates synchronization clock pulses in the circuit

2) serial data(SDA) :-

The SDA bus line helps in the transmission of serial data among the various available devices.

I²C is a shared bus interface which itself connects various number of I²C device known as 'Master' device or 'Slave' device.

• bus protocol

figure represents the connection of master and slave devices on I^2C bus. The sequence of operations carried out in order to communicate with an I^2C slave device are

Step 1

The SCL line of the I^2C bus is maintained high by the master device.

Step 2

If the SCL is at "t_{high}" the SDA is automatically pulled low by the master (piece) device. This device condition indicates the 'start' of data transfer.

Step 3:

The SDA line carries the address of the slave device given by the master so as to communicate with the other devices in the system. The slave device produces clock pulses at the SCL line in order to synchronize the reception of bits.

Step 4:

The read and write operations are carried out by sending the respective Read and write bits by the master device i.e. for Read operation, Bit value = 1 and for write operation, Bit value = 0.

Step 5:

The master device waits for the acknowledgement from the requested slave device.

Step 6

The slave device after receiving address from master device compares it with its own address. If a match occurs between the two addresses, the slave device responds to it by sending an acknowledgement bit (Bit value=1) to the master device through the SDA line.

Step 7

After accepting the acknowledgement bit from the slave device, transfer of data between master and slave takes place depending on the type of operation required.

- (i) For 'write to device' operation, the SDA line carries the 8 bit data to the slave device.
- (ii) For 'read from device' operations, the data is sent from slave to master device through the SDA line.

Step 8

The master device after finishing the read operation sends an acknowledgement bit and waits for the same from slave device after completing the transfer of data for write operation.

Step 9

SDA line is pulled to "HIGH" by the master device in order to stop the transfer of data when SCL is set at logic 'HIGH'.

Serial peripheral Interface(SPI) and its protocol

Serial peripheral Interface(SPI) is the 15 simplest synchronous communication protocol in general use which was developed by Motorola to support communications between a master host processor and one or multiple slave peripheral devices. SPI has the other name as "four wire" serial bus, hence the Serial Peripheral Interface(SPI) bus is a simple 4-wire serial communications interface used by many microprocessor and microcontroller chips to start communicating each other.

The SPI bus, usually operates at a full duplex mode and is a synchronous type data link setup with a Master/slave interface to support up to 1-megabaud or 10Mbps of speed. Both single-master and multi-master protocols are possible are possible in SPI, in which single master is most widely used and can be observed wide usage on printed circuit boards(PCBs).

[The SPI Bus was, usually operates at a full duplex mode and]

The SPI Bus was designed to transfer data between various IC chips, at very high speeds and due to which the bus lines are restricted in length, because their reactance increases too much and the bus becomes unusable. Due to its simplicity, SPI has been adopted by numerous manufacturers of serial EEPROMs, real-time clock modules, data converters, LCDs and other peripherals.

data and control lines of the SPI and the basic connection:

- An SPI protocol specifies 4 signal wires

System architecture and communication interface
Protocol
Hardware
Software
Implementation
Conclusion

i. Master Out Slave In (MOSI):

This signal is generated by the master, recipient is the slave.

2. Master In Slave Out (MISO):

MISO signal is generated by the slave where master will receive.

3. Serial clock (SCLK or SCK):

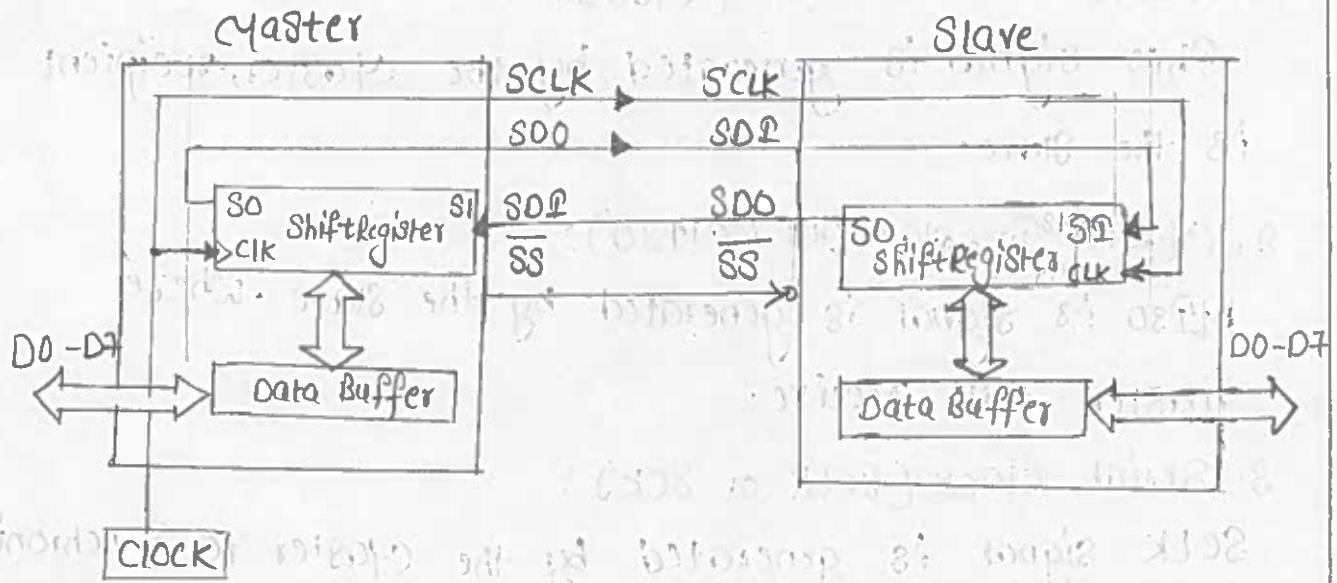
SCLK signal is generated by the master to synchronize data transfer b/w the master and the slave.

4. Slave Select (SS) from Master to Chip Select (CS) of Slave:

This is an active low signal, where SS signal is generated by master to select individual slave peripheral devices.

Here, MOSI and MISO are grouped as data lines and SS and SCLK are grouped as control lines. In SPI-bus communication there can be one master with multiple slaves. In single master protocol, usually one SPI device acts as the SPI master and controls the data flow by generating the clock signal (SCLK) and activating the slave it wants to communicate with slave-select signal (SS), then receives and/or transmits data via the two data lines.

A master, usually the host micro controller, always provides clock signal to all devices on a bus whether it is selected or not. The concept of SPI is shown in figure (1) for the minimal system of two devices.



fig(c) SPI Bus with single master and slave
for point to point communication.

Every time the master will initiate the communication by configuring the clock, using a frequency less than or equal to the maximum frequency that the slave device can support. The master then selects the required slave for

A full duplex data transmission can occur during each clock cycle. That means the master ends a bit on the MOSI line the slave reads it from that same line. (and the slave) Data transfer is organized by using shift register with some given word size such as 8-bits in both master and slave. They are connected in a ring. While master shifts register value out through MOSI line, the slave shifts data to its shift register.

Data are usually shifted out with the MSB first, while shifting a new LSB into the same register. After that register has been shifted out

17

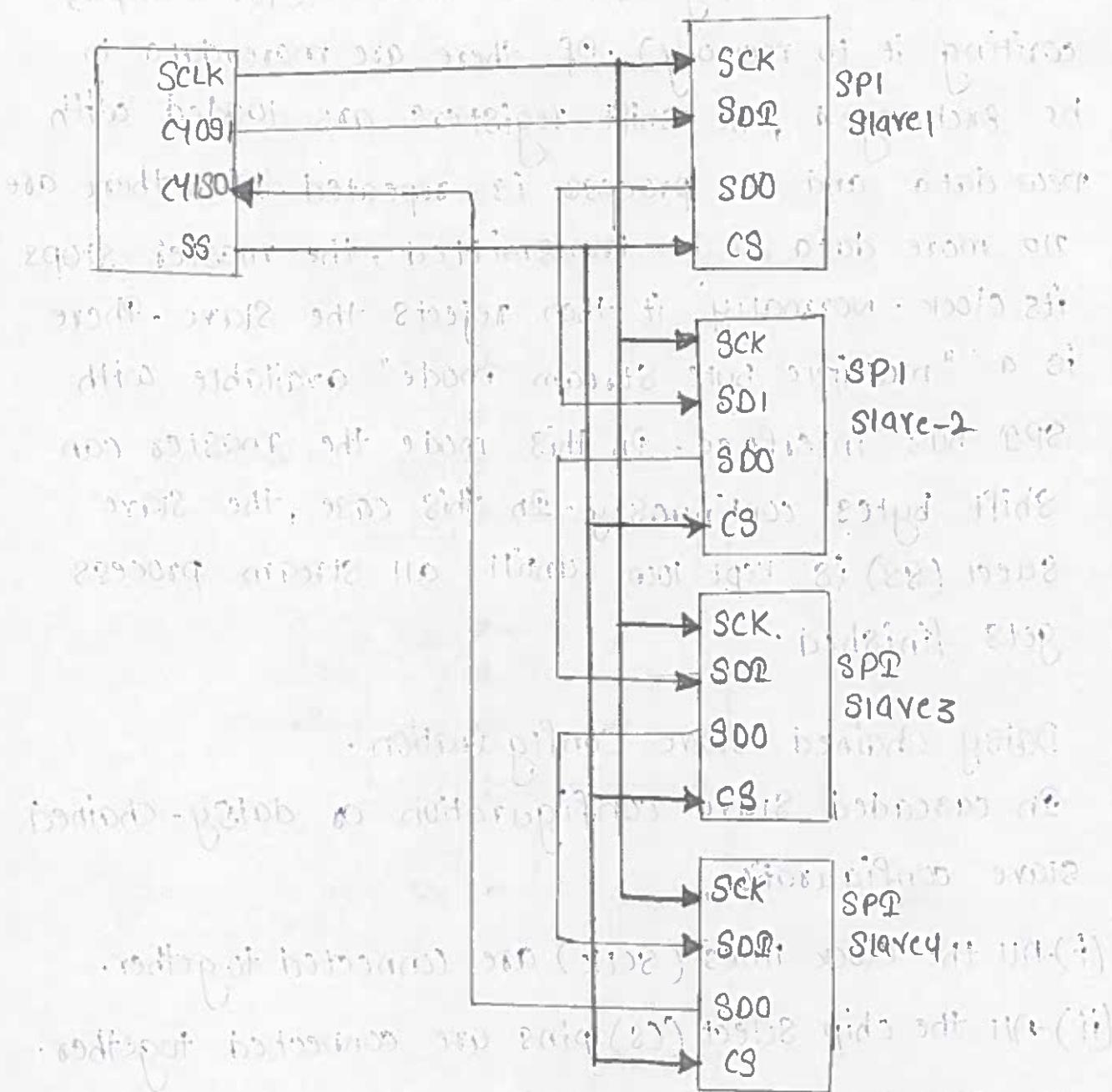
the master and slave have exchanged their register values. Then each device takes that value and does the necessary operation with it (for example, writing it to memory). If there are more data to be exchanged, the shift registers are loaded with new data and the process is repeated. When there are no more data to be transmitted, the master stops its clock. Normally, it then rejects the slave. There is a "multiple byte stream mode" available with SPI bus interface. In this mode the master can shift bytes continuously. In this case, the slave select (SS) is kept low until all stream process gets finished.

Daisy chained slave configuration.

In cascaded slave configuration or daisy-chained slave configuration,

- (i) All the clock lines (SCLK) are connected together.
- (ii) All the chip select (CS) pins are connected together.
The data will flow out of the microcontroller, through each peripheral in turn, and back to the microcontroller. The data output of the preceding slave-device is tied to the data input of the next, thus forming a wider shift register. The SPI bus in Daisy-chained slave configuration is shown in figure (2).

designs for bidirectional serial ports has become well known over time with the development of various serial protocols.



Figure(2): 8PΩ Bus in Daisy Chained Slave configuration.

The SPI bus independent slave configuration is shown in figure(3) 18

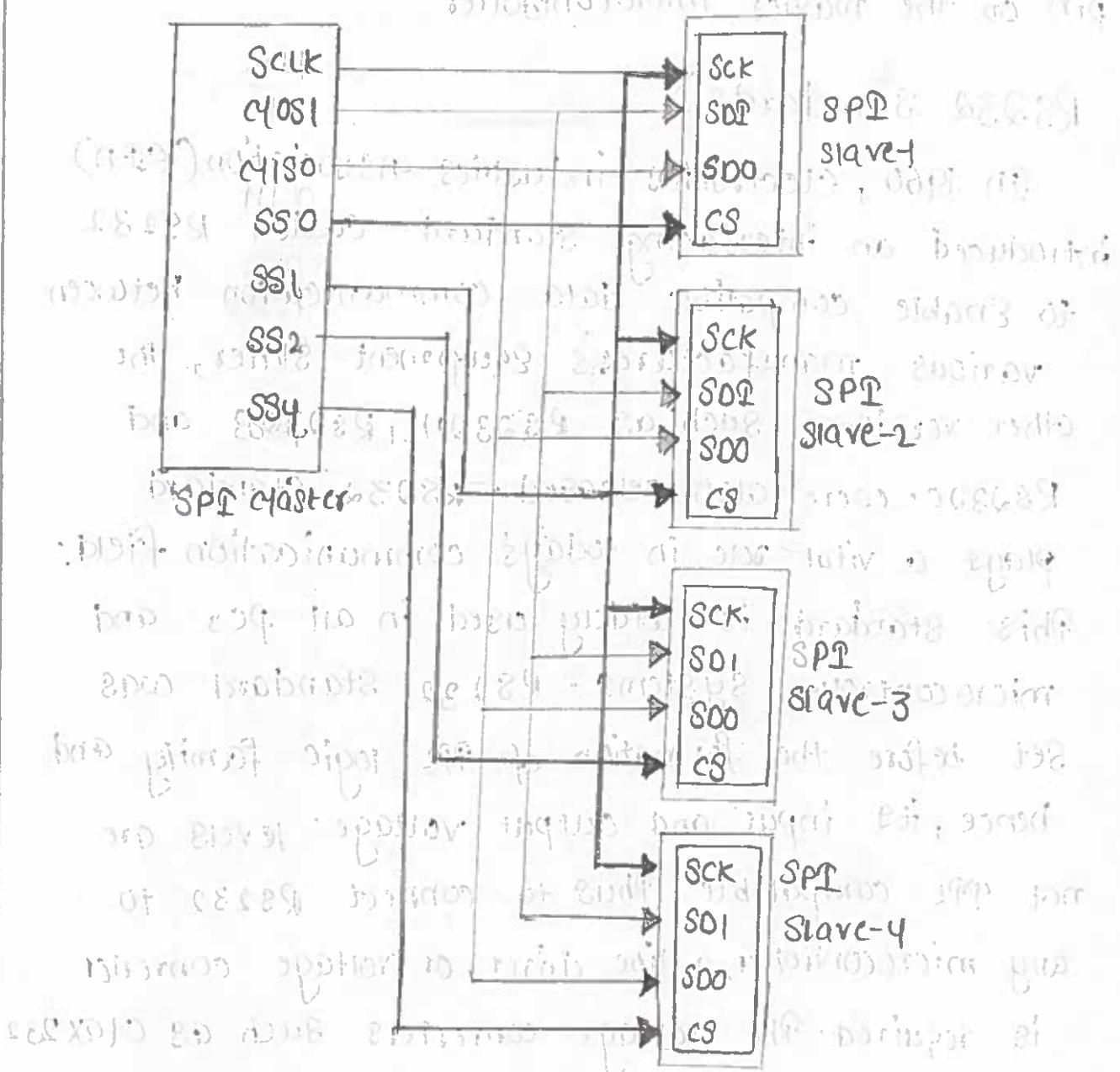


Figure (3) The SPI Bus In Independent slave configuration.

Independent slave configuration

This SPI-bus configuration has one SPI-master

and multiple slaves or peripherals. and the connections are made as follows,

- (i) All the clock lines (SCLK) are connected together
- (ii) All the MISO data lines are connected together

- iii.) All the GND lines are connected together.
- iv.) The chip select (CS) pin from each peripheral must be connected to a separate slave select (SS) pin on the master microcontroller.

RS232 Standards

In 1960, electronics industries association (EIA) introduced an interfacing standard called RS232 to enable compatible data communication between various manufacturers equipment. Later, the other versions such as RS232A, RS232B and RS232C were also released. RS232 standard plays a vital role in today's communication field. This standard is widely used in all PCs and microcontroller systems. RS232 standard was set before the formation of TTL logic family and hence, its input and output voltage levels are not TTL compatible. Thus to connect RS232 to any microcontroller, a line driver or voltage converter is required. The voltage converters such as MAX232 and MAX233 are widely used and the function of them is to convert TTL logic levels to the RS232 voltage levels and vice versa. Table shows the voltage levels of RS232 standard.

RS232 levels: Logic level voltages have

-3 to -25V

+3 to +25V

Output bits (0) and (1) are swapped

(Q)

However, in RS232 standard, the voltage level -3 to +3 is undefined.

RS232 pins of DB25 and DB9 connectors

DB-25 connector.

DB-25 in an RS232 connector having 25 pins.

It is available in two forms i.e DB-25P and DB-25S.

DB-25P is a plug type connector (male connector)

whereas, DB-25S is a socket type connector

(female connector) figure (1) shows the DB-25 connector

The table shows the specifications of three standards.

Parameter	RS-232C	RS-422	RS-485
1. Signaling technique	single-ended (unbalanced)	differential (balanced)	differential (balanced)
2. Drivers and receivers on bus	1 Driver and 1 Receiver	1 Driver and 10 Receivers	32 Drivers and 32 Receivers
3. Maximum cable length	50 feet	4000 feet	4000feet
4. Original standard maximum date rate	20 kbps	10 Cbps down to 100 kbps	10 Cbps down to 100 kbps.
5. Maximum loaded driver output voltage levels	$\pm 5.0V$	$\pm 2.0V$	$\pm 1.5V$

6.	Driver load impedance	3 to 7K	100	54
7.	Receiver input impedance	3 to 7K	4K	12K

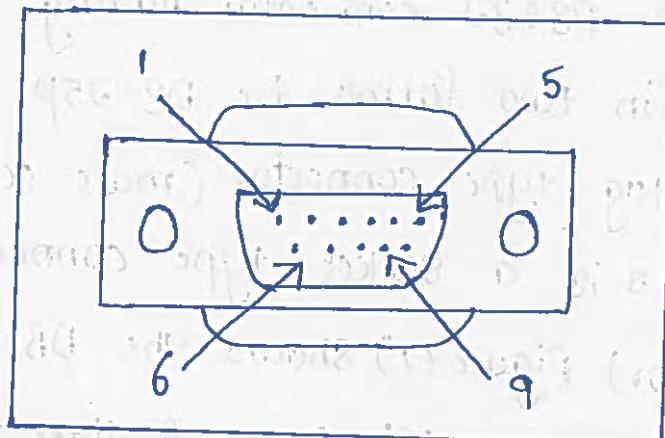


Figure (2): DB-9 connector

Pin	Description
1.	Data carrier detect (\overline{DCD})
2.	Received data (RXD)
3.	Transmitted data (TXD)
4.	Data terminal ready (DTR)
5.	signal ground (GND)
6.	Data set ready (\overline{DSR})
7.	Request to Send (\overline{RTS})
8.	clear to send (\overline{CTS})
9.	Ring indicator (\overline{RI})

UART structure and functionality

The UART is a module used to establish a serial interface in a full-duplex mode. The block diagram of UART is shown in the figure.

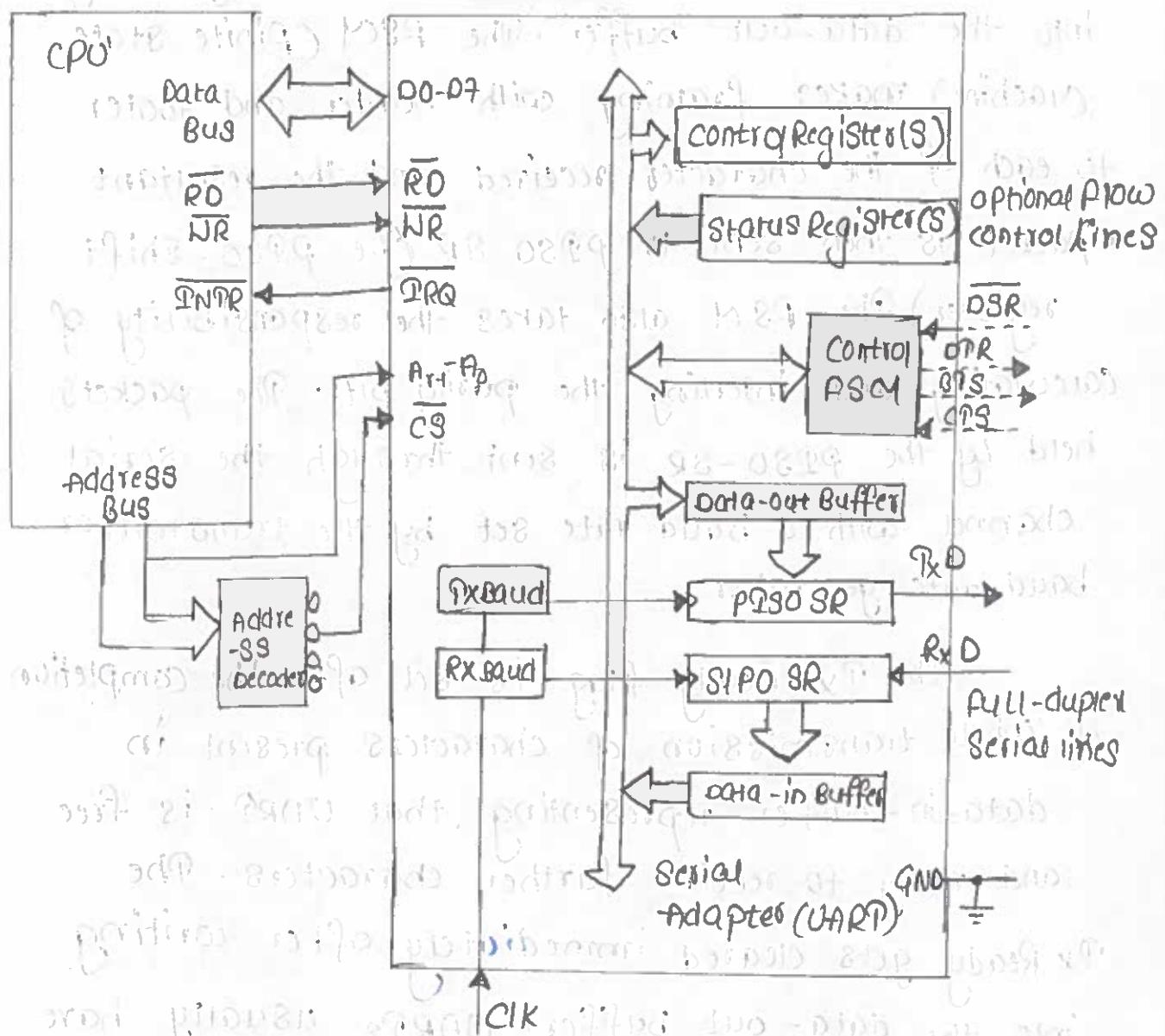


Figure: Block diagram of UART

The UART has several components microprocessor among which two shift registers play key role in the data transfer one is PISO and the other is SIPO. that are in transmitter and receiver respectively. The CPU initiates the character transmission by writing into the data-out buffer. The FSCM (Finite state machine) makes framing with header and footer to each of the characters received and the resultant packet is then sent to PISO SR (i.e PISO-Shift register). The FSCM also takes the responsibility of calculating and inserting the parity bit. The packets held by the PISO-SR is sent through the serial channel with a baud rate set by the transmitter baud rate generator.

The Tx Ready flag is set after the completion of whole transmission of characters present in data-in-buffer representing that UART is free and ready to receive further characters. The Tx Ready gets cleared immediately after writing into the data-out buffer. UARTs usually have a single character buffer, but multiple characters can be supported by more advanced UARTs with greater Kilobyte capacities.

The SIPO is reset by sending a valid start bit before being received and it is loaded with the packets that are transmitted. The control FSCM

92

does removing of start bit and stop bit from the bits of SIPO SR and clears the parity bits after completing parity check operation. Finally, the received character is placed in data-in buffer. The Rx Ready will get cleared automatically after reading the data-in-buffer and the parity flag is set if there is any parity/framing error. In the reception process there is a possibility/chance to an error to occur and is called "reception error" or "Overflow error" which is due to fast filling of characters into data-in buffer then the CPU results in loss of some data or characters received by the CPU. This problem can be avoided by adopting data-in buffers that has multi-character capability where the CPU needs to remove the characters immediately after receiving them to eliminate the loss of data.

The transmitting and receiving clocks are used to provide clock for PT80 SR and SIPO SR respectively, thereby originating data (refers) rates for transmission and reception. The status register indicates the status of the transmitter, receiver and Errors if any and the parameters of UART are composed by the control register. The function of control registers and status registers is based on the capabilities of UART. The following are the module parameters which can be set by the internal control register,

- 1.) Length of the character
- 2.) Type of parity
- 3.) parity Enable (Disable)
- 4.) Length of Stop bit
- 5.) Baud rate value.

Bulk Transfer

This type of transfer is used when the rate of transfer can vary. The bulk transfers have a low priority bus bandwidth allocation. That is, if all the above three transfer types share 100% of the bus bandwidth, then the bulk transfer will be postponed and will continue only when the load is decreased. This transfer supports error detection and recovery.

USB protocols and USB packed formats

a.) USB protocols

USB supports data transmission in the form of packets that comprise of one or more bytes of information. Each information on the USB is categorized into two types, namely control and data. The task of control packet is to address a device for initiating data transfer, provide acknowledgement after receiving of correct information, indicates an error if it arises. On the other hand, the data

packet performs the task of carrying information which has been delivered to a device.

b.) USB packet formats:

A USB packet has the following format:

8-bits	8-bits	Information about packet	5/16 bits	3-bits
SYNC	PID	Information about packet	CRC	EOP

A USB packet contains five fields. They are

i.) SYNC or synchronization field: It is the first field of each USB packet. It is an 8-bit field which is used by the receivers to synchronize the data rate of the incoming packets. The value of this field is 00000001.

PID or packet ID field: - It follows the SYNC field. It is an 8-bit field whose first four bits correspond to the type field. A 4-bit type field is used to determine the check field. A 4-bit type field is used to determine the type of packet and a 4-bit check field is used to protect type field. It is the complement of type field.

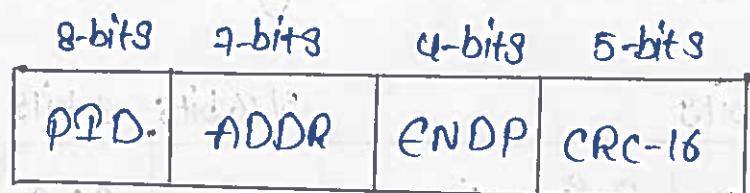
The three different types of USB packets are,

i.) Token

ii.) Data

iii.) Handshake.

Token packet:- The packets which are used to represent control information are also referred to as token packets. The format of token packet is as follows.

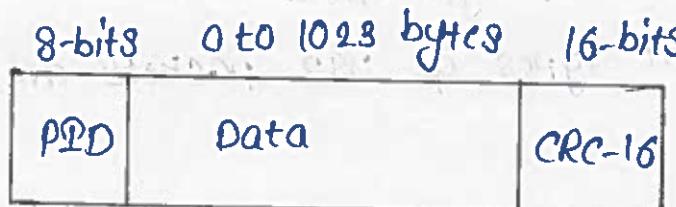


The size of a token packet is 24-bits and it contains following four fields.

- * An 8-bit PID field - It indicates the type of packet being sent.
- * A 7-bit address field - It holds address of initiating station.
- * A 4-bit end point field - It holds a number that represents the end point of transmitting device.
- * A 5-bit CRC-16 field - It is used for error detection based on address and endpoint fields.

ii) Data packet:

The packets which are used to carry data are called data packets. The format of data packet is as follows.



A data packet consists of three fields.

- * An 8-bit PDO field - it indicates the type of packet being sent.
- * A data field ranging from 0 to 1028 bytes.
- * A 16 bit CRC-16 field - It is used for error detection purposes. It is composed over the data field.

iii.) Handshake packet

The packets which are used to determine and return data transaction status are called handshake packets. The handshakes can be returned only by those types of transactions that support flow control.

Information about packet

This field holds the required data to be transferred cyclic Redundancy Check (CRC) Field

This field is used for error detection purposes. It can be of 5 or 16 bits of size depending on the type of packet being used.

End of packet (EOP) field

This field indicates the end of a packet. It is a 3-bit field and is encoded by hardware so that the same encoding is not included in the packet. Its first two bits are low for two bit periods and in the third bit period, it remains idle.

to say with certainty. I visited one place on the
island where there was no vegetation.

Visited two other spots where there was some
vegetation. At both of these places the plants were
mostly dead and the vegetation was sparse.

At the third place there was a great deal of
green vegetation. The ground was covered with
small green plants and the trees were tall and
thin. There were many birds and insects in the
trees.

At the fourth place there was a great deal of
green vegetation. The ground was covered with
small green plants and the trees were tall and
thin. There were many birds and insects in the
trees.

At the fifth place there was a great deal of
green vegetation. The ground was covered with
small green plants and the trees were tall and
thin. There were many birds and insects in the
trees.

At the sixth place there was a great deal of
green vegetation. The ground was covered with
small green plants and the trees were tall and
thin. There were many birds and insects in the
trees.

At the seventh place there was a great deal of
green vegetation. The ground was covered with
small green plants and the trees were tall and
thin. There were many birds and insects in the
trees.

At the eighth place there was a great deal of
green vegetation. The ground was covered with
small green plants and the trees were tall and
thin. There were many birds and insects in the
trees.

At the ninth place there was a great deal of
green vegetation. The ground was covered with
small green plants and the trees were tall and
thin. There were many birds and insects in the
trees.