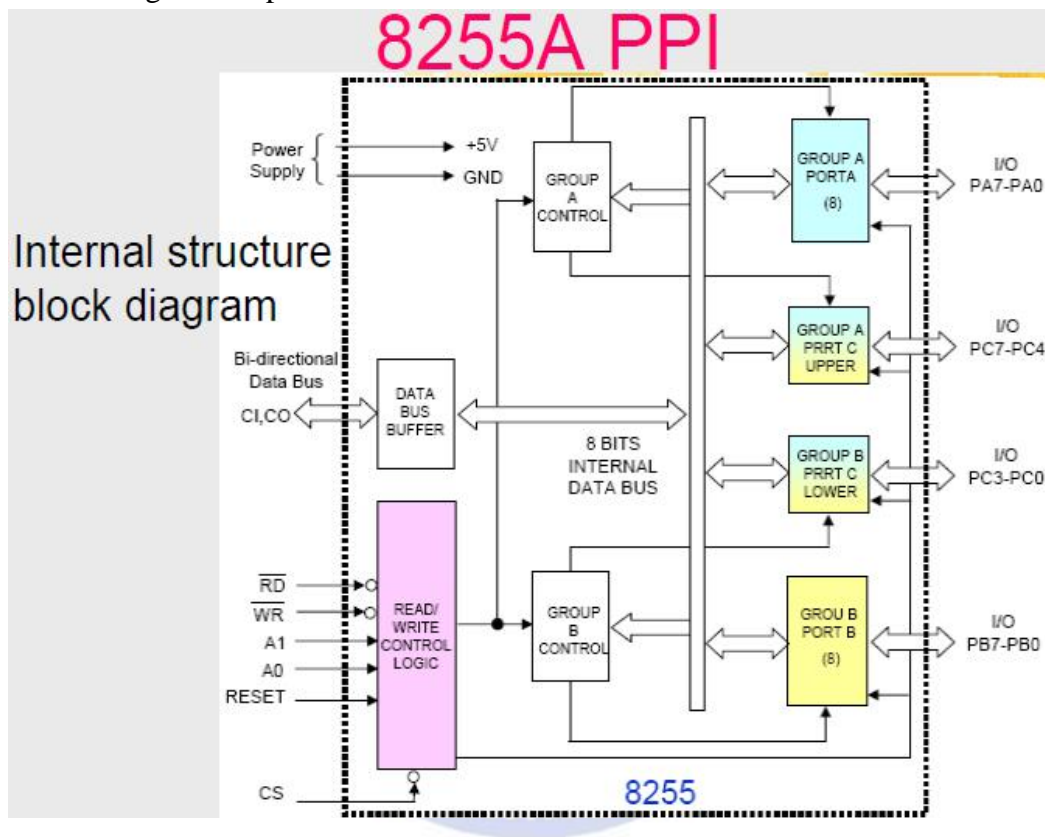


## UNIT-IV

### PROGRAMMABLE PERIPHERAL INTERFACE – 8255

Intel 8255A is a general purpose parallel I/O interface. It provides three I/O port (A, B and C). 8255A's mode of operation is determined by the contents of its control register. Port A and Port B can be set to different mode and input/output independently.

8255 has 24 I/O lines which may be individually programmed in two groups of 12 lines each or three groups of 12 line each. The two groups of I/O lines are named as Group A and Group B. There are three ports, port A and port B can be used as an 8 – bit I/O port. Port C can be used as an 8 – bit I/O port as two 4 – bit ports, or to produce handshake signals for port A and Port B.



#### Data Bus Buffer

- It is three-state bi-directional 8-bit buffer
- Used to interface the 82C55A to the system data bus.
- Data is transmitted or received by the buffer upon execution of input or output instructions by the CPU.
- Control words and status information are also transferred through the data bus buffer.

#### Read/Write and Control Logic

- **The function of this block is to manage all of the internal and external transfers of both Data and Control or Status words. It accepts inputs from the CPU Address and Control busses and in turn, issues commands to both of the Control Groups.**
- **(CS)** Chip Select. A “low” on this input pin enables the communication between the 82C55A and the CPU.
- **(RD)** Read. A “low” on this input pin enables 82C55A to send the data or status information to the CPU on the data bus. In essence, it allows the CPU to “read from” the 82C55A.

A1	A0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	INPUT OPERATION (READ)
0	0	0	1	0	Port A → Data Bus
0	1	0	1	0	Port B → Data Bus
1	0	0	1	0	Port C → Data Bus
1	1	0	1	0	Control Word → Data Bus

A0	A1	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	OUTPUT OPERATION (WRITE)
0	0	1	0	0	Data Bus → Port A
0	1	1	0	0	Data Bus → Port B
1	0	1	0	0	Data Bus → Port C
1	1	1	0	0	Data Bus → Control

**(WR)Write.** A “low” on this input pin enables the CPU to write data or control words into the 82C55A.

**(A0 and A1)Port Select 0 and Port Select 1.** These input signals, in conjunction with the RD and WR inputs, control the selection of one of the three ports or the control word register. They are normally connected to the least significant bits of the address bus (A0 and A1).

**(RESET) Reset.** A “high” on this input initializes the control register to 9Bh and all ports (A, B, C) are set to the input mode. “Bus hold” devices internal to the 82C55A will hold the I/O port inputs to a logic “1”state with a maximum hold current of 400mA.

## Group A and Group B Controls

The functional configuration of each port is programmed by the systems software. In essence, the CPU “outputs” a control word to the 82C55A. The control word contains information such as “mode”, “bit set”, “bit reset”, etc., that initializes the functional configuration of the 82C55A.

Each of the Control blocks (Group A and Group B) accepts “commands” from the Read/Write Control logic, receives “control words” from the internal data bus and issues the proper commands to its associated ports.

Control Group A -Port A and Port C upper (C7 -C4)

Control Group B -Port B and Port C lower (C3 -C0)

The control word register can be both written and read as shown in the “Basic Operation” table. When the control word is read, bit D7 will always be logic “1”, as this implies control word mode information.

A0	A1	RD	WR	CS	Disable Functions
X	X	X	X	1	Data Bus → Three-state
X	X	1	1	0	Data Bus → Three-state

## How to set 8255A? –by control words

Two control word formats are used:

*Mode-set control word format:* to set the modes of each port

*Port C bit set/reset control word format:* to set the particular bit in port C.

These two formats are differentiated by the MSB of the control word.

The control words can be sent to the corresponding 8255A’s address by using I/O instruction.

**Ex. Assume 8255A is located at 0FFF8H, control register address is 0FFFEH and the control word is to be set to 10001110B**

**MOV AL, 10001110B**

**MOV DX, 0FFFEH**

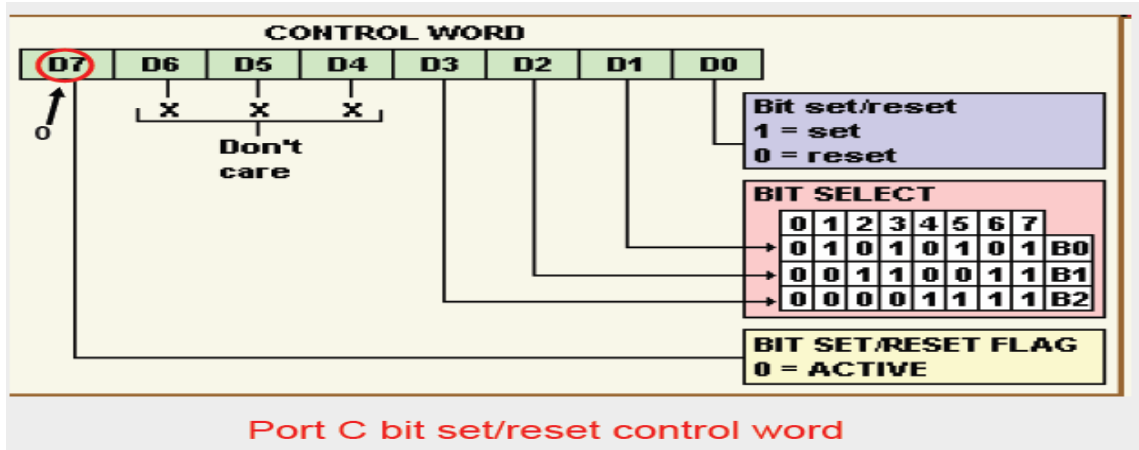
**OUT DX, AL**

**BSR MODE:** BSR mode stands for Bit Set Reset mode.

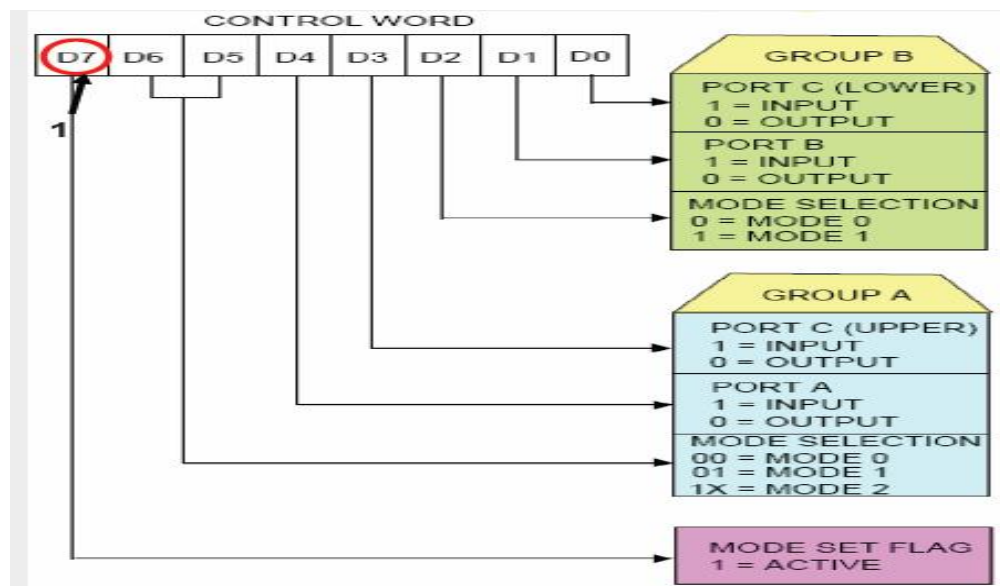
- The characteristics of BSR mode are:
- BSR mode is selected only when **D<sub>7</sub> = 0** of the Control Word Register (CWR).

# MICROPROCESSORS AND MICROCONTROLLERS MATERIAL

- Individual bits of Port C can either be Set or Reset.
- At a time, only a single bit of port C can be Set or Reset.
- Is used for control or on/off switch.
- BSR control word doesn't affect ports A and B functioning.



## MODE SET CONTROL WORD:



(a) Port A = Input port, Port B = not used, Port C<sub>U</sub> = Input port and Port C<sub>L</sub> = Output port.

(b) Port A = Output port, Port B = Input port, Port C = Output port

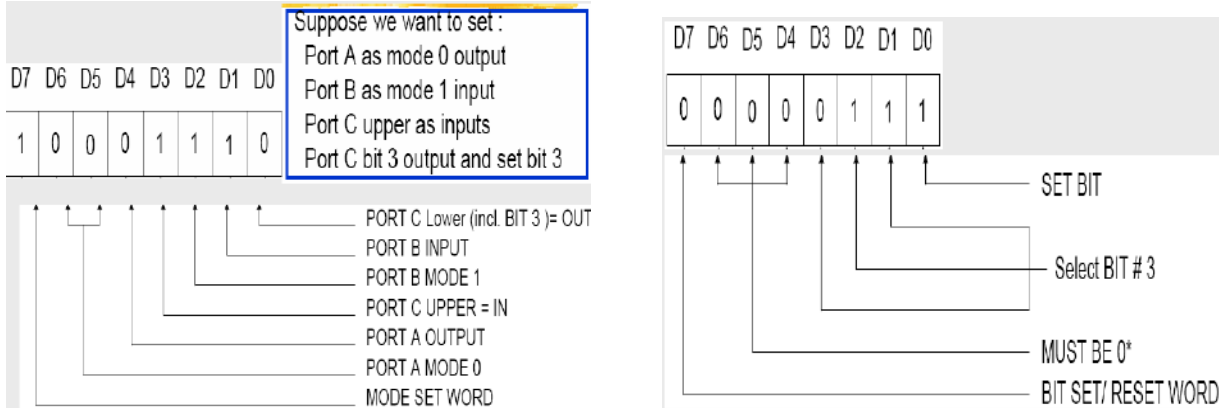
The control words for the two cases will be as follows:

(a)	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>	
	1	0	0	1	1	0	0	0	= 98 <sub>H</sub>
IO mode		Mode 0 for Port A		Port A input	Port C <sub>U</sub> input	Port B not used		Port C <sub>L</sub> output	

Thus, the control word would be = 98<sub>H</sub>

(b)	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>	
	1	0	0	0	0	0	1	0	= 82 <sub>H</sub>
IO mode		Mode 0 for Port A		Port A output	Port C <sub>U</sub> output	Mode 0 for Port B	Port B input	Port C <sub>L</sub> output	

Thus, the control word would be = 82<sub>H</sub>.



## 8255A Mode 0 Operation

- In Mode 0 operation, no handshaking will be used.
- If both port A and port B are initialized as mode 0 operation, port C can be used together as an additional 8-bit port, or two 4-bit ports.
- When used as an output, port C line can be set/reset by sending special control word to the control register address.
- The two halves of port C are independent and can be set as input or output port independently.

## 8255A Mode 1 Operation

- In Mode 1 operation, single handshaking (strobed) is used.
- In this mode, some of the port C pins are used for handshaking purpose.
- If Port A set to mode 1 & input port: PC3, PC4, and PC5 pins are used.
- If Port A set to mode 1 & output port: PC3, PC6, and PC7 pins are used.
- If Port B set to mode 1 & input port: PC0, PC1, and PC2 pins are used.
- If Port B set to mode 1 & output port: PC0, PC1, and PC2 pins are used.

## 8255A Mode 2 Operation

- Only port A can be initialized in mode 2.
- In mode 2, port A can be used as bi-directional handshake data transfer.
- In this mode, PC3-PC7 is used for handshake lines.
- PC0-PC2 can be used as I/O pins if port B is set to mode 0.
- PC0-PC2 is used as handshake lines if port B is set to mode 1.
- Note that mode 2 does not mean that it uses double handshake I/O.
- 8255A Control Words

## HANDSHAKING SIGNALS

- A byte of data is presented to the data bus of the receiver device.
- The receiver device is informed of the presence of a byte of data to be processed by activating **STROBE** input signal.
- Whenever the receiver device receives the data, it informs the sender by activating an output signal called **ACK** (acknowledge).
- The ACK signal initiates the process of providing another byte of data to the receiver device.

## Control Signals for MODE 1:

### Input mode:

The following are the control signals when ports A and B act as input ports (under mode 1) STBA , IBFA, INTEA for Port A and STBB , IBFB, INTEB for Port B, respectively. The details about the input control signals are discussed below:

**STB (Strobe input):** This is an active low signal generated by a peripheral device. When an external peripheral device provides a byte of the data to an input port (A or B), it informs the 8255 through the STB pin that it can load the data in to its internal register.

**IBF (Input buffer full):** On receipt of STB signal from peripheral device, the 8255 latches in to its internal register the data present at PA<sub>0</sub> – PA<sub>7</sub> or PB<sub>0</sub> – PB<sub>7</sub>, and through IBF indicates that it has latched all data, but it has not read by the CPU yet.

**INTR (Interrupt request):** This active high output signal is generated only if STB, IBF and INTE are all set at the same time. This signal interrupts the CPU via its INTR.

To get the attention of the CPU to read the data, IBF activates INTR (sets it high). When IBF goes active, INTR is activated to inform the CPU that there is a byte of data in the 8255. When the CPU reads the data by activating the RD signal of 8255 that INTR becomes inactive.

**INTE (Interrupt Enable):** This is an internal F/F which can be set/reset using the BSR mode. It must be set if INTR signal is to be effective.

**MODE1 – STROBED OUTPUT:** The following are the control signals when ports A and B act as output ports (under mode 1) OBFA, ACKA, INTEA for Port A and OBFB, ACKB, INTEB for Port B respectively. The details about the output control signals are discussed below:

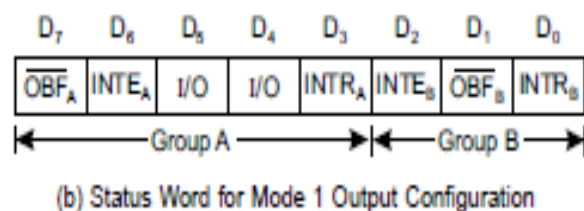
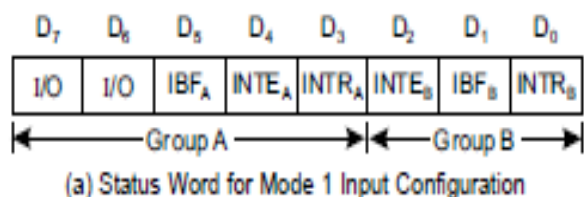
**OBF (Output buffer full):** This is an active low output signal. This signal indicates that CPU has written a byte of data in to port A. It must be connected to the STROBE input of the receiving equipment to inform the device that it can read a byte of data from the port latch.

**ACK (Acknowledge):** It informs the 8255 that the data at port A has been collected by the receiving device. When the receiving device collects data at port, it must inform the 8255 through the ACK signal.

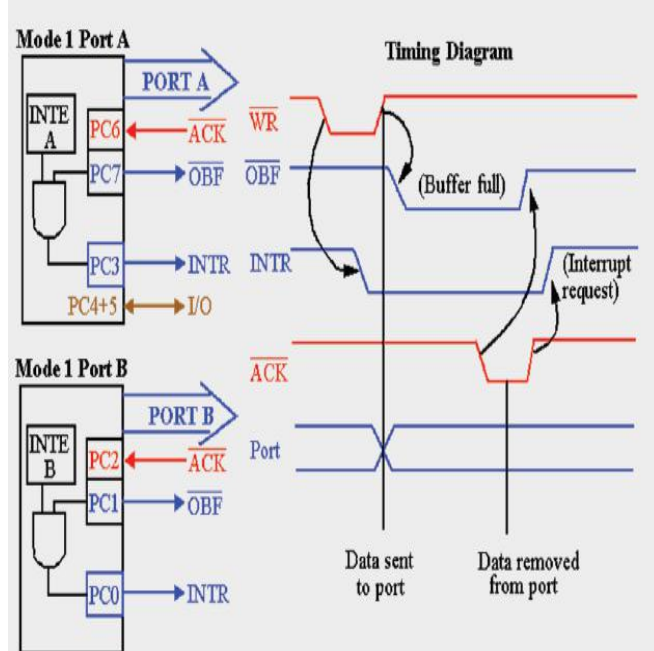
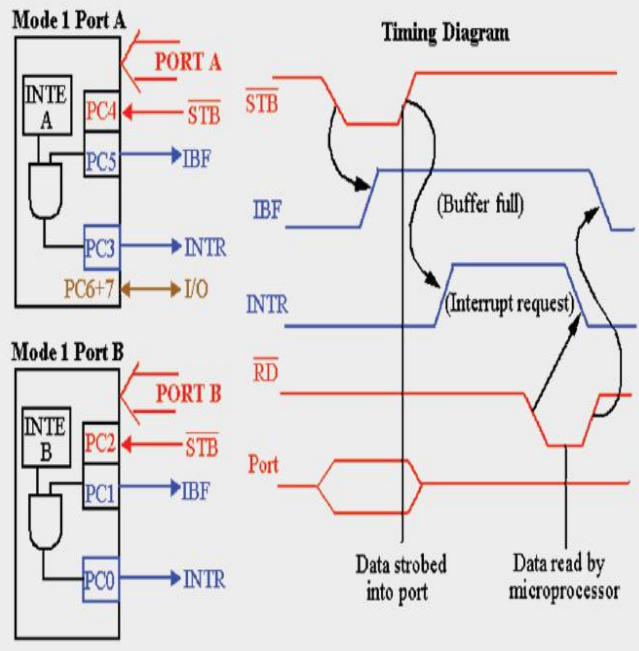
The 8255 in turn makes OBF signal as high, to indicate that the data at the port is old data now. OBF will not go low until the CPU writes a new byte of data to port A.

**INTR (Interrupt):** This signal is set only if OBF, ACK and INTE (internal F/F) are all at high (1) state. This output signal from 8255 goes to INTR interrupt the CPU. This signal is set to indicate the CPU that the receiving device has received the last byte and is ready to receive another one.

**INTE (Interrupt Enable):** This is an internal F/F which can be set/reset in BSR mode. This must be set if INTR signal is to be effective.







## 8253 PROGRAMMABLE INTERVAL TIMER

### NEED FOR PROGRAMMABLE INTERVAL TIMER

There are many situations where accurate time delays are required to be generated in a microcomputer system.

The 8253/54 solves one of most common problem in any microcomputer system, the generation of accurate time delays under software control. Instead of setting up timing loops in system software, the programmer configures the 8253/54 to match his requirements, initializes one of the counters of the 8253/54 with the desired quantity, then upon command the 8253/54 will count out the delay and interrupt the CPU when it has completed its task. It is easy to see that the software overhead is minimum and that multiple delays can be easily be maintained by assignment of priority levels.

*"The 8253/54 includes three identical 16 bit counters that can operate independently. To operate a counter, a 16-bit count is loaded in its register and, on command; it begins to decrement the count until it reaches 0. At the end of the count, it generates a pulse that can be used to interrupt the CPU. The counter can count either in binary or BCD. In addition, a count can be read by the CPU while the counter is decrementing".*

#### Differences between 8253 and 8254:

8253	8254
1. Operating frequency 0 - 2.6 MHz.	1. Operating frequency 0 - 10 MHz.
2. Uses N-MOS technology.	2. Uses H-MOS technology.
3. Read-Back command not available.	3. Read-Back command available.
4. Reads and writes of the same counter can not be interleaved.	4. Reads and writes of the same counter can be interleaved.

#### Features:

- 1) Three independent 16-bit down counters.
- 2) 8254 can handle inputs from DC to 10 MHz (5MHz 8254-5 8MHz 8254 10MHz 8254-2) where as 8253 can operate up to 2.6 MHz

- 3) Three counters are identical pre-settable, and can be programmed for either binary or BCD count.
- 4) Counter can be programmed in six different modes.
- 5) Compatible with all Intel and most other microprocessors.
- 6) 8254 has powerful command called READ BACK command which allows the user to check the count value, programmed mode and current mode and current status of the counter.

**Note:** Read back mode facility—which means that the status of a particular mode can be read after programming. This facility is not available with 8253.

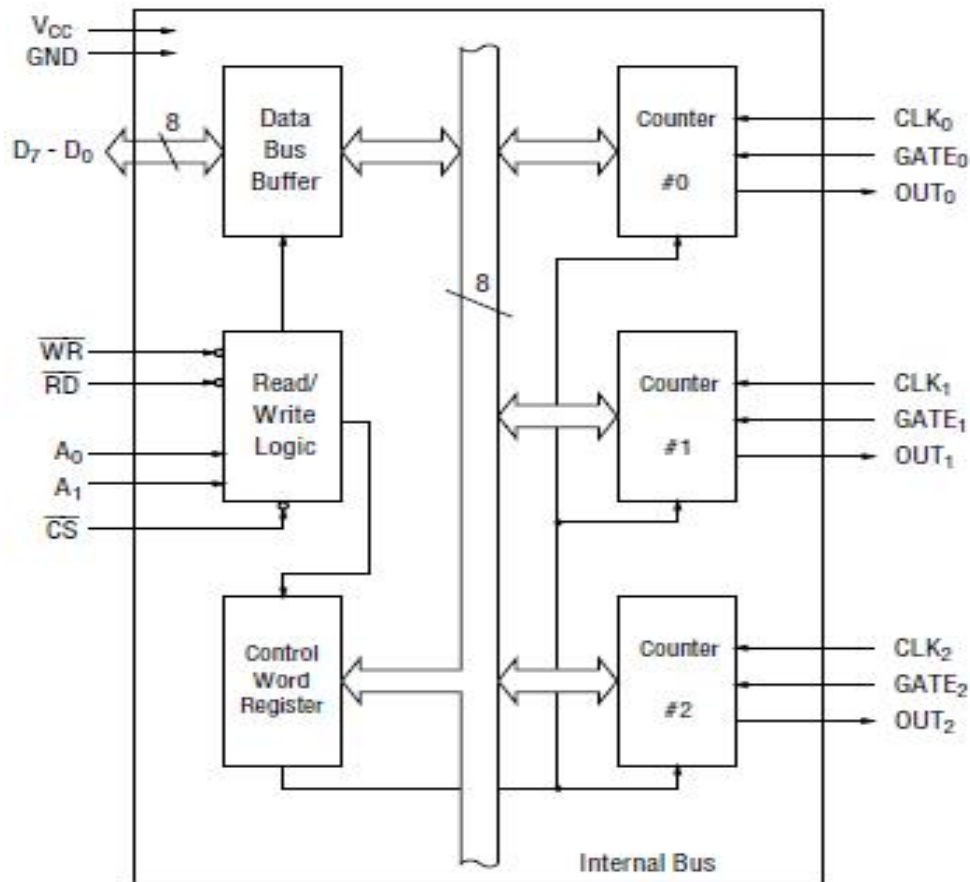
**Functional Diagram of 8253/8254:** It includes three counters, a data bus buffer, Read/Write control logic, and a control register. Each counter has two input signals CLOCK and GATE and one output signal OUT.

**Data Bus Buffer:** This tri-state, bi-directional, 8-bit buffer is used to interface the 8253/54 to the system data bus. The Data bus buffer has three basic functions.

1. Programming the modes of 8253/54.
2. Loading the count registers.
3. Reading the count values.

**Read/Write Logic:** The Read/Write logic has five signals: RD, WR, CS and the address lines A<sub>0</sub> and A<sub>1</sub>. The control word register and counters are selected according to the signals on lines A<sub>0</sub> and A<sub>1</sub>.

A <sub>1</sub>	A <sub>0</sub>	Selection
0	0	Counter 0
0	1	Counter 1
1	0	Counter 2
1	1	Control word Register



## BLOCK DIAGRAM OF PIT 8253

**Control Word Register:** This register is accessed when lines A<sub>0</sub> and A<sub>1</sub> are at logic 1. It is used to write a command word which specifies the counter to be used (binary or BCD), its mode, and either a read or write operation.

**Counters:** These three functional blocks are identical in operation. Each counter consists of a single, 16 bit, pre-settable, down counter. The counter can operate in either binary or BCD and its input, gate and output are configured by the selection of modes stored in the control word register. The counters are fully independent. The programmer can read the contents of any of the three counters without disturbing the actual count in process.

Each counter has a program control word used to select the way the counter operates. If two bytes are programmed, then the first byte (LSB) stops the count, and the second byte (MSB) starts the counter with the new count.

**GATE:** The gate input controls the operation of the counter in some modes.

**OUT:** A counter output is where the waveform generated by the timer is available.

**RD/WR:** Read/Write causes data to be read/written from the 8254 and often connects to the IORC/IOWC.

**Note:** Each counter can be programmed with maximum count as  $10^4$  i.e. 10,000 in decimal where as in hexadecimal  $2^{16}$  (65,536H).

### Operational Description

The complete functional definition of the 8253/54 is programmed by the system software. Once programmed, the 8253/54 is ready to perform whatever timing tasks it is assigned to accomplish.

**Programming the 8253/54:** Each counter of the 8253/54 is individually programmed by writing a control word into the control word register (A<sub>0</sub> - A<sub>1</sub> = 11). The Figure shows the control word format. Bits SC<sub>1</sub> and SC<sub>0</sub> select the counter, bits RW<sub>1</sub> and RW<sub>0</sub> select the read, write or latch command, bits M<sub>2</sub>, M<sub>1</sub> and M<sub>0</sub> select the mode of operation and bit BCD decides whether it is a BCD counter or binary counter.



## WRITE Operation:

1. Write a control word into control register.
2. Load the low-order byte of a count in the counter register.
3. Load the high-order byte of count in the counter register.

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
SC <sub>1</sub>	SC <sub>0</sub>	RW <sub>1</sub>	RW <sub>0</sub>	M <sub>2</sub>	M <sub>1</sub>	M <sub>0</sub>	BCD

### SC - Select counter

SC<sub>1</sub> SC<sub>0</sub>

0	0	Select counter 0
0	1	Select counter 1
1	0	Select counter 2
1	1	Illegal for 8253 Read -Back command for 8254 (See Read operations)

### M - Mode

M<sub>2</sub> M<sub>1</sub> M<sub>0</sub>

0	0	0	Mode 0
0	0	1	Mode 1
x	1	0	Mode 2
x	1	1	Mode 3
1	0	0	Mode 4
1	0	1	Mode 5

### RW - Read /Write

RW<sub>1</sub> RW<sub>0</sub>

0	0	Counter latch command (See Read operations)
0	1	Read / Write least significant byte only
1	0	Read / Write most significant byte only
1	1	Read / write least significant byte first, then most significant byte

### BCD :

0	Binary counter 16 - bits
1	Binary coded decimal (BCD) Counter (4 Decades)

## CONTROL WORD FORMAT 8253/54

**READ Operation:** In some applications, especially in event counters, it is necessary to read the value of the count in process. This can be done by three possible methods:

1. **Simple Read:** It involves reading a count after inhibiting the counter by controlling the gate input or the clock input of the selected counter, and two I/O read operations are performed by the CPU. The first I/O operation reads the low-order byte, and the second I/O operation reads the high order byte.
2. **Counter Latch Command:** In the second method, an appropriate control word is written into the control register to latch a count in the output latch, and two I/O read operations are performed by the CPU. The first I/O operation reads the low-order byte, and the second I/O operation reads the high order byte.
3. **Read-Back Command (Available only for 8254):** The third method uses the Read-Back command. This command allows the user to check the count value, programmed Mode, and current status of the OUT pin and Null count flag of the selected counter(s). Figure shows the format of the control word register for Read-Back command.

**Modes of Operation:** Six modes of operation are available to each of the 8253/54 counters.

### MODE 0: Interrupt on terminal count

#### Normal Operation:

- 1) The output will be initially low after the mode set operation.
- 2) After the count is loaded into the selected count Register the output will remain low and the counter will count.
- 3) When the terminal count is reached the output will go high and remain high until the selected count is reloaded.

#### b) Gate Disable

- 1) Gate = 1 enables counting. 2) Gate = 0 disables counting.

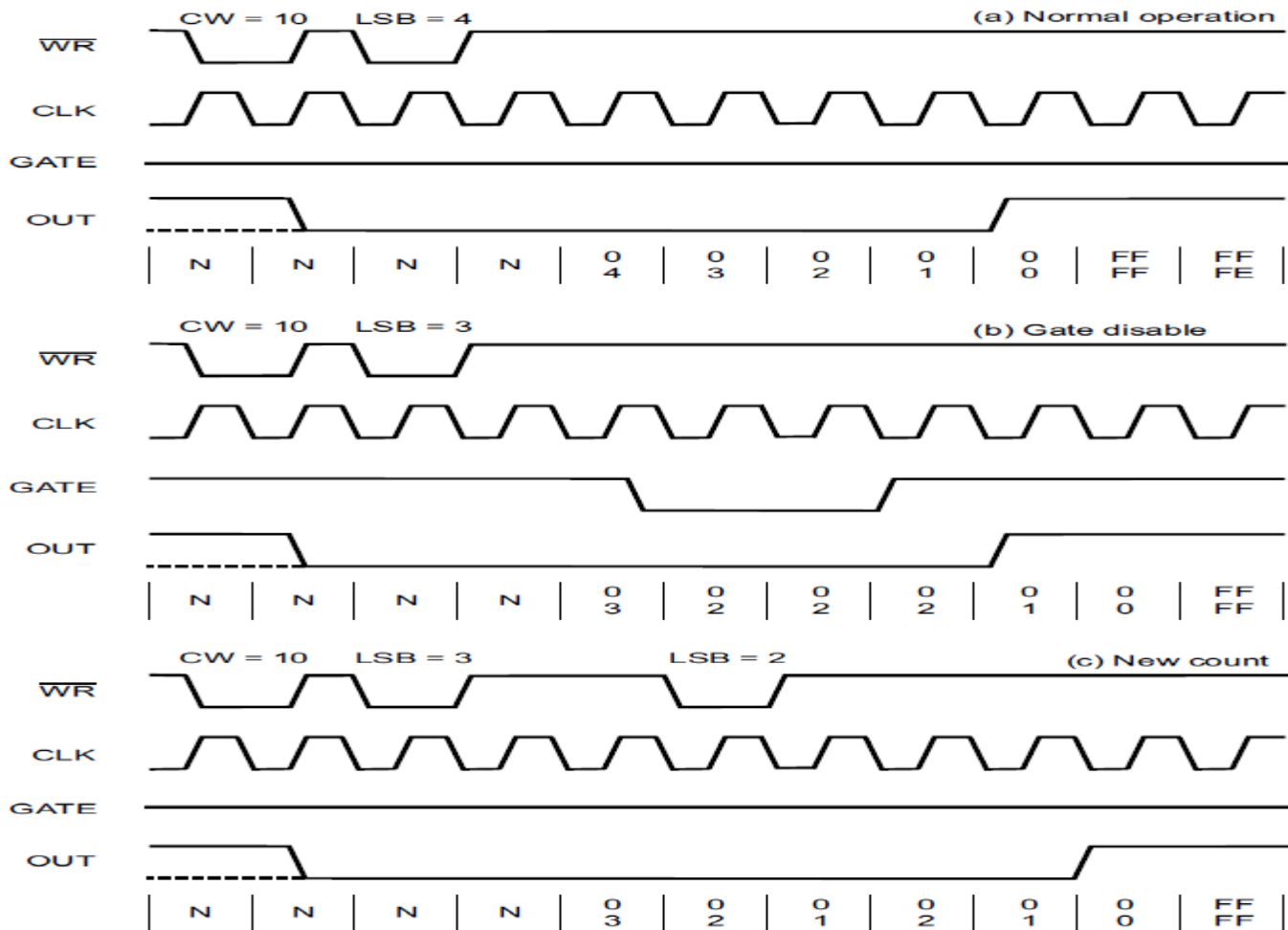
Note: Gate has no effect on OUT.

## c) New Count

If a new count is written to the counter, it will be loaded on the next CLK pulse and counting will continue from the new count

### In case of two byte count:

- 1) Writing the first byte disables counting.
- 2) Writing the second byte loads the new count on the next CLK pulse and counting will continue from the new count.



## Mode 0 interrupt on terminal count

### MODE 1: Hardware Retriggerable One-shot

#### a) Normal operation

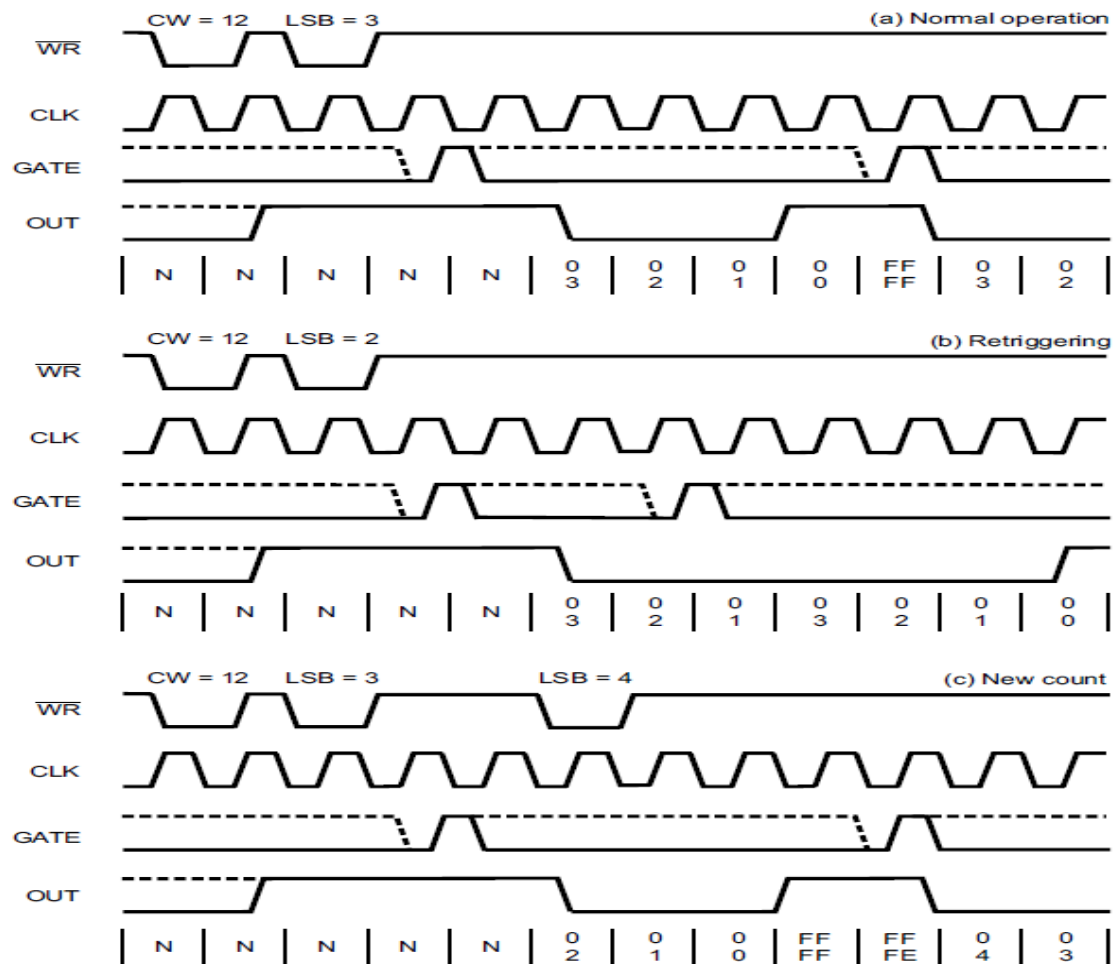
- 1) The output will be initially high
- 2) The output will go low on the CLK pulse following the rising edge at the gate input.
- 3) The output will go high on the terminal count and remain high until the next rising edge at the gate input.

#### b) Retriggering

The one shot is retriggerable, hence the output will remain low for the full count after any rising edge of the gate input.

#### c) New count

If the counter is loaded during one shot pulse, the current one shot is not affected unless the counter is retriggered. If retriggered, the counter is loaded with the new count and the one-shot pulse continues until the new count expires.



## MODE 1: HARDWARE RETRIGGERABLE ONE-SHOT

### MODE 2: Rate generator

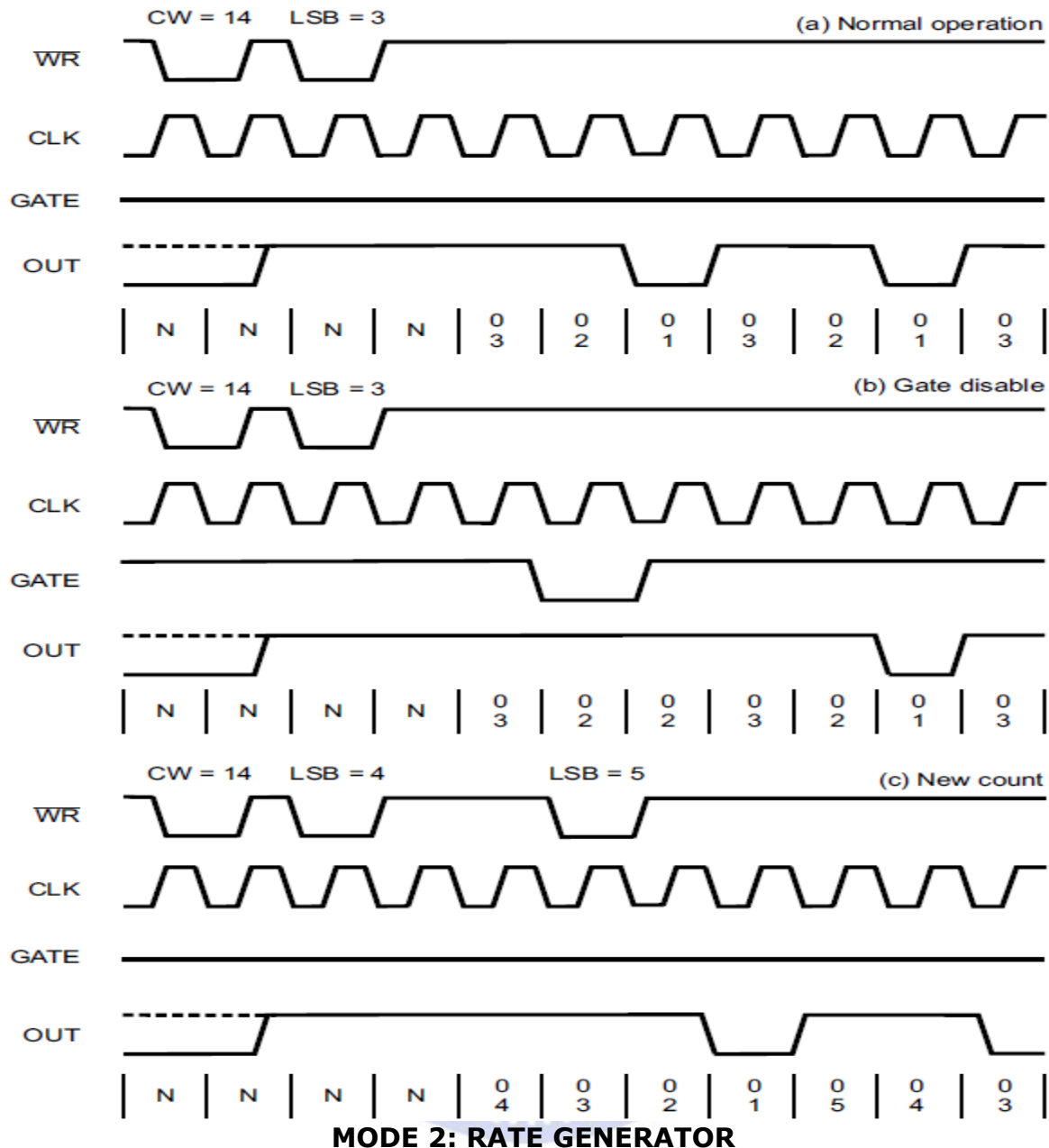
This mode functions like a divide by-N counter.

#### a) Normal Operation

- 1) The output will be initially high.
- 2) The output will go low for one clock pulse before the terminal count.
- 3) The output then goes high, the counter reloads the initial count and the process is repeated.
- 4) The period from one output pulse to the next equals the number of input counts in the count register.

#### b) Gate Disable

- 1) If Gate = 1 it enables a counting otherwise it disables counting (Gate = 0).
- 2) If Gate goes low during a low output pulse, output is set immediately high. A trigger reloads the count and the normal sequence is repeated.



## MODE 3: SQUARE WAVE GENERATOR

### a) Normal operation

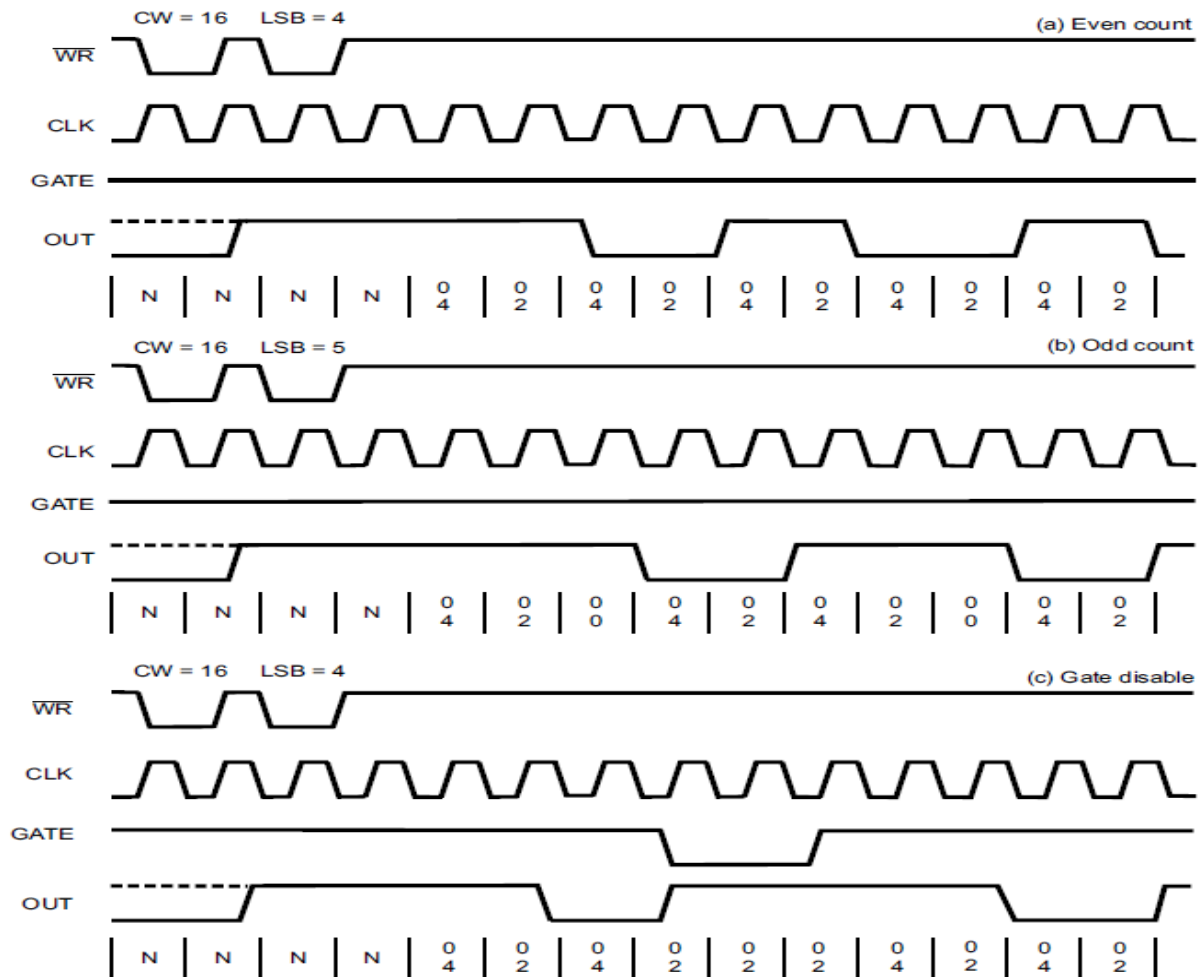
- 1) Initially output is high.
- 2) For even count, counter is decremented by 2 on the falling edge of each clock pulse. When the counter reaches terminal count, the state of the output is changed and the counter is reloaded with the full count and the whole process is repeated.
- 3) If the count is odd and the output is high the first clock pulse (after the count is loaded) decrements the count by 1. Subsequent clock pulses decrement the clock by 2. After timeout, the output goes low and the full count is reloaded. The first clock pulse (following the reload) decrements the count by 3 and subsequent clock pulse decrement the count by two. Then the whole process is repeated. In this way, if the count is odd, the output will be high for  $(n+1)/2$  counts and low for  $(n-1)/2$  counts.

### b) Gate Disable

If Gate is 1 counting is enabled otherwise it is disabled. If Gate goes low while output is low, output is set high immediately. After this, When Gate goes high, the counter is loaded with the initial count on the next clock pulse and the sequence is repeated.

## c) New Count

The current counting sequence does not affect when the new count is written. If a trigger is received after writing a new count but before the end of the current half-cycle of the square wave, the counter will be loaded with the new count on the next CLK pulse and counting will continue from the new count. Otherwise, the new count will be loaded at end of the current half-cycle.



### MODE 3: SQUARE WAVE GENERATOR

## MODE 4: Software Triggered Strobe.

### a) Normal operation

- 1) The output will be initially high
- 2) The output will go low for one CLK pulse after the terminal count (TC).

### b) Gate Disable

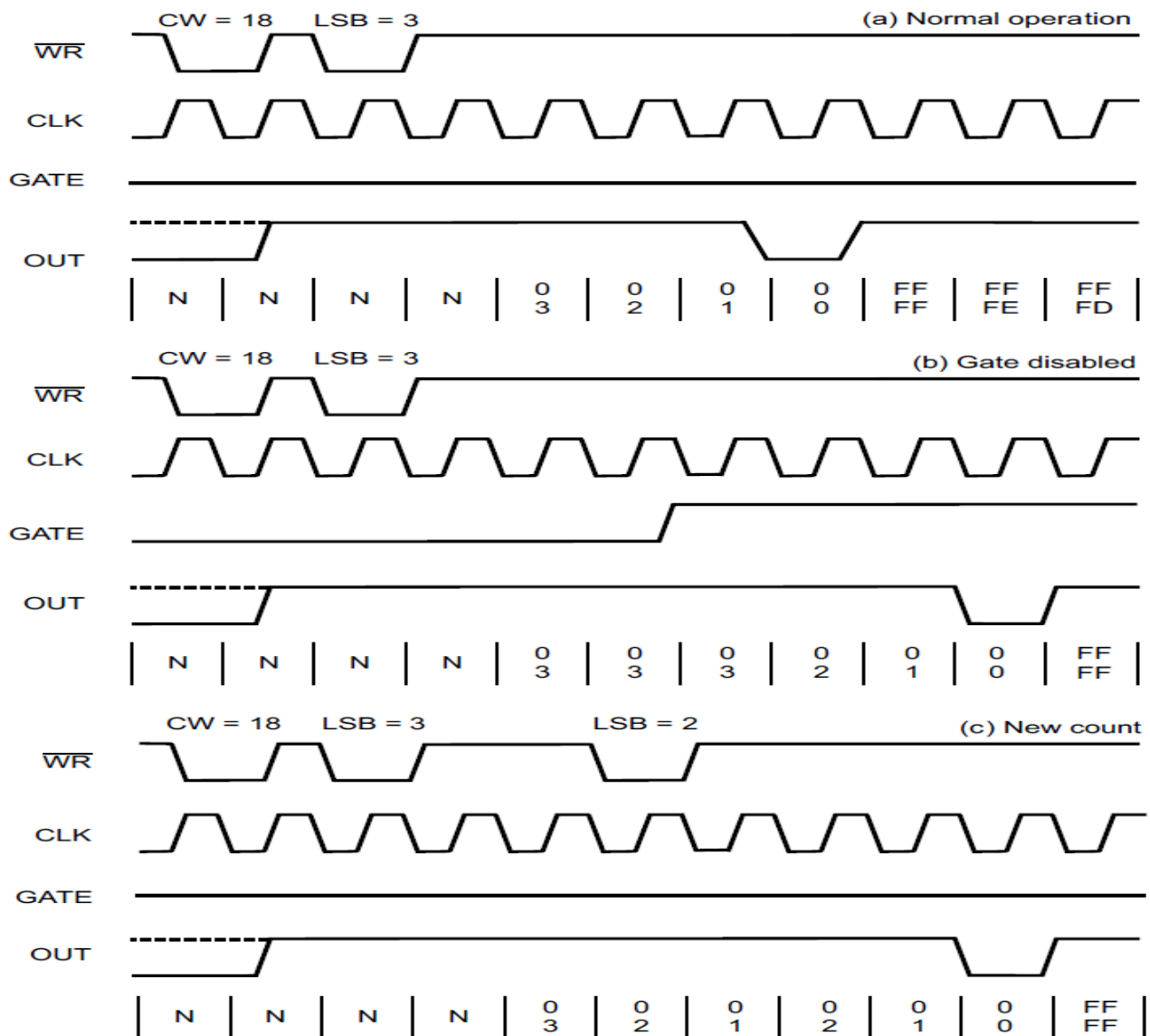
If Gate is one the counting is enabled otherwise it is disabled. The Gate has no effect on the output.

### c) New count

If a new count is written during counting, it will be loaded on the next CLK pulse and counting will continue from the new count. If the count is two byte then

- 1) Writing the first byte has no effect on counting.
- 2) Writing the second byte allows the new count to be loaded on the next CLK pulse.





## MODE 4: SOFTWARE TRIGGERED STROBE

### MODE 5: Hardware triggered strobe (Retriggerable).

#### a) Normal operation

- 1) The output will be initially high.
- 2) The counting is triggered by the rising edge of the Gate.
- 3) The output will go low for one CLK pulse after the terminal count (TC).

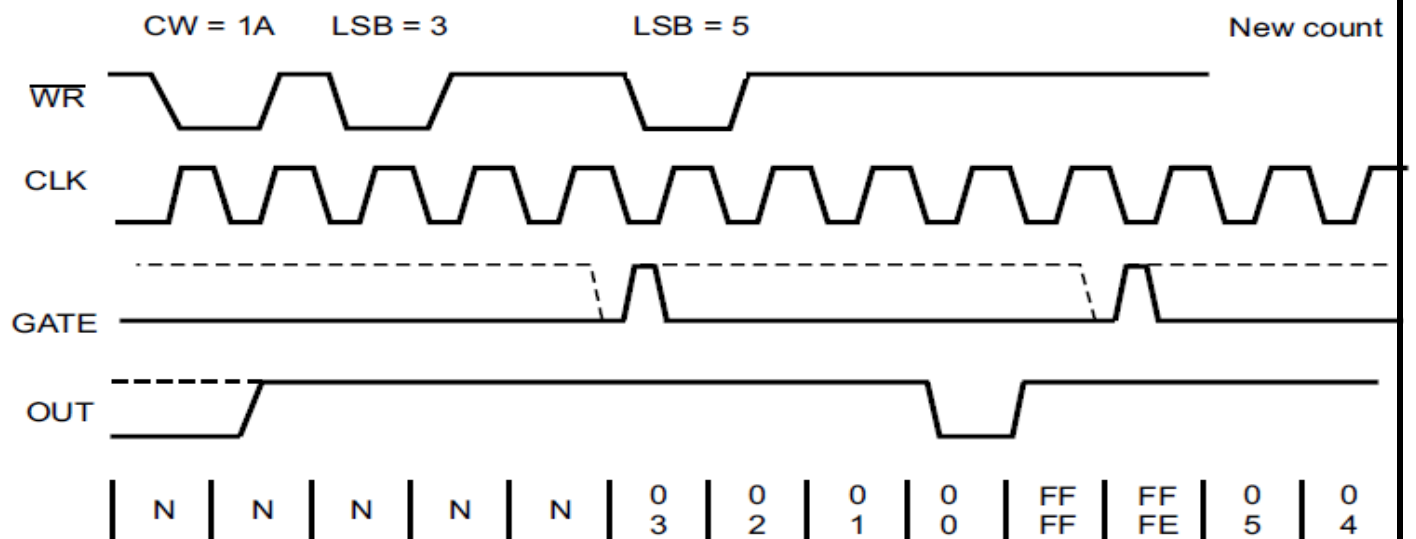
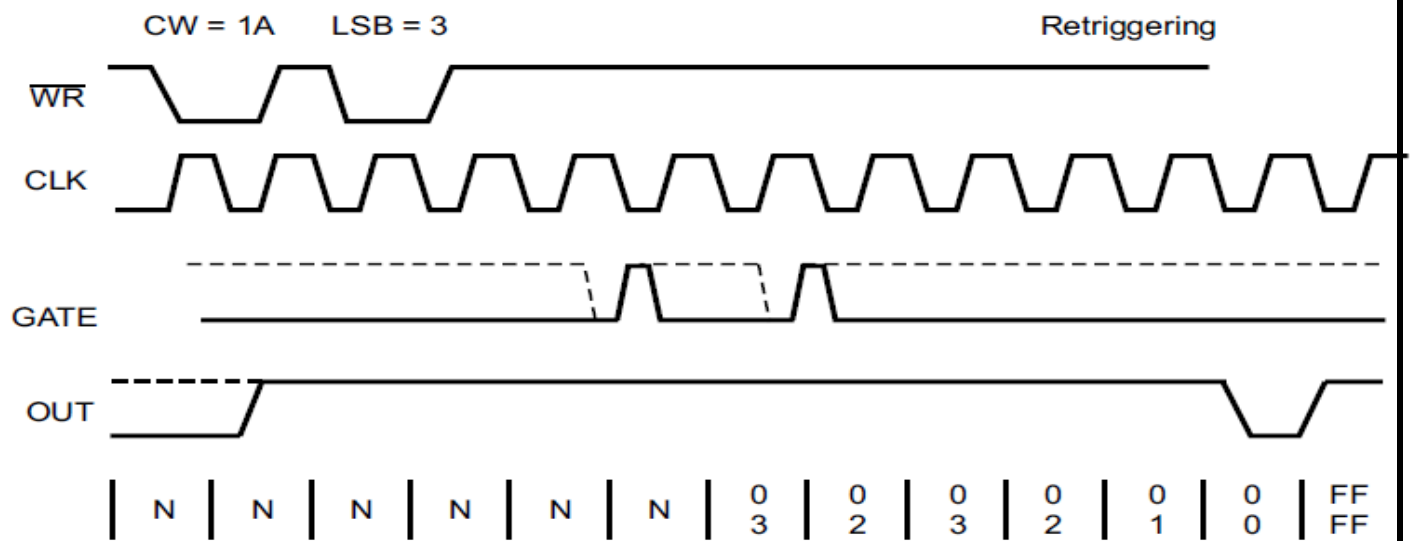
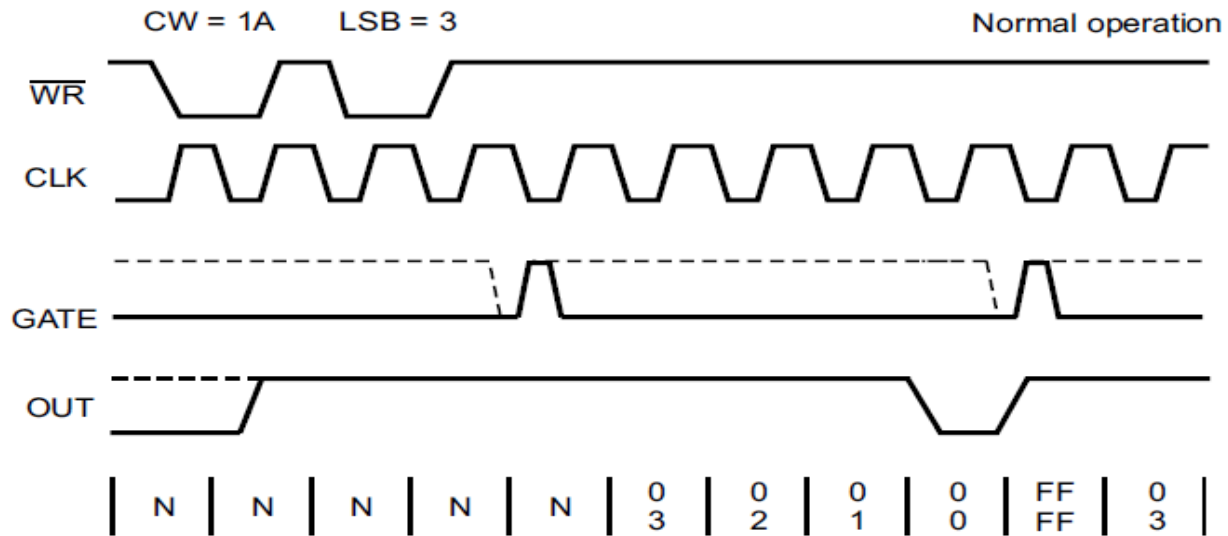
#### b) Retriggering

If the triggering occurs on the Gate input during the counting, the initial count is loaded on the next CLK pulse and the counting will be continued until the terminal count is reached.

#### c) New count

If a new count is written during counting, the current counting sequence will not be affected. If the trigger occurs after the new count is written but before the terminal count, the counter will be loaded with the new count on the next CLK pulse and counting will continue from there.

# MICROPROCESSORS AND MICROCONTROLLERS MATERIAL



**MODE 5: HARDWARE TRIGGERED STROBE**

**Example:** Use the port addresses as **binary address A7 - A2 = 100101**:

- (a) Counter 0 for binary count of mode 3 (square wave) to divide CLK0 by number 4282 (BCD)
- (b) Counter 2 for binary count of mode 3 (square wave) to divide CLK2 by number C26A hex
- (c) Find the frequency of OUT0 and OUT2 in (a) and (b) if CLK0 = 1.2 MHz, CLK2 = 1.8 MHz

**Solution:**

(a) To program counter 0 for mode 3, we have 00110111 for the control word. Therefore,

```
MOV AL, 37H      ; counter 0, mode 3, BCD
OUT 97H, AL      ; send it to control register
MOV AX, 4282H    ; load the divisor (BCD needs H for hex)
OUT 94H, AL      ; send the low byte
MOV AL, AH       ; to counter 0
OUT 94H, AL      ; and then the high byte to counter 0
```

(b) By the same token:

```
MOV AL, B6H      ; counter2, mode 3, binary (hex)
OUT 97H, AL      ; send it to control register
MOV AX, C26AH    ; load the divisor
OUT 96H, AL      ; send the low byte
MOV AL, AH       ; to count 2
OUT 96H, AL      ; send the high byte to counter 2
```

(c) The output frequency for OUT0 is 1.2MHz divided by 4282, which is 280 Hz. Notice that the program in part (a) used instruction "MOV AX,4282H" since BCD and hex numbers are represented in the same way, up to 9999. For OUT2, CLK2 of 1.8 MHz is divided by 49770 since C26AH = 49770 in decimal. Therefore, OUT2 frequency is a square wave of 36 Hz.

**Example:** Using the port addresses as 94H, show the programming of counter 1 to divide CLK1 by 10,000, producing the mode 3 square wave. Use the BCD option in the control byte.

**Solution:**

```
MOV AL, 77H      ; counter 1, mode 3, BCD
OUT 97H, AL      ; send it to control register
SUB AL, AL       ; AL =0 load the divisor for 10,000s
OUT 95H, AL      ; send the low byte
OUT 95H, AL      ; and then the high byte to counter 1
```

## Serial communication standards:

Most of the microprocessors are designed for parallel communication. In parallel communication for transferring each bit requires one cable, so for transferring large amount of data for long distances increases cost of cabling which is impractical solution. In such situations serial communication can be used which uses one single line for transmission of entire data but one bit after another only.

Serial communication can be classified in to three types based on how transmission can be done:

**Simplex:** In simplex, data transmission can be done in only one direction i.e. from transmitter to receiver only.

Eg: From computer to the printer

**Half Duplex:** This type of data transmission allows data transfer in both directions but not simultaneously.

Eg: Walkie – talkie

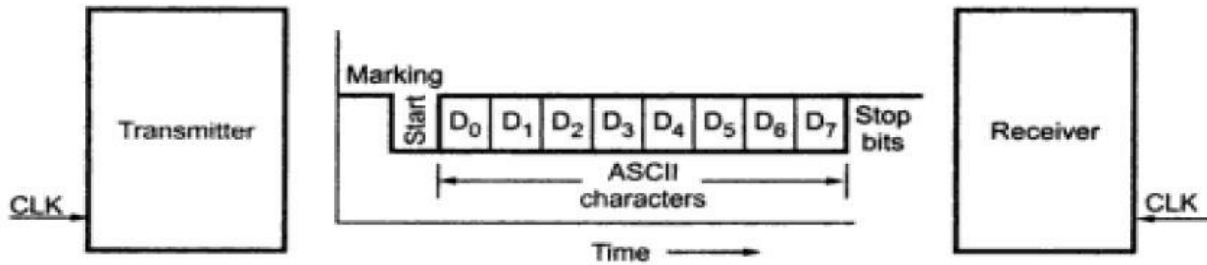
**Full Duplex:** This type of data transmission allows data transfer in both directions simultaneously.

Eg: Telephone line

**Transmission formats:** The data in the serial communication can be sent in two formats i.e. **Asynchronous** and **Synchronous**.

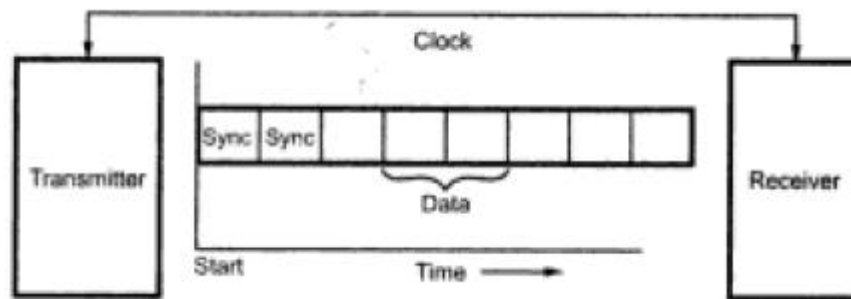
**Asynchronous transmission:** In this the bits of a character or word can be sent at a constant rate. When no characters are being sent a line stays at logic HIGH called MARK (STOP) and logic LOW called SPACE (START).

The beginning of a character is indicated by a START bit which is always LOW. This is used to synchronize the transmitter and receiver. After the start bit, the data bits are sent with LSB first, followed by one or more stop bits. The combination of start bit, character and stop bits is called FRAME.



**Figure 1: Transmission format for Asynchronous transmission**

**Synchronous transmission:** The start and stop bits in each frame format represents overhead bytes that reduce the overall character rate. This overhead bits can be eliminated by synchronizing receiver and transmitter through common clock signal such a communication is called Synchronous serial communication.



**Figure 2: Synchronous serial transmission format**

Synchronous bits are inserted instead of start and stop bits. Here a pair of SYNC bits are used at the start of data frame.

## Comparison between Asynchronous and synchronous transmission formats

SNO	ASYNCHRONOUS SERIAL COMMUNICATION	SYNCHRONOUS SERIAL COMMUNICATION
1.	Transmitters and receivers are not synchronized by clock.	Transmitters and receivers are synchronized by clock.
2.	Bits of data are transmitted at constant rate.	Data bits are transmitted with synchronisation of clock.
3.	Characters may arrive at any rate at receiver.	Character is received at constant rate.
4.	Data transfer is character oriented.	Data transfer takes place in blocks.
5.	Start and stop bits are required to establish communication of each other.	Start and stop bits are not required to establish communication of each character. However, synchronisation bits are required to transfer the data block.
6.	Used in low-speed transmission at about speed less than 20kbps.	Used in high-speed transmissions.

## UNIVERSAL ASYNCHRONOUS SYNCHRONOUS RECEIVER AND TRANSMITTER (USART – 8251A):

To implement serial communication in microprocessor system, we need two converters i.e. Parallel to Serial converter and Serial to parallel converter.

To provide this conversion UART and USART devices are designed. UART can be used for only asynchronous transmission whereas USART can be used for both asynchronous and synchronous transmissions.

## Features:

1. It supports Asynchronous standard protocol with:
  - 5 to 8 bit character format
  - Odd or even parity generation and detection
  - Baud rate from DC to 19.2k
  - False start bit detection
2. It supports Synchronous standard protocol with:
  - 5 to 8 bit character format
  - Internal or external character synchronization
  - Automatic SYNC insertion
  - Baud rate from DC to 64k baud
3. It has built in baud generator
4. It allows full duplex transmission and reception
5. It provides error detection logic, which detects parity , overrun and framing errors.
6. It has modem control logic which supports basic data set control signals.

## Block diagram:

**Data buffer:** This buffer is used to interface 8251A to the system bus. Data is transmitted or received up on execution of input and output instruction of the CPU. Command and status information are also transferred through this data buffer.

## Read/Write Control logic:

**Reset:** A high on this pin forces 8251A in to an **idle** condition.

**CLK (clock):** The clock input is used to generate internal device timing.

**C/ $\overline{D}$  (Control/Data):** This input informs the 8251A that the data on the data bus is either a data or control word/status information.

**$\overline{CS}$  (Chip Select):** If it is high no read or write operation can be carried out on 8251A.

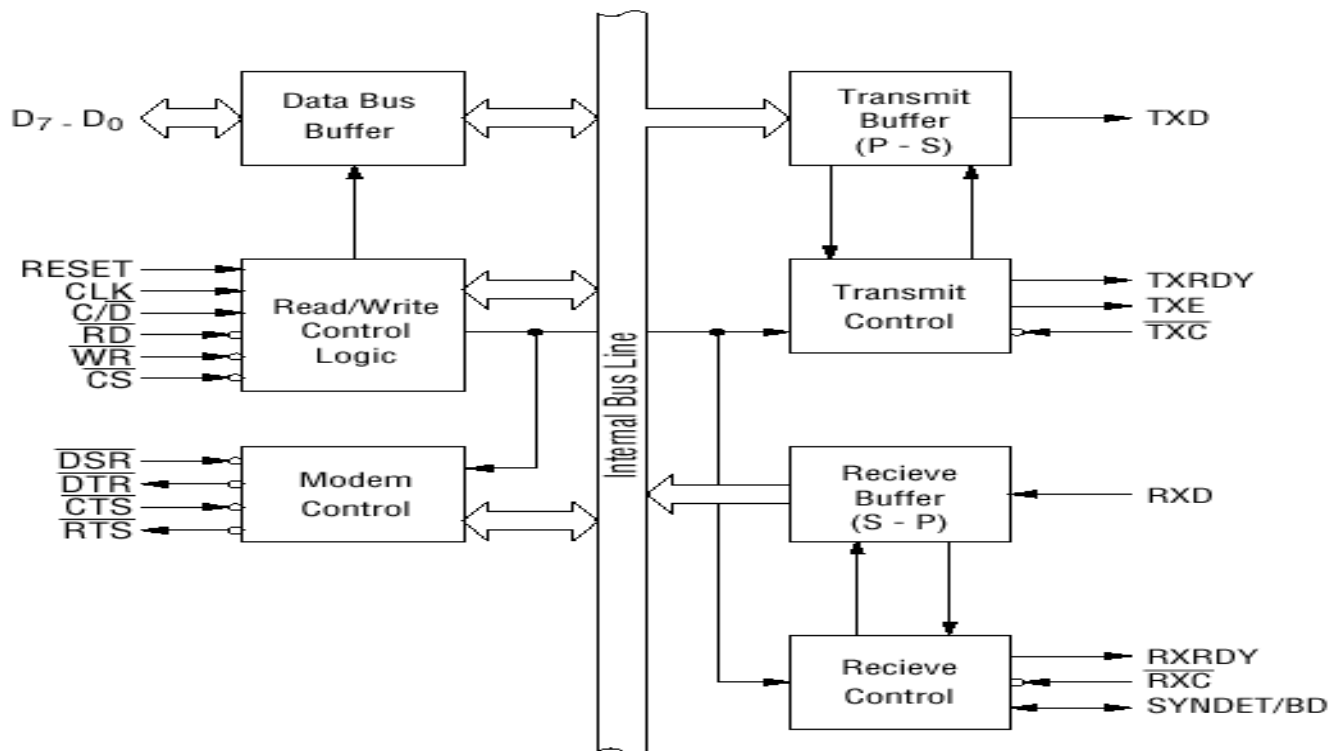


Figure : Block Diagram of USART



## Operation between CPU and 8251:

$\overline{CS}$	$C/\overline{D}$	$\overline{RD}$	$\overline{WR}$	
1	×	×	×	Data Bus 3-State
0	×	1	1	Data Bus 3-State
0	1	0	1	Status $\rightarrow$ CPU
0	1	1	0	Control Word $\leftarrow$ CPU
0	0	0	1	Data $\rightarrow$ CPU
0	0	1	0	Data $\leftarrow$ CPU

### Modem Control:

**DSR (Data Set Ready):** Its status can be checked by the CPU using a status read operation. It is used to check the status that modem is ready to communicate with CPU or not.

**DTR (Data Terminal Ready):** This is used to indicate that the device is ready to communicate with modem.

**RTS (Request To Send):** This signal indicates the beginning of transmission. This signal indicates that CPU wants to transmit some data.

**CTS (Clear To Send):** This signal indicates that modem is ready to accept the data from CPU.

### Transmitter signals:

**Transmitter buffer:** The transmitter buffer accepts parallel data from data bus buffer converts it to a serial bit stream and outputs a composite serial bit stream on TXD for transmission.

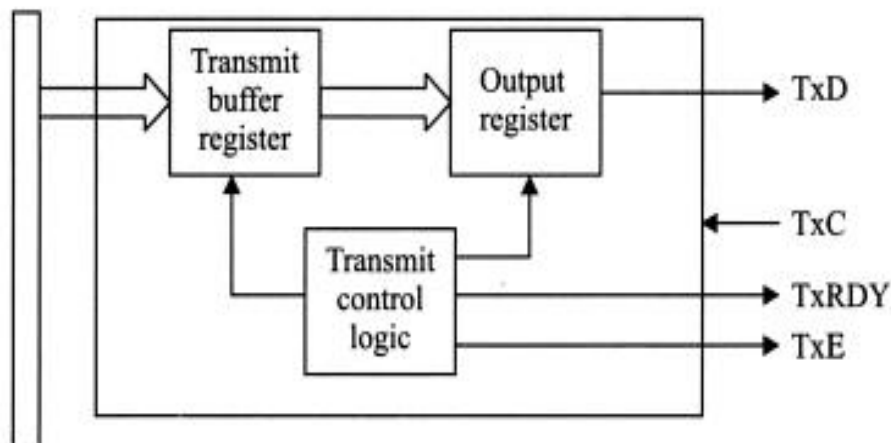
### Transmitter Control:

**TXRDY (Transmitter Ready):** This output signals the CPU that the buffer is empty and USART is ready to accept a data character. This signal is reset when buffer is loaded with data byte.

**TXEMPTY (Transmitter Empty):** While transmitting USART has no characters to transmit then this signal can be high and it automatically goes low when a character is received from CPU.

In synchronous mode, high on this pin indicates that a character is not loaded properly and SYNC characters are used as filters. TXE can be used to indicate end of transmission.

**TXC (Transmitter clock):** This transmitter clock controls the rate at which the character is to be transmitted.



### Receiver signals:

**Receiver buffer:** Receiver buffer receives serial data stream on the RXD line, converts serial data stream to parallel data stream.

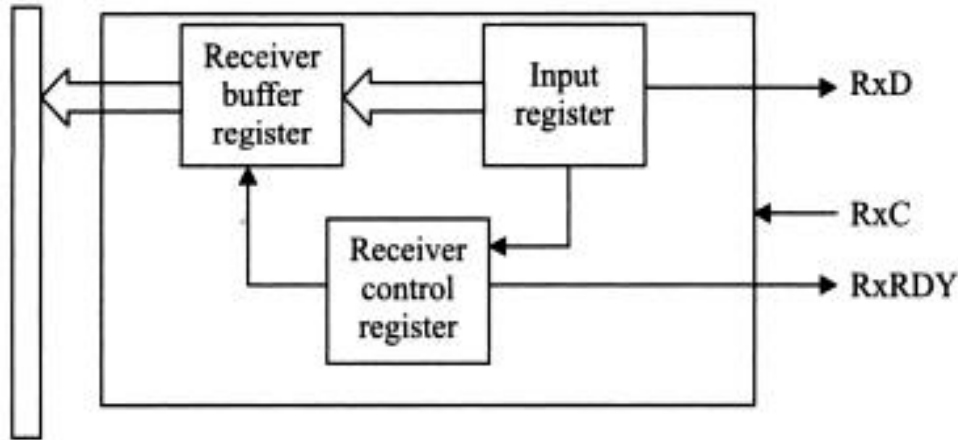
**Receive Control:** It performs false start bit detection, parity error detection, sync detection and break detection.

**RXRDY (Receiver Ready):** This signal is made high when USART has a character in the buffer register and is ready to transfer it to the CPU.

To set RXRDY in asynchronous mode, the receiver must be enabled to sense start bit and a complete character must be assembled and then transferred to the data output register.

To set RXRDY in synchronous mode, the receiver must be enabled and a complete character must be assembled and then transferred to the data output register.

If the data was not successfully read from data output register before assembly of the next data byte, **overflow condition error flag** is set i.e. previous byte is overwritten by the next byte of the incoming data and hence it is lost.



**RxC (Receiver clock):** This receiver clock controls the rate at which the character is to be received.

**SYNDET/BRKDET:** This pin is used in the synchronous mode SYNC characters (SYNDET) and it can be used either as input or output.

When used as an output, the SYNDET pin will go high which indicates that a SYNC character is located in the receive mode.

When this is used as an input, positive going signal will cause 8251A to start assembling a data character on the rising edge of the next RXC.

In asynchronous mode this pin acts as a BRKDET, a high on this pin goes low for RXD through two consecutive stop bit sequences. If RXD is high, it provides break during last bit of next character.

**8251 Registers:** The 8251 contains the following registers

Mode Instruction Register	Command Instruction Register	Status Register
Sync Character One Register	Sync Character Two Register	
Transmitter Buffer Register	Receive Buffer Register	

**Sync Character One Register:** The sync character one register holds the first sync character. This information is used by the receiver for sync comparison and by the transmitter for sync character transmission.

**Sync Character Two Register:** The sync character two register holds the first sync character. This information is used by the receiver for sync comparison and by the transmitter for sync character transmission.

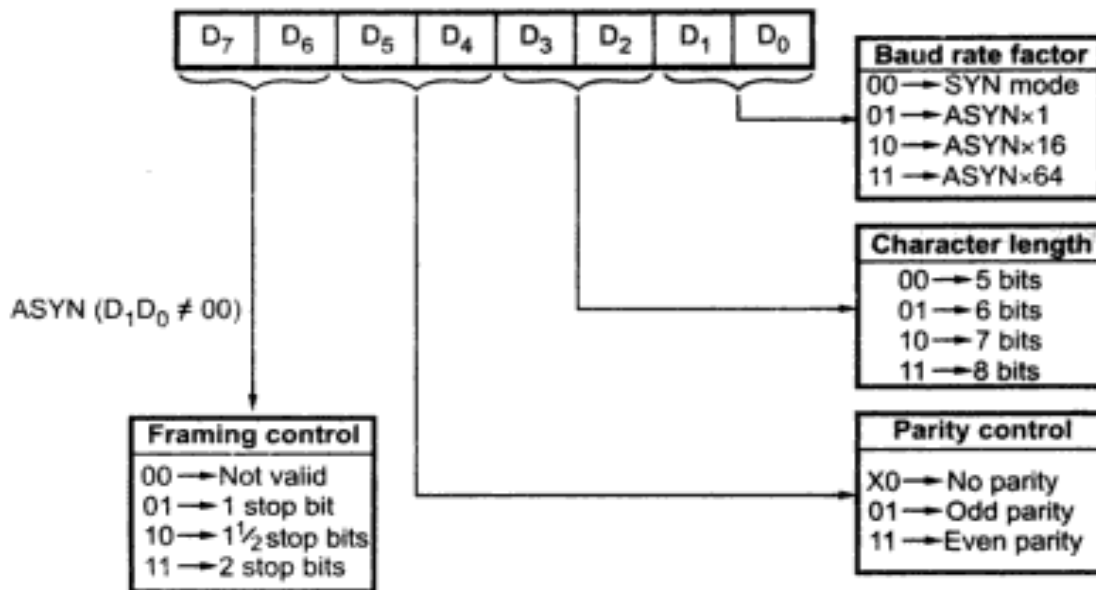
**Transmitter Buffer Register (TBR):** This register holds the transmitter data, in which 8251 formats, serializes and transmits on the TXD. Once the existing data bits in the shift register are completely transmitted, the TBR transfers new data in to the shift register.

**Receive Buffer Register (RBR):** This register holds the data received from the shift register. After the shift register receives a new data word, it is ready to transfer the new data word to the RBR. If the existing data in the RBR has already been read by the microprocessor, then the transfer takes place. If RBR data has not been read, the overrun flag is set.

## Operating modes of 8251A:

Control words of 8251A: The control word defines the functionality of 8251A, so they must be loaded before any transmission or reception. The control words of 8251A can be split into two formats: **1. Mode instruction** **2. Command instruction**

**Mode instruction:** The instruction can be considered as four 2 – bit fields.



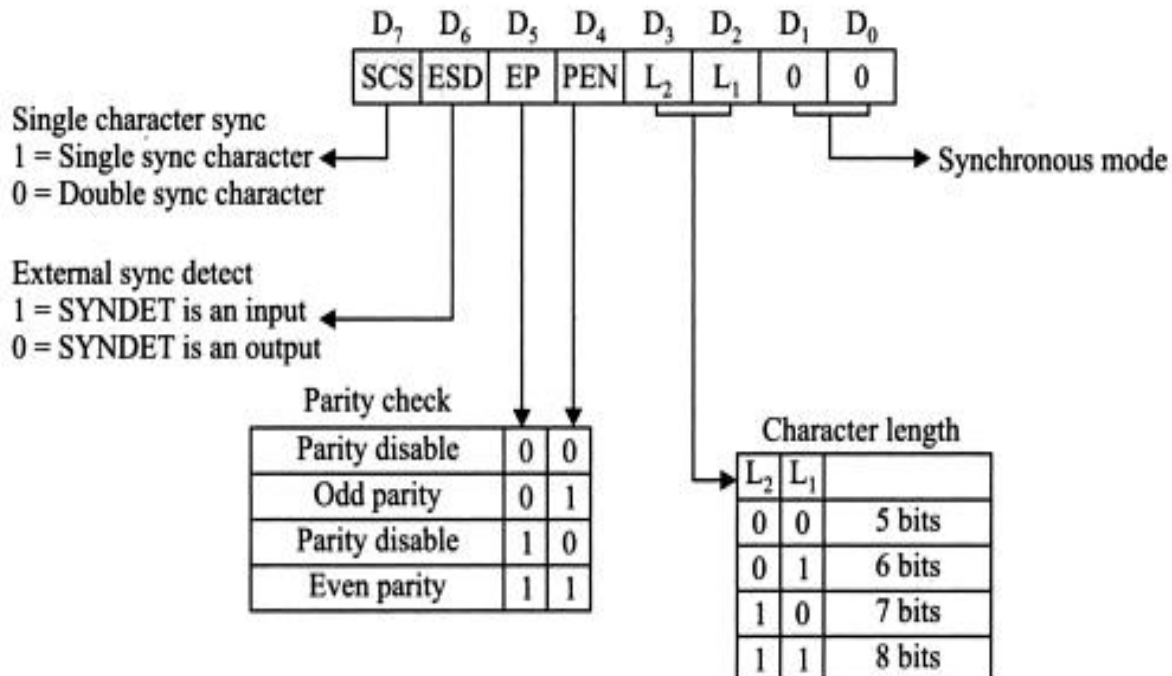
**Figure: Mode instruction format in Asynchronous mode.**

**D<sub>0</sub> – D<sub>1</sub>** determines whether the USART has to operate in synchronous (00) or asynchronous mode (01, 10, and 11). In asynchronous mode this field determines the baud rate based on clock.

**D<sub>2</sub> – D<sub>3</sub>** determines the number of data bits in one character. With this field we can set character length ranging from 5 to 8 bits.

**D<sub>4</sub> – D<sub>5</sub>** controls the parity generation. The parity bit is added to data bits only if parity is enabled.

**D<sub>6</sub> – D<sub>7</sub>** these bits can perform different functions depending on mode of transmission. In asynchronous mode, it controls number of stop bits to be transmitted with the character. In synchronous mode, it controls the synchronizing process i.e. to transmit single or two synchronizing characters.



**Figure: Mode instruction format in synchronous mode.**

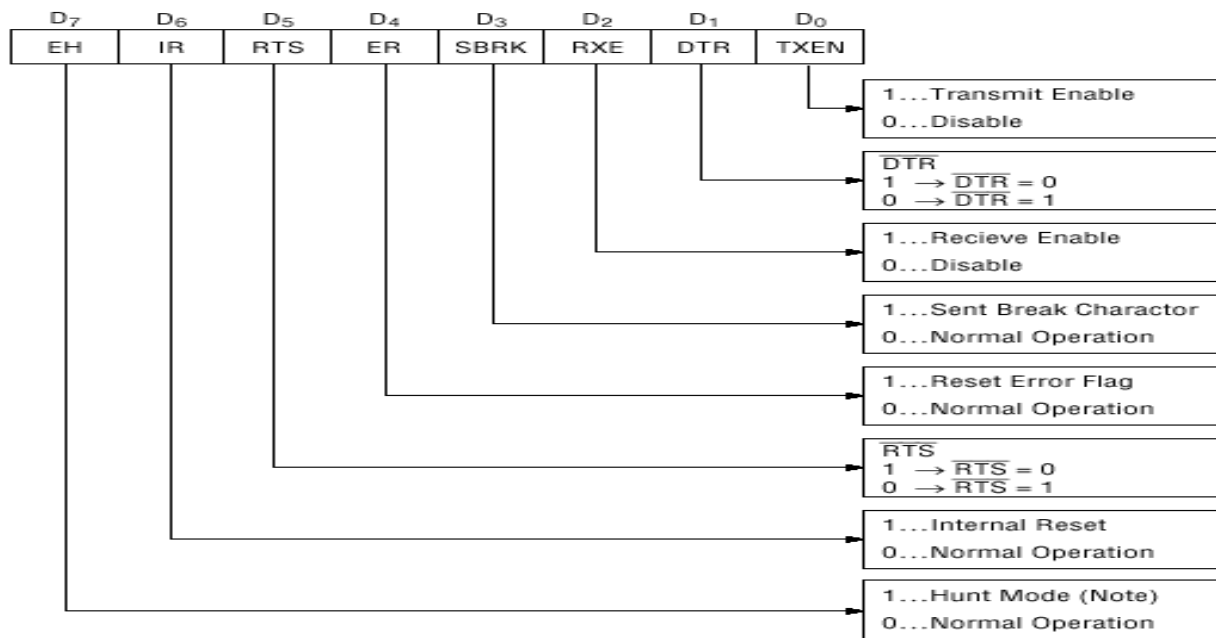
**D<sub>7</sub>** – determines whether single ( $D_7=1$ ) or double ( $D_7=0$ ) synchronous characters.

**D<sub>6</sub>** – determines whether the synchronous detection circuitry is internal ( $D_6=0$ ) or external ( $D_6=1$ ).

**D<sub>5</sub> – D<sub>2</sub>** are same as in asynchronous mode format.

**D<sub>1</sub> – D<sub>0</sub>** are always '0' representing synchronous mode.

**Command instruction:** After selecting the mode, command character should be transmitted to USART. It is used to enable transmit / receive, error reset and modem control.

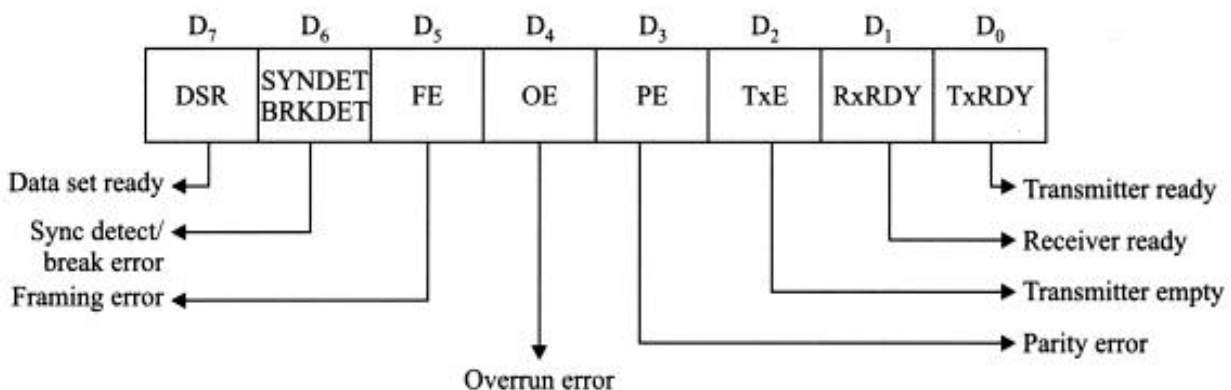


**Note:** Seach mode for synchronous characters in synchronous mode.

**Figure: Command word format**

**Status word of 8251A:** It is necessary for CPU to monitor the status of transmitter and receiver to know any error has occurred.

All the error flags are reset by ER bit (error reset) in the command instruction.



**Figure: Status word format**

## Need for DMA, DMA data transfer Method:

*Direct Memory Access is a method of transferring data between peripherals and memory without using the CPU.*

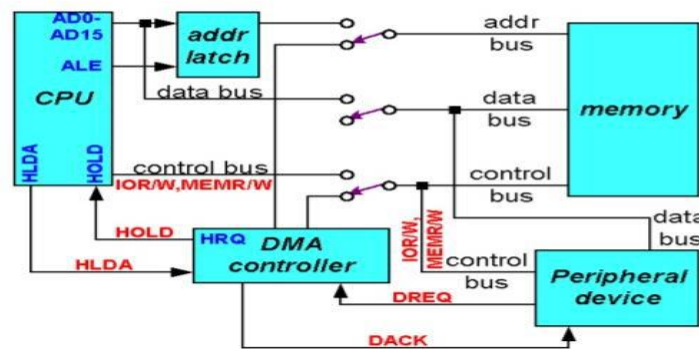
In some cases the CPU may not be fast enough to keep up with the peripheral or it may be desirable to allow the CPU to do other useful work while the I/O is in progress. In this case a special-purpose processor called a DMA controller (DMAC) can be used to transfer data between memory and I/O devices. The DMA controller

periodically takes over control of the system bus from the CPU, and, like the CPU, generates address, data and control signals to transfer data between memory and I/O devices.

DMA - Direct memory access or autonomous transfer

- Direct Memory Access (DMA) is a method of allowing data to be moved from one location to another in a computer without intervention from the central processor (CPU). It is also a fast way of transferring data within (and sometimes between) computer.
- The DMA I/O technique provides direct access to the memory while the microprocessor is temporarily disabled. The DMA controller temporarily borrows the address bus, data bus and control bus from the microprocessor and transfers the data directly from the external devices to a series of memory locations (and vice versa).

Basic DMA Operation: Direct memory accesses normally occur between an I/O device and memory without the use of the microprocessor. Two control signals are used to request and acknowledge a direct memory access (DMA) transfer in the microprocessor-based system.



## DMA OPERATION

- The HOLD signal as an input (to the processor) is used to request a DMA action.
- The HLDA signal as an output that acknowledges the DMA action. When the processor recognizes the hold, it stops its execution and enters hold cycles. HLDA becomes active to indicate that the processor has placed its buses at high- impedance state.
- A DMA read transfers data from the memory to the I/O device. A DMA write transfers data from an I/O device to memory. The system contains separate memory and I/O control signals. Hence the Memory & the I/O are controlled simultaneously.
- The DMA controller provides memory with its address, and the controller signal selects the I/O device during the transfer.
- Data transfer speed is determined by speed of the memory device or a DMA controller.
- In bus master mode, the DMA controller acquires the system bus from the CPU to perform the DMA transfers. Because the CPU releases the system bus for the duration of the data transfer, the process is sometimes referred to as **cycle stealing**.



- In bus slave mode, the DMA controller is accessed by the CPU, which programs the DMA controller's internal registers to set up DMA transfers. The internal register consist of source and destination address registers and transfer count registers for each DMA channel, as well as control and status registers for initiating, monitoring and sustaining the operation of the DMA controller.

## 8257 DMA Controller:

### Features of 8257:

1. It is a 4-channel Direct Memory Access (DMAC) interface IC which allows data transfer between memory and up to 4 I/O devices, bypassing CPU.
2. A maximum of 16 KB of data ( $= 2^{14}$ ) can be transferred by this IC sequentially at a time. When a DMA request comes from a peripheral, the DMAC 8257, via its HRQ (Hold Request) pin, requests the CPU on its HOLD pin. CPU then acknowledges this request via its HLDA pin which goes to HLDA pin of 8257. After this, DMAC generates the required MEMR, MEMW, I/OR, I/OW signals.
3. Initialization of the DMAC is done under program control for each channel. The parameters which need to be initialized for each channel are starting address, number of bytes of data to be transferred, mode of operation, etc.
4. DMAC can be operated in three modes: (a) **DMA Read** (reading from memory, writing into peripheral), (b) **DMA Write** (writing into memory, reading from peripheral), (c) **DMA verify**.
5. Priority for each of the 4 channels can be set in (a) **fixed priority**, (b) **rotating priority**.
6. A Terminal Count Register exists for each of 4 channels. The number of bytes of data to be transferred is stored in the D13–D0 positions of the 16-bit Terminal Count Register. On completion of data transfer, the Terminal Count (TC) pin goes high.
7. When the CPU is in control of its buses (address bus, data bus and control bus), it acts as master and DMA controller acts as the slave. When DMA controller takes control of the buses, it becomes the master and CPU becomes the slave.

### Description of pins:

**D0-D7:** It is a bidirectional ,tri state ,Buffered ,Multiplexed data (D0-D7)and (A8-A15). In the slave mode it is a bidirectional (Data is moving). In the Master mode it is a unidirectional (Address is moving).

**IOR:** It is active low, tristate, buffered, Bidirectional lines. In the slave mode it functions as an input line. IOR signal is generated by microprocessor to read the contents of 8257 registers. In the master mode it functions as an output line. IOR signal is generated by 8257 during write cycle.

**IOW:** It is active low, tristate, buffered, Bidirectional lines. In the slave mode it functions as an input line. IOW signal is generated by microprocessor to write the contents of 8257 registers. In the master mode it functions as an output line. IOR signal is generated by 8257 during read cycle.

**CLK:** It is the input line, connected with TTL clock generator. This signal is ignored in slave mode.

**RESET:** It is used to clear mode set registers and status registers.

A0-A3: These are the tristate, buffer, bidirectional address lines. In slave mode, these lines are used as address inputs lines and internally decoded to access the internal registers. In master mode, these lines are used as address outputs lines, A0-A3 bits of memory address on the lines.

CS: It is active low, Chip select input line. In the slave mode, it is used to select the chip. In the master mode, it is ignored.

A4-A7: These are the tristate, buffer, output address lines. In slave mode, these lines are used as address outputs lines.

In master mode, these lines are used as address outputs lines, A0-A3 bits of memory address on the lines.

READY: It is a asynchronous input line. In master mode, When ready is high it is received the signal. When ready is low, it adds wait state between S1 and S3. In slave mode, this signal is ignored.

HRQ: It is used to receiving the hold request signal from the output device.

HLDA: It is acknowledgment signal from microprocessor.

MEMR: It is active low, tristate, Buffered control output line. In slave mode, it is tristated. In master mode, it activated during DMA read cycle.

MEMW: It is active low, tristate, Buffered control input line. In slave mode, it is tristated. In master mode, it activated during DMA write cycle.

AEN (Address enable): It is a control output line. In master mode, it is high. In slave mode, it is low. Used it isolate the system address, data and control lines.

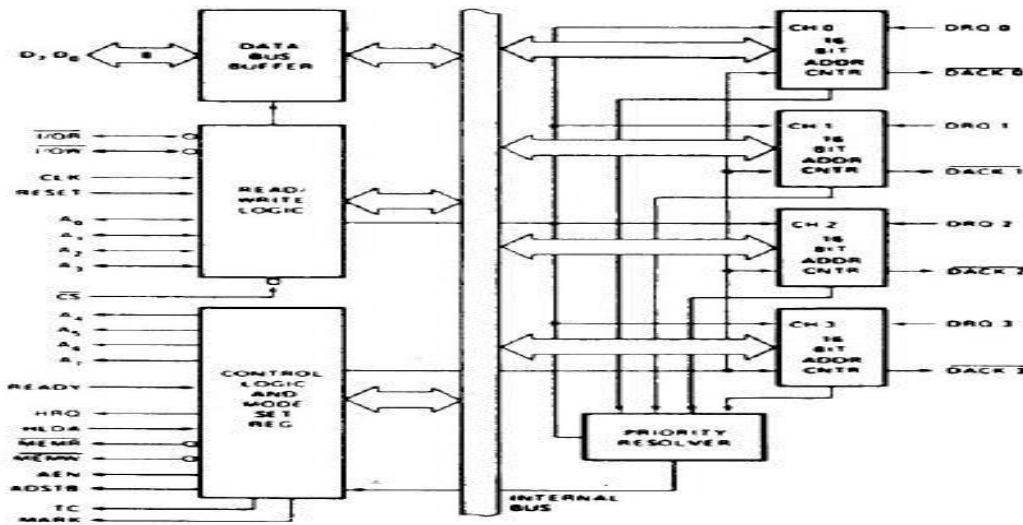
ADSTB: (Address Strobe) It is a control output line. Used to split data and address line. It is working in master mode only. In slave mode it is ignore.

TC (Terminal Count): It is a status of output line. It is activated in master mode only. It is high, it selected the peripheral. It is low, it free and looking for a new peripheral.

MARK: It is a modulo 128 MARK output line. It is activated in master mode only. It goes high, after transferring every 128 bytes of data block.

DRQ0-DRQ3(DMA Request): These are the asynchronous peripheral request input signal. The request signals are generated by external peripheral device.

DACK0-DACK3: These are the active low DMA acknowledge output lines. Low level indicates that, peripheral is selected for giving the information (DMA cycle). In master mode it is used for chip select.



**BLOCK DIAGRAM OF 8257**

It contains five main Blocks:

1. Data bus buffer
2. Read/Control logic
3. Control logic block
4. Priority resolver
5. DMA channels.

**DATA BUS BUFFER:** It contains tristate, 8 bit bi-directional buffer. Slave mode, it transfer data between microprocessor and internal data bus. Master mode, the outputs A8-A15 bits of memory address on data lines (Unidirectional).

**READ/CONTROL LOGIC:** It controls all internal Read/Write operation. Slave mode, it accepts address bits and control signal from microprocessor. Master mode, it generate address bits and control signal.

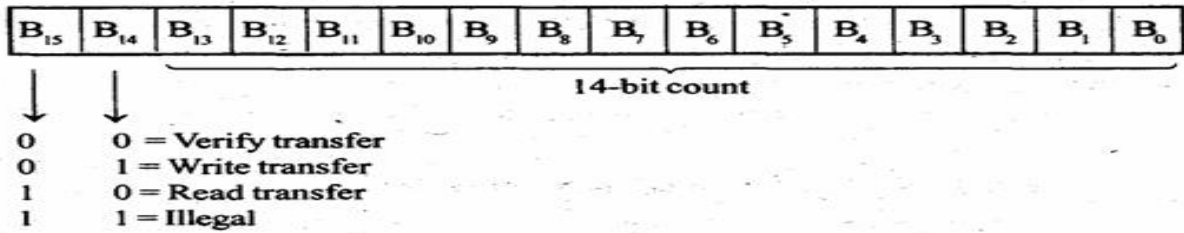
**Control logic block:** It contains, Control logic, Mode set register and Status Register.

**CONTROL LOGIC:** Master mode, It control the sequence of DMA operation during all DMA cycles. It generates address and control signals. It increments 16 bit address and decrement 14 bit counter registers. It activates a HRQ signal on DMA channel Request. Slave, mode it is disabled.

**Register organization of 8257:** The 8257 performs the DMA operation over four independent DMA channels. Each DMA channel has a pair of two 16-bit registers i.e. address register and terminal count register. Also there are two common registers for all the channels are Mode set register and Status register. The CPU selects one of these registers using address lines A<sub>0</sub> to A<sub>3</sub>.

**DMA address register:** The function of this register is to store the address of the starting memory location, which will be accessed by the DMA channel.

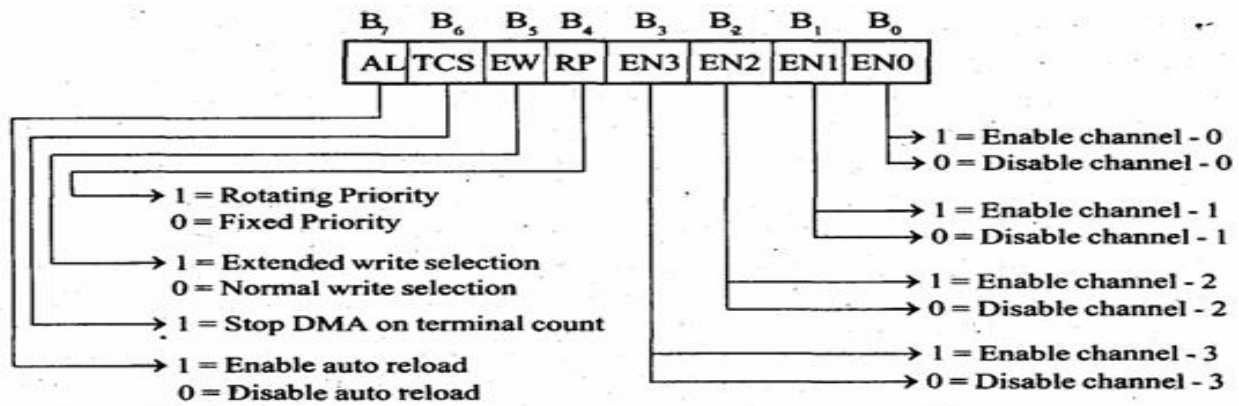
**Terminal Count register:** The function of this register is to indicate the end of the data transfer after required number of DMA cycles. The low order 14-bits of the terminal count register are initialized with the number of DMA cycles minus one. After each DMA cycle this register content can be decremented by 1 and becomes zero after all DMA cycles are over. The bits 14 and 15 indicate the type of the DMA operation.



## Terminal Count Register

**Mode Set Register:** The function of mode set register is to enable the DMA channels individually and also to set the various modes of operation.

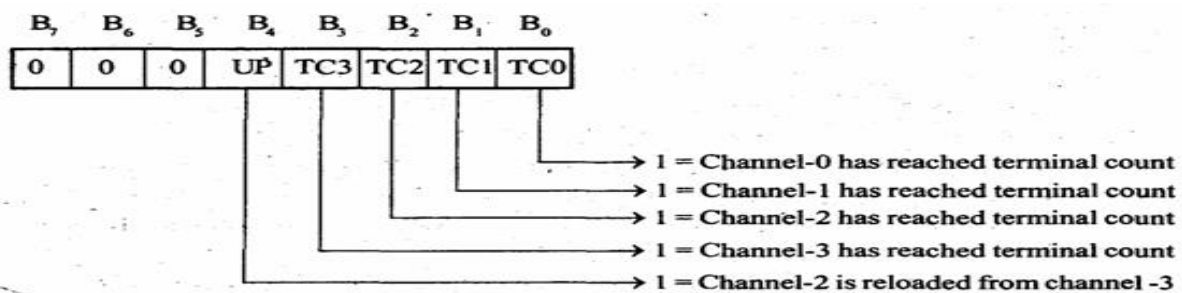
**Note:** A DMA channel should not be enabled till the DMA address and terminal count registers have valid data.



## Mode Set Register

The bits B<sub>0</sub> – B<sub>3</sub> enable one of the four DMA channels of 8257. The extended write bit extends the duration of MEMW and IOW signals. B<sub>4</sub> provides rotating and normal priority. TC stop bit is set, the selected channel is disabled, so that it can be prevented for further DMA cycle on the channel. If auto load bit is set enables the same channel for the repeat block chain operations without software intervention between two successive blocks.

**Status Register:** The lower order 4-bits are used to indicate the terminal count condition for the four individual channels. These bits remain set till either the status is read by CPU or reset by 8257. The update flag can only be cleared by resetting 8257 or by resetting the auto load bit of the mode set register. If update flag register is set, the contents of channel 3 registers are reloaded to the corresponding registers of channel 2.



## Status Register



**8279 Programmable Keyboard/Display Controller and Interfacing****The Keyboard/Display Controller 8279**

Intel's 8279 is a general purpose Keyboard Display controller that simultaneously drives the display of a system and interfaces a Keyboard with the CPU.

The Keyboard Display interface scans the Keyboard to identify if any key has been pressed and sends the code of the pressed key to the CPU. It also transmits the data received from the CPU, to the display device. Both of these functions are performed by the controller in repetitive fashion without involving the CPU.

The Keyboard is interfaced either in the interrupt or the polled mode. In the interrupt mode, the processor is requested service only if any key is pressed, otherwise the CPU can proceed with its main task.

In the polled mode, the CPU periodically reads an internal flag of 8279 to check for a key pressure. The Keyboard section can interface an array of a maximum of 64 keys with the CPU.

The Keyboard entries (key codes) are debounced and stored in an 8-byte FIFO RAM, that is further accessed by the CPU to read the key codes. If more than eight characters are entered in the FIFO (i.e. more than eight keys are pressed), before any FIFO read operation, the overrun status is set. If a FIFO contains a valid key entry, the CPU is interrupted (in interrupt mode) or the CPU checks the status (in polling) to read the entry.

Once the CPU reads a key entry, the FIFO is updated, i.e. the key entry is pushed out of the FIFO to generate space for new entries. The 8279 normally provides a maximum of sixteen 7-seg display interface with CPU. It contains a 16-byte display RAM that can be used either as an integrated block of 16x8-bits or two 16x4-bit block of RAM. The data entry to RAM block is controlled by CPU using the command words of the 8279.

**Architecture of 8279:**

The Keyboard display controller chip 8279 provides

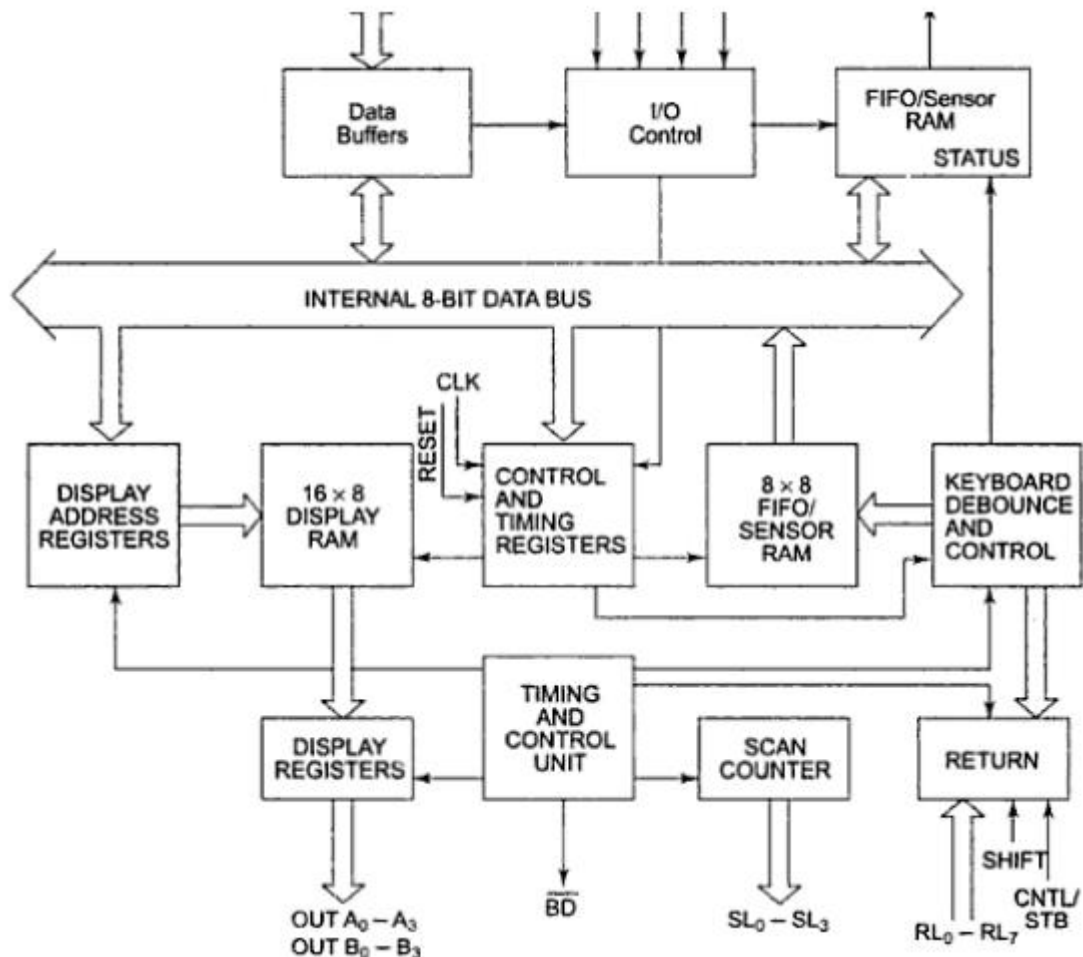
1. A set of four scan lines and eight return lines for interfacing keyboards.
2. A set of eight output lines for interfacing display.

**I/O Control and Data Buffer**

The I/O control section controls the flow of data to/from the 8279. The data buffer interfaces the external bus of the system with internal bus of 8279 and the I/O section is enabled only if D is low.

**8279 Internal Architecture:**





The pin A<sub>0</sub>, RD and WR select the command, status or data read/write operations carried out by the CPU with 8279.

## Control and Timing Register and Timing Control

These registers store the keyboard and display modes and other operating conditions programmed by CPU. The registers are written with A<sub>0</sub>=1 and WR =0. The timing and control unit controls the basic timings for the operation of the circuit. Scan Counter divide down the operating frequency of 8279 to derive scan keyboard and scan display frequencies.

## Scan Counter

The Scan Counter has two modes to scan the key matrix and refresh the display. In the Encoded mode, the counter provides a binary count that is to be externally decoded to provide the scan lines for keyboard and display (four externally decoded scan lines may drive up to 16 displays).

In the decoded scan mode, the counter internally decodes the least significant 2 bits and provides a decoded 1 out of 4 scan on SL<sub>0</sub>-SL<sub>3</sub> (four internally decoded scan lines may drive up to 4 Displays). The Keyboard and Display both are in the same mode at a time.

## Return Buffers and Keyboard Debounce and Control

This section scans for a Key closure row-wise. If it is detected, the Keyboard debounce unit debounces the key entry (i.e. wait for 10 ms). After the debounce period, if the key continues to be detected. The code of the Key is directly transferred to the sensor RAM along with SHIFT and CONTROL key status.

## **FIFO/Sensor RAM and Status Logic**

In Keyboard or strobed input mode, this block acts as 8-byte first-in-first-out (FIFO) RAM. Each key code of the pressed key is entered in the order of the entry, and in the meantime, read by the CPU, till the RAM becomes empty. The status logic generates an interrupt request after each FIFO read operation till the FIFO is empty.

In scanned sensor matrix mode, this unit acts as sensor RAM. Each row of the sensor RAM is loaded with the status of the corresponding row of sensors in the matrix. If a sensor changes its state, the IRQ line goes high to interrupt the CPU.

## **Display Address Registers and Display RAM.**

The Display address registers hold the addresses of the word currently being written or read by the CPU to or from the display RAM. The contents of the registers are automatically updated by 8279 to accept the next data entry by CPU. The 16-byte display RAM contains the 16-byte of data to be displayed on the sixteen 7-seg displays in the encoded scan mode.

## **SL0-SL3 – Scan Lines:**

These lines are used to scan the keyboard matrix and display digits. These lines can be programmed as encoded or decoded, using the mode control register.

## **RL0-RL7 – Return Lines:**

These are the input lines which are connected to one terminal of keys, while the other terminals of the keys are connected to the decoded scan lines. These are normally high, but pulled low when a key is pressed.

## **SHIFT:**

The status of the Shift input line is stored along with each key code in FIFO in the scanned keyboard mode. Till it is pulled low with a key closure it is pulled up internally to keep it high.

## **CNTL/STB-CONTROL/STROBED I/P Mode:**

In the Keyboard mode, this line is used as a control input and stored in FIFO on a key closure. The line is a strobe line that enters the data into FIFO RAM, in the strobed input mode. It has an internal pull up. The line is pulled down with a Key closure.

## **BD – Blank Display :**

This output pin is used to blank the display during digit switching or by a blanking command.

## **OUTA0 – OUTA3 and OUTB0 – OUTB3:**

These are the output ports for two 16x4 (or one 16 x 8) internal display refresh registers. The data from these lines is synchronized with the scan lines to scan the display and keyboard. The two 4-bit ports may also be used as one 8-bit port.

## Modes of Operation of 8279

The Modes of operation of 8279 are

- i. Input (Keyboard) modes
- ii. Output (Display) modes

### Input (Keyboard) modes:

8279 provides three input modes, they are:

#### 1. Scanned Keyboard Mode:

This mode allows a key matrix to be interfaced using either encoded or decoded scans. In the encoded scan, an 8 x 8 keyboard or in decoded scan, a 4 x 8 Keyboard can be interfaced. The code of key pressed with SHIFT and CONTROL status is stored into the FIFO RAM.

#### 2. Scanned Sensor Matrix:

In this mode, a sensor array can be interfaced with 8279 using either encoder or decoder scans. With encoder scan 8 x 8 sensor matrix or with decoder scan 4 x 8 sensor matrix can be interfaced. The sensor codes are stored in the CPU addressable sensor RAM.

#### 3. Strobed Input:

In this mode, if the control line goes low, the data on return lines is stored in the FIFO byte by byte.

### Output (Display) Modes:

8279 provides two output modes for selecting the display options.

#### 1. Display Scan:

In this mode, 8279 provides 8 or 16 character multiplexed displays those can be organized as dual 4-bit or single 8-bit display units.

#### 2. Display Entry:

The Display data is entered for display either from the right side or from the left side.

## Details of Modes of Operation

### Keyboard Modes

#### 1. Scanned Keyboard Mode with 2 Key Lockout

In this mode of operation, when a key is pressed, a debounce logic comes into operation. The Key code of the identified key is entered into the FIFO with SHIFT and CNTL status, provided the FIFO is not full.

#### 2. Scanned Keyboard with N-key Rollover

In this mode, each key depression is treated independently. When a key is pressed, the debounce circuit waits for 2 keyboard scans and then checks whether the key is still depressed. If it is still depressed, the code is entered in

FIFO RAM. Any number of keys can be pressed simultaneously and recognized in the order, the Keyboard scan record them.

### 3. Scanned Keyboard Special Error Mode

This mode is valid only under the N-Key rollover mode. This mode is programmed using *end interrupt/error mode set* command. If during a single debounce period (two Keyboard scan) two keys are found pressed, this is considered a simultaneous depression and an error flag is set. This flag, if set, prevents further writing in FIFO but allows generation of further interrupts to the CPU for FIFO read.

### 3. Sensor Matrix Mode

In the Sensor Matrix mode, the debounce logic is inhibited the 8-byte memory matrix. The status of the sensor switch matrix is fed directly to sensor RAM matrix. Thus the sensor RAM bits contains the row-wise and column-wise status of the sensors in the sensor matrix.

### Display Modes

There are various options of data display. The first one is known as left entry mode or type writer mode. Since in a type writer the first character typed appears at the left-most position, while the subsequent characters appears successively to the right of the first one.

The other display format is known as right entry mode, or calculator mode, since the calculator the first character entered appears at the right-most position and this character is shifted one position left when the next character is entered.

#### 1. Left Entry Mode

In the Left entry mode, the data is entered from the left side of the display unit. Address 0 of the display RAM contains the leftmost display character and address 15 of the RAM contains the rightmost display character.

#### 2. Right Entry Mode

In the right entry mode, the first entry to be displayed is entered on the rightmost display. The next entry is also placed in the right most display but after the previous display is shifted left by one display position.

### Command Words of 8279

All the Command words or status words are written or read with  $A_0 = 1$  and  $CS = 0$  to or from 8279.

#### a. Keyboard Display mode set

The format of the command word to select different modes of operation of 8279 is given below with its bit definitions.

D7	D6	D5	D4	D3	D2	D1	D0	A0
0	0	0	D	D	K	K	K	1

<i>D</i>	<i>D</i>	<i>Display modes</i>
0	0	Eight 8-bit character Left entry
0	1	Sixteen 8-bit character Left entry ( Default after reset)
1	0	Eight 8-bit character Right entry
1	1	Sixteen 8-bit character Right entry

<i>K</i>	<i>K</i>	<i>K</i>	<i>Keyboard modes</i>
0	0	0	Encoded scan, 2 key lockout ( Default after reset )
0	0	1	Decoded scan, 2 key lockout
0	1	0	Encoded scan N-Key roll over
0	1	1	Decoded scan N-Key roll over
1	0	0	Encoded scan sensor matrix
1	0	1	Decoded scan sensor matrix
1	1	0	Strobed Input Encoded scan
1	1	1	Strobed Input Decoded scan

## b. Programmable Clock

The clock for operation of 8279 is obtained by dividing the external clock input signal by a programmable constant called prescaler.

PPPPP is a 5-bit binary constant. The input frequency is divided by a decimal constant ranging from 2 to 31, decided by the bits of an internal prescaler, PPPPP.

<i>D<sub>7</sub></i>	<i>D<sub>6</sub></i>	<i>D<sub>5</sub></i>	<i>D<sub>4</sub></i>	<i>D<sub>3</sub></i>	<i>D<sub>2</sub></i>	<i>D<sub>1</sub></i>	<i>D<sub>0</sub></i>	<i>A<sub>0</sub></i>
0	0	1	P	P	P	P	P	1

## c. Read FIFO/Sensor RAM

The format of this command is given as shown below

<i>D<sub>7</sub></i>	<i>D<sub>6</sub></i>	<i>D<sub>5</sub></i>	<i>D<sub>4</sub></i>	<i>D<sub>3</sub></i>	<i>D<sub>2</sub></i>	<i>D<sub>1</sub></i>	<i>D<sub>0</sub></i>	<i>A<sub>0</sub></i>
0	1	0	AI	X	A	A	A	1

X - don't care

AI - Auto increment flag

AAA - Address pointer to 8 bit FIFO RAM

This word is written to set up 8279 for reading FIFO/Sensor RAM. In scanned keyboard mode, AI and AAA bits are of no use. The 8279 will automatically drive data bus for each subsequent read, in the same sequence, in which the data was entered.



## d. Read Display RAM

This command enables a programmer to read the display RAM data

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>	A <sub>0</sub>
0	1	1	AI	A	A	A	A	1

The CPU writes this command word to 8279 to prepare it for display RAM read operation. AI is auto incremented flag and AAAA, the 4-bit address, points to the 16-byte display RAM that is to be read. If AI = 1, the address will be automatically, incremented after each read or write to the display RAM.

## e. Write Display RAM

The format of this command is given as shown below

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>	A <sub>0</sub>
1	0	0	AI	A	A	A	A	1

AI - Auto increment flag

AAAA - 4-bit address for 16-bit display RAM to be written

Other details of this command are similar to the 'Read Display RAM Command'.

## f. Display Write Inhibit/Blanking

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>	A <sub>0</sub>
1	0	1	X	IW	IW	BL	BL	1

Output nibbles →                      A                      B                      A                      B

The IW (Inhibit write flag) bits are used to mask the individual nibble Here D<sub>0</sub> and D<sub>2</sub> corresponds to OUTB<sub>0</sub> – OUTB<sub>3</sub> while D<sub>1</sub> and D<sub>3</sub> corresponds to OUTA<sub>0</sub>-OUTA<sub>3</sub> for blanking and masking respectively.

## g. Clear Display RAM

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>	A <sub>0</sub>
1	1	0	CD <sub>2</sub>	CD <sub>1</sub>	CD <sub>0</sub>	CF	CA	1

The CD<sub>2</sub>, CD<sub>1</sub>, CD<sub>0</sub> is a selectable blanking code to clear all the rows of the display RAM as given below. The characters A and B represents the output nibbles.

CD	CD1	CD0	
1	0	x	All Zeros (x don't care) AB = 00
1	1	0	A3-A0 = 2(0010) and B3-B0 = 00(0000)
1	1	1	All ones (AB = FF), i.e. clear RAM

Here, CA represents clear All and CF represents Clear FIFO RAM

## End Interrupt/Error Mode Set

D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>	A <sub>0</sub>
1	1	1	E	x	x	x	x	1

x—do not care

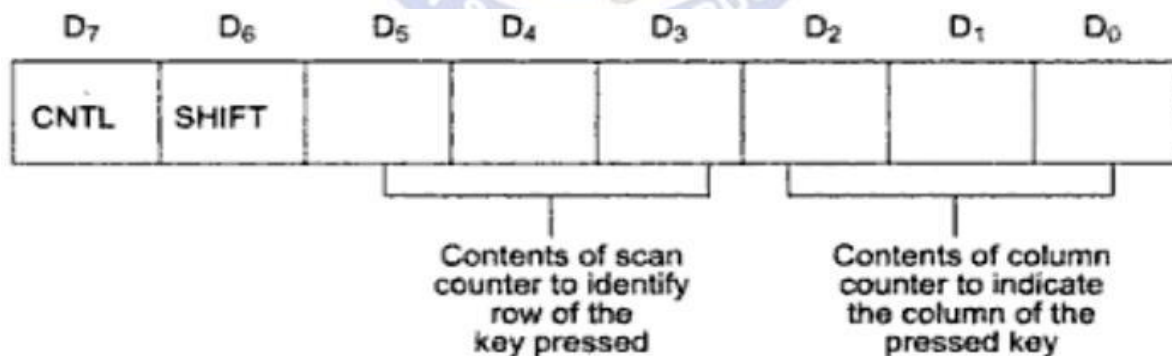
For the sensor matrix mode, this command lowers the IRQ line and enables further writing into the RAM. Otherwise, if a change in sensor value is detected, IRQ goes high that inhibits writing in the sensor RAM.

## Key-code and status Data Formats

This briefly describes the formats of the Key-code/Sensor data in their respective modes of operation and the FIFO Status Word formats of 8279.

### Key-code Data Formats:

After a valid Key closure, the key code is entered as a byte code into the FIFO RAM, in the following format, in scanned keyboard mode. The Key code format contains 3-bit contents of the internal row counter, 3-bit contents of the column counter and status of the SHIFT and CNTL Keys. The data format of the Keycode in scanned keyboard mode is given below.



In the sensor matrix mode, the data from the return lines is directly entered into an appropriate row of sensor RAM that identifies the row of the sensor that changes its status. The SHIFT and CNTL Keys are ignored in this mode. RL bits represent the return lines.

R<sub>n</sub> represents the sensor RAM row number that is equal to the row number of the sensor array in which the status change was detected. Data Format of the sensor code in sensor matrix mode.

R <sub>n</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
	RL <sub>7</sub>	RL <sub>6</sub>	RL <sub>5</sub>	RL <sub>4</sub>	RL <sub>3</sub>	RL <sub>2</sub>	RL <sub>1</sub>	RL <sub>0</sub>

## FIFO Status Word:

The FIFO status word is used in keyboard and strobed input mode to indicate the error. Overrun error occurs, when an already full FIFO is attempted an entry, under run error occurs when an empty FIFO read is attempted. FIFO status word also has a bit to show the unavailability of FIFO RAM because of the ongoing clearing operation.

In sensor matrix mode, a bit is reserved to show that at least one sensor closure indication is stored in the RAM, The S/E bit shows the simultaneous multiple closure error in special error mode. In sensor matrix mode, a bit is reserved to show that at least one sensor closure indication is stored in the RAM, The S/E bit shows the simultaneous multiple closure error in special error mode. The FIFO status word format is as shown below:

