Q.2 a. Draw and explain the V-I characteristics (forward and reverse biasing) of a pn junction. (8)

Answer: Please refer Page No 14-17 I.J.Nagrath Electronic Devices and Circuits 5th Edition.

b. Draw and explain the I-V characteristics of a Zener diode. What are the two breakdown mechanisms in a Zener diode? (8)

Answer: Please refer Page No 23 I.J.Nagrath Electronic Devices and Circuits 5th Edition.

Q.3 a. A full-wave rectifier with a center-tapped transformer supplies a dc current of 100mA to a load resistance of R=20Ω. The secondary resistance of the transformer is 1Ω. Each diode has a forward resistance of 0.5Ω. Determine the following:

(i) RMS Value of the signal voltage across each half of the secondary.

(ii) DC power supplied to the load.

(iii) PIV rating for each diode.

(iv) AC power input to the rectifier.

(v) Conversion efficiency (8)

Answer: Please refer Page No 44-45 I.J. Nagrath Electronic Devices and Circuits 5th Edition.

b. Draw the positive and negative voltage clipper circuits. Explain its working along with the waveforms. (8)

Answer: Please refer Page No 56-57 I.J.Nagrath Electronic Devices and Circuits 5th Edition.

Q.4 a. Explain the operation of NPN transistor with neat diagrams. (8)

Answer: Please refer Page No 80-83 I.J.Nagrath Electronic Devices and Circuits 5th Edition.

b. Fig.1 shows the voltage divider bias method. Draw the DC load line and determine the operating point. Assume transistor to be Silicon. (8)

Answer:

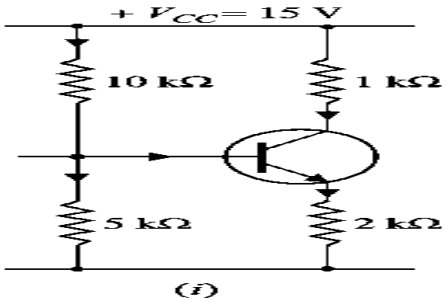


Fig. 1

Solution.

d.c. load line. The collector-emitter voltage V_{CE} is given by :

$$V_{CE} = V_{CC} - I_C (R_C + R_E)$$

When $I_C = 0$, $V_{CE} = V_{CC} = 15$V. This locates the first point B ($OB = 15$V) of the load line on the collector-emitter voltage axis.

$$\text{When } V_{CE} = 0, \quad I_C = \frac{V_{CC}}{R_C + R_E} = \frac{15 \text{ V}}{(1 + 2)\text{ k}\Omega} = 5 \text{ mA}$$

This locates the second point A ($OA = 5$ mA) of the load line on the collector current axis. By joining points A and B, the d.c. load line AB is constructed as shown in Fig. 9.25 (*ii*).

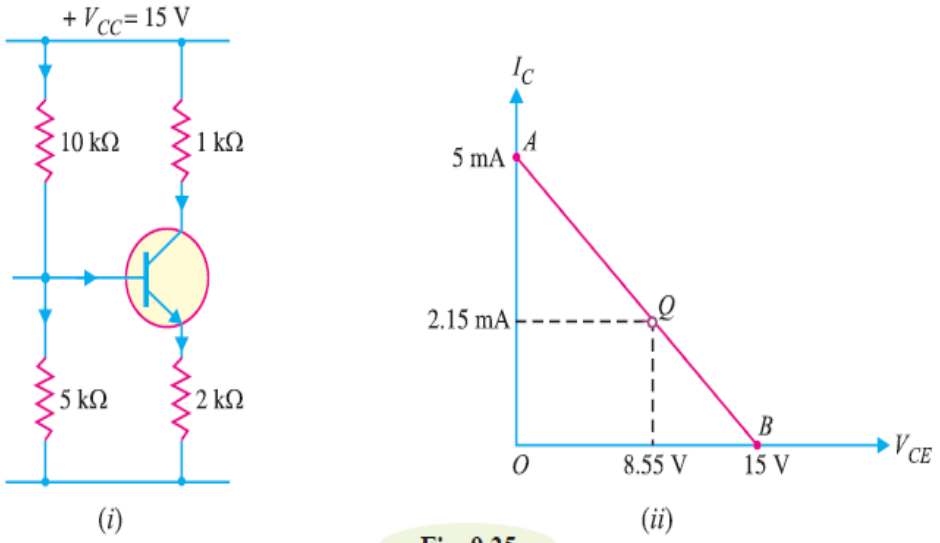


Fig. 9.25

Operating point. For silicon transistor,

$$V_{BE} = 0.7 \text{ V}$$

Voltage across 5 kΩ is

$$V_2 = \frac{V_{CC}}{10 + 5} \times 5 = \frac{15 \times 5}{10 + 5} = 5 \text{ V}$$

$$\therefore \quad \text{Emitter current, } I_E = \frac{V_2 - V_{BE}}{R_E} = \frac{5 - 0.7}{2\text{ k}\Omega} = \frac{4.3 \text{ V}}{2\text{ k}\Omega} = 2.15 \text{ mA}$$

$\therefore$ Collector current is

$$I_C \simeq I_E = 2.15 \text{ mA}$$

$$\text{Collector-emitter voltage, } V_{CE} = V_{CC} - I_C (R_C + R_E)$$
$$= 15 - 2.15 \text{ mA} \times 3 \text{ k}\Omega = 15 - 6.45 = 8.55 \text{ V}$$

$\therefore$ Operating point is **8.55 V, 2.15 mA.**

Fig.9.25 (*ii*) shows the operating point Q on the load line. Its co-ordinates are $I_C = 2.15$ mA, $V_{CE} = 8.55$ V.

Q.5 a. Draw the circuit of single stage CE amplifier and explain the function of bypass capacitor and coupling capacitors. (8)

Answer:

It is important to note that a transistor can accomplish faithful amplification only if proper associated circuitry is used with it. Fig. 10.3 shows a practical single stage transistor amplifier. The various circuit elements and their functions are described below :

***(i)* Biasing circuit.** The resistances R_1, R_2 and R_E form the biasing and stabilisation circuit. The biasing circuit must establish a proper operating point otherwise a part of the negative half-cycle of the signal may be cut off in the output.

***(ii)* Input capacitor C_{in}.** An electrolytic capacitor C_{in} ($\simeq 10\ \mu F$) is used to couple the signal to the base of the transistor. If it is not used, the signal source resistance will come across R_2 and thus change the bias. The capacitor C_{in} allows only a.c. signal to flow but isolates the signal source from R_2.*

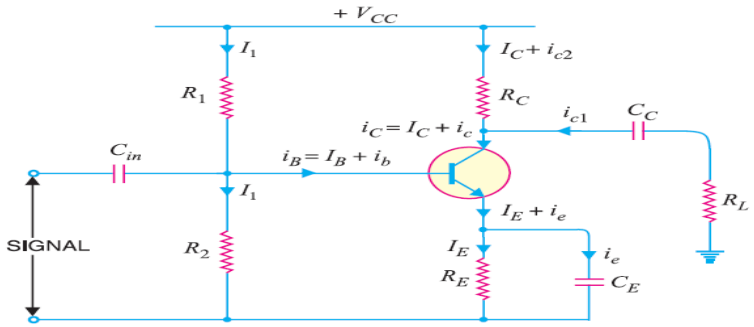


Fig. 10.3

***(iii)* Emitter bypass capacitor C_E.** An emitter bypass capacitor C_E ($\simeq 100\mu F$) is used in parallel with R_E to provide a low reactance path to the amplified a.c. signal. If it is not used, then amplified a.c. signal flowing through R_E will cause a voltage drop across it, thereby reducing the output voltage.

***(iv)* Coupling capacitor C_C.** The coupling capacitor C_C ($\simeq 10\mu F$) couples one stage of amplification to the next stage. If it is not used, the bias conditions of the next stage will be drastically changed due to the shunting effect of R_C. This is because R_C will come in parallel with the upper resistance R_1 of the biasing network of the next stage, thereby altering the biasing conditions of the latter. In short, the coupling capacitor C_C isolates the d.c. of one stage from the next stage, but allows the passage of a.c. signal.

* It may be noted that a capacitor offers infinite reactance to d.c. and blocks it completely whereas it allows a.c. to pass through it.

Various circuit currents. It is useful to mention the various currents in the complete amplifier circuit. These are shown in the circuit of Fig. 10.3.

***(i)* Base current.** When no signal is applied in the base circuit, d.c. base current I_B flows due to biasing circuit. When a.c. signal is applied, a.c. base current i_b also flows. Therefore, with the application of signal, total base current i_B is given by:

$$i_B = I_B + i_b$$

***(ii)* Collector current.** When no signal is applied, a d.c. collector current I_C flows due to biasing circuit. When a.c. signal is applied, a.c. collector current i_c also flows. Therefore, the total collector current i_C is given by:

$$i_C = I_C + i_c$$

where

$$I_C = \beta I_B = \text{zero signal collector current}$$

$$i_c = \beta i_b = \text{collector current due to signal.}$$

***(iii)* Emitter current.** When no signal is applied, a d.c. emitter current I_E flows. With the application of signal, total emitter current i_E is given by :

$$i_E = I_E + i_e$$

It is useful to keep in mind that :

$$I_E = I_B + I_C$$

$$i_e = i_b + i_c$$

Now base current is usually very small, therefore, as a reasonable approximation,

$$I_E \simeq I_C \quad \text{and} \quad i_e \simeq i_c$$

b. Explain the working of Hartley oscillator with a neat circuit diagram. (8)

Answer:

14.11 Hartley Oscillator

The Hartley oscillator is similar to Colpitt's oscillator with minor modifications. Instead of using tapped capacitors, two inductors L_1 and L_2 are placed across a common capacitor C and the centre of the inductors is tapped as shown in Fig. 14.13. The tank circuit is made up of L_1, L_2 and C. The frequency of oscillations is determined by the values of L_1, L_2 and C and is given by :

$$f = \frac{1}{2\pi\sqrt{CL_T}} \qquad ...(i)$$

where $L_T = L_1 + L_2 + 2M$

Here M = mutual inductance between L_1 and L_2

Note that $L_1 - L_2 - C$ is also the feedback network that produces a phase shift of 180°.

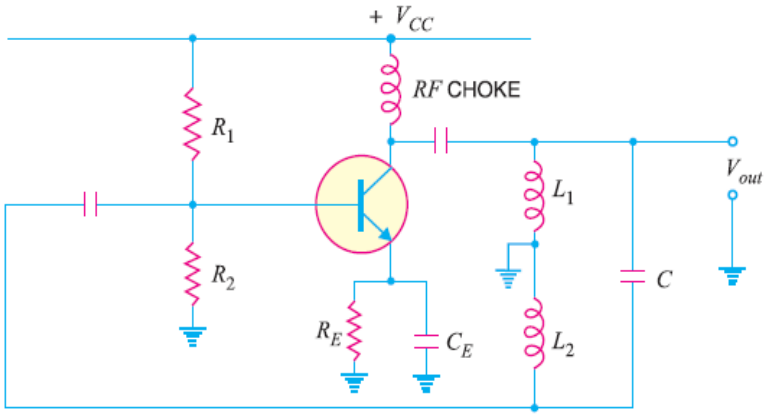


Fig. 14.13

Circuit operation. When the circuit is turned on, the capacitor is charged. When this capacitor is fully charged, it discharges through coils L_1 and L_2 setting up oscillations of frequency determined by *exp. (*i*). The output voltage of the amplifier appears across L_1 and feedback voltage across L_2. The voltage across L_2 is 180° out of phase with the voltage developed across L_1 (V_{out}) as shown in Fig. 14.14. It is easy to see that voltage fedback (*i.e.*, voltage across L_2) to the transistor provides positive feedback. A phase shift of 180° is produced by the transistor and a further phase shift of 180° is produced by $L_1 - L_2$ voltage divider. In this way, feedback is properly phased to produce continuous undamped oscillations.

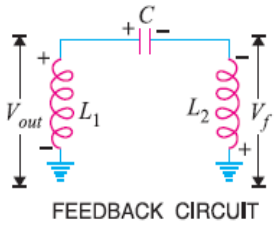


Fig. 14.14

Feedback fraction m_v. In Hartley oscillator, the feedback voltage is across L_2 and output voltage is across L_1.

$$\therefore \quad \text{Feedback fraction, } m_v = \frac{V_f}{V_{out}} = \frac{X_{L_2}}{X_{L_1}} = \frac{**L_2}{L_1}$$

or $$m_v = \frac{L_2}{L_1}$$

Q.6 a. Explain the parallel and serial transmission of information in digital systems. (8)

Answer: Please refer :Digital Systems – Principles and Applications, Tenth Edition, Ronald J Tocci, Neal S Widmerand Gregory L. Moss, Pearson Education, 2011.

b. What is the need for error detection and correction codes? Explain the parity method for error detection. (8)

Answer:

Environmental interference and physical defects in the communication medium can cause random bit errors during data transmission. Error coding is a method of detecting and correcting these errors to ensure information is transferred intact from its source to its destination. Error coding is used for fault tolerant computing in computer memory, magnetic and optical data storage media, satellite and deep space communications, network communications, cellular telephone networks, and almost any other form of digital data communication. Error coding uses mathematical formulas to encode data bits at the source into longer bit words for transmission. The "code word" can then be decoded at the destination to retrieve the information. The extra bits in the code word provide redundancy that, according to the coding scheme used, will allow the destination to use the decoding process to determine if the communication medium introduced errors and in some cases correct them so that the data need not be retransmitted. Different error coding schemes are chosen depending on the types of errors expected, the communication medium's expected error rate, and whether or not data retransmission is possible. Faster processors and better communications technology make more complex coding schemes, with better error detecting and correcting capabilities, possible for smaller embedded systems, allowing for more robust communications. However, tradeoffs between bandwidth and coding overhead, coding complexity and allowable coding delay between transmissions, must be considered for each application. Even if we know what type of errors can occur, we can't simple recognize them. We can do this simply by comparing this copy received with another copy of intended transmission. In this mechanism the source data block is send twice. The receiver compares them with the help of a comparator and if those two blocks differ, a request for re-transmission is made. To achieve forward error correction, three sets of the same data block are sent and majority decision selects the correct block. These methods are very inefficient and increase the traffic two or three times. Fortunately there are more efficient error detection and correction codes. There are two basic strategies for dealing with errors. One way is to include enough redundant information (extra bits are introduced into the data stream at the transmitter on a regular and logical basis) along with each block of data sent to enable the receiver to deduce what the transmitted character must have been. The other way is to include only enough redundancy to allow the receiver to deduce that error has occurred, but not which error has occurred and the receiver asks for a retransmission. The former strategy uses Error-Correcting Codes and latter uses Error-detecting Codes. To understand how errors can be handled, it is necessary to look closely at

what error really is. Normally, a frame consists of m-data bits (i.e., message bits) and r-redundant bits (or check bits). Let the total number of bits be n (m + r). An n-bit unit containing data and check-bits is often referred to as an n-bit codeword. Given any two code-words, say 10010101 and 11010100, it is possible to determine how many corresponding bits differ, just EXCLUSIVE OR the two code-words, and count the number of 1's in the result. The number of bits position in which code words differ is called the Hamming distance. If two code words are a Hamming distance d-apart, it will require d single-bit errors to convert one code word to other. The error detecting and correcting properties depends on its Hamming distance. • To detect d errors, you need a distance (d+1) code because with such a code there is no way that d-single bit errors can change a valid code word into another valid code word. Whenever receiver sees an invalid code word, it can tell that a transmission error has occurred. • Similarly, to correct d errors, you need a distance 2d+1 code because that way the legal code words are so far apart that even with d changes, the original codeword is still closer than any other code-word, so it can be uniquely determined.

3.2.3.1 Simple Parity Checking or One-dimension Parity Check

The most common and least expensive mechanism for error- detection is the simple parity check. In this technique, a redundant bit called **parity bit**, is appended to every data unit so that the number of 1s in the unit (including the parity becomes even).

Blocks of data from the source are subjected to a check bit or *Parity bit* generator form, where a parity of 1 is added to the block if it contains an odd number of 1's (ON bits) and 0 is added if it contains an even number of 1's. At the receiving end the parity bit is computed from the received data bits and compared with the received parity bit, as shown in Fig. 3.2.3. This scheme makes the total number of 1's even, that is why it is called *even parity checking*. Considering a 4-bit word, different combinations of the data words and the corresponding code words are given in Table 3.2.1.

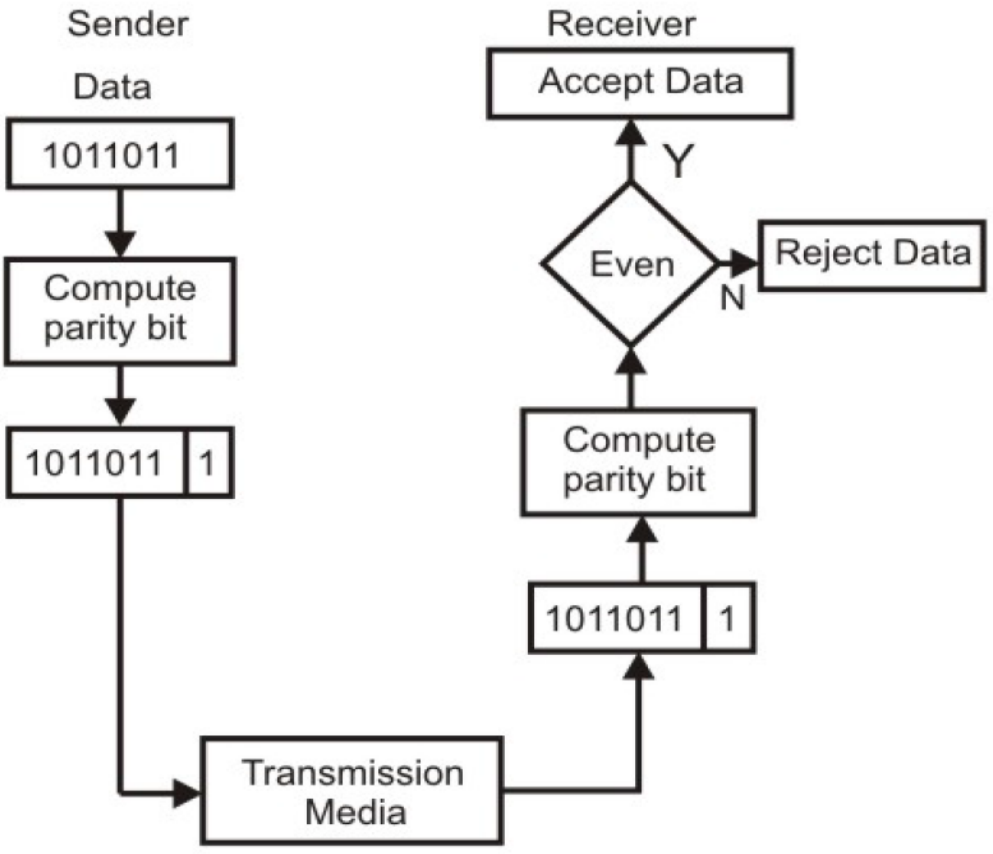


Figure 3.2.3 Even-parity checking scheme

Table 3.2.1 Possible 4-bit data words and corresponding code words

Decimal value	Data Block	Parity bit	Code word
0	0000	0	00000
1	0001	1	00011
2	0010	1	00101
3	0011	0	00110
4	0100	1	01001
5	0101	0	01010
6	0110	0	01100
7	0111	1	01111
8	1000	1	10001
9	1001	0	10010
10	1010	0	10100
11	1011	1	10111
12	1100	0	11000
13	1101	1	11011
14	1110	1	11101
15	1111	0	11110

Note that for the sake of simplicity, we are discussing here the even-parity checking, where the number of 1's should be an even number. It is also possible to use *odd-parity* checking, where the number of 1's should be odd.

Performance

An observation of the table reveals that to move from one code word to another, at least two data bits should be changed. Hence these set of code words are said to have a minimum distance (*hamming distance*) of 2, which means that a receiver that has knowledge of the code word set can detect all single bit errors in each code word. However, if two errors occur in the code word, it becomes another valid member of the set and the decoder will see only another valid code word and know nothing of the error. Thus errors in more than one bit cannot be detected. In fact it can be shown that a single parity check code can detect only odd number of errors in a code word.

Q.7 a. Prove the following identities using Boolean algebra: (8)

(i) $(A+B)(A+\overline{AB})C+\overline{A}(B+\overline{C})+\overline{A}B+ABC=C(A+B)+\overline{A}(B+\overline{C})$

(ii) $\overline{\overline{A(A\cdot B)}\cdot\overline{B(A\cdot B)}}=A\oplus B$

Answer:

(i)

$(A+B)(A+\overline{AB})C + \bar{A}(B+\bar{C}) + \bar{A}B + ABC$

Using Demorgan's law $\overline{AB} = \bar{A} + \bar{B}$

$= (A+B)(A+\bar{A}+\bar{B})C + \bar{A}B + \bar{A}\bar{C} + \bar{A}B + ABC$

$= (A+B)(1+\bar{B})C + \bar{A}B + \bar{A}\bar{C} + ABC$ as $(A+\bar{A})=1$

$= (A+B)\cdot 1\cdot C + \bar{A}(B+\bar{C}) + ABC$

$= AC + BC + ABC + \bar{A}(B+\bar{C})$

$= AC(1+B) + BC + \bar{A}(B+\bar{C})$

$= AC + BC + \bar{A}(B+\bar{C})$

$= C(A+B) + \bar{A}(B+\bar{C})$

(ii) $\overline{\overline{A(A.B)}.\overline{B(A.B)}} = A \oplus B$

Let us take $X = \overline{A(\overline{A.B})}$

$Y = \overline{B(\overline{A.B})}$

So we have $\overline{X.Y} = A \oplus B$ -------3

Also $X = \overline{A(\overline{A.B})}$

$= \overline{A(A'+B')}$

By using DeMorgan's Law (AB)'=A'+B'

X = (A(A'+B'))'=(AA'+AB')'=(AB')'=(A'+B) ------1

Now Y = (B(AB)')'=[B(A'+B')]'= [A'B+BB']'=(A'B)'=(A+B') ------2

Now Combining X & Y from 1 & 2 above, we have L.H.S in 3 as :

((A+B')(A'+B))'

=[AA'+BB+A'B'+AB]'

=(AB+A'B')'

=A XOR B = **RHS**

Hence Proved

b. Reduce the following equation using k-map (8)

$$Y = \overline{A}BC + AC\overline{D} + A\overline{B} + ABC\overline{D} + \overline{A}\overline{B}C$$

Answer:

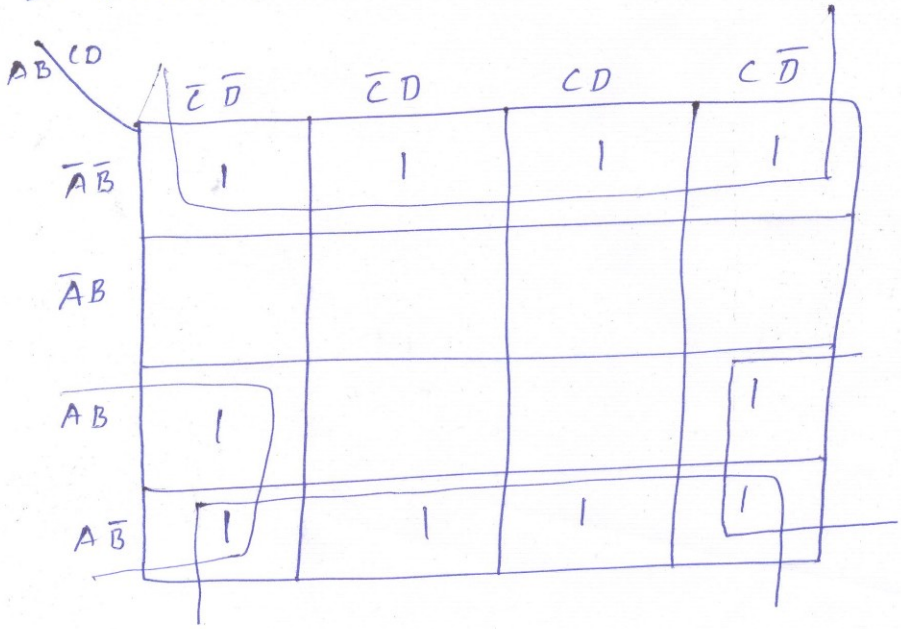
$Y = A'B'C' + A\bar{C}\bar{D} + A\bar{B} + ABC\bar{D} + \bar{A}\bar{B}C$

$= A'B'C'(D+D') + A(B+\bar{B})\bar{C}\bar{D} + A\bar{B}(C+\bar{C})(D+\bar{D}) + ABC\bar{D} + \bar{A}\bar{B}C(D+\bar{D})$

$= \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}\bar{C}\bar{D} + AB\bar{C}\bar{D} + A\bar{B}\bar{C}\bar{D} + (A\bar{B}C + A\bar{B}\bar{C})(D+\bar{D}) + ABC\bar{D} + \bar{A}\bar{B}CD + \bar{A}\bar{B}C\bar{D}$

$= \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}\bar{C}D + AB\bar{C}\bar{D} + A\bar{B}\bar{C}\bar{D} + A\bar{B}CD + A\bar{B}\bar{C}D + A\bar{B}C\bar{D} + A\bar{B}\bar{C}\bar{D} + ABC\bar{D} + \bar{A}\bar{B}CD + \bar{A}\bar{B}C\bar{D}$

$= \Sigma(0, 1, 2, 3, 8, 9, 10, 11, 12, 14)$



$= \bar{B} + A\bar{D}$

Q.8 a. Explain Full adder with an example. (8)

Answer:

1.11.2 FULL ADDER

The adder circuit is capable of adding the content of two registers. It must include provision for handling carries as well as an addend and augends bits. So there must be three inputs to each stage of a multi digit adder, except the stage for the least significant bits. One for each input from the numbers being added, one for any carry that might have been generated or propagated by the previous stage.

There are three inputs to the full-adder X and Y inputs from the respective digits of the registers to be added, the C_i input, which is for any carry generated by the previous stage. The two outputs are S, which is the output value for that stage of the addition, and C_0, which produces the carry to be added into the next stage. The Boolean expressions for the input output relationships for each of the two outputs are as follows:

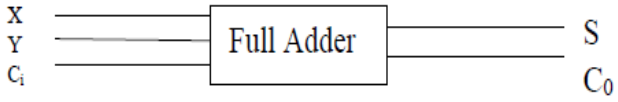


Figure 1.3 Full adder circuit – Block Diagram

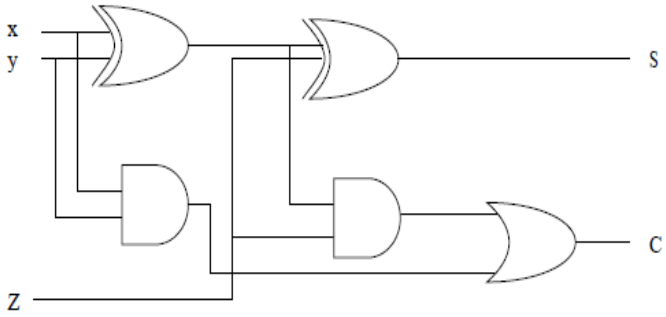


Figure. 1.4 Logic Diagram

INPUT			OUTPUT	
X	Y	CI	S	C0
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	1	0	0	1
1	1	1	1	1

Table 1.10 Truth Table

1.11.4 BINARY CODED DECIMAL ADDER

Arithmetic units which perform operations on numbers stored in BCD form must have the ability to add 4-bit representations of decimal digits. To do this BCD adder is used. A block diagram symbol for an adder is shown in the figure 1.6.

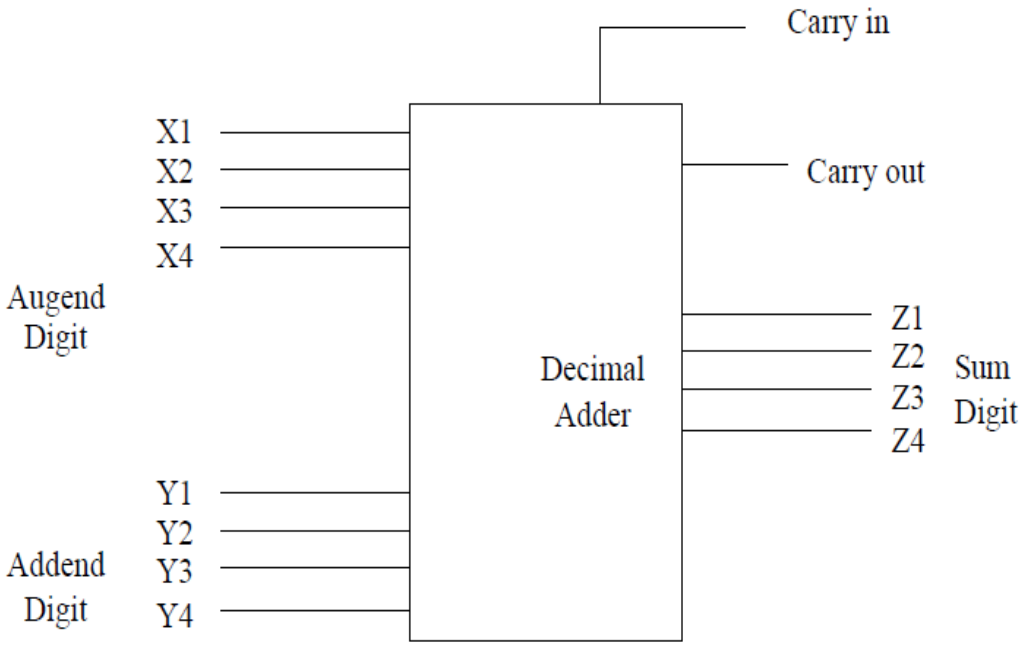


Figure 1.6 BCD Adder

The adder has an augends digit input consisting of four lines, an addend digit input for four lines, a carry-in and a carryout, and a sum digit with four output lines. The augend digit, addend digit and sum digit with four-output line. The augend digit, addend digit and sum digit are each represented 8, 4, 2, 1 BCD form. The purpose of the BCD adder in figure 1.6 is to add the augend and addend digits and the carry-in and produce a sum digit and carry out.

There are eight inputs to the BCD adder; four X_i, or augend, inputs and four Y_i (or) addend digits. Each input will represent a 0 or a 1, during a given addition. If 3 (0011) is to be added to 2 (0010), then $X_8=0$, $X_4=0$,$X_2=1$ and $X_1=1$; $Y_8=0$,$Y_4=0$,$Y_2=1$ and $Y_1=0$.

A further difficulty arises when a carry is generated. If 7_{10} (0111) is added to 6_{10} (0110), a carry will be generated, but the output from the base – 16 adder will be 1101. This 1101 does not represent any decimal digit in the 8,4,2,1 system and must be connected. The method used to correct this is to add 6_{10} (0110) to the sum from the base – 16 address whenever a carry is generated. This addition is performed by adding 1's to the weight 4 and weight 2 position output lines from the base – 16 adder when a carry is generated.

b. What is a decoder? Draw the logic circuit of a 3 line to 8 line decoder and explain its working. (8)

Answer:

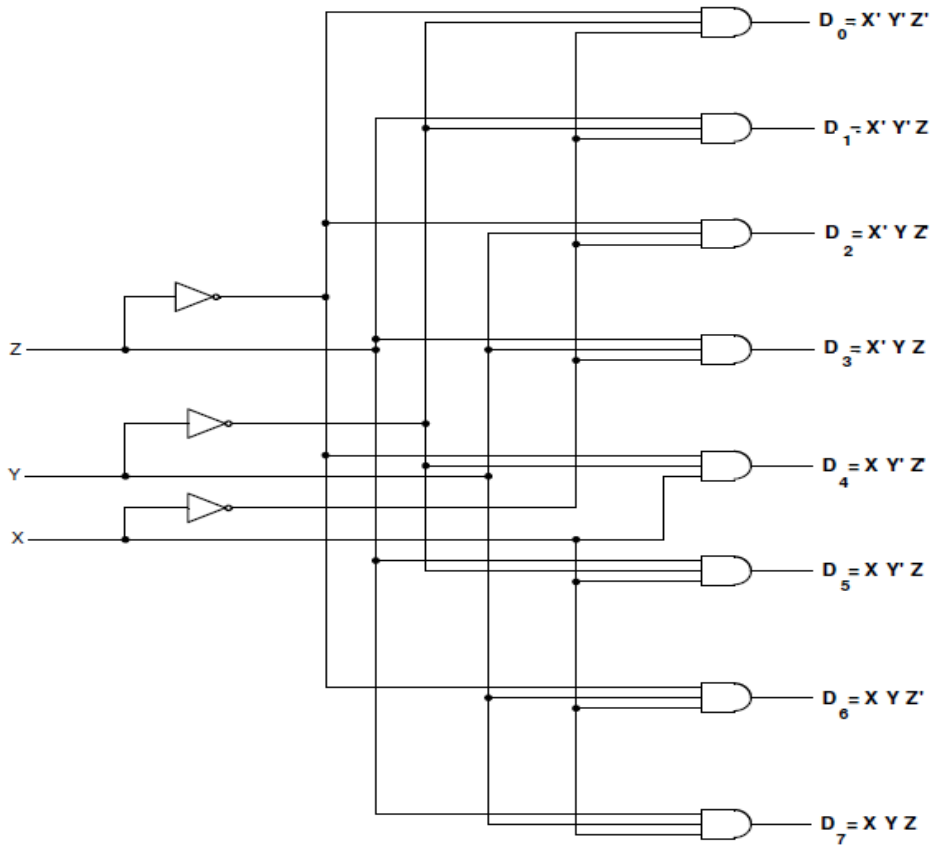
Decoder: A Decoder is a combinational logic circuit that converts Binary words into alphanumeric characters. Thus the inputs to a decoder are the bits 1, 0 and their combinations. The output is the corresponding decimal number. It converts binary information from *n* input lines to a maximum of 2_n unique output lines. If the *n-bit* decoded information has unused or don't-care combinations, the decoder output will have less than 2_n outputs.

Working: The logic circuit of a 3 line to 8 line decoder is shown in fig.6 (a). The three inputs (x, y, z) are decoded into eight outputs (from D_0 to D_7), each output representing one of the minterms of the 3-input variables. The three inverters provide the complement of the inputs, and each one of the eight AND gates generate one of the minterms. A particular application of this decoder is a binary-to-octal conversion. The input variables may represent a binary number, and the outputs will then represent the eight digits in the octal number system. However, a 3-to-8 line decoder can be used for decoding any 3-bit code to provide eight outputs, one for each element of the code.

The operation of the decoder may be verified from its input-output relationships shown in Table 6.1.The table shows that the output variables are mutually exclusive because only one output can be equal to 1 at any one time. Consider the case when X=0, Y=0 and Z=0, the output line D_0 (X', Y', Z') is equal to 1 represents the minterm equivalent of the binary number presently available in the input lines.

Inputs			Outputs							
X	Y	Z	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	**1**	0	0	0	0	0	0	0
0	0	1	0	**1**	0	0	0	0	0	0
0	1	0	0	0	**1**	0	0	0	0	0
0	1	1	0	0	0	**1**	0	0	0	0
1	0	0	0	0	0	0	**1**	0	0	0
1	0	1	0	0	0	0	0	**1**	0	0
1	1	0	0	0	0	0	0	0	**1**	0
1	1	1	0	0	0	0	0	0	0	**1**

Table 6.1 Truth Table of 3-to-8 line Decoder



Q.9 a. With relevant diagram explain the working of master- slave JK Flip-Flop. (8)

Answer:

Ans:

Master-Slave *J-K* FLIP-FLOP: A master-slave *J-K* FLIP-FLOP is a cascade of two *S-R* FLIP-FLOPS. One of them is known as Master and the other one is slave. Fig.11(a) shows the logic circuit. The master is positively clocked. Due to the presence of inverter, the slave is negatively clocked. This means that when clock is high, the master is active and the slave is inactive.

When the clock is low, the master is inactive and the slave is active. Fig.11(b) shows the symbol. This is a level clocked Flip-Flop. When clock is high, any changes in J and K inputs can affect S and R outputs. Therefore, J and K are kept constant during positive half of clock. When clock is low, the master is inactive and J and K inputs can be allowed to be changed. The different conditions are Set, Reset, and Toggle. The race condition is avoided because of feedback from slave to master and the slave being inactive during positive half of clock.

(i) **SET State**: Assume that Q is low (and $\overline{Q}$ is high). For high J, low K and high CLK, the Master goes to SET state giving High S and Low R. Since Slave is inactive, Q and $\overline{Q}$ do not change. When CLK becomes Low, the Slave becomes to Set state giving High Q (and low $\overline{Q}$).

(ii) **RESET State**: At the end of Set State Q is High (and $\overline{Q}$ low). Now if J is low, K is high and CLK is high, the Master Resets giving Low S and High R. Q and $\overline{Q}$ do not change because Slave is inactive. When CLK becomes Low, the Slave becomes active and resets giving Low Q (and High $\overline{Q}$).

		Inputs			Output
PR	CLR	CLK	J	K	Q
0	0	X	X	X	Race Condition
0	1	X	X	X	1
1	0	X	X	X	0
1	1	X	0	0	No change
1	1		0	1	0
1	1		1	0	1
1	1		1	1	Toggle

Table 11.1 Truth Table of JK Master-Slave Flip-Flop

(iii) Toggle State: If both J and K are High, the Slave copies the Master. When CLK is High, the Master toggles once. Then the Slave toggles once when CLK is low. If the Master toggles into Set state, the slave copies the Master and toggles into Set state. If the Master toggles into Reset state, the slave again copies the Master and toggles into Reset state. Since the second FLIP-FLOP simply follows the first one, it is referred to as the *slave* and the first one as the *master*. Hence, this configuration is referred to as *master-slave(M-S) FLIP*-FLOP.

Truth Table of JK Master Slave Flip-Flop in Table 11.1 shows that a Low PR and Low CLR can cause race condition. Therefore, PR and CLR are kept High when inactive. To clear, we make CLR Low and to preset we make PR Low. In both cases we change them to High when the system is to be run.

Low J and Low K produce inactive state irrespective of clock input. If K goes High, the next clock pulse resets the Flip-Flop. If J goes High by itself, the next clock pulse sets the Flip-Flop. When both J and K are High, each clock pulse produces one toggle.

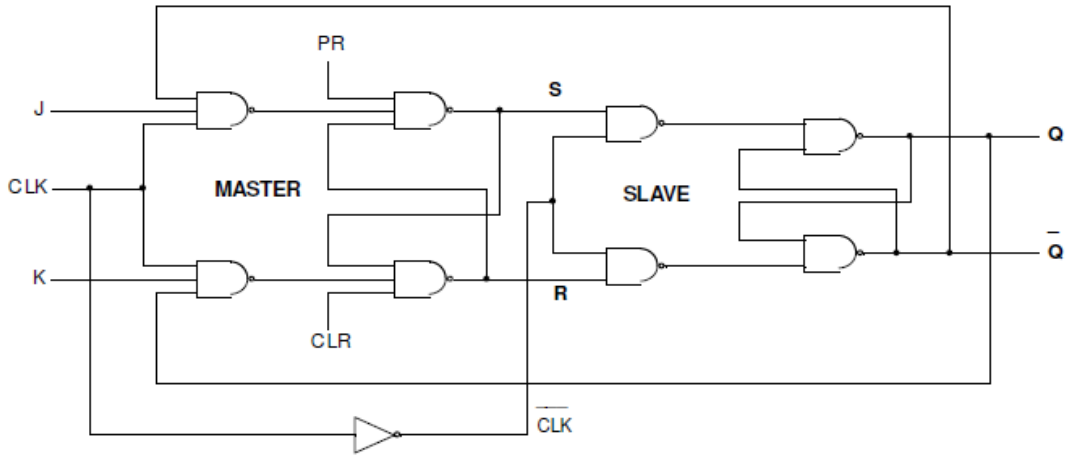


Fig.11(a) Logic Diagram of Master-Slave J-K FLIP-FLOP

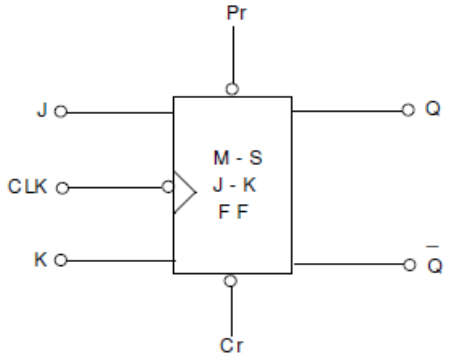
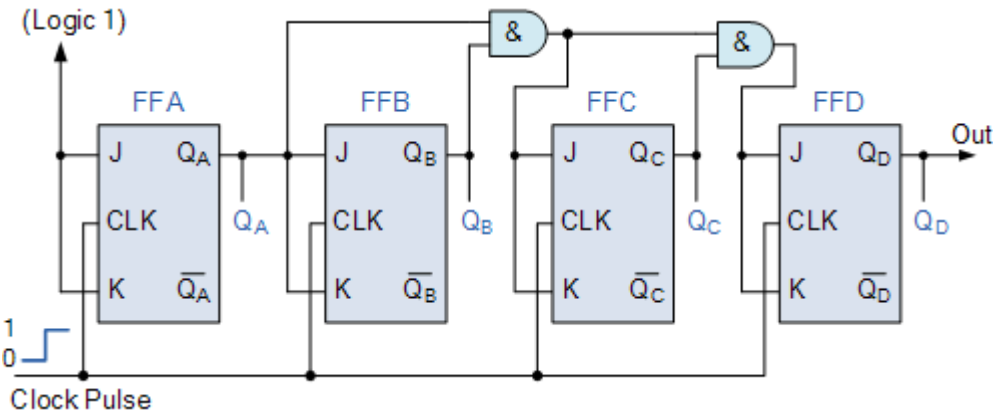


Fig.11(b) Logic Symbol of Master-Slave J-K FLIP-FLOP

b. Draw the diagram of a 4-bit synchronous up Counter and explain its working along with the waveforms. (8)

Answer:

Binary 4-bit Synchronous Up Counter

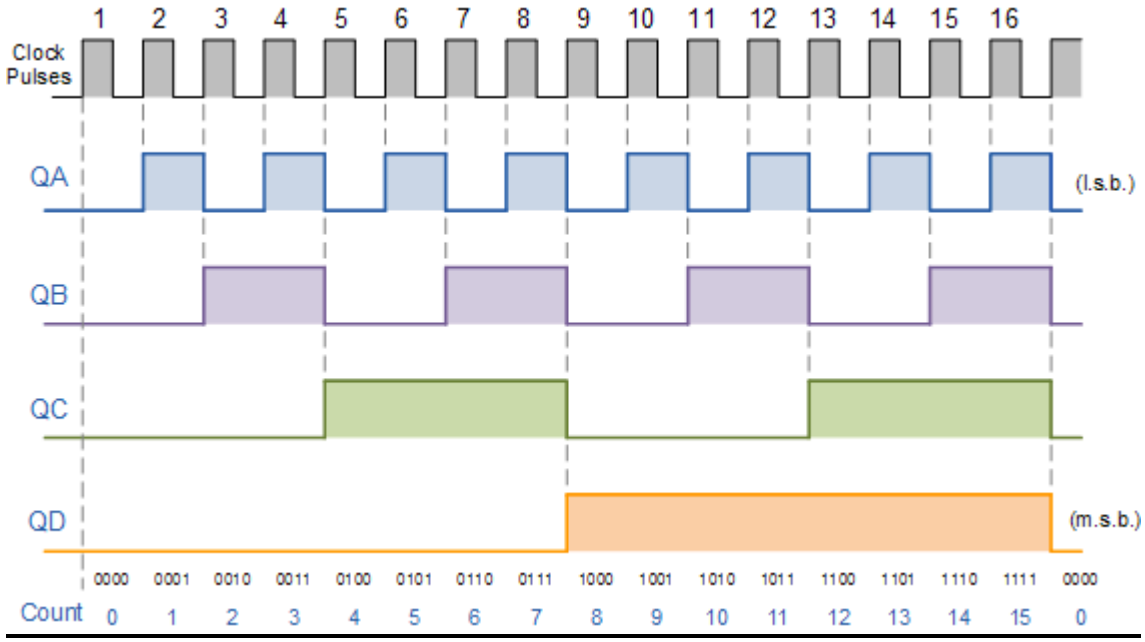


It can be seen above, that the external clock pulses (pulses to be counted) are fed directly to each of the **J-K flip-flops** in the counter chain and that both the J and K inputs are all tied together in toggle mode, but only in the first flip-flop, flip-flop FFA (LSB) are they connected HIGH, logic "1" allowing the flip-flop to toggle on every clock pulse. Then the synchronous counter follows a predetermined sequence of states in response to the common clock signal, advancing one state for each pulse.

The J and K inputs of flip-flop FFB are connected directly to the output Q_A of flip-flop FFA, but the Jand K inputs of flip-flops FFC and FFD are driven from separate AND gates which are also supplied with signals from the input and output of the previous stage. These additional AND gates generate the required logic for the JK inputs of the next stage.

If we enable each JK flip-flop to toggle based on whether or not all preceding flip-flop outputs (Q) are "HIGH" we can obtain the same counting sequence as with the asynchronous circuit but without the ripple effect, since each flip-flop in this circuit will be clocked at exactly the same time.

Then as there is no inherent propagation delay in synchronous counters, because all the counter stages are triggered in parallel at the same time, the maximum operating frequency of this type of frequency counter is much higher than that for a similar asynchronous counter circuit.



Because this 4-bit synchronous counter counts sequentially on every clock pulse the resulting outputs count upwards from 0 (0000) to 15 (1111). Therefore, this type of counter is also known as a **4-bit Synchronous Up Counter**.

However, we can easily construct a **4-bit Synchronous Down Counter** by connecting the ANDgates to the Q output of the flip-flops as shown to produce a waveform timing diagram the reverse of the above. Here the counter starts with all of its outputs HIGH (1111) and it counts down on the application of each clock pulse to zero, (0000) before repeating again.

TEXT BOOKS

1. Electronic Devices and Circuits, Fifth Edition, David A Bell, OXFORD University Press,Thirteenth Impression-2010

2. Digital Systems – Principles and Applications, Tenth Edition, Ronald J Tocci, Neal S Widmer and Gregory L. Moss, Pearson Education, 2011