**CMR INSTITUTE OF TECHNOLOGY**

## Scheme Of Evaluation
## IAT-2 – May'17

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Sub:** | DATA COMMUNICATION | | | | | | | **Code:** | 15CS46 |
| Date: | 10/05/17 | Duration: | 90mins | Max Marks: | 50 | **Sem:** | IV | **Branch:** | ISE |

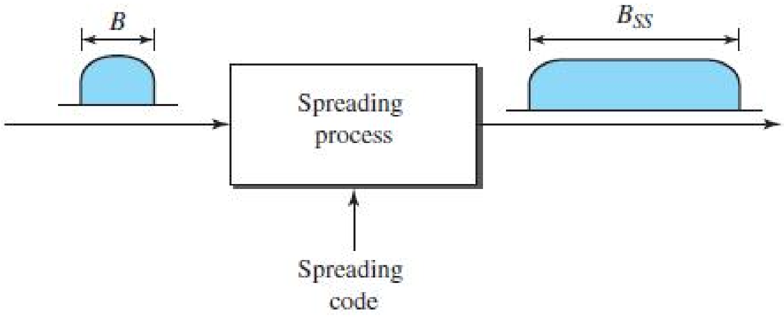
**Note: Answer Any Five Question**

| Question # | | Description | Marks Distribution | | Max Marks |
|---|---|---|---|---|---|
| 1 | a) | Definition($B_{SS}$>>B)<br>FHSS, Diagram and explanation with graph | 2M<br>8M | 10M | 10 M |
| 2 | a) | Binary PSK, Bandwith for PSK,Implementation | 5M | 5M | 10 M |
| | b) | Statistical TDM with explanation | 5M | 5M | |
| 3 | a) | Stop and wait protocol,<br>States(Ready and block state on sender site,ready state on receiver),<br>Diagram with explanation | 1M<br>3M<br>4M | 8M | 10 M |
| | b) | Piggybacking | 2M | 2M | |
| 4 | i)<br>ii) | Calculating Checksum value at sender<br>Calculating result at receiver | 5M<br>5M | 10M | 10 M |
| 5 | | Virtual Circuit Network<br>Phases<br>Set up and, ack diagram and data transfer diagram | 2M<br>4+4M | 10M | 10 M |
| 6 | | Finding polynomial expression<br>Finding Codeword | 3M<br>7M | 10M | 10 M |
| 7 | | HDLC,3 frames and control fields for each frames | 3M<br>3M | 6M | 10M |
| 8 | | CDMA, data and chip representation, solution | 10M | 10M | 10M |

Solution

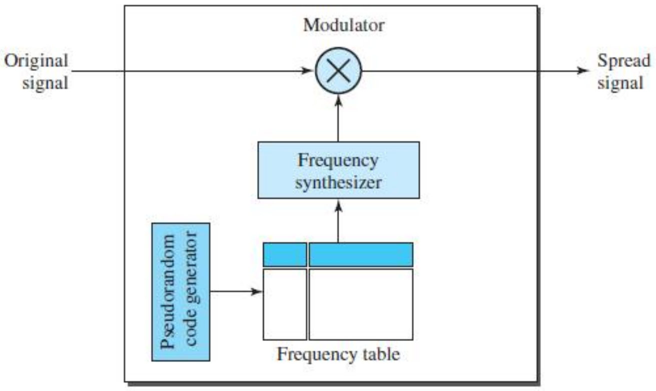
1. What is Spread Spectrum? Discuss FHSS technique to spread the bandwidth

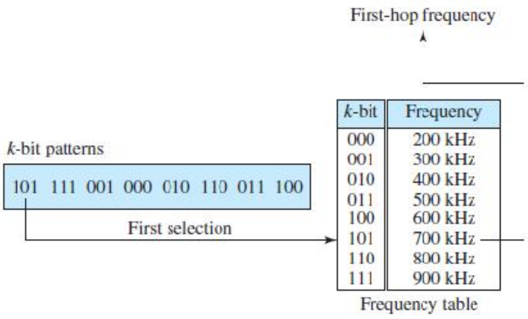
**(SS),** spread spectrum techniques combine signals from different sources to fit into a larger bandwidth ; they spread the original spectrum needed for each station. If the required bandwidth for each station is *B*, spread spectrum expands it to $B_{ss}$, such that $B_{ss} >> B$. The expanded bandwidth allows the source to wrap its message in a protective envelope for a more secure transmission.
Spread spectrum achieves its goals through two principles:
1. The bandwidth allocated to each station needs to be, by far, larger than what is needed. This allows redundancy.
2. The expanding of the original bandwidth *B* to the bandwidth $B_{ss}$ must be done by a process that isindependent of the original signal. In other words, the spreading process occurs after the signal is created by the source.



## Frequency Hopping Spread Spectrum

The **frequency hopping spread spectrum (FHSS)** technique uses *M* different carrier frequencies that are modulated by the source signal. At one moment, the signal modulates one carrier frequency; at the next moment, the signal modulates another carrier frequency. Although the modulation is done using one carrier frequency at a time,*M* frequencies are used in the long run. The bandwidth occupied by a source after spreading is $B_{FHSS} >> B$.
A **pseudorandom code generator,**
called ***pseudorandom noise*** **(PN),** creates a *k*-bit pattern for every **hopping period** $T_h$. The frequency table uses the pattern to find the frequency to be used for this hopping period and passes it to the frequency synthesizer. The frequency synthesizer creates a carrier signal of that frequency, and the source signal modulates the carrier signal. The pattern for this station is 101, 111, 001, 000, 010, 011, 100. Note that the pattern is pseudorandom; it is repeated after eight hoppings. This means that at hopping period 1, the pattern is 101. The frequency selected is 700 kHz; the source signal modulates this carrier frequency. The second *k*-bit pattern selected is 111, which selects the 900-kHz carrier; the eighth pattern is 100, and the frequency is 600 kHz. After eight hoppings, the pattern repeats, starting from 101 again.

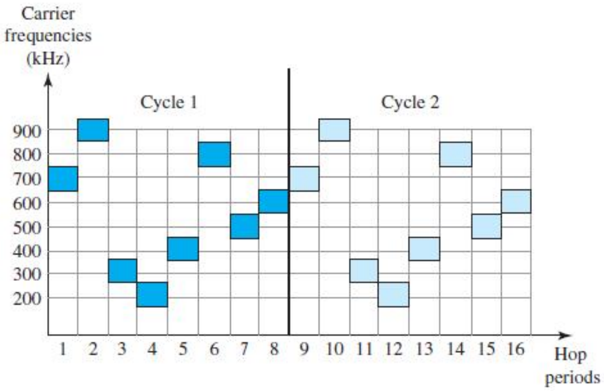




| *k*-bit | Frequency |
|---|---|
| 000 | 200 kHz |
| 001 | 300 kHz |
| 010 | 400 kHz |
| 011 | 500 kHz |
| 100 | 600 kHz |
| 101 | 700 kHz |
| 110 | 800 kHz |
| 111 | 900 kHz |

***Bandwidth Sharing***

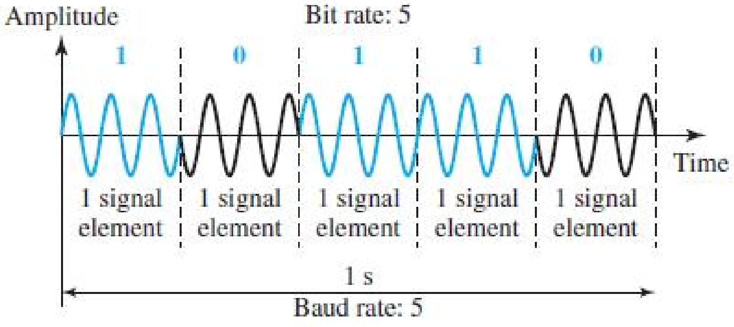
If the number of hopping frequencies is *M*, we can multiplex *M* channels into one by using the same *B*ss bandwidth. This is possible because a station uses just one frequency in each hopping period; *M* 1 other requencies can be used by *M* 1 other stations.

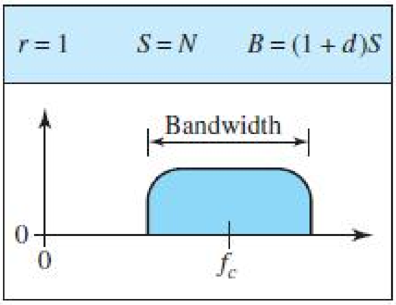


**2.** Explain about

i) Binary PSK,

In phase shift keying, the phase of the carrier is varied to represent two or more different signal elements. Both peak amplitude and frequency remain constant as the phase changes. Today, PSK is more common than ASK or FSK. The simplest PSK is binary PSK, in which we have only two signal elements, one with a phase of 0°, and the other with a phase of 180°. Binary PSK is as simple as binary ASK with one big advantage—it is less susceptible to noise. In ASK, the criterion for bit detection is the amplitude of the signal; in PSK, it is the phase. Noise can change the amplitude easier than it can change the phase. In other words, PSK is less susceptible to noise than ASK. PSK is superior to FSK because we do not need two carrier signals. However, PSK needs more sophisticated hardware to be able to distinguish between phases.



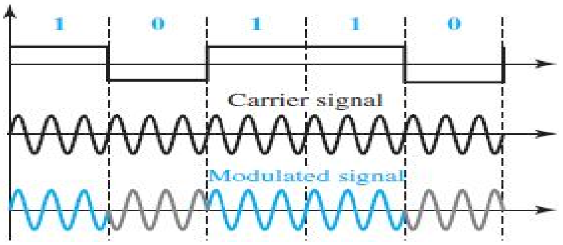


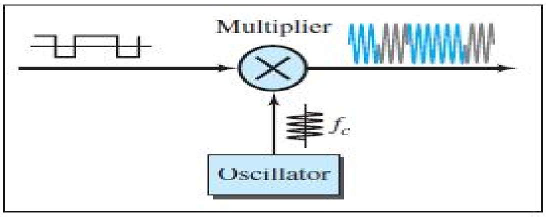
***Bandwidth***

The bandwidth is the same as that for binary ASK, but less than that for BFSK. No bandwidth is wasted for separating two carrier signals.

***Implementation***

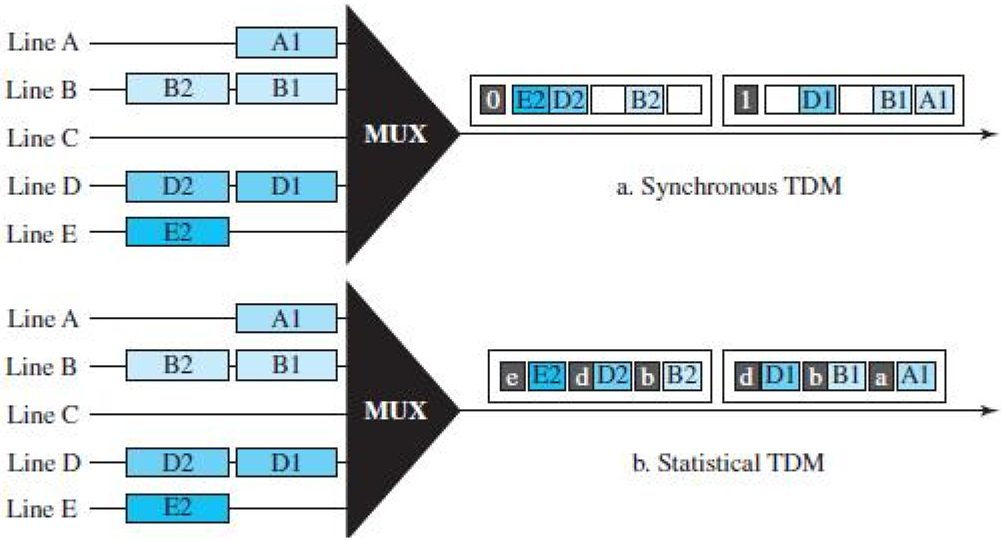
The implementation of BPSK is as simple as that for ASK. The reason is that the signal element with phase 180° can be seen as the complement of the signal element with phase 0°. This gives us a clue on how to implement BPSK. The polar NRZ signal is multiplied by the carrier frequency; the 1 bit (positive voltage) is represented by a phase starting at 0°; the 0 bit (negative voltage) is represented by a phase starting at 180°.





## ii) Statistical TDM

In synchronous TDM, each input has a reserved slot in the output frame. This can be inefficient if some input lines have no data to send. In statistical time-division multiplexing, slots are dynamically allocated to improve bandwidth efficiency. Only when an input line has a slot's worth of data to send is it given a slot in the output frame. In statistical multiplexing, the number of slots in each frame is less than the number of input lines. The multiplexer checks each input line in roundrobin fashion; it allocates a slot for an input line if the line has data to send; otherwise, it skips the line and checks the next line.



3. a) Explain stop and wait protocol with neat diagram.

Stop-and-Wait Protocol

Our second protocol is called the Stop-and-Wait protocol, which uses both flow and error control. We show a primitive version of this protocol here, In this protocol, the sender sends one frame at a time and waits for an acknowledgment before sending the next one. To detect corrupted frames, we need to add a CRC (see Chapter 10) to each data frame. When a frame arrives at the receiver site, it is checked. If its CRC is incorrect, the frame is corrupted and silently discarded.

The silence of the receiver is a signal for the sender that a frame was either corrupted or lost. Every time the sender sends a frame, it starts a timer. If an acknowledgment arrives before the timer expires, the timer is stopped and the sender sends the next frame (if it has one to send). If the timer expires, the sender resends the previous frame, assuming that the frame was either lost or corrupted. This means that the sender needs to keep a copy of the frame until its acknowledgment arrives. When the corresponding acknowledgment arrives, the sender discards the copy and sends the next frame if it is ready Note that only one frame and one acknowledgment can be in the channels at any time.

**We describe the sender and receiver states below:**

**Sender States**

The sender is initially in the ready state, but it can move between the ready and blocking state.

**Ready State:**

When the sender is in this state, it is only waiting for a packet from the network layer. If a packet comes from the network layer, the sender creates a frame, saves a copy of the frame, starts the only timer and sends the frame. The sender then moves to the blocking state.

**Blocking State:**

When the sender is in this state, three events can occur:

a. If a time-out occurs, the sender resends the saved copy of the frame and restarts the timer.

b. If a corrupted ACK arrives, it is discarded.

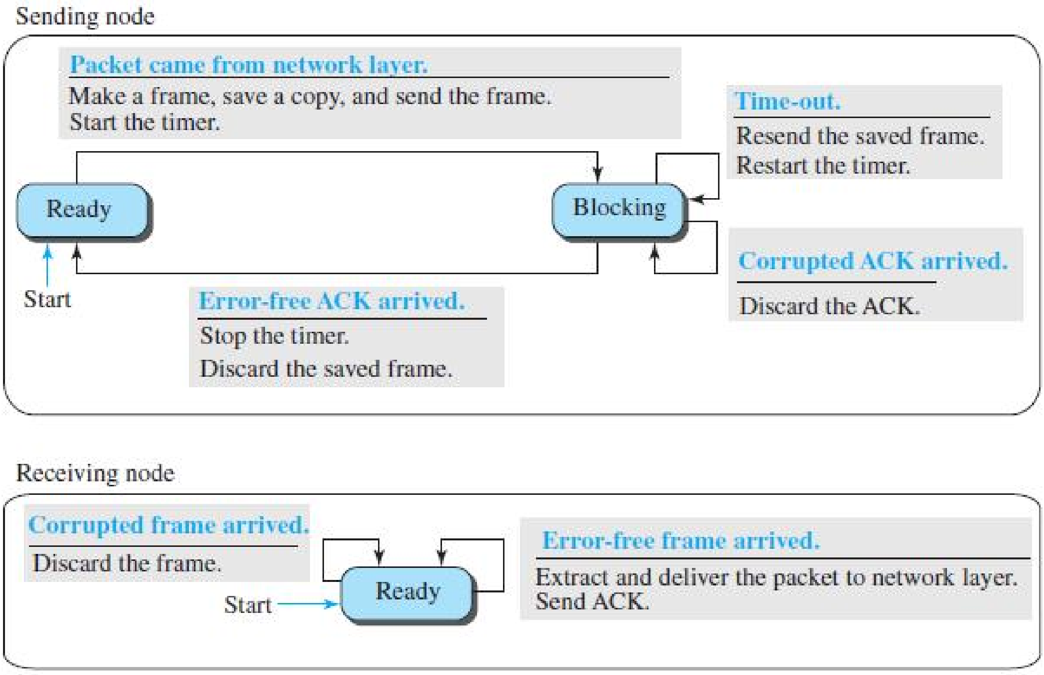
c. If an error-free ACK arrives, the sender stops the timer and discards the saved copy of the frame. It then moves to the ready state.

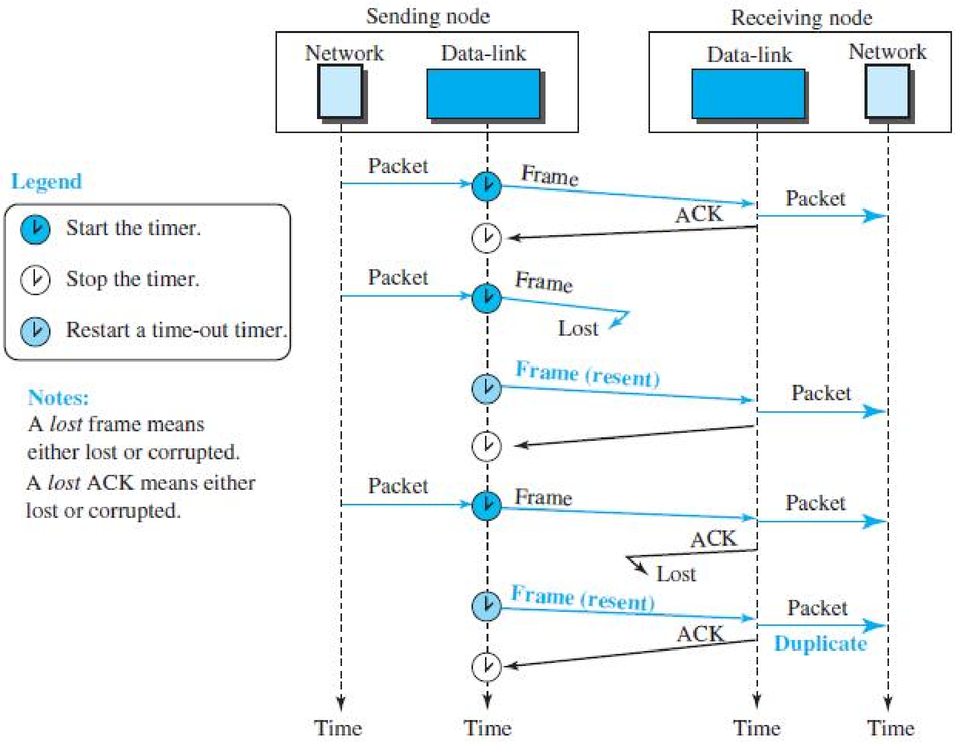
***Receiver***

The receiver is always in the *ready* state. Two events may occur:

a. If an error-free frame arrives, the message in the frame is delivered to the network layer and an ACK is sent.

b. If a corrupted frame arrives, the frame is discarded.

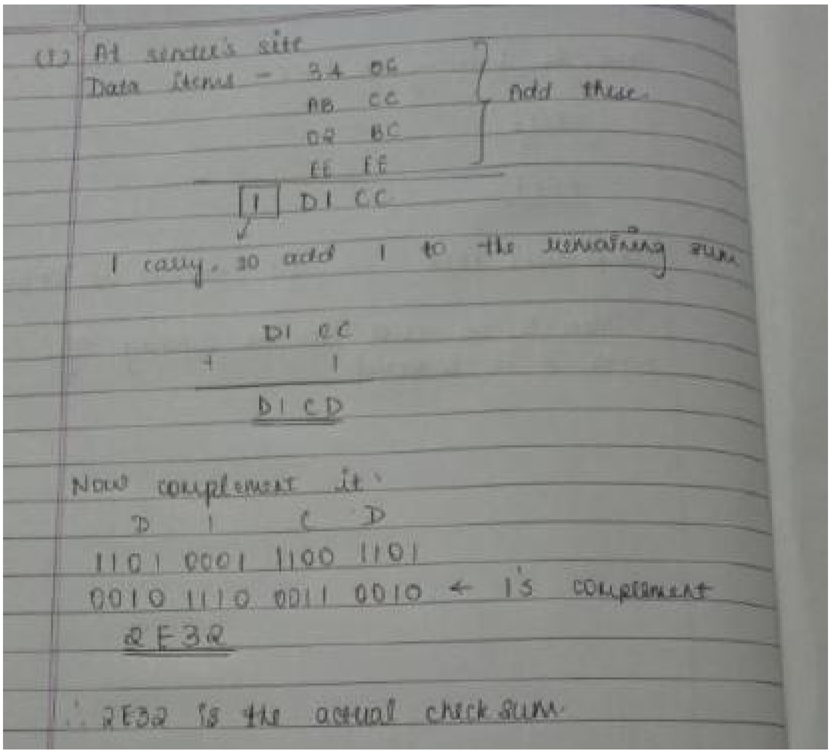




b) What is Piggybacking?

Unidirectional communication, in which data is flowing only in one direction although the acknowledgment may travel in the other direction. Protocols have been designed in the past to allow data to flow in both directions. However, to make the communication more efficient, the data in one direction is piggybacked with the acknowledgment in the other direction. In other words, when node A is sending data to node B, Node A also acknowledges the data received from node B. Because piggybacking makes communication at the datalink layer more complicated, it is not a common practice.

**4.** What is Internet checksum? If sender needs to send four data items 3456,ABCD, 02BC,EEEE(given in Hex) i) Find the checksum at sender's site ii) Find the checksum at receiver's site if there is error.



(i) At sender's site

```
Data items -  34 56  }
              AB CC  }  Add these.
              D2 BC  }
              EE EE  }
           ----------
           [1] D1 CC
```

1 carry, so add 1 to the remaining sum

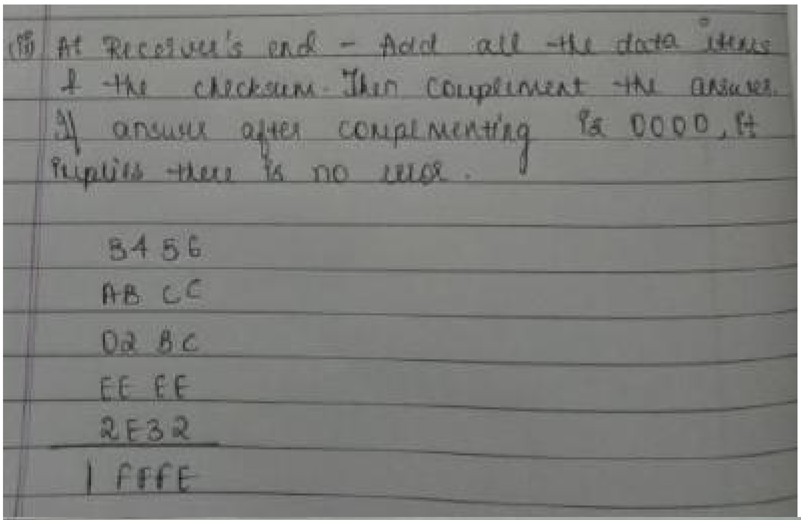
```
     D1 CC
  +      1
  ---------
     D1 CD
```

Now complement it:

```
   D    1    C    D
1101 0001 1100 1101
0010 1110 0011 0010  ← 1's complement
 2E32
```

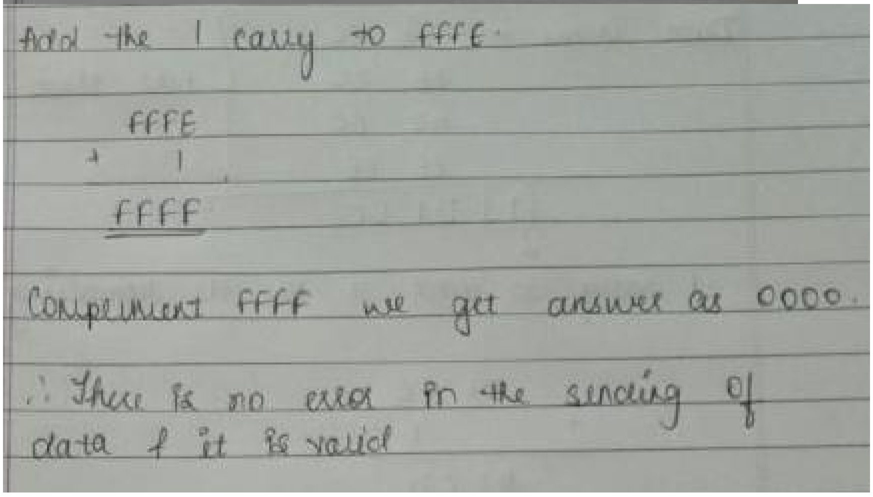
∴ 2E32 is the actual check sum.

**At receiver,**



(ii) At Receiver's end - Add all the data items & the checksum. Then Complement the answer. If answer after complementing is 0000, it implies there is no error.

```
 34 56
 AB CC
 D2 BC
 EE EE
 2E32
------
1 FFFE
```



Add the 1 carry to FFFE.

```
   FFFE
 +    1
 ------
   FFFF
```

Complement FFFF we get answer as 0000.

∴ There is no error in the sending of data & it is valid

### 5. Explain in detail about Virtual Circuit Network.

A **virtual-circuit network** is a cross between a circuit-switched network and a datagram network. It has some characteristics of both.

1.As in a circuit-switched network, there are setup and teardown phases in addition to the data transfer phase.

2.Resources can be allocated during the setup phase, as in a circuit-switched network, or on demand, as in a datagram network.

3.As in a datagram network, data are packetized and each packet carries an address in the header. However, the address in the header has local jurisdiction (it defines what the next switch should be and the channel on which the packet is being carried), not end-to-end jurisdiction. The reader may ask how the intermediate switches know where to send the packet if there is no final destination address carried by a packet. The answer will be clear when we discuss virtual-circuit identifiers in the next section.

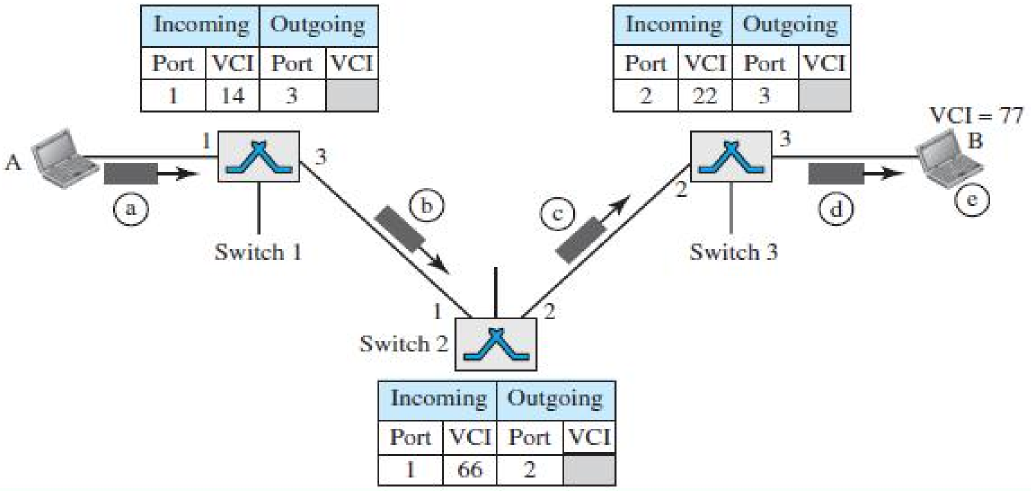
4. As in a circuit-switched network, all packets follow the same path established during the connection.

5. A virtual-circuit network is normally implemented in the data-link layer, while a circuit-switched network is implemented in the physical layer and a datagram network in the network layer.

#### *Setup Request*

A setup request frame is sent from the source to the destination. Figure 8.14 shows the process.

**Figure 8.14** *Setup request in a virtual-circuit network*



| Incoming | | Outgoing | |
|---|---|---|---|
| Port | VCI | Port | VCI |
| 1 | 14 | 3 | |

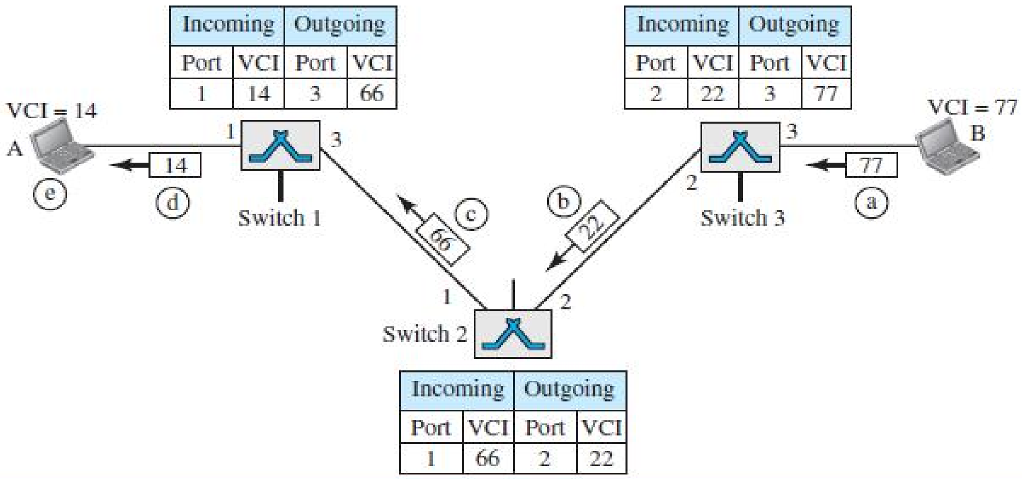
| Incoming | | Outgoing | |
|---|---|---|---|
| Port | VCI | Port | VCI |
| 2 | 22 | 3 | |

| Incoming | | Outgoing | |
|---|---|---|---|
| Port | VCI | Port | VCI |
| 1 | 66 | 2 | |

*Acknowledgment*

A special frame, called the *acknowledgment frame,* completes the entries in the switching tables. Figure 8.15 shows the process.

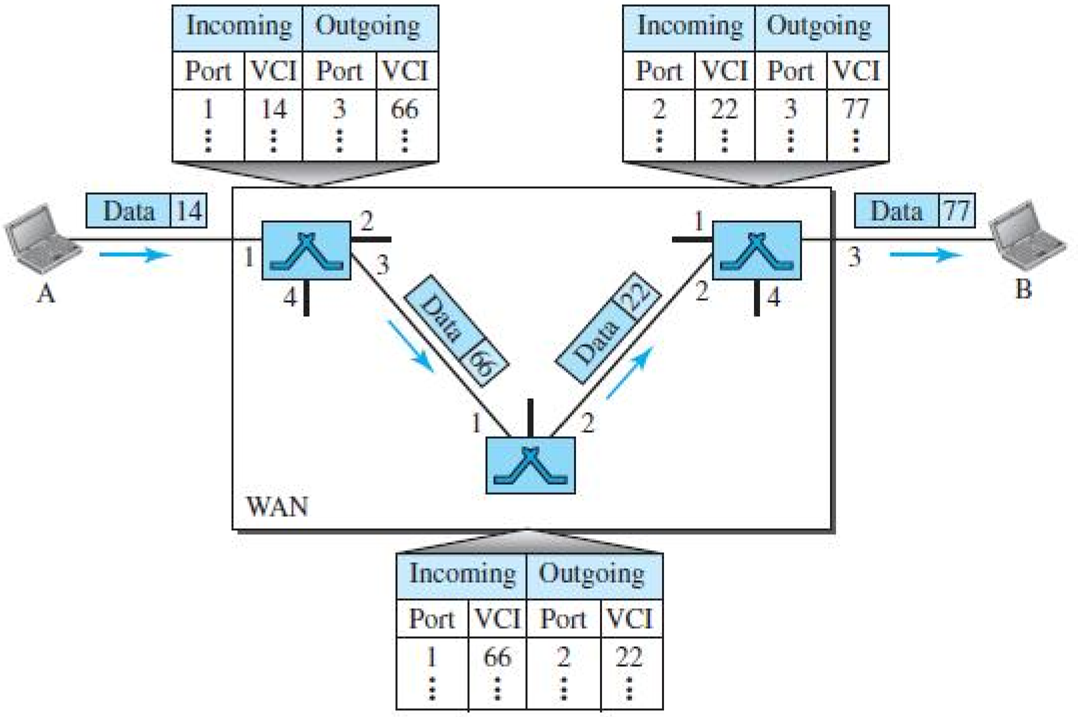
**Figure 8.15** *Setup acknowledgment in a virtual-circuit network*



| Incoming | | Outgoing | |
|---|---|---|---|
| Port | VCI | Port | VCI |
| 1 | 14 | 3 | 66 |

| Incoming | | Outgoing | |
|---|---|---|---|
| Port | VCI | Port | VCI |
| 2 | 22 | 3 | 77 |

| Incoming | | Outgoing | |
|---|---|---|---|
| Port | VCI | Port | VCI |
| 1 | 66 | 2 | 22 |



| Incoming | | Outgoing | |
|---|---|---|---|
| Port | VCI | Port | VCI |
| 1<br>⋮ | 14<br>⋮ | 3<br>⋮ | 66<br>⋮ |

| Incoming | | Outgoing | |
|---|---|---|---|
| Port | VCI | Port | VCI |
| 2<br>⋮ | 22<br>⋮ | 3<br>⋮ | 77<br>⋮ |

| Incoming | | Outgoing | |
|---|---|---|---|
| Port | VCI | Port | VCI |
| 1<br>⋮ | 66<br>⋮ | 2<br>⋮ | 22<br>⋮ |

g(x) = 1011 = $x^3+0+x^1+1$

dataword = dw = $x^{13}+0+x^{11}+0+0+x^8+x^7+x^6+x^5+0+0+0+0+0$

$x^{10}+0+0+x^7+0+x^3+x^2+x^2$

$x^3+0+x^1+1$ | $x^{13}+0+x^{11}+0+0+x^8+x^7+x^6+x^5+0+0+0+0+0$

$x^{12}+0+x^{11}+x^{10}$

$0+0+x^{10}+0$

$0+x^{10}+x^8+x^7$

~~$0+x^{10}+0+x^7$~~

~~$0+0+0+0$~~

$x^{10}+0+x^8+x^7$

$x^{10}+0+x^8+x^7$

$0+0+0+0+x^6+x^5+0+0$

$x^6+0+x^4+x^3$

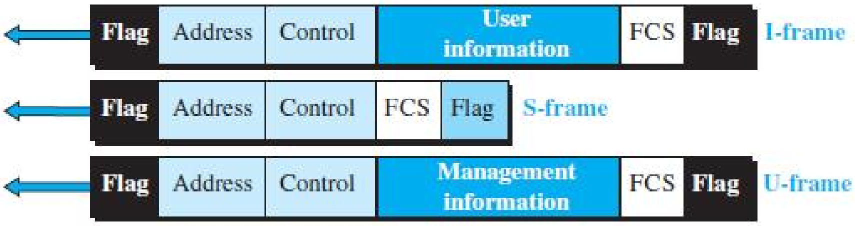
$x^5+x^4+x^3+0$

$x^5+0+x^3+x^2$

$x^4+0+x^2+0$

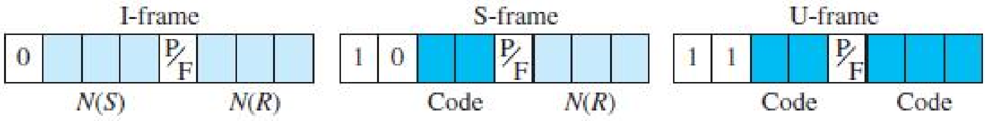
$x^4+0+x^2+0$

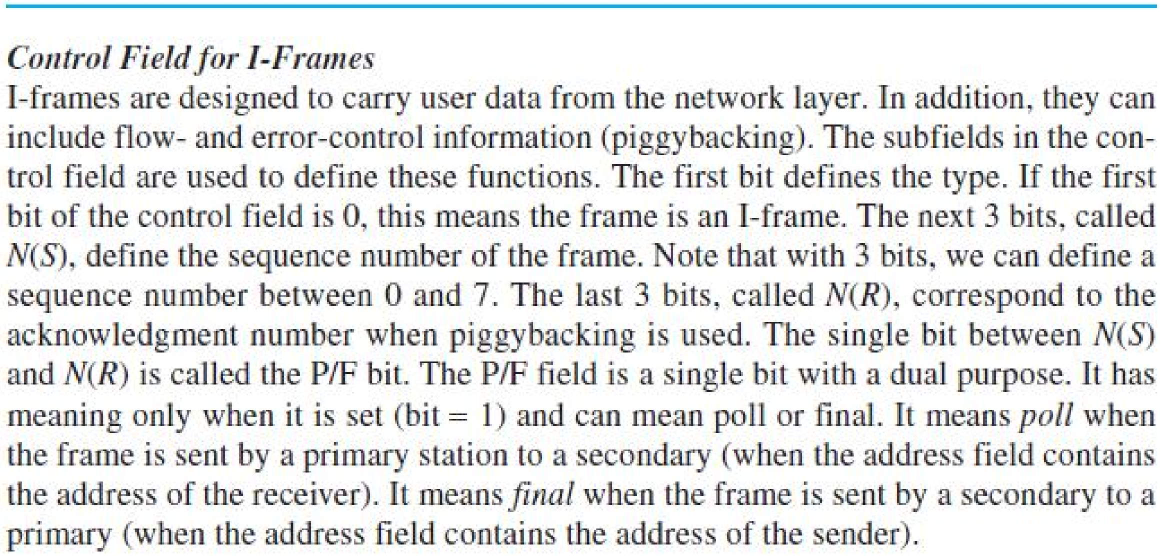
7.



Let us now discuss the fields and their use in different frame types.

- *Flag field.* This field contains synchronization pattern 01111110, which identifies both the beginning and the end of a frame.
- *Address field.* This field contains the address of the secondary station. If a primary station created the frame, it contains a *to* address. If a secondary station creates the frame, it contains a *from* address. The address field can be one byte or several bytes long, depending on the needs of the network.





### Control Field for I-Frames

I-frames are designed to carry user data from the network layer. In addition, they can include flow- and error-control information (piggybacking). The subfields in the control field are used to define these functions. The first bit defines the type. If the first bit of the control field is 0, this means the frame is an I-frame. The next 3 bits, called *N*(*S*), define the sequence number of the frame. Note that with 3 bits, we can define a sequence number between 0 and 7. The last 3 bits, called *N*(*R*), correspond to the acknowledgment number when piggybacking is used. The single bit between *N*(*S*) and *N*(*R*) is called the P/F bit. The P/F field is a single bit with a dual purpose. It has meaning only when it is set (bit = 1) and can mean poll or final. It means *poll* when the frame is sent by a primary station to a secondary (when the address field contains the address of the receiver). It means *final* when the frame is sent by a secondary to a primary (when the address field contains the address of the sender).

### Control Field for S-Frames

Supervisory frames are used for flow and error control whenever piggybacking is either impossible or inappropriate. S-frames do not have information fields. If the first 2 bits of the control field are 10, this means the frame is an S-frame. The last 3 bits, called *N*(*R*), correspond to the acknowledgment number (ACK) or negative acknowledgment number (NAK), depending on the type of S-frame. The 2 bits called *code* are used to define the type of S-frame itself. With 2 bits, we can have four types of S-frames, as described below:

***Receive ready (RR).*** If the value of the code subfield is 00, it is an RR S-frame. This kind of frame acknowledges the receipt of a safe and sound frame or group of frames. In this case, the value of the *N*(*R*) field defines the acknowledgment number.

***Receive not ready (RNR).*** If the value of the code subfield is 10, it is an RNR Sframe. This kind of frame is an RR frame with additional functions. It acknowledges the receipt of a frame or group of frames, and it announces that the receiver is busy and cannot receive more frames. It acts as a kind of congestion-control mechanism by asking the sender to slow down. The value of *N*(*R*) is the acknowledgment number.

***Reject (REJ).*** If the value of the code subfield is 01, it is an REJ S-frame. This is a NAK frame, but not like the one used for Selective Repeat ARQ. It is a NAK that can be used in Go-Back-*N* ARQ to improve the efficiency of the process by informing the sender, before the sender timer expires, that the last frame is lost or damaged. The value of *N*(*R*) is the negative acknowledgment number.

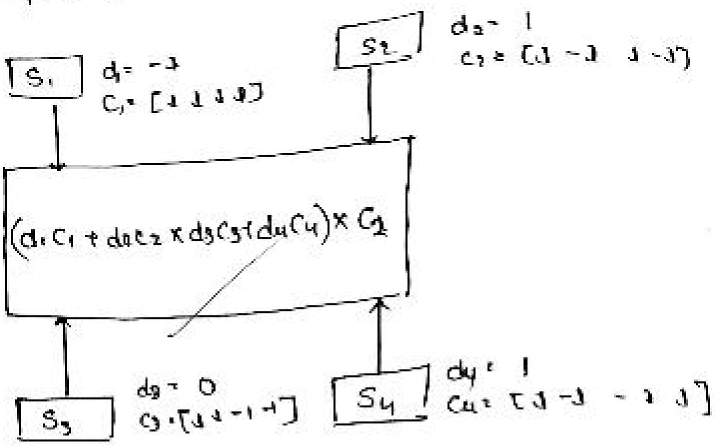
***Selective reject (SREJ).*** If the value of the code subfield is 11, it is an SREJ Sframe. This is a NAK frame used in Selective Repeat ARQ. Note that the HDLC Protocol uses the term *selective reject* instead of *selective repeat*. The value of *N*(*R*) is the negative acknowledgment number.

### Control Field for U-Frames

Unnumbered frames are used to exchange session management and control information between connected devices. Unlike S-frames, U-frames contain an information field, but one used for system management information, not user data. As with S-frames, however, much of the information carried by U-frames is contained in codes included in the control field. U-frame codes are divided into two sections: a 2-bit prefix before the P/F bit and a 3-bit suffix after the P/F bit. Together, these two segments (5 bits) can be used to create up to 32 different types of U-frames.

8. CDMA

## CDMA: code Concept division multiple access



Let us consider there are four stations namely $S_1, S_2, S_3, S_4$ which are connected to main system. The data representation of each system is given by

$0 \rightarrow -1$

$1 \rightarrow 1$

silent $\rightarrow 0$

so for $S_1$ the data will be $= -1$
code word will be $= [1\ 1\ 1\ 1]$

$S_2 \rightarrow$ data will be $= 1$
code will be $= [1\ -1\ 1\ -1]$

$S_3 \rightarrow$ data will be $= 0$

$$W_{2N} = \begin{bmatrix} w_N & w_N \\ w_N & \bar{w}_N \end{bmatrix} \quad [\ 1 \ -1\ ]$$

$$W_{4N} = \begin{bmatrix} w_N & w_N & w_N & w_N \\ w_N & \bar{w}_N & w_N & \bar{w}_N \\ w_N & w_N & \bar{w}_N & \bar{w}_N \\ w_N & \bar{w}_N & \bar{w}_N & w_N \end{bmatrix} = \begin{array}{c|c} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} & \begin{matrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{matrix} \end{array}$$

$c_1 = [1\ 1\ 1\ 1]$

$c_2 = [1\ -1\ \ 1\ -1]$

$c_3 = [1\ 1\ -1\ -1]$

$c_4 = [1\ -1\ -1\ 1]$

To callculate CDMA we need to multiply each data and code of each soure and and worth all remaning soure.
The answer will be multiplied by code from which the data will be sent

Then we need to devide the answer by 4 becaause no of sources are four

If the answer is same as data then we can say that the data will be send from the soure $S_2$

so

$[d_1c_1 + d_2c_2 + d_3c_3 + d_4c_4] \times c_2$

$[\ d_1c_1 = [-1\ -1\ -1\ -1]$
$d_2c_2 = [1\ -1\ \ 1\ -1]$
$d_3c_3 = [0\ \ 0\ 0\ 0]$
$d_4c_4 = [1\ -1\ -1\ 1]$

$= [-1\ -1\ -1\ -1] + [1\ -1\ \ 1\ -1] + [0\ 0\ 0\ 0]\ [1\ -1\ -1\ 1]$

$= [1\ \ -3\ \ -1\ \ -1]\ [1\ -1\ \ 1\ -1]$

$= [1\ \ 3\ \ -1\ \ 1]$

$= \quad 4$

so total soure are 4 so devide by four

$- \quad 4/4 = 1 = \underline{\underline{S_2(d_2)}}$