UNIT-II

1. Discuss the issues in the data link layer.

Answer:

Data Link Layer Design Issues:

The data link layer has a number of specific functions it can carry out. These functions include

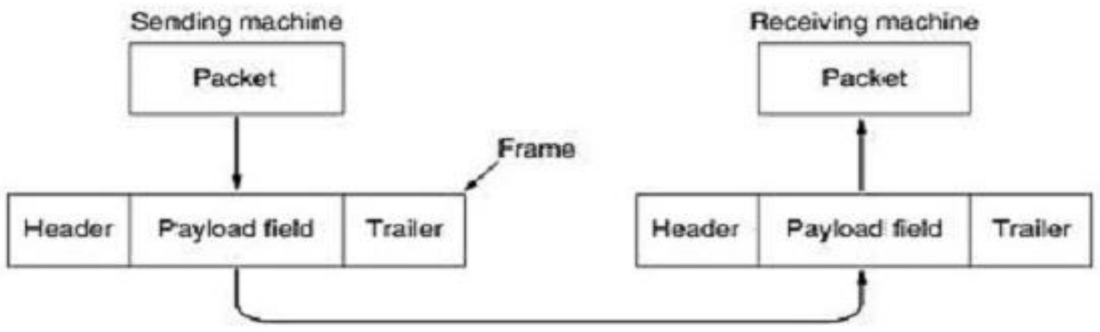
1. Providing a well-defined service interface to the network layer.
2. Dealing with transmission errors.
3. Regulating the flow of data so that slow receivers are not swamped by fast senders.

To accomplish these goals, the data link layer takes the packets it gets from the network layer and encapsulates them into frames for transmission. Each frame contains a frame header, a payload field for holding the packet, and a frame trailer, as illustrated in Fig.1. Frame management forms the heart of what the data link layer does.

In fact, in many networks, these functions are found only in the upper layers and not in the datalink layer. However, no matter where they are found, the principles are pretty much the same, so it does not really matter where we study them. In the data link layer they often show up in their simplest and purest forms, making this a good place to examine them.

2. Suppose we want to transmit the message 11001001 and protect it from errors using the CRC polynomial $x^3 + 1$. Use polynomial long division to determine the message that should be transmitted.

Answer:



(a) In this case, x^3+1 correspond to 1001. We multiply 11001001 since the degree of CRC is 3 and we get 11001001000.

```
            11010011
1001 / 11001001000
                  1001
                    1011
                    1001
                        1000
                        1001
                              1100
                              1001
                                1010
                                1001
                                  011
```

The remainder is 11, so we should send 11001001000 + 011 = 1100 1001 011

(b) The inverting of the first bit gives us 0100 1001 011. The receiver divides the message by 1001, which is x^3+1, and finally gets 01000001 with a remainder of 10. With the remainder occurring, the receiver will know that an error has occurred.

3. Discuss the principle of stop and wait flow control algorithm. Draw time line diagrams and explain how loss of a frame and loss of an ACK are handled. What is the effect of delay-bandwidth product on link utilization?

Answer:

* Stop and Wait Automatic Repeat Request :-

* First protocol is stop and wait Automatic Repeat Request.

* To detect and correct corrupted frames, we need to add redun-dancy bits to our data frame.

* When the frame arrives at the receiver site, it is checked and if it is corrupted, it is silently discarded

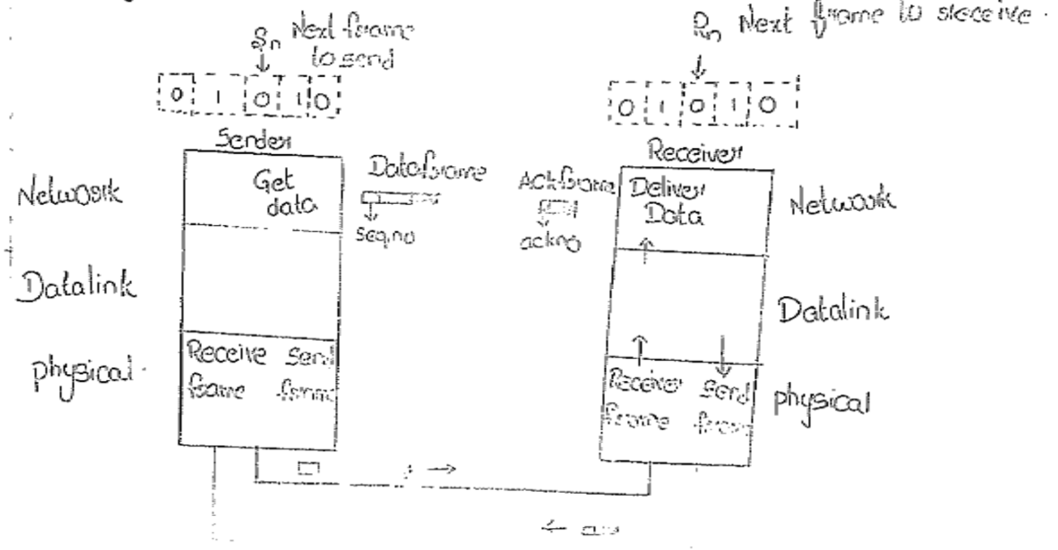
* The detection of errors in this protocol is manifested by the silence of the receiver.

* Error correction in stop and wait ARQ is done by keeping a copy of the sent frame and retransmitting of the frame when the timer expires.

* The protocol specifies that frames need to be numbered. This is done by using sequence numbers.

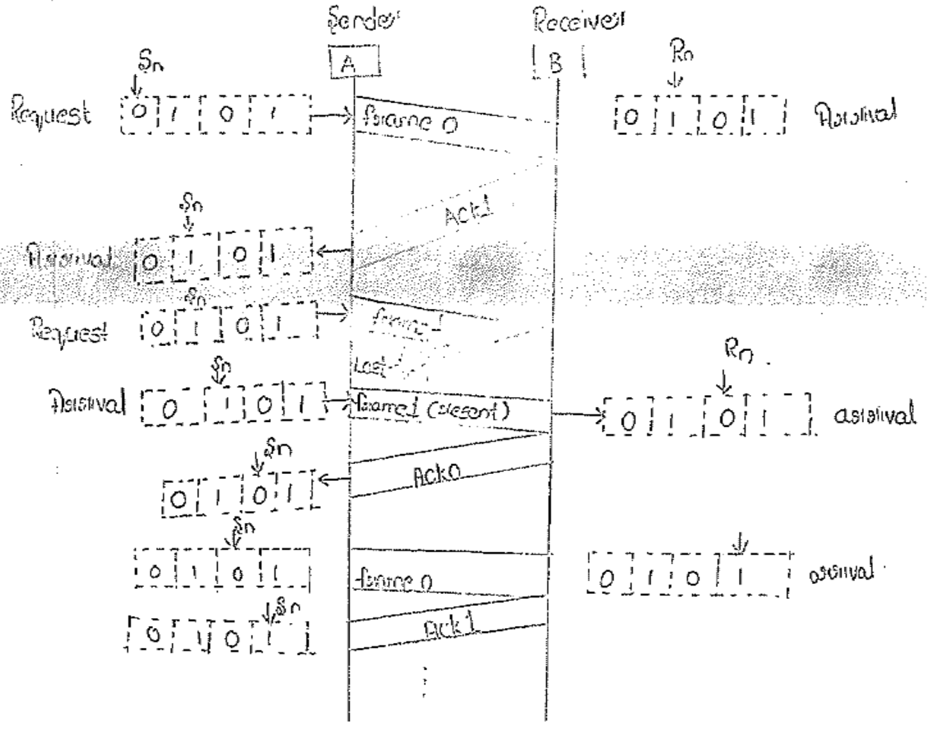
* Let us reason out of the range of sequence numbers we need. Assume we have used x as a sequence number, we only need

Design of the stop and wait ARQ protocol :-



* The acknowledgment numbers always announce the sequence number of the next frame expected by the receiver.

flow diagram :-



4. Assume that a frame consists of 6 characters encoded in 7-it ASCII. Attach a parity bit for every character to maintain even parity. Also attach a similar parity bit for each bit position across each of the bytes in the frame. Show that such a 2-dimensional parity scheme can detect all 1-bit, 2-bit and 3-bit errors and can correct a single bit error.

5. Explain the following Error Detection Mechanisms.
(i) Cyclic Redundancy Check.
(ii) Discuss briefly about link level flow control.

6. List the three main functions performed by the data link layer of the ISO OSI model.
Answer :

Data link layer is most reliable node to node delivery of data. It forms frames from the packets that are received from network layer and gives it to physical layer. It also synchronizes the information which is to be transmitted over the data. Error controlling is easily done. The encoded data are then passed to physical.
Error detection bits are used by the data link layer. It also corrects the errors. Outgoing messages are assembled into frames. Then the system waits for the acknowledgements to be received after the transmission. It is reliable to send message.

1. **Framing:** Frames are the streams of bits received from the network layer into manageable data units. This division of stream of bits is done by Data Link Layer.
2. **Physical Addressing:** The Data Link layer adds a header to the frame in order to define physical address of the sender or receiver of the frame, if the frames are to be distributed to different systems on the network.
3. **Flow Control:** A flow control mechanism to avoid a fast transmitter from running a slow receiver by buffering the extra bit is provided by flow control. This prevents traffic jam at the receiver side.
4. **Error Control:** Error control is achieved by adding a trailer at the end of the frame. Duplication of frames are also prevented by using this mechanism. Data Link Layers adds mechanism to prevent duplication of frames.
5. **Access Control:** Protocols of this layer determine which of the devices has control over the link at any given time, when two or more devices are connected to the same link.

7. State which layers of the ISO OSI model does the following interconnecting devices operate.

(1) Repeaters
(2) Bridges
(3) Routers
(4) Gateways

8. What is cyclic code and explain Cyclic Redundancy Check (CRC) code?
Answer:

Cyclic codes in Error Detection and Correction

Cyclic codes are special linear block codes with one extra property. In a cyclic code, if a code word is cyclically shifted (rotated), the result is another code word. For example, if 1011000 is a code word and we cyclically left-shift, then 0110001 is also a code word.
In this case, if we call the bits in the first word a0 to a6 and the bits in the second word b0 to b6, we can shift the bits by using the following:
b1=a0 b2=a1 b3=a2 b4=a3 b5=a4 b6=a5 b0=a6

Cyclic Redundancy Check:

One of the categories of cyclic codes called the cyclic redundancy check (CRC) that is used in networks such as LANs and WANs.
The following table shows an example of a CRC code which shows both the linear and cyclic properties of this code.

Dataword	*Codeword*	*Dataword*	*Codeword*
0000	0000000	1000	1000101
0001	0001011	1001	1001110
0010	0010110	1010	1010011
0011	0011101	1011	1011000
0100	0100111	1100	1100010
0101	0101100	1101	1101001
0110	0110001	1110	1110100
0111	0111010	1111	1111111

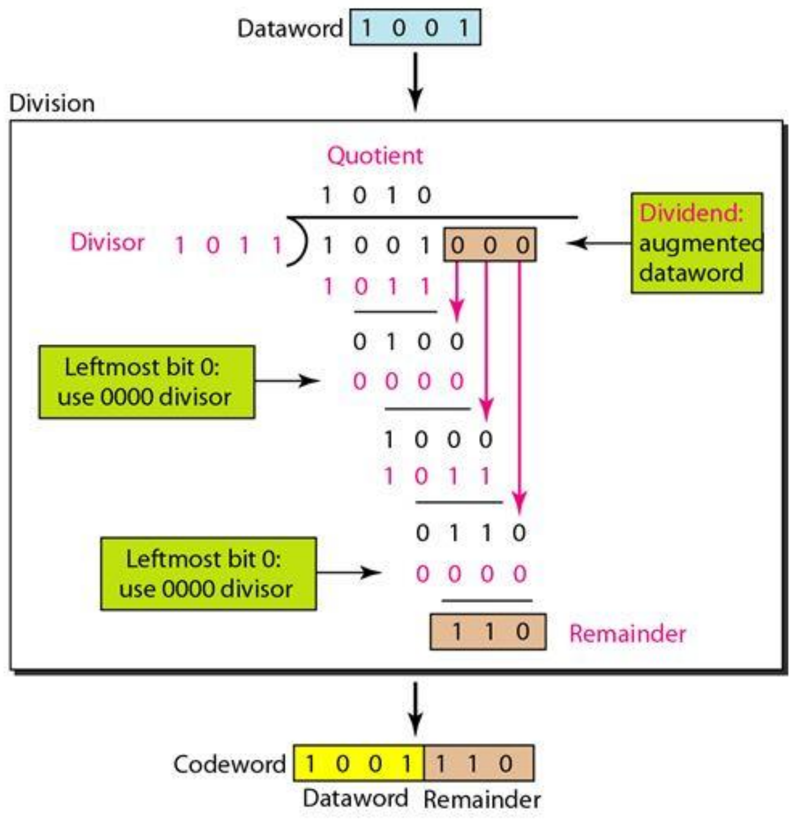
In the encoder, the data word has k bits (4 here); the code word has n bits (7 here). The size of the data word is augmented by adding n - k (3 here) 0s to the right-hand side of the word. The n-bit result is fed into the generator.

The generator uses a divisor of size n - k + I (4 here), predefined and agreed upon. The generator divides the augmented Data word by the divisor (modulo-2 division). The quotient of the division is discarded; the remainder (r2r1r0) is appended to the data word to create the code word.

The decoder receives the possibly corrupted code word. A copy of all n bits is fed to the checker which is a replica of the generator. The remainder produced by the checker is a syndrome of n - k (3 here) bits, which is fed to the decision logic analyzer.

The analyzer has a simple function. If the syndrome bits are all 0s, the 4 leftmost bits of the code word are accepted as the data word (interpreted as no error); otherwise, the 4 bits are discarded (error).

Encoder:

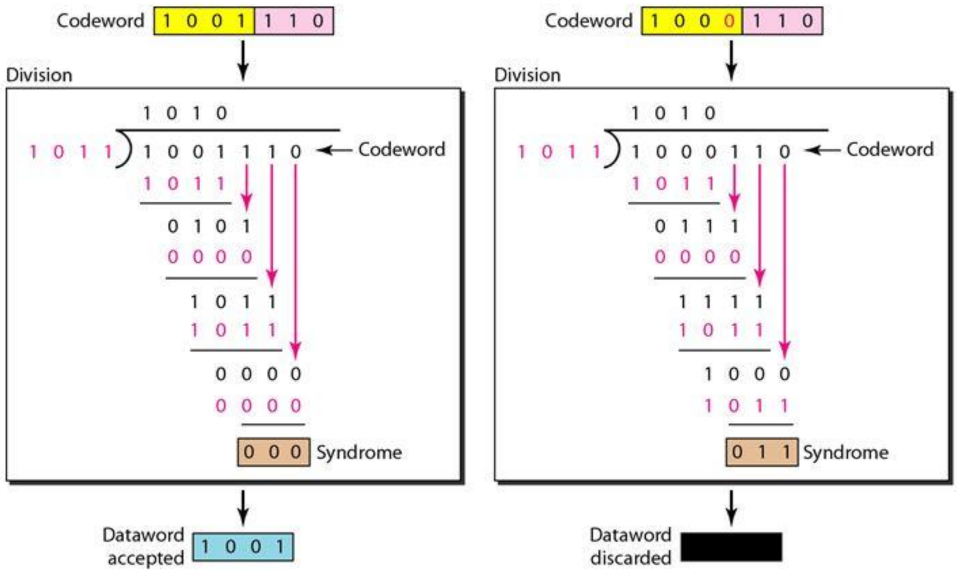


The process of modulo-2 binary division is the same as the familiar division process for decimal numbers. In each step, a copy of the divisor is XORed with the 4 bits of the dividend. The result of the XOR operation (remainder) is 3 bits (in this case), which is used for the next step after 1 extra bit is pulled down to make it 4 bits long. There is one important point we need to remember in this type of division. If the leftmost bit of the dividend (or the part used in each step) is 0, the step cannot use the regular divisor; we need to use an all-0s divisor.

When there are no bits left to pull down, we have a result. The 3-bit remainder forms the check bits (r2, r1 and r0). They are appended to the data word to create the code word.

Decoder:

The code word can change during transmission. The decoder does the same division process as the encoder. The remainder of the division is the syndrome. If the syndrome is all 0s, there is no error; the data word is separated from the received code word and accepted. Otherwise, everything is discarded. Consider the following figure which shows two cases: The left hand figure shows the value of syndrome when no error has occurred; the syndrome is 000. The right-hand part of the figure shows the case in which there is one single error. The syndrome is not all 0s (it is 011).



For example, consider the following figure where the encoder takes the data word and augments it with n - k number of 0s. It then divides the augmented data word by the divisor.

9. What is meant by linear Block Code and explain Simple Parity-Check Code?

Answer:

Simple Parity Check Code

Linear Block Codes:

A linear block code is a code in which the exclusive OR (addition modulo-2) of two valid code words creates another valid code word.

The scheme in the above table is a linear block code because the result of XORing any code word with any other code word is a valid code word. For example, the XORing of the second and third code words creates the fourth one.

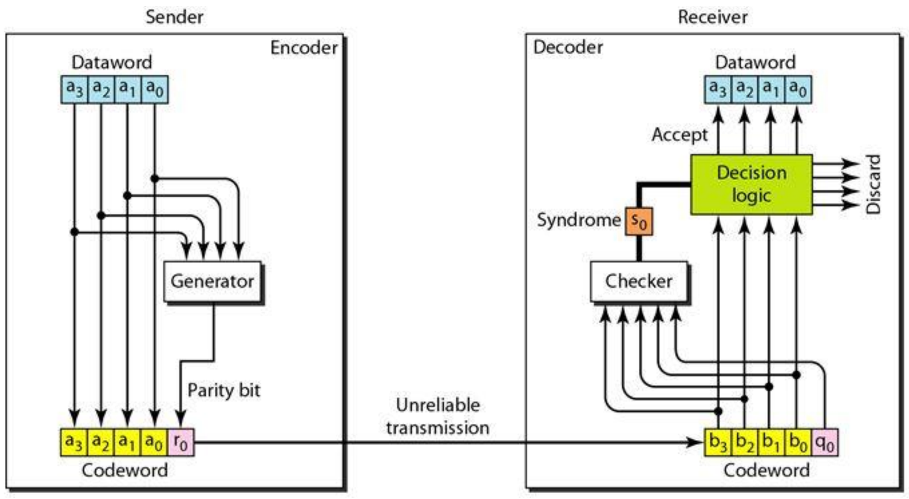
Minimum Distance for Linear Block Codes:

It is simple to find the minimum Hamming distance for a linear block code. The minimum Hamming distance is the number of 1s in the nonzero valid code word with the smallest number of 1s. In the above table the numbers of 1s in the nonzero code words are 2, 2, and 2. So the minimum Hamming distance is dmin =2.

Simple Parity-Check Code:

The simple parity-check code is the most familiar error-detecting code. In this code, a k-bit data word is changed to an n-bit code word where n = k + 1. The extra bit, called the parity bit, is selected to make the total number of 1s in the code word even. Although some implementations specify an odd number of 1s. The minimum Hamming distance for this category is dmin =2, which means that the code is a single-bit error-detecting code and it cannot correct any error.

The following figure shows possible structure of an encoder (at the sender) and a decoder (at the receiver).



The encoder uses a generator that takes a copy of a 4-bit data word (a0, a1, a2 and a3) and generates a parity bit r0. The data word bits and the parity bit create the 5-bit code word. The parity bit that is added makes the number of 1s in the code word even.

This is normally done by adding the 4 bits of the data word (modulo-2).

r0=a3+a2+a1+a0 (modulo-2)

The result is the parity bit. In other words, If the number of 1s is even, the result is 0; if the number of 1s is odd, the result is 1.In both cases, the total number of 1s in the code word is even.

The sender sends the code word which may be corrupted during transmission. The receiver receives a 5-bit word. The checker at the receiver does the same thing as the generator in the sender with one exception: The addition is done over all 5 bits.

s0=b3+b2+b1+b0+ q0 (modulo-2)

The result, which is called the syndrome, is just 1 bit. The syndrome is 0 when the number of 1s in the received code word is even; otherwise, it is 1.

The syndrome is passed to the decision logic analyzer. If the syndrome is 0, there is no error in the received code word, the data portion of the received code word is accepted as the data word, if the syndrome is 1, the data portion of the received code word is discarded. The data word is not created.

For example, the sender sends the data word 1011. The code word created from this data word is 10111, which is sent to the receiver. We examine five cases:

1. No error occurs; the received code word is 10111. The syndrome is 0. The data word 1011 is created.

2. One single-bit error changes a1. The received code word is 10011. The syndrome is 1. No data word is created.

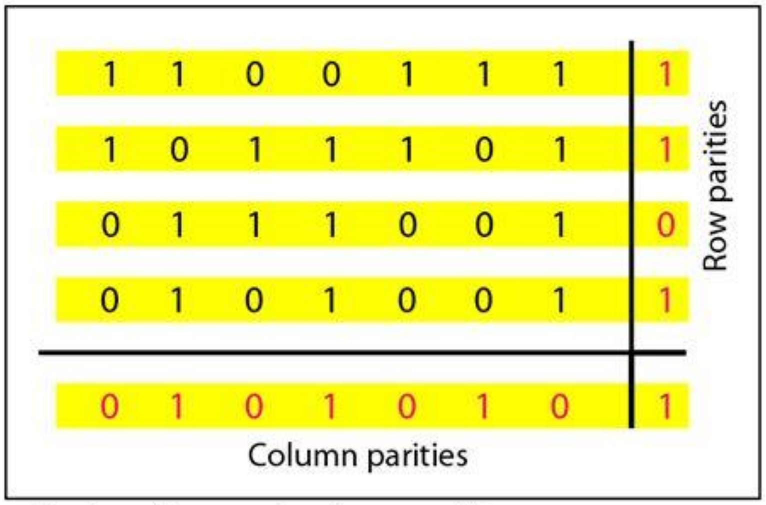
3. One single-bit error changes r0. The received code word is 10110. The syndrome is 1. No data word is created. Note that although none of the data word bits are corrupted, no data word is created because the code is not sophisticated enough to show the position of the corrupted bit.

4. An error changes r0 and a second error changes a3. The received code word is 00110. The syndrome is O. The data word 0011 is created at the receiver. Note that here the data word is wrongly created due to the syndrome value. The simple parity-check decoder cannot detect an even number of errors. The errors cancel each other out and give the syndrome a value of O.

5. Three bits-a3, a2, and a1-are changed by errors. The received code word is 01011. The syndrome is 1. The data word is not created. This shows that the simple parity check, guaranteed to detect one single error, can also find any odd number of errors.

A better approach is the two-dimensional parity check. In this method, the data word is organized in a table (rows and columns). In the following figure, the data to be sent, five 7-bit bytes, are put in separate rows. For each row and each column, 1 parity-check bit is calculated. The whole table is then sent to the receiver, which finds the syndrome for each row and each column.

As shown in the figure, the two-dimensional parity check can detect up to three errors that occur anywhere in the table (arrows point to the locations of the created nonzero syndromes). However, errors affecting 4 bits may not be detected.



							Row parities
1	1	0	0	1	1	1	1
1	0	1	1	1	0	1	1
0	1	1	1	0	0	1	0
0	1	0	1	0	0	1	1
0	1	0	1	0	1	0	1

Column parities

a. Design of row and column parities