UNIT-2

- The data link layer needs to pack bits into frames, so that each frame is distinguishable from another.
- Framing in the data link layer separates a message from one source to a destination, or from other messages to other destinations, by adding a sender address and a destination address.
- The destination address defines where the packet is to go; the sender address helps the recipient acknowledge the receipt.

FRAMING:

→FIXED SIZE FRAMING:

**Framing is of 2 types :1-fixed size framing 2-variable size framing.

- In fixed size framing , there is no need to define boundaries of a frames.
- The size itself can be used as a delimiter.
- An example of this network is ATM wide area network,which uses frames of fixed size called cells.

→VARIABLE SIZE FRAMING:

- Variable-size framing is prevalent in LAN.
- In this framing , we need to define a way to determine the beginning and end of the frame.
- There are two approaches were used for this purpose.
  - Character-oriented approach
  - Bit oriented approach

→CHARACTER-ORIENTED PROTOCOLS:

- In a character-oriented protocol,data to be carried are 8-bit characters from a coding system such as ASCII.
- The header, which normally carries the source and destination addresses and other control information.
- The trailer, which carries error detection or error correction redundant bits, are also multiples of 8 bits.
- To separate one frame from another an 8-bit flag is added at the beginning and end of the frame.
- The format of frame is as follows:

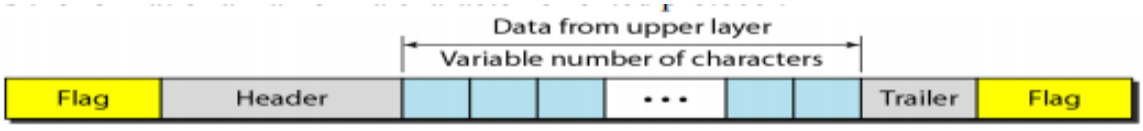
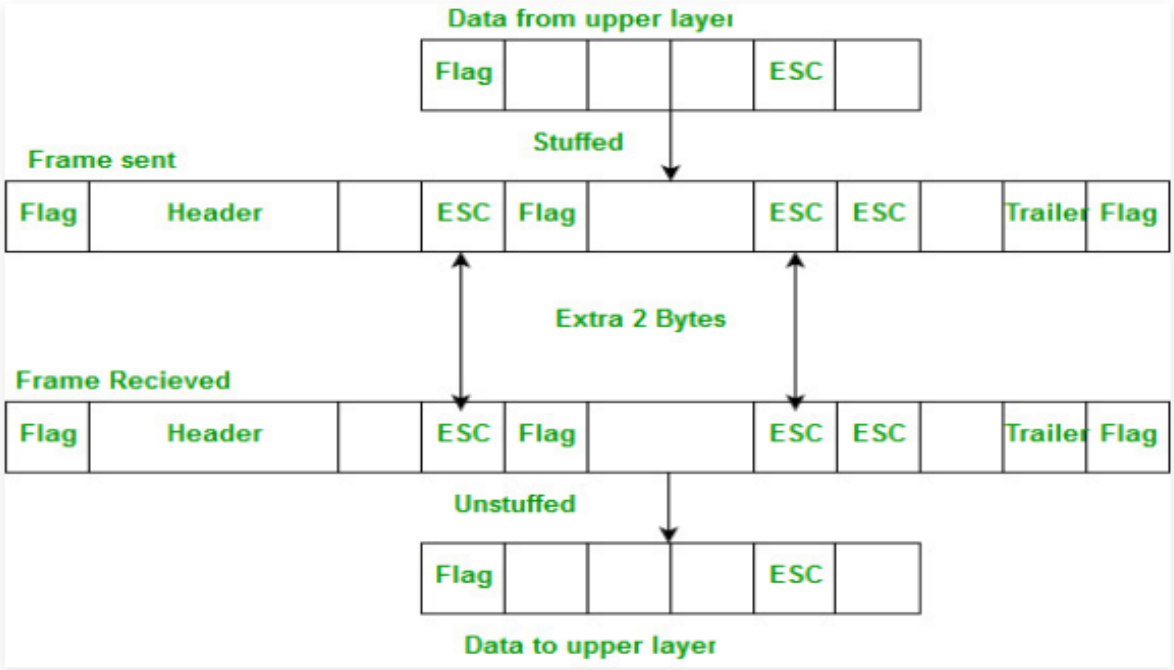


Figure: *A frame in a character-oriented protocol*

- So,If the data itself contains flag then the receiver may encounter it as the end of the frame.To overcome this a concept called byte stuffing is added to character-oriented approach.

→BYTE STUFFING AND UNSTUFFING:

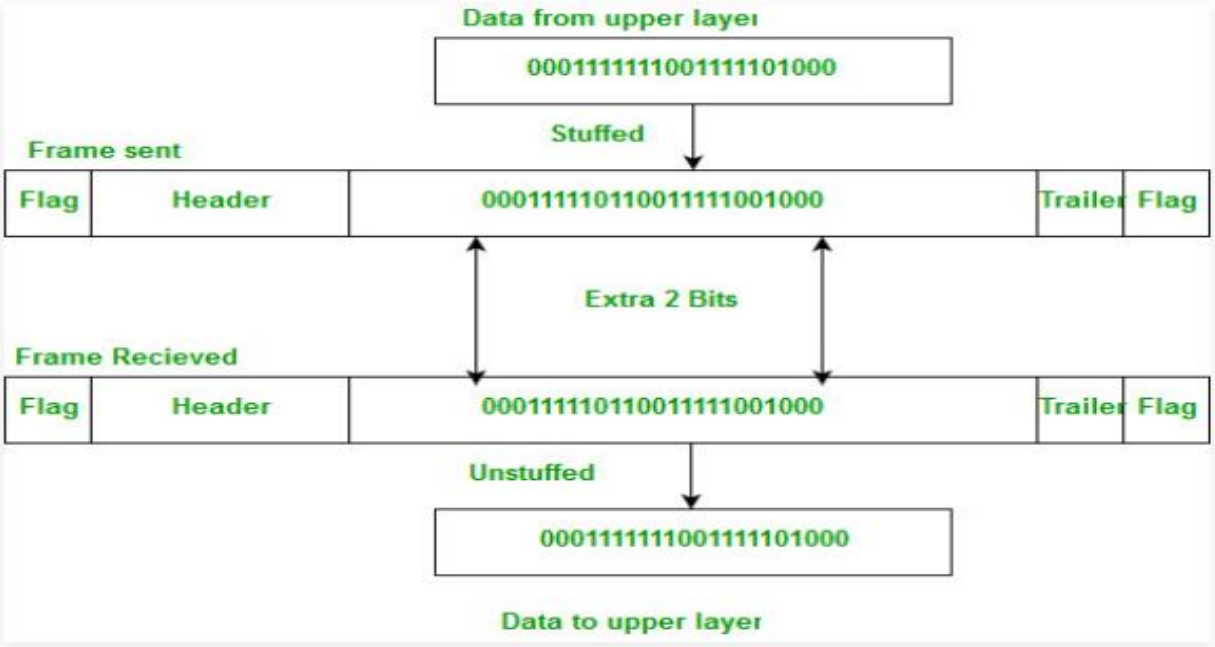
- Byte stuffing is the process of adding 1 extra byte whenever there is a flag or escape(esc) character in the set i.e., data.
- A character byte called esc is added before the flag or esc sequence..
- Consider the example,



- In this example, the data that is send to receiver is flag and esc so before the receiver receives any error a byte called esc is inserted during stuffing.
- So , When unstuffed the 2 inserted esc characters are skipped , and the data after the esc sequence is considered as the data . Then the data received is the actual data that is send by the sender.

→BIT-ORIENTED APPROACH:

- Generally each frame starts and end with a stream of bits called 01111110(called flag byte).
- So , If there is a data which contains 01111110 then there is a chance for the receiver to receive an error.
- Whenever sender datalink's layer encounters 5 consecutive 1's in the data then it automatically stuff's 0 into the outgoing bit stream.
- When the receiver sees five consecutive incoming 1 bits, followed by a 0 bit, it automatically destuffs (i.e., deletes) the 0 bit.
- If the user data contain the flag pattern, 01111110, this flag is transmitted as 011111010 but stored in the receiver's memory as 01111110.
- Consider the example,



- So, in this example the data in the sender side contains 5 consecutive 1's so a bit 0 is stuffed in the frame.
- When the frame is received by the receiver side after the 5 consecutive 1's the 0 is unstuffed(deleted) and the bits after that 0 is considered as data.

→FLOW AND ERROR CONTROL:

- Data Communication atleast requires 2 systems to communicate..one to send data and one to receive..
- So , This requires a great deal of coordination to transmit and receive correct data..
- The most important responsibilities of the data link layer are flow control and error control

FLOW CONTROL:

- Flow control is a set of procedures that tells the sender how much data it can transmit before it must wait for an acknowledgment from the receiver.
- Any receiving device has a limited speed at which it can process incoming data and a limited amount of memory in which to store incoming data.
- The receiving device must be able to inform the sending device before those limits are reached and to request that the transmitting device send fewer frames or stop temporarily.
- Incoming data must be checked and processed before they can be used.
- The rate of such processing is often slower than the rate of transmission.
- For this reason, each receiving device has a block of memory, called a buffer, reserved for storing incoming data until they are processed.
- If the buffer begins to fill up, the receiver must be able to tell the sender to halt transmission until it is once again able to receive.

ERROR CONTROL:

- Error control is both error detection and error correction.
- In error detection or control the receiver sends the sender the details of lost frames and asks the sender to retransmit the frames.
- In the data link layer, the term error control refers primarily to methods of error detection and retransmission.
- Error control in the data link layer is often implemented simply: Any time an error is detected in an exchange, specified frames are retransmitted. This process is called **automatic repeat request (ARQ).**

→ERROR DETECTION ERROR CORRECTION:

TYPES OF ERRORS:

- **Single-bit error:** The term single-bit error means that only 1 bit of a given data unit is changed from 1 to 0 or from 0 to 1.

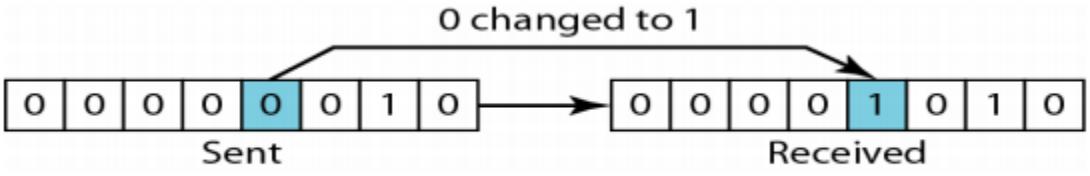


Figure: ***Single-bit error***

- **Burst error:** The term burst error means that 2 or more bits in the data unit have changed from 1 to 0 or from 0 to 1.

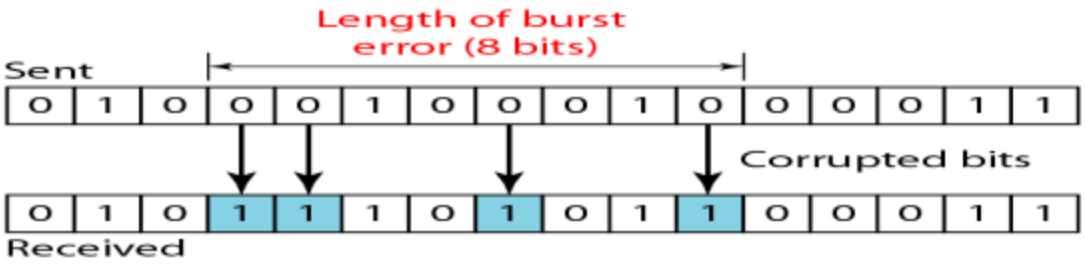


Figure: ***Burst error of length 8***

REDUNDANCY:

- The central concept in detecting or correcting errors is redundancy.
- To detect or correct errors, we need to send some extra bits with our data.
- These redundant bits are added by the sender and removed by the receiver. Their presence allows the receiver to detect or correct corrupted(error) bits.

ERROR DETECTION:

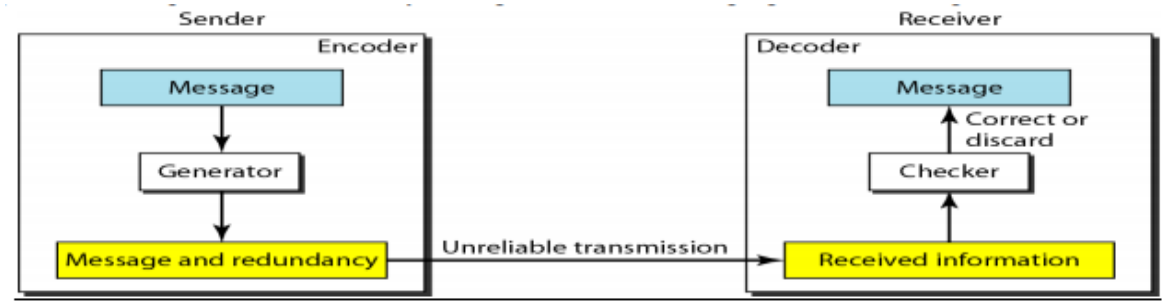
- In Error detection we are looking forward just to see if error has occurred or not.
- The simple is 'yes' or 'no'.

ERROR CORRECTION:

- The correction of error is more difficult than detection.In error correction we need to know the exact number of bits we have corrected and more importantly their location in the message.
- Two methods of error correction:
  - **Forward error correction :** Here , receiver tries to guess the , message by redundant bits . It is possible if number of errors are small.
  - **Correction by retransmission :** Receiver detects the occurence of an error and asks the sender to resend the frame . This process is repeated until the message received is 'error free?'

CODING:

- Redundancy is achieved through various coding schemes.
- The sender adds redundant bits through a process that creates a relationship between the redundant bits and the actual data bits.
- Following figure shows the general idea ABOUT THE STRUCTURE OF Encoder and Decoder.



- Coding schemes can be divided into two broad categories: **Block Coding** and **Convolution Coding**.

BLOCK CODING:

- This type of coding is used to detect and correct errors when the data received by the receiver is wrong.
- Assume the length of the data word be 'k' bits.
- And length of the code word is 'n' bits,to obtain code words so no.of redundant bits 'r' are added.

  n=k+r

- So,the no.of datawords formed are 2^k where k is the no.of bits in the dataword.
- As usual there occurs 2^n code words where n is the no.of bts in code word.
- The conditions that are remembered are n>k and 2^n-2^k code words are not used.

→ERROR DETECTION:

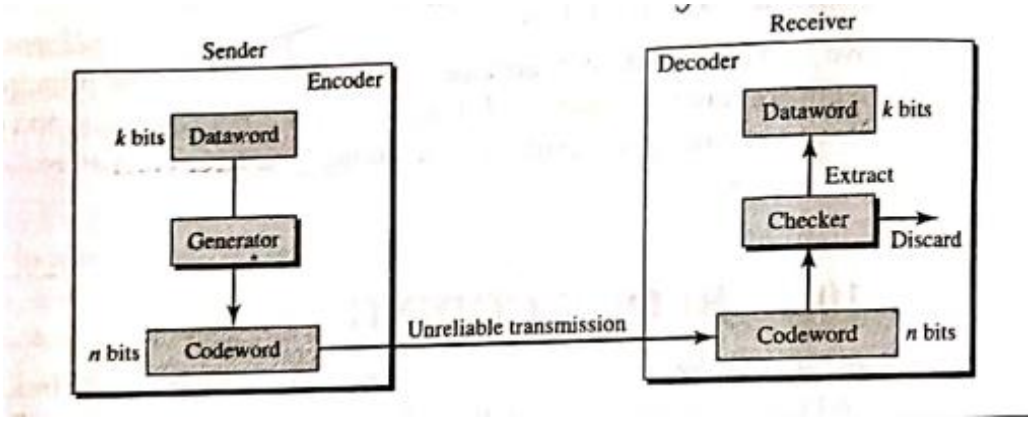
- Generally to obtain parity we opt for a procedure called "even parity" which contains even no.of 1's in the codeword.
- Table:Represents the error detection mechanism in Block coding

Datawords	*Codewords*
00	000
01	011
10	101
11	110

- So ,Here in this example a dataword of 2 bits is considered so k=2.
- And codeword of 3 bit is considered satisfying the conditions.
- Redundant bits are calculated using event parity.

CODEWORD	USED OR UNUSED
000	Used
001	Unused
010	Unused
011	Used
100	Unused
101	Used
110	Used
111	unused

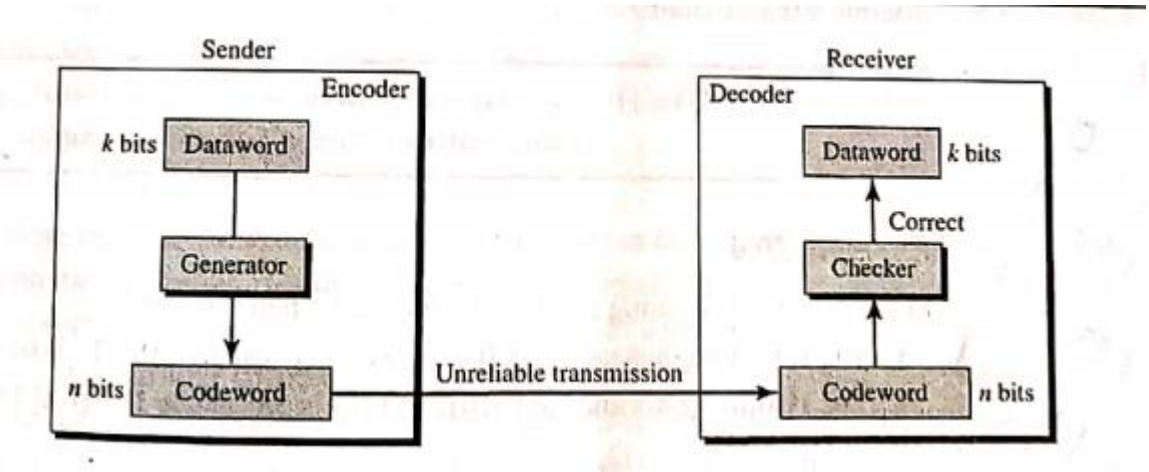
- The process of error detection is as follows:



- So, a data word on the sender site is send to generator so the redundant bits are added and the code word is sent to receiver.
- Then the code word on the receiver site is sent to checker to check for a valid codeword i.e., used code word.
- If the codeword is a valid code word then the data in the code word is extracted otherwise the code word is discarded.
- In error detection of this block codes there are some cases to be worked with:--Assume the dataword to be sent is 01 so the corresponding codeword is 011.
- Case1:If the receiver receives 011 codeword as it is a valid codeword the data is extracted and the 01and data is accepted.
- Case2:If the code word received is 111 it is not a valid codeword so it is discarded.
- Case 3:If the code word received is 000 which is valid codeword but the code word that is sent by the receiver is 011.Though the data from the 000 is accepted and data is extracted.

→ERROR CORRECTION:

- The process of error correction is as follows:



- So, In the mechanism of correcting errors in this block codes. There occurs a process or method called "guessing method" The corrupted code word is compared with valid code words and the code word with least no.of differences with the valid codewords is considered to be correct code word.
- Consider the data table which contains data words and the valid code words:

Dataword	*Codeword*
00	00000
01	01011
10	10101
11	11110

- Let us see this in detail with an example:
- Let us assume the dataword that is sent is 01 so the corresponding codeword is 01011.
- But the received codeword is 01001 which is a corrupted one and needs to be corrected.
- Case1:Compare 01001 with first valid code word 00000.

0 1 0 0 1

0 0 0 0 0

So, there is a change in 2 bits of data

Therefore 2.

- Case 2:Compare the codeword 01001 with 01011

0 1 0 1 1

0 1 0 0 1

So, there is a change in only one bit in the code words
Therefore it is 1.

- Case 3:Compare the codeword 01001 with 10101

0 1 0 0 1

1 0 1 0 1

So, there is a change in three bits in the code words
Therefore it is 3.

- Case 4: Compare 01001 with 11110.

0 1 0 0 1

1 1 1 1 0

So, there is a change in four bits in the code words
Therefore it is 4

- So , from these cases the minimum no of differences between 2 code words is the case 2.So the codeword of cas2 is considered and considering it to be the correct codeword it extracts data from the code word.

→HAMMING DISTANCE:

- This hamming distance is used for error control in the data.
- The hamming distance is nothing but the number of differences between the corresponding bits of two datawords.
- The hamming distance is easily calculated using XOR operation between the datawords.
- The no.of 1's determine the result.
- It is represented in the form of d(x,y).
- Assume the datawords are 000 and 111 to be compared. The hamming distance between these datawords is 3 since the XOR operation between these datawords is 111.

→MINIMUM HAMMING DISTANCE:

- The minimum hamming distance is the smallest hamming distance between all possible pairs.
- We use d_{min} to specify the minimum hamming distance.
- Consider the table.

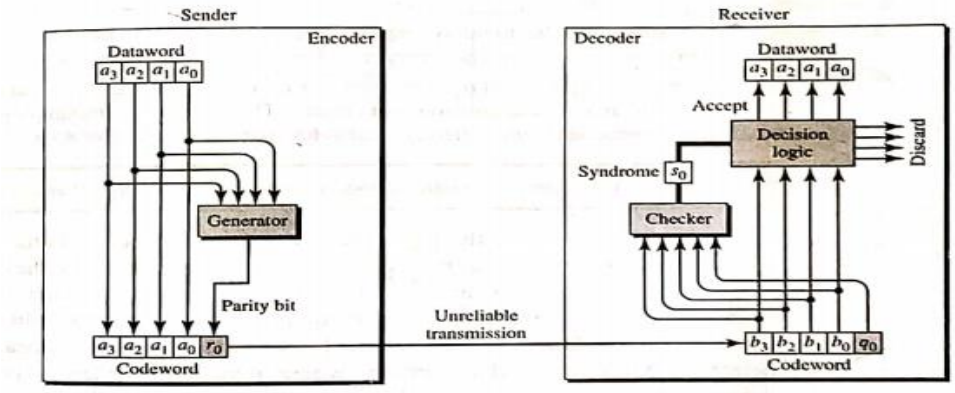
Datawords	Codewords
00	000
01	011
10	101
11	110

- So now calculate the minimum hamming distance for the set of datawords.
- d(000,011)=011 so the distance is 2.
- d(000,101)=101 so the distance is 2.
- D(000,110)=110 so the distance is 2.
- D(011,101)=110 so distance is 2.
- D(011,110)=101 so the distance is 2
- D(101,110)=001 so the distance is 2
- So the minimum hamming distance that is possible for the taken table is 2.

→LINEAR CODES:

SIMPLE PARITY CHECKER:

- A linear block code is a block code where the exclusive-OR of 2 codewords is a valid codeword.
- So, one type of this linear coding is simple parity checker.
- In this code ‘k’ is length of dataword ‘n’ is length of codeword where n=k+1 and that redundant bit is considered by even parity.
- The minimum hamming distance considered for this case is 2.
- So it can correct an single-bit error and not any error.
- The structure of encoder and decoder is as folloes:



- Consider the table which contains 4 bitdatawords and 5 bit code words.

Datawords	Codewords	Datawords	Codewords
0000	00000	1000	10001
0001	00011	1001	10010
0010	00101	1010	10100
0011	00110	1011	10111
0100	01001	1100	11000
0101	01010	1101	11011
0110	01100	1110	11101
0111	01111	1111	11110

- As dataword length is 4 it accepts 16 combinations and therefore 16 codewords of each 5 bit length.
- And Generally the redundant bit is added using modulo-2 addition

  R0=A0+A1+A2+A3
- Now the sender sends a codeword and is corrupted during the transmission.
- The receiver receives a codeword, and to check whether is a correct code word to be received it performs the same operation as the sender site i.e.., It performs he modulo-2 addition for the 5 bits in the codeword that is received.
- There is a bit called Syndrome on the receiver site which is used to determine the received codeword is valid or not.
- So,If the syndrome bit is 0 the no.of 1's in the codeword is even which says no error condition and if it is 1 the no.of 1's in the codeword is odd says error conditions.

  S=B0+B1+B2+B3+Q0
- Assume the dataword that has to be sent by the sender is 1011 so the corresponding codeword is 10111..
- So there are some cases to be studied to detect errors.
- Case1:Let the received codeword be 10111

  The receiver calculates s value which is 0 determing no error.
- Case2:Let the received codeword be 10011

  The receiver calculates the syndrome value and is= 1

  As it determines an error the codeword is discarded and no dataword is taken.
- Case3: Let the received code word be 10110.

  The receiver calculates the syndrome value and is=1

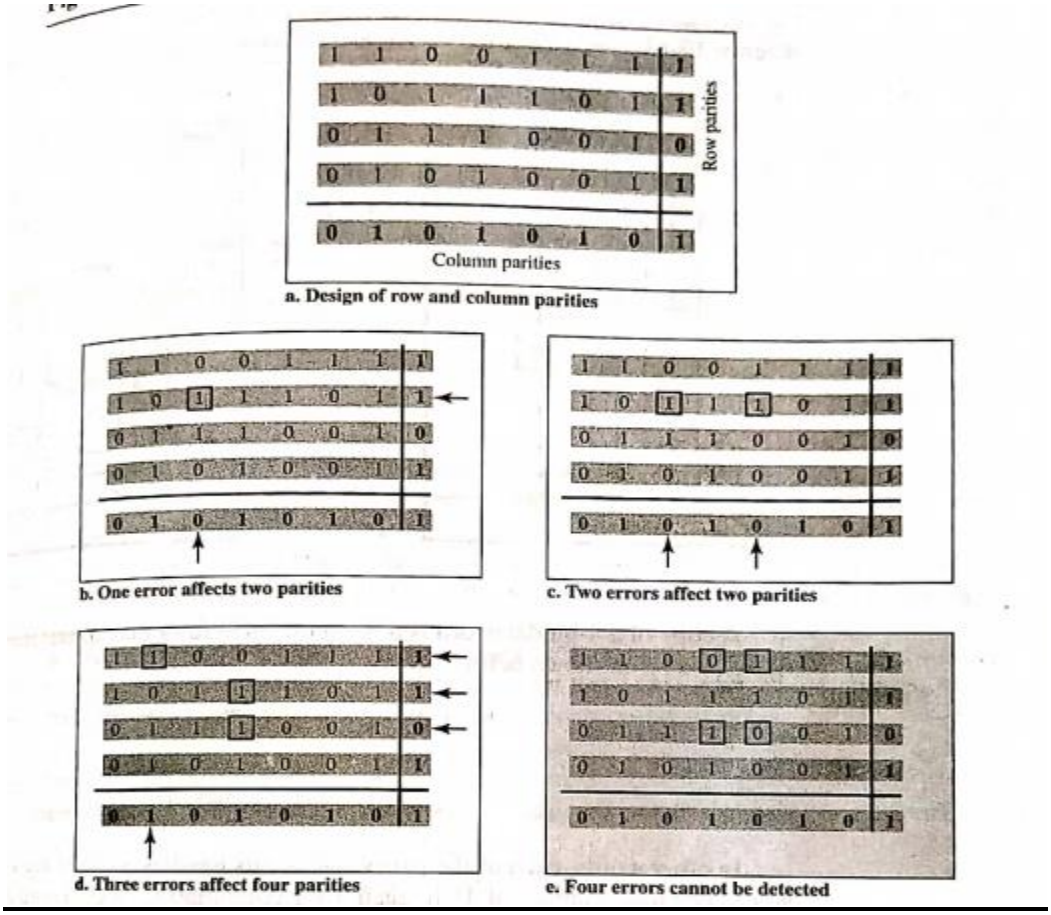
  So, the codeword is discarded and data is taken.
- Case4:Let the received code word be 00110

The receiver checks the syndrome bit which is 0.
Though its an error it is considered to be valid codeword and the data
. of 00110 is accepted.

- Case5:Let the code word received be 01011
  The syndrome bit is calculated and the value is 1
  So the codeword is discarded and data is not considered.
- **The simple parity check codes cannot detect even no.of errors.**
- **The simple parity check detects one single error and also odd no.of errors.**

→TWO DIMENSIONAL PARITY CHECK CODE:

- This two dimensional parity check is an error detection and error correction code.
- This parity check code uses minimum hamming distance is 3 .So, it detects 3 errors and corrects single error.
- Let 'k' be the length of dataword and 'n' be the length of the codeword and assume an integer 'm' whose values are greater or equal to 3.
- So, the n becomes 2^m-1.



a. Design of row and column parities

b. One error affects two parities

c. Two errors affect two parities

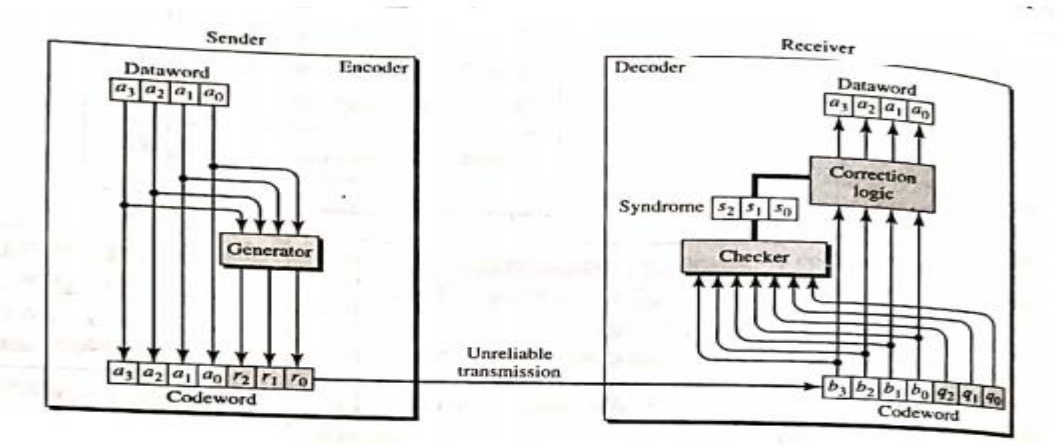
d. Three errors affect four parities

e. Four errors cannot be detected

- The above diagram tells about the number of errors and number of bits that are affected by the error.
- In the fig.a., The row and columns are generated using even parities.
- In fig(b), The bit in the 2nd row and 3rd column is error and so it effects the two parity bits.
- In the same way two errors in the same row/column affects 2 parity bits.
- Three errors affect 4 parity bits.
- Four errors in the pattern given in the fig(d) cannot be detected because they doesn't change or affect the parity bits.
- The table determines the 4 bits datawords and the corresponding codewords.

Datawords	Codewords	Datawords	Codewords
0000	0000000	1000	1000110
0001	0001101	1001	1001011
0010	0010111	1010	1010001
0011	0011010	1011	1011100
0100	0100011	1100	1100101
0101	0101110	1101	1101000
0110	0110100	1110	1110010
0111	0111001	1111	1111111

- The diagram represents the structure of encoder and decoder of two parity checker.



- When a dataword is sent by the sender the corresponding code word is generated by adding the redundant bits. The formulas of redundant bits are as follows.

$$r_0 = a_2 + a_1 + a_0 \quad \text{modulo-2}$$
$$r_1 = a_3 + a_2 + a_1 \quad \text{modulo-2}$$
$$r_2 = a_1 + a_0 + a_3 \quad \text{modulo-2}$$

- When the code word is received by the receiver it sends the code word to checker which checks the code word as valid or not by calculating syndrome bits.The formula of syndrome bits are as follows:

$$s_0 = b_2 + b_1 + b_0 + q_0 \quad \text{modulo-2}$$
$$s_1 = b_3 + b_2 + b_1 + q_1 \quad \text{modulo-2}$$
$$s_2 = b_1 + b_0 + b_3 + q_2 \quad \text{modulo-2}$$

- If all the 3 bits of syndrome are 0 then there is no error in the codeword received by the received by the receiver.If any bit in the syndrome is 1 then it determines the condition of error.
- The table represents the Logical decision made by the correction logic analyserof the decoder.

SYNDROME	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
ERROR	NONE	q0	q1	b2	q2	b0	b3	b1

→CYCLIC CODES:

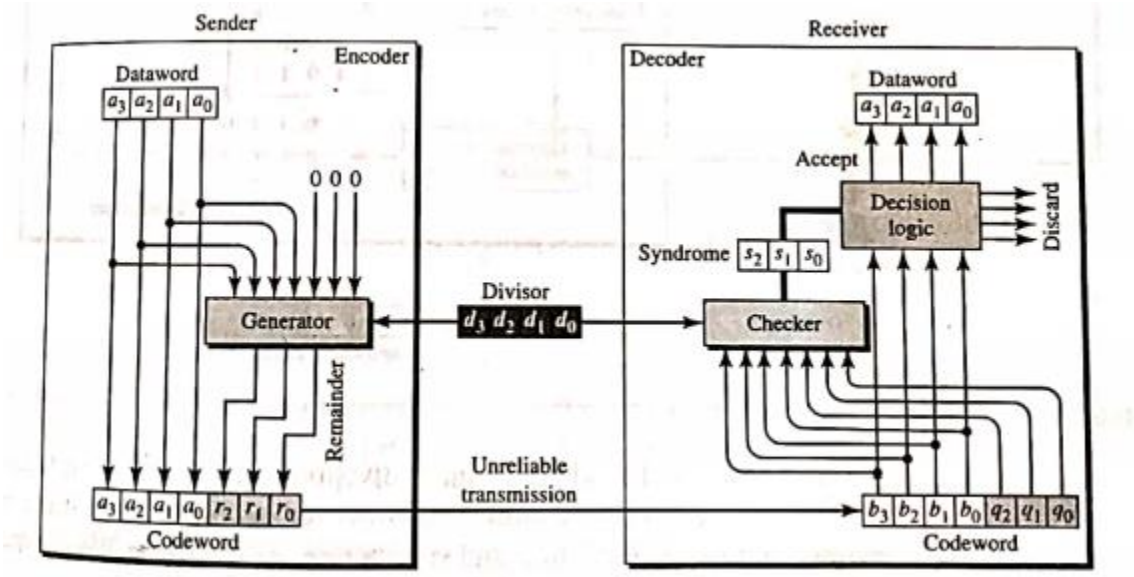
Cyclic codes are special linear block codes with one extra property. If a codeword is cyclically shifted or rotated, the result is another codeword. For example 1011000 is a code word then after a left shift 0110001 is also a valid code word.

→CYCLIC REDUNDANCY CHECK:

- We create cyclic codes to correct errors.
- This cyclic redundancy check's (CRC's) are used in networks such as LAN and WAN.
- The following table represents the cyclic codes table which can represent both linear and cyclic properties of the code.

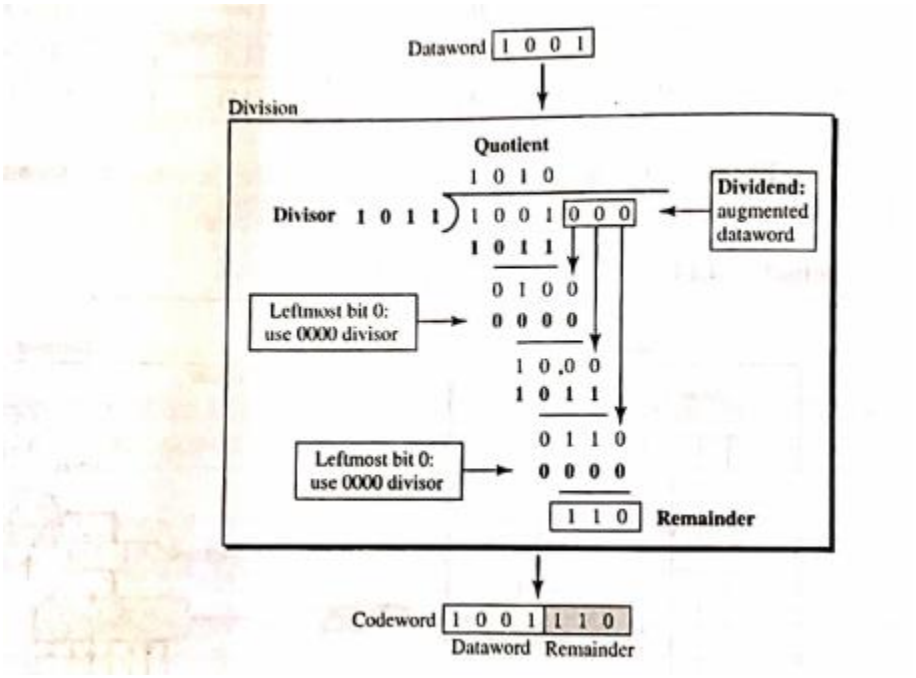
Dataword	Codeword	Dataword	Codeword
0000	0000000	1000	1000101
0001	0001011	1001	1001110
0010	0010110	1010	1010011
0011	0011101	1011	1011000
0100	0100111	1100	1100010
0101	0101100	1101	1101001
0110	0110001	1110	1110100
0111	0111010	1111	1111111

- This figure determines the one possible design for encoder and decoder.

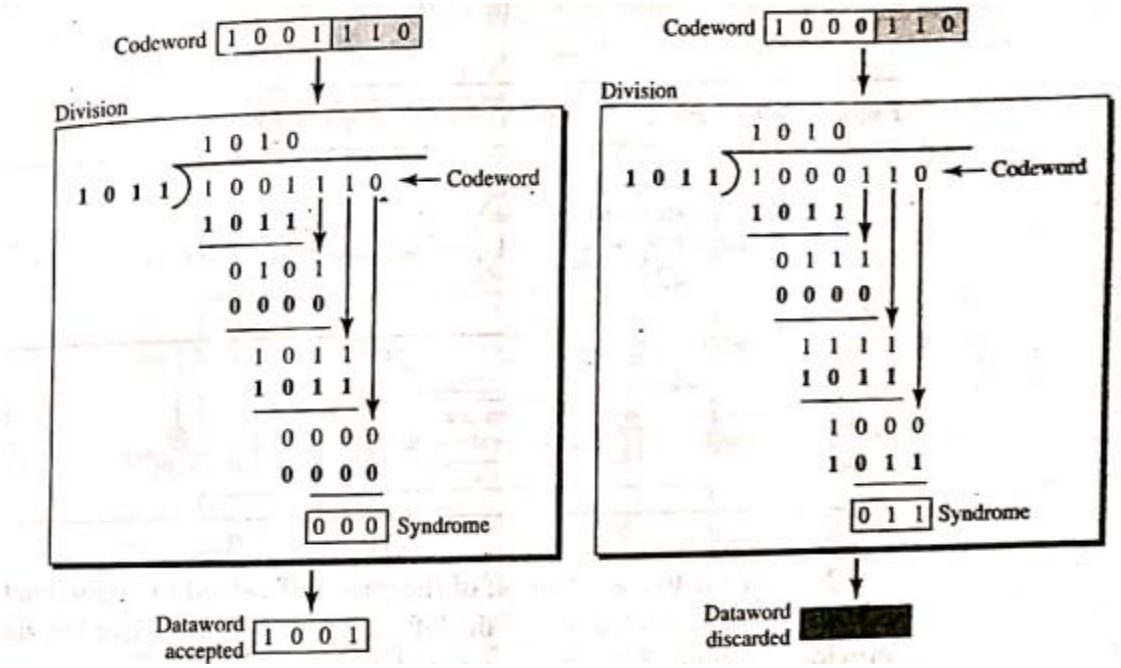


- In the encoder the dataword has ‘k’ bits in length ., the codeword has ‘n’ bits.
- The size of the dataword is augumented by adding n-k 0’s on the rightmost side of dataword.
- The n-bit result is fed into generator.
- The generator uses n-k+1 bits of divisor.The quotient of the division is discarded and the remainder is appended to the dataword to create the codeword.
- The receiver or decoder after receiving a code word sends it to checker a replica of generator to check a valid code word.
- When a code word is received by the checker it performs division operation same like encoder and the remainder is considered to be as syndrome value.
- If the remainder at the decoder is 000 (all 0’s) then there is no error in the received code word , otherwise it is an error codeword.

- The figure shows the division operation of encoder.



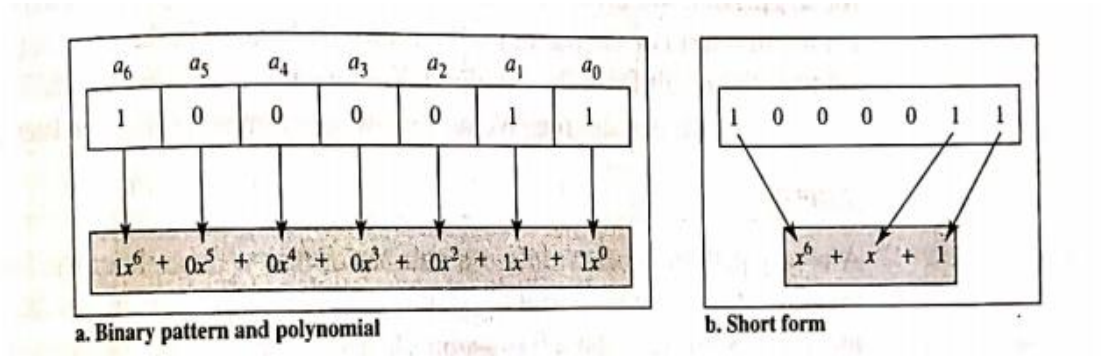
- The process of modulo-2 division is same as division process we use for decimal numbers.
- So same like division process the first 4-bits of dividend is XORed with the divisor and the 3-bit remainder pulls down the extra bits in the dividend.
- If no extra bits are present to pull down from dividend then that 3 bit remainder is used as the rendundant code that is to be added to dataword to make a code word.
- The following figure represents the division in decoder.



- The figure represents 2 cases where there is no error and there is an error.
- In the first case as the codeword is received the checker performs the division operation with the same divisor and checks the remainder
- If the remainder is 0 then that means the syndrome is zero represents there is no error and the dataword is taken
- In the second case as the remainder has the non zero value so that represents an error and the data word is discarded.

POLYNOMIALS:

- A better way to understand cyclic codes is using polynomials.
- A pattern of 0's and 1's can be represented as polynomials with coefficients 0 and 1.
- The power of each term shows the position of the bit; coefficient shows the value.
- The following figure shows a binomial pattern along with its polynomial representation.



- Using this polynomial representation we can reduce the no.of terms that have coefficient value as '0'.

ADDING AND SUBTRACTING POLYNOMIALS:

- In general mathematics the polynomials are added by adding the coefficients of variables with same power or exponent, but here the coefficients are only 1 and 0 hence it is modulo-2 addition.
- It has 2 consequences.
- First, addition and subtraction are same.
- Second, adding or subtracting is done by combining terms and deleting pairs of identical terms.

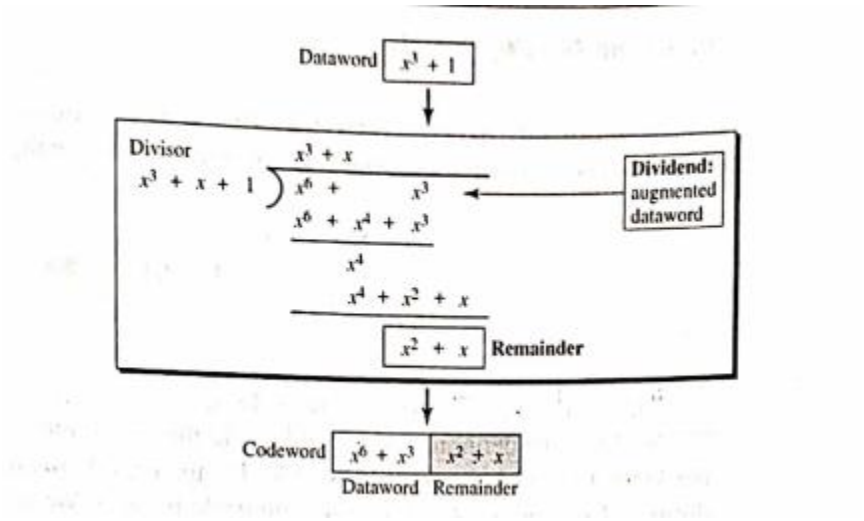
MULTIPLYING OR DIVING TERMS:

- The multiplication in polynomials is quite similar to the multiplication of polynomials in mathematics. Just add the powers of the terms.

- Division of polynomials is quite similar to the division of polynomials i.e.,, divide the first term of dividend with first term of divisor to obtain the first term of quotient.
- We multiply the term in quotient with the divisor and subtract it from the dividend
- We continue this process till we get a term having less exponent than that of divisor.

SHIFTING:

- A binary pattern is often shifted a no.of bits to the right or left**.**
- Shifting to left means adding extra zeros on the right side..So shift left is determined as multiplying each term of polynomial with x^m where m is no.of bits to be shifted.
- The following figure shows the example of a CRC division using polynomials.



→CYCLIC CODE ANALYSIS (Write the procedure for CRC division in theoretical form.)

→ADVANTAGES:

- These cyclic codes can detect single-bit error,double errors, an odd number of errors amd burst errors
- Easily implemented in hardware and software.
- Used in many networks.

→<u>CHECKSUM:</u>

- This is an error detection method.
- It is used in the internet by several protocols although not at the datalink layer.

- Checksum is based on the concept of redundancy.

IDEA:

- The basic idea of check sum is along with the data that is to be sent the sum of those data is also sent.
- Assume if the set of numbers we sent are (7,11,12,0,6) along with this data sum of these values are also sent (7,11,12,0,6,36).
- When the receiver receiver receives data the receiver adds 5 numbers and check the sum with the original sum.
- If both are same then it considers the data to be no error data and accepts the data by neglecting sum.
- Else data is discarded considering the error

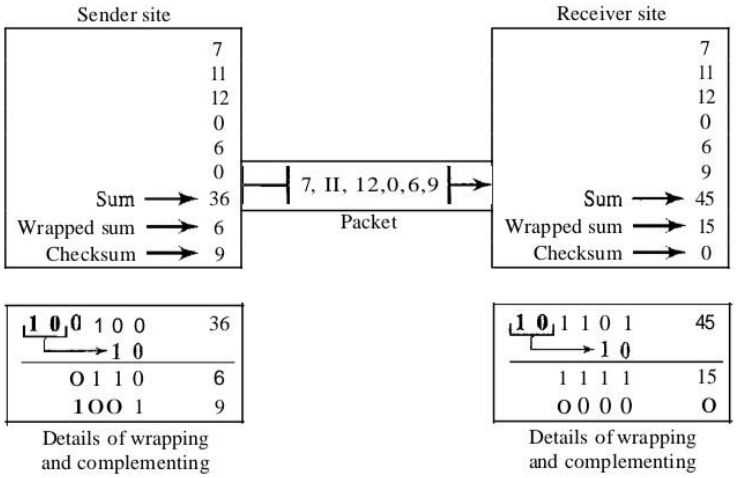
ONE'S COMPLEMENT:

- If all of our data can be written as as 4-bit word except for the checksum one solution is to use one's complement, where we can represent numbers in 0 to 2^n-1 using n bits.
- If the number has more than n bits then the left most bits are added to the rightmost bits of number which is wrapping.
- In one's complement arithmetic negative numbers can be represented by inverting all bits i.e.,, Subtracting the number from 2^n-1.
- An example to explain one's complement arithmetic is as follows:
- How to represent number 21 in one's complement arithmetic using 4 bits?
- The solution is …The binary value of 21 is 10101 so the left most bit is wrapped by adding the bit to rightmost variable. (0101+1) =6.

EXAMPLE:

- The figure shows the process at sender and receiver.
- Assume the data to be (7,11,12,0,6) to be sent
- The sender initialises checksum to 0 and adds all data along with checksum.
- The sum value is 36 which cannot be represented using 4 bits. So the left most 2 bits are wrapped with the rightmost 2 bits and its value becomes 6.
- The wrapped sum is then complemented to obtain the check sum value which is (15-6)=9.
- The receiver follows the same procedure and the sum is 45 (sum of data and checksum) and the wrapped sum becomes 15. And the wrapped sum is complemented to obtain the checksum whose value is 0.
- Which determines there is no error in the data received by the receiver.

- So, the data is accepted and the sum is discarded.



INTERNET CHECKSUM:

- Internet has been using a 16-bit checksum.
- **Sender site:**
  i) The message is divided into 16-bit words.
  ii) The value of checksum word is set to zero.
  iii) All words using checksum are added using one,s complement aritnmetic.
  iv) The sum is complemented and becomes the checksum.
  v) The check sum is sent with data.
- **Receiver site:**
  i) The message including checksum is divided into 16-bit words.
  ii) All words are added using one's complement addition.
  iii) The sum is complemented and becomes the new checksum.
  iv) If the value of checksum is 0, the message is accepted; otherwise rejected.
- The checksum is well-suited for software implementation
- Short programs are written to calculate the checksum at the receiver site or to to check the validity of the message at the receiver site.

→DATALINK LAYER PROTOCOLS:

NOISELESS CHANNELS:

Noiseless channel is a type of ideal channel where there is no chance of frame lost,data corruption or duplication.

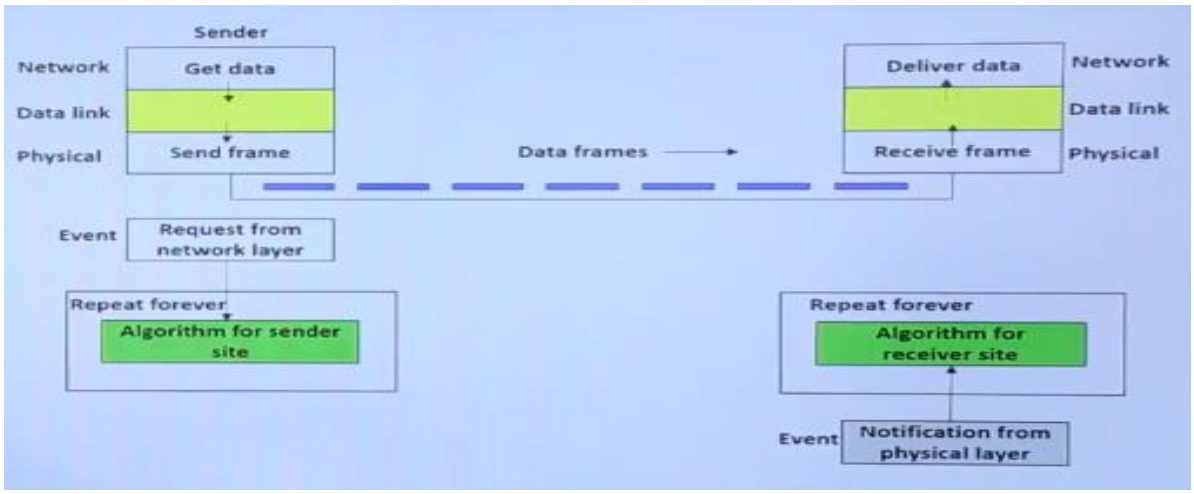
The noiseless channel contains two protocols:

- **Simplest protocol**

- **Stop and Wait ARQ**

Simplest Protocol:

- In this protocol we are not considering about error control because the channel considered is noiseless channel.
- The design of simplest protocol is as follows



→Sender Site:

- On sender site of this protocol the data from network layer is sent to datalink layer made frames and is sent to physical layer and then to receiver.
- So, an event from network layer is generated to ask datalink layer to send data in the form of packets.
- If the event is accepted then the data is made to frames and is sent to receiver.
- This process has an Repeat Forever algorithm to be executed.
- The algorithm is as follows:

```
1  while(true)                    // Repeat forever
2  {
3    WaitForEvent();              // Sleep until an event occurs
4    if(Event(RequestToSend))     //There is a packet to send
5    {
6       GetData();
7       MakeFrame();
8       SendFrame();              //Send the frame
9    }
10 }
```

- The loop is infinite loop because the condition of the loop is always true.
- If the request of the network layer is accepted then the data is taken from network layer and is made to frame and the frame is sent to receiver.

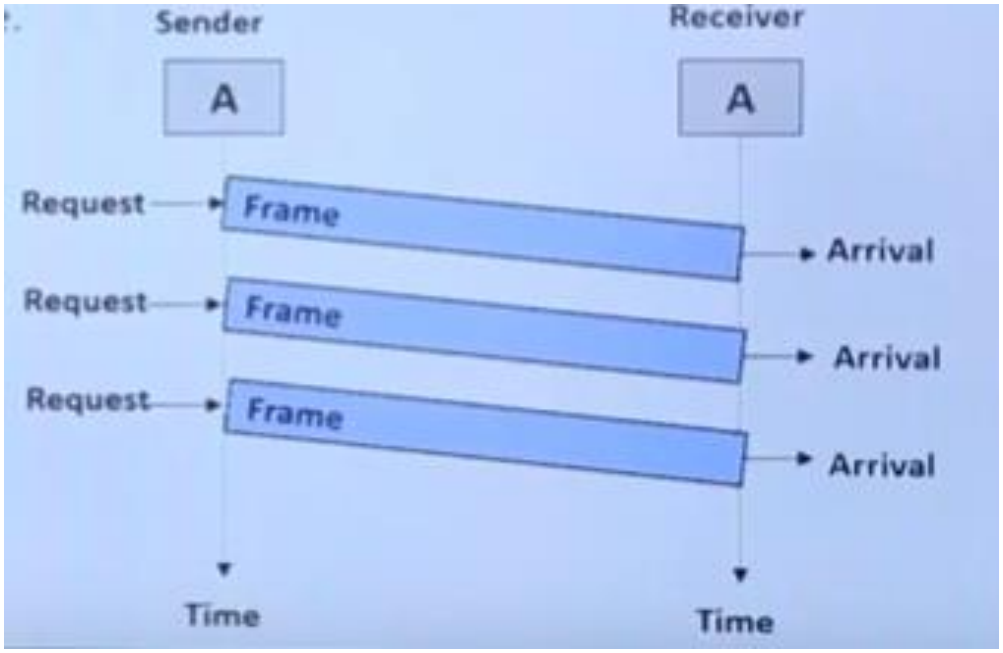
→Receiver Site:

- On receiver site of this protocol the data from physical layer is sent to datalink layer and then datalink layer to network layer.
- An event of arriving of frame is occurred so that the frame from the sender site is accepted by the receiver.
- If the event is accepted then the data is sent to upper layers.
- So, same like sender receiver also has an repeat forever algorithm to be executed.
- The algorithm is as follows:

```
1  while(true)                          // Repeat forever
2  {
3   WaitForEvent();                     // Sleep until an event occurs
4   if(Event(ArrivalNotification))      //Data frame arrived
5   {
6      ReceiveFrame();
7      ExtractData();
8      DeliverData();                   //Deliver data to network layer
9   }
10 }
```

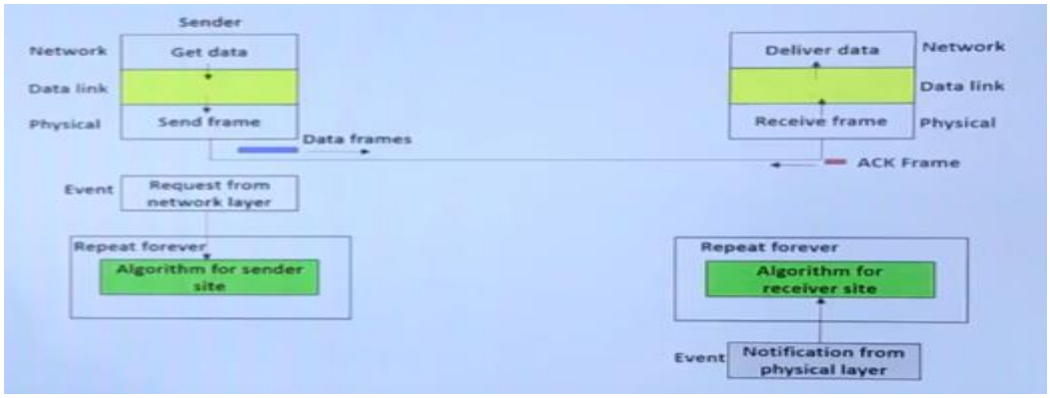
- So when an arrival notification is accepted then the frame is received and then datais extracted from the frame and delivers the data to network layer which is upper layer.

So In this protocol the sender doesn't think about the receiver .to send three frames 3 events occur at sender site and three events at receiver site.



→STOP AND WAIT PROTOCOL:

- In this protocol we are not considering about error control because the channel considered is noiseless channel.
- The stop and wait protocol came to light because in Simplest protocol the data frames are sent irrespective of the receiver's bandwidth.
- So, In this protocol the frames are sent one by one till an acknowledgement of the next frame is received.
- The design of simplest protocol is as follows



→Sender Site:

- On sender site of this protocol the data from network layer is sent to datalink layer made frames and is sent to physical layer and then to receiver.
- So, an event from network layer is generated to ask datalink layer to send data in the form of packets.
- If the event is accepted then the data is made to frames and is sent to receiver.
- After the frame is received by the receiver then a acknowledgement frame is sent by the receiver to sender after the acknowledgement frame is received next frame is sent by the sender.
- This process has an Repeat Forever algorithm to be executed.
- The algorithm is as follows:

```
1  while(true)                          //Repeat forever
2  canSend = true                       //Allow the first frame to go
3  {
4   WaitForEvent();                     // Sleep until an event occurs
5   if(Event(RequestToSend) AND canSend)
6   {
7      GetData();
8      MakeFrame();
9      SendFrame();                     //Send the data frame
10     canSend = false;                 //Cannot send until ACK arrives
11  }
12  WaitForEvent();                     // Sleep until an event occurs
13  if(Event(ArrivalNotification) // An ACK has arrived
14   {
15     ReceiveFrame();                  //Receive the ACK frame
16     canSend = true;
17   }
18 }
```

→Receiver Site:

- On receiver site of this protocol the data from physical layer is sent to datalink layer and then datalink layer to network layer.
- An event of arriving of frame is occurred so that the frame from the sender site is accepted by the receiver.
- After the frame is accepted by the receiver a new frame called acknowledgement frame is created so that the sender knows whether the frame is received by the receiver or not.
- After the acknowledgement is sent the data is extracted from frame and sent to upper layers.
- So, same like sender receiver also has an repeat forever algorithm to be executed.
- The algorithm is as follows:

```
1  while(true)                          //Repeat forever
2  {
3   WaitForEvent();                     // Sleep until an event occurs
4   if(Event(ArrivalNotification))      //Data frame arrives
5   {
6      ReceiveFrame();
7      ExtractData();
8      Deliver(data);                   //Deliver data to network layer
9      SendFrame();                     //Send an ACK frame
10  }
11 }
```