Data Link Layer Design Issues

Dr. Sunil Kumar Singh
Assistant Professor
School of Computer Science & Engineering
VIT-AP University
Amaravati

3 septembre 2020

Outline

1. Introduction
2. Data link layer design issues
   - Services provided to the network layer
   - Framming
   - Error Control
   - HDLC and PPP
   - Flow Control
3. Summary

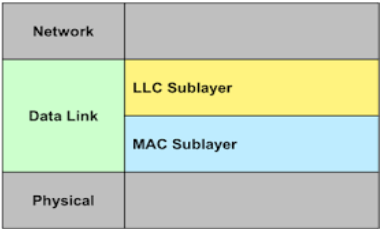
Introduction

Data Link Layer

- Data Link Layer is second layer of OSI Layered Model.
- This layer is one of the most complicated layers and has complex functionalities and liabilities.
- Data link layer works between two hosts which are directly connected.

Data link layer has two sub-layers :

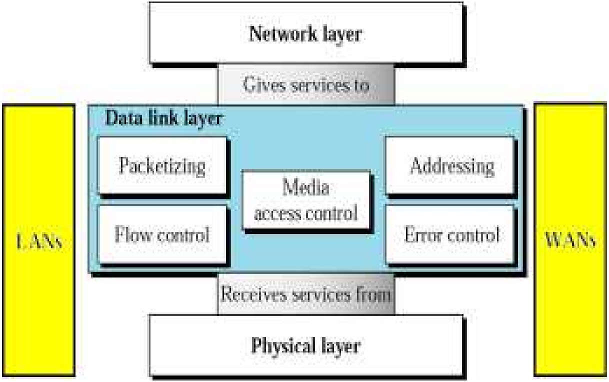
- **Logical Link Control** : It deals with protocols, flow-control, and error control
- **Media Access Control** : It deals with actual control of media



Data link layer design issues

The main functions and the design issues of this layer are

- Services provided to the network layer
- Framing
- Error Control
  - Error detection
  - Error correction
- Flow Control



Data link layer design issues (cont..)

Services provided to the network layer

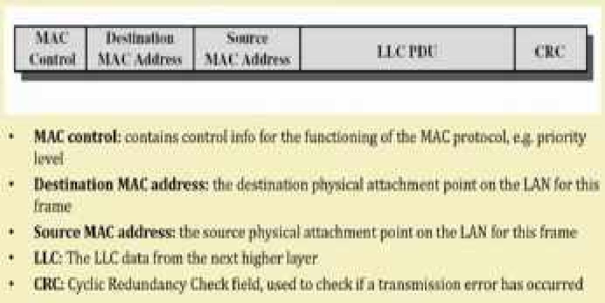
The types of services provided can be of three types :

- Unacknowledged connectionless service.
  - no recovering of lost and corrupted frame.
  - when the error rate is very low.
  - real-time traffic, like speech or video.
- Acknowledged connectionless service.
  - return information a frame has safely arrived.
  - when the error rate is very low.
  - unreliable channels, such as wireless networks.
- Acknowledged connection-oriented service.
  - established connection before any data is sent.
  - provides the reliable bit stream to network layer.
  - Satellite communication channel and long distance telephone communication is the best examples.

Data link layer design issues (cont..)

Framming

- Framing = How to break a bit-stream into frames
- Need for framing : Error Detection/Control work on chunks and not on bit streams of data



MAC Control	Destination MAC Address	Source MAC Address	LLC PDU	CRC

- **MAC control:** contains control info for the functioning of the MAC protocol, e.g. priority level
- **Destination MAC address:** the destination physical attachment point on the LAN for this frame
- **Source MAC address:** the source physical attachment point on the LAN for this frame
- **LLC:** The LLC data from the next higher layer
- **CRC:** Cyclic Redundancy Check field, used to check if a transmission error has occurred

Data link layer design issues (cont..)

Framing methods :

How can frame be transmitted so that a receiver can detect frame bounderies ? That is, how a receiver recognize the start and end of a frame ?

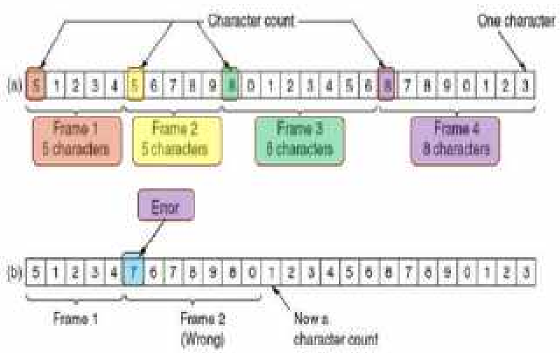
- Character count
- Byte stuffing
- Bit stuffing

Data link layer design issues (cont..)

Character count

- First field in the frame's header = the length of the frame.
- A transmission error can cause an incorrect count causing the source and destination to get out of synchronization.
- Rarely used in actual data link protocols.

A character stream. (a) Without errors (b) With one error



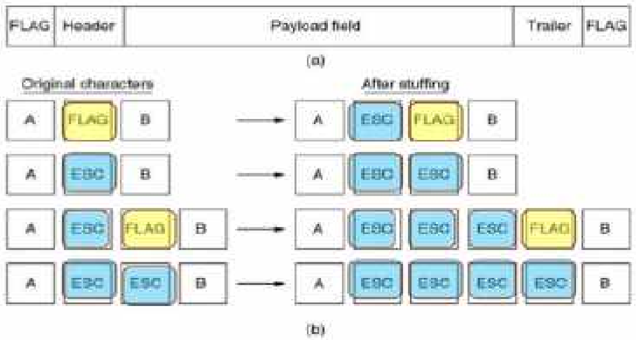
(a) 5 1 2 3 4 5 6 7 8 9 8 0 1 2 3 4 5 6 8 7 8 9 0 1 2 3

(b) 5 1 2 3 4 7 6 7 8 9 8 0 1 2 3 4 5 6 8 7 8 9 0 1 2 3

Data link layer design issues (cont..)

Byte stuffing

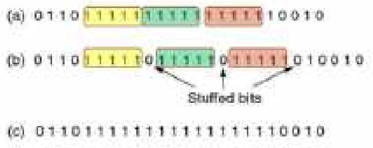
- Byte stuffing is the process of adding one extra byte whenever there is a flag or escape character in the payload.
- it fix the problems of character count framming but it has also some issues.
  - fixed character size of 8 bits
  - can't handle heterogeneous(different system architecture) environment.
- Rarely used in actual data link protocols.



Data link layer design issues (cont..)

Bits stuffing

- Delimit with special bit pattern (bit flags).
- Stuff bits if pattern appears in data
- Remove stuffed bits at destination.


(a) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0
(b) 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 0 0 1 0
Stuffed bits
(c) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

Bit stuffing

(a) The original data.

(b) The data as they appear on the line.

(c) The data as they are stored in receiver's memory after destuffing.

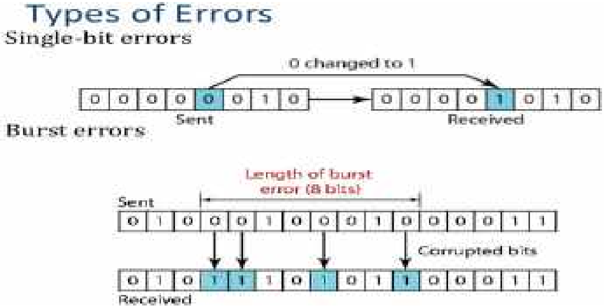
Error Control

Error Control = Deliver frames without error, in the proper order to network layer.

Error control in the data link layer is based on ARQ (Autometic Repeat Request), which is the retransmission of data.

- Error control Mechanisms :
    - Ack/Nak : Provide sender some feedback about other end.
    - Time-out : for the case when entire packet or ack is lost.
    - Sequence numbers : to distinguish retransmissions from originals.

Error Control (Cont..)



Error Detection and Correction

Data can be corrupted during transmission. For reliable communication, errors must be detected and corrected.

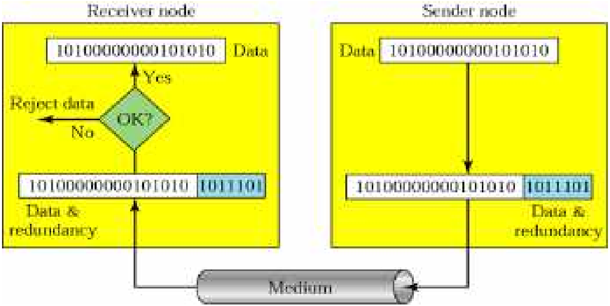
- Error detection Methods
  - Parity Check
  - Cyclic Redundancy Check (CRC)
  - Checksum

Error Detection

Redundancy

Error detection uses the concept of redundancy, which means adding extra bits for detecting errors at the destination.

n-bit codeword = m message bits + r redundancy or check bits

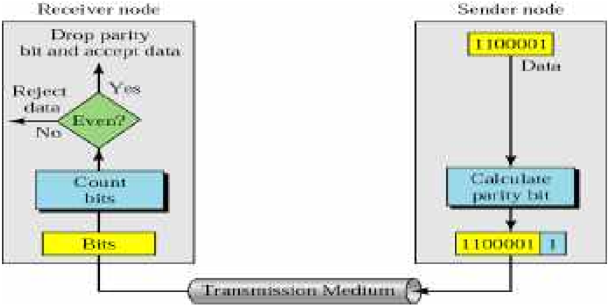


Error Detection (Cont..)

Parity check

In parity check, a parity bit is added to every data unit so that the total number of 1's is even (or odd for odd-parity).

- Simple parity : for single bit errors
- Two dimentional : for burst errors



Error Detection (Cont..)

Simple parity check Example

Suppose the sender wants to send the word world. In ASCII the five characters are coded as

1110111 1101111 1110010 1101100 1100100
The following shows the actual bits sent
11101110 11011110 11100100 11011000 11001001

Now suppose the word world in Example 1 is received by the receiver without being corrupted in transmission.

11101110 11011110 11100100 11011000 11001001
The receiver counts the 1s in each character and comes up with even numbers (6, 6, 4, 4, 4). The data are accepted.

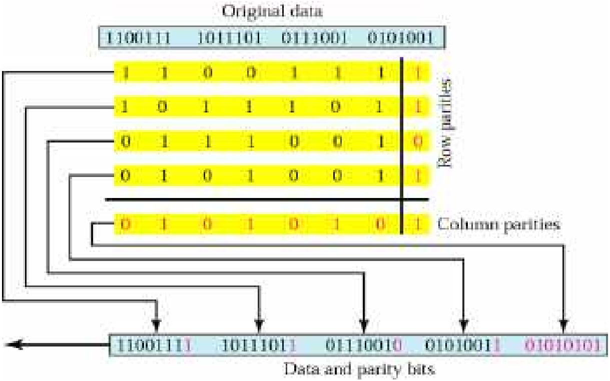
Now suppose the word world in Example 1 is corrupted during transmission.

11111110 11011110 11101100 11011000 11001001
The receiver counts the 1s in each character and comes up with even and odd numbers (7, 6, 5, 4, 4). The receiver knows that the data are corrupted, discards them, and asks for retransmission.

Error Detection (Cont..)

Two dimentional parity check

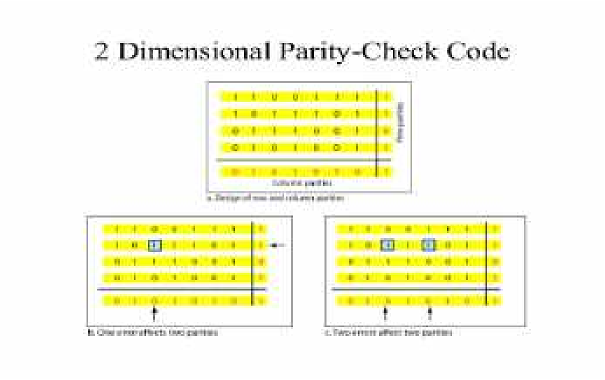
In two-dimensional parity check, a block of bits is divided into rows and a redundant row of bits is added to the whole block.



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

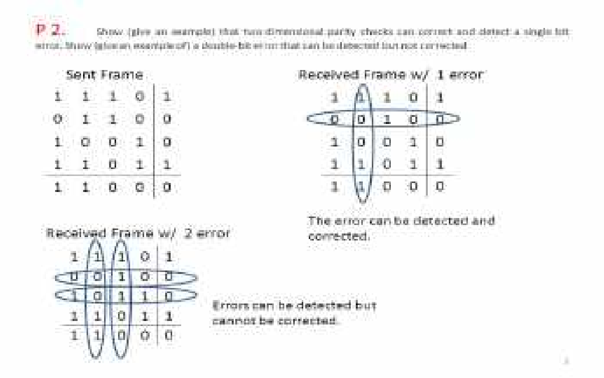
Error Detection (Cont..)

Two dimentional parity check


2 Dimensional Parity-Check Code
Row parities
Column parities
a. Design of row and column parities
b. One error affects two parities
c. Two errors affect two parities

Error Detection (Cont..)

Two dimentional parity check



P 2. Show (give an example) that two-dimensional parity checks can correct and detect a single bit error. Show (give an example of) a double-bit error that can be detected but not corrected.

Sent Frame

1	1	1	0	1
0	1	1	0	0
1	0	0	1	0
1	1	0	1	1
1	1	0	0	0

Received Frame w/ 1 error

1	1	1	0	1
0	0	1	0	0
1	0	0	1	0
1	1	0	1	1
1	1	0	0	0

The error can be detected and corrected.

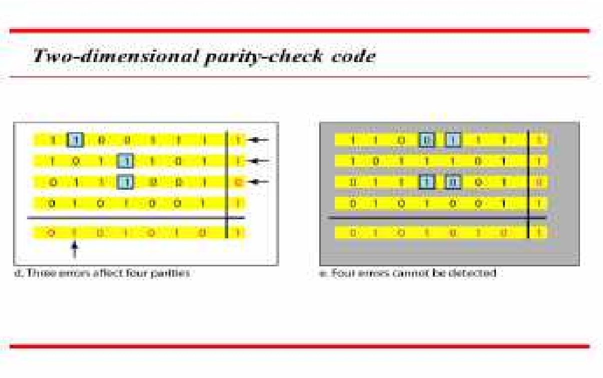
Received Frame w/ 2 error

1	1	1	0	1
0	0	1	0	0
1	0	1	1	0
1	1	0	1	1
1	1	0	0	0

Errors can be detected but cannot be corrected.

Error Detection (Cont..)

Two dimentional parity check



Two-dimensional parity-check code

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

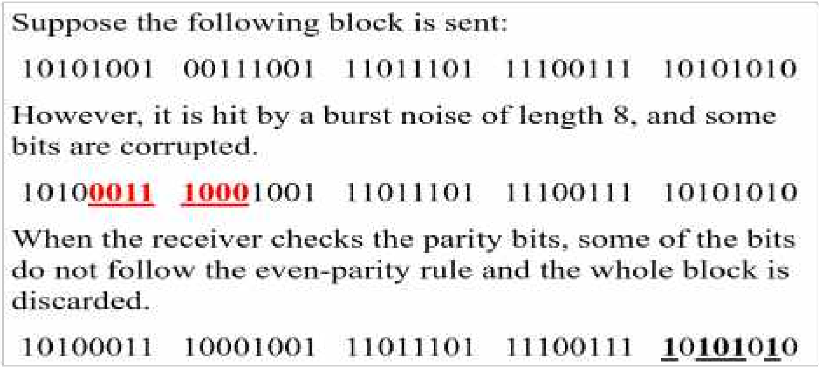
d. Three errors affect four parities

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

e. Four errors cannot be detected

Error Detection (Cont..)

Two dimentional parity check Example



Suppose the following block is sent:

10101001 00111001 11011101 11100111 10101010

However, it is hit by a burst noise of length 8, and some bits are corrupted.

10100011 10001001 11011101 11100111 10101010

When the receiver checks the parity bits, some of the bits do not follow the even-parity rule and the whole block is discarded.

10100011 10001001 11011101 11100111 10101010

Error Detection (Cont..)

Checksum

The sender follows these steps :

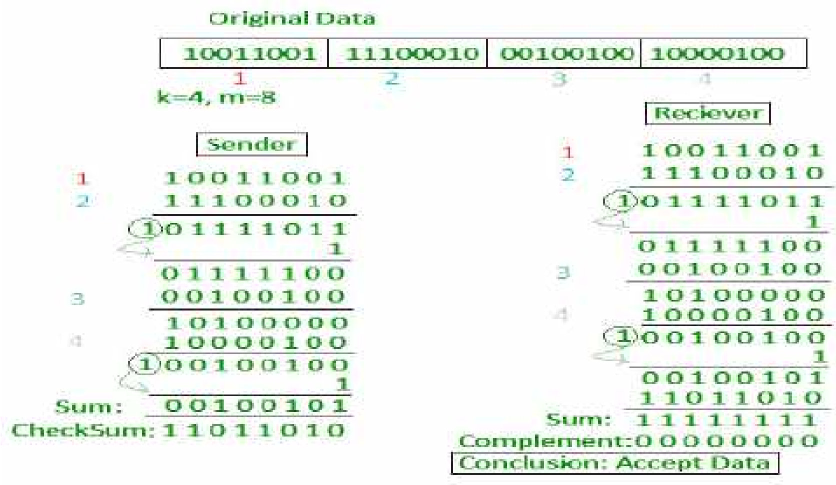
- The unit is divided into k sections, each of n bits.
- All sections are added using one's complement to get the sum.
- The sum is complemented and becomes the checksum.
- The checksum is sent with the data.

The receiver follows these steps :

- The unit is divided into k sections, each of n bits.
- All sections are added using one's complement to get the sum.
- The sum is complemented.
- If the result is zero, the data are accepted : otherwise, rejected.

Error Detection (Cont..)

Checksum Example



Original Data

10011001	11100010	00100100	10000100
1	2	3	4

k=4, m=8

Sender

```
1          1 0 0 1 1 0 0 1
2          1 1 1 0 0 0 1 0
          (1)0 1 1 1 1 0 1 1
                         1
           0 1 1 1 1 1 0 0
3          0 0 1 0 0 1 0 0
           1 0 1 0 0 0 0 0
4          1 0 0 0 0 1 0 0
          (1)0 0 1 0 0 1 0 0
                         1
Sum:       0 0 1 0 0 1 0 1
CheckSum:  1 1 0 1 1 0 1 0
```

Reciever

```
1          1 0 0 1 1 0 0 1
2          1 1 1 0 0 0 1 0
          (1)0 1 1 1 1 0 1 1
                         1
           0 1 1 1 1 1 0 0
3          0 0 1 0 0 1 0 0
           1 0 1 0 0 0 0 0
4          1 0 0 0 0 1 0 0
          (1)0 0 1 0 0 1 0 0
                         1
           0 0 1 0 0 1 0 1
           1 1 0 1 1 0 1 0
Sum:       1 1 1 1 1 1 1 1
Complement:0 0 0 0 0 0 0 0
Conclusion: Accept Data
```

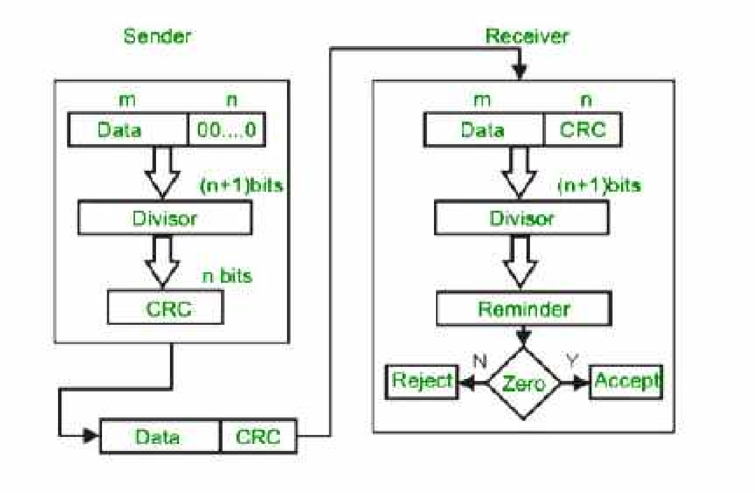
Error Detection (Cont..)

Cyclic Redundancy Check (CRC)

- Unlike checksum scheme, which is based on addition, CRC is based on binary division.
- In CRC, a sequence of redundant bits, called cyclic redundancy check bits, are appended to the end of data unit so that the resulting data unit becomes exactly divisible by a second, predetermined binary number.
- At the destination, the incoming data unit is divided by the same number. If at this step there is no remainder, the data unit is assumed to be correct and is therefore accepted.
- A remainder indicates that the data unit has been damaged in transit and therefore must be rejected.

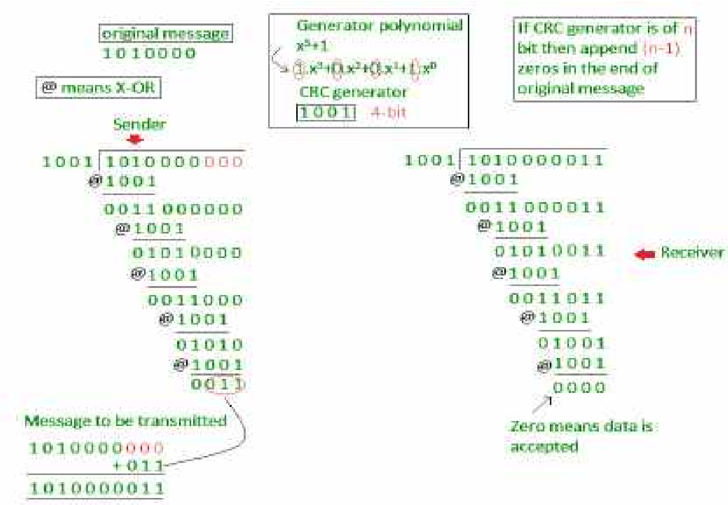
Error Detection (Cont..)

Cyclic Redundancy Check (CRC)



Error Detection (Cont..)

Cyclic Redundancy Check (CRC) Example



original message
1010000

@ means X-OR

Generator polynomial
x^3+1
$1.x^3+0.x^2+0.x^1+1.x^0$
CRC generator
1001 4-bit

If CRC generator is of n bit then append (n-1) zeros in the end of original message

Sender

```
1001 | 1010000000
      @1001
       0011000000
        @1001
         0101000
        @1001
           0011000
          @1001
             01010
            @1001
              0011
```

Message to be transmitted
1010000000
+011
1010000011

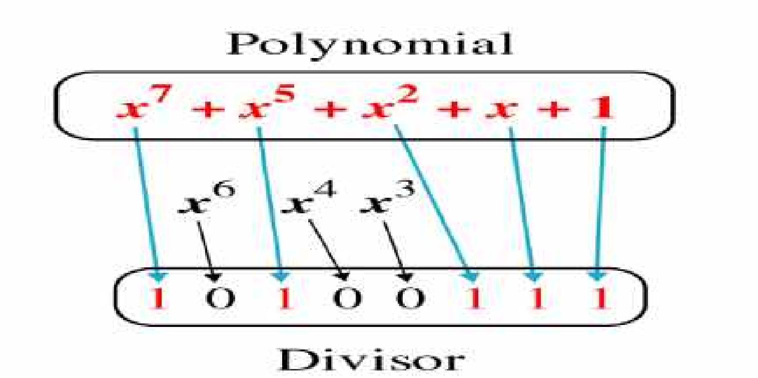
```
1001 | 1010000011
      @1001
       0011000011
        @1001
         01010011
        @1001
          0011011
         @1001
            01001
           @1001
             0000
```

Receiver

Zero means data is accepted

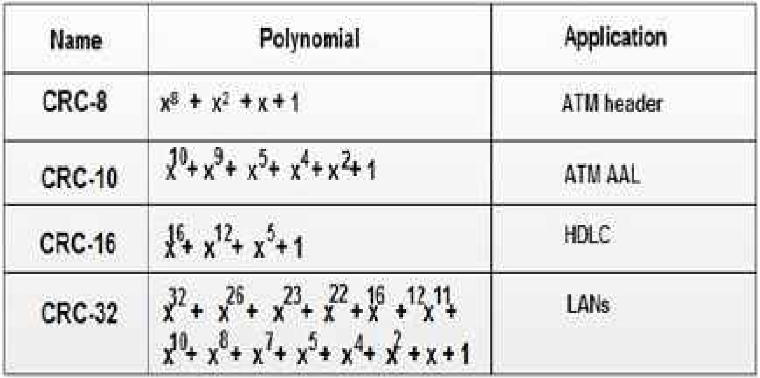
Error Detection (Cont..)

Cyclic Redundancy Check (CRC)



Error Detection (Cont..)

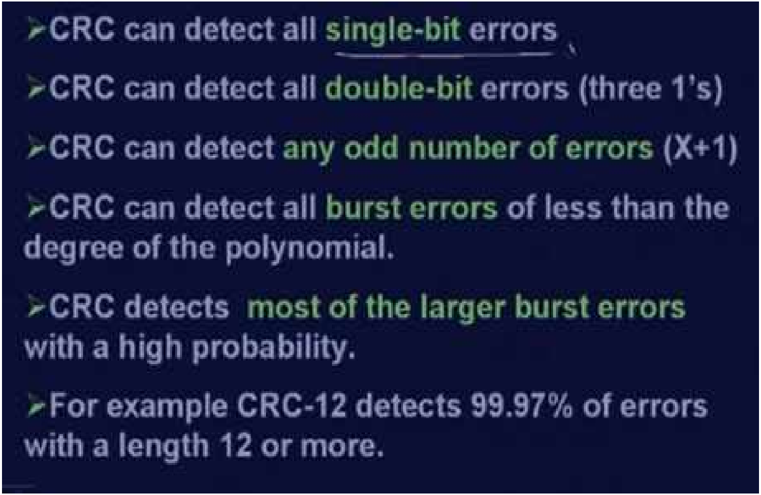
Some standard polynomials for CRC



Name	Polynomial	Application
CRC-8	$x^8 + x^2 + x + 1$	ATM header
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^2 + 1$	ATM AAL
CRC-16	$x^{16} + x^{12} + x^5 + 1$	HDLC
CRC-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$	LANs

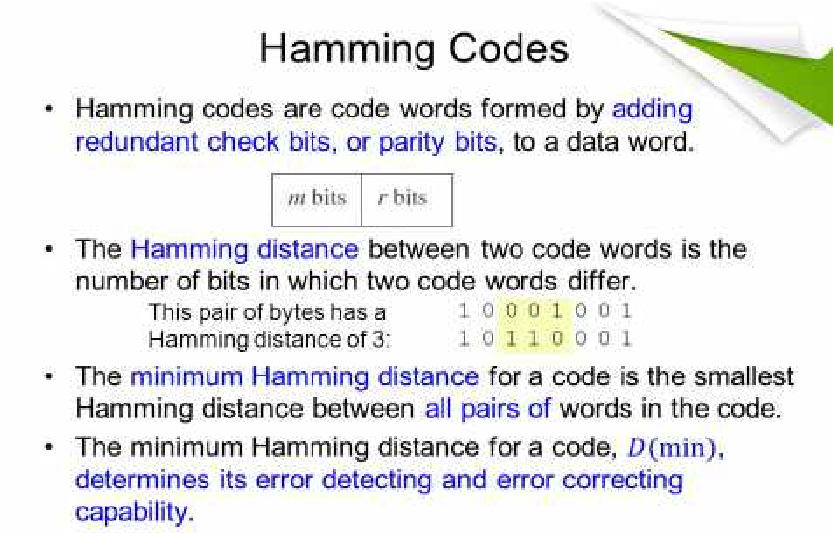
Error Detection (Cont..)

Performance of CRC



- CRC can detect all single-bit errors
- CRC can detect all double-bit errors (three 1's)
- CRC can detect any odd number of errors (X+1)
- CRC can detect all burst errors of less than the degree of the polynomial.
- CRC detects most of the larger burst errors with a high probability.
- For example CRC-12 detects 99.97% of errors with a length 12 or more.

Error Correction



Hamming Codes

- Hamming codes are code words formed by adding redundant check bits, or parity bits, to a data word.

m bits	r bits

- The Hamming distance between two code words is the number of bits in which two code words differ.

  This pair of bytes has a Hamming distance of 3:

  1 0 0 0 1 0 0 1
  1 0 1 1 0 0 0 1

- The minimum Hamming distance for a code is the smallest Hamming distance between all pairs of words in the code.
- The minimum Hamming distance for a code, $D(\min)$, determines its error detecting and error correcting capability.

Error Correction

Hamming Distance Code

Hamming code is a set of error-correction codes that can be used to detect and correct the errors that can occur when the data is moved or stored from the sender to the receiver. It is technique developed by R.W. Hamming for error correction.

The number of redundant bits can be calculated using the following formula :

$2^r >= m + r + 1$ where, r = redundant bit, m = data bit

For Example :
Suppose the number of data bits is 7, then the number of redundant bits can be

calculated using := $2^4 >= 7 + 4 + 1$

Thus, the number of redundant bits= 4

Parity bits or Redundant bits

A parity bit is a bit appended to a data of binary bits to ensure that the total number of 1's in the data is even or odd. Parity bits are used for error detection. There are two types of parity bits :

Error Correction

General Algorithm of Hamming code

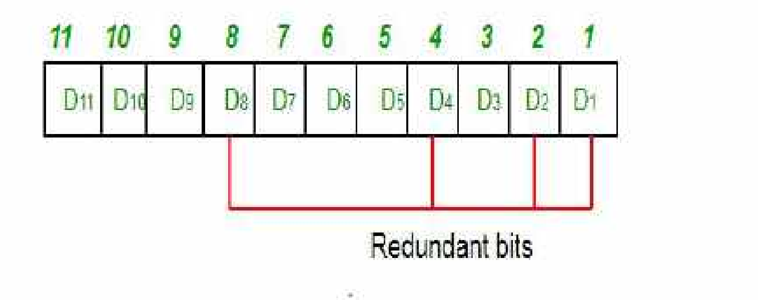
- Write the bit positions starting from 1 form LSB or MSB in given binary sequence.
- All the bit positions that are a power of 2 are marked as parity bits (1, 2, 4, 8, etc).
- All the other bit positions are marked as data bits.
- Each data bit is included in a unique set of parity bits, as determined its bit position in binary form.
  - Parity bit 1 covers all the bits positions whose binary representation includes a 1 in the least significant position (1, 3, 5, 7, 9, 11, etc).
  - Parity bit 2 covers all the bits positions whose binary representation includes a 1 in the second position from the least significant bit (2, 3, 6, 7, 10, 11, etc).
  - Parity bit 4 covers all the bits positions whose binary representation includes a 1 in the third position from the least significant bit (4–7, 12–15, 20–23, etc).
  - Parity bit 8 covers all the bits positions whose binary representation includes a 1 in the fourth position from the least significant bit bits (8–15, 24–31, 40–47, etc).
  - In general, each parity bit covers all bits where the bitwise AND of the parity position and the bit position is non-zero.
- Since we check for even parity set a parity bit to 1 if the total number of ones in the positions it checks is odd.
- Set a parity bit to 0 if the total number of ones in the positions it checks is even.

Error Correction

Determining the position of Parity bits

These redundant bits are placed at the positions which correspond to the power of 2. As in the above example :

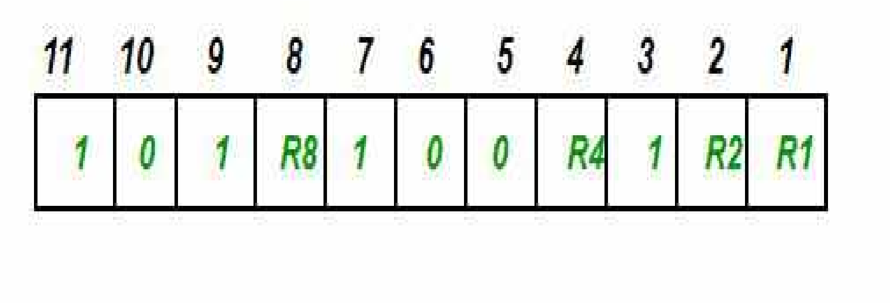
- The number of data bits = 7
- The number of redundant bits = 4
- The total number of bits = 11
- The redundant bits are placed at positions corresponding to power of 2- 1, 2, 4, and 8



Error Correction

Determining the position of Parity bits

Suppose the data to be transmitted is 1011001, the bits will be placed as follows :

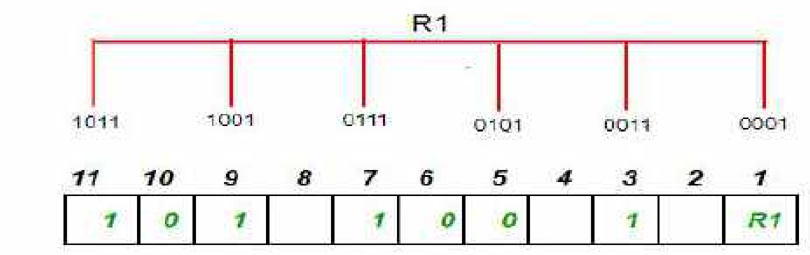


11	10	9	8	7	6	5	4	3	2	1
1	0	1	R8	1	0	0	R4	1	R2	R1

Error Correction

Determining the Parity bits

R1 bit is calculated using parity check at all the bits positions whose binary representation includes a 1 in the least significant position.
R1 : bits 1, 3, 5, 7, 9, 11



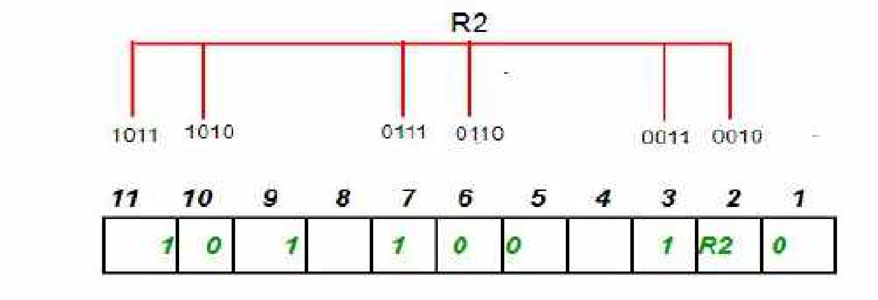
11	10	9	8	7	6	5	4	3	2	1
1	0	1		1	0	0		1		R1

To find the redundant bit R1, we check for even parity. Since the total number of 1's in all the bit positions corresponding to R1 is an even number the value of R1 (parity bit's value) = 0

Error Correction

Determining the Parity bits

R2 bit is calculated using parity check at all the bits positions whose binary representation includes a 1 in the second position from the least significant bit. R2 : bits 2,3,6,7,10,11



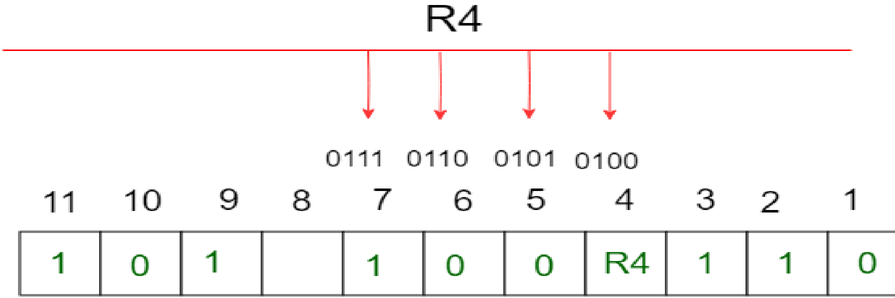
11	10	9	8	7	6	5	4	3	2	1
1	0	1		1	0	0		1	R2	0

To find the redundant bit R2, we check for even parity. Since the total number of 1's in all the bit positions corresponding to R2 is odd the value of R2(parity bit's value)=1

Error Correction

Determining the Parity bits

R4 bit is calculated using parity check at all the bits positions whose binary representation includes a 1 in the third position from the least significant bit. R4 : bits 4, 5, 6, 7



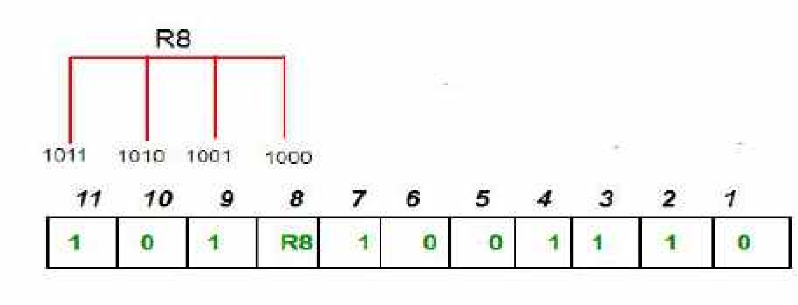
				0111	0110	0101	0100			
11	10	9	8	7	6	5	4	3	2	1
1	0	1		1	0	0	R4	1	1	0

To find the redundant bit R4, we check for even parity. Since the total number of 1's in all the bit positions corresponding to R4 is odd the value of R4(parity bit's value) = 1

Error Correction

Determining the Parity bits

R8 bit is calculated using parity check at all the bits positions whose binary representation includes a 1 in the fourth position from the least significant bit. R8 : bit 8,9,10,11



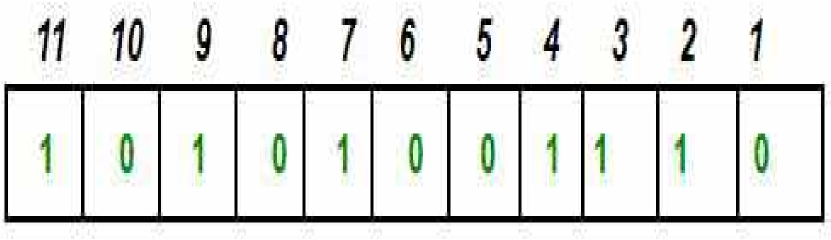
11	10	9	8	7	6	5	4	3	2	1
1	0	1	R8	1	0	0	1	1	1	0

To find the redundant bit R8, we check for even parity. Since the total number of 1's in all the bit positions corresponding to R8 is an even number the value of R8(parity bit's value)=0.

Error Correction

Transmitted data

The sender transmitted data is as follows :

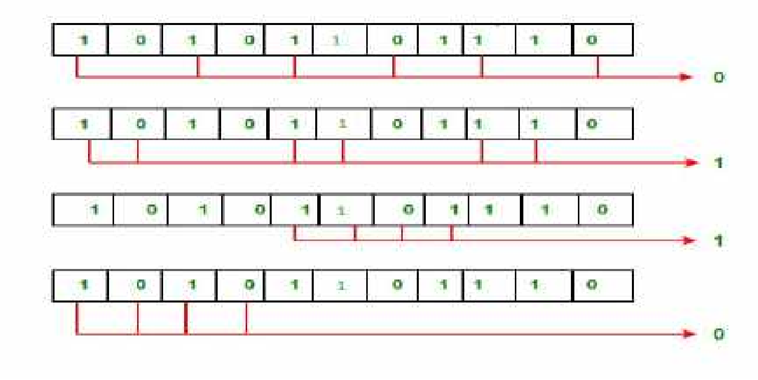


11	10	9	8	7	6	5	4	3	2	1
1	0	1	0	1	0	0	1	1	1	0

Error Correction

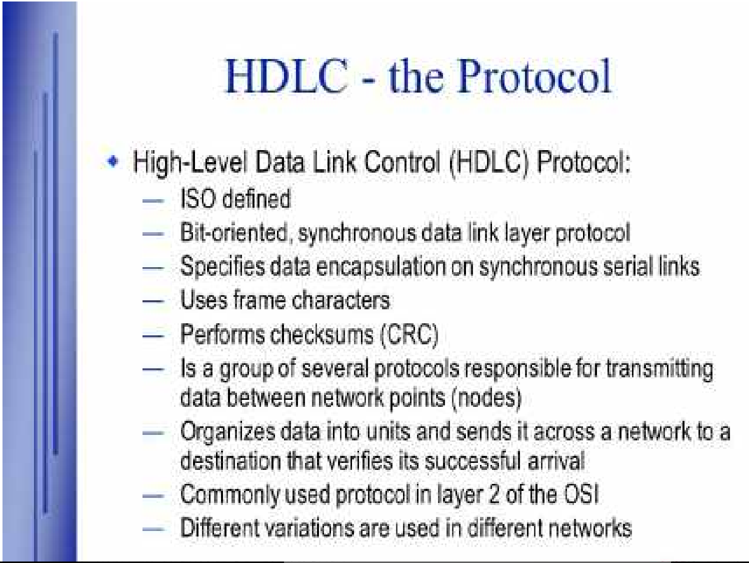
Error detection and correction

Suppose in the above example the 6th bit is changed from 0 to 1 during data transmission, then it gives new parity values in the binary number :



The bits give the binary number as 0110 whose decimal representation is 6. Thus, the bit 6 contains an error. To correct the error the 6th bit is changed from 1 to 0.

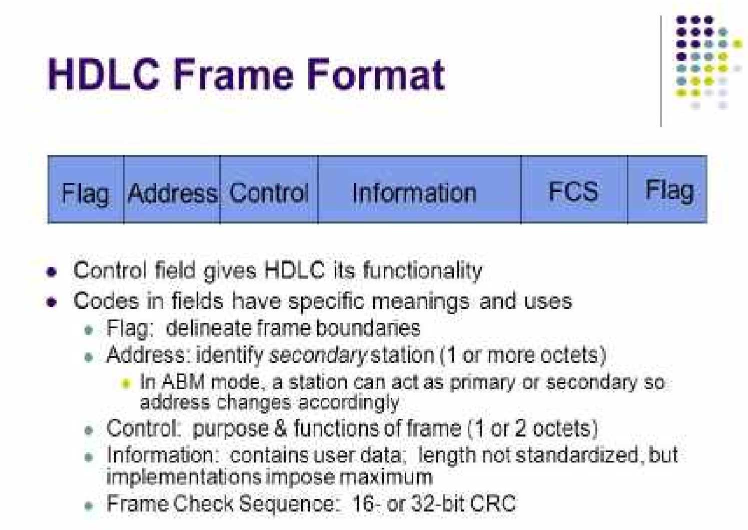
HDLC



HDLC - the Protocol

- High-Level Data Link Control (HDLC) Protocol:
  - ISO defined
  - Bit-oriented, synchronous data link layer protocol
  - Specifies data encapsulation on synchronous serial links
  - Uses frame characters
  - Performs checksums (CRC)
  - Is a group of several protocols responsible for transmitting data between network points (nodes)
  - Organizes data into units and sends it across a network to a destination that verifies its successful arrival
  - Commonly used protocol in layer 2 of the OSI
  - Different variations are used in different networks

HDLC Frame format

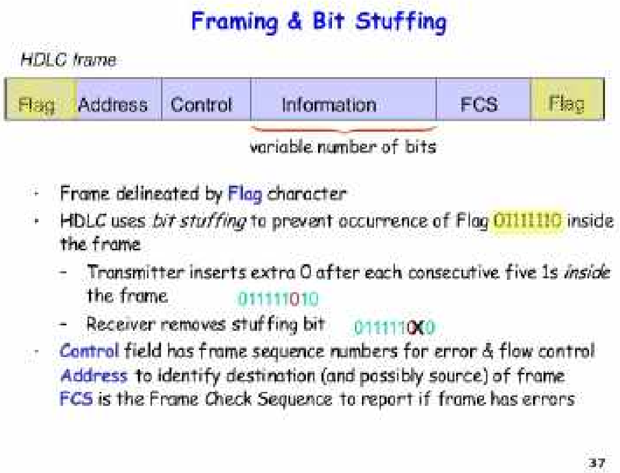


HDLC Frame Format

Flag	Address	Control	Information	FCS	Flag

- Control field gives HDLC its functionality
- Codes in fields have specific meanings and uses
  - Flag: delineate frame boundaries
  - Address: identify *secondary* station (1 or more octets)
    - In ABM mode, a station can act as primary or secondary so address changes accordingly
  - Control: purpose & functions of frame (1 or 2 octets)
  - Information: contains user data; length not standardized, but implementations impose maximum
  - Frame Check Sequence: 16- or 32-bit CRC

HDLC Frame format



Framing & Bit Stuffing

HDLC frame

Flag	Address	Control	Information	FCS	Flag

variable number of bits

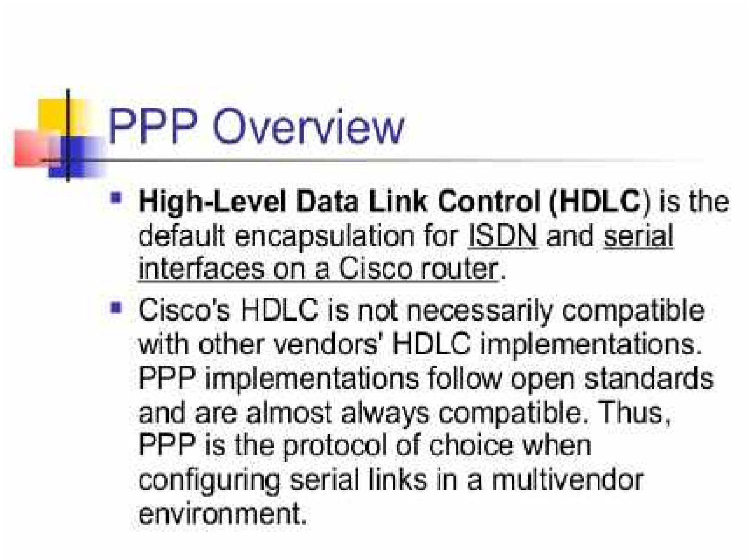
- Frame delineated by Flag character
- HDLC uses *bit stuffing* to prevent occurrence of Flag 01111110 inside the frame
  - Transmitter inserts extra 0 after each consecutive five 1s *inside* the frame 011111010
  - Receiver removes stuffing bit 0111110X0
- Control field has frame sequence numbers for error & flow control

Address to identify destination (and possibly source) of frame

FCS is the Frame Check Sequence to report if frame has errors

37

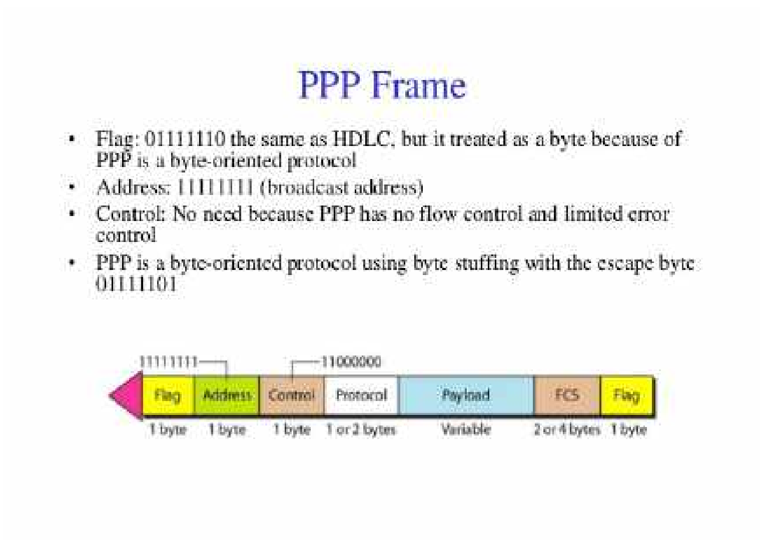
Point-to-point (PPP)



PPP Overview

- **High-Level Data Link Control (HDLC)** is the default encapsulation for ISDN and serial interfaces on a Cisco router.
- Cisco's HDLC is not necessarily compatible with other vendors' HDLC implementations. PPP implementations follow open standards and are almost always compatible. Thus, PPP is the protocol of choice when configuring serial links in a multivendor environment.

PPP Frame format

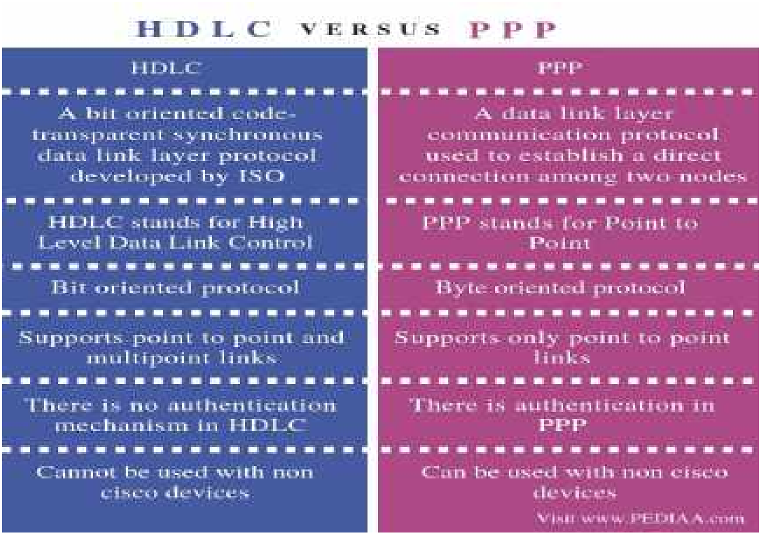


PPP Frame

- Flag: 01111110 the same as HDLC, but it treated as a byte because of PPP is a byte-oriented protocol
- Address: 11111111 (broadcast address)
- Control: No need because PPP has no flow control and limited error control
- PPP is a byte-oriented protocol using byte stuffing with the escape byte 01111101

	11111111	11000000				
Flag	Address	Control	Protocol	Payload	FCS	Flag
1 byte	1 byte	1 byte	1 or 2 bytes	Variable	2 or 4 bytes	1 byte

HDLC V/C PPP



HDLC VERSUS PPP

HDLC	PPP
A bit oriented code-transparent synchronous data link layer protocol developed by ISO	A data link layer communication protocol used to establish a direct connection among two nodes
HDLC stands for High Level Data Link Control	PPP stands for Point to Point
Bit oriented protocol	Byte oriented protocol
Supports point to point and multipoint links	Supports only point to point links
There is no authentication mechanism in HDLC	There is authentication in PPP
Cannot be used with non cisco devices	Can be used with non cisco devices

Visit www.PEDIAA.com

Data link layer design issues (cont..)

Flow Control

Flow control is a technique that allows two stations working at different speeds to communicate with each other. It is a set of measures taken to regulate the amount of data that a sender sends so that a fast sender does not overwhelm a slow receiver.

Two types of mechanisms can be deployed to control the flow :

- Stop and Wait

This flow control mechanism forces the sender after transmitting a data frame to stop and wait until the acknowledgement of the data-frame sent is received.

- Stop and Wait

In this flow control mechanism, both sender and receiver agree on the number of data-frames after which the acknowledgement should be sent.

Summary

- Data link layer deals with the design and procedures for node-to-node communication.
- Framming in this layer separates one packet from another.
- Data link layer handles physical address or MAC address to locate the host in a LAN.
- Error handling and flow control are the two main tasks of this layer.

Thanks

sksingh.cse@gmail.com