Fundamentals of Data Link Control Protocols

DLC Basic Functions

DLC Flow Control

DLC Error Control

DLC Protocols Performance Analysis

DLC Protocol Classes

Data Link Control Protocol Service

- The data link layer provides reliable data transmission service over a potentially unreliable transmission link
- The basic unit of communication at the data link layer is referred as frame
  - A frame is defined as a serial block of bits containing data and control information.

Data Link Control Characteristics

- Two aspects characterize DLC protocols:
  - Correctness
    - Ordered, duplicate-free, error-free, frame delivery
  - Efficiency
    - How much bandwidth is wasted due to unnecessary transmission

Data Link Control Functions

- Establish an active connection over an already existing transmission link
- Maintain a sustained exchange of data frames between the sender and the receiver
- Cope with abnormal conditions such as link failure
- Initiate the termination process to relinquish the control of the link

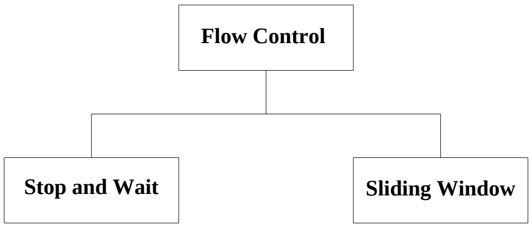
Data Link Control Requirements

- **Frame Segmentation**
  - **Break long streams of data into more manageable, less error prone data frames**
- **Frame Synchronization**
  - **Allow the receiver to correctly interpret transmitted data frames**
    - **The beginning and end of each frame must be recognizable**
- **Flow Control**
  - **Regulate the traffic to prevent the sender from overflowing the buffers of the receiver**
- **Error Control**
  - **Detect and recover from potential frame errors**
    - **Commonly used error detection methods include, Parity bit, Longitudinal, Vertical and Cyclic Redundancy Check.**

Why Flow Control?

- **Receiver must perform a certain amount of processing on the frame header**
  - **Sender may attempt to send frames faster than the receiver can process them**
- **Receiver may buffer incoming frames before delivery to higher level protocols**
  - **Buffers may overflow if the high level protocol user is slow in retrieving frames**
- **Receiver, usually switch or router, may buffer incoming frames for retransmission on a different I/O port**
  - **Output buffers may overflow if the incoming flow rate exceeds the outgoing flow rate**
- **Flow control allows the receiver to regulate the flow of frames sent by a source and potentially limit the rate of incoming data**

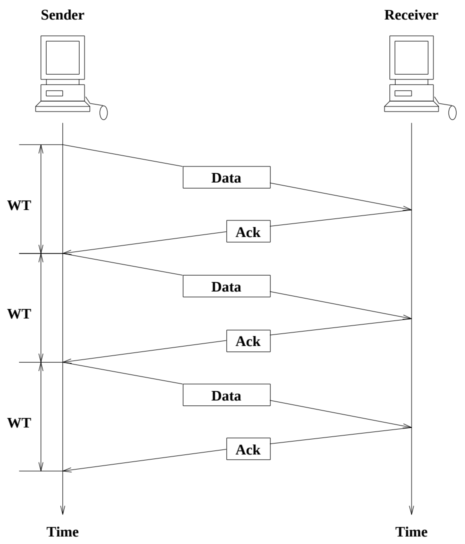
Flow Control Techniques



DLC Flow Control Stop-and-Wait

- **The simplest form of flow control is stop-and-wait**
- **The sender transmits a frame and waits for an acknowledgment from the receiver**
  - **Based on the receiver's response, or lack thereof, the sender either sends the next frame or retransmits the last one**
- **If the two stations are simultaneously exchanging data, acknowledgements may be piggybacked in data frames**

DLC Flow Control
Stop-and-Wait



WT: Waiting Time

Stop-And-Wait Flow Control

- **The main advantage of Stop-And-Wait is its simplicity**
- **Its major drawback is the time spent by the sender waiting for acknowledgements**
  - **The waiting time increases with the link propagation delay, causing poor performance of the scheme**

Flow Control
Sliding Window Protocol

- **Sliding window protocol allows multiple frames to be in transit at one time.**
- **Prior to data transfer, the sender and the receiver agree on an initial window size**
  - **The window size determines the number of frames the sender can send without waiting for an acknowledgement**
- **The efficiency of the sliding window protocol depends on the size of the window and the link propagation delay**

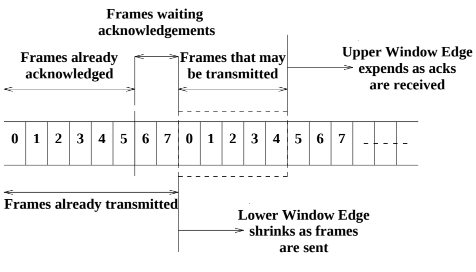
Flow Control
Window Management

- **The sender maintains a sending window of the frames it is allowed to send**
- **The receiver maintains a receiving window of the frames it expects to receive**
- **Each frame is uniquely identified by a sequence number**
- **The receiver acknowledges frames by sending an acknowledgement containing the sequence number of the next frame expected**
  - **Multiple frames can be acknowledged simultaneously, as an acknowledgement containing a sequence number $N+1$ may be used to acknowledge all frames including N**

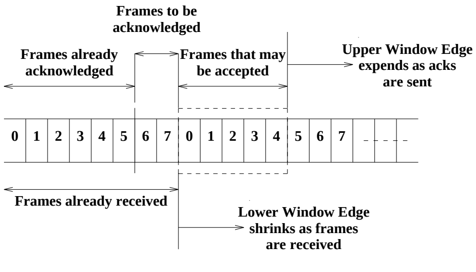
Sliding Window Parameters

- A window is characterized by :
  - Upper Window Edge (UWE)
  - Lower Window Edge (LWE)
  - K, the size of the window
- The frame flow is stopped if the difference between UWE and LWE becomes equal to K

Sliding Window Depiction

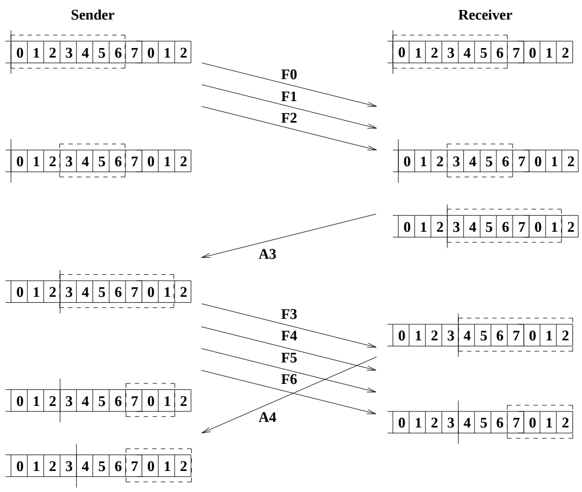


Sender's Perspective



Receiver's Perspective

Sliding Window Dynamics



Max Window Size of 7

Sliding Window Size

- Sequence numbers occupy fields of bounded size
  - A k-bit field produces a range of sequence numbers from 0 to $2^k - 1$
  - Frames are numbered modulo 2^k
- Given a k-bit sequence number field, what should be the maximum window size allowed?

Sliding Window Size
Wrap Around Problem

- Assume that the frame sequence numbers are modulo 2^k and the window size is 2^k
  - For $k = 3$ sequence numbers are modulo 8 and window size is 8
  - Sender sends frame 0 and receives Ack 1
  - Sender expands its window, sends 1,2,3,4,5,6,7 and 0 and receives Ack 1
    - This could mean that all eight frames were received correctly and Ack 1 is cumulative, or it could also mean that all 8 frames were lost and Ack 1 is a duplicate of the previous Ack 1
  - Limiting maximum window size to 7 eliminates this problem
- In general, if frames are numbered modulo 2^k, the maximum window can only be $2^k - 1$

DLC Error Control

- Errors may occur during transmission
  - Lost frames fail to arrive at the destination
  - Damaged frames are not recognizable
- Automatic Repeat Request (ARQ) is the most common technique used to control errors over transmission links
  - Error detection
  - Positive acknowlegments are sent in response to error-free frames
  - Negative acknowlegments are sent in response to frames in error
  - Retransmission after timeouts in response to absence of acknowledgments

Error Control

- Regardless of the design of the transmission system, there will be errors
  - While thermal noise errors may be reduced to vanishingly small rates, bursty impulse noise may still result in substantial errors

Error Control

- Errors can occur both at the bit and the packet level
  - At the bit level, bit corruption due to noise
  - At the packet level, packet loss, duplication, or delivery out of order

Error Control

- Bit level error control use redundancy to detect, and in some cases, correct errors
- Bite level error control is usually enforced on a hop-by-hop basis
  - Hop-by-hop error control is usually required for links with high error rates
- Higher level errors are enforced end-to-end

Block Codes

- Block code based error control schemes use redundancy to detect errors
  - The coded stream contains blocks of bits in direct correspondence to a block of data bits
    - k bits are encoded in n bits so that $(n-k)$ bits are used for redundancy

Error Detection Techniques Parity Check

- The simplest error detection scheme is to append a parity bit to the end of each word in the frame
- Even Parity:
  - The parity bit, ϕ_{even}, is set as :

$$\phi_{even} = \sum_{i=1}^{n} d_i \ \ modulo \ \ 2$$

  - d_i is the ith data bit (0 or 1)
- Odd Parity:
  - The odd bit, ϕ_{odd}, is set as :

$$\phi_{odd} = \sum_{i=1}^{n} d_i \ \ modulo \ \ 2 \ XOR \ 1$$

  - d_i is the ith data bit (0 or 1)

Parity Bit

- If two, or any even number of bits, are inverted, an undetected error occurs
- Another problem with the use of parity bits is that noise impulses are often long enough to destroy more than one bit, particularly at high data rates
  - May not be very effective in data communication networks

Error Detection Techniques
Row and Column Parity

- **Vertical Redundancy Check:**
  - **Add a row r such that bit, $\phi_{i,r}$, is set as :**

$$\phi_{i,r} = \sum_{k=1}^{n} d_{ik} \ \ modulo \ \ \mathbf{2}$$

- **Longitudinal Redundancy Check:**
  - **Add a column c such that bit, $\phi_{c,j}$, is set as :**

$$\phi_{c,j} = \sum_{k=1}^{m} d_{kj} \ \ modulo \ \ \mathbf{2}$$

where m is the number of characters and n is the number of bits per character, and d_{ij} is the bit j of character i

VRC and LRC

Character Parity Bits

Character 1	d_{11}	d_{12}	d_{13}	. . .	d_{1n}	f_{1r}
Character 2	d_{21}	d_{22}	d_{23}	. . .	d_{2n}	f_{2r}
Character 3	d_{31}	d_{32}	d_{33}	. . .	d_{3n}	f_{3r}
.		. . .		. . .	. .	.
.		. . .		. . .	. .	.
.		. . .		. . .	. .	.
Character m	d_{m1}	d_{m2}	d_{m3}	. . .	d_{mn}	f_{mr}
Block Check Character	f_{c1}	f_{c2}	f_{c3}	. . .	f_{cn}	f_{cr}

Rectangular Code
Reliability

- **A single bit error, in one row or in one column, is detected as a parity failure and corrected**

Initial String

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

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

Transmitted

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

Received

	1	0	1	1	0	0	1
	0	1	1	1	0	0	1
Inverted →	$\boxed{1}$	0	1	0	0	1	0 ← Wrong Parity
	1	0	1	0	0	0	0
	0	0	1	1	0	1	1
	0	1	0	1	0	0	0
	0	0	1	0	0	0	1
	↑ Wrong Parity						

Error Detected and Corrected

- **It two errors occur in the same row or column, the code can only detect them**
  - **Error cannot be corrected**

Rectangular Code
Reliability

- **In general, rectangular codes are easy to compute**
  - **It can detect and correct single bit errors**
  - **It can detect multiple-bit errors, but may not correct them**
- **Before computing the code, at least one row of bits must be accumulated in memory**
  - **May introduce coding delay**

Error Detection Techniques
Cyclic Codes

- **The technique is known as Cyclic Redundancy Check (CRC)**
  - **Given a k-bit frame message, the transmitter generates an n-bit sequence, CRC, so that the resulting frame, consisting of k+n bits, is exactly divisible by a generator polynomial**
  - **The receiver divides the incoming frame by the same generator polynomial, and if there is no remainder, assumes the message is correct**

Cyclic Codes

- **The main idea is based on Modulo 2 arithmetic**
- **Polynomials**
- **Shift registers and exclusive-or gates**

Cyclic Codes
Modulo 2 Arithmetic

- **Modulo 2 arithmetic uses binary addition with no carries, which is just exclusive-or operation**

$$\begin{array}{cccc} 1 & 1 & 1 & 1 \\ \multicolumn{4}{l}{+} \\ 1 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 1 \end{array} \qquad \begin{array}{cccccc} & 1 & 1 & 0 & 0 & 1 \\ \multicolumn{6}{l}{\times} \\ & & & & 1 & 1 \\ \hline & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & \\ \hline 1 & 0 & 1 & 0 & 1 & 1 \end{array}$$

Cyclic Codes
Polynomials

- M **= k-bit message, the first k bits of T**
- F **= n-bit FCS, the last n bits of T**
- T **= (k+n)-bit frame to be transmitted, with n<k**
- P **= Pattern of n + 1 bits**
  - P **is a predetermined divisor**
- **The objective is to produce a frame** T **such that** T/P **has no remainder**

Cyclic Codes
Polynomials

- **It should be clear that** $T = \mathbf{2}^n M + F$
  - **Multiplying** M **by** $\mathbf{2}^n$ **causes it to be shifted to the left by n bits and padded with 0's**
  - **Adding** F **results in the concatenation of** M **and** F **which is** T

Cyclic Codes
Polynomials

- **How must** T **be constructed to be exactly divisible by** P**?**
  - **Notice that** $\frac{\mathbf{2}^n M}{P} = Q + \frac{R}{P}$**, where** Q **is a quotient and** R **is a remainder**
  - **Since division is binary, the remainder is always one bit less than the divisor**
    - **Therefore, use** R **as the Frame Check Sequence**
    - $T = \mathbf{2}^n M + R$

Cyclic Codes
Polynomials

- $T = \mathbf{2}^n M + R$ **satisfies the condition**
- $\frac{T}{P} = \frac{\mathbf{2}^n M + R}{P}$
- $\frac{T}{P} = Q + \frac{R}{P} + \frac{R}{P}$
  - $\frac{R}{P} + \frac{R}{P} = \mathbf{0}$**, modulo 2 addition (XOR)**
- $\frac{T}{P} = Q$**, is divisible by** P

Cyclic Codes
Sending Procedure

- **Given a k-bit message** M**, a (n+1)-bit pattern** P**, produce the message** T **to be transmitted**
  - **1. Multiply** M **by** $\mathbf{2}^n$
  - **2. Divide the product by** P**, to produce a remainder** R
  - **3. Add the remainder** R **to** $\mathbf{2}^n M$ **to produce** T
  - **4. Transmit** T

Cyclic Codes
Receiving Procedure

- **Receiver receives T and divides it by P**
  - **If the receiver produces no remainder, it is assumed that there have been no errors**
  - **A non-zero remainder signals the presence of errors**

Cyclic Codes Analysis

- **An error will only be undetectable if it is divisible by $P(X)$**
  - $P(X) = \sum_{i=0}^{n} a_i x^i$, **where a_i represents the coefficient corresponding to the bit in the binary number**
  - $P = 110011$ **results in :**
    - $P(X) = X_5 + X_4 + X + 1$
- **It can be shown that all of the following are not divisible by P and hence are detectable**
  - **All single-bit errors**
  - **All double-bit errors, as long as $P(X)$ has at least three terms**
  - **Any odd number of errors, as long as $P(X)$ contains a factor $(X + 1)$**
  - **Any burst of errors for which the length of the burst is less than the length of the FCS**
  - **Most of large burst errors**

Standard Generator Polynomials

- **CRC-12:** $1 + X + X^2 + X^3 + X^{11} + X^{12}$
- **CRC-16:** $1 + X^2 + X^{15} + X^{16}$
- **CRC-ITU:** $1 + X^5 + X^{12} + X^{16}$
- **CRC-32:**
  $1 + X + X^2 + X^4 + X^5 + X^7 + X^8 + X^{10} +$
  $X^{11} + X^{12} + X^{16} + X^{22} + X^{23} + X^{26} + X^{32}$

CRC Implementation

- **The CRC process can be implemented as a dividing circuit consisting of XOR gates and a shift register**
  - **The shift register is a string of 1-bit storage device,**
    - **Each device has an output line which indicates the value currently stored, and an input line**

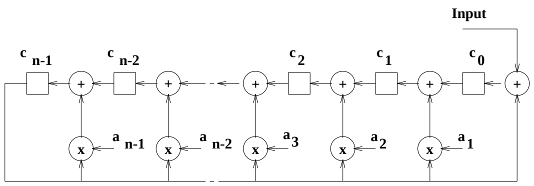
CRC Implementation

- The circuit is implemented according to the following rules:
  - The register contains n bits, equal to the length of the FCS
  - There are up to n XOR gates
  - The presence or the absence of a gate corresponds to the presence or the absence of a term in the divisor polynomial, $P(X)$

CRC Implementation

- General CRC architecture to implement divisor

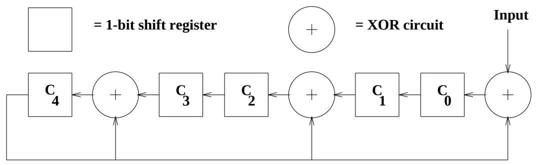
$$1 + a_1X + a_2X^2 + \cdots + a_{n-1}X^{n-1} + X^n$$



CRC Implementation Example

- $M(X) = X^9 + X^7 + X^3 + X^2 + 1$
  - Message $M = 1010001101$
- $P(X) = X^5 + X^4 + X^2 + 1$
  - Divisor $P = 110101$

CRC Implementation Example



Steps	C_4	C_3	C_2	C_1	C_0	$C_4 \oplus C_3$	$C_4 \oplus C_1$	$C_4 \oplus$ Input	Input	
Init	0	0	0	0	0	0	0	1	1	
1	0	0	0	0	1	0	0	0	0	M
2	0	0	0	1	0	0	1	1	1	e
3	0	0	1	0	1	0	0	0	0	s
4	0	1	0	1	0	1	1	0	0	s
5	1	0	1	0	0	1	1	1	0	a
6	1	1	1	0	1	0	1	0	1	g
7	0	1	1	1	0	1	1	1	1	e
8	1	1	1	0	1	0	1	1	0	(MSB)
9	0	1	1	1	1	1	1	1	1	
10	1	1	1	1	1	0	0	1	0	5
11	0	1	0	1	1	1	1	0	0	Z
12	1	0	1	1	0	1	0	1	0	e
13	1	1	0	0	1	0	1	1	0	r
14	0	0	1	1	1	0	1	0	0	o
15	0	1	1	1	0	1	1	1	-	s

Error Correction Code
Hamming Code

- Error correction is based on ‘‘valid codewords’’
  - A valid codeword is a combination of a user’s data bits and redundant information
  - An ‘‘errored codeword’’ is a valid codeword with one or more corrupted bits
- A key idea in error coding is that we can NOT detect an error if it corrupts a valid codeword in way such that the resulting codeword is identical to another valid codeword
  - Valid codewords must be distant enough so that codewords derived from them cannot resemble other valid codewords

Hamming Distance

- The Hamming distance between two codewords is the minimum number of bit inversions required to transform one codeword into another
  - The Hamming distance between "101101" and "011101" is 2
- In general, to detect up to E errors all valid codewords must be at least $E+1$ apart from each other
  - To correct up to E errors, the minimum distance must be at least $2E+1$

Error Correction
Single Bit Error

- Given any sequence of n bits, a single additional bit can detect single-bit errors
  - The additional bit is enough to distinguish between two states: error or no error
  - Not enough to correct errors
- In general, to correct a single error in a sequence of n bits, enough redundancy bits must be added to distinguish between $n+1$ states
  - No error, error in bit 1, error in bit 2, ... error in bit n
- How about if the errors occur in the redundancy bits?

Redundancy Bits

- Let m be the number of data bits, and r the number of redundancy bits required to correct errors
  - Total number of bits in a codeword is $m+r$
  - r must then distinguish between $m+r+1$ states
    - One state of no error, and $m+r$ states for errors in each location
  - r must satisfy:

$$2^r \geq m + r + 1$$

Hamming Code

- The question still remains how to manipulate these bits to correct errors?
  - Hamming code: perfect code

Perfect Parity Code

- Let $n = m + r$ be the n-bit code word written as :

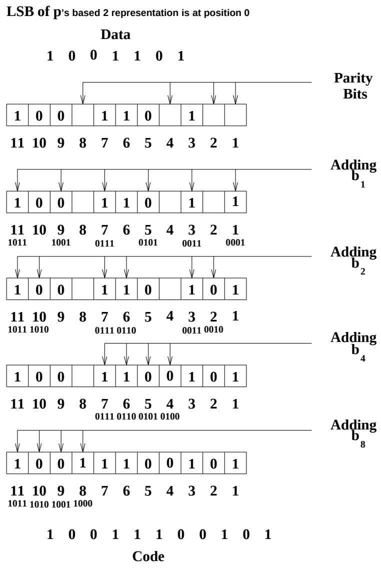
$$b_1, b_2, \ldots, b_n$$

- Parity bits are defined as bits in positions corresponding to powers of 2

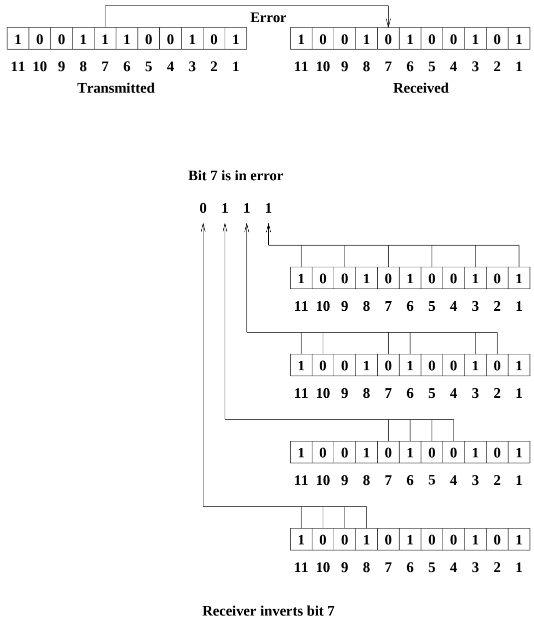
$$b_1, b_2, b_4, \ldots, b_{2^r}$$

- The rule for creating parity bits is that the parity bit b_{2^l} checks all bits b_p whose bit positions p, when written in base 2 representation, have 1 in the lth position

Perfect Parity Code Example



Hamming Code Error Checking



DLL Error and Flow Control
ARQ Standard Techniques

- Two major types of ARQ protocols:
  - Stop-and-Wait ARQ, used typically with character oriented data transmission protocols
    - Also referred to as Idle ARQ
  - Sliding Window ARQ, also referred to as continuous ARQ, used primarily with bit oriented transmission protocols
    - Selective Repeat ARQ
    - Go-back-N ARQ

Stop-and-Wait ARQ

- Stop-and-Wait ARQ is a form of Stop-and-Wait flow control extended to include retransmission of data in case of lost or damaged frames
  - Sender waits until it receives an acknowledgement from the receiver before it sends the next frame
    - A copy of the last frame transmitted is kept in the sender's buffer until the corresponding acknowledgment is received

Stop-and-Wait ARQ
Acknowledgments

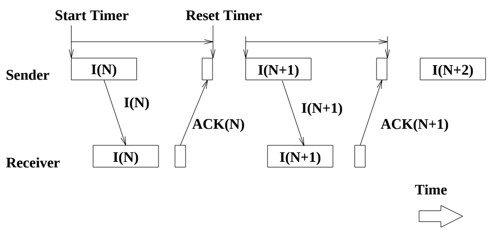
- Acknowledgements may be either positive or negative.
  - Positive acknowledgements (ACK) are issued by the receiver upon receiving an error free frame.
    - An invitation from the receiver to the sender to transmit the next frame.
  - Negative acknowledgements (NAK) are sent by the receiver upon detection of an error.
    - An indication to the sender that the frame needs to be retransmitted.
- To account for possible losses of the acknowledgements, Stop-and-Wait ARQ uses timers
  - At the occurrence of a time out, the sender retransmits the unacknowledged frame.

Stop-and-Wait ARQ
Implementation

- Implicit Retransmission
  - Receiver acknowledges correctly received frames
  - Sender interprets absence of acknowledgments as ‘‘transmitted frame was corrupted’’
- Explicit Request
  - Reciever sends negative acknowledgments to request retransmission of corrupted frames

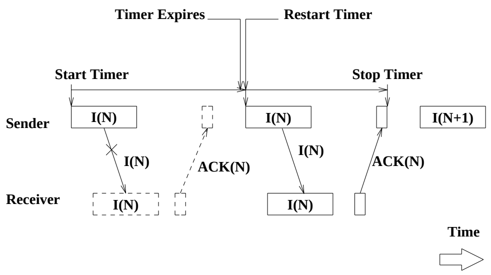
Stop-and-Wait ARQ
Implicit Retransmission

- **Error Free Transmission**



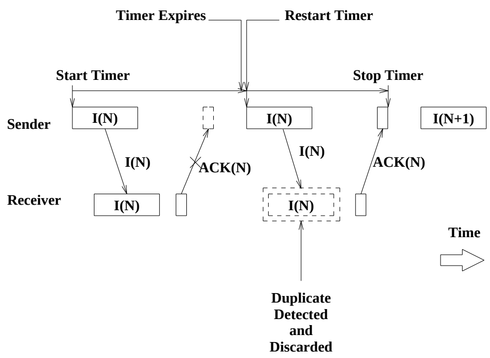
Stop-and-Wait ARQ
Implicit Retransmission

- **Frame Error**



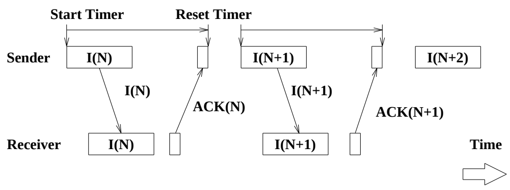
Stop-and-Wait ARQ
Implicit Retransmission

- **Acknowledgment Error**



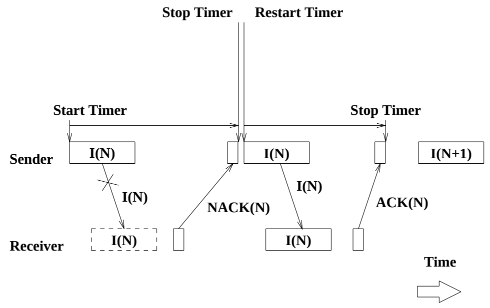
Stop-and-Wait Operation
Explicit Request

- **Error Free Transmission**



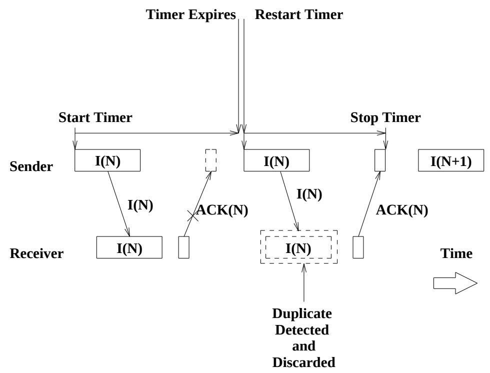
Stop-and-Wait ARQ
Explicit Request

- **Frame Error**



Stop-and-Wait ARQ
Explicit Request

- **Acknowledgment Error**



Sliding Window ARQ

- **Three features are added to extend sliding window flow control to deal with lost and damaged frames**
  - **Sender keeps copies of all transmitted frames until they have been acknowledged**
  - **In addition to ACKs, the receiver has the option of returning a NAK if the received data frame is damaged**
    - **Both ACKs and NAKs must be numbered**
    - **ACKs carry the sequence number of the next frame expected**
    - **NACKs carry the sequence number of the damaged frame**
  - **Sender is equipped with a timer to enable it to handle lost acknowledgments**

Sliding Window ARQ
Timers

- **In sliding window ARQ, a window size of N frames my be sent before an acknowledgment must be received**
- **If N frames are waiting acknowledgments, the sender starts a timer and waits before sending any more frames**
- **If the alloted time runs out and no acknowledgment is received, the sender assumes that the frames are lost**
  - **Sender retransmits one or all of the frames, depending on the protocol**
    - **Two strategies are used, Go-back-N and Selective Repeat**

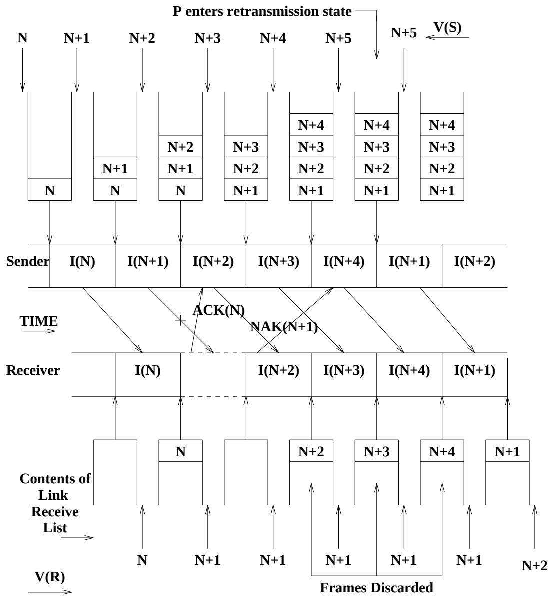
Go-back-N

- Upon detecting an out-of-sequence frame, the secondary informs the primary to start retransmission from a specified frame number
  - A negative ACK is sent
    - Usually referred to as a ''Reject Frame''

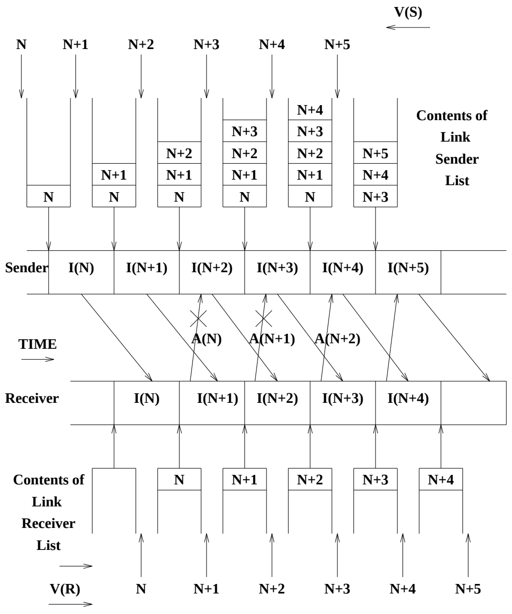
DLC Flow and Error Control
Go-back-N ARQ

- Go-back-N ARQ, a sliding window based-protocol, for error and flow control
  - Go-back-N ARQ allows the sender to transmit a series of data frames, within the sending window, without waiting for an acknowledgement.
  - The receiver uses the multiple acknowledgement scheme to positively acknowledge error free frames.
  - In case of an error, the receiver responds with a NAK containing the sequence number of the frame in error, and discards all further incoming frames that the transmitter may have sent.
  - Upon receiving a NAK, the sender retransmits the frame in question and all succeeding frames.
  - Go-back-N protocol uses timers to detect time-outs.

Go-back-N
Corrupted Information Frame



Go-back-N
Corrupted ACK Frame



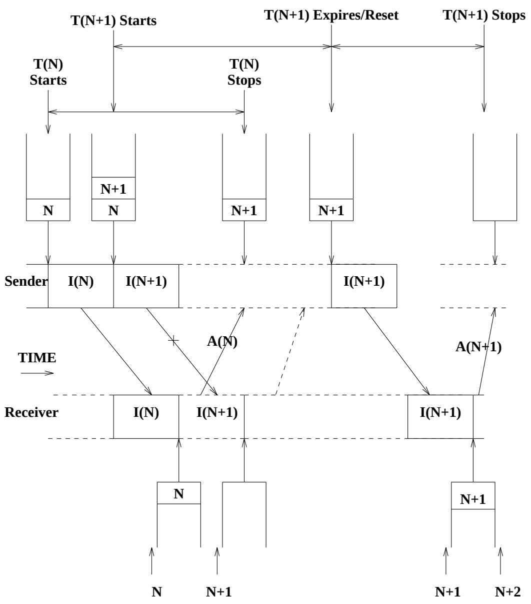
Go-back-N Timeout

- **Loss of an information frame is detected after the next information frame is received**
  - **Requires a continuous stream of information frames being ready to send**
    - **Otherwise, unacceptable long delays may occur, while secondary waits for next frame to be transmitted**

Go-back-N Timers

- **To avoid long delays, primary employs a timeout mechanism**
  - **Timer is started each time primary sends an information frame**
  - **Timer is reset when an ACK is received**
- **Frame is retransmitted when the timer expires**

Go-back-N Timers



Timer Values

- **Timeout interval selected must be greater than the worst-case propagation delay between transmitting a frame and receiving its corresponding acknowledgment**
- **With a timeout mechanism, it is possible for secondary to receive duplicates**
  - **This happens when acknowledgements are lost**

Selective Repeat ARQ

- Selective Repeat uses a more refined approach which requires sender to retransmit only the negatively acknowledged frames
  - Although more efficient, the scheme requires that the receiver contains enough buffer space to hold the out-of-sequence frames
  - More complex logic is required at the sender and the receiver side, respectively to reorder frames
  - An ordered list of the last W received frames may need to be maintained in order to detect duplicates

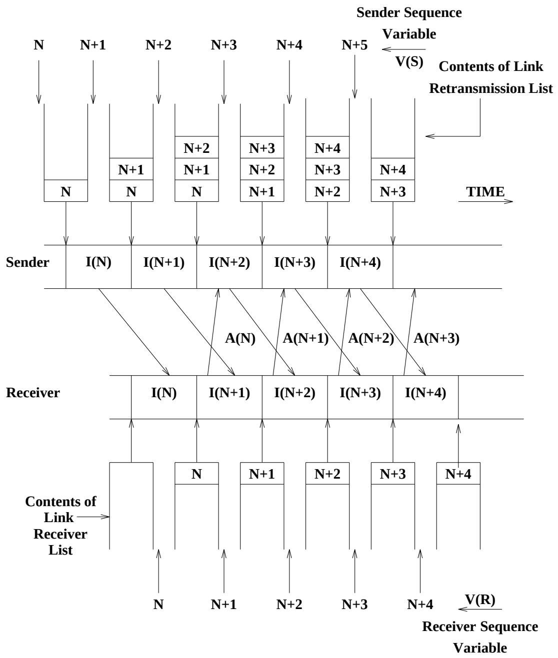
Selective Repeat ARQ
Rules

- The sender stores copies of the unacknowledged frames and retransmits them afer a timeout
- The receiver stores copies of the packets it receives out of order and acknowledges all the correct packets
- The receiver delivers packets to upper layer in the correct order

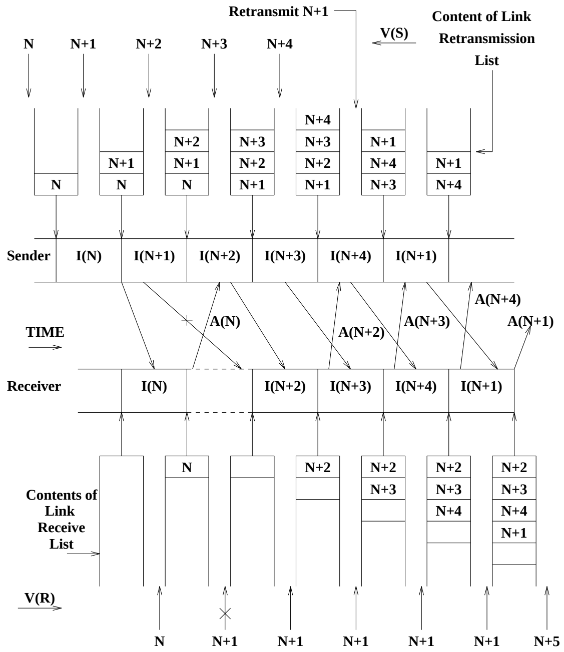
Selective Repeat ARQ
Acknowledgment Strategies

- Selective Repeat ARQ can be implemented using one of two possible strategies:
  - Implicit retransmissions
    - Receiver acknowledges ''individually'' each frame received correctly
    - Sender implicitly determines lost frames from sequence numbers carried by ACKs
  - Explicit retransmission
    - Receiver sends explicit negative ACKs for missing frames

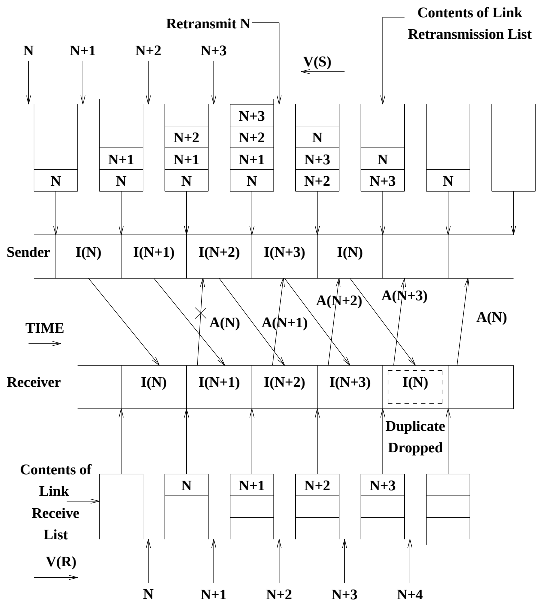
Selective Repeat RQ
Error Free Transmission



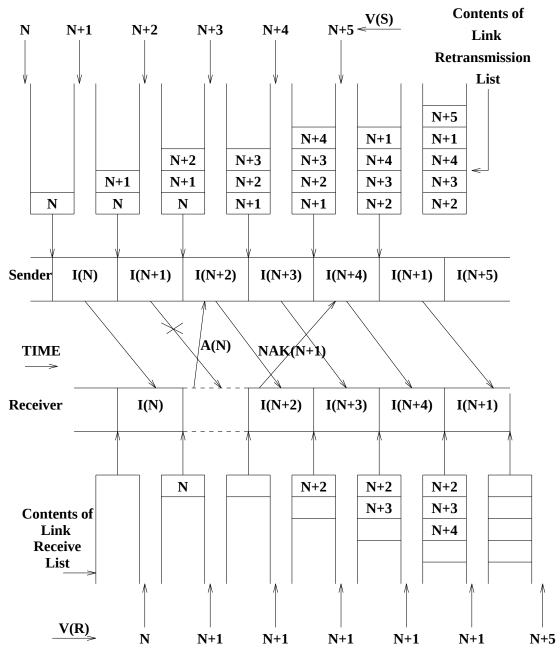
Implicit Selective Repeat Corrupted Information Frame



Implicit Selective Repeat Corrupted ACK Frame



Explicit Selective Repeat Corrupted Information Frame



Selective Repeat ARQ Window Size

- Consider a 3-bit sequence number field and allow a window size of 7
- Sender sends frames 0 through 6 to receiver
- Receiver cumulatively acknowledges with Ack 7
- Ack 7 is corrupted and lost
- Sender times out and retransmits frame 0
- Receiver has advanced its window to accept 7, 0, 1, 2, 3, 4, and 5
  - Receiver concludes that frame 7 is lost and accepts the received frame as the ‘‘new’’ frame 0.

Selective Repeat ARQ Window Size

- Ambiguity in the previous scenario is caused by the overlap between the sending and receiving window
  - To overcome this problem, window size limitation must be more restrictive for Selective Repeat ARQ
- In general, for a k-bit sequence number field, which provides a sequence number range of 2^k, the maximum window size is limited to 2^{k-1}

Selective Repeat ARQ Window Size Rules

- If W is the window size, the frames are numbered consecutively 0,1,2, ..., $2W-1$, 0,1, ...,$2W-1$
- The $(n+W)$th packet cannot be transmitted while the nth packet is still unacknowledged

Effect of Duplicates

- With Go-back-N, duplicates are easily detected and discarded
  - The sequence number of the duplicate differs from the current sequence number of the expected frame
- With selective repeat, duplicates can only be detected if secondary maintains an ordered list of the last N received frames
  - Value of N depends on the flow control mechanism used

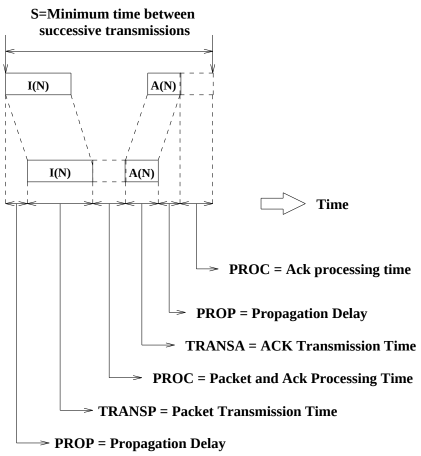
Link Utilization

- The efficiency of a data link protocol is defined as the fraction of time that new packets are being sent,
- Assuming that the sender always has packets to transmit and the link transmission error is e, the efficiency of data link protocol, p, can be defined as:

$$\eta(p,e) = \frac{TRANSP}{S}$$

- S is the total time to transmit a packet

Stop-and-Wait
Link Utilization



- $S = TRANSP + \mathbf{2}PROP + \mathbf{2}PROC + TRANSA$

Stop-and-Wait Link Efficiency
Error Free Link

- **Assuming an error free link, Stop-and-Wait efficiency is:**

$$\eta(S\&W, \mathbf{0}) = \frac{TRANSP}{TRANSP + \mathbf{2}PROP + \mathbf{2}PROC + TRANSA}$$

Stop-and-Wait Link Efficiency
Error Free Link

- **In practice, in most environments where Stop-and-Wait is adequate:**
  - **Time to process an frame,** $PROC$**, is negligible compared to transmission time,** $TRANSP$
  - **Time to process an ACK,** $PROC$**, is negligible compared to transmission time,** $TRANSA$
  - **Also,** $TRANSA$ **is negligible compared to** $TRANSP$
- **The efficiency of error-free Stop-and-Wait is:**

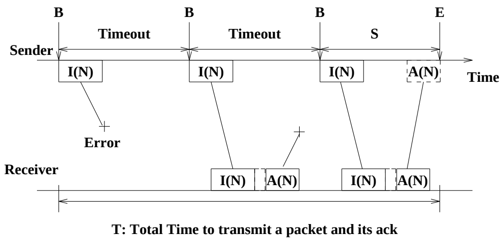
$$\eta(S\&W, \mathbf{0}) = \frac{TRANSP}{TRANSP + \mathbf{2}PROP} = \frac{\mathbf{1}}{\mathbf{1} + \mathbf{2}a}$$

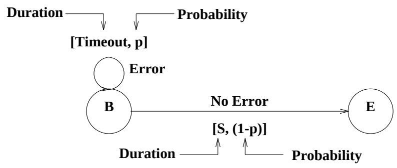
  - $a = \dfrac{PROP}{TRANSP}$

Significance of Parameter a

- **For shorter links** $a \ll \mathbf{1}$
  - **Link utilization is close to 100%, and is independent of the data rate**
    - **Thus, Stop-and-Wait is adequate for modem based data packet networks**
- **For longer terrestrial links, the utilization is high for low data rates, but falls significantly as data rate increases**
  - **Stop-and-Wait utilization is poor for satellite links, even at low data rates**

Stop-and-Wait Link Utilization Error Prone





- **The parameter p represents the probability that a packet or its acknowledgment is in error**

Total Transmission Time Average Value, $E[T]$

- **The average value of the random time, T, can be expressed as follows:**

$$E[T] = (1-p)\times S + p\times(timeout + E[T])$$

- **Solving for $E[T]$, results in :**

$$E[T] = S + \frac{p\times timeout}{(1-p)}$$

Stop-And-Wait Efficiency Error Prone

- **The efficiency of Stop-And-Wait can be expressed as:**

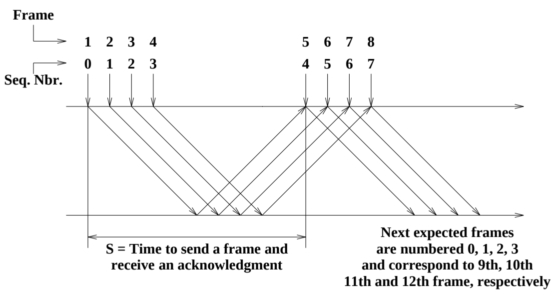
$$\eta(S\&W, p) = \frac{TRANSP}{E[T]}$$

$$\eta(S\&W, p) = \frac{(1-p)\times TRANSP}{(1-p)\times S + p\times timeout}$$

Selective Repeat Protocol Analysis

- **In the absence of errors, the timing diagram of SRP is as follows:**

W= 4



Selective Repeat Protocol Analysis

- **SRP evolution proceeds in cycles of duration S during which W frames are transmissstted**
  - $S = TRANSP + 2PROP + 2PROC + TRANSA$
- **The efficiency of SRP depends on the length of** S **in comparison to** $W \times TRANSP$

Selective Repeat Protocol Analysis

- **Assume that** $W \times TRANSP > S$
  - **The sender transmits continuously since the acknowledgement of the frame numbered 0 is received before frame** $W-1$ **has been transmitted**

$$\eta(SRP,0) = 100\%$$

- **Assume that** $W \times TRANSP < S$
  - **The acknowledgement of the frame numbered 0 is received after the frames 0, 1, ...,** $W-1$ **have been transmitted**

$$\eta(SRP,0) = W \times \frac{TRANSP}{S}$$

- **In general,**

$$\eta(SRP,0) = \min\left(W \times \frac{TRANSP}{S}, 1\right)$$

Selective Repeat Protocol Analysis

- **In this case, we assume errors and** $W = \infty$
  - **Select** *timeout* $> S$
  - **Let** p **be the probability that a transmission error corrupts either a frame or its ACK**
  - **Let** Y **= Average number of times a frame is transmitted**

$$Y = 1 \times (1-p) + 2 \times p(1-p) + \cdots i \times p^{i-1}(1-p) + \cdots$$

$$Y = (1-p) \sum_{i=1}^{\infty} i \times p^{i-1} = \frac{(1-p)}{(1-p)^2} = \frac{1}{(1-p)}$$

Selective Repeat Protocol Analysis With Infinite Buffers

- **Since** $W = \infty$**, the sender never waits and keeps sending continuously**
- **The total time a frame takes to be transmitted successfully is** $\frac{TRANSP}{(1-p)}$

$$\eta_{\infty} = \frac{TRANSP}{\frac{TRANSP}{(1-p)}} = 1 - p$$

Selective Repeat Protocol Analysis With Finite Buffers

- **In this case the following assumptions are made**
  - **Let** p **be the probability that a transmission error corrupts either a frame or its ACK**
  - **Let** $S = W \times TRANSP = timeout$
    - **These assumptions mean that the sender detects a transmission error just when it is about to get an ACK**
  - **It is also assumed that errors are rare**
    - $\mathbf{p \times W \leq 0.1}$

Selective Repeat Protocol Analysis With Finite Buffers

- **Probability of more than one error out of** W **successive transmission, either frame or ACK, is negligeable**

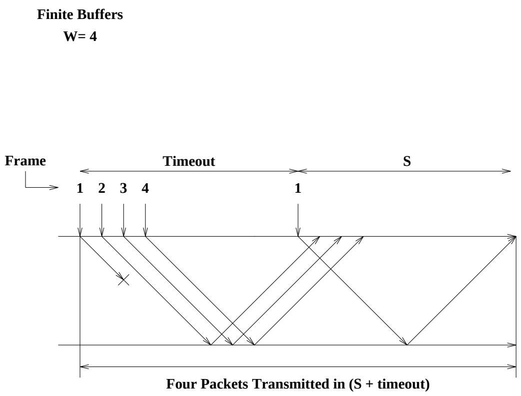
$$Prob(more\ \ than\ \ one\ \ error) =$$

$$\mathbf{1} - (\mathbf{1}-p)^{W} - Wp \times (\mathbf{1}-p)^{W-\mathbf{1}}$$

- **Since** $p \times W$ **is small,** $Prob$ **(more than one error) is also small**

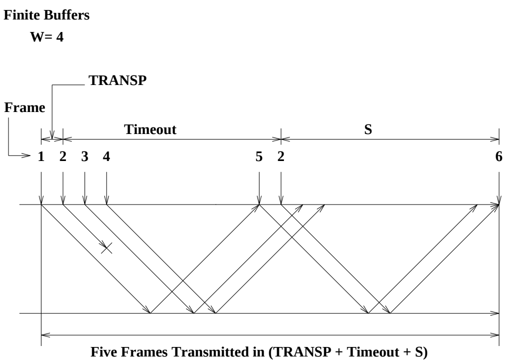
Selective Repeat Protocol Analysis With Finite Buffers

- **Assume frame 1, or its ACK, is lost then four frames are transmitted during the period** $timeout + S$



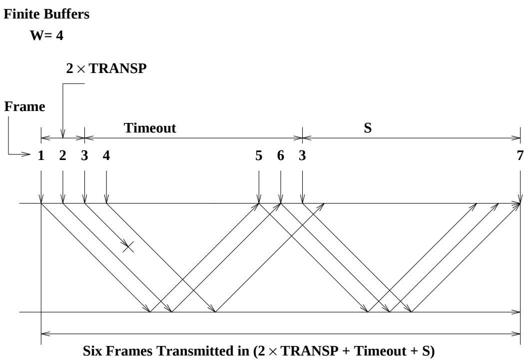
Selective Repeat Protocol Analysis With Finite Buffers

- **If frame 2 or its ACK is lost, then five frames are transmitted during** $TRANSP + timeout + S$



Selective Repeat Protocol Analysis With Finite Buffers

- **If frame 3 or its ACK is lost, then six frames are transmitted during $2 \times TRANSP + timeout + S$**



Selective Repeat Protocol Analysis With Finite Buffers

- **Based on the above, the evolution of SRP can be viewed as a succession of cycles that are statistically equivalent**
  - **A sequence of events starts after the res-transmitted copy has been acknowledged**
  - **Each event occurs with a probability $p(1-p)^3$**
    - **Probability that a specific frame, or its ACK, is lost and not the other frames or ACKs**

Selective Repeat Protocol Analysis With Finite Buffers

- **Let N_i, $(i \geq 1)$ represent the number of frames transmitted if the *ith* frame, or its acknowledgement, is in error**
- **Let T_i, $(i \geq 1)$ represent the duration of time in which N_i packets are transmitted**
  - **For example, assuming W=4 :**
    - $N_0 = 4 = W$ **and** $T_0 = S = W \times TRANSP$
    - $N_1 = 4 = W + (1 - 1)$, **and** $T_1 = S + timeout = 2 \times W \times TRANSP$
  - **In general, for $1 \leq i \leq W$, we can show:**
    - $N_i = W + i - 1$, **and** $T_i = (2 \times W + i - 1) \times TRANSP$

Selective Repeat Protocol Analysis With Finite Buffers

- **Let τ_n = duration of cycle number n (random variable)**
- **Let X_n = Number of packets transmitted during cycle number n (random variable)**

$$(\tau_n, X_n) = \begin{cases} (T_0, N_0) & with\ prob\ (1-p)^W \\ (T_i, N_i) & with\ prob\ p(1-p)^{W-1} \end{cases}$$

where

- $N_0 = W$, **and**
- $T_0 = W \times TRANSP$
- $N_i = W + i - 1$, **and**
- $T_i = (2 \times W + i - 1) \times TRANSP$, **for $1 \leq i \leq W$**

Selective Repeat Protocol Analysis With Finite Buffers

- **Using the fact that $(1-p)^W \approx 1 - pW$ and $p(1-p)^{W-1} \approx p$, we have:**

$$(\tau_n, X_n) \approx \begin{cases} (T_0, N_0) & with\ prob\ (1-pW) \\ (T_i, N_i) & with\ prob\ p \end{cases}$$

- **The efficiency of SRP is :**

$$TRANSP \times \frac{(X_1 + X_2 + \cdots + X_n)}{(\tau_1 + \tau_2 + \cdots + \tau_n)}$$

- **Recall that the law of large numbers states that:**
  - **The arithmetic mean of a large number of statistically equivalent and independent random variables is almost equal to their expected values**

Selective Repeat Protocol Analysis With Finite Buffers

- **Using the law of large number, the efficiency of SRP can be written as :**

$$TRANSP \times \frac{\frac{(X_1 + X_2 + \cdots + X_n)}{n}}{\frac{(\tau_1 + \tau_2 + \cdots + \tau_n)}{n}} = TRANSP \frac{E(X_n)}{E(\tau_n)}$$

$$E(X_n) \approx (1 - Wp)N_0 + p(N_1 + \cdots + N_W)$$

$$E(X_n) \approx W + \frac{pW(W-1)}{2}$$

$$E(\tau_n) \approx (1 - Wp)T_0 + p(T_1 + \cdots + T_W)$$

$$E(\tau_n) \approx W \times TRANSP \times (1 + \frac{p(3W-1)}{2})$$

Selective Repeat Protocol Analysis With Finite Buffers

- **Using the above analysis, the efficience of SRP can be expressed as:**

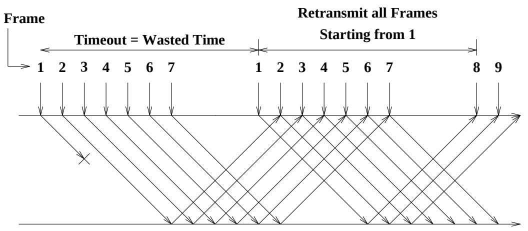
$$\eta(SRP, p) \approx \frac{2 + p(W-1)}{2 + p(3W-1)}$$

Go-Back-N Analysis

- **Let p be the probability that a transmission error corrupts either a frame or its ACK**
- **Let $S = timeout = W \times TRANSP$**
  - **The assumptions mean that the sender detects a transmission error just when it is about to get an ACK**

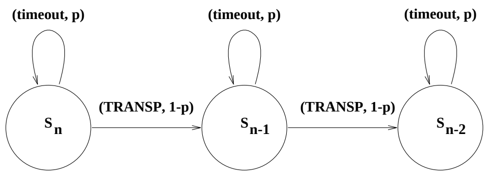
Go-Back-N Analysis

- **The evolution of G-Back-N can be described as :**



Go-Back-N Analysis

- **The state diagram of G-Back-N can be described as :**



- **S_n represents the state where n frames remain to be transmitted**

Go-Back-N Analysis

- **Using a similar argument as in ABP analysis**
  - **The average time per frame (i.e., move from one arbitrary state i to the next adjacent state $i-1$) is :**

$$TRANSP + \frac{p \times timeout}{(1-p)}$$

  - **Go-Back-N efficiency can then be expressed as:**

$$\eta(GBN, p) = \frac{1}{1 + \frac{pW}{1-p}}$$

Classes of Data Link Layer Protocols

- **Based on the framing technique:**
  - **Character-oriented protocol**
    - **Asynchronous and Synchronous**
      - **Control characters are used to delimit frames and to control data interchange.**
  - **Byte-count oriented protocol**
    - **A byte count is included into the frame header to determine the length of the data field.**
  - **Bit oriented protocol**
    - **The objective is to achieve code independence, adaptability, high reliability, and high efficiency.**

Character-Oriented Protocols Asynchronous

- A number of asynchronous character-oriented protocols have been developed
  - Kermit
  - XMODEM, YMODEM and ZMODEM
- No timing coordination is required between the sender and the receiver
  - Not complex
- Used mostly in modem based connections

Character-Oriented Protocols Synchronous

- Character-oriented DLC protocols were also developed for synchronous links
- They use special characters to delineate the various fields of a packet and to control the necessary protocol functions
  - Transparency is achieved by stuffing characters
- Bisync (Binary Synchronous Communication System) is the most popular of synchronous character-oriented protocols
  - Bisync was the de facto standard DLC protocol for synchronous communications.
  - In use since 1968 between IBM computers and batch and video display terminals

Byte-Count Oriented Protocols, DDCMP

- DDCMP, is an acronym for Digital's Data Communication Messages Protocol, developed by Digital Equipment Corporation.
- Keep track of the ''byte count''.
- DDCMP can be used on synchronous or asynchronous data links, half duplex or full duplex transmission, serial or parallel communications, and on multipoint or point-to-point links.
- Balanced protocol (e.g., no controlling station) for point-to-point links.
- Sliding Window Go-Back-N flow control

Bit Oriented Protocols

- Currently the most implemented DLC protocols.
- Bit streams not divided into characters.
- Designed to satisfy a wide variety of data link requirements
  - Point-to-Point and Multipoint links
  - Half-duplex and Full-duplex operations
  - Primary-secondary interactions
  - Links with large and small ratio of propagation time to frame transmission time.

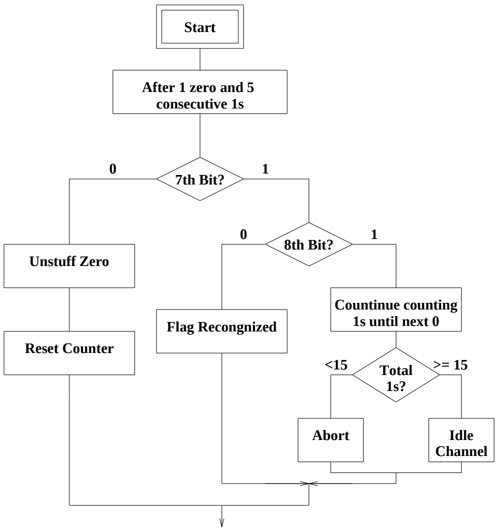
Popular Bit Oriented DLC Protocols

- A number of very similar bit-oriented protocols have achieved widespread use.
- HDLC, High Level Data Link Control, developed by the ISO for standardization (ISO 3309).
- ADCCP, Advanced Data Communication Control Procedures, developed by ANSI (ANSI X3.66) and adopted with very minor changes by the U.S. National Bureau of Standards (equivalent to HDLC).
- LAP-B, Link Access Protocol Balanced, adopted by the CCITT as part of X.25 packet-switching network standard (subset of HDLC).
- SDLC, Synchronous Data Link Control, used by IBM. This is not a standard but it is the original bit-oriented protocol (subset of HDLC).

Bit Stuffing
Sender Perspective

- Flag Fields delimit the frame at both ends with a unique pattern 01111110.
  - Bit stuffing prevents this pattern from occurring in the middle of the data.
- For every sequence of five consecutive 1s of information data, the sender inserts one redundant 0 after the 5th 1.
  - 011111111000 $\rightarrow$ 0111110111000
    - The redundant 0 is inserted regardless of whether the sixth bit is 1 or not

Bit Stuffing
Receiver Perspective

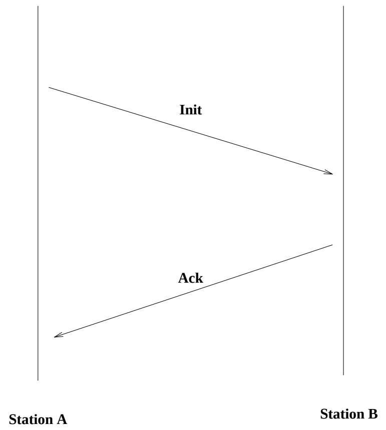


Characteristics of HDLC Environment

- Two link configurations
  - Unbalanced configuration, used in point-to-point and multipoint (not in LAP-B).
  - Balanced configuration, used only in point-to-point operation (not in SDLC).

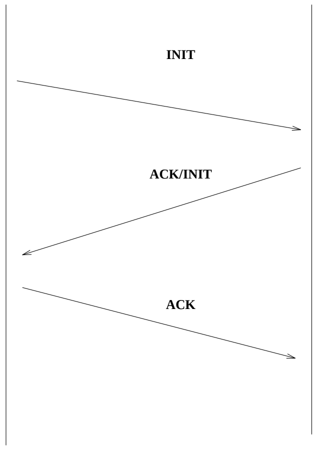
HDLC Link Types

Unbalanced (Master-Slave) Link initialization



**Station A knows initialization has succeeded
Station B Does not know that.**

Balanced Link Initialization



3-way Handshake

HDLC Station Types

- **Three types of stations**
  - **Primary, controls the operations of the link.**
  - **Secondary, operates under the control of the primary station.**
  - **Combined station, combines the features of both primary and secondary stations.**

HDLC Transfer Modes

- **Three data transfer modes of operations**
  - **Normal Response Mode (NRM)**
  - **Asynchronous Response Mode (ARM)**
  - **Asynchronous Balanced Mode (ABM)**
    - **LAP-B**

HDLC Transfer Modes
Normal Response Mode

- Unbalanced configuration.
- The transfer is initiated by the primary, secondary can only respond to a poll.
- Appropriate for Half Duplex

HDLC Transfer Modes
Asynchronous Response Mode

- Unbalanced configuration.
- In this mode the secondary station can initiate a transfer without permission of the primary station.
  - The primary still retains control over the line.
- Appropriate for Full Duplex

HDLC Transfer Modes
Asynchronous Balanced Mode

- Balanced configuration.
- Transfer may be initiated by either station.
- Appropriate for Full Duplex.

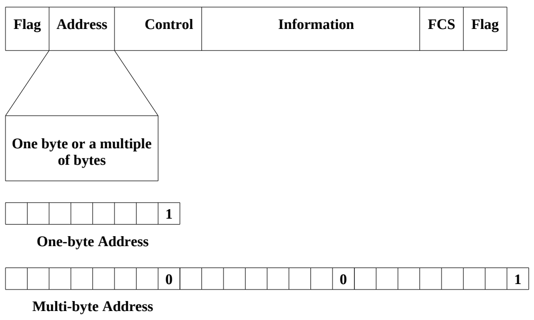
HDLC Frame Format

Flag	Address	Control	Information	FCS	Flag
8 bits	8 bits Extndbl	8 bits Extndbl	Variable	16 or 32 bits	8 bits

Frame Format, Address Field

- **Address Field, used to identify the secondary station that is either the originator or the destination of the frame.**
  - **If a primary station creates the frame, the address contains a to address**
  - **If a secondary station creates the frame, the address contains a from address**
- **Normally, the field is 8 bits long and only 7 bits are used for addressing**
  - **The 8th bit is used to extend the address format and allow variable length addresses as a multiple of 7 bits.**

Extended Adress Field



- **The pattern 11111111 is a broadcast address.**

HDLC Frames

- **3 types of frames**
  - **Information frame (I-frame)**
    - **carries data to be transmitted to end user**
  - **Supervisory frame (S-frame)**
    - **Used for flow and error control**
  - **Unnumbered frames (U-frames)**
    - **Used for control**

Frame Format, Control Field

	1	2	3	4	5	6	7	8
Information	0	N(S)			P/F	N(R)		
Supervisory	1	0	S		P/F	N(R)		
Unnumbered	1	1	M		P/F	M		

Control Field

N(S) : Send sequence number **S : Supervisory function bit**
N(R) : Receive sequence number **M : Unnumbered function bit**
P/F : Poll/Final bit

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	N(S)							P/F	N(R)						
1	0	S		0	0	0	0	P/F	N(R)						

Extended Control Field

Information Frame

- I-frame carries the data to be transmitted to the end user
- Flow and error control data may be piggybacked on I-frames
  - Automatic Repeat reQuest (ARQ) mechanism is used
- Each I-frame contains the Sequence number of the transmitted frame, and the Sequence number of the next frame expected
  - Sequence number of next frame expected is a positive acknowledgement
  - Maximum number of unacknowledged packets is 7 or 127.

Information Frame

- I-frames contain Poll/Final (P/F) bit
  - P bit is a poll bit for commands from primary
  - F bit is a final bit from secondary for responses
    - The primary sets the P bit to 1 to give permission to a secondary to transmit
    - The secondary sets the F bit to 1 on the last frame of its response
  - Characteristics
    - P/F switches control.
    - P requires a response.

Supervisory Frames

- S-frame is used for flow and error control
  - Both go-back-N ARQ (REJ) and selective repeat ARQ (SREJ) are supported
- A received I-frame may be acknowledged by a Receive Ready (RR) frame when no I-frame is available for transmission
- A Receive Not Ready (RNR) frame forces the sender to stop transmission until a subsequent RR frame is issued

Uses of P/F bit

- Polling for data.
  - The primary sets the P bit in an RR frame to poll the secondary.
  - The secondary may respond with either
    - an I-frame (if it has one)
    - an RR frame with F bit set
- Polling for status
  - The primary sets the P bit in an RNR frame to poll the secondary for its status without giving it the permission for transmission
  - The secondary may respond with either
    - RR if it is ready to receive
    - RNR if the station is busy
- ''Primary'' (secondary) can be combined station acting as primary (secondary).

Unnumbered Frames

- U-frames are used for a variety of control functions
- U-frames are defined for commands and responses
  - Mode setting commands and responses
  - Information transfer and responses
  - Recovery commands and responses
  - Miscellaneous commands and responses

Mode setting U-frames

- Transmitted by primary/combined to initialize or change the mode of the secondary/combined
  - Types of commands
    - Commands are SIM, SNRM, SNRME, SARM, SARME, SABM, SABME and DISC
    - SIM (Set Initialization Mode) causes the secondary/combined to initiate a station-specified procedure to initialize data link functions
    - SARME (Set Asynchronous Response Mode Extended)
    - DISC terminates the logical link connection
  - The mode remains in effect until the next mode setting command

Responses to Mode setting U-frames

- UA (= positive unnumbered acknowledgement) with the F bit set to the same value as the received P bit.
- DM indicates a refusal to set the requested mode.
  - The addressed station is logically disconnected
- RD is used to request a disconnect of the logical link

Information transfer U-frames

- To exchange system information between stations.
  - Mainly achieved by the UI-frame (command/response)
- UI-frame information include higher level status, time of the day, link initialization parameters

Recovery U-frames

- **Used when the normal ARQ mechanism does not apply or will not work**
  - **FRMR frame is used to report error in a received frame**
    - **Invalid control field**
    - **Data field too long**
    - **Data field not allowed with received frame type**
    - **Invalid count (A frame that has not been sent is acknowledged)**
  - **RSET command is used to clear FRMR condition**

Miscellaneous U-frames

- **Two miscellaneous commands**
  - **XID used to exchange station identifications and characteristics**
  - **TEST command/response to test that the link and the addressed station are still functioning**
    - **A test command must be echoed with a test response at the earliest opportunity**

HDLC Adavantages

- **Bit oriented protocols are flexible and efficient.**
- **Code independence.**
- **Low overhead.**
- **Not necessarily integral number of bytes**
- **Less vulnerable to bit errors**
- **No transparency ''modes''**

HDLC Limitations

- **Different devices may not communicate if they do not support the same options**
- **Limitation to a single address.**
  - **Adequate for point-to-point and multipoint with a single primary, but not for multipoint.**
- **Initial and trailing flags, primary source of errors, are not covered by the FCS (Frame Check Sequence).**
  - **An error may cause the breaking of a frame or the collapsing of two consecutive ones.**
- **Consecutive 1's can cause loss of synchronization / abort**
- **The reliability achieved by HDLC is not always needed**

Emerging DLCs
Internet Link Layers

- **SLIP, Serial Line IP, devised to connect Sun Workstations to the Ineternet using modems (RFC 1055)**
- **SLIP is used to send raw IP packets over the communication link**
  - **A special flag (0xC00 is used for framing**
  - **Transparecny in IP packets is achieved by (0xDB, 0xDC)**
- **Recent versions of SLIP perform TCP and IP header compression (RFC 1144)**

SLIP Shortcomings

- **SLIP is not an approved IP Standard**
  - **Multiple different versions are available**
- **SLIP does not perform error detection and control**
  - **Recovery from lost, damaged or merged frames are left to higher layers**
- **SLIP supports only IP,**
  - **Protocols that does not support IP as a native protocol are leftout (e.g., Novell LANs)**

SLIP Shortcomings

- **SLIP communicating nodes must know their respective addresses**
  - **No dynamic address assignment**
    - **Serious problem, given the current shortage of IP addresses**
- **SLIP does not support authentication**

Internet Link Layers
PPP

- **Point-to-Point Protocol, is devised to address SLIP shortcomings (RFC 1661, RFC 1662, and RFC 1663)**
- **PPP provides a feature-reach protocol**
  - **Support for multiple protocols, IP, IPX, OSI CLNP, XNS, AppleTalk, ..**
  - **Support for error detection and control**
  - **Dynamic negotiation of IP addresses at setup time**
  - **Support for authentication**

PPP Main Components

- Framing strategy to unambiguously delineate frames
  - Frame format also handles error detection
- A Link Control Protocol (LCP) to control line and negotiate parameters
  - Line initialization
  - Line testing
  - Line options negotiation
  - Line termination
- A Network Layer Protocol (NCP) to negotiate network layer options
  - A different NCP is provided for each network layer supported
    - Provides independence of the network layer protocol

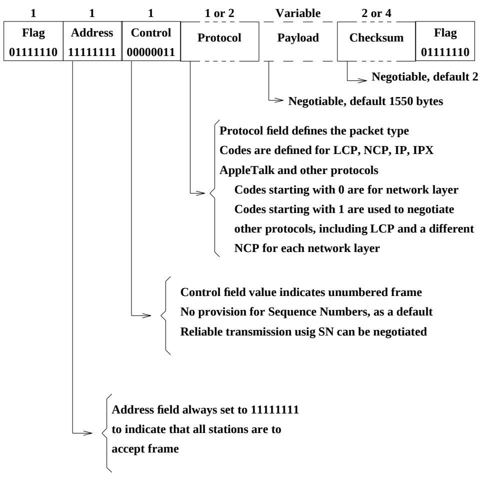
PPP Typical Scenario

- PC uses modem to call ISP router
  - Physical connection is established when ISP router's modem answers
- PC sends ISP router a series of LCP packets within PPP frame(s) payload field
  - LCP packets and responses determine the PPP parameters to be used
- PC sends a series of NCP packets to configure network layer
  - NCP for IP is used to dynamically negotiate IP address
- NCP and LCP packets are used to tear down the network layer and data link layer, respectively
  - Modems terminate physical link

PPP Frame Format

- PPP frame format closely resembles HDLC frame format
  - PPP is character oriented
  - Character stuffing is used on dial-up modem lines to make frame lengths an integral number of bytes
- PPP frame format can be send over diverse lower-layer protocols
  - Serial or parallel, synchronous or asynchronous, low speed or high-speed, electrical or optical, switched or non-switched link.

PPP Frame Format



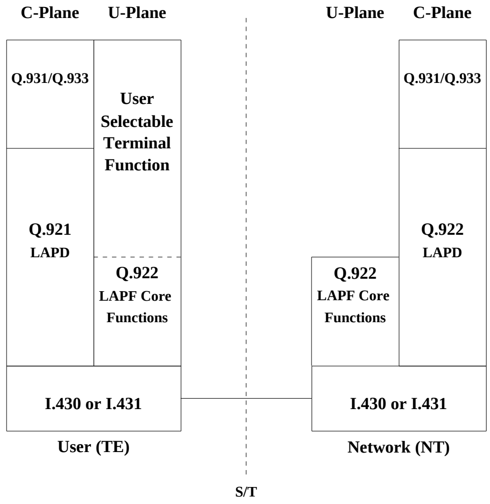
Frame Relay

- Motivation
  - Error rates decreasing
  - Bandwidth (data rate into switch) increasing
  - Processing capability at host increasing
  - Duplication of functionality in existing suites
- Approach
  - shift functionality from switches to source and destination
  - Simplify low-level point-to-point protocols

Frame Relay features

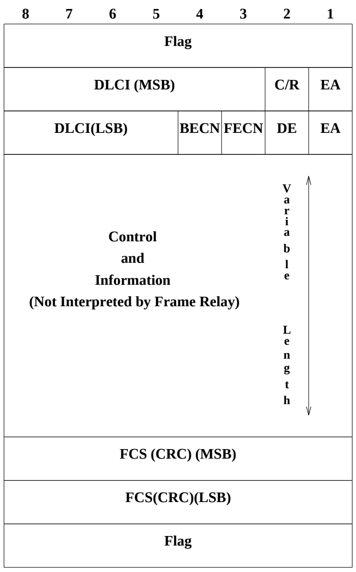
- Framing: bit-oriented
- Error control:
  - Error detected with CRC
  - Corrupted packets discarded, no ARQ
- Congestion control notification capability
  - To support higher layer functions
- Variable length frames

Frame Relay Architecture



C-Plane: D-Channel
U-Plane: D,B, or H Channel

Frame Relay Format



DLCI : Data Link Connection ID
FCS : Frame Check Sequence
CRC : Cyclic Redundancy Check
DE : Discard eligibility
FECN : Forward Explicit Congestion Notification Bit
BECN : Backward Explicit Congestion Notification Bit
EA : Extended Address
C/R : Command/Response Bit

Fields

- Addressing
  - DLCI (Data Link Connection Identifier)
  - Only locally significant
    - Each switch along path may use different number to identify the same circuit
  - 10 Bit, 16 bit and 23 bit address
    - Depending on Extended Address (EA) bits
    - Only 10-bit so far
- FECN/BECN
  - Forward & Backward Explicit Congestion Notification Bit
  - Set at congested switch along the way to notify higher layers

Fields (Continued)

- DE (Discard Eligibility)
  - 1-bit dropping priority
- C/R -- Command/Response bit
  - Carried by the frame relay network without changes

Conclusion

- Discussed DLL flow and error control
- Discussed DLL performance
- Described different classes of DLL Protocols
- Discussed emerging DLL protocols