Data Link Layer (part 2)

Reference:
Chapters 6 and 7 Stallings
Study Guide 6

Review off Lecture 5

- Question - What is a major disadvantage of asynchronous transmission?

Review off Lecture 5

- Question - What is a major disadvantage of asynchronous transmission?
- Answer - Asynchronous transmission requires an overhead of two or three bits per character, and is, therefore, significantly less efficient than synchronous transmission.

Review off Lecture 5

- Question - How is interference avoided by using frequency division multiplexing?

Review off Lecture 5

- Question - How is interference avoided by using frequency division multiplexing?
- Answer - Interference is avoided under frequency division multiplexing by the use of guard bands, which are unused portions of the frequency spectrum between sub-channels.

Review off Lecture 5

- Question - Explain how synchronous time division multiplexing (TDM) works.

Review off Lecture 5

- Question - Explain how synchronous time division multiplexing (TDM) works.
- Answer - A synchronous time division multiplexer interleaves bits from each signal and takes turns transmitting bits from each of the signals in a round-robin fashion.

Review off Lecture 5

- Why is a statistical time division multiplexer more efficient than a synchronous time division multiplexer?

Review off Lecture 5

- Why is a statistical time division multiplexer more efficient than a synchronous time division multiplexer?
- Answer - A statistical time division multiplexer is more efficient than a synchronous time division multiplexer because it allocates time slots dynamically on demand and does not dedicate channel capacity to inactive low speed lines.

Lecture 6 Learning Objectives

- understand the reasons for flow control;
- understand the types of flow control;
- understand the reasons for having error control; and
- understand the types of error control.

Introduction

- Material discussed so far was concerned with sending signals over a transmission link
- For effective digital data communications, much more is needed to control and manage the exchange
  - That is, concerns over sending data over a data communications link
- To achieve the necessary control, a layer of logic is added above the physical interfacing
  - This logic is referred to as data link control or a data link control protocol

Introduction Contd.

- When a data link control protocol is used, the transmission medium between systems is referred to as a data link
- For effective data communication between two directly connected transmitting-receiving stations the following requirements need to be addressed:
  - Frame synchronisation
    - The beginning and end of each frame must be recognisable
  - Flow control
    - The sending station must not send frames at a rate faster than the receiving station can absorb

Introduction Contd.

- Error control
  - Bit errors introduced by the transmission system should be corrected
- Addressing
  - On a multipoint line, such as a LAN, the identity of the two stations involved in a transmission must be specified
- Control and data on the same link
  - The receiver must be able to distinguish control information from the data being transmitted
- Link Management
  - The initiation , maintenance, and termination of a sustained data exchange need to be addressed

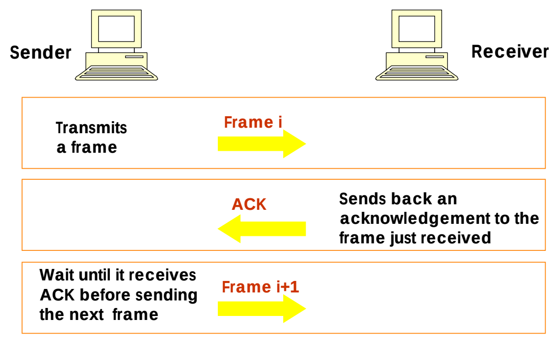
Error and Flow Control

- Synchronization and interfacing techniques are not enough to control and manage error-free exchange of data because of
  - possibility of transmission error
  - Receiver may need to regulate the rate of arriving data
- To achieve necessary control, *flow control* and *error control* mechanisms are necessary

Flow Control

- Technique for controlling the data transmission so that receiver has sufficient buffer space to accept data before processing. This is needed as
  - Receiver buffer is limited
  - with long data transmissions, chances of error are high. More retransmission time.
  - If one station hogs the line, then delay is experienced by other stations
- Transmission time
  - Time taken to emit all bits into medium
- Propagation time
  - Time for a bit to traverse the link
- Common flow control schemes:
  - **Stop-and-Wait**
  - **Sliding Windows**

Stop-and-Wait Flow Control



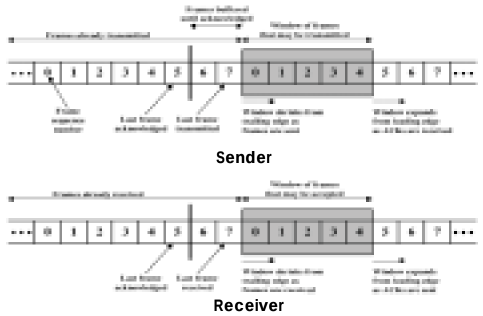
Stop-and-Wait Flow Control

- Works well for a few large frames
- Usually smaller frames are sent
  - Buffer size of the receiver may be limited
  - It is not desirable to permit one station to occupy the medium for long time
- Simple
- Provides inefficient link utilization for
  - Very high data rate
  - Very long distance between sender and receiver

Sliding-Windows

- Improves efficiency greatly by allowing multiple frames to be sent without acknowledgement
- Each frame is numbered for identification purpose
  - eg., 3 bits frame number will allow us to have 2^3 distinct frames which can be numbered from 0- 7
- Size of the window determines the maximum number of frames that can be sent without overflowing the receiver buffer
- Window shrinks (at the trailing edge) when a frame is sent and expands (at the leading edge) when acknowledged
- Window size can't exceed the maximum number of distinct frames

Sliding-Windows

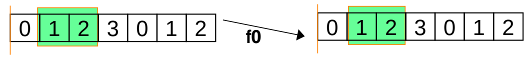


Sliding-Windows: Example

Window size is 3 frames, station A- sender, station- B- receiver

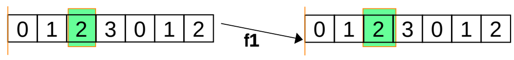
Before any transmission occurred, window size on each stations is 3 frames

A sends f0 and received by B, A's window is reduced to 2 as soon as f0 leaves A. B's window is reduced to 2 only after f0 arrives at B

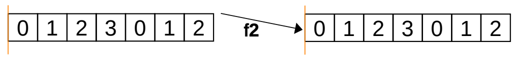


Sliding-Windows: Example…

A sends f1, there is no ACK from B yet



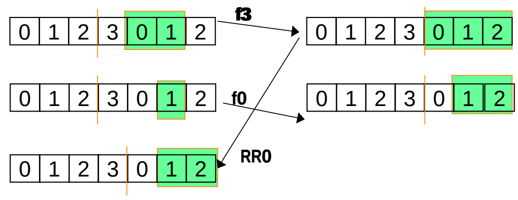
A sends f2, no ACK



B sends acknowledgement of receiving up to frame 2 (3 frames) and A receives this ACK. B's window expands back to 3 frames as soon as RR3 leaves B. A's window size will only expand back to 3 when RR3 physically arrives at A

Sliding-Windows: Example…

A sends F3, B sends acknowledgement of F3, B's window shrinks to 2 frames after receiving F3 and expands back to 3 frames when RR0 is sent, A's window will still remain the same until RR0 physically arrived at A, At the meantime, A keeps sending and sends f0



Error Detection

- Additional bits added by transmitter for error detection code
- Parity
  - simple
  - can't detect position of error
  - even number of bit errors goes undetected
- CRC - Cyclic redundancy check
  - uses dynamic bit pattern attached with data
  - able to detect multiple-bit error

Modulo-2 Arithmetic

- Binary arithmetic without carry.
- Addition is the same as subtraction
- Examples:
  - 1+1=0 1-1=0
  - 1+0=1 1-0=1
  - 0+1=1 0-1=1
  - 0+0=0 0-0=0

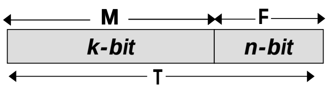
Division

- X is greater than or equal to Y iff the position of the highest 1 bit (i.e., the Most Significant Bit) of X is the same or greater than the position of the highest 1 bit of Y

Division Example

```
          1101010110
110101 | 101000110101110
         110101
          111011
          110101
            111010
            110101
              111110
              110101
                101111
                110101
                  110101
                  110101
                        0 ← Remainder
```

CRC Concept



$T = 2^n M + F$	What should be F?
$\frac{2^n M}{P} = Q + \frac{R}{P}$	Q,R quotient, remainder P is the predetermine divisor
$T = 2^n M + R$	let F equals R
$\frac{T}{P} = \frac{2^n M}{P} + \frac{R}{P}$	divides T by P
$= Q + \frac{R}{P} + \frac{R}{P} = Q$	T is divisible by P

Given a k-bit block of message (M), transmitter generates an n-bit frame check sequence (FCS), so that the resulting frame of k+n bits (T), is exactly divisible by some predetermines divisor (P). The divisor divides the incoming frame (T) by P, if there is no remainder, assume there was no error

CRC

M=***k-bit*** message F=***n-bit*** FCS

P=***(n+1)***-bit divisor T=***(k+n)***-bit frame

- Sender
  - multiply M with 2^n
  - divide that by P, remainder gives F
  - add F to 2^n*M to get T
  - send T to receiver
- Receiver
  - divide T by P
    - no remainder ➡ no error
    - remainder ➡ error

CRC-Example

M=***101010 (6-bit)***

P=***101(3-bit)***

F=***(3-1)-bit=2-bit=?***

T=***(6+2)***-bit frame

M*$2^{(3-1)}$=10101000

```
      100010
101 | 10101000
      101
         0100
          101
           10 ← Remainder
```

So F=10

T=10101000+10=10101010

At receiver:

T/P=10101010/101 ➡ remainder 0

➡ no error

How do we select P ?

- Minimum requirement:
  - One bit longer than FCS employed by data link layer protocol
    - HDLC uses either 16 or 32 bits FCS, hence P(x) for 16 bits FCS has to be at least 17 bits long, or 33 bits long for 32 bits FCS
  - Both high- order (left most) and low- order (right most) bits must be 1, eg 101, 1001, 10001
  - Exact bit pattern depends on the type of error expected

What errors can be detected by CRC?

- All single bit error
- All double bit errors, P(x) used as divisor has to contain at least three 1s
  - Example, $P(X)=X^4+X^3+1$, P=11001
- Any odd number of errors, if P(x) has a factor of (x+1)
- All burst errors with length < length of P(x)
- Some burst errors with length > length of P(x)

Common CRCs

- Four versions of P(X) are widely used
  - CRC-12, $P(X)= X^{12}+ X^{11}+ X^3+ X^2+X+1$
  - CRC-16, $P(X)= X^{16}+ X^{15}+ X^2+1$
  - CRC-CCITT, $P(X)= X^{16}+ X^{12}+ X^5+1$
  - CRC-32, $P(X)= X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11} +X^{10} +X^8 +X^7 +X^5 +X^4 +X^2 +X +1$

Error Control

- Mechanism to:
  - detect errors
  - correct errors
- Two type of errors
  - lost frame/ ack
  - damaged frame/ ack

Error Control Techniques

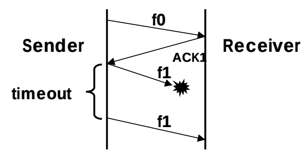
- Error Control Techniques
  - Error detection
    - parity bit, CRC
  - Acknowledgment
    - Positive (RR) - destination receives error- free frame
    - Negative(REJ) - destination receives frame with some errors
  - Retransmission after timeout
    - Source retransmits a frame that has not been acknowledged after a predetermined amount of time
- Collectively above mechanisms are referred as automatic repeat request (**ARQ**)

Error Control Techniques

- Standard Techniques
  - Stop-and-Wait ARQ
  - Go-back-N ARQ
  - Selective-reject ARQ
- Go- back-N ARQ and selective-reject ARQ use sliding-window flow control

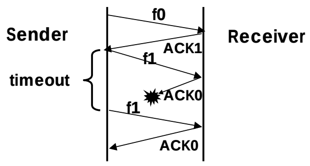
Stop-and-Wait ARQ

- Based on Stop-and Wait flow control
- Possible errors:
  - Damaged frame
    - receiver discards the frame, no acknowledgment sent
    - Sender retransmit the frame after timeout
    - Sender keeps a copy of a transmitted frame until an acknowledgment is received



Stop-and-Wait ARQ

- Possible errors:
  - damaged acknowledgment
    - receiver receives correct frame
    - ACK damaged in transit, not recognizable by sender
    - sender re- sends the frame after timeout
    - Receiver receives a duplicate copy and discard the copy. To identify a duplicate copy, frames are alternately labeled with 0 and 1



Go-Back-N ARQ

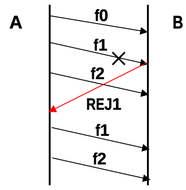
- Based on sliding-window flow control
- Acknowledgments:
  - RR(n) = positive acknowledgment of frame n-1
  - REJ(n)=negative acknowledgment of frame n
- Handles cases:
  - Damage/lost frame
  - Damage/lost RR
  - Damage/lost REJ

Go-Back-N ARQ

- Consider that station A is sending frames to station B.
- Possible situations:
  - Frame contains error and is detected
  - Frame is lost during transmission and sender's window size > 0
  - Frame is lost during transmission and sender's window size =0 (time out occurs)

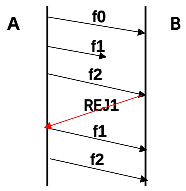
Damaged Frame

A transmit frame *n* ***(f1)***. B detects an error and has previously successfully received frame *(n-1) (* ***f0)***. B sends REJ *n* **(REJ1)** to indicate that frame *n* ***(f1)*** is received with error. When A receives the REJ *n* ***(REJ1***), it will retransmit frame *n* ***(f1)*** and all subsequent frames that it has transmitted since the original transmission of *n**(f1***), ie ***f1*** and ***f2*** in the example



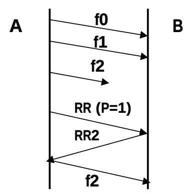
Lost Frame ..

Frame *n* ***(f1)*** is lost in transit. A subsequently sends frame (*n+1*) ***(f2***). B receives frame (*n+1*) ***(f2)*** out of order and send REJ *n* ***(REJ1)*** A must retransmit frame *n* ***(f1)*** and all subsequent frames



Lost Frame ...

Frame *n* ***(f2)*** is lost in transit but A does not send additional frames. B receives nothing and returns neither RR or REJ. When A's timer expires, it transmits RR frame with P- bit of 1 as a command that must be acknowledged by sending RR ***(RR2)*** indicating the next frame that it expects. When A receives the RR, it retransmits frame *n* ***(f2)***.

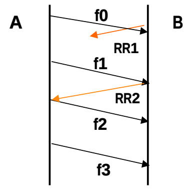


Damaged RR

- B receives frame *i* and sends RR(*i+1*), which is lost in transit
  - Before timeout - if subsequent acknowledgments eg. RR(*i+2)* are received, RR(*i+2*) acknowledged both frame *i* and *i+1.*
  - After timeout - If A's timer expires before subsequent acknowledgment is received, A sends RR command and sets another timer called P-bit timer. If B fails to respond to the RR command, A's P-bit timer will expire. At this point, A will restart the P-bit timer and send a new RR command. If A fails to receive any acknowledgment after a pre-determined number of tries, it initiates reset procedure.

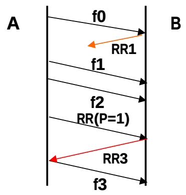
Damaged /Lost RR

- Before timeout - if subsequent acknowledgments eg. RR(*i+2) **(RR2)*** are received, RR(*i+2*) **(RR2)** acknowledged both frame *i (**f0**)* and *i+1 (**f1**).*



Damaged/Lost RR

- After timeout - If A's timer expires before subsequent acknowledgment is received, A sends RR command and sets another timer called P-bit timer



Damaged REJ

- A transmit frame *i* which is received by B with error. B sends REJ*i* which is lost during transmission.
- B waits for A to re-transmit frame *i*, if A's window still allows additional frames to be sent to B, A will send these additional frames and wait for any ACK from B. Since B is still waiting for frame *i*, it will not send any ACK. After sometime, timer on A will expire and A will send RR command with P-bit=1.

Selective-Reject ARQ

- Only frames with negative ACK are retransmitted
- Send **SREJ*n*** to signal that frame *n* needs retransmission
- Advantage:
  - reduce the number of re-transmission required
- Disadvantages
  - requires larger buffer at the receiver to save any frames arrive after frames *i* and before receiving the re-transmitted frame *i.*
  - Requires more complex logic at the sender because it has to be able to send a frame out of sequence.