Chapter 7 Data Link Control

• Objectives: Effective & reliable data communication between two directly connected transmitting-receiving stations

• Requirements:
  ➢ Frame Synchronization: Data are sent in blocks called frames. The beginning & end of each frame must be recognizable.
  ➢ Flow Control: The sending station must not send frames at a rate faster the receiving station can absorb them.
  ➢ Error Detection: There is no 100% reliable data transmission existing, bit errors should at least be detected.
  ➢ Error Control: Bit errors introduced by the transmission system should be corrected.

Flow Control

• Ensuring the sending entity does not overwhelm the receiving entity
  ➢ Preventing buffer overflow

• Transmission time
  ➢ Time taken to emit all bits into medium

• Propagation time
  ➢ Time for a bit to traverse the link

Model of Frame Transmission

Stop and Wait

• Source transmits frame

• Destination receives frame and replies with acknowledgement

• Source waits for ACK before sending next frame

• Destination can stop flow by not send ACK
Stop and Wait Link Utilization

- Large block of data may be split into small frames
  - Limited buffer size
  - Errors detected sooner (when whole frame received)
  - On error, retransmission of smaller frames is needed
- Prevents one station occupying medium for long periods
- Stop and wait becomes inadequate

Sliding Windows Flow Control

- Allow multiple frames to be in transit
- Receiver has buffer W long
- Transmitter can send up to W frames without ACK
- Each frame is numbered
- ACK includes the number of next frame expected
- Sequence number bounded by size of field (k)
  - Frames are numbered modulo $2^k$
Example Sliding Window

Sliding Window Enhancements

- Receiver can acknowledge frames without permitting further transmission (Receive Not Ready)
- Must send a normal acknowledge to resume
- If duplex, use piggybacking
  - If no data to send, use acknowledgement frame
  - If data but no acknowledgement to send, send last acknowledgement number again, or have ACK valid flag (TCP)

Error Detection

- Additional bits added by transmitter for error detection code
- Parity Check
  - Value of parity bit is such that character has even (even parity) or odd (odd parity) number of ones -> even number of bit errors goes undetected
  - Two-dimensional parity: catches all 1-, 2-, and 3-bit errors, and most 4-bit errors

<table>
<thead>
<tr>
<th>Parity bits</th>
<th>Data</th>
<th>Parity byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>0101001</td>
<td>1</td>
<td>1111011</td>
</tr>
<tr>
<td>1101001</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1011110</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0001111</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0110100</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1011111</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Cyclic Redundancy Check

- Theoretical Foundation: a branch of mathematics called “finite fields”
- Add $k$ bits of redundant data to an $n$-bit message
  - want $k << n$
  - e.g., $k = 32$ and $n = 12,000$ (1500 bytes)
- Represent $n$-bit message as $n$-1 degree polynomial
  - e.g., MSG=10011010 as $M(x) = x^7 + x^4 + x^3 + x^1$
- Let $k$ be the degree of some divisor polynomial
  - e.g., $C(x) = x^3 + x^2 + 1$
**CRC (cont)**

- Transmit polynomial $P(x)$ that is evenly divisible by $C(x)$
  - shift left $k$ bits ( $k$ is the degree of $C(x)$ ), i.e., $M(x)x^k$
  - subtract remainder of $M(x)x^k/C(x)$ from $M(x)x^k$

- Receiver polynomial $P(x) + E(x)$
  - $E(x) = 0$ implies no errors

- Divide $(P(x) + E(x))$ by $C(x)$; remainder zero if:
  - $E(x)$ was zero (no error), or
  - $E(x)$ is exactly divisible by $C(x)$

**Selecting $C(x)$**

- It is possible to prove that the following types of errors can be detected by a $C(x)$ with the stated properties:
  - All single-bit errors, as long as the $x^k$ and $x^0$ terms have non-zero coefficients.
  - All double-bit errors, as long as $C(x)$ contains a factor with at least three terms
  - Any odd number of errors, as long as $C(x)$ contains the factor $(x+1)$
  - Any ‘burst’ error (i.e., sequence of consecutive error bits) for which the length of the burst is less than $k$ bits.
  - Most burst errors of larger than $k$ bits can also be detected

**Error Control at Data Link Layer**

- Detection and correction of errors that occur in the trans. of frames
  - Lost frames: a frame fails to arrive the other side.
  - Damaged frames: some bits of the arrival frame are in error

- Error Control Ingredients
  - Error detection
  - Positive acknowledgment (ACK)
  - Retransmission after **timeout**
  - Negative acknowledgement and retransmission (NACK)

- Automatic Repeat Request (ARQ): Three versions of ARQ standardized
  - Stop and wait ARQ
  - Go back N ARQ
  - Selective reject (selective retransmission) ARQ

**Stop and Wait**

- Source transmits single frame
- Wait for ACK
- If received frame damaged, discard it
  - Transmitter has timeout
  - If no ACK within timeout, retransmit
- If ACK damaged, transmitter will not recognize it
  - Transmitter will retransmit
  - Receive gets two copies of frame ->
  - Use ACK0 and ACK1

- Pros and Cons
  - Simple
  - Inefficient
**Go Back N ARQ**

- Based on sliding window
- If no error, ACK as usual with next frame expected
  - Use window to control number of outstanding frames
- If error (damaged/lost), reply with NACK
  - Receiver discard that frame and all future incoming frames until error frame received correctly
  - Transmitter must go back and retransmit that frame and all subsequent frames

**Go Back N – Lost Frame (1)**

- Frame $i$ lost
- Transmitter sends $i+1$
- Receiver gets frame $i+1$ out of sequence
- Receiver sends reject $i$
- Transmitter goes back to frame $i$ and retransmits

**Go Back N – Lost Frame (2)**

- Frame $i$ lost and no additional frame sent
- Receiver gets nothing and returns neither acknowledgement (ACK) nor rejection (REJ)
- Transmitter times out and sends acknowledgement frame with P bit set to 1
- Receiver interprets this as command which it acknowledges with the number of the next frame it expects (frame $i$) – by sending RR
- Transmitter then retransmits frame $i$

**Go Back N - Damaged Frame**

- Receiver detects *error* in frame $i$
- Receiver sends rejection-$i$ (REJ)
- Transmitter gets rejection-$i$ (REJ)
- Transmitter retransmits frame $i$ and all subsequent
Go Back N – Damaged/lost Acknowledgement (RR)

- Receiver gets frame $i$ and send RR $(i+1)$ which is lost
- Acknowledgements are cumulative, so next acknowledgement $(i+n)$ may arrive before transmitter times out on frame $i$
- If transmitter times out, it sends acknowledgement with P bit set as before (Lost Frame (2))
- This can be repeated a number of times before a reset procedure is initiated

Selective Reject ARQ

- Also called selective retransmission
- Only rejected frames are retransmitted
- Subsequent frames are accepted by the receiver and buffered
- Pros: minimizes retransmission -> more efficient
- Cons:
  - **Receiver** must maintain *large enough buffer* to save post-SREJ frames until the frame in error is retransmitted
  - More complex logic in **transmitter**: be able to send out a frame out of sequence.

Selective Reject - Diagram

Maximum Seq # and Maximum Window Size

- Note: in real world, most of the transmissions are full-duplex, therefore, *temporarily delaying* outgoing ACK so that they can be hooked onto the next outgoing data frame -> **Piggybacking**
- Maximum Seq #: $S = 2^m$
- Maximum Window Size: $W_s$
  - Without error or "out of order", Sliding Window Flow Control: $W_s = 2^m$ *(textbook error on P199, P211)*
  - Go-Back-N ARQ: $W_s = 2^{m-1}$ *(textbook error on P211)*
  - Selective Repeat ARQ: $W_s = 2^{m-1}$

Reading Assignment: P274-287 of Leon-Garcia; Textbook: relevant pages; P222-227 of Tanenbaum;
Maximum Window Size in Go-Back-N ARQ