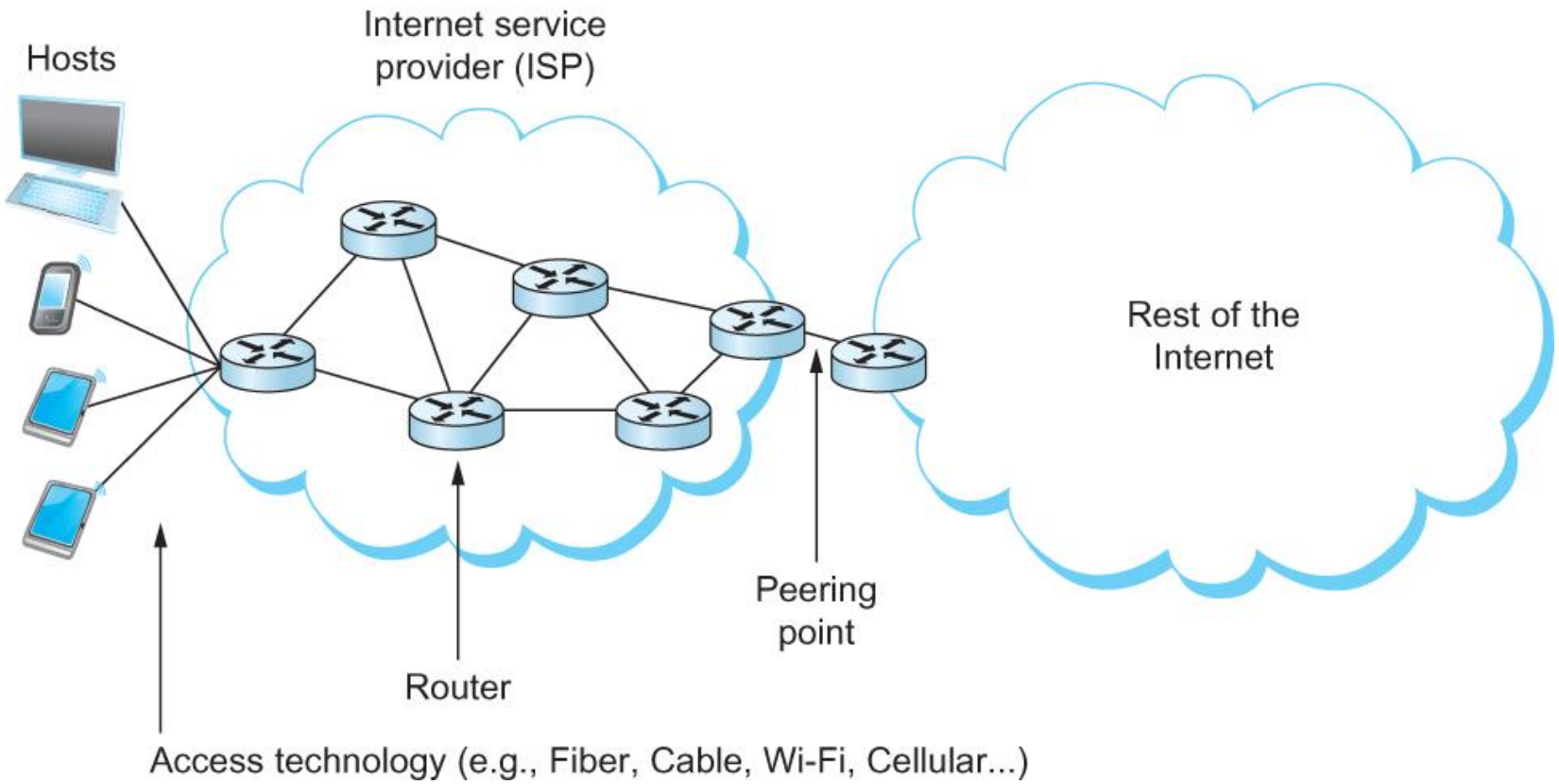
Lecture 3
Getting Connected

Perspectives on Connecting



An end-user's view of the Internet

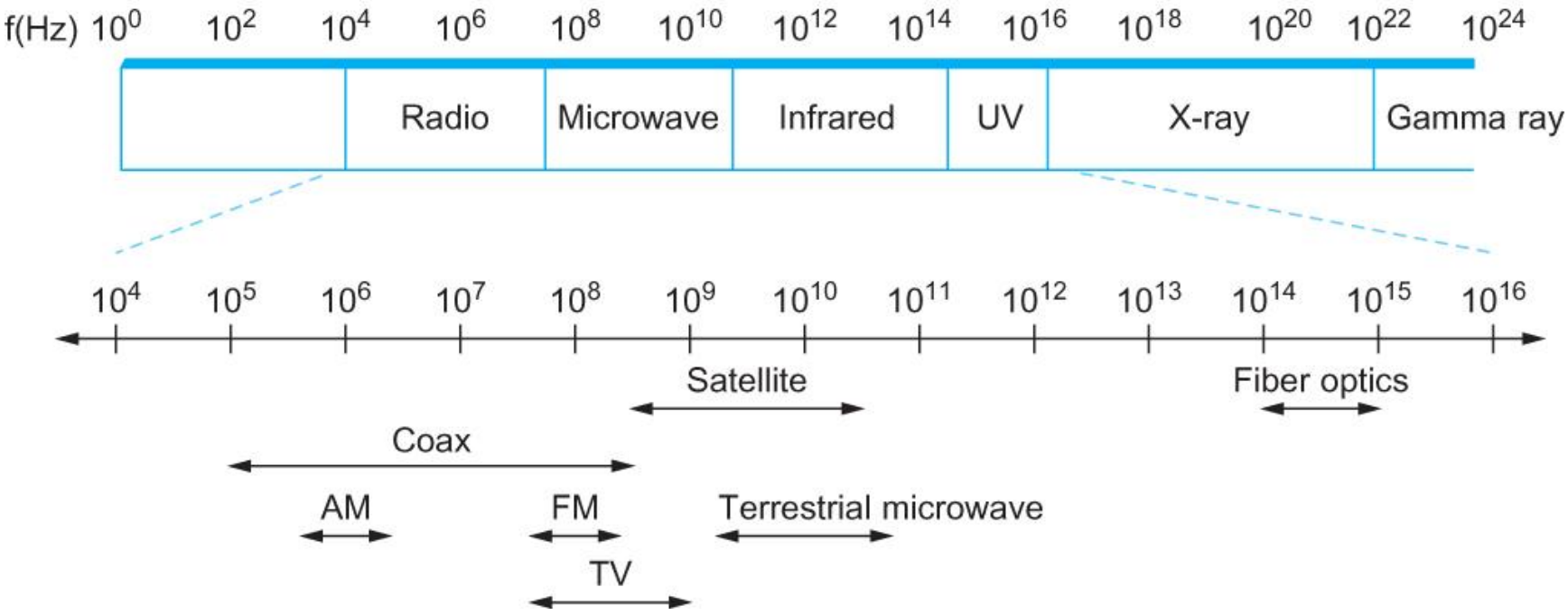
Link Capacity and Shannon-Hartley Theorem

- Gives the upper bound to the capacity of a link in terms of bits per second (bps) as a function of signal-to-noise ratio (SNR) of the link measured in decibels (dB)
- $C = B \times \log_2(1+S/N)$
  - Where $B = 3300 - 300 = 3000$Hz, S is the signal power, N is the average noise power
  - The SNR is measured in decibels as $\text{dB} = 10 \times \log_{10}(S/N)$. If SNR is 30 dB, then $S/N = 1000$
  - Now $C = 3000 \times \log_2(1000) = 30$ kbps

Links

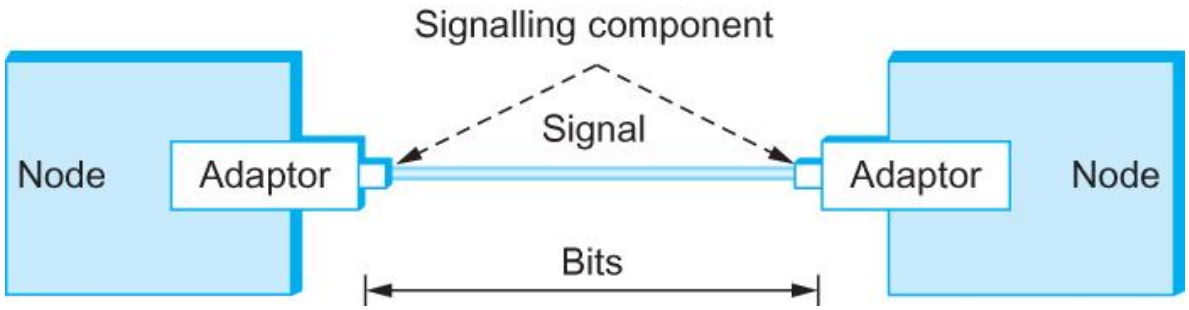
- All practical links rely on some sort of electromagnetic radiation propagating through a medium or, in some cases, through free space
- One way to characterize links, then, is by the medium they use
  - Typically copper wire in some form (as in Digital Subscriber Line (DSL) and coaxial cable),
  - Optical fiber (as in both commercial fiber-to-the home services and many long-distance links in the Internet's backbone), or
  - Air/free space (for wireless links)
- Another important link characteristic is the frequency
  - Measured in hertz, with which the electromagnetic waves oscillate
- Placing binary data on a signal is called encoding
- Modulation involves modifying the signals in terms of their frequency, amplitude, and phase

Links (2)

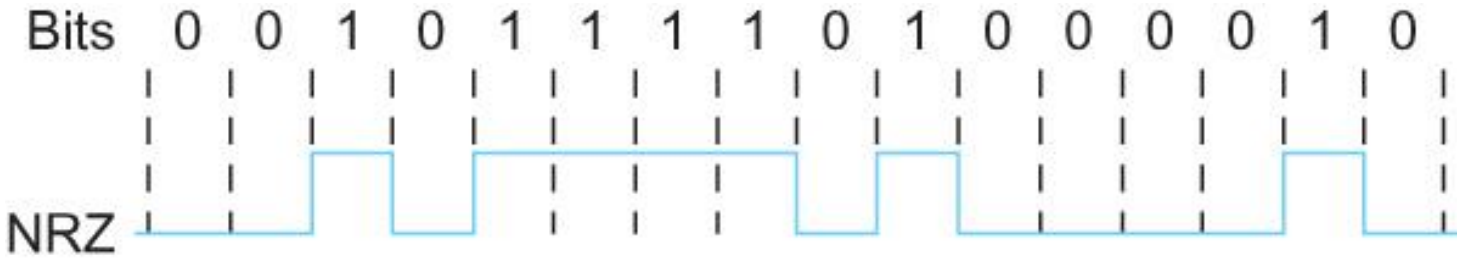


Electromagnetic spectrum

Encoding (1)



Signals travel between signaling components; bits flow between adaptors



NRZ encoding of a bit stream

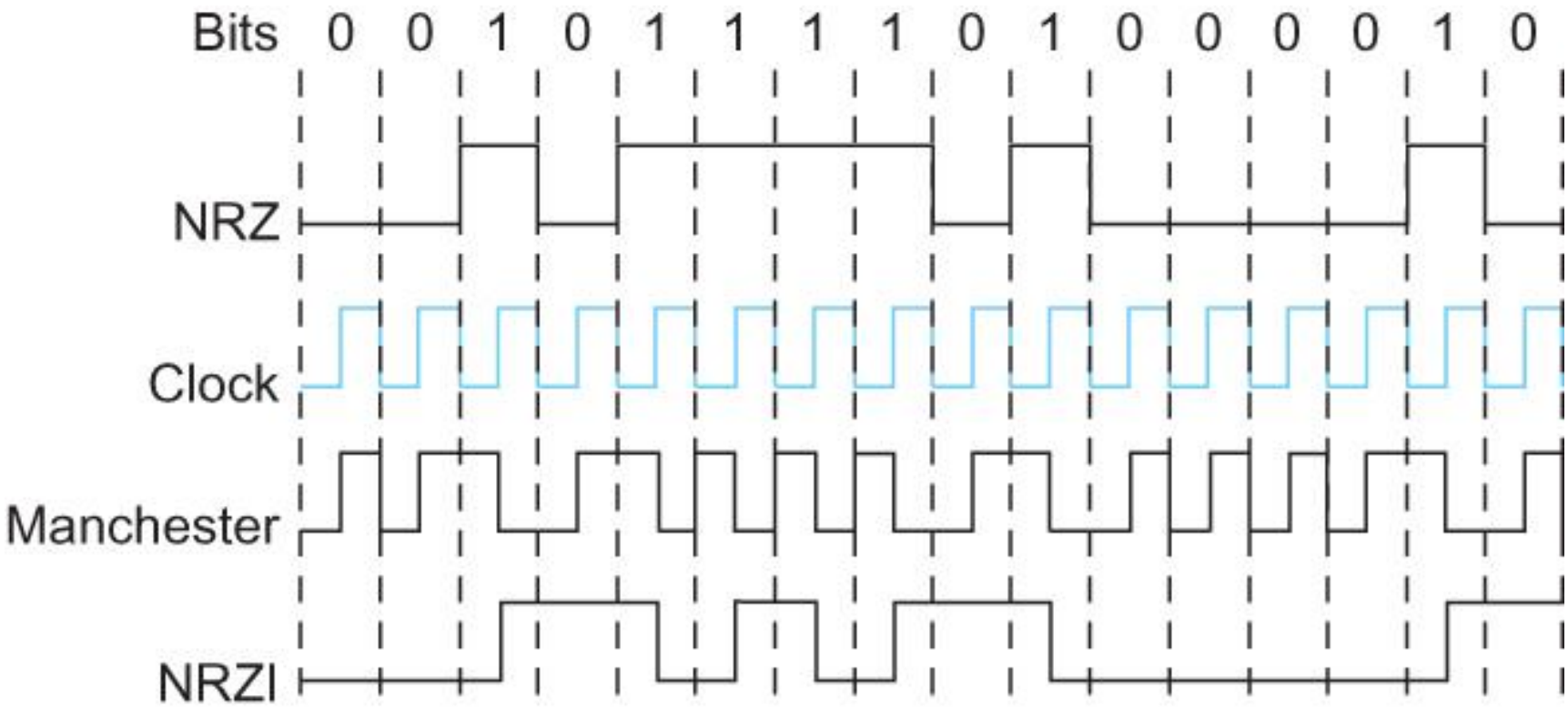
Encoding (2)

- Problem with NRZ
  - Baseline wander
    - The receiver keeps an average of the signals it has seen so far
    - Uses the average to distinguish between low and high signal
    - When a signal is significantly low than the average, it is 0, else it is 1
    - Too many consecutive 0's and 1's cause this average to change, making it difficult to detect
  - Clock recovery
    - Frequent transition from high to low or vice versa are necessary to enable clock recovery
    - Both the sending and decoding process is driven by a clock
    - Every clock cycle, the sender transmits a bit and the receiver recovers a bit
    - The sender and receiver have to be precisely synchronized

Encoding (3)

- NRZI
  - Non Return to Zero Inverted
  - Sender makes a transition from the current signal to encode 1 and stay at the current signal to encode 0
  - Solves for consecutive 1's
- Manchester encoding
  - Merging the clock with signal by transmitting Ex-OR of the NRZ encoded data and the clock
  - Clock is an internal signal that alternates from low to high, a low/high pair is considered as one clock cycle
  - In Manchester encoding
    - 0: low→ high transition
    - 1: high→ low transition

Encoding (4)



Different encoding strategies

Encoding (5)

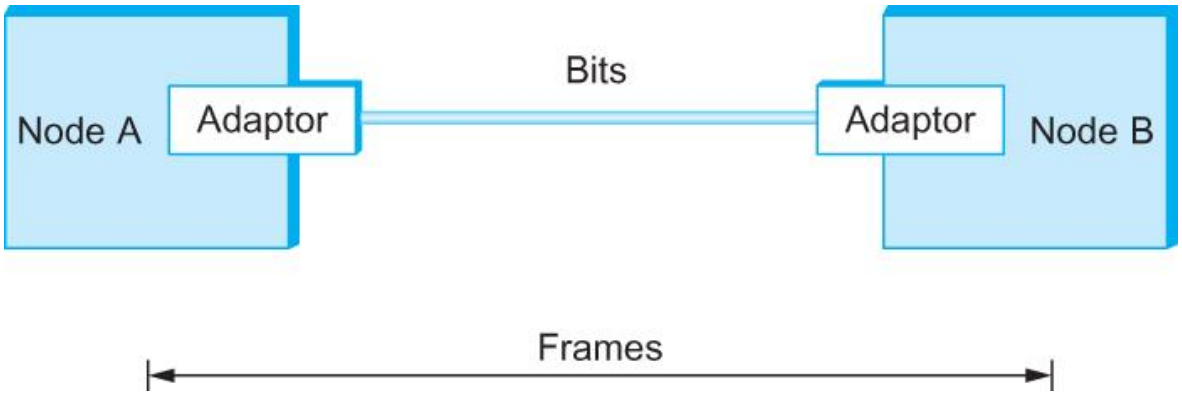
- 4B/5B encoding
  - Insert extra bits into bit stream so as to break up the long sequence of 0's and 1's
  - Every 4-bits of actual data are encoded in a 5- bit code that is transmitted to the receiver
  - 5-bit codes are selected in such a way that each one has no more than one leading 0 (zero) and no more than two tailing 0's.
  - No pair of 5-bit codes results in more than three consecutive 0's

4B	5B	4B	5B	4B	5B	4B	5B
0000	11110	0100	01010	1000	10010	1100	11010
0001	01001	0101	01011	1001	10011	1101	11011
0010	10100	0110	01110	1010	10110	1110	11100
0011	10101	0111	01111	1011	10111	1111	11101

11111 – when the line is idle; 00000 – when the line is dead; 00100 – to mean halt; 13 left : 7 invalid, 6 for various control signals

Framing (1)

- We are focusing on packet-switched networks, which means that blocks of data (called *frames* at this level), not bit streams, are exchanged between nodes
- It is the network adaptor that enables the nodes to exchange frames



Bits flow between adaptors, frames between hosts

Framing (2)

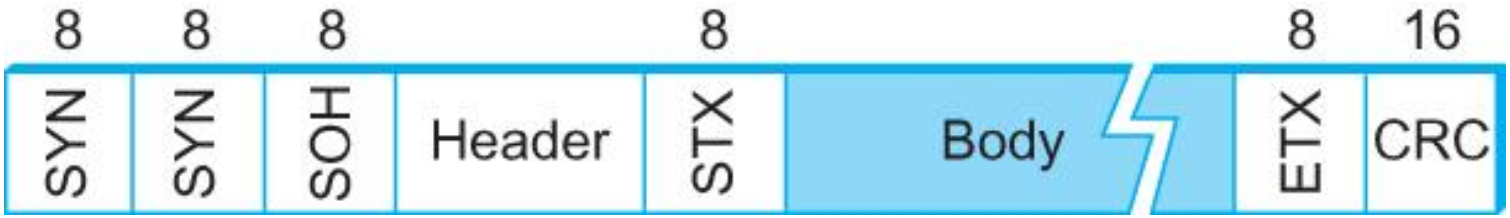
- When node A wishes to transmit a frame to node B, it tells its adaptor to transmit a frame from the node's memory
- This results in a sequence of bits being sent over the link
- The adaptor on node B then collects together the sequence of bits arriving on the link and deposits the corresponding frame in B's memory
- Recognizing exactly what set of bits constitute a frame—that is, determining where the frame begins and ends—is the central challenge faced by the adaptor

Framing (3)

- Byte-oriented Protocols: view each frame as a collection of bytes (characters) rather than bits
  - Two byte-oriented protocols: BISYNC (Binary Synchronous Communication) Protocol, which was developed by IBM (late 1960), and DDCMP (Digital Data Communication Protocol) which was used in DECNet

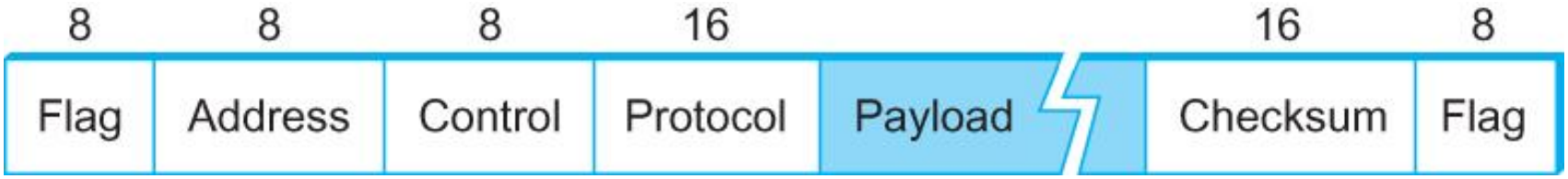
Framing (4)

- BISYNC – sentinel approach
  - Frames transmitted beginning with leftmost field
  - Beginning of a frame is denoted by sending a special SYN (synchronize) character
  - Data portion of the frame is contained between special sentinel character STX (start of text) and ETX (end of text)
  - SOH : Start of Header
  - DLE : Data Link Escape (i.e., DLE+ETX, character stuffing)
  - CRC: Cyclic Redundancy Check



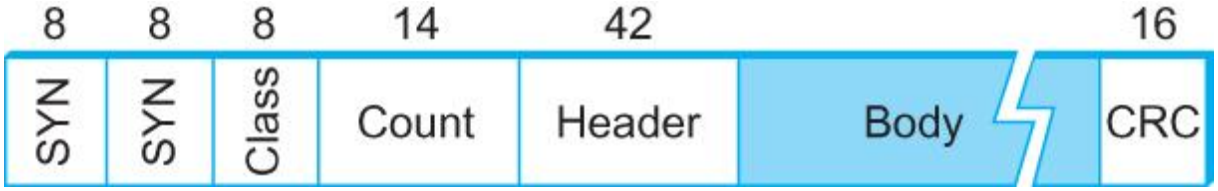
Framing (5)

- Recent PPP which is commonly run over Internet links uses sentinel approach
  - Special start of text character denoted as Flag (0 1 1 1 1 1 1 0)
  - Address, control : default numbers
  - Protocol for demux : IP/IPX
  - Payload : negotiated (1500 bytes)
  - Checksum : for error detection



Framing (6)

- DDCMP
  - Count : how many bytes are contained in the frame body
  - If Count is corrupted → framing error



DDCMP Frame Format

Note: Three classes: data, control, maintenance

Framing (7)

- Bit-oriented Protocol: HDLC (High Level Data Link Control)
  - Beginning and Ending Sequences: 0 1 1 1 1 1 1 0
  - On the sending side, any time five consecutive 1's have been transmitted from the body of the message (i.e. excluding when the sender is trying to send the distinguished 01111110 sequence), the sender inserts 0 before transmitting the next bit
  - On the receiving side
    - 5 consecutive 1's, next bit 0 is stuffed, so discard it; next bit is 1, either end of the frame marker or Error has been introduced in the bitstream
      - Look at the next bit. If 0 (01111110) → End of the frame marker; if 1 (01111111) → Error, discard the whole frame
      - The receiver needs to wait for next 01111110 before it can start receiving again

Error Detection (1)

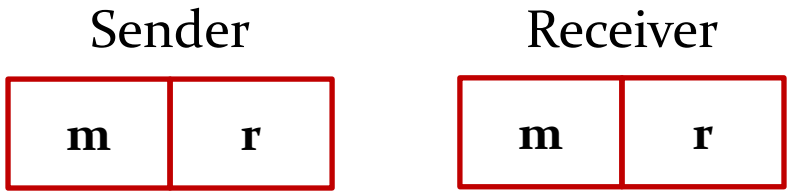
- Two approaches when the recipient detects an error
  - Notify the sender that the message was corrupted, so the sender can send again
    - If the error is rare, then the retransmitted message will be error-free
  - Using some error detection and correction algorithm, the receiver reconstructs the message
- Common technique for detecting transmission error
  - CRC (Cyclic Redundancy Check)
    - Used in HDLC, DDCMP, CSMA/CD, Token Ring
  - Other approaches
    - Two-dimensional parity (BISYNC)
    - Checksum (IP)

Error Detection (2)

- Basic idea of error detection
  - Add redundant information to a frame that can be used to determine if errors have been introduced
  - Imagine (extreme case)
    - Transmitting two complete copies of data
      - Identical → No error
      - Differ → error
      - Poor scheme ???
        - » *n*-bit message, *n*-bit redundant information
        - » Error can still go undetected
    - In general, we can provide strong error detection technique
      - *k* redundant bits, *n* bits message, $k << n$
      - In Ethernet, a frame carrying up to 12,000 bits of data requires only 32-bit CRC

Error Detection (3)

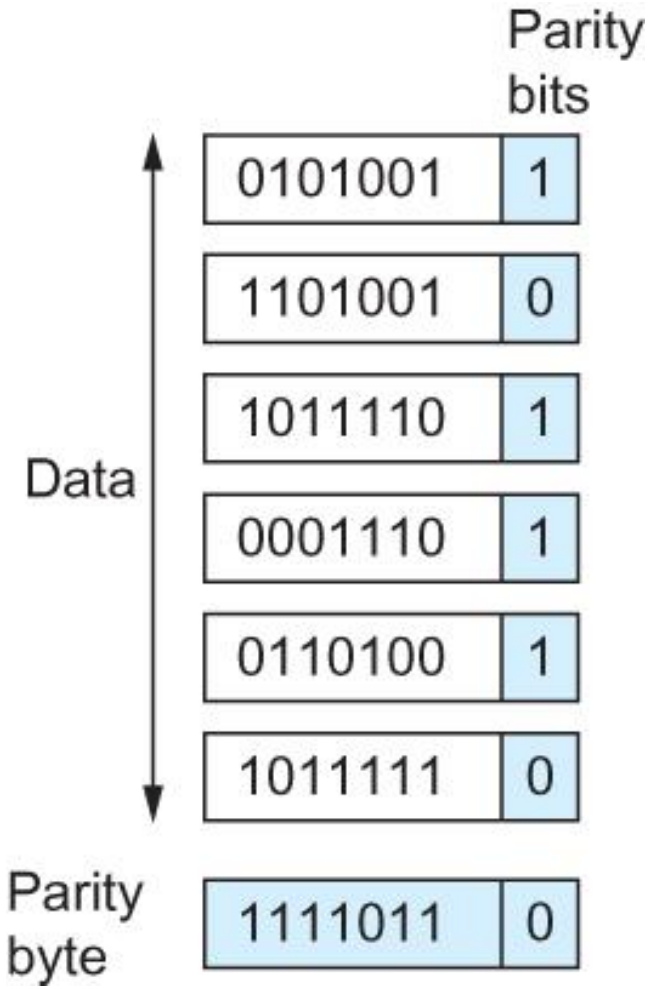
- Extra bits are redundant
  - They add no new information to the message
  - Derived from the original message using some algorithms
  - Both the sender and receiver know the algorithm
  - Receiver computes *r* using *m*. If they match, no error



Two-Dimensional Parity (1)

- Two-dimensional parity is exactly what the name suggests
- It is based on "simple" (one-dimensional) parity, which usually involves adding one extra bit to a 7-bit code to balance the number of 1s in the byte
- For example, odd parity sets the eighth bit to 1 if needed to give an odd number of 1s in the byte, and even parity sets the eighth bit to 1 if needed to give an even number of 1s in the byte
- Two-dimensional parity does a similar calculation for each bit position across each of the bytes contained in the frame
- This results in an extra parity byte for the entire frame, in addition to a parity bit for each byte
- Two-dimensional parity catches all 1-, 2-, and 3-bit errors and most 4-bit errors

Two-Dimensional Parity (2)



Two Dimensional Parity

Internet Checksum Algorithm (1)

- Not used at the link level
- Add up all the words that are transmitted and then transmit the result of that sum. The result is called the checksum
- The receiver performs the same calculation on the received data and compares the result with the received checksum
- If any transmitted data, including the checksum itself, is corrupted, then the results will not match, so the receiver knows that an error occurred
- Consider the data being checksummed as a sequence of 16-bit integers, add them together using 16-bit ones complement arithmetic (explained next slide) and then take the ones complement of the result. That 16-bit number is the checksum

Internet Checksum Algorithm (2)

- In ones complement arithmetic, a negative integer –x is represented as the complement of x; so each bit of x is inverted
- When adding numbers in ones complement arithmetic, a carryout from the most significant bit needs to be added to the result
- Consider, for example, the addition of –5 and –3 in ones complement arithmetic on 4-bit integers
  - +5 is 0101, so –5 is 1010; +3 is 0011, so –3 is 1100
- If we add 1010 and 1100 ignoring the carry, we get 0110
- In ones complement arithmetic, the fact that this operation caused a carry from the most significant bit causes us to increment the result, giving 0111, which is the ones complement representation of –8 (obtained by inverting the bits in 1000), as we would expect

Cyclic Redundancy Check (CRC) (1)

- Reduce the number of extra bits and maximize protection
- Given a bit string 110001 we can associate a polynomial on a single variable *x* for it
  - $1 \times x^5 + 1 \times x^4 + 0 \times x^3 + 0 \times x^2 + 0 \times x^1 + 1 \times x^0 = x^5 + x^4 + 1$ and the degree is 5
  - A *k*-bit frame has a maximum degree of *k*-1
- Let M(x) be a message polynomial and C(x) be a generator polynomial
- Let M(x)/C(x) leave a remainder of 0
- When M(x) is sent and M'(x) is received we have M'(x) = M(x)+E(x)
- The receiver computes M'(x)/C(x) and if the remainder is nonzero, then an error has occurred
- The only thing the sender and the receiver should know is C(x)

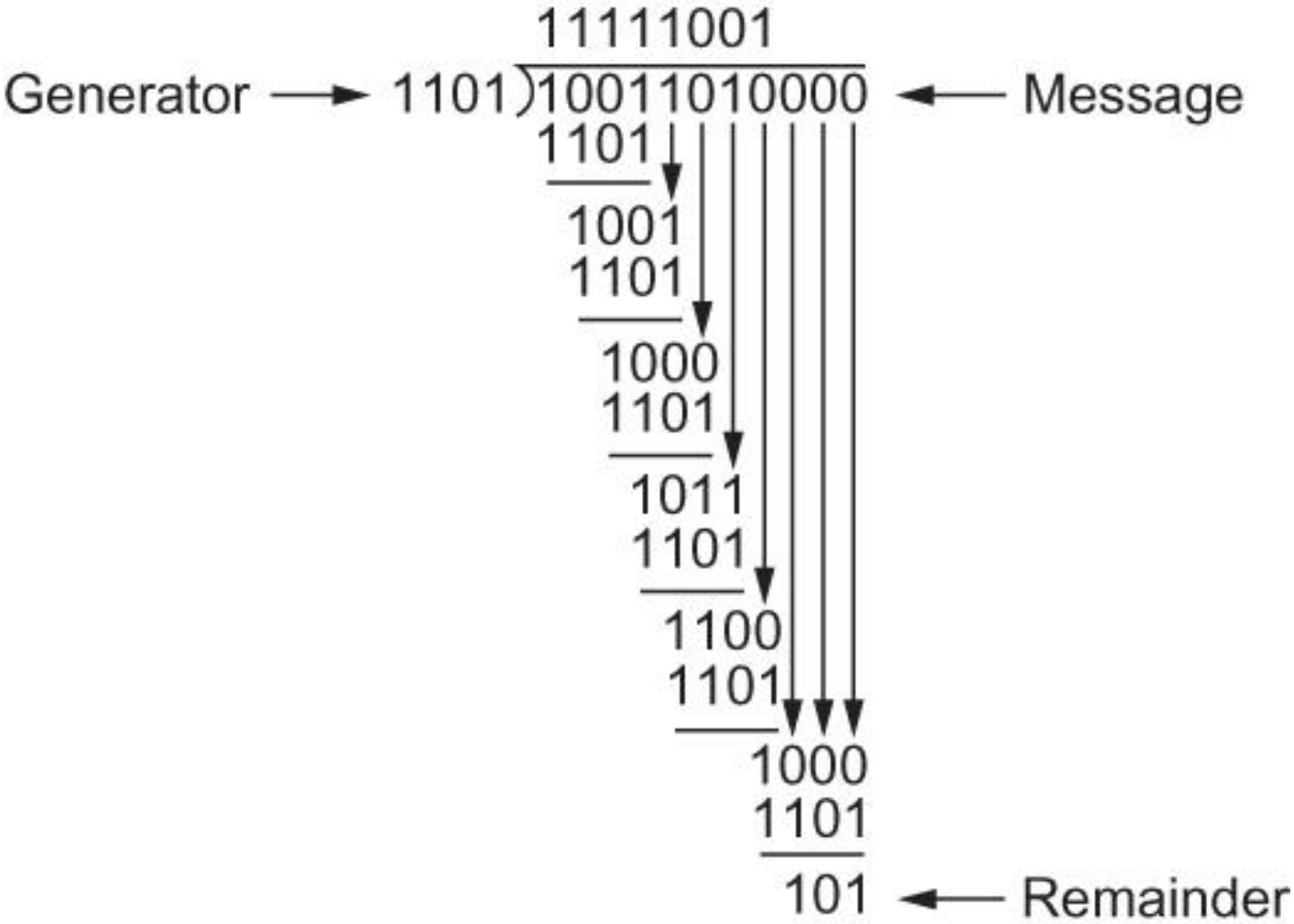
Cyclic Redundancy Check (CRC) (2)

- Polynomial Arithmetic Modulo 2
  - Any polynomial B(x) can be divided by a divisor polynomial C(x) if B(x) is of higher degree than C(x)
  - Any polynomial B(x) can be divided once by a divisor polynomial C(x) if B(x) is of the same degree as C(x)
  - The remainder obtained when B(x) is divided by C(x) is obtained by subtracting C(x) from B(x)
  - To subtract C(x) from B(x), we simply perform the exclusive-OR (XOR) operation on each pair of matching coefficients
- Let M(x) be a frame with *m* bits and let the generator polynomial have less than *m* bits say equal to *r*
- Let *r* be the degree of C(x). Append *r* zero bits to the low-order end of the frame, so it now contains *m*+*r* bits and corresponds to the polynomial $x^r M(x)$

Cyclic Redundancy Check (CRC) (3)

- Divide the bit string corresponding to $x^r M(x)$ by the bit string corresponding to C(x) using modulo 2 division
- Subtract the remainder (which is always *r* or fewer bits) from the string corresponding to $x^r M(x)$ using modulo 2 subtraction (addition and subtraction are the same in modulo 2)
- The result is the checksummed frame to be transmitted. Call it polynomial M'(x)

Cyclic Redundancy Check (CRC) (4)



CRC Calculation Using Polynomial Long Division

Cyclic Redundancy Check (CRC) (5)

- Six generator polynomials that have become international standards are:
  - CRC-8 = x^8+x^2+x+1
  - CRC-10 = $x^{10}+x^9+x^5+x^4+x+1$
  - CRC-12 = $x^{12}+x^{11}+x^3+x^2+x+1$
  - CRC-16 = $x^{16}+x^{15}+x^2+1$
  - CRC-CCITT = $x^{16}+x^{12}+x^5+1$
  - CRC-32 = $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$

Reliable Transmission (1)

- CRC is used to detect errors
- Some error codes are strong enough to correct errors
- The overhead is typically too high
- Corrupt frames must be discarded
- A link-level protocol that wants to deliver frames reliably must recover from these discarded frames
- This is accomplished using a combination of two fundamental mechanisms: acknowledgements and timeouts

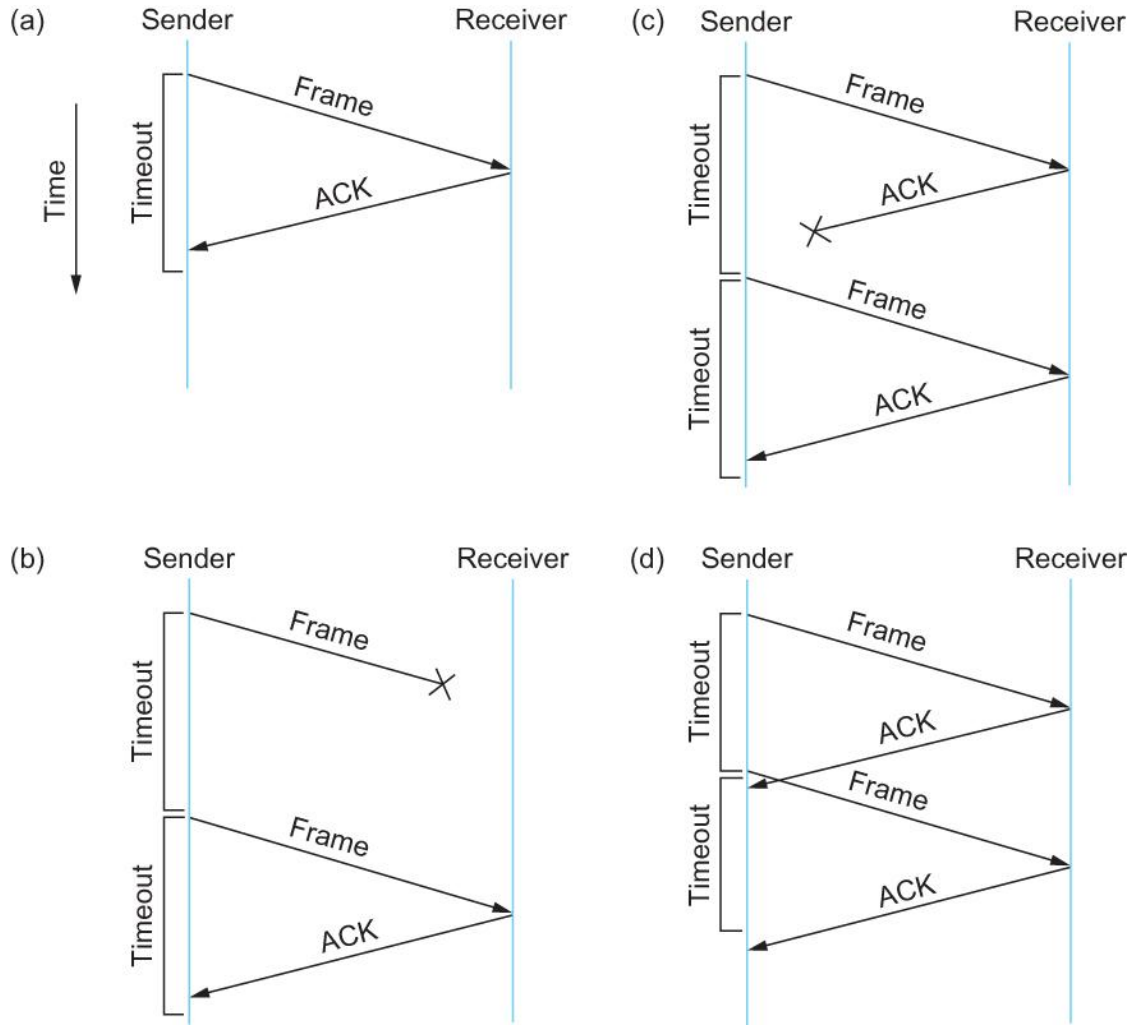
Reliable Transmission (2)

- An acknowledgement (ACK for short) is a small control frame that a protocol sends back to its peer saying that it has received the earlier frame
- The receipt of an acknowledgement indicates to the sender of the original frame that its frame was successfully delivered
- If the sender does not receive an acknowledgment after a reasonable amount of time, then it retransmits the original frame
- The action of waiting a reasonable amount of time is called a timeout
- The general strategy of using acknowledgements and timeouts to implement reliable delivery is sometimes called Automatic Repeat reQuest (ARQ)

Stop and Wait Protocol (1)

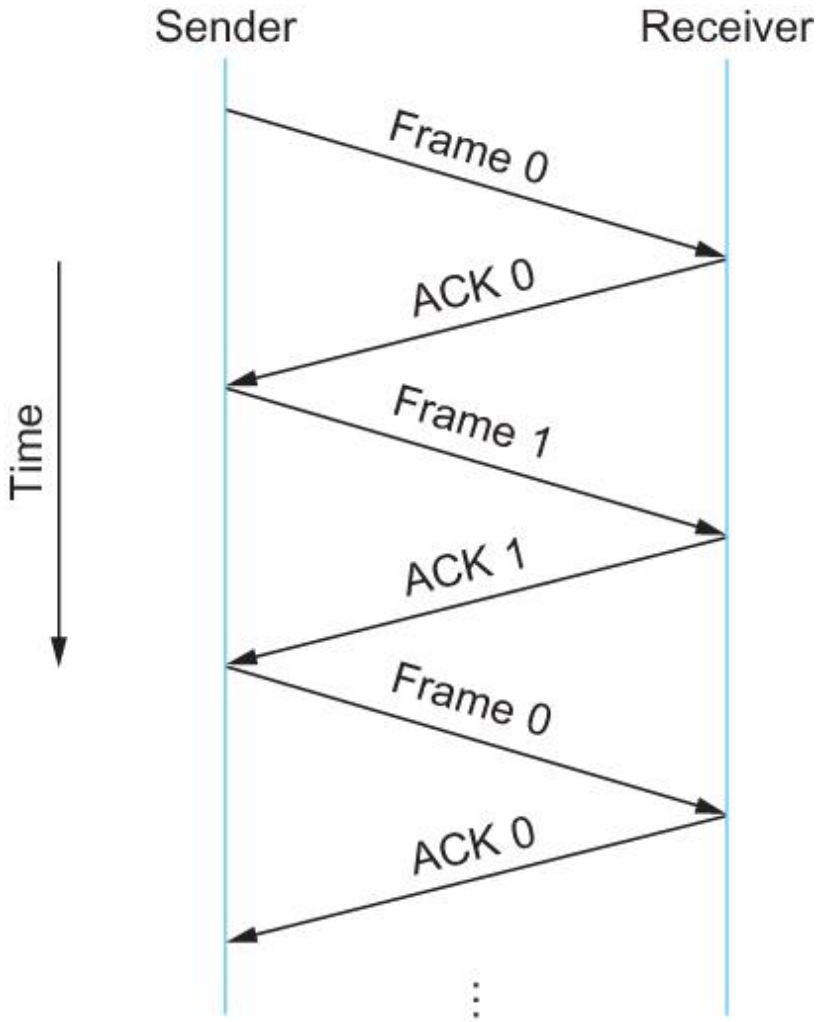
- After transmitting one frame, the sender waits for an acknowledgement before transmitting the next frame
- If the acknowledgement does not arrive after a certain period of time, the sender times out and retransmits the original frame
- If the acknowledgment is lost or delayed in arriving
  - The sender times out and retransmits the original frame, but the receiver will think that it is the next frame since it has correctly received and acknowledged the first frame
  - As a result, duplicate copies of frames will be delivered
- How to solve this?
  - Use 1 bit sequence number (0 or 1)
  - When the sender retransmits frame 0, the receiver can determine that it is seeing a second copy of frame 0 rather than the first copy of frame 1 and therefore can ignore it (the receiver still acknowledges it, in case the first acknowledgement was lost)

Stop and Wait Protocol (2)



Timeline showing four different scenarios for the stop-and-wait algorithm.
(a) The ACK is received before the timer expires; (b) the original frame is lost; (c) the ACK is lost; (d) the timeout fires too soon

Stop and Wait Protocol (3)



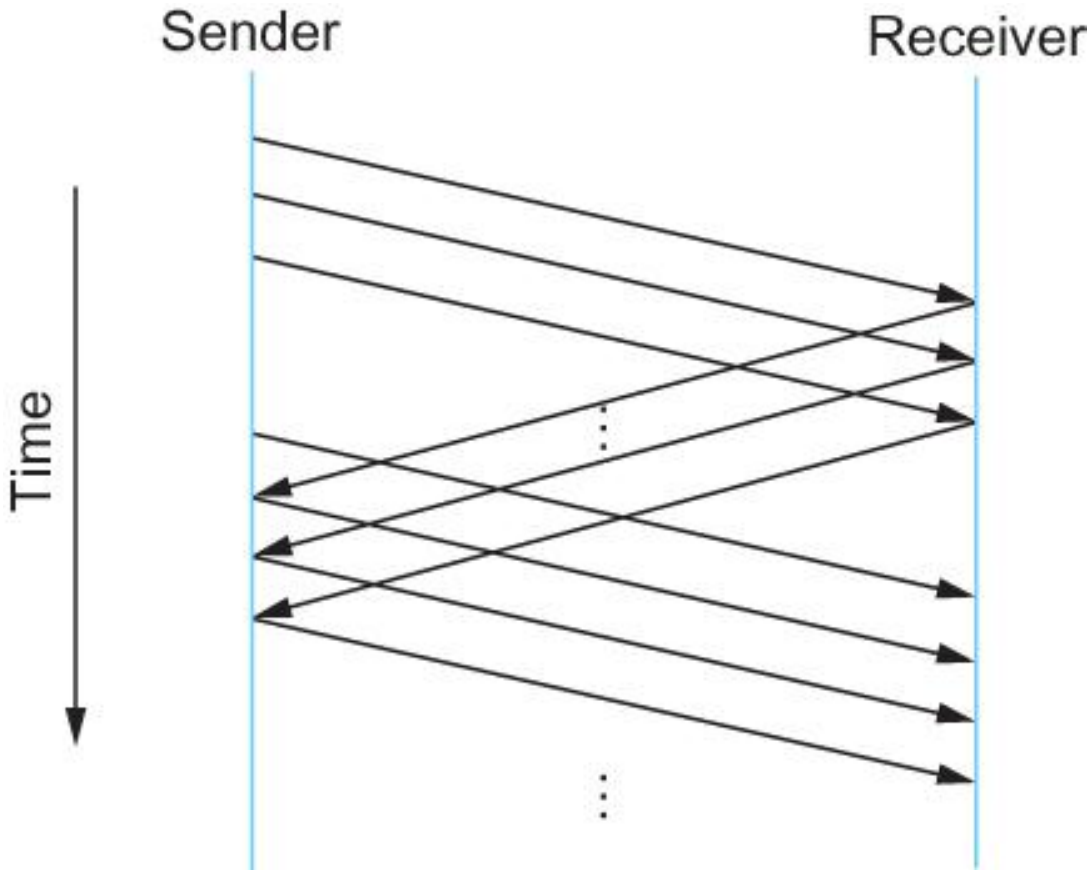
Timeline for stop-and-wait with 1-bit sequence number

Stop and Wait Protocol (4)

- The sender has only one outstanding frame on the link at a time
  - This may be far below the link's capacity
- Consider a 1.5 Mbps link with a 45 ms RTT
  - The link has a delay × bandwidth product of 67.5 Kb or approximately 8 KB
  - Since the sender can send only one frame per RTT and assuming a frame size of 1 KB
  - Maximum Sending rate
    - Bits per frame ÷ Time per frame = 1024 × 8 ÷ 0.045 = 182 Kbps

      Or about one-eighth of the link's capacity
  - To use the link fully, then sender should transmit up to eight frames before having to wait for an acknowledgement

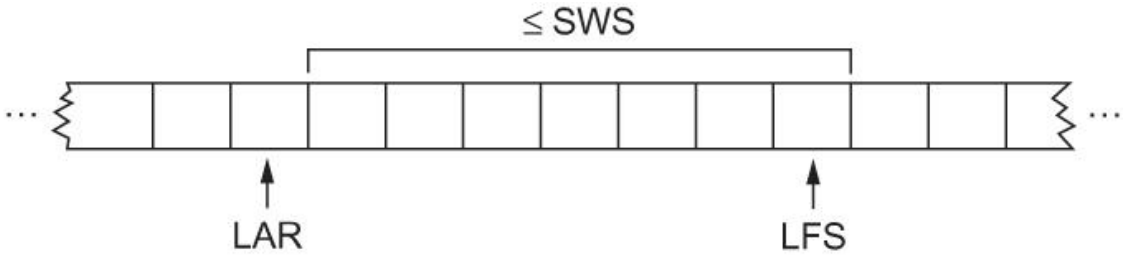
Sliding Window Protocol (1)



Timeline for Sliding Window Protocol

Sliding Window Protocol (2)

- Sender assigns a sequence number denoted as **SeqNum** to each frame
- Sender maintains three variables
  - Sending Window Size (SWS): upper bound on the number of outstanding (unacknowledged) frames that the sender can transmit
  - Last Acknowledgement Received (LAR): sequence number of the last acknowledgement received
  - Last Frame Sent (LFS): sequence number of the last frame sent
- Sender also maintains the following invariant: LFS – LAR ≤ SWS



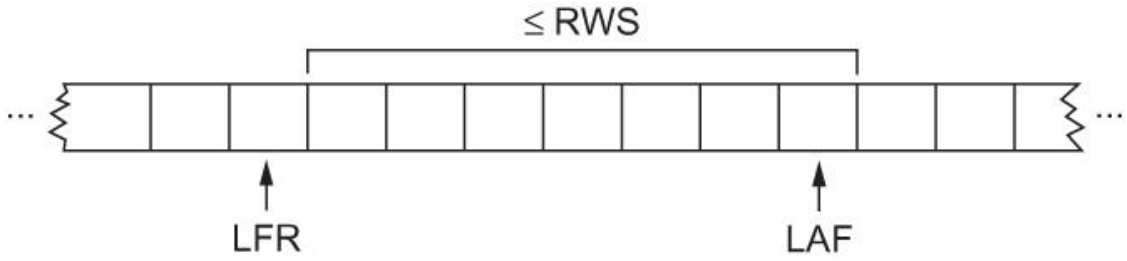
Sliding Window on Sender

Sliding Window Protocol (3)

- When an acknowledgement arrives, the sender moves LAR to right, thereby allowing the sender to transmit another frame
- Also the sender associates a timer with each frame it transmits, it retransmits the frame if the timer expires before the ACK is received

Sliding Window Protocol (4)

- Receiver maintains three variables
  - Receiving Window Size (RWS)
    - Upper bound on the number of out-of-order frames that the receiver is willing to accept
  - Largest Acceptable Frame (LAF)
    - Sequence number of the largest acceptable frame
  - Last Frame Received (LFR)
    - Sequence number of the last frame received
- Receiver also maintains the following invariant: LAF – LFR ≤ RWS



Sliding Window on Receiver

Sliding Window Protocol (5)

- When a frame with sequence number SeqNum arrives, what does the receiver do?
  - If SeqNum ≤ LFR or SeqNum > LAF: discard it (the frame is outside the receiver window)
  - If LFR < SeqNum ≤ LAF: accept it; now the receiver needs to decide whether or not to send an ACK
    - Let **SeqNumToAck** denote the largest sequence number not yet acknowledged, such that all frames with sequence number less than or equal to SeqNumToAck have been received
    - The receiver acknowledges the receipt of SeqNumToAck even if high-numbered packets have been received. This acknowledgement is said to be cumulative
    - The receiver then sets LFR = SeqNumToAck and adjusts LAF = LFR + RWS

Sliding Window Protocol (6)

- For example, LFR = 5 and RWS = 4 $\Rightarrow$ LAF = 9
- If frames 7 and 8 arrive, they will be buffered because they are within the receiver window
- But no ACK will be sent since frame 6 is yet to arrive and frames 7 and 8 are out of order
- When frame 6 arrives (it is late because it was lost first time and had to be retransmitted), the Receiver acknowledges frame 8, and bumps LFR to 8 and LAF to 12

Issues with Sliding Window Protocol (1)

- When timeout occurs, the amount of data in transit decreases, since the sender is unable to advance its window
- When the packet loss occurs, this scheme is no longer keeping the pipe full, the longer it takes to notice that a packet loss has occurred, the more severe the problem becomes
- How to improve this?
  - Negative Acknowledgement (NAK)
  - Additional Acknowledgement
  - Selective Acknowledgement

Issues with Sliding Window Protocol (2)

- Negative Acknowledgement (NAK)
  - Receiver sends NAK for frame 6 when frame 7 arrive (in the previous example). However, this is unnecessary since sender's timeout mechanism will be sufficient to catch the situation
- Additional Acknowledgement
  - Receiver sends additional ACK for frame 5 when frame 7 arrives. Sender uses duplicate ACK as a clue for frame loss
- Selective Acknowledgement
  - Receiver will acknowledge exactly those frames it has received, rather than the highest number frames
    - Receiver will acknowledge frames 7 and 8
    - Sender knows frame 6 is lost
    - Sender can keep the pipe full (additional complexity)

Issues with Sliding Window Protocol (3)

- How to select the window size
  - SWS is easy to compute: Delay × Bandwidth
  - RWS can be anything
    - Two common setting
      - RWS = 1: no buffer at the receiver for frames that arrive out of order
      - RWS = SWS: the receiver can buffer frames that the sender transmits
    - It does not make any sense to keep RWS > SWS
- Finite Sequence Number: frame sequence number is specified in the header field
  - Finite size: 3-bit means eight possible sequence number, i.e., 0, 1, 2, 3, 4, 5, 6, 7
  - It is necessary to wrap around

Issues with Sliding Window Protocol (4)

- How to distinguish between different incarnations of the same sequence number?
  - Number of possible sequence number must be larger than the number of outstanding frames allowed
    - Stop and Wait: 1 outstanding frame, and thus 2 distinct sequence number (0 and 1)
    - Let MaxSeqNum be the number of available sequence numbers, then is SWS + 1 ≤ MaxSeqNum sufficient?
      - It depends on RWS
      - If RWS = 1, then it's sufficient
      - If RWS = SWS, then it's not good enough

Issues with Sliding Window Protocol (5)

- Why RWS = SWS is not good enough?
  - For example, we have eight sequence numbers 0, 1, 2, 3, 4, 5, 6, 7; RWS = SWS = 7
    - Sender sends 0, 1, …, 6
    - Receiver receives 0, 1, … ,6
    - Receiver acknowledges 0, 1, …, 6
    - ACK (0, 1, …, 6) are lost
    - Sender retransmits 0, 1, …, 6
    - Receiver is expecting 7, 0, …., 5
  - To avoid this, if RWS = SWS, then SWS < (MaxSeqNum + 1)/2

Issues with Sliding Window Protocol (6)

- Serves three different roles
  - Reliable transmission
  - Preserve the order
    - Each frame has a sequence number
    - The receiver makes sure that it does not pass a frame up to the next higher-level protocol until it has already passed up all frames with a smaller sequence number
  - Frame control
    - Receiver is able to throttle the sender to keep the sender from overrunning the receiver by means of transmitting more data than the receiver is able to process

Ethernet (1)

- Most successful local area networking technology of last 20 years.
- Developed in the mid-1970s by researchers at the Xerox Palo Alto Research Centers (PARC)
- CSMA/CD (Carrier Sense Multiple Access with Collision Detection) operations
  - A set of nodes send and receive frames over a shared link
  - Carrier sense means that all nodes can distinguish between an idle and a busy link
  - Collision detection means that a node listens as it transmits and can therefore detect when a frame it is transmitting has collided with a frame transmitted by another node

Ethernet (2)

- ALOHA (packet radio network) is the root protocol
  - Developed at the University of Hawaii to support communication across the Hawaiian Islands
  - For ALOHA the medium was atmosphere, for Ethernet the medium is a coax cable
- DEC and Intel joined Xerox to define a 10-Mbps Ethernet standard in 1978
- This standard formed the basis for IEEE standard 802.3
- More recently 802.3 has been extended to include a 100-Mbps version called Fast Ethernet and a 1000-Mbps version called Gigabit Ethernet

Ethernet (3)

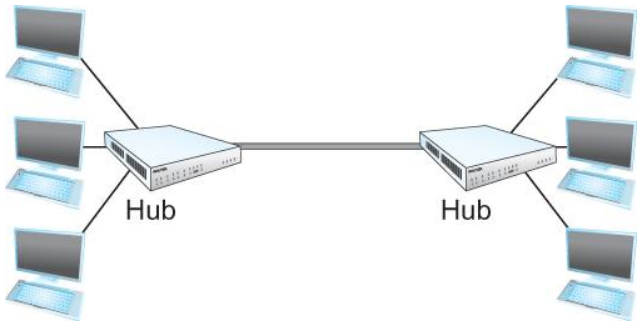
- An Ethernet segment is implemented on a coaxial cable of up to 500 meters
  - This cable is similar to the type used for cable TV except that it typically has an impedance of 50 ohms instead of cable TV's 75 ohms
- Hosts connect to an Ethernet segment by tapping into it
- A transceiver (a small device directly attached to the tap) detects when the line is idle and drives signal when the host is transmitting
- The transceiver also receives incoming signal
- The transceiver is connected to an Ethernet adaptor which is plugged into the host
- The protocol is implemented on the adaptor

Ethernet (4)

- Multiple Ethernet segments can be joined together by *repeaters*
- A *repeater* is a device that forwards digital signals
- No more than four repeaters may be positioned between any pair of hosts. As a result, an Ethernet has a total reach of only 2500 meters
- Any signal placed on the Ethernet by a host is broadcast over the entire network
  - Signal is propagated in both directions
  - Repeaters forward the signal on all outgoing segments
  - Terminators attached to the end of each segment absorb the signal
- Ethernet uses Manchester encoding scheme

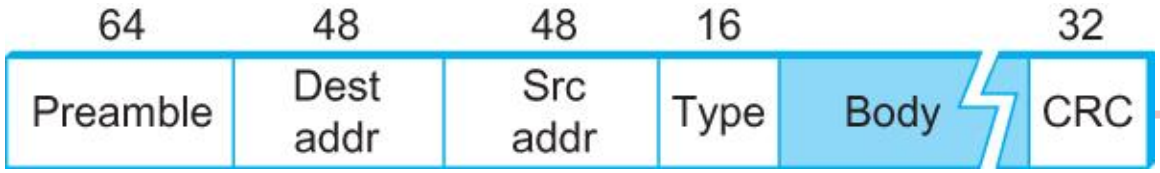
Ethernet (5)

- New technologies in Ethernet
  - Instead of using coax cable, an Ethernet can be constructed from a thinner cable known as 10Base2 (the original was 10Base5)
    - 10 means the network operates at 10 Mbps
    - Base means the cable is used in a baseband system
    - 2 means that a given segment can be no longer than 200 meters
  - Another cable technology is 10BaseT
    - T stands for twisted pair
    - Limited to 100 meters in length
    - With 10BaseT, the common configuration is to have several point to point segments coming out of a multiway repeater, called *Hub*



Access Protocol for Ethernet

- The algorithm is commonly called Ethernet's Media Access Control (MAC), which is implemented in hardware on the network adaptor
- Frame format
  - Preamble (64bit): allows the receiver to synchronize with the signal (sequence of alternating 0s and 1s)
  - Host and Destination Address (48bit each)
  - Packet type (16bit): acts as demux key to identify the higher level protocol
  - Data (up to 1500 bytes)
    - Minimally a frame must contain at least 46 bytes of data
    - Frame must be long enough to detect collision
  - CRC (32bit)

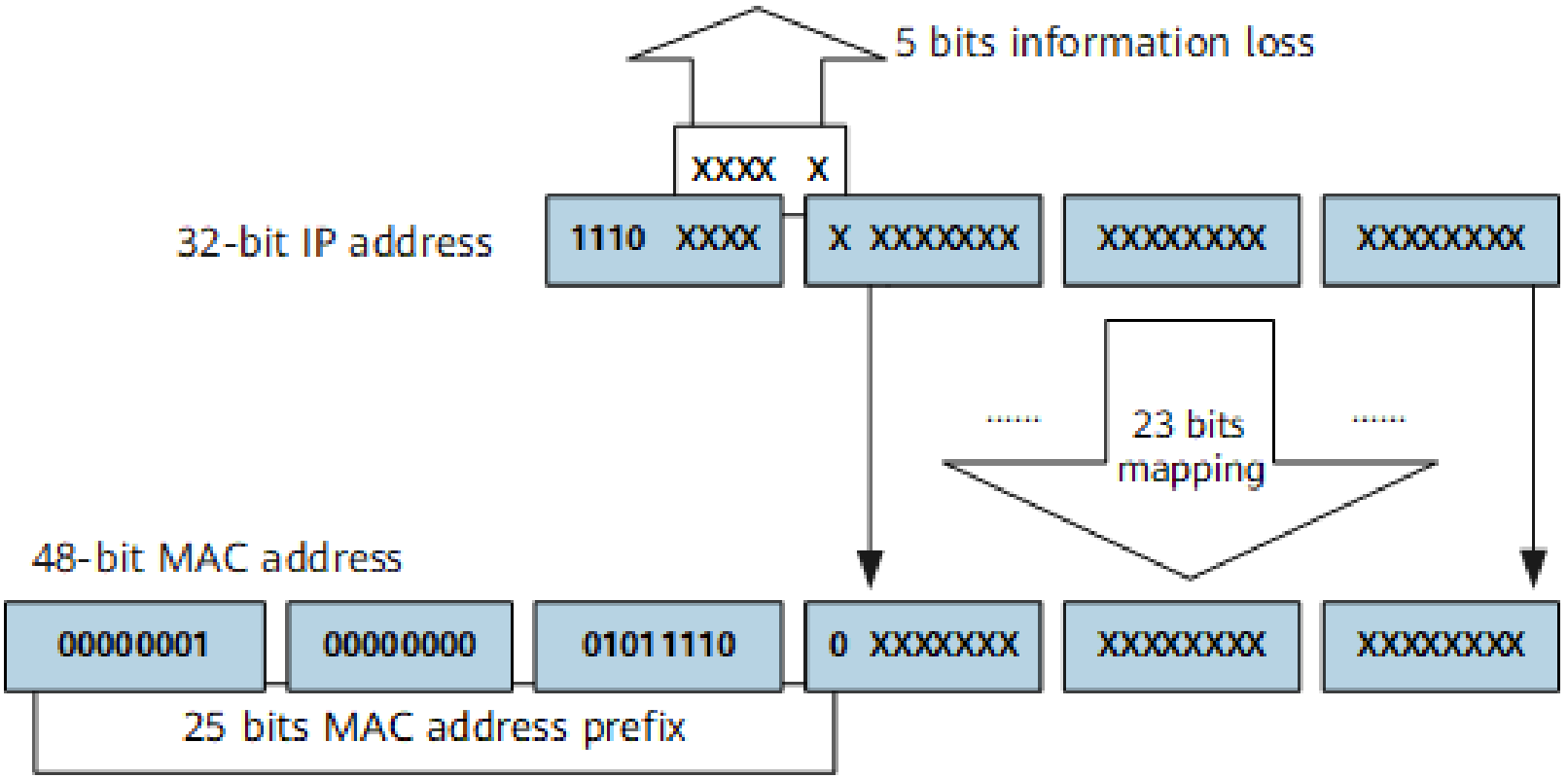


Ethernet Addresses (1)

- Each host on an Ethernet (in fact, every Ethernet host in the world) has a unique Ethernet Address
- The address belongs to the adaptor, not the host. It is usually burnt into ROM
- Ethernet addresses are typically printed in a human readable format
  - As a sequence of six numbers separated by colons
  - Each number corresponds to 1 byte of the 6 byte address and is given by a pair of hexadecimal digits, one for each of the 4-bit nibbles in the byte
  - Leading 0s are dropped
  - For example, 8:0:2b:e4:b1:2 is 00001000 00000000 00101011 11100100 10110001 00000010

Ethernet Addresses (2)

- To ensure that every adaptor gets a unique address, each manufacturer of Ethernet devices is allocated a different prefix that must be prepended to the address on every adaptor they build
- Each frame transmitted on an Ethernet is received by every adaptor connected to that Ethernet
- Each adaptor recognizes those frames addressed to its address and passes only those frames on to the host
- In addition, to *unicast* address, an Ethernet address consisting of all 1s is treated as a *broadcast* address
  - All adaptors pass frames addressed to the *broadcast* address up to the host
- Similarly, an address that has the first bit set to 1 but is not the *broadcast* address is called a *multicast* address
  - A given host can program its adaptor to accept some set of *multicast* addresses



MAC-IP multicast address mapping

Ethernet Transmitter Algorithm (1)

- When the adaptor has a frame to send and the line is idle, it transmits the frame immediately
  - The upper bound of 1500 bytes in the message means that the adaptor can occupy the line for a fixed length of time
- When the adaptor has a frame to send and the line is busy, it waits for the line to go idle and then transmits immediately
- The Ethernet is said to be 1-persistent protocol because an adaptor with a frame to send transmits with probability 1 whenever a busy line goes idle

Ethernet Transmitter Algorithm (2)

- Since there is no centralized control, and thus it is possible for two (or more) adaptors to begin transmitting at the same time when either because both found the line to be idle or, both had been waiting for a busy line to become idle
- When this happens, the two (or more) frames are said to be *collide* on the network
- Since Ethernet supports collision detection, each sender is able to determine that a collision is in progress
- At the moment an adaptor detects that its frame is colliding with another, it first makes sure to transmit a 32-bit jamming sequence and then stops transmission
  - Thus, a transmitter will minimally send 96 bits in the case of collision (64-bit preamble + 32-bit jamming sequence)

Ethernet Transmitter Algorithm (3)

- Once an adaptor has detected a collision, and stopped its transmission, it waits a certain amount of time and tries again
- Each time the adaptor tries to transmit but fails, it doubles the amount of time it waits before trying again
- This strategy of doubling the delay interval between each retransmission attempt is known as *Exponential Backoff*
- The adaptor first delays either 0 or 1, selected at random
- If this effort fails, it then waits 0, 1, 2, 3 (selected randomly) before trying again
- After the third collision, it waits k slots for $k = 0 \ldots 2^3 - 1$ (again selected at random)
- In general, the algorithm randomly selects a k between 0 and $2^n - 1$ and waits for k slots, where n is the number of collisions experienced so far