one in sequ
ments for greater bandwi
col, and the latter strategy is a selective rep

The Internet uses PPP as the main data link protocol over point-to-point lines. It provides a connectionless unacknowledged service, using flag bytes to delimit frames and a CRC for error detection. It is used to carry packets across a range of links, including SONET links in wide area networks and ADSL links for the home. DOCSIS is used when Internet service is provided over the existing cable TV network.

PROBLEMS

1. Ethernet uses a preamble in combination with a byte count to separate the frames. What happens if a user tries to send data that contains this preamble?
2. The following data fragment occurs in the middle of a data stream for which the byte-stuffing algorithm described in the text is used: A B ESC C ESC FLAG FLAG D. What is the output after stuffing?
3. What is the maximum overhead in byte-stuffing algorithm?
4. You receive the following data fragment: 0110 0111 1100 1111 0111 1101. You know that the protocol uses bit stuffing. Show the data after destuffing.
5. When bit stuffing is used, is it possible for the loss, insertion, or modification of a single bit to cause an error not detected by the checksum? If not, why not? If so, how? Does the checksum length play a role here?
6. An upper-layer packet is split into 10 frames, each of which has an 80% chance of arriving undamaged. If no error control is done by the data link protocol, how many times must the message be sent on average to get the entire thing through?
7. Can you think of any circumstances under which an open-loop protocol (e.g., a Hamming code) might be preferable to the feedback-type protocols discussed throughout this chapter?
8. To provide more reliability than a single parity bit can give, an error-detecting coding scheme uses one parity bit for checking all the odd-numbered bits and a second parity bit for all the even-numbered bits. What is the Hamming distance of this code?
9. After noticing that the instant messaging service that you use daily does not provide error detection, you decide to adopt a simple error detection mechanism yourself: you send all your messages twice. What is the corresponding Hamming distance and code rate? How does this compare to adding a parity bit?
10. Consider an error detection mechanism that sends every message twice. Assuming that exactly two single-bit errors occur, what is the probability that the errors will pass

undetected? What is the probability when using a parity bit? Which method detects more errors?

11. An 8-bit byte with binary value 10101111 is to be encoded using an even-parity Hamming code. What is the binary value after encoding?

12. An 8-bit byte with binary value 1001 1010 is to be encoded using an *odd*-parity Hamming code. What is the binary value after encoding?

13. A 12-bit *odd*-parity Hamming code whose hexadecimal value is 0xB4D arrives at a receiver. What was the original value in hexadecimal? Assume that not more than 1 bit is in error.

14. Hamming codes have a distance of three and can be used to correct a single error or detect a double error. Can they be used to do both at the same time? Explain why or why not. In general, if the Hamming distance is n, how many errors can be corrected? How many errors can be detected?

15. Consider a protocol that for every 16 bytes of message data adds 1 byte of redundant data. Can this protocol use a Hamming code to correct single errors?

16. One way of detecting errors is to transmit data as a block of n rows of k bits per row and add parity bits to each row and each column. The bit in the lower-right corner is a parity bit that checks its row and its column. Will this scheme detect all single errors? Double errors? Triple errors? Show that this scheme cannot detect some four-bit errors.

17. In the previous problem, how many errors can be detected and corrected?

18. Give a formula for the lower limit on the number of redundant bits r, that need to be added to a message m, to correct all single *and double* errors.

19. Given the answer to the previous question, explain the popularity of complex probabilisitic error-correction mechanisms, such as convolutional codes and the Low-Density Parity Check, discussed in this chapter.

20. Suppose that data are transmitted in blocks of 1000 bits. What is the maximum error rate under which error detection and retransmission mechanism (1 parity bit per block) is better than using Hamming code? Assume that bit errors are independent of one another and no bit error occurs during retransmission.

21. A block of bits with n rows and k columns uses horizontal and vertical parity bits for error detection. Suppose that exactly 4 bits are inverted due to transmission errors. Derive an expression for the probability that the error will be undetected.

22. Suppose that a message 1001 1100 1010 0011 is transmitted using the Internet Checksum (4-bit word). What is the value of the checksum?

23. What is the remainder obtained by dividing $x^7 + x^5 + 1$ by the generator polynomial $x^3 + 1$?

24. A bit stream 10011101 is transmitted using the standard CRC method described in the text. The generator polynomial is $x^3 + 1$. Show the actual bit string transmitted. Suppose that the third bit from the left is inverted during transmission. Show that this error

is detected at the receiver's end. Give an example of bit errors in the bit string transmitted that will not be detected by the receiver.

25. A bit stream 11100110 is transmitted using the standard CRC method described in the text. The generator polynomial is $x^4 + x^3 + 1$. Show the actual bit string transmitted. Suppose that the third bit from the left is inverted during transmission. Show that this error is detected at the receiver's end. Give an example of bit errors in the bit string transmitted that will not be detected by the receiver.

26. Data link protocols always put the CRC in a trailer rather than in a header. Why?

27. In the discussion of ARQ protocol in Section 3.3.3, a scenario was outlined that resulted in the receiver accepting two copies of the same frame due to a loss of acknowledgement frame. Is it possible that a receiver may accept multiple copies of the same frame when none of the frames (message or acknowledgement) are lost?

28. A channel has a bit rate of 4 kbps and a propagation delay of 20 msec. For what range of frame sizes does stop-and-wait give an efficiency of at least 50%?

29. Two protocols, A and B, only differ in their sending window sizes. Protocol A uses a sending window of 20 frames. Protocol B is a stop-and-wait protocol. The two protocols run on two identical channels. If Protocol A achieves almost 100% bandwidth efficiency, what is the bandwidth efficiency of Protocol B?

30. A stop-and-wait protocol achieves 25% bandwidth efficiency using 900-bit frames over a channel with a one-way propagation delay of 50 msec. What is the bandwidth of this channel in bits per second?

31. A stop-and-wait protocol achieves 60% bandwidth efficiency using 300-bit frames over a channel with a bandwidth of 50 kbps. What is the one-way propagation delay of this channel?

32. A stop-and-wait protocol that uses 800-bit frames runs on a channel with a one-way propagation delay of 8 msec and a bandwidth of 1200 kbps. What is the bandwidth efficiency this protocol achieves on this channel?

33. A sliding window protocol uses 1000-bit frames and a fixed sending window size of 3. It achieves almost 100% bandwidth efficiency on a 250 kbps channel. The same protocol is used on an upgraded channel that has the same delay, but double the bandwidth. What is the protocol's bandwidth efficiency on the new channel?

34. In protocol 3, is it possible for the sender to start the timer when it is already running? If so, how might this occur? If not, why is it impossible?

35. A 3000-km-long T1 trunk is used to transmit 64-byte frames using protocol 5. If the propagation speed is 6 μsec/km, how many bits should the sequence numbers be?

36. Imagine a sliding window protocol using so many bits for sequence numbers that wraparound never occurs. What relations must hold among the four window edges and the window size, which is constant and the same for both the sender and the receiver?

37. In protocol 6, when a data frame arrives, a check is made to see if the sequence number differs from the one expected and *no_nak* is true. If both conditions hold, a NAK is

sent. Otherwise, the auxiliary timer is started. Suppose that the else clause were omitted. Would this change affect the protocol's correctness?

38. Suppose that the three-statement while loop near the end of protocol 6 was removed from the code. Would this affect the correctness of the protocol or just the performance? Explain your answer.

39. In the previous problem, suppose a sliding window protocol is used instead. For what send window size will the link utilization be 100%? You may ignore the protocol processing times at the sender and the receiver.

40. Suppose that the case for checksum errors were removed from the switch statement of protocol 6. How would this change affect the operation of the protocol?

41. In protocol 6, the code for *frame_arrival* has a section used for NAKs. This section is invoked if the incoming frame is a NAK and another condition is met. Give a scenario where the presence of this other condition is essential.

42. Consider the operation of protocol 6 over a 1-Mbps error-free line. The maximum frame size is 1000 bits. New packets are generated 1 second apart. The timeout interval is 10 msec. If the special acknowledgement timer were eliminated, unnecessary timeouts would occur. How many times would the average message be transmitted?

43. In protocol 6, $MAX\_SEQ = 2^n - 1$. While this condition is obviously desirable to make efficient use of header bits, we have not demonstrated that it is essential. Does the protocol work correctly for $MAX\_SEQ = 4$, for example?

44. Frames of 1000 bits are sent over a 1-Mbps channel using a geostationary satellite whose propagation time from the earth is 270 msec. Acknowledgements are always piggybacked onto data frames. The headers are very short. Three-bit sequence numbers are used. What is the maximum achievable channel utilization for

    (a) Stop-and-wait?
    (b) Protocol 5?
    (c) Protocol 6?

45. Negative acknowledgements directly trigger a response at the sender, while the lack of positive acknowledgements only triggers an action after a timeout. Is it possible to build a reliable communication channel using only negative acknowledgements, and no positive acknowledgements? If it is possible, give an example. If it is impossible, explain why.

46. Consider an error-free 64-kbps satellite channel used to send 512-byte data frames in one direction, with very short acknowledgements coming back the other way. What is the maximum throughput for window sizes of 1, 7, 15, and 127? The earth-satellite propagation time is 270 msec.

47. A 100-km-long cable runs at the T1 data rate. The propagation speed in the cable is 2/3 the speed of light in vacuum. How many bits fit in the cable?

48. Give at least one reason why PPP uses byte stuffing instead of bit stuffing to prevent accidental flag bytes within the payload from causing confusion.

49. What is the minimum overhead to send an IP packet using PPP? Count only the overhead introduced by PPP itself, not the IP header overhead. What is the maximum overhead?

50. The following data stream is sent using a PPP frame over SONET: ESC FLAG FLAG ESC. What is the sequence of bytes transmitted in the payload? Write your answer as a sequences of bytes, each represented by eight ones or zeros. The bit sequence used to represent ESC is 01111101 and the bit sequence to use for FLAG is 01111110.

51. A 100-byte IP packet is transmitted over a local loop using ADSL protocol stack. How many ATM cells will be transmitted? Briefly describe their contents.

52. The goal of this lab exercise is to implement an error-detection mechanism using the standard CRC algorithm described in the text. Write two programs, *generator* and *verifier*. The *generator* program reads from standard input a line of ASCII text containing an n-bit message consisting of a string of 0s and 1s. The second line is the k-bit polynomial, also in ASCII. It outputs to standard output a line of ASCII text with $n + k$ 0s and 1s representing the message to be transmitted. Then it outputs the polynomial, just as it read it in. The verifier program reads in the output of the generator program and outputs a message indicating whether it is correct or not. Finally, write a program, *alter*, that inverts 1 bit on the first line depending on its argument (the bit number counting the leftmost bit as 1) but copies the rest of the two lines correctly. By typing

```
generator <file | verifier
```

you should see that the message is correct, but by typing

```
generator <file | alter arg | verifier
```

you should get the error message.