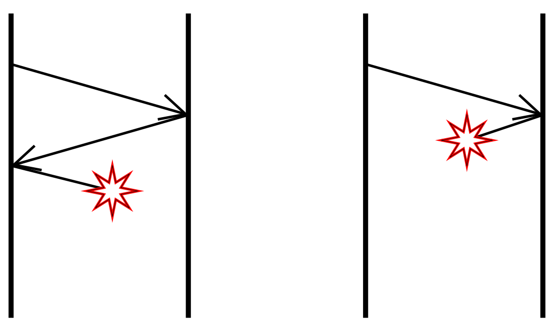
Homework 9

Solution Key

Problem 9.1

> *Hosts A and B follow a connect/disconnect protocol that uses three-way handshakes.... Suppose host A sends a connection request to host B, which is (correctly) acknowledged by B, but A is not ready to actually send data yet, so it decides not to respond. What happens? What state do A and B think the connection is in?*
>
> *What happens if A then changes his mind and, having not sent any data, initiates the disconnect protocol. How would B respond?*

The key thing to understand is that as far as B (the server) is concerned, this situation is indistinguishable from either of the following lost-packet scenarios:



Either way, it's A's move next, either re-sending the first data packet (if in the situation on the left) or re-sending the connection request (if in the situation on the right). So B is waiting. A knows that it hasn't sent anything, but B thinks they're still in the connecting part of the protocol, with an incomplete handshake.

As a result, if A then sends a disconnect request, B's action is based on the assumption that no connection was ever successfully made—either this is a spurious old disconnect request, or data that was sent has been lost. Either way, its response should be to ignore or reject it (depending on the precise details of the protocol).

Problem 9.2

The payload of a packet to be transmitted is the two-byte value 0xBEEF (as represented in hex). If the Hamming code were used (with even parity), how many redundant bits would there be? How many bits wide is the encoded frame? What kinds of errors can it detect and/or correct? Calculate the transmitted frame (and show your work).

First, convert to binary:

B	E	E	F
1 0 1 1	1 1 1 0	1 1 1 0	1 1 1 1

We lay this out in a table skipping all power-of-2 bits, bringing the total bit count up to 21:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
		1		0	1	1		1	1	1	0	1	1	1		0	1	1	1	1

The five redundant bits are then filled in:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
0	1	1	1	0	1	1	0	1	1	1	0	1	1	1	0	0	1	1	1	1

Any single-bit error can be detected and corrected.

Problem 9.3

The payload of a packet to be transmitted is the two-byte value 0xCC6D (as represented in hex). If the CRC code were used with $G(x) = x^4 + x^3 + x^2 + 1$, *how many redundant bits would there be? How many bits wide is the checksummed frame* $T(x)$*? What kinds of errors can it detect and/or correct? Calculate the transmitted frame* $T(x)$ *(and show your work).*

Again, the first step is representing it in binary:

C	C	6	D
1 1 0 0	1 1 0 0	0 1 1 0	1 1 0 1

The generator polynomial is represented as 11101. It will generate four redundant bits: all possible remainders are four bits or less. (If a remainder were five bits, you would be adding another bit to the quotient.) $T(x)$ will thus be 20 bits wide, and able to detect any burst error of length at least 4, or (because $x+1$ is a factor of $G(x)$) any error with an odd number of bits. We append 4 zero bits and do the long division (extra spaces inserted as visual aid, not otherwise significant):

```
             1011 0010 1011 0011 R 111
      --------------------------
11101 ) 1100 1100 0110 1101 0000
        1110 1
          10 010
          11 101
           1 1110
           1 1101
               11 011
               11 101
                  1100 1
                  1110 1
                    10 010
                    11 101
                     1 1111
                     1 1101
                         10 000
                         11 101
                          1 1010
                          1 1101
                             111
```

We XOR the remainder with the extra zero bits, yielding a transmitted bitstream of

1100 1100 0110 1101 0111