

Homework Assignment 2

Due in a week

Directions: Start early! Programming takes time. Problems 1, 2 and 3 will be graded. Problem 4 is given so that you can prepare for the exam and a solution will be provided before the exam.

- **1, HDLC Framing, 25 points:** The HDLC protocol uses a flag 01111110 at the start and end of frames. In order to prevent data bits from being confused with flags, the sender stuffs a zero after every 5 consecutive ones in the data. We want to understand the effect of the following three modifications. For each modification, *assume that all other parts of the framing protocol remain unchanged.*

- Will the framing rules work if we change the stuffing rule to stuff a zero only after 6 consecutive ones? Explain.
- Will the protocol work if we change the stuffing rule to stuff a 0 only after a zero followed by 5 consecutive ones? Explain.
- Will the protocol work if we change the stuffing rule to stuff a 1 only after a zero followed by 6 consecutive ones? Explain.
- The protocol does work if we change the stuffing rule to stuff a zero after 4 consecutive ones. Explain why this is inferior to the HDLC rule.

- **2. CRCs Polynomial View, 25 points:** Consider the generator $x^8 + x^6 + x^2 + x + 1$.

- (20 points). How many undetected burst errors (starting at Offset 0) can there be of burst length 12? To help you do this, note that a burst error of length 14 is a polynomial that starts with x^{11} and ends with 1 with optional terms for the remaining powers between 10 and 1. For an undetected error, the resulting burst error must be divisible by the generator $x^8 + x^6 + x^2 + x + 1$. Instead of seeing which bursts are divisible by the generator, find how many bursts of length 12 are *multiples* of the generator? For example, if we multiply $x^8 + x^6 + x^2 + x + 1$ with $x^3 + 1$ we get a length 12 burst. For 2 points write down the resulting burst error as a polynomial.

Next, notice that the only way to get x^{11} as the highest power and 1 as the lowest power is to multiply by polynomials whose highest power is x^3 and lowest power is 1. Can you write down all such polynomials. How many such polynomials can there be?

- (5 points) How many undetected burst errors (starting at Offset 0) can there be of burst length 11?
 - (Extra Credit, 5 points) Based on the last two answers, can you argue briefly why the probability of a detecting a burst of arbitrary length is $1/2^k$ where k is the degree of the CRC (8 in this case). Hint: consider the ratio of the number of undetected bursts to the total number of possible bursts of any given length.
- **3, CRC program, 75 points:** Write a program (in C, C++, or Java) for a *sender* that takes a bit stream and computes its CRC, as well as a second program for a *receiver* that is very similar to the first except that it checks to see whether a received packet has the right CRC.

The input to the sender program should be a string in C (which you read from a file) containing ASCII 0's and 1's that represents the message to be checksummed. Your output should be a 16-bit CRC for the CRC-16 polynomial $x^{16} + x^{12} + x^7 + x^5 + 1$.

Please read the hints also on the class web page.

To test your programs do the following. Consider the data frame of 40 bits consisting of 111111 (six 1's) followed by 33 zeroes followed by a 1. With the CRC the whole frame becomes 56 bits. You will do your testing on this frame.

- Describe the logic for your 2 programs clearly using a flow chart, a picture, and good comments. (20 points)
 - First find the value of the CRC for the 56-bit frame described above (25 points for a working program)
 - Next, consider all possible 4 bit random errors that can occur in the last 20 bits of the frame (for example, bit 0, bit 10, bit 18 and bit 19 could be flipped in an example 4-bit error). Don't forget that bit 0 is the LSB of the CRC so that CRC bits are flipping as well. For each such error, flip the corresponding bits, and recompute the CRC and see whether the resulting CRC is correct. Print out a line describing undetectable random 4-bit errors only within the last 20 bits (there should be at most a few), stating the bit positions in error. Number the LSB of the frame as bit 0 and the MSB as bit 55.) Also, print out what fraction of 4-bit errors are detected. (20 points)
 - Next, repeat your experiments above for a frame that is the complement of the frame we studied above (all bits are flipped). Do your results change? Explain your results using CRC theory. (10 points)
- **4, Data Link Protocols on Synchronous Links:** So far in all our Data Link protocols we have assumed the links to be asynchronous in that the delay of a frame or ack could be arbitrary. Now we consider the case that the time taken for a message or ack is 0.5 time units. Further senders send frames only at integer times like 0,1,2. When a receiver gets an error-free frame (sent at time n) at time $n + 0.5$, the receiver sends an ack back that arrives (if successful) just before time $n + 1$. Suppose we use the standard alternating bit protocol except that the sender also waits to send at integer times.
- Does the sender need to number the data frames? If your answer is yes give a counterexample to show what goes wrong when it does not.
 - Does the receiver need to number the ack frames? If your answer is yes give a counterexample to show what goes wrong when it does not.
 - Describe a simple protocol for the sender to initialize the receiver state after a crash?