CPS 470/570: Homework 2 **Due Thursday October 16, 1995**

Parity and CRC Comparison

This assignment is intended to give you some hands-on experience in communications programming and protocol evaluation. You will use a network simulation program for evaluating error detection codes. The system currently uses a simple even-parity error detection mechanism. You are to develop the CRC-8 calculation and then perform experiments to compare the error detection capabilities of the parity and CRC techniques.

The ultimate goal of this project is to fill in the following table based on the results of your experiments.

	$P_b = .001$	$P_b = .0001$
Expected % of frames in error		
Observed % of frames in error		
% errors detected using parity		
% false errors indicated using parity		
% errors detected using CRC		
% false errors indicated using CRC		

To obtain your observed percentages, your experiments should include at least 10 observations (trials) per configuration. For each set of parameters, you should run the simulator 10 times and note the results. You should summarize the results by calculating the percentages requested in the table. For instance, if the parity experiments using $P_b = .001$ show an error on 7 runs out of 10 then the *Observed % of frames in error* is 70%. If only 4 of those 7 error runs are correctly identified by the parity bit, then the *% errors detected using parity* is $\frac{4}{7}$ or 57%. If in 1 of the 3 error-free runs the parity check indicates an error where there was none, then the *% false errors indicated using parity* is $\frac{1}{3}$ or 33%.

1 The Simulator

The simulator software consists of three programs currently running on *udcps* as well as the Sun SPARC systems in Anderson 131. The programs are:

- **Sender** The sender reads data from a file and sends the first 1024 bytes of the file to the network simulator. It calculates an error detection block, currently a parity bit, and appends that to the data as a trailer. The sender then transmits this combined block of data to the simulator.
- **Simulator** The network simulator is designed to simulate the network between two communicating devices. The simulator receives data from the sender and forwards it on to the receiver. It takes a bit error rate argument and then randomly introduces errors in the data stream according to the given error rate.
- **Receiver** The receiver receives data from the network simulator. It then performs the error detection function on the received data. Currently that function is a parity check. The receiver then compares the result to the information received in the trailer. If the checks match then the data is declared to be correct.

The receiver also reads the same file that was originally sent and compares it to the received data. In this way, the receiver can determine for sure whether the data was in error and report on the accuracy of the error detection mechanism.

The simulator is implemented in C using the UNIX operating system. The code should be fairly portable to other UNIX systems and MSDOS. If you decide to move it, be very careful of your data sizes.

2 Program Usage

First, you should copy all of the files from the directory */home/rjc/pub/cps470/nwsim* to your own directory. These are the source files for all three programs.

The programs are written to use standard input and standard output. You will perform your experiments by sending the output of the sender to the network simulator and then sending the output of the simulator to the receiver. You may do this using intermediate files or using pipes.

For example:

```
sender datafile | nwsim -p .001 | receiver datafile
```

or

```
sender datafile >send.out
nwsim -p .001 <send.out >sim.out
receiver datafile <sim.out
```

Note: The datafile can be any file larger than 1024 bytes. The program will only use the first 1K. Just make sure you use the same file for both programs. The programs will run for just one transfer.

3 Writing the CRC Function

You will need to write a CRC calculation function to take the place of the parity function currently used by the sender and receiver. Like the parity function, this function should take a character pointer argument that points to the data to be used in the calculation. A second integer argument should indicate the number of bytes in this data field.

The function should calculate a CRC-8 which uses a 9-bit divisor (X^8+X^2+1) and returns a 8-bit frame check sequence. This 8-bit sequence should be appended to the data at the sender before transmission to the simulator.

There are several different ways to implement the CRC. You can do a software implementation of the "hardware shift registers" described in the text or you can implement the long division. Your textbook provides a division based algorithm for doing an 8-bit CRC.