COMP 3980 Final Project

Radio Modem Protocol Driver

Objective:

- To learn how to write a simple half-duplex protocol driver.
- To understand and implement error recovery algorithms on a wireless channel.

Your Mission:

Write a basic driver for half-duplex protocol. For the purposes of testing and protocol verification you will provide a GUI to control and configure the operation. I suggest using the terminal emulator you have written as the basic GUI. The task is to have to machines carry out data transfers using the your protocol. The class will be divided into five groups for this
assignment. Each group will appoint a group leader who will manage this project.

Background :

The BISYNC protocol has been covered in your lectures. This is an example of a half-duplex protocol and it can be used as the basis for designing your own protocol. The basic task is no different from what you have been doing for your terminal emulators. However in carrying out the data transfer, each device must bid for the link and proceed when given the go ahead from its peer. In addition you will have to incorporate time-outs into your program. These are best illustrated using a state diagram.

Bear in mind that the data you will send back and forth will be text data, broken up into discreet packets. You can make the packets constant or variable. Both devices will be transferring data at the same time. This means they will be sharing the line so no one device can "hog" the line for large amounts of time.

Note that there will be a certain amount of flexibility in this project. If you have good ideas on how to implement this driver then by all means discuss these with me during the labs. You will be provided with more information on the implementation of this protocol during your lectures.

You will have to make extensive use of the protocol analyzer for this assignment when debugging your programs. Make sure you are comfortable with using this tool.

Project

- This project will deal strictly with the protocol implementation itself without any error correction other than ARQ.
- You will simply implement the half-duplex protocol details and provide for basic error detection and retransmissions.
- For error detection you can use either checksums or CRC. If an error is detected the previous packet will simply be retransmitted. You are free to use publicly available source code for CRC and checksums in your implementation.

Constraints

- Your Win32 implementation of this protocol must be completely event driven.
- The program will be initialized and executed using menus and buttons provided with your GUI.
- You are required to collect some protocol statistics such as number of packets transferred, the BER (Bit Error Rate), number of ACKs and NAKs and display these statistics in real time.
- Collect statistics for both parts with identical operating parameters. This way you will get a good idea of the affect of error correction versus ARQ.
- Your submission must include a brief **technical report** commenting on your findings together with a conclusion.

DUE DATES:

Design Work: November 17 – 0930 hrs

Must include all STD's, pseudocode, timelines and task breakdowns for each team, member and associated deadlines.

In Lab Test: December 3

You will be required to demonstrate your programs during the lab. Each group must test their program with another group to demonstrate the validity of the protocol.

TO BE HANDED IN: (Due Date: December 3, 2008 - 0930 hrs)

1. Detailed design work (State transition diagrams and pseudocode) of the driver functions.
2. A technical report on your findings as outlined above.
3. Printed program listing of the program.
4. Complete C/C++ source code and executable version of your program (on CD).