CS 536 (Park) **Assignment VI** **Due: Oct. 3 (Thu.), 1996**

PROBLEM 1

(a) Read Section 4.4 from P & D. Define *blocking* and *nonblocking* interconnection networks. Give an explanation of what power is gained by going from a $\log n$-redundancy factor switch to a $2\log n$-redundancy factor switch. *Redundancy factor* is the multiplicative size increase needed for a system to achieve a form of "fault-tolerance." Your summary/explanation should be no more than 2 pages long.

(b) Read Sections 5.1 and 5.2 from P & D.

PROBLEM 2

Extend the client/server application of Problem 2 of Assignment V such that client and server processes are allowed to run on different hosts and communication is achieved via socket calls invoking TCP/IP. Thus the FIFO IPC structure needs to be replaced by the appropriate networking calls. Note that TCP is a connection-oriented, reliable service. Test your application using 4 machines in the Xinu Lab; for now, you may hardcode their IP addresses in your program. On your host machine, open four windows, one for each separate host, running the server on one machine and three copies of the client on the other machines. The clients should be requesting the `host` command to be executed on the server. In addition to the code and script files, hand in a screen dump which shows the contents of the four windows. Note, for this problem, you need to be physically in the Xinu Lab.

PROBLEM 3

Repeat Problem 2, however, with socket calls invoking UDP in place of TCP. Which communication mechanism is more suited for this client/server application? Justify your answer.

PROBLEM 4

This is an extension of Problem 4, Assignment V. Give a pseudo-code description of your `send` and `receive` modules and their submodules. They may implement either stop-and-wait or the generic sliding window control. Invent your own HDLC-like frame format and describe the meaning of the various fields. Also, justify their choices. Your pseudo-code description needs to be sufficiently detailed and thought-through such that it is only steps away from being implementable.