ABSTRACT

This project is the development of a communications program designed to simulate a token-ring network. The program provides limited LAN capabilities to a set of personal computers without special interface hardware. The computers are connected through serial ports using null modems and serial cables. The program provides facilities to send messages from any computer on the network to any other computer on the network and keeps track of the current users.
TABLE OF CONTENTS

I. INTRODUCTION 1

II. PHYSICAL DESIGN 3

III. MODULE DESCRIPTIONS 5

IV. SUMMARY AND CONCLUSIONS 22

V. BIBLIOGRAPHY 23

VI. APPENDIX A. PROGRAM DOCUMENTATION 24

VII. APPENDIX B. PROGRAM LISTING 27
I. INTRODUCTION

Background and Rationale

As computer networking has grown in popularity, so has the desire to study network design and operation. A great deal of literature exists on the subject of computer networking, but very little is available to study the design of network software. The publications that are available are very technical and difficult to study in a classroom environment.

The intent of this project has been to develop a program that provides a limited set of network functions on a basic personal computer, without the need for specialized hardware. The functions provided are sufficiently elementary for students to investigate data communications and network applications.

The idea for the project came from a telecommunications instructor. The instructor wanted to teach the methods for interfacing with a LAN by having students write programs utilizing LAN services. Since no simple, inexpensive LAN program was available, the instructor was unable to accomplish this. After taking the class and speaking with the instructor, the author determined that such a program could be written that would be small enough to study, yet provide an environment simulating the workings of a simple token-ring network.

Project Goals

The goal of this project is to produce a program that is simple enough to be studied in detail by telecommunications students, yet provide a simple token-ring network environment for the development of network applications. When these goals are met, the program will be a useful tool for teaching both network design and network applications.

The system that has been developed is called GOTOKEN. GOTOKEN accomplishes these goals in the following ways:

1) The functions in GOTOKE were written in the C programming language. C is sufficiently powerful to provide efficient code, yet remain easy to understand in a classroom environment. This provides the tool for network design studies.

2) GOTOKE provides a simple, but effective, protocol for the transmission of data between computers. It provides a simple token-ring network environment for the development of network access applications.
Project Procedures

A detailed analysis was performed to determine the feasibility of this project. During this analysis, several meetings took place with the telecommunications instructor. The desired results of the program were discussed as well as several references that would be useful in the research. "Programmer's Problem Solver" by Mr. Robert Jourdain was instrumental in developing the necessary knowledge to prepare the computer for interrupt-driven serial data communications. The two references, "Data Communications, Principles & Problems" by Mr. George J. Moshos, and "Data Communications, Computer Networks and Open Systems" by Mr. Fred Halsall, imparted a great deal of knowledge about networks and protocols. The C programming language books, "Secrets of the Borland C++ Masters" by Mr. Ed Mitchell, and "Mastering Borland C++" by Mr. Tom Swan, were used for developing the code for the project. Through these meetings and a review of the available reference materials, it was determined that the project was feasible. The detailed analysis provided the framework necessary to develop the GOTOKEN package.

Several tasks had to be performed before GOTOKEN could become a functioning package. The first task was to select a protocol and design a workable set of data transmission formats. Next, a set of modules had to be identified to provide the functions desired of the program. After identifying the modules, the code had to be written to implement them. Finally, after coding all the modules, the program was tested on a set of computers to insure the functionality of the program.

With all of the above tasks completed, the project is now a success.
II. PHYSICAL DESIGN

System Requirements

The GOTOKEK system is designed to run on IBM-compatible personal computers. The minimum hardware requirements for each computer are: an 80286 microprocessor, two serial communications ports, 2 megabytes of random access memory and a hard disk drive. A minimum of two computers are needed, with serial cables and null modems for each computer. Computers in the system are to be connected in the following way: COMM 1 of a computer is connected to COMM 2 of its downstream neighbor. All connections must be in such a manner that a ring is formed (see FIGURE 1 on page 4). The program design allows for a maximum of eight computers to be connected to the ring.

All computers on the ring must be running MS/DOS version 5.0 or later. No terminate and stay resident programs can be loaded. No software requiring the use of the serial communications ports, or their associated interrupts, can be loaded.

The GOTOKEK system has been designed to execute from a hard disk drive.

System Interface

The GOTOKEK system is an interrupt-driven program once the initial setup is complete. To accomplish the setup, GOTOKEK prompts the user for a one-digit node address to identify the system on the network. GOTOKEK then prompts the user for the communications settings.

Starting The GOTOKEK System

The GOTOKEK system is a utility to be used by applications programs. An application program, PASSMSG, has been included to demonstrate the use of the GOTOKEK system. To start the GOTOKEK system, PASSMSG makes a call to MESSAGING which in turn establishes the token ring.
FIGURE 1

diagram of physical connections