OREGON STATE UNIVERSITY COMPUTER SCIENCE DEPARTMENT

A Comparative Analysis of the Tektronix 4051 Computer System

T. G. Lewis
W. Leggett
J. Joiner
K. Malik
S. Simon
J. Wilson
T. Elton
D. Kim

Department of Computer Science
Oregon State University

78-1-4

A COMPARATIVE ANALYSIS OF THE TEKTRONIX 4051 COMPUTER SYSTEM

T. G. Lewis, Ph.D.

W. Leggett

J. Joiner

K. Malik

S. Simon

J. Wilson

T. Elton

D. Kim

Computer Science Dept.
Oregon State University
Corvallis, OR 97331
(503) 754-3273

June 25, 1977

INTRODUCTION

In March, 1977, a team of students under the guidance of Dr. T. G. Lewis began studying the Tektronix 4051 personal computer system. The purpose of this study was to determine strengths and weaknesses of the 4051 when placed in the data processing environment. This report summarizes the team's findings and terminates with a list of recommendations. If these recommendations are followed by Tektronix, the 4051 can become a viable competitor in the small business applications market.

The evaluation team set about to answer five (5) broad questions using intuition, reason, and hard facts collected from actual use of a 4051. The system was subjected to a variety of tests, used to develop several "typical" data processing application programs, and evaluated as if the team was a first-time user with very little computer background or training. The questions raised, and to a great extent answered by this report are:

I. How Does The 4051 Compare With Other Systems In Its Price Range?

The team selected two other successful systems to compare. The Wangco 2200-T processor was selected as representative of firmware BASIC language systems with a scientific heritage--the same heritage as the 4051. The Datapoint 1100

processer was selected as representative of software systems with a business data processing heritage. The team hoped to obtain a wide range of test results from such diverse equipment.

II. How Well Suited Is The 4051 To Business Data Processing Applications?

The team reports a list of "laws" that should be followed by any manufacturer of business data processing computers. These "laws of personal computing" are reiterated throughout the report. Furthermore, the 4051 was used to implement a variety of applications in business. These application programs, and the problems encountered by attempting to implement application programs, gave the team a basis for evaluating the 4051 as a data processing system.

III. How Effective Are The 4051 Documents, Manuals, And Programming Standards?

The team evaluated six manuals made available to them by Tektronix. The programming standards were not provided, and so were not evaluated as originally planned. The objective of this part of the study was to place the team in the role of a first-time user and determine if the manuals and documents are adequate.

IV. How Well Does The Hardware And System Software Perform In "Typical" Data Processing Applications? The team's goal in this part of the evaluation was to perform a variety of benchmark tests on the 4051, Wangco 2200-T and Datapoint 1100 and use the results to compare performance. The benchmarks were selected to be simple, repeatable, and oriented toward a specific subcomponent of each system. For example, the team was able to determine the effect of a random access diskette subsystem versus a sequential access tape subsystem on the performance of the total system. This result will be useful in predicting improvement when a similar diskette subsystem is added to the 4051.

V. What Should Be Done To Improve The 4051 As A Competitive Alternative To Other Systems? A great deal was learned about performance of all the systems studied. The lessons are summarized in this final portion of the report. The recommendations include new ideas, combinations of old ideas, duplication of features found in the competitor's systems, and small or minor modifications that contribute to increased desirability of the 4051 for small data processing applications and novice users.

This report contains the methods, programs, results, and combined summary of findings of two teams (3 and 4 members, each). Additional questions regarding this report should be directed to Dr. T. G. Lewis.

HOW DOES THE 4051 COMPARE WITH OTHER SYSTEMS IN ITS PRICE RANGE?

This part of the study briefly details three systems used in the comparative analysis: The Wangco 2200-T Datapoint 1100, and Tektronix 4051.

WANG 2200-T

The Wangco and Tektronix 4051 aystems share a common conception. Both machines were originally designed for sophisticated, engineering oriented users. Both systems are built around an interpretive BASIC language and command interpreter stored in a firmware ROM. Both systems were originally designed as an "intelligent" terminal containing only a cassette tape drive intended to store programs.

The Wangco 2200-T processor is an extremely fast processor with a large bright, character CRT, aesthetic terminal, and well engineered keyboard. The basic price of the 2200-T is $9,200 for a 32K memory version.

The 2200-T may be connected to other devices and/or telephone lines via an intelligent interface. This interface contains an Intel 8080 processer and ROM that simulate a variety of common protocols. This gives the system a limited amount of communications capability at very low cost.

The 2200-T BASIC interpreter, operating system commands, and system control resided in a 42.5K byte ROM

This basic ROM may be expanded to include additional scientific and/or business related functions. For example, in addition to an editor ROM, an opticnal sort/merge ROM, General I/O ROM, and diskette file maintenance ROM (KFAM, keyed file access method) may be purchased at additional cost.

Additional devices for printing and storage may be included in a total Wang system. Dual diskettes and a printer may increase the purchase price to over $20,000.

The top of the line, Wangco VP is an even faster processor (by a factor of up to 10) that uses distributed cpu's to allow multiterminal access to a common database stored on a cartridge disk subsystem. This concept reduces software complexity by replicating hardware, and produces a highly modular, inexpensive, upgrade for businesses that expand beyond the 2200-T capability.

The Wang 2200-T BASIC system is completely described in a single, easy to use, easy to reference, and aesthetic manual. While a beginner may have difficulty understanding many of the terms found in the Wang manual, the team's evaluation placed the Wang manual far above the others in this evaluation with respect to "usefulness".

The Wang 2200-T is an attractive machine, utilizing colorful hardware and aesthetic enclosures likely to be admired by a businessman. The only feature possibly resulting in unfavorable comment by a businessman or novice user is the Wang keyboard. The keyboard is too cluttered,

containing additional control keys (beyond a typewriter) and dual designation to assist a BASIC language programmer. The inclusion of BASIC and operating system keywords is viewed as a cryptic remnant by the naive user.

Additional features that may be important to a user in selecting a small business system also found in a 2200-T system include: atomization of keywords (short tokens replace the keywords to save memory space), and multiple statements per line (saves up to 4 times the space).

The 2200-T system is upward compatible with the larger, more powerful VP processor system. Upward compatibility of programs adds to the "comfort level" of a first-time user. Also, Wang is supported by a nationwide network of analysts and customer service offices, and a users group (one of the largest).

DATAPOINT 1100

The datapoint 1100 system was originally conceived as an intelligent terminal designed to replace IBM terminals. As a result of this orientation, the 1100 and larger Datapoint systems (2200 and 5500) offer the most sophisticated communications options of the three systems studied here. IBM emulators, autodial peripherals, and network software is provided as part of the 1100 operating system. Therefore, a user is able to communicate with other 1100's, IBM equipment, etc. without programming or purchasing an interface (the purchase price includes a communications adapter, but may be dropped to lower the price).

The basic processor price of $8,500 places the 1100 at the bottom of the price competition when compared with the 4051 and 2200-T. Furthermore, the price becomes more attractive when quantity discounts of up to 40% are considered.

The 1100 offers an intriguing benchmark system because it is designed around an Intel 8080 architecture (the 8008 was originally designed for Datapoint, but Datapoint eventually went its own way). The 1100 has evolved as a stand-alone minicomputer system for business data processing, and in many ways is more suited to business applications than the other systems evaluated because it:

1) supports virtual memory,
2) supports sequential, random, and inverted file structures,
3) supports uniform record and file formats for a variety of languages,
4) can operate unattended under control of a time-of-day clock,
5) supports protected files, error detection, and error recovery procedures.

The Datapoint 1100 supports a variety of languages, but for purposes of comparison we evaluate only the BASIC interpreter here. This interpreter is <u>not</u> used to program solutions to business problems, however because the DATABUS language is more suited to dataprocessing than BASIC (virtual memory, forms control, user prompting, error recovery, unattended operation, communications interfacing, and numerous other features that partially meet the goals of personal computing).

The Datapoint 1100 with two diskettes, 16K main memory, a CRT, and line printer was used in this evaluation.

As indicated before, a complete system as used in these tests could sell for considerably less than the other two systems ($12,500 in quantities of 3).

The 1100 BASIC is a minimal version of Dartmouth BASIC with few useful features for business applications. It is a software interpreter developed over 5 years ago, and whose further development has not been pursued by the company.

The 1100 runs under control of an operating system that includes autokeyed operation for unattended operation, communications utilities for (DAA) autocalling other systems, merge/sort utility, file maintenance utilities, and file security (Delete, write protect) utilities.

The BASIC interpreter does not run under the virtual memory mapping interpreter, hence, we were unable to measure its performance. The system does support sequential and random access to disk files, however.

The 1100 system used by the evaluation team was aesthetically pleasing. It is the least obtrusive system (to a novice user) of the three systems we tested. There are only 5 keys on the keyboard that are not found on a typewriter, and the console, diskette, and CRT are housed in one attractive desk. The console CRT screen, however, is much too small for easy viewing.

The Datapoint system is described in several manuals, each one easy to index into, but in many cases not clearly written. Service is maintained by a nationwide network of regional offices.

THE TEKTRONIX 4051 SYSTEM

The Tektronix 4051 computer system was originally conceived as an intelligent graphics terminal. It consists of a keyboard, cassette tape, and storage scope CRT.

The system is driven by a Motorola 6800 processer under the control of 32K of ROM. The ROM system contains a BASIC interpreter with extensions for graphics picture processing, text editing, control of I/O devices, and a moderate level of autokey operation.

The storage screen has a hardware viewport of 100 x 135 grid points. It has high resolution capability, but the screen is very dim and difficult to view in normal light. Plotting speed is moderate, and since it is a storage tube, interactive graphics is very restricted.

Additional devices were not available to the study, but a Diablo printer was connected to the RS232-C port (@41) for the duration of the tests. The system without printer was priced at $11,500 (32K memory). This placed the 4051 at the top of the price scale when compared with Wang and Datapoint systems.

The Tektronix system is described in several extensive

manuals, mostly overlapping in content. Indeed, the manuals contributed to a level of confusion because they were too prolific.

Additional communication is provided by an IEEE Standard GPIO bus and firmware to allow access to other devices from BASIC. The system may be further enhanced in this direction if access to the 6800 processor is gained through the BASIC interpreter's CALL primitive.

The processor is enclosed in an attractively styled box, but the rather austere color and confusing keyboard will alienate many novice businessmen. The keyboard is designed for a developer rather than a user.

Service is provided by Tektronix offices located throughout the world.

SUMMARY

The Wangco and Datapoint computer systems were chosen as benchmarks for the Tektronix 4051 system for the following reasons:

1) They are among the dominant systems currently in the small data processing market.

2) They represent both extremes (a system similar to a 4051, and a system much different than a 4051).

3) They were readily available.

The results of performance evaluation reveal strengths and weaknesses of all three systems. An examination of each system is essential in order to conclude which is "best" for a specific application.

HOW WELL SUITED IS THE 4051 TO BUSINESS DATA PROCESSING.

APPLICATIONS

The suitability of any computer system for use in a data processing environment depends upon the ability of the system to perform the data manipulation tasks of the given environment. While data processing covers a broad range of computer applications, there are certain basic capabilities that are required by most data processing environments. Data processing generally involves the functions of data entry, data storage and access, calculations with data, and the preparation and printing of reports. These basic processing functions may need to be performed separately, together, or not at all, depending on the application. Some common data processing applications which use the basic processing functions are listed below.

A. Accounting Functions:
   1. Payroll file processing.
   2. Sales record processing.
   3. Expenditure record processing.
   4. Mortgage or loan payment schedules.
   5. Depreciation schedules.
   6. Tax calculations.
   7. Simulation and trend analysis.
   8. Updating of current budget and general ledger.
   9. Calculations of customer billing and services.
  10. Calculations of computer usage and cost.

B. Report Writing Functions:
   1. Printing of payroll checks.
   2. Printing audit trails on all processing and file updating.
   3. Printing sales reports.
   4. Printing expenditure reports.
   5. Printing budget reports (current and annual)
   6. Printing net and gross profit reports.
   7. Printing bills for customers.
   8. Printing inventory reports (price and level control)

9. Printing mailing list.
10. Printing advertisements (media Preparation).
11. Printing file inquiry reports.
12. Printing of government and other agency report forms.

C. Database Management Functions (Updating, Data Entry, and Creation):
1. Maintain employee records such as salary, benefits, tax, and other personal statistics.
2. Maintain inventory file of parts, stock on hand, and equipment used by the business.
3. Maintain a budget file for current and annual expenditures and income (including liabilities, assests, and accounts receivable.
4. Maintain current customer billing, service records, and mailing lists.

The applications of a small computer system, when placed in a data processing environment, reflect a set of well defined objectives and rules. These rules, once discovered, can be used by a computer manufacturer to improve the product.

The evaluation team was able to condense many of the thoughts and lessons learned into the following rules. Awareness of these rules can be a great aid in understanding the goals of a well designed business system.

RULES OF PERSONAL COMPUTERS IN THE BUSINESS DATA PROCESSING WORLD

The small business market for data processing equipment is one of the fastest growing segments of the computer industry, according to Datapro Research and Frost and Sullivan. This market is characterized by hardware configuration involving a printer, control processor, one or more CRT terminals, and disk or diskette mass storage units.

In a typical data processing application, the central computer is only one part of a system designed to store large files, print reports, checks, and statements, and interact with naive users. In a small business system the emphasis is on total system performance rather than cpu speed or "power".

Indeed, the true or effective "power" of a small computer system is measured by the amount of work it can perform when compared to human workers. For example, a system that prints mailers and eliminates the need for an additional employee, is worth the salary required for a human employee to perform the identical function.

In short, there are a few known "laws" of small business computing that guide the current market. These laws are called <u>personal computing maxims</u> because the main feature of the small business system is its impact on personal computing.

1). Personal Computing Equals Interactive Computing: The "personal-ness" of a computer increases directly proportional to its interactiveness.

The trend of the past decade has been toward more personalized (humanized) systems. Personal computers require little training to use, little maintenance, and no computer specialists to manage.

From batch to remote-job-entry, and eventually to timesharing; computers have tended toward more dispersed useage. In the last 8 years the trend has been even more highly oriented toward stand-alone, dispersed, systems. In fact, hardware/software integration has lead to turnkey systems requiring very little involvement on the part of a businessman.

2). As The Power Of A Personal Computer Increases, Its Price Decreases.

This law states a counter intuitive observation that repeals Grosch's Law; "as the power quadruples, the price doubles". A very rough measure of power follows:

$$\text{Power} = \frac{\text{MIPS} * \text{STORAGE CAPACITY}}{\text{COST}}$$

MIPS = Million instructions per second

STORAGE CAPACITY = Characters of Mass (on-line) Storage in bytes

COST = purchase price

As a systems's capability increases, the number of problems it can solve increases. The increased base of applications leads to greater sales volume which in turn leads to mass production, quantity discounts, and a general decline in unit cost. As a good example of this phenomena, observe the effect of increased cpu speeds in the mid-1960's. Greater increases in MIPS and STORAGE CAPACITY supported timesharing. As a result of timesharing, the cost per unit of computation declined.

Currently, increased storage density is propelling development of large databases. The database craze is creating more applications, leading to decreases in storage device costs.

The logical conclusion of the second law is that hardware will become so inexpensive as to remove itself from consideration in a small business system. This effect has already removed the cpu from dominance in computer selection. This leads to the next maxim.

3). Software Is Hard, Hardware Is Soft: It is economically more feasible to build a computer than to program it.

Clearly, the increasing cost of software, and the increasing complexity it represents is the major cost factor in contemporary computer based systems. There are two consequences of this rule:

a) Programs and data should be shared, but hardware should be replicated.

It is far better to replicate a hardware unit than to risk increased software complexity by placing burdens on the software. For example, a multiterminal system may be desired if common access to a database is needed (shared data). The most economical solution to this problem dictates that each terminal contain its own cpu rather than burden the software with multiplexing algorithums. This philosophy is seen in the Wangco VP system in which dispersed processors are connected to a central disk processor. The second consequence is this:

b) Timesharing failed: people couldn't understand it.

Complex software has a tendency to be visible to a user, whereas hardware has a tendency to be transparent to a user. Exposing a businessman to complex operating systems, (OS/360) languages, (COBOL, PL/I) and operating procedures (switches and knobs) alien to the operation of his business, is counter to personal computing.

A proper man-machine interface is most important in a data processing environment. The "best" system is one that requires very little effort to use. This is succinctly summarized in the next law.

4). Knowledge Costs More Than Software and Hardware: The usefullness of a personal computer increases inversely proportional to how much people need to know in order to us it.

Past systems have failed due to the amount of training required (JCL, telephone modems, jargon), the

myths ("it's complicated"), and poor service that results when a user is unwilling to pay for advice.

In short, the system selected by a businessman must appear to be simple, elegant, and aesthetic.

5). The Color, Shape, and Size of a Personal Computer Is More Important To A User Than What Is Inside Of It.

The "package" is the message in data processing environments. The system should require as few buttons, switches, lights, and knobs as possible. The least amount of jargon (especially in manuals), and least number of steps to follow for a given procedure is the most desirable.

Service fills the gap between a user's knowledge about computers, and the computer's lack of capability (in human terms). Since service is a very expensive commodity, it is to the manufacturer's advantage to minimize the need for service. This can be done by conforming to the rules of simple use, elegant appearance, and incomplicated interactiveness built into the system.

Finally, there are several aspects of interactive computing that impact software. In particular, effective high level languages and hidden operating systems are critical to a simple, aesthetic, and articulate interface with business application users.

6). BASIC is to Personal Computers As Sign Language Is To English.

English has it's "pig Latin", and programming languages has BASIC. Like speaking in pig Latin, BASIC programs are easy to write. Unfortunately they are hard to understand. Few systems permit indentation, structuring, comments (without memory penalty), and full control over the system. Some common shortcomings of BASIC are:

a) poor error recovery facilities,

b) dynamic memory mapping of programs too large for main memory,

c) restricted data structures,

d) limited user prompting,

e) inadequate software security and protection--copyrighting, file security, interlock mechanisms for shared files.

f) slow executions--interpretive,

g) inadequate primitives for data processing, e.g. no sorts, file access constructs, or forms handing constructs.

One compensating advantage of BASIC is that it is easily implemented, easily learned, and rapidly put to use in developing programs. It is an ideal language for beginners and for implementing small programs that will eventually be thrown away rather than modified.

7). An Operating System Is A Feeble Attempt To Include What Was Overlooked In The Design Of A Programming Language.

Clearly, the operating system of most systems represents nothing more than an extension of the programming language used to develop an application.

A job-control-language is usually invented as a remedy to the inadequacy of the language. In interpretive systems, the operating system disappears because the designer of any interpreter is confronted with this hoax.

The systems analyzed in this evaluation do not realize all of the desired features stated here. How well each system meets these criterion must be judged after examining data presented in subsequent sections of this report. The following "ultimate" law should be kept in mind while examining the data.

8). The Ultimate Personal Computer Is A Robot; The goal of personal computing is to reduce the differences between people and computers.

Computers are designed to do what people do, only

1) faster--e.g. summarize the population census in 2 to 3 years instead of 15 years.

2) more accurately--e.g. perform air traffic control over an airport.

3) cheaper--e.g. replace menial, semi-skilled and lower management jobs.

As a result of this last law, personal computers pose a challenge, or threat, to mankind.

BUSINESS APPLICATION BENCHMARKS

The team examined the features of the 4051 Graphics Terminal by applying it to some typical data processing applications. Sample programs were taken from How to Profit From Youn Personal Computer (Hayden Books, 1978), by T. G. Lewis, and were modified to run on the 4051.

The application programs include: (1) An accounts receivable program (data entry, file update, and posting); (2) A general ledger program (data entry, file update, and simple accounting); (3) a mailing list program (sorting); (4) and a database retrieval program (file creation, update, and inquiry). Also, to evaluate the graphics capability of the 4051, a program was developed to draw pie charts for the general ledger program data.

The following includes a brief description of each application program followed by a discussion of the useful and less useful features of the 4051 observed while running the application programs.

Accounts Receivable Program

The accounts receivable program is a typical data processing application program used by a business. The idea is to maintain a customer data base with a master file and transaction file. A program must create transaction files containing customer charges. Another program is used to update the master file and print bills for customers owing a payment each month. There are three programs altogether. The first is used to create the master file, the second is to create the transaction file, and the third program is used to merge the master file with the transaction file, compute balances, print a statement for each active account, and update the master records.

The 4051 programs for AR are shown below:

```
690 PRINT "--------------------------------------"
700 REM ----------------------------------------
710 LET T1=T1+1
720 REM ----------------------------------------
730 GO TO 270
740 REM----------------------------------------
750 FIND 13
760 WRITE @33:S1
770 WRITE @33:P2
780 WRITE @33:P1
790 WRITE @33:T1
800 CLOSE
810 PRINT S1;P2;P1;T1
820 STOP
```

```
100 REM ****************************************
110 REM        A/R TRANSACTION BUILD PROGRAM
120 REM              COPYRIGHT 1977
130 REM ****************************************
140 DIM A$(8),C$(16),Y$(1)
150 REM---------------------------------------
160 REM FILE 13 IS TRANS
170 REM FILE 14 IS ITRANS
180 REM---------------------------------------
190 FIND 13
200 READ @33:S1
210 READ @33:P2
220 READ @33:P1
230 READ @33:T1
240 CLOSE
250 REM---------------------------------------
260 PRINT "THIS IS A PAYMENT (P), OR A CHARGE(C) ?"
270 INPUT Y$
280 IF Y$="P" THEN 310
290 IF Y$="C" THEN 330
300 GO TO 260
310 LET S=-1
320 GO TO 350
330 LET S=1
340 REM---------------------------------------
350 PRINT "PATIENT ID :"
360 INPUT I
370 IF I=0 THEN 150
380 PRINT "DATE OF TRANSACT. MM/DD/YY :"
390 INPUT A$
400 PRINT "COMMENT (16 CHAR. MAX) :"
410 INPUT C$
420 PRINT "AMOUNT $ :"
430 INPUT B
440 REM-------------------------------------
450 LET B=S*B
460 LET P2=P2+B
470 REM-------------------------------------
480 FIND 13
490 FOR J=1 TO S1*4
500 READ @33:D
510 NEXT J
520 WRITE @33:I
530 WRITE @33:A$
540 WRITE @33:C$
550 WRITE @33:B
560 CLOSE
570 FIND 14
580 FOR J=1 TO (S1-1)*2
590 READ @33:D
600 NEXT J
610 WRITE @33:I
620 WRITE @33:S1
630 CLOSE
640 REM-------------------------------------
650 LET S1=S1+1
660 REM-------------------------------------
670 PRINT I,S1-1,A$
680 PRINT " ",C$
690 PRINT " ","$";B
```

```
700 PRINT "-----------------------------------"
710 GO TO 260
720 FIND 13
730 WRITE @33:S1
740 WRITE @33:P2
750 WRITE @33:P1
760 WRITE @33:T1
770 CLOSE
780 PRINT S1,P2,P1,T1
790 REM-------------------------------------------
800 STOP
```

```
100 REM*****************************************
110 REM              A/R POSTING PROGRAM
120 REM                COPYRIGHT 1977
130 REM*****************************************
140 DIM N$(16),S$(16),C$(16),P$(16),A$(8),D$(16),U$(8),X$(1)
150 REM-----------------------------------------------
160 REM FILE 11 IS MASTER
170 REM FILE 13 IS TRANS
180 REM FILE 15 IS ERROR
190 REM-----------------------------------------------
200 FIND 13
210 READ @33:S1
220 READ @33:P2
230 READ @33:P1
240 READ @33:T1
250 CLOSE
260 IF S1<=1 THEN 1580
270 FIND 13
280 FOR N=1 TO 5
290 READ @33:D
300 NEXT N
310 READ @33:J
320 READ @33:X$
330 READ @33:A$
340 READ @33:D$
350 READ @33:B2
360 CLOSE
370 LET K=2
380 LET S1=S1-1
390 REM-----------------------------------------------
400 PRINT "ENTER DATE MM/DD/YY  :"
410 INPUT U$
420 REM-----------------------------------------------
430 FOR K1=0 TO T1-1
440 FIND 11
450 FOR N=1 TO 8*K1
460 READ @33:D
470 NEXT N
480 READ @33:I
490 READ @33:N$
500 READ @33:S$
510 READ @33:C$
520 READ @33:Z
530 READ @33:X$
540 READ @33:P$
550 READ @33:B1
560 CLOSE
570 IF I<J THEN 1250
580 REM-----------------------------------------
590 IF I=J THEN 680
600 FIND 15
610 WRITE @33:J
620 WRITE @33:A$
630 WRITE @33:D$
640 WRITE @33:B2
650 LET P2=P2-B2
660 GO TO 800
670 REM-----------------------------------------------
680 PRINT
```

```
690 PRINT " ",I
700 PRINT " ",N$,"FAMILY",P$
710 PRINT " ",S$
720 PRINT " ",C$,Z
730 PRINT " "," "," "," ","$";B1
740 PRINT " ",A$," ",D$,"$";B2
750 REM-------------------------------------
760 LET P1=P1+B2
770 LET B1=B1+B2
780 LET P2=P2-B2
790 REM-------------------------------------
800 REM     LOOK
810 IF S1=1 THEN 1110
820 FIND 13
830 FOR N=1 TO 5*K
840 READ @33:D
850 NEXT N
860 READ @33:J
870 READ @33:X$
880 READ @33:A$
890 READ @33:D$
900 READ @33:B2
910 CLOSE
920 LET K=K+1
930 LET S1=S1-1
940 REM-------------------------------------
950 IF NOT(J=I) THEN 1020
960 LET P1=P1+B2
970 LET B1=B1+B2
980 LET P2=P2-B2
990 PRINT " ",A$," ",D$,"$";B2
1000 GO TO 800
1010 REM-----------------------------------------
1020 IF J>I THEN 1110
1030 FIND 15
1040 WRITE @33:J
1050 WRITE @33:A$
1060 WRITE @33:D$
1070 WRITE @33:B2
1080 CLOSE
1090 GO TO 800
1100 REM-------------------------------------
1110 FIND 11
1120 FOR N=1 TO K1*8
1130 READ @33:D
1140 NEXT N
1150 WRITE @33:I
1160 WRITE @33:N$
1170 WRITE @33:S$
1180 WRITE @33:Z
1190 WRITE @33:P$
1200 WRITE @33:B1
1210 WRITE @33:U$
1220 CLOSE
1230 PRINT " "," "," ","BALANCE $";B1
1240 REM-----------------------------------------
1250 NEXT K1
1260 IF S1=1 THEN 1480
1270 FIND 13
```

```
1280 FOR N=1 TO 5*K
1290 READ @33:D
1300 NEXT N
1310 READ @33:J
1320 READ @33:X$
1330 READ @33:A$
1340 READ @33:D$
1350 READ @33:B2
1360 CLOSE
1370 LET K=K+1
1380 LET S1=S1-1
1390 FIND 15
1400 WRITE @33:J
1410 WRITE @33:A$
1420 WRITE @33:D$
1430 WRITE @33:B2
1440 CLOSE
1450 LET P2=P2-B2
1460 GO TO 1260
1470 REM-------------------------------------------------
1480 FIND 13
1490 WRITE @33:S1
1500 WRITE @33:P1
1510 WRITE @33:P2
1520 WRITE @33:T1
1530 CLOSE
1540 REM-------------------------------------------------
1550 PRINT "PROBE, MASTER CHECKSUM=";P1
1560 PRINT "PROBE, TRANSACTION CHECKSUM=";P2
1570 REM-------------------------------------------------
1580 STOP
```

General Ledger Program

The general ledger program uses a chart of accounts file which contains the budget for the year, the year-to-date income and expenses, and the current month income and expenses for each item in the chart. Each month a sequential transaction file is created to update the year-to-date totals and calculate the balance (income - expenses) for the month that has just ended. The program listing for the pie chart General ledger is included below.

```
100 PAGE
110 REM*****************************************
120 REM       CHART OF ACCOUNTS
130 REM*****************************************
140 DIM C$(1),T$(20),K$(3)
150 PRINT "ARE YOU GONNA USE A SCRATCH FILE OR CREATE A NEW FILE?"
160 PRINT "ENTER -YES OR JUST Y- TO CREATE A NEWFGFILE; ";
170 INPUT A$
180 IF A$<>"YES" AND A$<>"Y" THEN 240
190 PRINT "ENTER THE ""LAST"" FILE NUMBER ";
200 INPUT F
210 FIND F
220 MARK 1,10240
230 GO TO 260
240 PRINT "ENTER THE SCRATCH FILE NUMBER ";
250 INPUT F
260 FIND F
270 PRINT @41:" CHART OF ACCOUNTS FILE NUMBER : ",F
280 PRINT @41:""
290 B=0
300 Y=0
310 M=0
320 S$=""
330 LET T1=0
340 LET T2=0
350 LET T3=0
360 LET M=0
370 PRINT @41: USING """ACCT:CODE:     TITLE"",32T,""BUDGET"",S":
380 PRINT @41: USING " 8T,""YEAR-TO-DATE"",23T,""CURRENT MO.""":
390 PRINT "ACCOUNT(3DIGIT)              :";
400 INPUT A
410 IF A=0 THEN 880
420 PRINT "ENTER ACTION CODE(T,S,OR +):";
430 INPUT C$
440 IF C$<>"T" AND C$<>"S" AND C$<>"+" THEN 420
450 IF C$="S" THEN 730
460 PRINT "TITLE(20 CHARS MAX)          :";
470 INPUT T$
480 IF C$="T" THEN 570
490 PRINT "BUDGET AMOUNT                :";
500 INPUT B
510 PRINT "YEAR-TO-DATE AMOUNT          :";
520 INPUT Y
530 PRINT "ENTRIES O.K.?                :";
540 INPUT K$
550 IF K$="NO" THEN 390
560 REM*****************************************
570 WRITE A,C$,T$,B,Y,M
580 PAGE
590 IF C$="T" THEN 690
600 IF C$="S" THEN 730
610 REM**************SUMS**********************
620 T1=T1+B
630 T2=T2+Y
640 T3=T3+M
650 PRINT @41: USING 660:A,C$,T$,B,Y,M
660 IMAGE 3D,7T,1A,9T,20A,30T,"$",4D.2D,48T,"$",4D.2D,63T,"$",4D.2D
670 GO TO 390
680 REM************ PRINT ******************
```

```
690 PRINT @41: USING "/,3D,7T,1A,10T,20A":A,C$,T$
700 S$="TOTAL "&T$
710 GO TO 390
720 REM************ TOTAL ********************
730 PRINT @41: USING 660:A,C$,S$,T1,T2,T3
740 PRINT @41:""
750 WRITE A,C$,T$,T1,T2,T3
760 S$=""
770 PAGE
780 E1=T1
790 E2=T2
800 E3=T3
810 T1=0
820 T2=0
830 T3=0
840 PRINT "MORE TO COME? ";
850 INPUT Z$
860 IF Z$="YES" OR Z$="Y" THEN 390
870 REM***************** WHOA ******************
880 END
```

```
100 REM****************************************
110 REM       PROCESS TRANSACTIONS
120 REM****************************************
130 DIM C$(1),T$(20),D$(5),K$(3),U$(20),Y$(3),X$(1),Z$(20),V$(10)
140 PAGE
150 PRINT "ENTER MONTH:  ";
160 INPUT V$
170 PRINT @41: USING "25T,""MONTH OF "",10A,2/,":V$
180 PRINT "ENTER CHART OF ACCOUNTS FILE #: ";
190 INPUT F
200 PRINT @41:"CHART OF ACCOUNTS FOR FILE NUMBER    :";F
210 PRINT
220 PRINT "BE CAREFUL IN ENTERING OUTPUT CHART OF ACCOUNT FILE"
230 PRINT
240 PRINT "IF YOU WANT TO CREATE A NEW OUTPUT FILE,"
250 PRINT "     ENTER - YES OR JUST Y -    "
260 INPUT Y$
270 IF Y$<>"YES" AND Y$<>"Y" THEN 330
280 PRINT "ENTER THE ""LAST"" FILE NUMBER (IN TLIST)  ";
290 INPUT E
300 FIND E
310 MARK 1,2000
320 GO TO 350
330 PRINT "ENTER THE SCRATCH FILE NUMBER FOR OUTPUT ";
340 INPUT E
350 FIND E
360 PRINT @41:"OUTPUT CHART OF ACCOUNTS FILE NUMBER :";E
370 PRINT @41: USING "2/":
380 PRINT @41: USING """ACCT:CODE     TITLE"",32T,""BUDGET"",S":
390 PRINT @41: USING "8T,""YEAR-TO-DATE"",23T,""CURRENT MO.""":
400 S2=0
410 S3=0
420 T1=0
430 T2=0
440 T3=0
450 E1=0
460 E2=0
470 E3=0
480 W=0
490 R=0
500 REM***************** LOOP **********************
510 R=R+1
520 C=0
530 FIND F
540 FOR I=1 TO R
550 READ @33:A,C$,T$,B,Y,M
560 NEXT I
570 S2=Y
580 S3=0
590 REM*****************************
600 IF C$="T" THEN 760
610 IF C$="S" THEN 790
620 PRINT @41: USING 810:A,C$,T$,B,Y,M
630 REM*************************
640 PAGE
650 IF C<>0 THEN 690
660 PRINT "DO YOU HAVE ANY  ";A;" ACCOUNTS THIS MONTH? ";
670 C=1
680 GO TO 700
```

```
690 PRINT "DO YOU HAVE MORE ";A;" ACCOUNTS TO ENTER? ";
700 INPUT Y$
710 IF Y$="YES" OR Y$="Y" THEN 970
720 PAGE
730 IF Y$="NO" OR Y$="N" THEN 1170
740 GO TO 640
750 REM***************** PRINT ********************
760 PRINT @41: USING "/,3D,7T,1A,9T,20A":A,C$,T$
770 S$=" TOTAL "&T$
780 GO TO 1170
790 REM:***************** TOTAL ***************
800 PRINT @41: USING 810:A,C$,S$,T1,T2,T3
810 IMAGE 3D,7T,1A,9T,20A,30T,"$",4D.2D,48T,"$",4D.2D,63T,"$",4D.2D
820 FIND E
830 FOR I=1 TO R-1
840 READ @33:X1,X$,Z$,X2,X3,X4
850 NEXT I
860 WRITE @33:A,C$,T$,T1,T2,T3
870 PRINT @41: USING "2/":
880 IF W=1 THEN 1340
890 W=1
900 E1=T1
910 E2=T2
920 E3=T3
930 T1=0
940 T2=0
950 T3=0
960 GO TO 510
970 REM*********************************
980 A1=A
990 PRINT "ACCOUNT(3DIGIT)                    :";A1
1000 PRINT "DATE(MM/DD)                       :";
1010 INPUT D$
1020 PRINT "CHECK NUMBER(3 DIGITS)            :";
1030 INPUT K$
1040 PRINT "DESCRIPTION(20 CHARS MAX)         :";
1050 INPUT U$
1060 PRINT "AMOUNT                            :";
1070 INPUT M
1080 PRINT "ENTRIES, OK? ";
1090 INPUT Y$
1100 IF Y$<>"NO" AND Y$<>"N" THEN 1120
1110 GO TO 990
1120 S2=S2+M
1130 S3=S3+M
1140 PRINT @41: USING 1150:A1,D$,K$,U$,M
1150 IMAGE 3D,8T,5A,15T,3A,20T,20A,63T,"$",4D.2D
1160 GO TO 640
1170 REM***   UPDATE CHART OF ACCOUNTS **************
1180 FIND E
1190 IF R=1 THEN 1230
1200 FOR I=1 TO R-1
1210 READ @33:X1,X$,Z$,X2,X3,X4
1220 NEXT I
1230 WRITE @33:A,C$,T$,B,S2,S3
1240 IF C$<>"+" THEN 510
1250 PRINT @41:"-----------------------------------------------------";
1260 PRINT @41:"------------------"
1270 PRINT @41: USING 810:A,C$,T$,B,S2,S3
```

```
1280 PRINT @41: USING "2/":
1290 T1=T1+B
1300 T2=T2+S2
1310 T3=T3+S3
1320 C=0
1330 GO TO 510
1340 REM************* END LOOP ************************
1350 T1=E1-T1
1360 T2=E2-T2
1370 T3=E3-T3
1380 PRINT @41: USING 1400:T1
1390 PRINT @41: USING 1410:T2,T3
1400 IMAGE 10T,"INC-EXPENSES",30T,"$",-4D.2D,48T,S
1410 IMAGE"$",-4D.2D,15T,"$",-4D.2D
1420 CLOSE
1430 PRINT "THANK YOU !!"
1440 END
```

```
100 REM************************************************************
110 REM THIS PROGRAM DRAWS A PIE GRAPH
120 REM   ACCORDING TO THE FILE MADE BY THE PROGRAM
130 REM   "CHART OF ACCOUNT" AND/OR "PROCESS TRANSACTIONS"
140 REM************************************************************
150 SET DEGREES
160 DIM E(20)
170 PAGE
180 PRINT "TYPE THE NUMBER OF YOUR FILE CONTAING THE DATA  ";
190 INPUT H
200 FIND H
210 PAGE
220 PRINT "YOU ARE GOING TO SELECT THE ITEM YOU WANT TO DRAW"
230 PRINT
240 PRINT "TYPE 1 FOR INCOME BUDGET, TYPE 2 FOR INCOME YEAR-TO-DATE"
250 PRINT "TYPE 3 FOR INCOME CURRENT MONTH"
260 PRINT "TYPE 4 FOR EXPENSE BUDGET,TYPE 5 FOR EXPENSE YEAR-TO-DATE"
270 PRINT "TYPE 6 FOR EXPENSE CURRENT MONTH"
280 INPUT X
290 GO TO X OF 360,360,360,330,330,330
300 PRINT "TYPE THE RIGHT NUMBER, OK?"
310 GO TO 230
320 REM**************GET THE SECOND SET IN THE FILE***************
330 READ @33:A,C$,T$,B,Y,N
340 IF C$<>"S" THEN 330
350 X=X-3
360 I=0
370 READ @33:A,C$,T$,B,Y,N
380 IF X=1 THEN 430
390 IF X=2 THEN 420
400 B=N
410 GO TO 430
420 B=Y
430 IF C$="T" THEN 770
440 IF C$="+" THEN 920
450 IF C$<>"S" THEN 1070
460 R=10
470 GOSUB 970
480 L=0
490 FOR G=1 TO K
500 L=L+E(G)
510 M=L/B*360
520 REM**************DRAW THE PICTURE*****************************
530 DRAW R*COS(M),R*SIN(M)
540 D1=(L-E(G)/2)/B*360
550 REM****************ARRANGE THE POSITION FOR TITLE***********
560 IF D1<90 OR D1>270 THEN 660
570 IF E(G)/B*360<60 THEN 610
580 IF D1>125 AND D1<225 THEN 640
590 MOVE R/2*COS(D1)-5,R/2*SIN(D1)
600 GO TO 670
610 IF D1>115 AND D1<225 THEN 640
620 MOVE 3*R/4*COS(D1)-5,3*R/4*SIN(D1)-1
630 GO TO 670
640 MOVE R*COS(D1)-5,R*SIN(D1)
650 GO TO 670
660 MOVE R/2*COS(D1),R/2*SIN(D1)
670 PRINT USING "2D.2D""% FOR ITEM""2D":E(G)/B*100,G
680 MOVE 0,0
690 NEXT G
```

```
700 PRINT "HJJJJJJJJJJJJ"
710 REM*********************************************************
720 PRINT "DO YOU WANT TO TRY ANOTHER GRAPH? ";
730 INPUT Q$
740 IF Q$<>"NO" AND Q$<>"N" THEN 200
750 GO TO 1080
760 REM**************ARRANGE THE AREA FOR DRAWING*****************
770 PAGE
780 PRINT USING 790:T$
790 IMAGE 12T,15A,/,10T,10("*")
800 PRINT
810 IF X=1 THEN 870
820 IF X=2 THEN 850
830 P$="CURRENT MONTH"
840 GO TO 880
850 P$="YEAR-TO-DATE"
860 GO TO 880
870 P$="BUDGET"
880 PRINT P$
890 PRINT
900 GO TO 370
910 REM**************STORE THE DATA TO BE DRAWN*******************
920 I=I+1
930 PRINT USING "2D,""."",20A,$FD.FD":I,T$,B
940 E(I)=B
950 K=I
960 GO TO 370
970 VIEWPORT 55,115,15,75
980 REM**************DRAW A CIRCLE*********************************
990 REM*************IF YOU WANT TO DO FASTER, INCREASE THE STEP****
1000 WINDOW -R,R,-R,R
1010 MOVE R,0
1020 FOR J=5 TO 360 STEP 5
1030 DRAW R*COS(J),R*SIN(J)
1040 NEXT J
1050 DRAW 0,0
1060 RETURN
1070 PRINT "ERROR IN YOUR FILE"
1080 PRINT "THANK YOU! THIS IS THE END OF YOUR RUN"
1090 END
```

Mailing List Program

The mailing list program comes from a problem which required a large list of names and addresses to be sorted by zip code. The program has several versions. One version uses a selection sort with arrays in memory. Another version uses a merge/sort on arrays in memory. The last version uses a merge/sort on mass storage files. (The listing is not included to save space).

Database Retrieval Program

The database retrieval program is another typical data processing application which involves the creation, update, and inquiry of a database. The program consists of two segments. The first segment is used to create the database. The second segment is used to allow additions, updates, or inquiries-by-attribute to be made of the information in the database. Index files are used to access the records in the database. The indexes store information about the existance and location of records stored in the database. A system of chains is used with a hashing function based on multiple valued attributes. Since the program and method of access is unique, the listing is included, here.

```
5 INIT
7 PAGE
10 REM ************************************************************
11 REM                  REAL ESTATE SYSTEM
12 REM
13 REM ************************************************************
15 DIM F(50),I(50),K(50),Z(50)
20 PRINT "CENTURY 21 REAL ESTATE SYSTEM"
25 PRINT "ENTER FILE NUMBER : ";
27 INPUT F4
28 IF F4<>6 THEN 25
30 PRINT "ENTER NAME OF MLS FILE : ";
32 INPUT E$
35 REM ------------------------------------------------------------
40 FIND F4
42 MARK 1,10000
44 FIND F4+1
45 MARK 1,50000
47 FIND F4+2
48 MARK 1,10000
49 FIND F4
50 PRINT @33:E$
60 PRINT "INPUT SIZE OF DIRECTORY : ";
62 INPUT N
65 LET W=0
69 REM ------------------------------------------------------------
70 PRINT @33:N
71 PRINT @33:"#"
72 PRINT @33:0
80 FOR J=1 TO N
90 LET F(J)=-1
92 LET I(J)=0
95 LET K(J)=-1
100 PRINT @33:F(J)
101 PRINT @33:"F"
102 PRINT @33:I(J)
103 PRINT @33:"I"
104 PRINT @33:K(J)
110 NEXT J
111 CLOSE
112 FIND F4+2
113 FOR J=1 TO N
114 LET Z(J)=0
115 PRINT @33:Z(J)
116 NEXT J
117 CLOSE
119 REM ------------------------------------------------------------
120 PRINT "MLS DATABASE PREPARED"
125 DELETE 5,120
130 FIND 5
140 OLD
150 RUN
```

```
4 PAGE
5 LET D=0
10 REM ************************************************************
11 REM              MLS SUBSYSTEM
12 REM
13 REM ************************************************************
15 DIM F(50),I(50),K(50),Z(50)
16 FOR J=1 TO 50
17 Z(J)=0
18 NEXT J
19 PAGE
20 PRINT "CENTURY 21 REAL ESTATE SYSTEM"
25 PRINT "ENTER FILE NUMBER : ";
27 INPUT F4
30 PRINT "ENTER NAME OF MLS FILE : ";
32 INPUT E$
40 FIND F4
50 INPUT @33:G$
60 IF G$=E$ THEN 90
70 PRINT "IMPROPER ACCESS, TRY AGAIN"
80 GO TO 20
90 INPUT @33:N,X$,V1
95 ON EOF (0) THEN 121
100 FOR J=1 TO N
110 INPUT @33:F(J),X$,I(J),X$,K(J)
115 NEXT J
121 FIND F4+2
122 FOR J=1 TO V1
123 INPUT @33:Z(J)
124 NEXT J
130 FIND F4
135 REM ----------------------LOOP-------------------------------
137 PAGE
138 PRINT
139 PRINT
140 PRINT "SELECT ONE OF THE FOLLOWING :"
141 PRINT "1. LOOK-UP MLS PROPERTY"
142 PRINT "2. ADD AN MLS PROPERTY"
144 PRINT "3. STOP"
145 PRINT "ENTER 1, 2, OR 3 :";
146 INPUT C
147 IF C=3 THEN 1000
149 IF C=2 THEN 200
150 IF C<>1 THEN 140
151 REM ----------------DO ONE OF THESE--------------------------
152 REM ------LOOK UP-----------------------------------------
155 GOSUB 6000
160 LET F1=-1
165 GOSUB 5000
170 IF R<>0 THEN 178
174 PRINT "PROPERTY NOT FOUND"
176 GO TO 140
178 ON EOF (0) THEN 185
179 LET K1=K(R)
180 FIND F4+1
181 FOR J=1 TO K1
182 INPUT @33:M$,C$,L$,B$,A$,S$,Z$
183 NEXT J
184 LET J=J-1
185 GOSUB 2000
186 LET K1=Z(J)
188 IF K1<=0 THEN 140
189 GO TO 180
```

```
190 REM ---------ADD A PROPERTY-----------------------------------
200 GOSUB 6000
210 GOSUB 7000
212 LET F1=1
215 GOSUB 5000
220 IF R=0 THEN 275
222 REM ------------------------------------------------------------
223 ON EOF (0) THEN 230
224 LET V1=V1+1
226 FIND F4+1
227 IF V1=1 THEN 230
228 INPUT @33:Q$,Q$,Q$,Q$,Q$,Q$,Q$
229 GO TO 228
230 PRINT @33:M$
231 PRINT @33:C$
232 PRINT @33:L$
233 PRINT @33:B$
234 PRINT @33:A$
235 PRINT @33:S$
236 PRINT @33:Z$
237 Z(V1)=-1
239 IF I(R)=I1 THEN 246
240 LET K(R)=V1
242 LET I(R)=I1
243 LET F(R)=1
245 GO TO 140
246 Z9=K(R)
250 FOR J=1 TO V1
252 IF Z(Z9)<0 THEN 255
253 Z9=Z(Z9)
254 NEXT J
255 Z(Z9)=V1
271 GO TO 140
274 REM --------------------DIRECTORY FULL-------------------------
275 PRINT "DIRECTORY FULL...REORGANIZE"
280 GO TO 140
400 REM ------------------------------------------------------------
1000 FIND F4
1005 PRINT @33:E$
1010 PRINT @33:N
1011 PRINT @33:"#"
1012 PRINT @33:V1
1020 FOR J=1 TO N
1021 PRINT @33:F(J)
1022 PRINT @33:"F"
1023 PRINT @33:I(J)
1024 PRINT @33:"I"
1025 PRINT @33:K(J)
1028 NEXT J
1030 CLOSE
1031 FIND F4+2
1032 FOR J=1 TO V1
1033 PRINT @33:Z(J)
1034 NEXT J
1035 CLOSE
1036 PRINT "MLS SYSTEM ENDED "
1040 GO TO 7090
```

```
2000 REM ********************
2001 REM  PRINT MLS LIST
2002 REM ********************
2003 PRINT
2004 PRINT
2010 PRINT "MLS PROPERTY : ",M$
2020 PRINT "PRICE      $",C$
2030 PRINT "LOCATION     :",L$
2040 PRINT "#BEDROOMS    :",B$," AND BATHS : ",A$
2050 PRINT "SALES STATUS :",S$
2060 PRINT "COMMENTS :",Z$
2070 REM ----------------------
2071 PRINT
2072 PRINT
2080 RETURN
5000 REM ************************************************************
5001 REM              HASHED-ARRAY SEARCH SUBPROGRAM
5002 REM
5003 REM ************************************************************
5010 LET I1=6*C+3*L+B-10
5020 LET Q1=INT(I1/N)
5022 LET R1=INT(I1-Q1*N)
5030 IF Q1>0 THEN 5050
5040 LET Q1=1
5050 LET R=R1+1
5055 LET Q=Q1
5060 REM *****************REPEAT************************************
5070 FOR J=1 TO N-1
5075 IF F(R)>0 THEN 5100
5080 IF F1>0 THEN 5500
5090 LET R=0
5095 GO TO 5500
5100 REM ---------------3.2.2-----------------------------------
5105 IF I(R)=I1 THEN 5500
5110 LET R=R+Q1
5112 LET Q=INT(R/N)
5115 LET R1=INT(R-Q*N)
5117 LET R=R1+1
5200 NEXT J
5300 REM ------------------END LOOP-------------------------------
5310 LET R=0
5500 RETURN
6000 REM ***********************
6001 REM  INPUT ATTRIBUTES
6002 REM ***********************
6003 PAGE
6010 PRINT "ASKING PRICE :"
6020 PRINT "1. $10,000 - 19,999"
6030 PRINT "2. $20,000 - 29,999"
6040 PRINT "3. $30,000 - 39,999"
6050 PRINT "4. MORE THAN $40,000"
6060 PRINT "ENTER 1,2,3 OR 4 : ";
6070 INPUT C
6110 REM
6120 PRINT "LOCATION :"
6130 PRINT "1. THIS SIDE OF TRACKS"
6140 PRINT "2. THAT SIDE OF TRACKS"
6150 PRINT "ENTER 1 OR 2 : ";
6160 INPUT L
6170 REM
6210 PRINT "HOW MANY BEDROOMS :"
6220 PRINT "1. 2 BEDROOMS"
6230 PRINT "2. 3 BEDROOMS"
```

```
6260 INPUT B
6270 REM
6280 PAGE
6300 RETURN


7000 REM **************************
7001 REM     INPUT MLS RECORDS
7002 REM **************************
7003 PAGE
7010 PRINT "MLS PROPERTY DETAILS"
7020 PRINT "MLS NUMBER        : ";
7022 INPUT M$
7025 PRINT "ASKING PRICE      $ ";
7027 INPUT C$
7030 PRINT "LOCATION          : ";
7032 INPUT L$
7035 PRINT "NUMBER BEDROOMS   : ";
7037 INPUT B$
7040 PRINT "NUMBER BATHS      : ";
7042 INPUT A$
7045 PRINT "SALES STATUS      : ";
7047 INPUT S$
7050 PRINT "COMMENTS          : ";
7055 INPUT Z$
7060 REM
7080 RETURN
7090 END
```

MLS PROPERTY : 945
PRICE $ 38,595
LOCATION : 15938 S.W. TRUMAN RD.
#BEDROOMS : THREE AND BATHS : 1 1/2
SALES STATUS : EARNEST MONEY
COMMENTS : CORNER LOT WITH TREES

MLS PROPERTY : 328
PRICE $ 49,995
LOCATION : 1825 OAK BLVD
#BEDROOMS : FOUR AND BATHS : 3
SALES STATUS : SOLD
COMMENTS : FINANCING AVAILABLE

MLS PROPERTY : 1109
PRICE $ 29,800
LOCATION : 5220 RAILROAD ST.
#BEDROOMS : TWO AND BATHS : 2
SALES STATUS : FOR SALE
COMMENTS : UNFINISHED FAMILY ROOM

RESULTS OF THE BENCHMARKS

In the process of converting and running the test programs listed above, several strong and weak points of the 4051 were noted. While all of the application programs were able to be converted and run, some features had to be eliminated where the application depended upon a disk based computer system.

The 4051 is easy to start up. The user is only required to use one on-off switch and the operating system does not have to be loaded to be used. An auto load feature allows an inexperienced user to place a tape in in the slot and press one button to execute a canned program. Ability to control the graphics display from the program allows menus to be displayed for a user to make a selection and then have the screen automatically erased to avoid cluttering of information and avoid having the user worry about the page control. These features make the 4051 very good for a first time business user who wants a system that requires as little knowledge as possible to operate.

While entering programs or data the backspace key, the clear key, and the rubout key allow the user to modify entries easily. However once a program is created, the editing is limited to the line editor keys. A user is thus unable to search and replace characters and the insert and delete operations require using BASIC line numbers or

the DELETE command of the operating system. While programs can be modified to the limited degree listed above, data and text files cannot be easily changed, therefore hindering program debugging.

The immediate feedback of the interpretive system which detects syntax errors aids greatly in program creation on the 4051. However error messages generated at run-time are often difficult to understand to a programmer and even more confusing to an inexperienced or first time user. While the ON THEN statement allows some error recovery to be made in application programs, writing error trap and restart routines is difficult due to a lack of convenient instructions. In order for a non-programmer to use a data processing system with little difficulty, error recovery is required. A non-programmer half way through a data entry procedure cannot recover the operation without assistance if a system error should arise or he should accidently press the wrong key (especially the break key which is located dangerously close to the return key). Also errors such as a system error 0 which destroys the contents of memory should allow some form of recovery, instead.

The graphics display is probably one of the most useful features that sets the 4051 apart from other systems. Using graphics, data may be compacted into a form that is easily visualized by users of the system. Comparisons and trend analysis can be done much more efficiently with the graphics capability. The graphics

output may be sent to peripheral devices such as plotters and hardcopy units which avoids the need to wait for data to be plotted by hand or by a service group external to the user who wants the information. The resolution of the graphics output is excellent. While having many advantages, the storage scope on the 4051 has some important limitations. Color cannot be shown on the screen and the nature of the screen makes it difficult to read in brightly lit rooms as is the case in many business offices. Movement, scrolling, and selective erasing are also graphics features that may be desirable in some business data processing applications, but are not available on the storage scope of the 4051.

The manuals are well written and complete. However, there is a considerable amount of redundancy which adds to both the voluminous nature of the manuals and the confusion of the user who is often not sure which manual to consult or how many of the manuals contain the needed answer. Although all of the manuals contain examples, they are of little help to the first time business user who only wants to solve his application and not learn everything there is to to know about the 4051 Graphics Terminal in the process. Additional comments about the manuals are given in the next section of this report.

The most severe limitation found on the 4051 Graphics System is the reliance of the system (and the BASIC commands) on the use of the magnetic tape drive for mass storage of

data and programs. The tape commands and medium are easy to use for some data processing applications, but not all applications in business data processing are well suited to sequential processing.

In the test programs which were used for this report, the accounts receivable program and the database retrieval program required the capability of rewriting records in order to do updating of the files. To simulate these updating functions on a sequential tape system requires either rewriting of the updated files or the use of index files which can be changed more easily than the master file, and then rewritten.

In doing file-to-file interactions the 4051 tape system requires that each file read or written be accessed from the start each time an access occurs. The user must therefore find the first record in a file for each access and then do dummy reads until the desired record in the file is found. On large data files, it is very time consuming to do random inquiry on a sequential file.

Also it is almost impossible and impractical to do a sort/merge operation on data stored on a sequential tape. Other limitations noted with the tape usage were the inability to erase a file in the middle of a tape cartridge, and the lack of good security commands for the tape files. The MARK statement which destroys the contents of a file can only be used at the end of the current list of files,

If a user inadvertently uses MARK in the middle of a tape, all files following the file which is marked are effectively lost. This feature of the tape system greatly inhibits rewriting of data files for application programs.

The SECRET command allows a file to be protected, but the contents of the file can never be modified or listed. The only other means for protecting tape files is the mechanical device on the tapes themselves. With such limited security provisions, it is difficult to write application programs that prevent users from accidently destroying a vital file containing either programs or data. The limited flexibility of the SECRET command makes it almost useless in practice to enforce file security.

Other limitations noted in the system are mentioned below. When using the tape, strings can be written with a single PRINT statement, but if a single INPUT statement is used the entire record is read into the first string variable on the list. This limitation requires that information in string variables be printed each with a separate PRINT statement in order to be read in separately again. Also string arrays are not allowed which restricts the amount of string data that can be stored and manipulated in memory.

Composing more than one BASIC statement on a line is not allowed. Thus a lot of space is wasted to store one BASIC command per line.

Finally, in order to be used as a business application machine, the 4051 should have an optional data processing ROM which contains operations for file manipulation, sorting and merging, and other data file utility programs. Such an addition to the 4051 would help to decrease the overhead needed in programs which perform common data processing tasks with files.

HOW EFFECTIVE ARE THE 4051 DOCUMENTS AND MANUALS?

PHYSICAL: The manuals are bulky, heavy and ugly. The available page area should be better utilized. Illustrations can be reduced in size without giving a cramped or cluttered appearence.

ORGANIZATION & GENERAL INPRESSIONS: The manuals as such are very well written. Example programs and a generous number of illustrations shortens the learning time considerably. However because of the number of manuals for the system it becomes very difficult to figure out which manual to look into for a particular piece of information. At times to find the correct information the searching involves going through two or more manuals. This piecing together of information from two or more manuals is difficult for the user.

The manuals are also very "extensive". By just reorganizing some parts of the manuals, a pretty good textbook on BASIC can be made. A lot of unnecessary detail and description is provided for each BASIC statement which can be avoided. On the whole the manuals can be combined and condensed, say, into the following possible sequence:

POSSIBLE SEQUENCE

a) Quick reference manual with commands and procedures that are commonly used on the system. This manual has sufficient information in a concise manner for a user to sit at the machine, switch it on (where

is the on/off switch!), and enter, run, edit and create a file of a program.

b) BASIC language and system reference manual for users who need more information on the system, language and the commands.

c) PLOT 50 manual with example programs to acquaint new users with the system and suggest applications that a new user might not think about.

Detailed corrections to the manuals can be found in the Appendix section of this report.

HOW WELL DOES THE HARDWARE AND SYSTEM SOFTWARE PERFORM IN A "TYPICAL" DATA PROCESSING APPLICATION?

To answer this question we will look at three areas of performance:

1) Language/ Processor speed

2) Processor/ peripheral speed

3) Special Features

In each of these areas two similar systems, Datapoint 1100 and Wang 2200-T, were used as a basis for analyzing the performance of the Tektronix 4051. All three systems were programmed in BASIC and programs were written as closely identical as possible in order to give an accurate comparison between systems.

Our aim is to illustrate how the Tektronix 4051 competes along side of other similar systems, rather than the development of precise timing data.

1) Language/ processor speed

To evaluate this area of performance a series of benchmark programs were written for all three systems. The areas measured are

a) Empty Loop Performance

   (1) Single loop
   (2) nested loop

b) Calculation loop Performance

   (1) Assignment Statement
   (2) Add and Subtract
   (3) Multiply
   (4) Divide

(5) False "If"
(6) True "If"

The programs were executed for 100, 1000, 10000 iterations and the times compared for each machine.

In all cases the Tektronix 4051 performance figures fell between the Wangco and Datapoint systems. The Wangco was always the fastest of the three systems but the time margin between the Wangco and Tektronix 4051 does not appear to be significant. The Tektronix was always faster than the Datapoint system by a significant margin.

2) Processor/ peripheral speed

Again a series of benchmark programs were written for all three systems. In this area we measured the speed of the following:

(1) Tape speed of the Tektronix 4051 compared to the Disk speed of the Wangco and Datapoint Systems.

    (a) Sequential Read/ Write
    (b) Merge/sort on Auxillary Storage
    (c) Sequential Update of a file on Auxillary Storage

(2) Display speed and capacity

The Tektronix 4051 tape system compared very unfavorably with the disk based systems when two or more files must be alternately accessed, or random accesses are required. In both the merge/sort test and sequential update test the performance of the Tektronix tape system significantly increased the required processing time.

Also the programming became significantly more complex and inefficient on the Tektronix system when two or more files were involved. Efforts to simulate random accesses to tape records and update entries were hopeless.

Comparison of display speed revealed the Tektronix system to be slower than the Wangco. But this time margin is not felt to be significant since no detectable wait is experienced by the user. In other words, information is displayed at least as fast as the user can absorb it.

A number of conclusions can be obtained by examining the differences in measured performance. First, the arithmetic capability of the 6800 used to support the Tektronix 4051 is roughly equivalent to the arithmetic capability of the Wang processor. The Datapoint processor sacrificed speed for control of arithmetic round-off and file record uniformity. All Datapoint calculations are performed as string (character) manipulations using ACSII encoded numerals. As opposed to the Wang and Tektronix machines, the Datapoint machine requires no internal conversion or additional record formats for binary versus character storage (all files can be interrogated as an aid to debugging).

The Tektronix divide algorithm is slower than expected. This is explained if the Wang system includes a hardware divide and the 4051 uses a firmware divide. The team was not able to verify which divide algorithm is used by the Wang system.

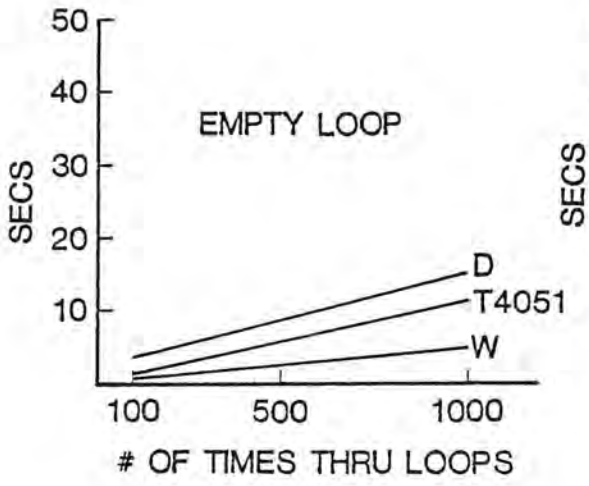
The implementation of BASIC appears to be much more

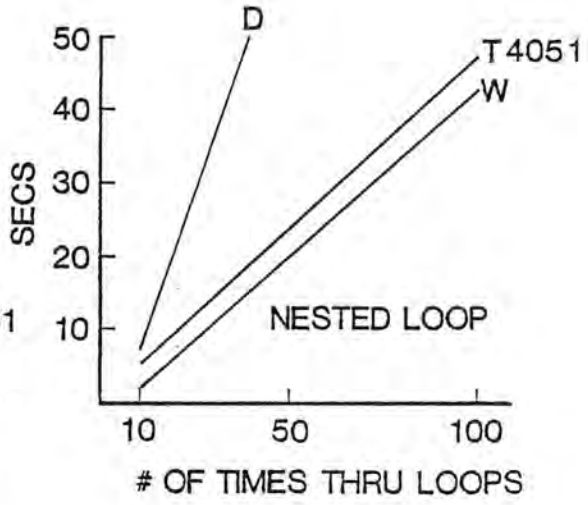
efficient in the Wang system than in the other two systems. Loop, PRINT, IF, and other BASIC statements are performed much more quickly in the Wang than in the other systems.

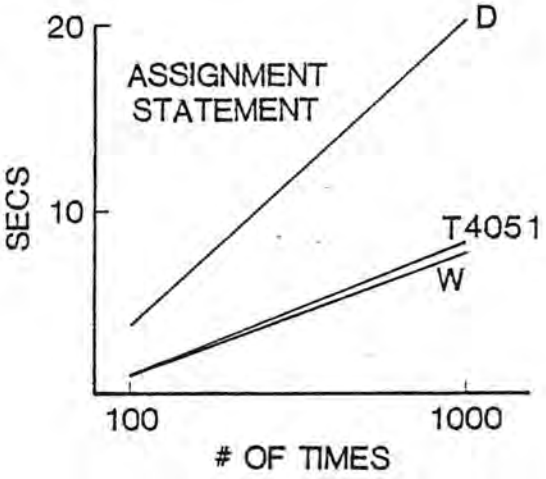
The performance of I/O to mass storage reveals several important subsystem features. First, tape systems can compete only in storing program files that are read sequentially. Nearly all data files destroy the usefulness of tape.

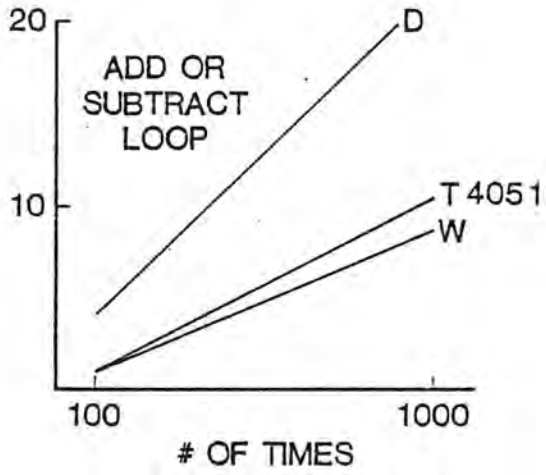
The Datapoint diskette system performs slower than the Wang diskette system, even though they both use the same diskette system supplier. Why is there a difference, then? The Datapoint system includes dynamic space allocation that automatically allocates more space to a growing file. The Wang system forces the programmer to provide space allocation. Diskette access times for the Datapoint were much faster when repeating the benchmarks on an allocated file. The lesson learned is that automatic allocation costs in system performance.

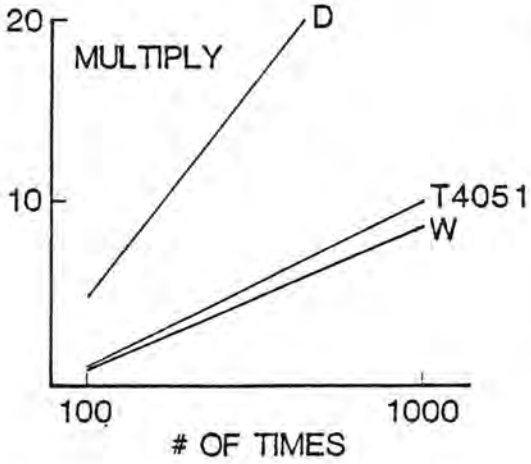
The Tektronix 4051 can be vastly improved in its ability to handle files of data if a diskette mass storage device is added. The tape system will remain as a valuable storage device for programs, however.

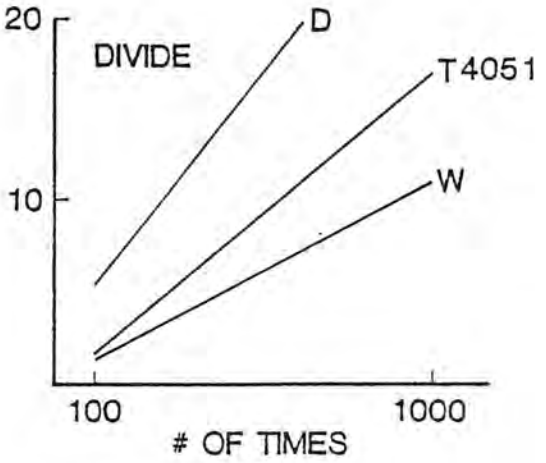

50
40
30
20
10
SECS
EMPTY LOOP
D
T4051
W
100
500
1000
OF TIMES THRU LOOPS

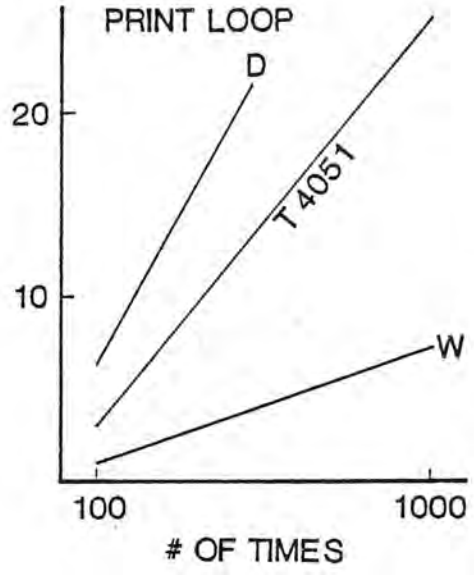

50
40
30
20
10
SECS
D
T4051
W
NESTED LOOP
10
50
100
OF TIMES THRU LOOPS

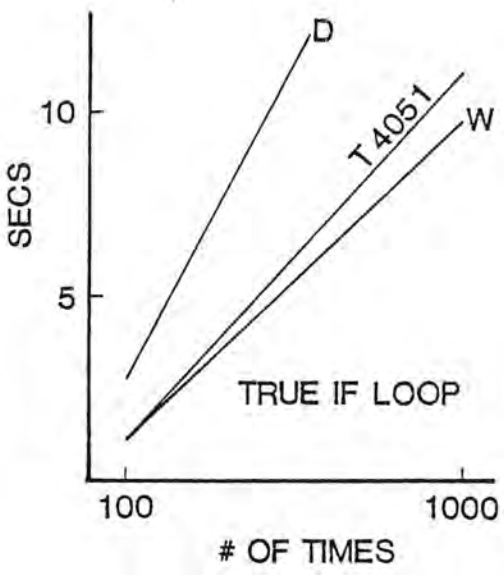

20
10
SECS
ASSIGNMENT
STATEMENT
D
T4051
W
100
1000
OF TIMES

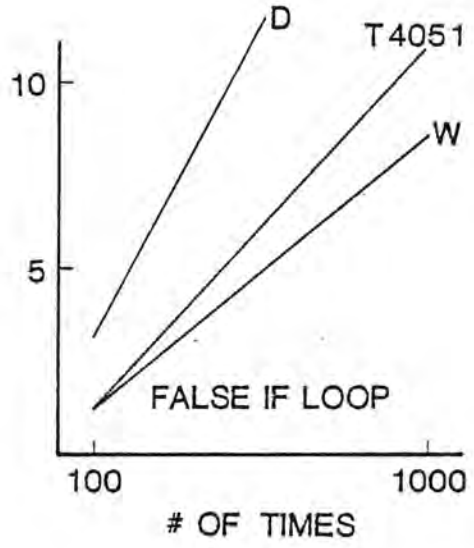

20
10
ADD OR
SUBTRACT
LOOP
D
T4051
W
100
1000
OF TIMES

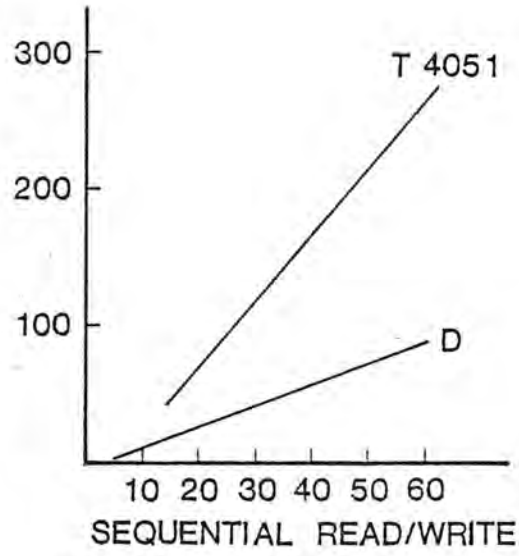

20
10
MULTIPLY
D
T4051
W
100
1000
OF TIMES


20
10
DIVIDE
D
T4051
W
100
1000
OF TIMES

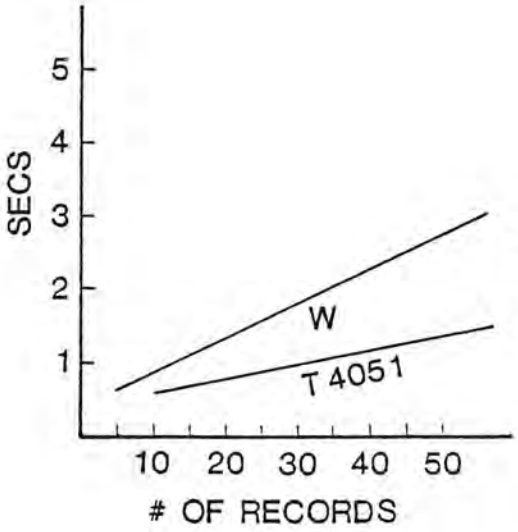


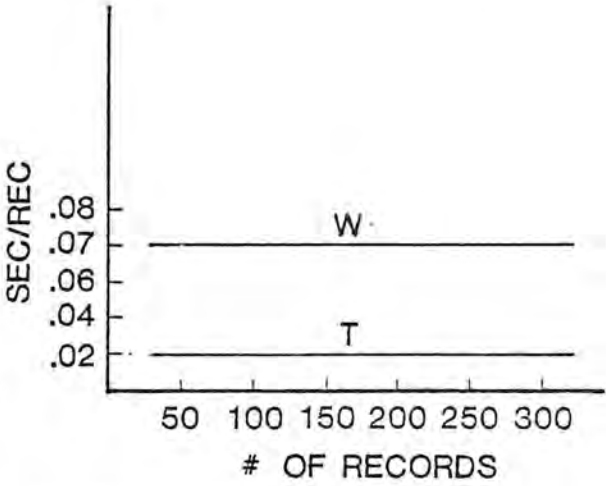




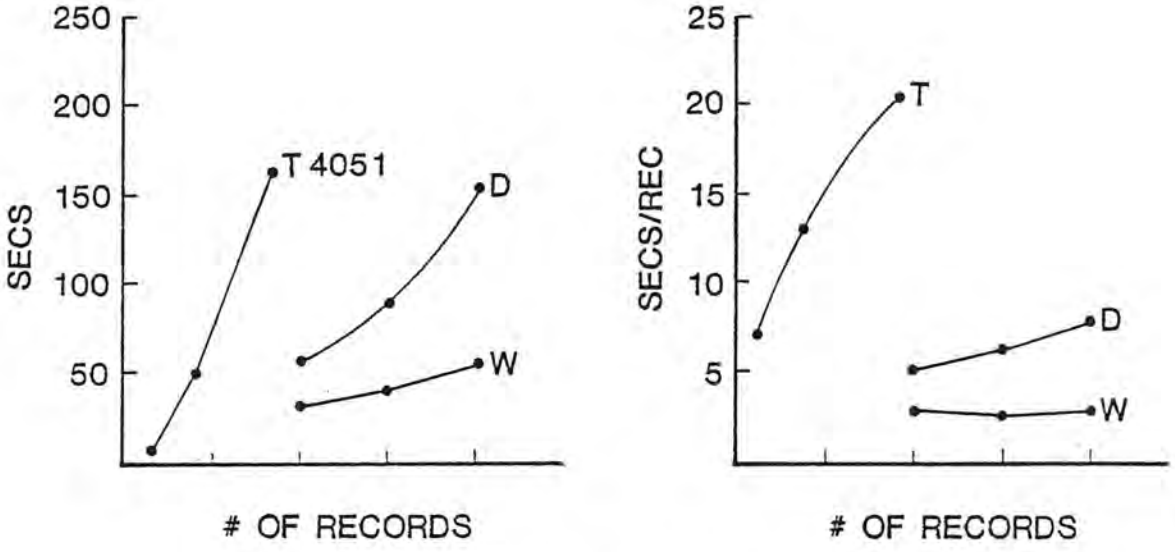


READ ONE FILE-UPDATE RECORD-
WRITE ON ANOTHER FILE

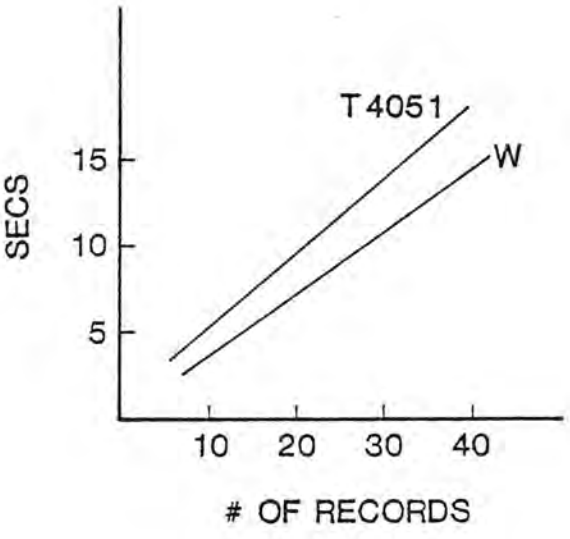




SEQUENTIAL READ/WRITE DISK VS TAPE



MERGE/SORT ON AUXILLARY STORAGE



MERGE/SORT OF ARRAY

```
100 REM  ------TEK 4051 SPEED TEST-----
110 REM  ---------EMPTY  LOOP----------
120 PRINT "ENTER NUMBER TIMES THROUGH LOOP: ";
130 INPUT N
135 PRINT "G"
140 FOR I=1 TO N
150 NEXT I
155 PRINT "G"
160 END
170 REM  0.5 SECONDS WHEN N=100
180 REM  5.0 SECONDS WHEN N=1000
190 REM 45.0 SECONDS WHEN N=10000


100 REM  -----TEK 4051 SPEED TEST-----
110 REM  ------NESTED  LOOP-----------
120 PRINT "NUMBER TIMES THROUGH LOOPS: "
130 INPUT N
135 PRINT "G"
140 FOR I=1 TO N
150 FOR J=1 TO N
160 NEXT J
170 NEXT I
180 PRINT "G"
190 END
200 REM 0.7 SECONDS WHEN N=10
210 REM 46 SECONDS WHEN N=100


100 REM  -----TEK 4051 SPEED TEST-----
110 REM  -------ASSIGNMENT LOOP-------
120 PRINT "NUMBER OF TIMES THROUGH LOOP: ";
130 INPUT N
135 PRINT "G"
140 FOR I=1 TO N
150 A=0
160 NEXT I
170 PRINT "G"
180 END
190 REM 1 SECOND WHEN N=100
200 REM 8 SECONDS WHEN N=1000
210 REM 84 SECONDS WHEN N=10000


100 REM  -----TEK 4051 SPEED TEST-----
110 REM  --------ADDITION LOOP--------
120 PRINT "NUMBER OF TIMES THROUGH LOOP: ";
130 INPUT N
135 A=0
140 PRINT "G"
150 FOR I=1 TO N
160 A=A+1
170 NEXT I
180 PRINT "G"
190 END
200 REM  <1 SECOND WHEN N=100
210 REM 10  SECONDS WHEN N=1000
220 REM 107 SECONDS WHEN N=10000
```

```
100 REM  -----TEK 4051 SPEED TEST-----
110 REM  ------SUBTRACTION  LOOP------
120 PRINT "NUMBER OF TIMES THROUGH LOOP: ";
130 INPUT N
135 A=0
140 PRINT "G"
150 FOR I=1 TO N
160 A=A-1
170 NEXT I
180 PRINT "G"
190 END
200 REM  <1 SECOND WHEN N=100
210 REM 10  SECONDS WHEN N=1000
220 REM 107 SECONDS WHEN N=10000


100 REM  -----TEK 4051 SPEED TEST-----
110 REM  -------MULTIPLY  LOOP-------
120 PRINT."NUMBER OF TIMES THROUGH LOOP: ";
130 INPUT N
135 A=0
140 PRINT "G"
150 FOR I=1 TO N
160 A=A*1
170 NEXT I
180 PRINT "G"
190 END
200 REM  <1 SECOND WHEN N=100
210 REM 10  SECONDS WHEN N=1000
220 REM 118 SECONDS WHEN N=10000


100 REM  -----TEK 4051 SPEED TEST-----
110 REM  --------DIVIDE  LOOP-------
120 PRINT "NUMBER OF TIMES THROUGH LOOP: ";
130 INPUT N
135 A=0
140 PRINT "G"
150 FOR I=1 TO N
160 A=A/1
170 NEXT I
180 PRINT "G"
190 END
200 REM 1.7 SECOND WHEN N=100
210 REM 17  SECONDS WHEN N=1000
220 REM 175 SECONDS WHEN N=10000


100 REM  -----TEK 4051 SPEED TEST-----
110 REM  --------PRINT   LOOP-------
120 PRINT "NUMBER OF TIMES THROUGH LOOP: ";
130 INPUT N
135 A=0
140 PRINT "G"
150 FOR I=1 TO N
160 PRINT A;
170 NEXT I
180 PRINT "G"
190 END
200 REM 2.7 SECOND WHEN N=100
210 REM 25  SECONDS WHEN N=1000
220 REM 252 SECONDS WHEN N=10000
```

```
100 REM  -----TEK 4051 SPEED TEST-----
110 REM  --------FALSE IF LOOP------
120 PRINT "NUMBER OF TIMES THROUGH LOOP: ";
130 INPUT N
135 A=0
140 PRINT "G"
150 FOR I=1 TO N
160 IF A=1 THEN 170
170 NEXT I
180 PRINT "G"
190 END
200 REM 1.0 SECOND WHEN N=100
210 REM 11  SECONDS WHEN N=1000
220 REM 113 SECONDS WHEN N=10000
```

```
100 REM  -----TEK 4051 SPEED TEST-----
110 REM  --------TRUE  IF LOOP------
120 PRINT "NUMBER OF TIMES THROUGH LOOP: ";
130 INPUT N
135 A=1
140 PRINT "G"
150 FOR I=1 TO N
160 IF A=1 THEN 170
170 NEXT I
180 PRINT "G"
190 END
200 REM 1.0 SECOND WHEN N=100
210 REM 11  SECONDS WHEN N=1000
220 REM 113 SECONDS WHEN N=10000
```

```
100 REM  -----TEK 4051 SPEED TEST-----
110 REM  -----SEQUENTIAL TAPE READ--
115 PRINT "FILE TO BE USED: ";
116 INPUT F
117 FIND F
120 PRINT "NUMBER OF TIMES THROUGH LOOP: ";
130 INPUT N
140 PRINT "G"
150 FOR I=1 TO N
160 READ @33:A
170 NEXT I
180 PRINT "G"
190 END
200 REM 0.6 SECONDS WHEN N=10
210 REM 0.8 SECONDS WHEN N=20
220 REM 0.8 SECONDS WHEN N=25
230 REM 1.0 SECONDS WHEN N=30
240 REM 1.0 SECONDS WHEN N=40
245 REM 1.15 SECONDS WHEN N=45
250 REM 1.3 SECONDS WHEN N=50
```

```
100 REM  -----TEK 4051 SPEED TEST-----
110 REM  ----SEQUENTIAL TAPE WRITE---
120 PRINT "FILE TO BE USED: ";
130 INPUT F
140 FIND F
150 PRINT "NUMBER OF TIMES THROUGH LOOP: ";
160 INPUT N
170 A=1
180 PRINT "G"
190 FOR I=1 TO N
200 WRITE @33:A
210 NEXT I
220 PRINT "G"
230 END
240 REM 0.55 SECONDS WHEN N=10
250 REM 0.7  SECONDS WHEN N=15
260 REM 0.8  SECONDS WHEN N=20
270 REM 0.95 SECONDS WHEN N=25
280 REM 1.2  SECONDS WHEN N=30
290 REM 1.25 SECONDS WHEN N=35
300 REM 1.35 SECONDS WHEN N=40
310 REM 1.4  SECONDS WHEN N=45
320 REM 1.5  SECONDS WHEN N=50
```

```
200 REM::::::::::::::::::::::::::::::::::::::::::
210 REM            SEQUENTIAL FILE
220 REM             UPDATE
230 REM::::::::::::::::::::::::::::::::::::::::::
240 PRINT "INPUT FILE NUMBER IS";
250 INPUT F1
260 PRINT "OUTPUT FILE NUMBER IS";
270 INPUT F2
280 PRINT "N=";
290 INPUT N
300 REM.....................................
310 REM        CREATE TEST FILE
320 REM.....................................
330 FIND F1
340 FOR I=1 TO N
350 WRITE @33:I
360 NEXT I
370 PRINT "GGG"
380 J=0
390 J=J+1
400 REM.....................................
410 REM          READ AN OLD RECORD
420 REM.....................................
430 FIND F1
440 FOR I=1 TO J
450 READ @33:N
460 IF TYP(0)=1 THEN 620
470 NEXT I
480 REM.....................................
490 REM        WRITE UPDATED RECORD
500 REM.....................................
510 FIND F2
520 IF J=1 THEN 560
530 FOR K=1 TO J-1
540 READ @33:Q
550 NEXT K
560 N=N+1
570 WRITE @33:N
580 GO TO 390
590 REM.....................................
600 REM      UPDATE COMPLETED
610 REM.....................................
620 PRINT "GGG"
630 END
```

```
100 REM -------- WANG SPEED TEST --------------
110 REM --------    EMPTY LOOP   --------------
120 PRINT HEX(03);"NUMBER OF TIMES THROUGH LOOP";
130 INPUT N
140 FOR I=1 TO N
150 NEXT I
160 END
170 REM  0.50 SECONDS, WHERE N=100
180 REM  4.25 SECONDS, WHERE N=1000
190 REM 41.30 SECONDS, WHERE N=10000
```

```
100 REM -------- WANG SPEED TEST --------------
110 REM --------   NESTED LOOP   --------------
120 PRINT HEX(03);"NUMBER OF TIMES THROUGH LOOP";
130 INPUT N
140 FOR I=1 TO N
150 FOR J=1 TO N
160 NEXT J
170 NEXT I
180 END
190 REM  0.60 SECONDS, WHERE N=10
200 REM 42.30 SECONDS, WHERE N=100
```

```
100 REM -------- WANG SPEED TEST --------------
110 REM -------- ASSIGNMENT LOOP --------------
120 PRINT HEX(03);"NUMBER OF TIMES THROUGH LOOP";
130 INPUT N
140 FOR I=1 TO N
150 A=0
160 NEXT I
170 END
180 REM  1.00 SECONDS, WHERE N=10
190 REM  7.60 SECONDS, WHERE N=100
200 REM 74.50 SECONDS, WHERE N=1000
```

```
100 REM -------- WANG SPEED TEST --------------
110 REM --------  ADDITION LOOP  --------------
120 PRINT HEX(03);"NUMBER OF TIMES THROUGH LOOP";
130 INPUT N
140 FOR I=1 TO N
150 A=A+1
160 NEXT I
170 END
180 REM  1.00 SECONDS, WHERE N=100
190 REM  8.50 SECONDS, WHERE N=1000
200 REM 96.00 SECONDS, WHERE N=10000
```

```
100 REM -------- WANG SPEED TEST --------
110 REM --------  SUBTRACT LOOP  --------
120 PRINT HEX(03);"NUMBER OF TIMES THROUGH LOOP";
130 INPUT N
140 FOR I=1 TO N
150 A=A-1
160 NEXT I
170 END
180 REM  1.00 SECONDS, WHERE N=100
190 REM  8.70 SECONDS, WHERE N=1000
200 REM 96.10 SECONDS, WHERE N=10000
```

```
100 REM -------- WANG SPEED TEST --------
110 REM --------  MULTIPLY LOOP  --------
120 PRINT HEX(03);"NUMBER OF TIMES THROUGH LOOP";
130 INPUT N
140 FOR I=1 TO N
150 A=A*1
160 NEXT I
170 END
180 REM  1.30 SECONDS, WHERE N=100
190 REM  8.60 SECONDS, WHERE N=1000
200 REM 84.50 SECONDS, WHERE N=10000
```

```
100 REM -------- WANG SPEED TEST --------
110 REM -------- DIVIDE LOOP (1) --------
120 PRINT HEX(03);"NUMBER OF TIMES THROUGH LOOP";
130 INPUT N
140 A=1
150 FOR I=1 TO N
160 A=A/1
170 NEXT I
180 END
190 REM   1.20 SECONDS, WHERE N=100
200 REM  11.10 SECONDS, WHERE N=1000
210 REM 119.30 SECONDS, WHERE N=10000
```

```
100 REM -------- WANG SPEED TEST --------
110 REM --------  PRINT LOOP (2) --------
120 PRINT HEX(03);"NUMBER OF TIMES THROUGH LOOP";
130 INPUT N
140 A=1
150 FOR I=1 TO N
160 PRINT A;
170 NEXT I
180 END
190 REM  1.00 SECONDS, WHERE N=100
200 REM  7.70 SECONDS, WHERE N=1000
210 REM 55.50 SECONDS, WHERE N=10000
```

```
100 REM -------- WANG SPEED TEST --------------
110 REM --------  FALSE IF LOOP  --------------
120 PRINT HEX(03);"NUMBER OF TIMES THROUGH LOOP";
130 INPUT N
140 A=1
150 FOR I=1 TO N
160 IF A=0 THEN 170
170 NEXT I
180 END
190 REM  1.00 SECONDS, WHERE N=100
200 REM  9.00 SECONDS, WHERE N=1000
210 REM 87.00 SECONDS, WHERE N=10000
```

```
100 REM -------- WANG SPEED TEST --------------
110 REM --------  TRUE IF LOOP   --------------
120 PRINT HEX(03);"NUMBER OF TIMES THROUGH LOOP";
130 INPUT N
140 A=0
150 FOR I=1 TO N
160 IF A=0 THEN 170
170 NEXT I
180 END
190 REM  1.20 SECONDS, WHERE N=100
200 REM  9.75 SECONDS, WHERE N=1000
210 REM 95.90 SECONDS, WHERE N=10000
```

```
100 REM ****************************************
110 REM       SEQUENTIAL DISK READ
120 REM ****************************************
130 REM
140 REM FILE SIZE:S=100
150 REM *** FILE 1 ***
160 REM FIRST SECTOR:F1=260
170 PRINT HEX(03);"INPUT NUMBER OF RECORDS -";:INPUT N:IF N<>INT(N) THEN 170:IF N<1 THEN
180 FOR I=F1 TO F1+N-1:DATA LOAD DA F(I,L) I:NEXT I:END
190 REM  0.6 SECONDS, WHERE N=5
200 REM  0.8 SECONDS, WHERE N=10
210 REM  1.1 SECONDS, WHERE N=15
220 REM  1.4 SECONDS, WHERE N=20
230 REM  1.9 SECONDS, WHERE N=30
240 REM  2.4 SECONDS, WHERE N=40
250 REM  2.9 SECONDS, WHERE N=50
```

```
100 REM ****************************************
110 REM       SEQUENTIAL DISK WRITE
120 REM ****************************************
130 REM
140 REM FILE SIZE:S=100
150 REM *** FILE 1 ***
160 REM FIRST SECTOR:F1=260
170 REM LAST  SECTOR:L1=F1+S-1
180 PRINT HEX(03);"INPUT NUMBER OF RECORDS -";:INPUT N:IF N<>INT(N) THEN 180:IF N<1 THEN
190 FOR I=F1 TO F1+N-1:DATA SAVE DA F(I,L) I:NEXT I:END
200 REM  0.6 SECONDS, WHERE N=5
210 REM  0.8 SECONDS, WHERE N=10
220 REM  1.1 SECONDS, WHERE N=15
230 REM  1.4 SECONDS, WHERE N=20
240 REM  1.9 SECONDS, WHERE N=30
250 REM  2.4 SECONDS, WHERE N=40
260 REM  2.9 SECONDS, WHERE N=50
```

```
100 REM --------- WANG SPEED TEST ----------
110 REM -------- SELECTION  SORT ----------
120 DIM D(128)
130 PRINT HEX(03);"NUMBER OF VALUES TO BE SORTED";
140 INPUT N
150 FOR I=1 TO N
160 D(I)=INT(RND(-1)*100000)+1
170 NEXT I
180 STOP "-- PRESS CONTINUE THEN (CR) -- START CLOCK"
190 M=N
200 B=1
210 FOR I=1 TO M
220 IF D(B)>D(I) THEN 240
230 B=I
240 NEXT I
250 T=D(M)
260 D(M)=D(B)
270 D(B)=T
280 M=M-1
290 IF M>1 THEN 200
300 END
310 REM  9.60 SECONDS, WHERE N=32
320 REM  44.30 SECONDS, WHERE N=64
330 REM 121.00 SECONDS, WHERE N=128
```

```
100 REM ************************************
110 REM        ARRAY MERGE/SORT
120 REM ************************************
130 DIM L(30),L1(30)
140 D=31
150 P1,P2,P3,P4,P5,P6=0
160 PRINT HEX(03);TAB(20);"MERGE/SORT ROUTINE"
170 PRINT
180 PRINT "INPUT LENGTH OF LIST ";
190 INPUT N
200 REM
210 REM INPUT DISORDERED LIST
220 REM
230 FOR I=1 TO N
240 L(I)=INT(1000*RND(-1))
250 NEXT I
260 P1=N
270 STOP " -- TYPE IN CONTINUE (CR) - START CLOCK"
280 REM
290 REM CALL "PASSES"
300 REM
310 GOSUB 430
320 REM
330 REM OUTPUT PROBES
340 PRINT "N = ";N,"PROBES ........"
350 PRINT
360 PRINT "PROBE 1 SHOULD BE ";N," IS ";P1
370 PRINT "PROBE 2 SHOULD BE ";INT(LOG(N)/LOG(2))," IS ";P2
380 PRINT "PROBE 3 SHOULD BE ";INT((N-1)/2)*2+1," IS ";P3
390 PRINT "PROBE 4 SHOULD BE ";N," IS ";P4
400 PRINT "PROBE 5 SHOULD BE ";0," IS ";P5-P6
410 PRINT
420 STOP " -- END OF MERGE/SORT"
430 REM ************************************
440 REM           PASSES SUBROUTINE
450 REM ************************************
460 REM
470 REM EXCHANGE PAIRS
480 REM
490 GOSUB 830
500 REM
510 REM    INITIALIZE PASSES,LENGTH,START SUBSCRIPT
520 REM
530 S3=INT(LOG(N)/LOG(2))
540 L2=1
550 REM
560 REM   REPEAT PASSES S3 TIMES
570 REM
580 FOR I=1 TO S3
590 L2=L2*2
600 S1=1
610 REM
620 REM AGAIN
630 REM
640 S2=S1+L2
650 REM
660 REM MERGE SUBLISTS
670 REM
680 GOSUB 1040
690 REM
700 REM READY FOR NEXT PASS
710 REM
720 S1=S2+L2
730 IF S1<=N THEN 640
```

```
760 REM
770 GOSUB 1390
780 REM
790 P2=I
800 NEXT I
810 REM
820 RETURN
830 REM ************************************
840 REM EXCHANGE PAIRS
850 REM ************************************
860 REM
870 REM REPEAT
880 REM
890 FOR J=1 TO N STEP 2
900 IF J=N THEN 1010
910 IF L(J)<L(J+1) THEN 1010
920 REM
930 REM SWAP PAIRS
940 REM
950 T=L(J)
960 L(J)=L(J+1)
970 L(J+1)=T
980 REM
990 REM COMPUTE PROBE
1000 REM
1010 P3=J
1020 NEXT J
1030 RETURN
1040 REM ************************************
1050 REM           MERGE SUBLISTS
1060 REM ************************************
1070 REM
1080 REM INITIALIZE
1090 REM
1100 I1=S1
1110 I2=S2
1120 M=S2+L2-1
1130 IF M<=N THEN 1180
1140 M=N
1150 REM
1160 REM REPEAT
1170 REM
1180 FOR K=S1 TO M
1190 IF I1>=S2 THEN 1250
1200 IF I2>=M+1 THEN 1310
1210 IF L(I1)<=L(I2) THEN 1310
1220 REM
1230 REM SMALL
1240 REM
1250 L1(K)=L(I2)
1260 I2=I2+1
1270 GOTO 1360
1280 REM
1290 REM LARGE
1300 REM
1310 L1(K)=L(I1)
1320 I1=I1+1
1330 REM
1340 REM END
1350 REM
1360 P4=K
1370 NEXT K
1380 RETURN
```

```
1400 REM COPY L1 INTO L
1410 REM ***************************************
1420 REM
1430 P5=0
1440 REM
1450 REM REPEAT LOOP
1460 REM
1470 FOR K=1 TO N
1480 L(K)=L1(K)
1490 P5=P5+L(K)
1500 NEXT K
1510 REM
1520 RETURN
1530 REM  3.35 SECONDS, WHERE N=10
1540 REM  4.60 SECONDS, WHERE N=15
1550 REM  7.50 SECONDS, WHERE N=20
1560 REM  9.40 SECONDS, WHERE N=25
1570 REM 11.10 SECONDS, WHERE N=30
```

```
100 REM ************************************
110 REM         DISK MERGE/SORT
120 REM ************************************
130 REM FILE SIZES:S=49
140 REM *** FILE 1 ***
150 REM FIRST SECTOR:F1=260
160 REM *** FILE 2 ***
170 REM FIRST SECTOR:F2=310
180 D=31
190 P1,P2,P3,P4,P5,P6=0
200 PRINT HEX(03);TAB(20);"DISK MERGE/SORT ROUTINE"
210 PRINT
220 PRINT "INPUT LENGTH OF LIST ";
230 INPUT N:IF N<>INT(N) THEN 220:IF N<1 THEN 220:IF N>S THEN 220
240 REM
250 REM INPUT DISORDERED LIST
260 REM
270 FOR I=1 TO N
280 Q1=INT(1000*RND(-1))
290 DATA SAVE DA F(F1+I,L) Q1
300 NEXT I
310 P1=N
320 STOP " -- TYPE IN CONTINUE (CR) - START CLOCK"
330 REM
340 REM CALL "PASSES"
350 REM
360 GOSUB 480
370 REM
380 REM OUTPUT PROBES
390 PRINT "N = ";N,"PROBES ........"
400 PRINT
410 PRINT "PROBE 1 SHOULD BE ";N," IS ";P1
420 PRINT "PROBE 2 SHOULD BE ";INT(LOG(N)/LOG(2))," IS ";P2
430 PRINT "PROBE 3 SHOULD BE ";INT((N-1)/2)*2+1," IS ";P3
440 PRINT "PROBE 4 SHOULD BE ";N," IS ";P4
450 PRINT "PROBE 5 SHOULD BE ";0," IS ";P5-P6
460 PRINT
470 STOP " -- END OF MERGE/SORT"
480 REM ************************************
490 REM           PASSES SUBROUTINE
500 REM ************************************
510 REM
520 REM EXCHANGE PAIRS
530 REM
540 GOSUB 880
550 REM
560 REM    INITIALIZE PASSES,LENGTH,START SUBSCRIPT
570 REM
580 S3=INT(LOG(N)/LOG(2))
590 L2=1
600 REM
610 REM   REPEAT PASSES S3 TIMES
620 REM
630 FOR I=1 TO S3
640 L2=L2*2
650 S1=1
660 REM
670 REM AGAIN
680 REM
690 S2=S1+L2
700 REM
```

```
730 GOSUB 1100
740 REM
750 REM READY FOR NEXT PASS
760 REM
770 S1=S2+L2
780 IF S1<=N THEN 690
790 REM
800 REM COPY L1 INTO L
810 REM
820 GOSUB 1490
830 REM
840 P2=I
850 NEXT I
860 REM
870 RETURN
880 REM ****************************************
890 REM          EXCHANGE PAIRS
900 REM ****************************************
910 REM
920 REM REPEAT
930 REM
940 FOR J=1 TO N STEP 2
950 IF J=N THEN 1070
960 DATA LOAD DA F(F1+J,L) Q1
970 DATA LOAD DA F(F1+J+1,L) Q2
980 IF Q1<Q2 THEN 1070
990 REM
1000 REM SWAP PAIRS
1010 REM
1020 DATA SAVE DA F(F1+J,L) Q2
1030 DATA SAVE DA F(F1+J+1,L) Q1
1040 REM
1050 REM COMPUTE PROBE
1060 REM
1070 P3=J
1080 NEXT J
1090 RETURN
1100 REM ****************************************
1110 REM            MERGE SUBLISTS
1120 REM ****************************************
1130 REM
1140 REM INITIALIZE
1150 REM
1160 I1=S1
1170 I2=S2
1180 M=S2+L2-1
1190 IF M<=N THEN 1240
1200 M=N
1210 REM
1220 REM REPEAT
1230 REM
1240 FOR K=S1 TO M
1250 IF I1>=S2 THEN 1330
1260 IF I2>=M+1 THEN 1400
1270 DATA LOAD DA F(F1+I1,L) Q1
1280 DATA LOAD DA F(F1+I2,L) Q2
1290 IF Q1<=Q2 THEN 1400
1300 REM
1310 REM SMALL
1320 REM
1330 DATA LOAD DA F(F1+I2,L) Q2
1340 DATA SAVE DA F(F2+K,L) Q2
1350 I2=I2+1
1360 GOTO 1460
```

```
1380 REM LARGE
1390 REM
1400 DATA LOAD DA F(F1+I1,L) Q1
1410 DATA SAVE DA F(F2+K,L) Q1
1420 I1=I1+1
1430 REM
1440 REM END
1450 REM
1460 P4=K
1470 NEXT K
1480 RETURN
1490 REM ****************************************
1500 REM                COPY L1 INTO L
1510 REM ****************************************
1520 REM
1530 P5=0
1540 REM
1550 REM REPEAT LOOP
1560 REM
1570 FOR K=1 TO N
1580 DATA LOAD DA F (F2+K,L) Q1
1590 DATA SAVE DA F (F1+K,L) Q1
1600 P5=P5+Q1
1610 NEXT K
1620 REM
1630 RETURN
1640 REM  35.9 SECONDS, WHERE N=10
1650 REM  41.1 SECONDS, WHERE N=15
1660 REM  58.1 SECONDS, WHERE N=20
```

```
900 REM   MODIFIED MERGE/SORT PACKAGE
901 REM
902 REM       OPEN FILES L AND L1
910 OPEN#1,"L"
920 OPEN#2,"L1"
922 PRINT "SELECT OUTPUT DEVICE (0=CRT,4=PRINTER) :";INPUT D
1000 REM ************************************************************
1001 REM                FILE MERGE/SORT PACKAGE
1002 REM                     COPYRIGHT 1977
1003 REM ************************************************************
1004 REM
1005 REM NO ARRAY NEEDED
1006 REM  INITIALIZE PROBES
1007 LET P1=0;LET P2=0;LET P3=0;LET P4=0;LET P5=0;LET P6=0
1008 REM
1010 REM       INPUT N,L(.)
1012 PRINT#D,"MERGE/SORT ROUTINE."
1014 PRINT "INPUT LENGTH OF LIST :";INPUT N
1016 PRINT "INPUT DISORDERED LIST....."
1017 FOR I=1 TO N
1018 REM    RANDOM FILE ACCESS
1019 RESTORE#1,I-1
1020 REM-----GENERATE RANDOM DATA-----------
1021 LET L=INT(1000*RND(1))
1022 PRINT#D,L
1023 REM-----GENERATE RANDOM DATA-----------
1024 PRINT#1,L;I
1025 REM    RANDOM WRITE DONE
1026 NEXT I
1028 PRINT "START CLOCK"
1029 REM SORT PROBE 1
1030 LET P1=N
1032 REM
1034 REM CALL "PASSES"
1036 GOSUB 2000
1038 REM
1039 REM OUTPUT
1040 PRINT#D,"N = ",N,"PROBES....."
1041 PRINT#D,"PROBE 1 SHOULD BE ",N," IS ",P1
1042 PRINT#D,"PROBE 2 SHOULD BE ",INT(LOG(N)/LOG(2))," IS ",P2
1043 PRINT#D,"PROBE 3 SHOULD BE ",INT((N-1)/2)*2+1," IS ",P3
1044 PRINT#D,"PROBE 4 SHOULD BE ",N," IS ",P4
1045 PRINT#D,"PROBE 5 SHOULD BE 0.0  IS ",P5-P6
1046 END#1
1047 END#2
1048 OPEN#1,"L"
1049 REM
1050 FOR I=1 TO N
1051 REM    RANDOM FILE READ
1052 RESTORE#1,I-1
1053 INPUT#1,L;J
1055 PRINT#D,I," :  ",L,J
1058 NEXT I
1060 PRINT#D,"END OF MERGE/SORT"
1065 END#1
1070 STOP
2000 REM ************************************************************
2001 REM                      PASSES
2002 REM ************************************************************
2003 REM
2004 REM     EXCHANGE PAIRS
2010 GOSUB 3000
2012 REM
```

```
2018 LET L2=1
2019 REM
2020 REM     REPEAT PASSES S3 TIMES
2022 FOR I=1 TO S3
2024 LET L2=2*L2: LET S1=1
2025 REM   AGAIN
2026 LET S2=S1+L2
2029 REM
2030 REM     MERGE SUBLISTS
2032 GOSUB 4000
2034 REM
2040 REM     READY FOR NEXT PASS
2042 LET S1=S2+L2
2043 IF S1<=N THEN 2026
2044 REM    COPY L1 INTO L
2045 GOSUB 5000
2046 REM
2047 LET P2=I
2048 NEXT I
2049 REM
2050 RETURN
3000 REM ************************************************************
3001 REM                        EXCHANGE PAIRS
3002 REM ************************************************************
3003 REM
3006 REM REPEAT
3010 FOR J=1 TO N STEP 2
3015 IF J=N THEN 3050
3018 REM     RANDOM ACCESS FILES
3020 RESTORE#1, J-1
3021 INPUT#1, L: W
3022 RESTORE#1, J
3023 INPUT#1, L9: W9
3030 IF L<L9 THEN 3050
3031 REM
3040 REM SWAP PAIRS
3042 RESTORE#1, J-1
3043 PRINT#1, L9: W9
3044 RESTORE#1, J
3045 PRINT#1, L: W
3046 REM
3048 REM COMPUTE PROBE
3050 LET P3=J
3052 NEXT J
3054 RETURN
4000 REM ************************************************************
4001 REM                        MERGE SUBLISTS
4002 REM ************************************************************
4003 REM
4010 REM    INITIALIZE.
4011 LET I1=S1: LET I2=S2: LET M=S2+L2-1
4012 IF M<=N THEN 4022
4013 LET M=N
4014 REM
4020 REM    REPEAT
4022 FOR K=S1 TO M
4023 IF I1>=S2 THEN 4030
4024 RESTORE#1, I1-1
4025 INPUT#1, L: W
4026 IF I2>=(M+1) THEN 4040
4027 RESTORE#1, I2-1
4028 INPUT#1, L9: W9
4029 IF L<=L9 THEN 4040
4030 RESTORE#2, K-1
```

```
4034 LET I2=I2+1
4035 GOTO 4050
4040 RESTORE#2,K-1
4041 PRINT#2,L;W
4042 LET I1=I1+1
4044 REM
4046 REM     END
4050 LET P4=K
4055 NEXT K
4060 RETURN
5000 REM ************************************************************************
5001 REM                              COPY L1 INTO L
5002 REM ************************************************************************
5003 REM
5004 LET P5=0
5009 REM     REPEAT COPY
5010 FOR K=1 TO N
5011 RESTORE#2,K-1
5012 INPUT#2,L1;W
5013 RESTORE#1,K-1
5014 PRINT#1,L1;W
5017 LET P5=P5+L1
5019 NEXT K
5020 REM
5030 RETURN
```

WHAT SHOULD BE DONE TO IMPROVE THE 4051 AS A COMPETITIVE ALTERNATIVE TO OTHER SYSTEMS?

When evaluating the performance of the Tektronix 4051 computer, there are areas of consideration important to the programmer and others of significant value to the end user. First, we shall analyze features for the software developer which appeal to a programmer.

The use of a BASIC interpreter rather than a BASIC compiler is a definite advantage to the programmer. An interpreter significantly reduces actual program development time by eliminating waiting for typically slow compilations. As the cost of personnel increases and the price of the computer decreases, this cost in lost time is magnified.

Program development time could also be decreased if 4051 BASIC allowed multiple statement lines. Currently, the 4051 recognizes only one BASIC statement or command per line which increases coding, typing, and listing time, all of which are quite expensive.

A feature which would enhance the readability of the listings produced by the 4051 would be the automatic indentation of FOR - NEXT loops. In addition to improving the cosmetics of the code, it enables the programmer and any future users to better understand the logic and flow of the program.

A feature which would aid both the programmer and the end user is the ability of the system to "trap" any fatal and non-fatal errors as they happen. If the 4051 is to be implemented

in turnkey business applications, it is essential that this feature be included. When an error of any kind occurs, the diagnostic message should not be displayed on the screen for the naive user to worry about. The turnkey software package should have all provisions to internally cope with these errors and explain, in English, how the user should proceed to correct the error, be it system or user caused.

Block and context editors would also be a widely used item if implemented on the 4051. They enable a programmer to cut actual code entering time by allowing him to use sections of code previously entered in other programs and by reducing the need to use the line editing keys at the top of the keyboard. It should be possible to scan an entire file for the occurance of a character string and it should be possible to replace one character string with another.

Provision should be made for cursor and data entry control. Having exact cursor control enables the programmer to design forms on the screen which closely resemble the paper forms that the user is keying from, and create a cosmetically attractive display.

During the design process of a turnkey business system, it is often helpful to have the ability for adequate file and program protection. The ability to give file names is also important when dealing with a complete system of perhaps twenty or more discrete programs. The "secret" provision of the 4051 does not adequately protect the user from himself or

other users of his data files. Note that without the "read Protect", which has become increasingly important since the advent of the Privacy Act, anyone with access to the machine and the magnetic tape cartridges has access to all data contained therein. The implementation of user defined passwords would greatly reduce the problem posed by the use of the "secret" provision. "Secret", programs are also ineffective in protecting the software developer because the "secret" program can still be executed even though it may not be alterable.

Virtual memory allocation is very important when designing turnkey business packages because the system very frequently must overlay various subroutines. Currently, provisions for overlaying of subroutines is very awkward to use.

If possible, improved diagnostics should be provided to the programmer. Time is lost when it becomes necessary to look up a diagnostic code in the user's manual. This addition, however, would only be necessary when the machine is used as a program development tool and if the ability to "trap" individual errors were adopted.

When a turnkey package has been developed and debugged, the system should provide an optional compiler rather than run on the interpreter basis. Obviously, the compiled version would run faster and would provide an added amount of software security. This would provide the software developer an option for programs that are used frequently (sort, index, formating files, etc.).

TO THE END USER

The end user will benefit from a few changes also. The appearance of the computer to the user is very important in business applications and greatly affects the time required to learn its operation. A simple, non-cluttered keyboard area would be much easier for the naive user to learn and operate. There are too many unused buttons and switches on the front of the computer which needlessly complicate the actual use of the machine.

The auto-load feature is valuable to have and use in a turnkey application and should definitely be retained. This enables the user to simply "load-and-go" with his application package and greatly reduces the visibility of the operating system.

File protection, as discussed earlier, will be of prime importance to the businessman in protecting personnel and customer files. The accidental destruction of a vital file could set a business back weeks while the data is re-entered. Privileged information must be protected from unauthorized access.

Another very nice feature of the 4051 is its almost, complete transparancy of an operating system. This feature would be complete with the addition of the ability to "trap" fatal and non-fatal errors treated during the execution of business packages. The naive user does not wish to be bothered by diagnostics, and the display of those diagnostics is usually

worthless in helping the user to correct the mistake.

Commonly used business utilities such as sorting, merging, indexing, and reformatting should be implemented on a separate ROM pack. These functions are vital and cannot be overstressed when discussing turnkey business applications.

GENERAL RECOMMENDATIONS

When comparing the modes of mass storage of the three computers utilized in this report, certain facts come to light. The one available magnetic tape drive should be augmented, in most applications, by the addition of at least one diskette drive. When dealing with the tape, random, logical record accessing is very clumsy and the ability for indexed sequential accessing is lost. The diskette would at least partially eliminate this problem, and if more than one diskette drive were added it would place the 4051 on a par with systems which are currently being used in small businesses.

For the end-user, the use of diskettes offers rapid random access and a storage medium which is quite easily achieved. Indexed and random access must be implemented to decrease the time required to retrieve a particular record. The diskette is lower in price than the tape cartridges, holds more data, and is more easily replaced in event of surface damage. Consequently, diskettes have been chosen by almost all manufacturers of OEM small business computers.

Regardless of the medium of storage, it is very important

to have the ability to hardware protect the storage device. The 4051 takes advantage of the write protect switch on the cartridge which is a very important feature.

The time necessary to train programmers in the use and programming of the 4051 system is critical. The manuals which are currently provided with the system are redundant, too verbose, and fragmented. These manuals should be written in a "top-down" format and should also be modularized. The manuals dealing with BASIC and the graphics potential of the system should be merged. The binding of the manuals is also very important. Currently, they are bound by a number of fragile plastic rings which have a tendancy to break. If at all possible, a soft binding should be provided.

The system reference manual should be condensed, making a smaller manual. Allowance should be made for rapid access of vital data such as diagnostic codes. A complete syntactical index would also be greatly appreciated by the novice 4051 programmer in determining the difference between 4051 BASIC and others currently available.

APPENDIX A

Error and Correction for

"INTRODUCTION TO PROGRAMMING IN BASIC"

Do Jin Kim

page	line #	error	correction
2-3	6	((A+B)/(A-B)↑ 2 should be	usually it would be as shown in the manual, but ((A+B)/(A-B) ↑ 2) is also valid representation ((A+B)/(A-B)↑ 2 would be (or could be) ((A+B)/(A-B)) ↑ 2 or ((A+B)/(A-B) ↑ 2)
2-3	16	NOT EQUAL TO < >	but >< is also valid, and automatically converted to < > internally, i.e., < > and >< both work as NOT EQUAL TO and should be included in the table = > , > = and < =, = > ex) 100 IF A><B THEN 120 LIST 100 IF A<>B THEN 120
2-6		Operator hierarchy PRECEDENCE #9 AND, OR and NOT	Obviously NOT has higher priority as in Appendix A-1 table cf) In FORTRAN, AND has higher precedence than OR
2-34	Table	300 READ A,B,C, 310 DATA 75.62,125.3	300 READ A,B,C 310 DATA 75.62,125,3
3-18	14	you can ause cause	you can cause
5-10	17	...PLUS THE NUMBER OF TIMES IT IS USED ...	On result, no print out for that number and program list 140 L=LEN(B$) is not used in program **programmer might have changed the program, but forgot to correct the description and eliminate Line 140

6-5 13

IF N IS LESS THAN 1, GOSUB INSTRUCTION IS IGNORED.

No!!

(ex)

```
100 GOSUB .5 OF 170,140,150
110 PRINT "RETURN"
120 GO TO 190
140 PRINT "ERROR 140"
150 PRINT "ERROR 150"
160 GO TO 190
170 PRINT "GOSUB OK"
180 RETURN
190 END
RUN
GOSUB OK
RETURN
```

7-1 Syntax Box

[line#] PRINT USING {line# / format} : [{ variable list / number expr }]

↑ this can be omitted in some cases

7-11 14

...TRAILING ZEROS ARE SUPPRESSED

(ex) 3.0E+10
There is no way computer can respond without decimal point for E-format

A=3E+10 is valid though,

```
PRINT A
 3.0E+10
```

7-24 program list

```
170  enter the number
       of data items
 ..............
220 PRINT @33:A;B;C;D;E
```

There is no proper reason to ask the number of data items at line 170.
This program prints out 5 items at line 120 regardless of the INPUT data.
...Suppose we answered 3 for line 170, but still get 5 data, i.e.,stupid program

Use Array and FOR I=1 to Y

8-15

5 X 8 matrix

8 X 5 matrix

2-14

PRINT A,B,C, is not allowed

last comma is ignored,

SUGGESTIONS FOR MANUAL

page	line#	suggestions
1-3	8	1.Figure 1-1; This figure misleads readers. One might guess that any number between 0.001 and 10,000,000 could be represented in standard form. Actually, the value of number has not much relationship with STANDARD/SCIENTIFIC notation. Important thing is the number of effective digits for that value and that fact should be clearly declared in the explanation. Therefore Figure 1-1 is almost useless. 2. Need to explain what happens if one tries to represent the big number by standard form, and the manual fails to tell the maximum number of effective digits for STANDARD form representation. ex) A=12345678912345678900 PRINT A 1.234567891E+19
1-5	18	"numeric values like A and B1" "array variables like A and B" 1. This example causes confusion to beginners. 2. Need a comment that numeric variable A and array variable A can not be used in the same program.
2-2	table	$\frac{Y-3}{4Q}$; (Y-3)/(4*Q) or (Y-3)/4/Q From my personal experience, most students do not know the second representation and some of them even think that it is an illegal one. It would be very instructive to show that both representations are equivalent and could comment that (Y-3)/4*Q = (Y-3)*Q/4 as foot note.
.....		1. In logical expression, no comment that 0 is for false and 1 for true. 2. No explanation for very useful logical expression as follow;

```
3-2=3  CR
 0
4-1=3  CR
 1
```

2-10 *Strongly recommend to include the following in the manual

The total number of variables in READ statements(or I/O list) in one program can not be less than the number of items in DATA statements, in other words those of DATA statements could be equal to or greater than those of READ statements.

Furthermore each READ statement does not have to correspond to each DATA statement necessarily.

ex)

```
100 READ A,B,C
110 READ D,A
120 DATA 1,2,3,4,5,6,
130 PRINT A,B,C,D
 5                2                3                4
```

2-6 More discussions recommended for AND and OR operators.

INDEX at the end of manual does not have any entries for them. INDEX should at least include all the key words used in BASIC programming language.

suggested example)

```
A=1
B=2
C=1
D=A AND (B OR C) OR A
PRINT D
 1
```

..... This manual does not have any comment on the combination of instructions having line numbers with those which have no line numbers.

ex)

```
A=123
100 PRINT A
RUN
 123
```

ex)

```
100 C=1
110 END
C=2
PRINT C
```

```
 2
RUN
PRINT C
 1
```

2-13

```
A=1
B=2
A1=3
B1=4
PRINT A 1,B 1  CR
       ↓ syntax error
PRINT A1,B1
```

Just push the RETURN key without fixing anything, then it prints out the values for A1 and B1,i.e., it works all right.

*Inconsistencies for this example.

1. A 1 gives error, but B 1 was recognized as B1 and works properly.
2. Works with "syntax error"
   -there is a comment in manual that Syntax error can not be run without fixing.

suggestion: error checking is not necessary for A 1, i.e., A 1 could be recognized as A1.

2-28
2-37

It is clear from Syntax boxes, however, better have comments that FOR..., DEF FN... instructions require line numbers.

6-1

Need a comment;
GOSUB also can be used instead of GOTO for all cases where GOTO can be used.

ex)

```
100 GOSUB 130
110 PRINT "ERROR"
120 GOTO 140
130 PRINT "OK"
140 END
RUN
 OK
```

7-19

1. Example tables for a various of output format is recommended to help understand FORMAT.

2. There is a comment in manual, but better have examples like these.

ex)
A=12345
B=-12345

```
PRINT "C5D":A       ...overflow
PRINT "$5D":A       ...not overflow
PRINT "5D":B        ...overflow
```