



Development of time shared basic system processor for Hewlett Packard 2100 series computers
by John Sidney Shema

A thesis submitted to the Graduate Faculty in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE in Electrical Engineering
Montana State University
© Copyright by John Sidney Shema (1974)

Abstract:

The subject of this thesis is the development of a low cost time share BASIC system which is capable of providing useful computer services for both commercial and educational users.

The content of this thesis is summarized as follows: First, a review of the historical work leading to the development of time share principles is presented. Second, software design considerations and related restrictions imposed by hardware capabilities and their impact on system performance are discussed. Third, 1000D BASIC operating system specifications are described by detailing user software capabilities and system operator capabilities. Fourth, TSB system organization is explained in detail. This involves presenting TSB system modules and describing communication between modules. A detailed study is made of the TSB system tables and organization of the system and user discs. Fifth, unique features of 1000D BASIC are described from a functional point of view. Sixth, a summary of the material presented in the thesis is made. Seventh, the recommendation is made that error detection and recovery techniques be pursued in order to improve system reliability.

In presenting this thesis in partial fulfillment of the requirements for an advanced degree at Montana State University, I agree that permission for extensive copying of this thesis for scholarly purposes may be granted by my major professor, or, in his absence, by the Director of Libraries. A program listing of the 1000D Time Share BASIC system is not submitted as part of this thesis since program development was performed by a private business corporation which considers the listing as proprietary information. Arrangements must be made with a representative of Western Telecomputing Corporation of Bozeman, Montana in order to obtain a program listing for evaluation. It is understood that any copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Signature John Sidney Thomas
Date February 15, 1974

DEVELOPMENT OF TIME SHARED BASIC SYSTEM PROCESSOR
FOR HEWLETT PACKARD 2100 SERIES COMPUTERS

by

JOHN SIDNEY SHEMA

A thesis submitted to the Graduate Faculty in partial
fulfillment of the requirements for the degree

of

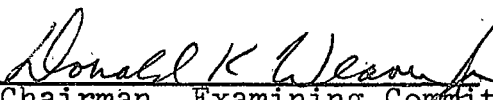
MASTER OF SCIENCE


in

Electrical Engineering

Approved:


Head, Major Department


Chairman, Examining Committee


Graduate Dean

MONTANA STATE UNIVERSITY
Bozeman, Montana

March, 1974

ACKNOWLEDGMENT

The development of any major computer software operating system is a lengthy and complex process requiring the cooperation and creative talents of many individuals. The author wishes to thank the programmers and system analysts at Hewlett Packard Company for their excellence in designing and documenting the HP 2000B time shared BASIC system which served as a starting point for this investigation.

The writer is deeply grateful to Western Telecomputing Corporation for supplying the facilities and equipment to develop and debug 1000D BASIC. He is especially thankful to Dr. Donald K. Weaver Jr. for his backing of the project and many suggestions for improvement.

J.S.S

TABLE OF CONTENTS

Chapter	Page
I. INTRODUCTION.....	1
Historical Development of Time Share Principles.....	1
Relevance of Previous Work.....	6
Organization of Remaining Chapters.....	7
II. OPERATING SYSTEM REQUIREMENTS.....	8
System Design Considerations.....	8
1000D System Specifications.....	11
III. OPERATING SYSTEM SPECIFICATIONS.....	16
User Software Capabilities.....	16
System Operator Software Capabilities.....	22
IV. TSB SYSTEM ORGANIZATION.....	27
TSB System Modules.....	27
TSB System Tables.....	45
Disc Organization.....	54
V. UNIQUE FEATURES OF 1000D.....	59
ASSIGN Statement.....	59
CHAIN Statement.....	67
FORMAT Command.....	73
DISC Command.....	78
VI. SUMMARY AND RECOMMENDATIONS.....	86
Summary.....	86
Recommendations.....	87
BIBLIOGRAPHY.....	89
APPENDIX.....	91
Appendix A: Hardware Requirements.....	92
Appendix B: 1000D Core Map.....	96

LIST OF TABLES

Table	Page
1. BASIC Matrix Operations.....	17
2. Base Page Links.....	30
3. System Console States.....	35
4. System Library Overlay Programs.....	42
5. ID Code Entry Format.....	45
6. Base Page DIREC Format.....	48
7. DIREC Values.....	49
8. COMTABLE Format.....	52
9. System Disc Organization.....	56
10. User/Library Disc Organization.....	58

LIST OF FIGURES

Figure	Page
1. Terminal Data Formats.....	13
2. Example of ROSTER Command.....	24
3. Example of REPORT Command.....	25
4. Example of DIRECTORY Command.....	25
5. Bit-Flag Representation.....	28
6. I/O Buffer States.....	32
7. Disc Address Word.....	34
8. System Queue.....	40
9. Directory Entry Format.....	46
10. Directory Pseudo-Entries.....	47
11. ADT Entry Format.....	49
12. COMTABLE Entry Coding Format.....	53
13. HP7900 Disc Organization.....	54
14. HP7901 Disc Organization.....	55
15. ASSIGN Syntax Processing Flow Diagram.....	62
16. ASSIGN Execution Flow Diagram.....	64
17. CHAIN Statement Syntax Processor.....	69
18. CHAIN Execution Flow Chart.....	71
19. FORMAT Command Flow Chart.....	74
20. DISC-UP Command Flow Diagram.....	80
21. DISC-DN Command Flow Diagram.....	84

ABSTRACT

The subject of this thesis is the development of a low cost time share BASIC system which is capable of providing useful computer services for both commercial and educational users.

The content of this thesis is summarized as follows: First, a review of the historical work leading to the development of time share principles is presented. Second, software design considerations and related restrictions imposed by hardware capabilities and their impact on system performance are discussed. Third, 1000D BASIC operating system specifications are described by detailing user software capabilities and system operator capabilities. Fourth, TSB system organization is explained in detail. This involves presenting TSB system modules and describing communication between modules. A detailed study is made of the TSB system tables and organization of the system and user discs. Fifth, unique features of 1000D BASIC are described from a functional point of view. Sixth, a summary of the material presented in the thesis is made. Seventh, the recommendation is made that error detection and recovery techniques be pursued in order to improve system reliability.

CHAPTER I

INTRODUCTION

Time sharing is a technique of computer system software design which allows many users, located at remote terminals, to have simultaneous access to a single central computer. Due to the extremely fast processing speed of the computer in comparison with that of the terminals, the computer is able to share its resources among the users. This makes it appear as if each terminal was directly connected to its own separate computer. Even if one terminal is running a program requiring large amounts of computer time, the other user terminals are required to wait only a few seconds for the computer to process their programs.

The objective of this thesis is to present an overall description of one specific time share system. This system was developed by Western Telecomputing Corporation in order to provide low cost computer services for both commercial and educational organizations in Montana

HISTORICAL DEVELOPMENT OF TIME SHARE PRINCIPLES

The introduction of the minicomputer in the late 1960's along with its associated low cost-high performance peripheral

equipment, caused the data processing industry to undergo a massive change. Early computer facilities emphasized the need to create powerful computer centers whose power was measured in terms of physical dimensions such as number of central processors, amount of main core memory available, and number of peripherals attached. Such thinking lead to the creation of multi-million dollar computer systems capable of handling tremendous quantities of problems and data. However, the cost of such systems was prohibitive to smaller companies and educational institutions.

With today's technologies of microminiaturization and large scale integration of circuits having reduced the size and cost of computers, a new philosophy of data processing has emerged. Rather than have one extremely powerful computer facility to handle all problems, it is now more economical to have several small processors, each handling a specific problem. In fact, in some cases the best solution to a specific problem is to design a special purpose processor using state of the art techniques rather than attempt to program a general purpose computer to perform the same function.¹

Historically, much data processing was accomplished by programmers writing programs to a particular customer's specifications,

¹ Kampe, Thomas W., "The Design of a General-Purpose Micro-program-Controlled Computer with Elementary Structure," in Computer Structures: Readings and Examples, pp. 341-347.

entering the program with data cards into a batch processing facility, and returning hours later to see if any error had occurred. Omission of even a single comma could mean the run produced no results. Additionally, time was required to analyze the printout to determine if the algorithm written had actually performed the desired task. If not, several more program runs, each printing diagnostic information would be required before the program ran error free. Such a procedure was acceptable after a program had been fully written and made free of logic errors. The program could then be run on a routine basis. However, this procedure is unacceptable to the programmer since little, if any, interaction with the program was allowed.

Another problem which was not handled by the large computing system was that of the small or medium sized program requiring only several seconds of computer processing time. In most cases, the person waited several hours for his program to be run on the computer even though actual execution time was less than a minute for the entire program. In many cases, the computer's central processor was idle, while it waited for the results of a previous computation to be printed.

Numerous attempts were made to solve the problem of obtaining maximum utilization of a computer's resources in the early 1960's. One of the most effective solutions was that of a time share system developed at Dartmouth College by John G. Kemeney, Chairman of the

Department of Mathematics, and Thomas E. Kurtz, Director of Computation Center. Their idea, which was first proposed in 1962, was to design a program language which would be both simple to learn yet powerful enough to perform complicated tasks. The language was implemented in 1964 and was termed BASIC, for Beginner's All-purpose Symbolic Instruction Code. Up to 40 simultaneous users were allowed to communicate with the BASIC Language processor directly through keyboard-printer terminals. A master program, called the supervisory control or executive, kept track of all communication between the users and computer. It decided who was to run next, how long they were allowed to run, and performed the appropriate task called upon.

An important feature of the Dartmouth BASIC system was that the user was not required to be in the proximity of the computer in order to run a program. Terminals could communicate directly with the computer over regular telephone circuits. Thus, computer service was never farther away than the nearest telephone.

The major advantage of a time share computer system over a batch system is the ability of the programmer to communicate directly with a "virtual processor" and have each line of code examined immediately upon entry into the computer. This allows for on-line debugging of programs which is not possible on most batch systems. Since the computer is being "shared" among many customers, the cost

of computer services can be shared among the various customers. This allows even small businesses to afford computer services. Finally, communication with a time-share computer is on a real-time basis. Data can be entered and analyzed immediately, with results printed on the user terminal in a matter of seconds rather than hours. This capability of time share gives a company the ability to analyze data on sales, inventory, expenses, etc. and receive up to the minute company reports, such as cash flow or sales forecasting.

The Dartmouth BASIC system, capable of supporting many terminals, required a substantial investment in terms of processors, discs, and peripheral equipment. One of the first breakthroughs in terms of providing a low cost but powerful BASIC Language processor was the announcement by Hewlett Packard Company of their minicomputer-based 2000A Time Shared BASIC System in 1969. The 2000A System could generate up to 16 channels of BASIC for an investment of approximately \$100,000. The 2000B System generated up to 32 channels of BASIC for about the same initial investment. The only major problem was that the initial system had limited disc space and additional disc space was very expensive.

In 1972, Western Telecomputing Corporation started development of a Hewlett-Packard 2000B - based time shared BASIC system designed around a new low cost moving-head disc drive, the Hewlett Packard 7900A. Design specifications stated that the system should be capable

of supporting up to 16 channels of BASIC, meet most of HP2000B Language specifications, provide large quantities of disc storage, and have a total system cost of less than \$40,000. The version of BASIC that was developed to meet these requirements was called the WTC-10000 Time Share BASIC System.

RELEVANCE OF PREVIOUS WORK

The WTC-10000 time shared BASIC system described in this thesis, is only one of the many variations upon the Dartmouth System as conceived by Kemeny and Kurtz. A study of all manufacturers of computers would show that each one has his own version of BASIC. Some of these are so powerful and have so many extensions that they tend to obscure the simplicity of the original language.

The system described is organized almost identically to that of Hewlett Packard 2000B. That particular implementation of BASIC was selected for two reasons. The first is that the source tapes were readily available. The second, and most important, was that the 2000B system had been documented and described in sufficient detail that modifications could be easily made and tested.

ORGANIZATION OF REMAINING CHAPTERS

Chapter 2 discusses restrictions imposed upon the design of the system in terms of hardware and performance requirements. A discussion of terminology is presented to acquaint the reader with various concepts underlying the principle of time sharing.

Chapter 3 is a presentation of the operating system specifications of 1000D BASIC. It presents an overview of the system from both the user's point of view and the system operator's point of view.

Chapter 4 describes TSB System organization. It presents an overview of TSB modules and describes communication between modules. This chapter describes the organization of TSB tables and explains their functions. Finally, an account of disc organization is given, both for system discs and user/library discs.

Chapter 5 is a description of unique capabilities of 1000D which were developed to enhance system operation or make the system better fit the needs of commercial customers.

The concluding chapter 6 gives a summary and conclusion of 1000D as a solution to providing economical computing services to both commercial and educational users. Finally, several recommendations for system improvement are presented and discussed.

CHAPTER 2

OPERATING SYSTEM REQUIREMENTS

In designing a computer operating system, two types of problems must be solved.² The first type deals with intrinsic problems or those which are inherent in the design itself. The designer must look at the function the software is to serve and evaluate the requirements needed to implement that function. A design consideration of 1000D BASIC was that it should perform a useful function for both commercial users and educational users. This decision immediately created several design problems. Commercial customers needed large files for storage of data and would be using programs requiring significant compute times. Increased file access was needed and this would tend to tie the system up. Educational users demanded rapid response time to programs which would be mostly input/output bound. Access to large data banks would not be required. Finally, a system of user security would be required to prevent unauthorized access to a user's private files by another user.

The second type of problem occurs when one attempts to solve the first, that is technological problems arise. These problems

² Walsh, Larry, 2000C Course Notes, p. 10.

can be looked upon as forming the constraints with which the system designer is forced to work. For example, limiting the amount of computer core memory implies an upper limit to user program length. Not all of the system coding may fit in memory so techniques must be derived to move blocks of coding into core when they are to be executed. These constraints force the system designer to carefully consider various algorithms for making maximum use of the system resources.

If the designer has decided upon a system language, BASIC in this case, communication with the system must be considered. Two levels of communication are required.³ One is a command language. A processor for this language is then required. This processor is responsible for handling system access, for control of system resources, and for file and program control. The second level of communication is that of a program language. An interpreter for that language is required. This is the processor which executes program statements that control file manipulations, carries out calculations, and perform specific operations on data.

Other technological problems must be solved while planning the operating system. The hardware system must be separated by software procedures into a number of virtual processors for

³ Ibid., p. 10.

simultaneous use by many users. Most importantly, such sharing of resources must be done in a way that makes sharing of resources "invisible" to all of the users. This places the requirement that the system be highly organized.

Sharing of resources implies that a master program must exist whose function is to determine which user will be allowed to execute a command or perform a calculation. Such a program is referred to as a scheduler. The time this program is running rather than a user program is called overhead time. Overhead is the time the system spends performing routines designed to implement the sharing of resources. It is time lost to the users since their programs are idle when the scheduler is operating. It is important that the overhead time be kept to an absolute minimum in a well organized system. Another function of the scheduler is to maximize utilization of system resources. This means the scheduler should attempt to keep each hardware subsystem busy and insure that the system does not "bottleneck" because one resource is overly busy while others remain idle.

Response time or the amount of time required for the system to respond to a user command will increase as system utilization increases. The system must wait for one user to complete a task before it can allow a new user to start its own task. The technique of "time-slicing" allows the computer to implement resource

sharing among the users and reduce response time. This procedure enables the computer to set an upper limit on the amount of time a user's program may execute in core. If the program exceeds this limit it is written back onto a disc from core and the next user's program is read from the disc into core and allowed to execute.

1000D SYSTEM SPECIFICATIONS

WTC-1000D BASIC was designed to run on a Hewlett Packard 2100 series computer. This section presents some of the design specifications and constraints which were placed on 1000D. Appendix A contains a complete list of all hardware required to operate the system.

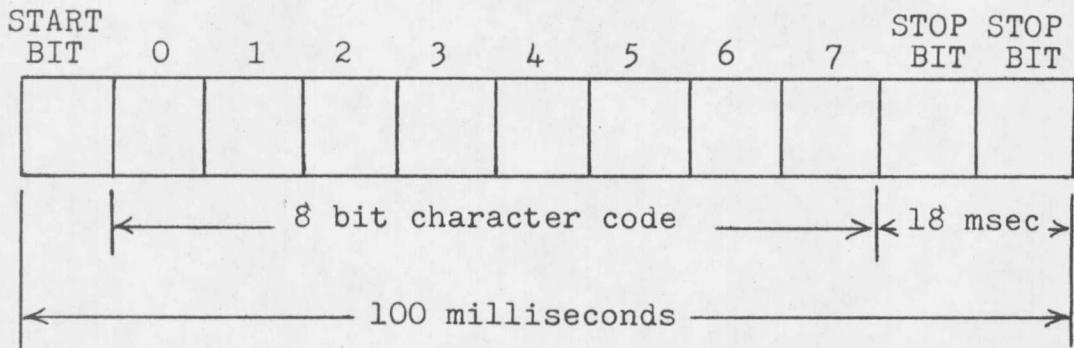
1000D was designed as a small-scale time-share system capable of supporting up to 16 simultaneous users. In order to keep response time of the system to less than five seconds, data communication transfer rate was set to be in the 10 to 30 character per second range. Faster rates would force the scheduler to spend most of its time character processing, leaving little time to execute user programs.

The most common time-share terminal in use at the time of system design was the ASR-33 teletype which operated at 10 characters per second (110 baud). This terminal communicated in a bit-serial

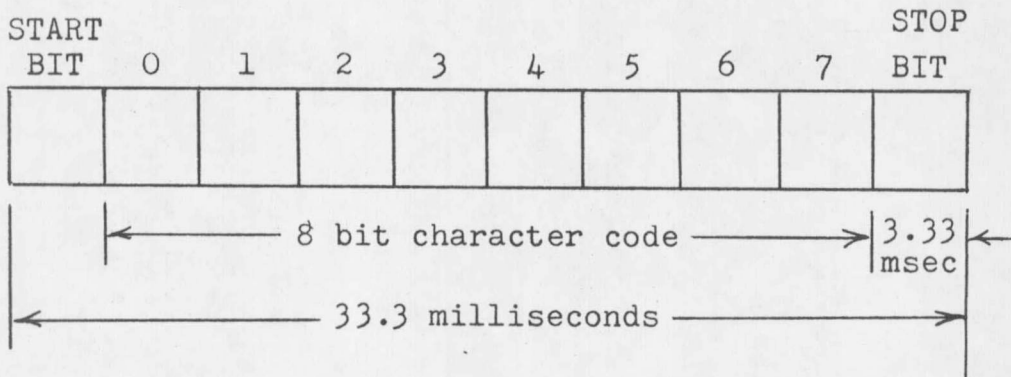
ASCII code. ASCII code was also the method of character representation internally within the HP 2100 series computers. Hence, no code conversion was required by the input/output processor. Figure 1 gives the format of the data words generated by a device operating at 10 characters per second and by one operating at 30 characters per second.

The input/output processing routine required at the worst case .15 milliseconds to process a single character. At an input speed of 10 characters per second, the routine has two bit lengths to process a character without loss of data, or 18 milliseconds. It is obvious that at least 100 channels could be processed at this speed. For terminals operating at 30 characters per second, timing is more critical. The computer has only one bit length or 3.33 milliseconds to process a character from each channel without loss of data. Calculations show that the computer can process at best, only 22 channels at 30 characters per second. Therefore, a full system of high speed terminals would tend to run much slower than a full system of slow speed terminals since most of the computer's time would be spent processing characters.

Since all communication from users to the system would be over telephone connections, a second design consideration was to allow system control of the telephone circuits. There were three reasons for doing so. First, a person dialing a wrong number could



Data Word at 10 Characters per Second.



Data Word at 30 Characters per Second.

Figure 1. Terminal Data Formats.

tie up one of the channels, preventing others from accessing the computer. To solve this problem, the computer allows only a certain number of seconds for the caller to log on the system. If he fails to do so, the line is disconnected. Second, if the phones are on a rotary call-in basis, a user can never be certain he will be able to access a particular channel if he is accidentally disconnected. A data set monitor in the system will log the user off the computer in the event his telephone carrier signal is lost. Third, a user's telephone is automatically placed back on hook when he logs off, making the channel immediately available to other users.

The amount of user space available for programming is very important to the customer and system designer. A customer would like to have unlimited program space. The system designer is aware that longer program areas mean the system will run slower since it takes longer to swap users in and out of core. A compromise was made in 1000D setting maximum program length to 4500 words, approximately 500 BASIC statements. For longer programs, the user was given the ability to "chain" programs together, resulting in virtually unlimited program length.

A final design consideration was that of selecting an optimum amount of disc storage. Fixed head discs were ruled out since they were very expensive and provided limited storage. However, access time was very fast, in the 5 millisecond range.

A decision was made to utilize the Hewlett Packard 7900 and 7901 moving-head disc drives. Access time was in the 35 millisecond range, fast for a moving head disc, and storage capacity of a single disc was roughly four times that of a fixed head disc. Each moving-head disc channel provided 203 tracks of 6144 words per track for a total channel capacity of over 1.2 million words. This was enough storage for approximately 960 medium-sized programs. 1000D allows up to four disc channels for a maximum capacity of over 4.8 million words or approximately 3840 programs.

CHAPTER III

OPERATING SYSTEM PROCEDURES

The language processor of 1000D BASIC is the same as that for HP2000B.⁴ This is an extended version of the original Dartmouth BASIC as first described by Kemeney and Kurtz.⁵ This chapter summarizes the main features of BASIC from both the user's and operator's view.

USER SOFTWARE CAPABILITIES

Extended BASIC enables the user to perform complex matrix operations in a single statement. Table 1 lists these operations and defines their function.

The working size of a matrix is defined by the program through the use of DIMENSION statements. For example, DIM A (5,3), creates a matrix of five rows and three columns. These are maximum limits which may be dynamically redefined in the program using the matrix IDN, ZER, CON, READ, or INPUT functions. In the above example, MAT

⁴ Hewlett Packard Company, 2000B: A Guide to Time Shared BASIC, 205 pp.

⁵ Kemeney, John G., and Kurtz, Thomas E., BASIC Programming, 122 pp.

TABLE 1. BASIC MATRIX OPERATIONS.

Operation	Example	Description
Addition	MAT C=A+B	Add matrix B to matrix A. Put result in matrix C.
Subtraction	MAT C=A-B	Subtract matrix B from Matrix A. Result in matrix C.
Multiplication	MAT C=A*B	Multiply matrix A by matrix B. Multiply matrix B by scalar A. Result in matrix C.
Inversion	MAT C=INV(A)	Generates matrix C as the inverse matrix of A.
Transposition	MAT C=TRN(B)	Generate C as the transpose matrix of B.
ZERO	MAT A=ZER	Set all elements of matrix A equal to zero.
Identity	MAT A=IDN	Generates A as the identity matrix.
Initialize	MAT B=CON	Sets all elements of matrix B to one.
READ	MAT READ C	Reads data from program DATA statements into matrix C.
Input	MAT INPUT B	Reads in elements of matrix B from user terminal.
Print	MAT PRINT A	Causes all elements of matrix A to be printed on user terminal.

Note: MAT READ may also input data from a user disc data file. MAT PRINT may write data to a user disc data file.

`A = CON (2,2)`, redefines A to be a matrix of two rows and two columns. Matrices can only be redefined as being smaller than that set up by the DIM statement. The maximum number of elements allowed in a matrix is 2000. An array of more than 2000 elements will produce an error condition.

BASIC contains special variables and language elements for manipulating string quantities. These are alphanumeric values rather than numeric values. String variables are defined in the system by declaring a single letter (A through Z) followed by the dollar sign (\$). The physical length of the string variable must be declared in a dimension statement. For example, `DIM A$(36)`, sets up a string variable A\$ of 36 elements. Maximum string length is 72 elements. A string value is a set of from 1 to 72 characters enclosed in quotes. Most printing and non-printing characters of the ASCII code set are valid string characters. During execution of a program, the logical length of the string is set. This is the count of the actual number of characters in the string. For example, the statement, `A$ = "SAMPLE STRING"`, has a logical length of 13 characters even though the physical length of A\$ was specified as 36 elements.

Substrings may be referenced by the use of subscripted variables. In the above example, `A$(2,6)` gives the second through sixth character or "AMPLE". BASIC allows string, substrings, and string variables to be used with relational operators. They are com-

pared and ordered as entries are in a dictionary. For example, the following statements may be written:

```
100 IF A$ = "TEST" THEN 400
```

```
200 IF A$(4,6) ≤ B$(7,8) THEN 500
```

The first statement will cause the program to go to statement 400 only if string A\$ has four elements which are exactly "TEST".

The second example will go to 500 if the fourth and fifth elements of A\$ are alphabetically less than the seventh and eighth elements of B\$. Thus, BASIC allows string values to be read, printed, compared, and sorted with very little programming effort required.

Two special features of extended BASIC are the ability to specify length of the terminal print line and ability to access the system clock. Use of the WIDTH command allows the user to inform the system that the terminal may print only 20 columns of data per line or up to 132 columns per line. Use of the TIM(X) function allows a program to obtain the current minute, hour, julian day, and year from the system real-time clock. This information is useful in identifying multiple runs of the same program or for reporting run times of various programs.

A serious limitation of 1000D BASIC is its rather small user program area of 4500 words. Since some of this area is used for program overhead, program lengths generally have a maximum length of 3500 words. This is sufficient to store a BASIC program of roughly

500 statements. If arrays are dimensioned, the user loses two words of program space for each element in the array. Commercial programs are typically very long and utilize large arrays for data storage. Use of the BASIC CHAIN statement allows the programmer to divide one large program into many smaller segments which are stored on the disc. As each segment of coding is needed, it is called off the disc by the currently executing program. Parameters and other values may be passed from one program to the next through COMMON or data files. Use of the CHAIN statement gives BASIC the capability of handling programs of virtually unlimited length.

All users have access to two levels of libraries. The first level is that of the system public library. These are programs and files saved on the disc by the master user, A000. Users may get these programs from the disc, list and punch them, modify them, and execute them. User A000 may protect any program in the public library. This makes them "run-only" to all other users. Such a procedure would be used to prevent unauthorized copying of proprietary software. A user also has access to a second level of libraries, an individual user library. Programs and files stored in an individual library are accessible only by the owner.

The most powerful program function of BASIC is its versatile file handling capability. Users may create data files on the disc to allow for direct manipulation of large volumes of data. Files

may be of two forms: sequential or random access.

Sequential files are treated as serial access storage devices. A program starts reading or writing data as a serial list of data items from the physical beginning of the file and continues on an element by element basis until either the physical end of file is encountered or an end of data condition occurs. During a serial file write operation, the new data element is written immediately following the previous data element. To retrieve an item from the file, the program must start at the beginning of the file and read through all items until it comes to the desired data element.

Files may also be used in a random access mode. This enables the program to break the file into a series of logical subfiles that can be modified independently of each other. Each subfile is a block of 128 words.

Data may be written into a file in either numeric form or string variable form. String and numeric data may be randomly mixed in any file. Use of the TYP(X) function enables the program to identify what type of data the next file item is during a file read. Numeric data requires two words of storage per number whereas two string characters may be stored in a single word.

File lengths are variable from 1 to 48 sectors. Each sector holds 128 words of data. Maximum file length is 6144 words. Each program may reference up to 16 files at one time. This is equivalent

to having immediate access to 98,304 words of additional data storage. Any program may access an unlimited number of files during execution through the ASSIGN statement. This statement allows a new file to be referenced in the place of a previously defined file in the program.

The system allows users simultaneous access of files. The first user program to reference a file is allowed read/write privileges. All other users accessing the same file are allowed read-only access. Users whose account identification codes begin with an A (A000 through A999) always are allowed read/write access to the same files.

This section has described some of the more important capabilities of extended BASIC. A complete description of user capabilities may be found in the WTC-1000D BASIC user's guide.

SYSTEM OPERATOR SOFTWARE CAPABILITIES

The system operator is responsible for maintaining the TSB system, adding new accounts, modifying existing accounts, and removing old accounts from the system. These functions are accomplished through the system operator's console using special system commands and through the user account A000.

System control through user A000 is restricted to two functions.

Only programs stored under that account become a part of the public library, accessible to all users. Placing programs and files in this library is the primary function of A000. The second function involves maintenance of a particular program in the public library, the "HELLO" program. This program is automatically run on the user's terminal following a successful attempt to log on TSB. HELLO is a standard BASIC program whose function is to describe the system, give the current date and time, and print important messages from the system operator to the users.

Primary system control takes place through operator commands entered on the system console. One function performed is that of limiting access of system resources by the users. The system operator controls resource allocation to up to 562 user accounts, the maximum allowed by the system. This is accomplished by setting restrictions on each account through operator commands. Each account is assigned a unique ID code and password required for system access. The operator also specifies a maximum number of minutes of connect time allowed each user and additionally, a maximum number of sectors of disc storage the user is allowed. The password, maximum connect time and maximum disc storage allowed may be modified at any time by the system operator. If required, an entire account may be removed from the system by an operator command.

The system operator may request TSB to generate reports

useful for evaluating system utilization. The ROSTER command generates a listing of currently active channels as shown in Figure 2.

```
ROSTER
..... S231 ..... A000 A000 ..... C246
K341 ..... ..... ..... B260 ..... .....
```

Figure 2. Example of ROSTER Command

The first line lists, from left to right, the users logged on terminals 0 through 7. User S231 is on channel 2, user A000 is on channels 4 and 5, and user C246 is on channel 7. The second line lists users on channels 8 through 15. User K341 is on channel 8. User B260 is on channel 13. Four dots indicate an inactive channel.

A listing of all accounts with the total connect time used to date and disc space used to date may be obtained with the REPORT command. This command is used to generate a report for account purposes. An example of the REPORT command output is given in Figure 3.

