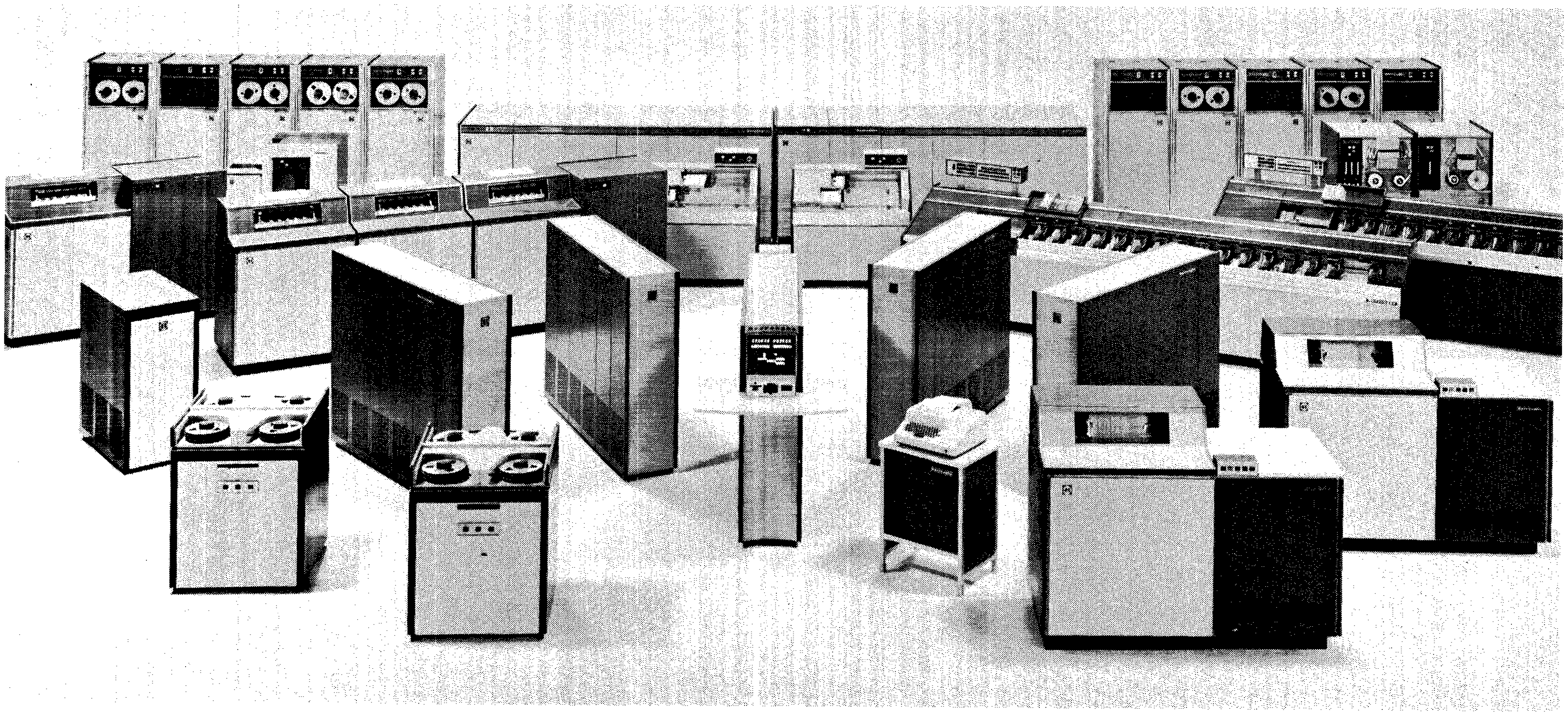
Burroughs

B 2500 and B 3500 SYSTEMS

CHARACTERISTICS MANUAL



Burroughs
B 2500 and B 3500 SYSTEMS CHARACTERISTICS MANUAL

Burroughs Corporation
Detroit, Michigan 48232

Printed in U.S. America 3-70 1025517

This edition incorporates the information released under the following PCN:
1025517 - 001.

Correspondence regarding this document should be forwarded using the Remarks Form at the back of the manual, or may be addressed directly to Systems Documentation, Sales Technical Services, Burroughs Corporation, 6071 Second Avenue, Detroit, Michigan 48232.

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont)

TABLE OF CONTENTS (Cont)

LIST OF ILLUSTRATIONS

LIST OF TABLES

INTRODUCTION

This manual contains a description of the B 2500 and B 3500 Systems operation. It is intended to provide technical information and reference for those directly associated with data processing – data processing managers, systems analysts, programmers, and management personnel acquainted with the concepts of electronic data processing. It is not intended as a teaching manual, programing primer, or operation manual.

The B 2500 and B 3500 Information Processing Systems are modular systems incorporating monolithic circuitry, high speed storage, and special hardware features designed to permit complete integration of hardware capabilities with software systems. The design philosophy of the B 2500 and B 3500 Systems is unique. The hardware and supporting software were designed in parallel, instead of the traditional approach of building a machine and then determining what software is needed for system support. Through this design concept, the hardware contains the necessary logic to allow the software to be written and executed in the most efficient manner and, conversely, the software makes optimum use of all the capabilities of the hardware. The fact that the B 2500 and B 3500 function as systems, rather than advanced sets of hardware, has determined the organization of this manual.

The programing languages, COBOL, FORTRAN, Assemblers, and Report Program Generator are presented first (section 1) because they constitute the communications link between the user and the system. Through their use, the problem is stated and the method of receipt of the ultimate solution is defined. The next three sections, System Description, System Organization, and System Operation discuss the reasons for, and the way in which the B 2500 and B 3500 function as systems. Overall coordination and control of processing, so important to total production through maximum use of the components of the B 2500 and B 3500 Systems, is supplied by the Master Control Program (MCP) which is described in section 5. The operational characteristics of the peripheral components which implement this unique system are specified in section 6.

The B 2500 and B 3500 Systems were designed as a complete system, combining components and built-in programing aids to bring the user simplified programing, ease of operation, and complete freedom of system expansion.

The programmer is allowed to choose the language best suited to his current needs. For normal business applications he may use the English narrative statements of COBOL, or if the current problem is of a mathematical nature, he may use the formula notation of FORTRAN. Also for those programs which are of a generative nature requiring extensive instruction modification based on parameters furnished at execute time, the programmer is furnished with an assembler language which allows for complete flexibility of manipulation on an instruction-for-instruction level.

In the area of operations, as in programing, alternatives are provided. Operator intervention may be nearly eliminated through the use of the Master Control Program (MCP) which provides for complete management of the system. The MCP is a comprehensive operating system housed either on the disk file or system memory of the B 2500 and B 3500 Systems, and provides for simultaneous input, output and compute operations and time sharing. By controlling the sequence of processing, initiating all input/output operations, and providing automatic handling procedures, the MCP can obtain maximum usage of the system components. Thus, the system achieves greater production and efficiency.

It is this complete flexibility of programing and control of the processing pattern which provides the B 2500 and B 3500 Systems with such smooth growth potential. The user may start with the minimum configuration to process his current work load and expand at will with small increments as his volume increases, or as it becomes necessary to run more and larger programs. As more components and larger memories become available on a system, the MCP will automatically make use of them, through multiprograming, gaining increased system production and efficiency.

SECTION 1

PROGRAMING LANGUAGES

GENERAL

Three levels of programing languages are available with the B 2500 and B 3500 Systems: Problem-oriented compiler languages, report-oriented generative languages, and a machine-oriented assembler language. This allows the programmer to choose a language directly suited to the solution of his problem. Two problem-oriented languages are available: COBOL for the solution of business data processing applications and FORTRAN for use in solving arithmetic problems. The machine-oriented Advanced Assembler language produces programs that are executed under control of the Master Control Program. The report-oriented generative language is a single compiler capable of producing programs for the Master Control system.

PROBLEM-ORIENTED LANGUAGES

Problem-oriented languages have many advantages. They allow the programmer to state a problem in a language directly adapted to a given problem. Also, programs are less machine dependent and allow greater flexibility for program modification and exchange between users of different types of equipment. The B 2500 and B 3500 Systems lend themselves well to compiler implementation. For example, the Edit command of the Systems includes within a single instruction all of the editing functions required in COBOL, but is not limited to them. Three advantages of this design as compared to conventional machine and compiler techniques are:

a. Reduced programing time.

b. Reduced time to compile and run the object program.

c. Simplified debugging and program maintenance.

Data Processing Language (COBOL)

Knowledgeable computer users, while desiring the advantages of automatic programing systems, became alarmed because many different languages were being developed and put into use. In May, 1959, a meeting was held with representatives from industry, government, and computer manufacturers in attendance. They agreed that the development of one common language tailored for business use was both desirable and feasible. There were three major requirements to be met:

a. The need to translate existing solutions to business problems efficiently from one type of computer to another with minimum conversion costs.

b. The need for program documentation in a form allowing changes and additions with minimum time and expense.

c. The need for reducing the time requirements for training of programing personnel.

A report outlining initial specifications of a COmmon Business Oriented Language (COBOL) was published in April, 1960. In February, 1961, the COBOL Maintenance Committee – a group of 12 Electronic Data Processing Equipment Manufacturers, including Burroughs Corporation, and 10 interested industrial users – met and completed revisions to the original COBOL specifications. This revised version of COBOL as adopted by the United States of America Standards Institute is one of the problem-oriented languages of the Burroughs B 2500 and B 3500 Systems. The B 2500/B 3500 COBOL Compiler, as well as the resultant object program, can be multiprogramed in an unrestricted environment.

Computational Language (FORTRAN)

FORmula TRANslation (FORTRAN) was originally designed for the IBM Corporation, but has been widely accepted both by computer users and manufacturers. Because of this wide acceptance, the United States of America Standards Institutes FORTRAN IV version was selected as the computational language of the Burroughs B 2500 and B 3500 Systems.

When a scientific programmer attempts to solve an arithmetic problem, his thoughts are channeled in terms of algebraic notation, or a formula for the solution. FORTRAN allows the problem to be written with formula notation very similar to normal arithmetic. This allows communication with the machine in a language more applicable to the problem and more familiar to the programmer, thus reducing the time required to write and debug programs. The Burroughs B 2500 and B 3500 FORTRAN Compiler, as well as the resultant object program, can be multiprogramed in an unrestricted environment.

ASSEMBLER LANGUAGE

For those problems of a generative nature requiring extensive instruction modification, according to parametric input at execute time, the B 2500 and B 3500 Systems are provided with a machine-oriented assembler language. The Systems Assembler language permits programs to be coded in symbolic form using mnemonics and symbolic addressing.

The assembler is constructed very similar to the Burroughs B 2500/B 3500 COBOL Compiler construction. Data declarations are required and program segmentation is allowed. Due to the macro instructions at assembly time, and the MCP at execute time, the programmer need not concern himself with the detailed problems of input/output and error conditions. However, with all these advantages, he is still allowed to program at the machine level with complete flexibility of instruction modification, indexing, incrementation, and character or bit manipulation. In general, the programmer is provided with the generative macro capabilities of compilers and still allowed the flexibility inherent in the assembler. The Burroughs B 2500 and B 3500 Advanced Assembler as well as the resultant object program, can be multiprogramed in an unrestricted environment.

REPORT PROGRAM GENERATOR

The Report Program Generator provided with the B 2500 and B 3500 Systems allows the user to produce many programs in minimum time with optimum efficiency. It compiles relatively complete symbolic programs from a brief, simplified, problem-oriented language similar to COBOL. Program generation is fast and the resulting programs will normally run at the rated speeds of the designated peripheral equipment. The Report Program Generator creates an assembler source program which is compatible with the M.C.P. The Burroughs B 2500 and B 3500 Report Program Generator, as well as the resultant object programs themselves, can be multiprogramed in an unrestricted environment on an MCP controlled system.

COMPILATION OF A SOURCE PROGRAM

When a program has been written in one of the languages acceptable to the B 2500 and B 3500 Systems, it must be converted from its source language to the internal machine language. This is accomplished for the user by the applicable Compiler, Generator and/or Assembler program. The source program is punched into either cards or paper tape and loaded into the system. The object program is compiled or assembled with any of the following optional outputs:

a. Object program compiled to the Disk File Library with immediate execution (compile and go under control of the MCP).

b. Object program compiled to Disk File Library for future execution (compile for library under control of the MCP).

The compiled object program consists of:

a. Program parameter record containing the object program's memory requirement, and the location of the segment dictionary.

b. A File Information Block containing information related to each file used in the object program, as well as data areas for the files.

c. Main object program body.

d. Object program segments as applicable.

SECTION 2

SYSTEM DESCRIPTION

GENERAL

The following paragraphs describe the processor and related control components which give the B 2500 and B 3500 Systems their unusual capabilities.

CENTRAL CONTROL

All peripheral unit operations are independent of each other and of the processor; therefore, any combination of simultaneous input/output and compute operations is possible. Each peripheral device, along with the processor, competes against all other peripherals for use of core memory. When a particular device wants a memory access, it makes an access request to central control, which grants the request as soon as all requests from higher priority devices have been satisfied. Because of the independent and concurrent operation of all devices, a request for memory access may be one of several. A rather sophisticated priority system is handled in a very simple way within the central control. In the event of multiple access requests, the unit with the highest priority is granted access first. Because of the unique structure of the priority control, this is accomplished without the necessity of a scan operation. The priority in determining memory access for the various devices can be established or changed by a field engineering adjustment, with the exception of the processor which has the lowest priority. The interrelationship of the central control unit to the various components of the B 2500 and B 3500 Systems is shown in figure 2-1.

CORE MEMORY

At this point, it is necessary to make a distinction between the two system groups which constitute the B 2500 and B 3500 Systems. The basic differences between the two systems are total core memory availability, number of I/O channels available, and internal operational speeds. Since core memory is one of the areas of difference, the two system groups will be covered separately at this point.

B 2500

The B 2500 System is expandable in memory size from 10,000 characters (bytes) to 120,000 characters. The core memory in this system requires a memory cycle time of two microseconds for every two bytes accessed.

B 3500

The B 3500 System is expandable in memory size to 500,000 characters (bytes). This expandability is in increments of 10,000, up to 90,000; in increments of 30,000, from 90,000 to 240,000; in increments of 60,000, from 240,000 through 360,000; in an increment of 90,000, from 360,000 through 450,000 and in a final increment of 50,000 for a total of 500,000. The core memory in this system requires a memory cycle time of one microsecond for every two bytes accessed.

ADDRESS MEMORY

While the address memory is physically located within the processor, it may be addressed by other components of the system independent of the processor. For this reason it is covered as a separate control component. The purpose of address memory is to replace many of the costly "hard registers" of conventional systems and at the same time limit core memory access, thus greatly increasing the processing speed of the system.

Minimum address memory consists of 24 6-digit cells. Eight of these are assigned to the processor for its use, and two are assigned for each I/O channel. Address memory is expandable in increments of 12 cells up to a maximum of 120. During execution, the processor addresses core memory with words from address memory so that memory accesses are not required for information relative to the command itself during execution; that is, accesses during the execution phase are for data only. I/O controls use address memory in a similar fashion to reduce core accesses. A word from address memory requires an access time of only 100 nanoseconds for both the B 2500 and the B 3500.

THE INPUT/OUTPUT SYSTEM

The input/output system consists of the peripheral control units and their related input/output chan-

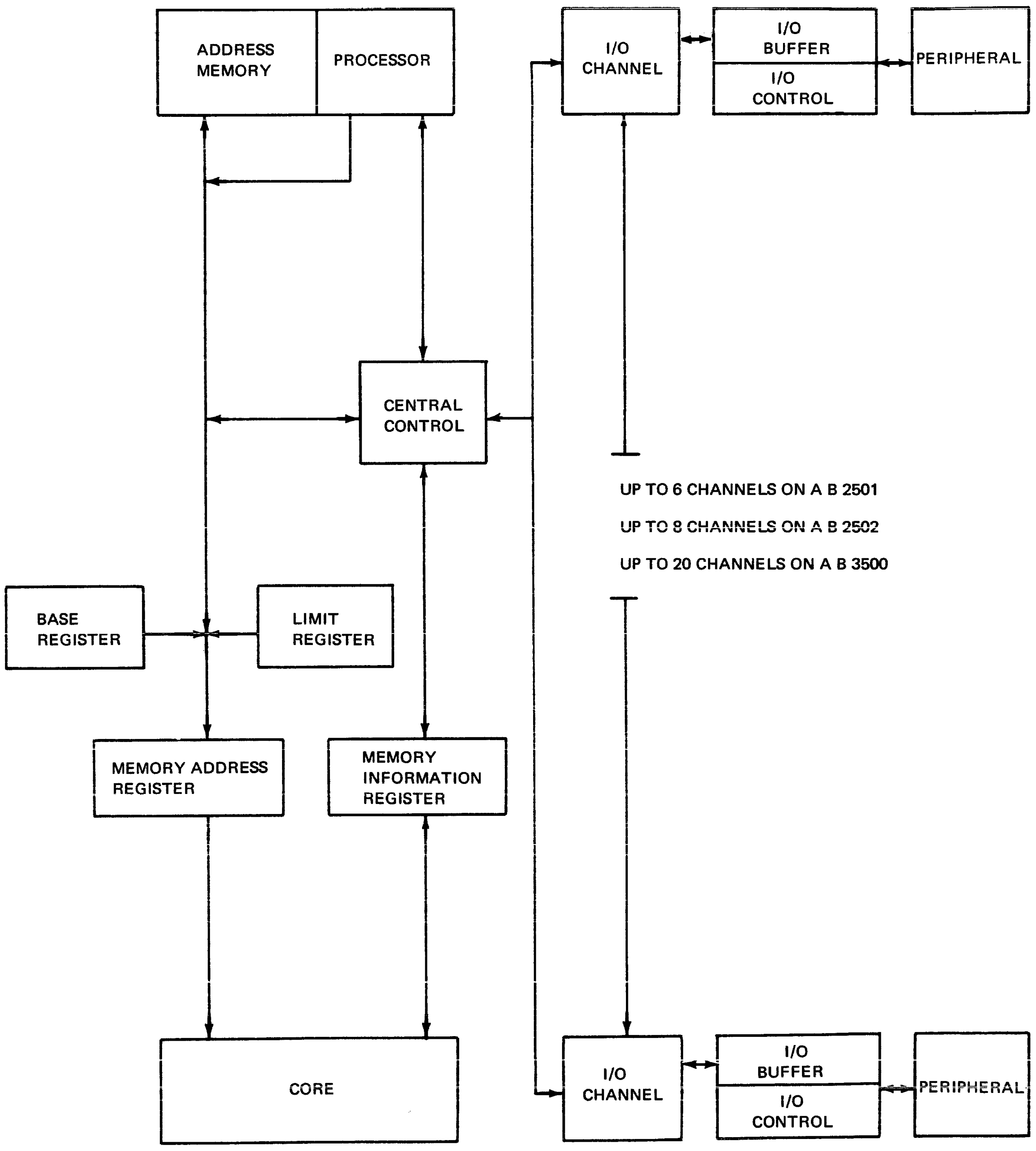


Figure 2 - 1. **Interrelationship of B 2500 and B 3500 Systems Components to Central Control**

nels and operates independently of the processor. The processor issues a command to the I/O system and then proceeds independently until the I/O system completes the operation and interrupts the processor. I/O operations are independent of each other and any or all I/O channels may operate simultaneously. The I/O system time-shares core memory and address memory with the processor, under control of the central control unit. There is an I/O channel and a peripheral control for each peripheral unit, or group of units of the same type. For example, a card reader requires an individual I/O channel and peripheral control, while up to 10 magnetic tape units may use only one I/O channel and peripheral control. Also, two or more I/O channels may be floated between groups of peripheral units of the same type. For example, assume two I/O channels are assigned to the same five tape units. This means any of the five units may use either channel that is available, allowing for the simultaneous reading of any one unit while writing on any other unit.

The I/O channels are of two basic types: type A channels and type B channels. Type B channels permit the use of high-speed peripheral units which transfer two characters simultaneously (in parallel) while type A channels permit the use of low speed peripheral control units which transfer only one character at a time (serially). Different types of peripheral units require different types of channels. Table 2-1 shows a list of peripheral unit types and the I/O channel type required for each.

The B 2500 and B 3500 Systems differ in I/O channel capacity. B 2500 Systems may have from four to eight I/O channels with a maximum of five of these being type B channels. B 3500 Systems may have up to 20 I/O channels with a maximum of 10 type B channels.

THE PROCESSOR

The processor contains the arithmetic units and the logical controls of the system. Object programs are floated in memory through the use of a base register thus allowing several programs to be resident in memory at one time. The Master Control Program (MCP) will set the base register for one program, retrieve it, and turn control over to that program. After handling an interrupt, the MCP may reset the base register and turn control over to another program. All addresses are compiled for an object program as base relative to zero and the programmer need not concern himself with the location of his program in core memory at a given time, or with the type of other programs which are present in memory during its execution. The MCP will call in a given program when there is memory available to meet that program's requirements by analyzing contiguous core available and automatically pushing other operating programs down in memory locations to facilitate the new program, and will set the base register to the location to which the program is to be assigned.

Table 2-1

Type I/O Channel Requirements

Peripheral Unit	Channel A	Channel B
Magnetic Tape Unit		X
Card Reader	X	
Card Punch	X	
Printer (Externally Buffered)	X	
Printer (Internally Buffered)	X	
Multiple Tape Lister	X	
Disk File		X
System Memory		X
MICR Reader Sorter	X	
Console Printer	X	
Paper Tape Reader	X	
Paper Tape Punch	X	
Data Communications Devices		X

Along with the base register there is also a program limit register so that memory can be protected from programs attempting to access areas outside of their boundaries. Any program attempting to access memory below its base register setting or above its limit register setting will be automatically discontinued and the operator will be notified.

There are three index registers available to every program in the memory mix.
Indirect addressing and indirect field length are standard features of the system and are allowable to any level.

A more detailed description of the processor is given in section 3 which covers processor organization, and in section 4 which explains the manner in which the system operates.

SINGLE-LINE AND MULTI-LINE CONTROLS

The single-line and the multi-line controls are functionally equivalent, except that the single-line control services one communications line, whereas the multi-line control services a number of communications lines on a time-shared basis. Both the single-line control and the multi-line control can be adapted to accommodate a variety of data sets and remote terminals.

No translation is provided for any code by either control. Code translation is provided for by the Data Communications MCP.

All controls include the capability to respond to a Read Address instruction from the processor by allowing the transfer of the contents of either word of its associated address memory to the processor. The invalid I/O descriptor bit of the processor is set ON in the event that the instruction is received by a busy channel. A multi-line channel is not considered busy after returning a channel result descriptor.

SECTION 3
SYSTEM ORGANIZATION

GENERAL

For further understanding of the total B 2500 and B 3500 System concept, hardware and software integration, this section will describe the way in which an instruction is executed, the methods of data representation, the format and techniques of instructions, and the logical units within the processor. These capabilities allow the MCP to control the system and free the programmer from the tedious responsibility of input/output and system control and also from the necessity of detailed scheduling.

READ-ONLY STORAGE

Much of the traditional hard logic of conventional systems is replaced by a device known as read-only storage. Read-only storage is an extremely fast, resistive type memory which is wired with interpretive routines that are executed at hardware level within 100 nanoseconds. These routines are called microprograms and control all of the actions taken by the processor, namely:

a. Memory reads and writes for the processor.

b. Transmission of data from register to register within the processor.

c. Loading and unloading the processor's eight words of address memory.

d. Counting and setting of all the various registers.

e. The initiation of I/O operations.

The microprograms are automatically initiated by the systems operation codes contained in the object program instructions as they are fetched from core memory.

DATA REPRESENTATION

The internal code of the B 2500 and B 3500 Systems is Extended Binary Coded Decimal Interchange Code (EBCDIC) which is an 8-bit alphanumeric code with no forbidden combination of bit arrangements. Because many data communication devices transmit in United States of America Standard Code for Information Interchange (USASCII), provisions are also made within the processor to directly process USASCII. This is done by a mode switch that is programmatically-selectable. In either the EBCDIC mode or USASCII mode, the processor accepts and acts upon data in the form of either digits, characters, or words.

Digits

Decimal numbers are represented by four bits. There are no forbidden combinations of the 16 possibilities, but arithmetic manipulation cannot be performed on digits greater than nine. Numeric fields may be either signed or unsigned. Where a sign is expected, it is interpreted to be a separate, leading 4-bit unit. All unsigned fields are considered plus. When an arithmetic result is specified as a signed 4-bit numeric, the system will set the leading unit to the proper internal sign configuration. The internal sign digits are as follows:

a.	EBCDIC:	plus-1100;	minus-1101.
b.	USASCII:	plus-1011;	minus-1101.

A plus will collate higher than a minus in either mode when it is specified as a sign digit.

Characters

Alphanumeric data are represented internally by eight bits. As previously stated, alphanumeric characters may be either EBCDIC or USASCII. If an arithmetic operation uses one or more fields, specifying 8-bit characters as operands, only the least significant four bits of each character are used. The most significant four bits are considered to be the numeric subset of the applicable 8-bit code. If the receiving field for the result is also specified as 8-bit characters, the most significant four bits of each character are automatically set to the numeric subset of the selected 8-bit code. In other words, if arithmetic is done on 8-bit characters, the character is assumed to be a number and only the numeric portion of that character is considered. The result field is automatically set to a number. In common data processing terms, when arithmetic is done on a character, all zone bits are stripped. All alphanumeric fields are considered to be positive.

Generalized address control capability provides ability to automatically convert a field from its 4-bit representation to an 8-bit representation, or vice versa during normal data movement from one class of field to another or during arithmetic manipulation of different field classes.

Words

Certain data movement instructions are word sensitive. A word within the B 2500 and B 3500 Systems is considered to be 16 consecutive bits (two bytes). It may be represented as either four numeric digits or two alphanumeric characters. When these instructions are used, 16 bits will be moved in parallel, in binary order. There will be no change in the internal data representation.

All information is addressed by the most significant (or left-most) digit or character. This holds true whether data are represented as digits, characters, or words. If a field is signed, the sign digit will be the digit addressed.

There is one special case of numeric representation within the B 2500 and B 3500 Systems and it is for floating-point operands which are to be used by the optional floating-point arithmetic instructions. The floating-point operands are in the 4-bit digit mode, but have a special format. The components of the floating-point format are:

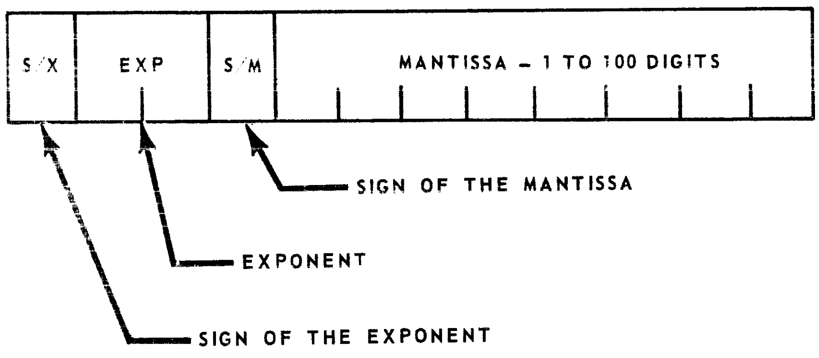
a. 1-digit sign of the exponent (S/X).

b. 2-digit exponent (EXP).

c. 1-digit sign of the mantissa (S/M).

d. Mantissa - a variable length number ranging from 1 to 100 digits.

The instruction will define the length of the mantissa. When floating-point operands are addressed, the sign of the exponent (SX) is the digit addressed. Figure 3-1 shows the format of a floating-point operand. The sign conventions are the same in fixed-point, signed-numeric representation.



NOTE

To address a floating-point field, point to the sign of the exponent.

Figure 3-1. Layout of Floating-Point Number

INSTRUCTION REPRESENTATION

All instructions are represented as 4-bit digits. Instructions are of two basic types: processor instructions and input/output instructions (called descriptors). The two types of instructions are discussed separately in the following paragraphs.

Processor Instructions

Processor instructions, with the exception of branches (a special case which will be covered separately), are variable in length from one to four syllables. That is, instructions may contain no address, one address, two addresses, or three addresses. Each syllable contains 24 bits or six digits. The first syllable contains the operation code (OP) which denotes the operation to be performed and implies the number of syllables comprising the instruction; the A field variant (AF) which specifies variable information pertaining to the A address; and the B field variant (BF) which specifies variable information pertaining to the B address. The second, third, and fourth fields, if applicable, contain addresses of data. The mode of operation, EBCDIC or USASCII, is irrelevant to code sensitivity within the instructions since the numeric portions of each code (digits 0-9) are identical and all instructions are in numeric digit representation.

OPERATION CODE

The first two 4-bit digits of an instruction are interpreted to be the operation code (OP). The operation code is indicative of the length of the instruction and triggers access to the proper string of micro-operators in read-only storage. The microprogram thus initiated will fetch the balance of the instruction and execute the proper steps to accomplish the required operation.

VARIANTS

The remaining four digits of the first instruction syllable are used as variants and are referred to as AF and BF. In arithmetic commands, the first two variant digits (AF) give the length of the field pointed to by the A address syllable and the second two variant digits (BF) give the length of the field indicated by the B address syllable. If the C address syllable is used, the length of the C field is determined by a combination of AF and BF. In some data movement instructions, AF specifies the length of the sending field and BF specifies the length of the receiving field. Since the variant fields are two digits and may range from 00 to 99, operands may be from 1 to 100 digits or characters in length (00 indicates 100). In some data movement instructions, the AF and BF variants are combined to represent values ranging from 0000 to 9999 (0000 indicates 10,000); thus, these instructions are capable of moving from 1 to 10,000 digits, characters, or words depending on the instruction being executed, in one execution of the instruction.

Figure 3-2 illustrates the first syllable of an instruction.

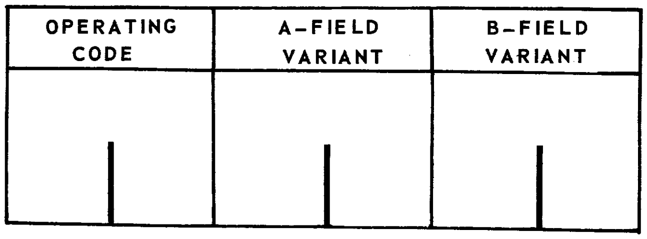
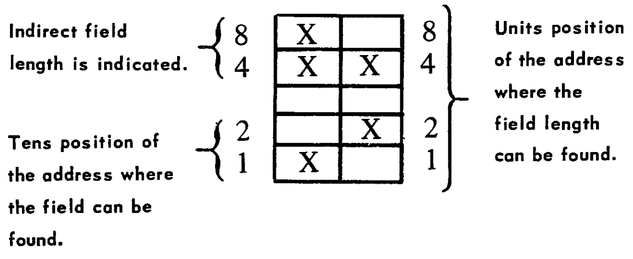


Figure 3-2. First Syllable of an Instruction

Another function available within the variant fields, when their use within an instruction specifies field length, is that of indirect field length. When the two high-order bits (8 and 4) of the most significant digit of AF or BF are ON, they indicate an indirect field length. The two low-order bits (2 and 1) plus the entire least significant digit indicate the tens and units digits of an address where the field may be found. This address must be an even number. For example:


Indirect field length is indicated.
8
4
Tens position of the address where the field can be found.
2
1
X
X X
X
X
8
4
2
1
Units position of the address where the field length can be found.

NOTE
X indicates the bit is ON.
Blank indicates OFF.

In this example, the variant reflects that an indirect address contains the field length of the instruction being executed and that the length of the field is to be found at base relative address 00016.

While it is logically possible to use indirect field length to any depth, it is obvious that the number of addresses generated by the technique is limited. The two low-order bits of the high-order digit may represent values of 0, 1, 2, or 3. Since the address must be even, the range may be from 00000 to 00038, base relative. This technique can be very useful in cases where many instructions in a program refer to the same field, and that field is variable in length. The field length may be modified in only one location. All instructions referring to that field will be so modified without necessity of any code change.

Another use of the AF variant is to indicate that the A syllable of the B 2500 and B 3500 Systems command does not contain an address, but instead contains literal data to be used directly by the command. This option is indicated by the 8-bit and the 2-bit, in the most significant digit of the AF field, being ON, with the 4-bit being OFF. A literal, as with any data, may be represented as 4-bit digits, either signed or unsigned, or as 8-bit characters. The 1-bit of the most significant digit of AF and the 8-bit of the least significant digit are used to indicate the internal representation to be used for the literal and are called controller bits which represent the following 2-bit values:

a. 0 - unsigned numeric (4-bit mode).

b. 1 - signed numeric (4-bit mode).

c. 2 - alphanumeric (8-bit mode).

d. 3 - unused.

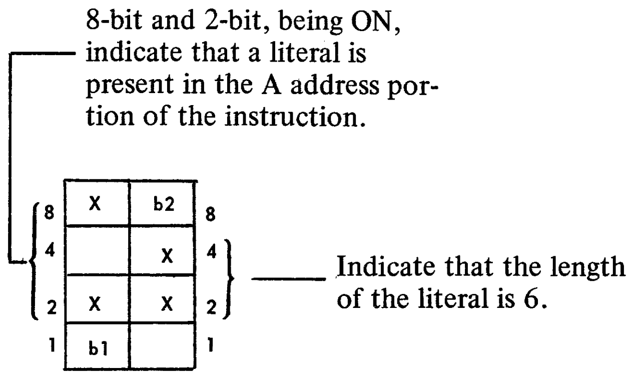
The remaining three bits of the low-order digit of AF specify the length of the literal. Since the A syllable is six digits in length, the following lengths are maximum:

a. Unsigned numeric - six 4-bit digits.

b. Signed numeric - five 4-bit digits plus a sign 4-bit digit.

c. Alphanumeric - three 8-bit (byte) characters.

For example:


8-bit and 2-bit, being ON, indicate that a literal is present in the A address portion of the instruction.
8
4
2
1
X
b2
X
X
X
b1
8
4
2
1
Indicate that the length of the literal is 6.

In the example, b1 and b2, both being OFF, indicate an unsigned numeric representation. The A syllable contains a literal of six unsigned numeric digits.

All field lengths specified for signed numeric fields reflect the length, excluding the sign. In floating-point instructions, field length reflects the length of the mantissa only.

ADDRESS SYLLABLES

As previously stated, an instruction may have none, one, two, or three addresses and all have the same format. The low-order five digits represent the base relative to zero address. The most significant digit of an address is split; the 8-bit and the 4-bit designate the index register by which it is to be incremented or decremented. There are three index registers available to each program being operated within the B 2500 and B 3500 Systems. The index register configurations are shown in table 3-1.

Table 3-1

Index Register Configurations

Configuration of 8 and 4-Bits	Value	Description
00	0	No indexing required.
01	1	Index register one.
10	2	Index register two.
11	3	Index register three.

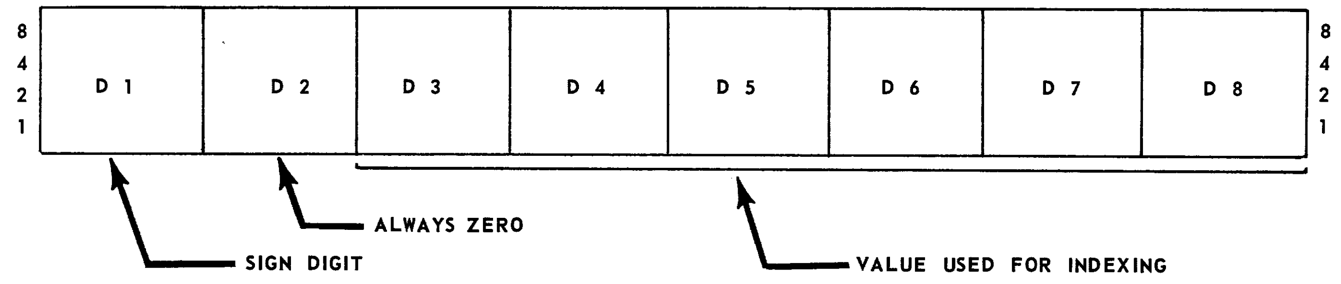


Figure 3 - 3 Format of Index Register

Each index register is eight digits in length. The most significant digit is a sign digit which allows the index register to either increment or decrement. The digit immediately to the right of the sign digit is always zero and is included for expansion purposes only. The remaining six digits contain the index value by which the address will be either incremented or decremented. Figure 3-3 illustrates the format of an index register.

The 1-bit and 2-bit of the most significant digit of an address are called the address controller which tells the system what the representation of the data addressed is, e.g., signed numeric, unsigned numeric or alphanumeric. The address controller may also specify that the contents of the addressed field is not data at all, but is an indirect address where the data may be found. This technique is known as indirect addressing and may be used to any depth. The address controller configurations are shown in table 3-2.

The indirect addressing technique may be used with any combination of index registers. For an example of indirect addressing, assume the common function of table look-up. A value within a table must be located, and once found must be referenced by several instructions. Once the value is located, the address of that value may be stored in one location and all other instructions may reference the value through indirect addressing, with no address modification necessary. Figure 3-4 is an illustration of the format of an address syllable, and figure 3-5 is an illustration of a maximum size, 4-syllable instruction.

Table 3 - 2
Address Controller Configurations

Configuration of the 1- Bit and 2- Bit	Value	Description
00	0	Unsigned 4-bit format
01	1	Signed 4-bit format
10	2	Alphanumeric 8-bit format
11	3	Indirect address

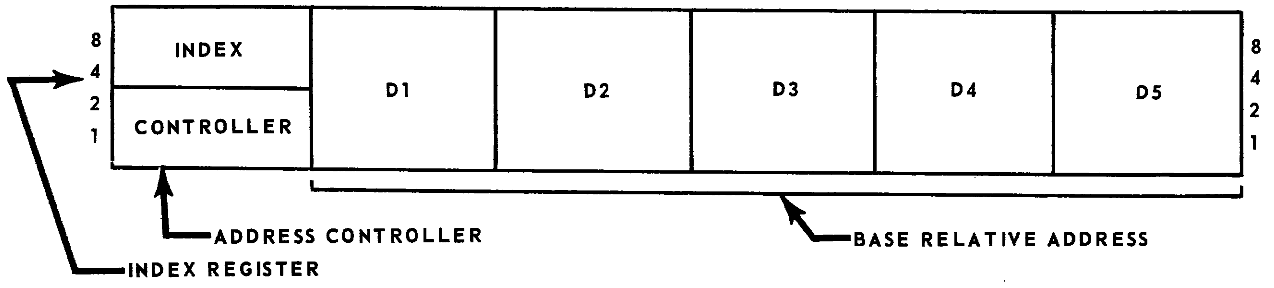


Figure 3–4. Address Syllable of Instruction

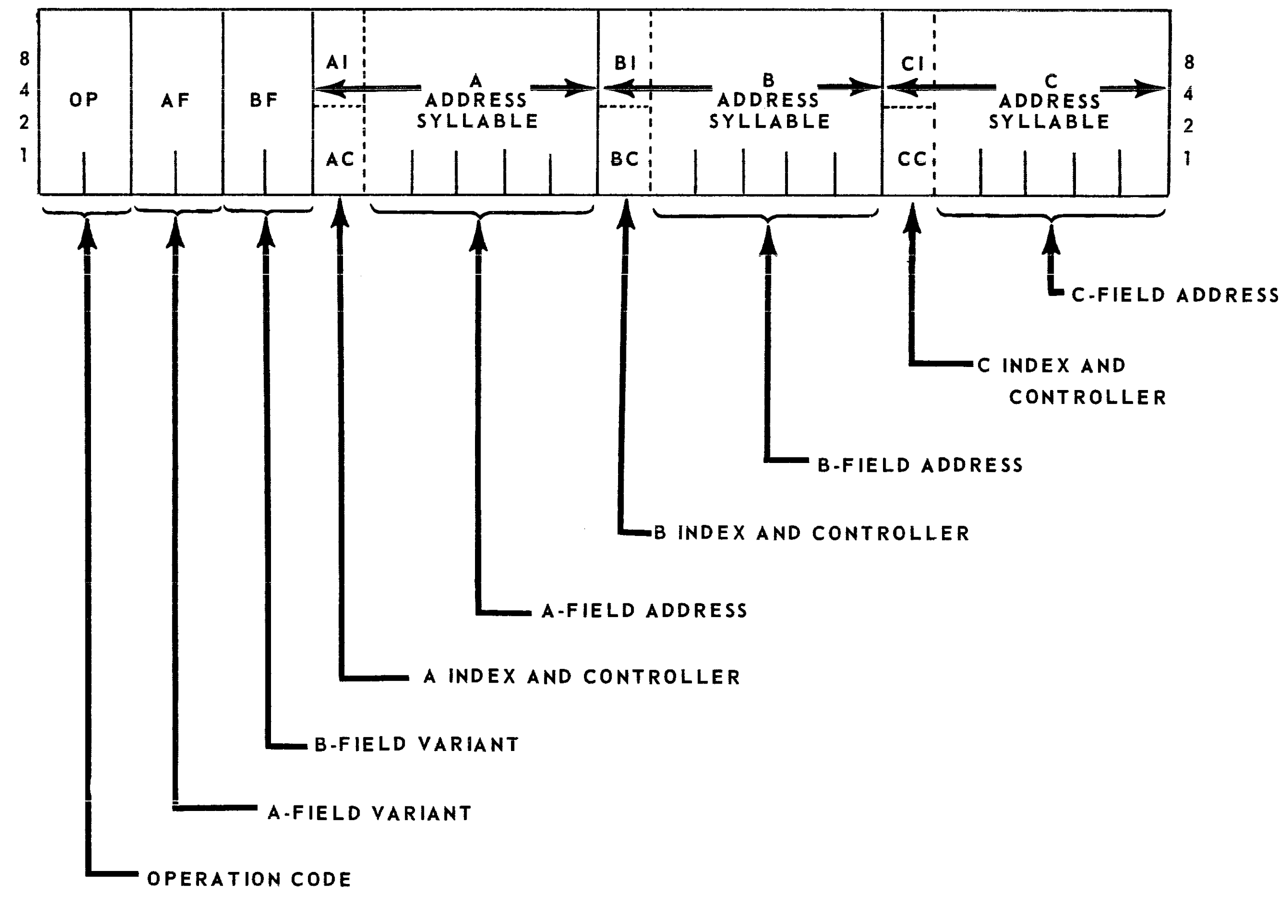
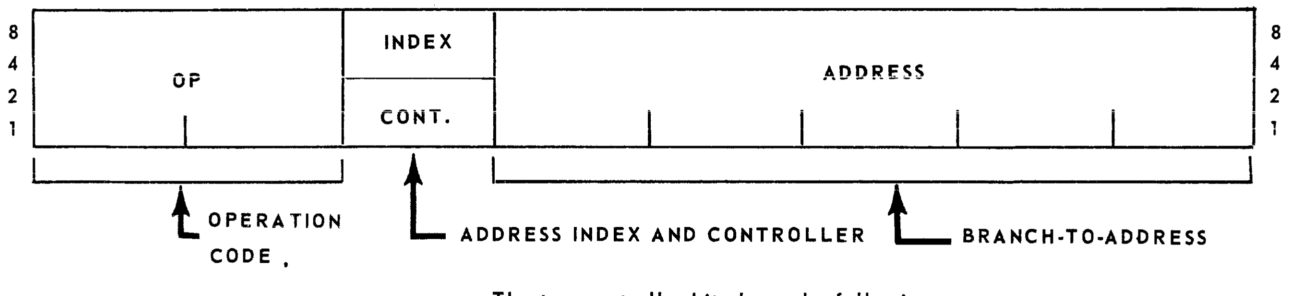


Figure 3–5. Format of a 4-Syllable Instruction

BRANCH INSTRUCTIONS

Branch instructions are considered as a special case within the processor of the B 2500 and B 3500 Systems and have a format different from that of the other processor instructions. The branch instruction contains only eight 4-bit digits. The first two digits are the operation code (OP) and the remaining six digits are a standard address syllable. Variants are not necessary in a branch instruction and an execute cycle is not required. If the condition demanding the branch is met. or if the branch is unconditional, the address to where the program is to branch is fetched. If the condition is not met, program control goes directly to the next instruction register (NIR) and the next instruction to be executed is fetched. This method of controlling branch instructions precludes the normal requirement of placing an exit address in address memory and in turn executing the instruction. Figure 3-6 shows the format of a branch instruction.



NOTE The two controller bits have the following meaning for Branch instructions:

$$\left.\begin{array}{l}00 = 0\\01 = 1\\10 = 2\end{array}\right\}\ \text{6th digit of address}$$

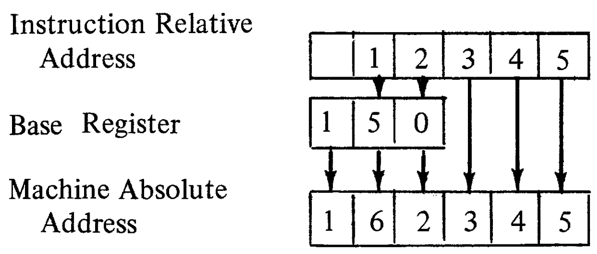
11 = indirect address

Figure 3–6. Format of a Branch Instruction

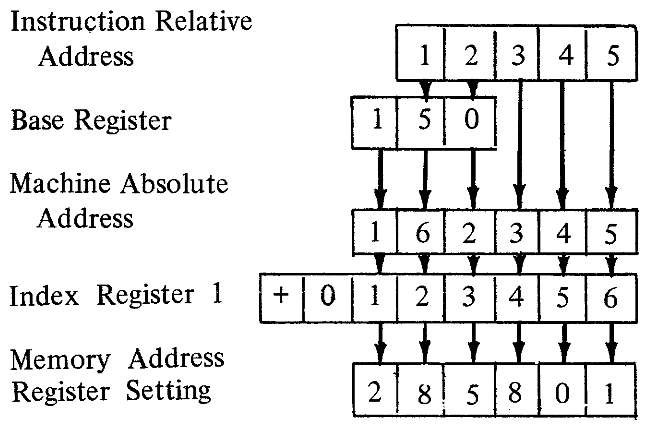
It should be noted that the address within a branch instruction is not pointing to data, but to another instruction to be executed. Since all instructions are represented in 4-bit digit format, the address controller cannot specify any other type of format; however, full generalities of indexing and indirect addressing are included.

ADDRESSING TECHNIQUE.

Addresses within all B 2500 and B 3500 Systems object programs are compiled and executed as being base relative to zero. This means that each program is created and executed with the assumption of zero being the object program's beginning point. When a program is assembled or compiled, it is assigned a five-digit relative address, starting at zero and continuing upward as far as necessary, depending on the size of the program. During execution of an instruction, each address contained in the instruction is automatically incremented by the three-digit base register, at no cost in execution time. The combination of the base register and the relative addresses creates a machine absolute location. This is a hardware capability and allows the MCP to assign a program to any contiguous area of core memory large enough to contain the program, and to subsequently relocate the program when necessary, by merely changing the base register. For example, assume the MCP loads a program starting at location 150000; the setting of the base register would be 150. The addresses within the program will be executed as illustrated at the top of the next column.



Assume further that the program has indicated that an instruction is to be indexed by the contents of index register 1. The final absolute location would be derived in the following manner:



To illustrate the relocation of programs, assume that a given system has 50,000 characters (bytes) of memory. Since each 4-bit digit may be addressed individually, the addresses would run from 00000 to 99999. If core memory contained the MCP requiring 13,000 bytes and three user programs, each requiring 10,000 bytes, the memory structure would be as follows:

Program	Starting Location	Base Register	Contiguous Digits Used
MCP	00000	000	25,999
Job-1	26000	026	45,999
Job-2	46000	046	65,999
Job-3	66000	066	85,999
Available core area	86000	086	99,999

Next assume that Job-2 finishes and Job-4 which requires 22,000 digits is introduced. When Job-2 is removed, the memory structure becomes:

Program	Starting Location	Base Register	Contiguous Digits Used
MCP	000000	000	25,999
Job 1	026000	026	45,999
Available core area	046000	046	65,999
Job 3	066000	066	85,999
Available core area	086000	086	99,999

At this point there are 33,000 digits available and Job-4 only requires 22,000, but Job-4 still cannot be initiated since the largest single block of core available is 20,000 digits. The very large operand capability of the B 2500 and B 3500 Systems allows the MCP to relocate Job-3 from location 066000 to 046000 with only one instruction. With minor reinitializations, e.g., changing the base register setting of Job-3 from 066 to 046, the MCP can now initiate Job-4 at location 066000 and processing of object programs may continue. During the next process cycle of Job-3, all of its base relative addresses will be incremented by its new base register setting, and the user will not be aware of the fact that the program was removed.

Another problem that must be solved before a system can effectively multiprogram is the accidental over-writing of one program by another, or the problem of memory protection. The B 2500 and B 3500 Systems contain, along with its base register, a limit register. When a program is assembled or compiled, the program is automatically provided with a program parameter record. This record contains, among other things, the total memory requirements of the program. As the MCP initiates a program, it will set the base register according to the memory location used, and the limit register to the maximum bounds of the program, i.e., the base register setting plus total memory requirements for the program. During the fetch cycle, after the machine absolute address has been assembled by applying the base register and index register, if applicable, the address is automatically checked against both the base and limit registers at no cost in execution time. If an address has been generated which is below the base register setting or above the limit register setting, the program is immediately interrupted, the contents of the memory location addressed are not changed and control is transferred to the MCP. The MCP will remove the program from processing and inform the operator of the address error and its location.

Input/output Instructions

The B 2500 and B 3500 Systems initiate an I/O instruction by sending control information to the peripheral control unit attached to the respective channel. The peripheral control unit then executes the operation independently from the processor, except that it time-shares address memory and the core memory sub-system with the processor and other peripheral control units. After having initiated an input or output instruction, the processor proceeds independently of the executed instruction and calls on the next instruction programed sequence. The peripheral control which received the descriptor will then execute the instruction on its own. Upon completion of the operation, it will send a result descriptor to core memory and set an interrupt to allow the processor to take the action necessary upon receiving an I/O complete condition from the control unit. Each peripheral control uses its own words of "scratch pad" address-memory to handle memory accesses when data is being passed to and from the peripheral. The peripheral control is able to count up the first address word and compares the result to the second address, which is the base address of the data field to be transferred which has been incremented by the physical size of that field. When they become equal, the peripheral control knows that the operation is finished, stops the transmission of information and initiates an I/O complete. The transfer of data may not be physically complete. For example, begin-and-end addresses within a card-read descriptor may be 40 characters apart, signifying that only the first 40 columns of the card are desired for a given program. The card reader would physically read all 80 columns, but the information from column 41 on would not be transferred to memory. When an I/O operation is completed, either because the specified memory bounds have been reached or because an end of record condition has been sensed, peripheral control sends a result descriptor to core memory, sets an interrupt within the processor, and readies itself to receive the next I/O descriptor.

SECTION 4

SYSTEM OPERATION

GENERAL

This section is intended to generally explain how the B 2500 and B 3500 function as systems. The modes (or states of operation), subroutine abilities, and types of instructions will be discussed.

OPERATIONAL STATES

There are two states of operation within the B 2500 and B 3500 Systems processor: the control state, when the MCP is being executed, and the normal state, when user programs are being executed.

Control State

There are several privileged instructions which cannot be executed in the normal state which allow the MCP to regulate the system; for example, initiate I/O operations, and control the program mix by setting and clearing registers.

All input/output operations are initiated while in the control state. During compilation or assembly, a privileged instruction to store the settings of the comparison and overflow indicators and the object program's return address will be automatically generated upon encountering a read or write request. The same instruction will also cause entry into control state. Base and limit registers are cleared and control is transferred to the proper point in the MCP.

Another method of entry into control state is the automatic interrupt system of the B 2500 and B 3500 Systems which is active during normal state and will be discussed later in this section. If an interrupt occurs during control state, the function is to set an indicator. The MCP will interrogate the interrupt indicators to determine if an interrupt has occurred while in control state before returning control to the object program currently being processed. If an interrupt has occurred, the MCP will first process it and then return to user program processing.

While the processor is in control state, any of the following conditions will cause the processor clock to be turned off with all processor conditions left static. That is, the system will halt and all registers and displays will be left unchanged.

a. Memory parity error.

b. Address error.

c. Instruction time out.

d. Invalid instruction.

Normal State

During normal state, the system executes arithmetic manipulations, performs data movement and editing, and does comparisons and control transfers as desired. In addition, the B 2500 and B 3500 Systems have two special instructions which permit efficient subroutine processing. Each of these areas will be covered and the manner in which the system executes them will be explained.

ARITHMETIC

The B 2500 and B 3500 Systems operation codes supply a complete set of arithmetic instructions capable of any type of calculation in either normal or floating-point operations. The B 2500 and B 3500 Systems allow two-address normal addition and subtraction; three-address normal addition, subtraction, multiplication, and division; and floating-point addition, subtraction, multiplication, and division. All floating-point instructions are three-address. All operand fields, both normal and floating-point, are variable in length from 1 to 100 digits or characters. Operands may be represented in unsigned numeric 4-bit format, signed numeric 4-bit format, or alphanumeric 8-bit format. Any combination of data representation is allowed in all normal arithmetic operations. Floating-point operands have a special format which is illustrated in figure 3-1. When unsigned numeric information and/or alphanumeric information is used, all values are considered positive. If only one operand has a sign, that sign is considered. If the result is to be without sign, the absolute value is stored.

After each arithmetic operation, the comparison indicator is set to either low for minus, equal for zero, or high for plus. If the arithmetic operation will cause overflow or underflow, the operation is terminated, all operands are unchanged, and the overflow indicator is turned on. This indicator may be tested by a Branch on Overflow instruction. If an overflow condition occurs and no test is made of the overflow indicator, subsequent arithmetic will not affect the indicator setting. Only the Branch on Overflow instruction will turn the indicator off. If a Branch on Overflow is executed when the overflow indicator is in the off condition, the branch will be treated as no operation. If overflow occurs in an arithmetic instruction, the comparison indicator is unaffected.

All arithmetic in the B 2500 and B 3500 Systems is accomplished from left to right (high-order to low-order).

DATA MOVEMENT.

The B 2500 and B 3500 Systems offer hardware flexibility in data movement. Numeric fields may be moved to alphanumeric fields creating alphanumeric representation; alphanumeric fields may be moved to numeric fields creating numeric representation; and source fields may be moved to destination fields of unequal length being either filled with zeros in numeric fields or blanks in alphanumeric fields from the left or right depending on the relation of field sizes and the type of instruction used. Data may be edited during movement operations by using floating insertion characters, fixed insertion characters, zero suppression, or any combination of the above. There are instructions available to allow data to be translated from any code to any other code during data movements through use of a translation table. The size of operands may range from one 4-bit digit to 10,000 16-bit words within one instruction.

DATA COMPARISON.

Hardware instructions within the B 2500 and B 3500 Systems allow two operands to be compared and the comparison indicator to be set to less-than, equal-to, or greater-than according to the relationship of the first (A) operand to the second (B) operand. The operands may be of different data representation and of unequal length if desired.

BRANCHING.

Branching may take place unconditionally or conditionally, depending upon the state of the comparison indicator or the overflow indicator. If conditional branching is executed on the basis of the overflow indicator, the indicator is reset following the operation. If branching is made on the basis of the comparison indicator, the indicator remains unchanged following the operation. The following types of branches are available:

a. No Operation (NOP).

b. Branch Unconditionally (BUN).

c. Branch on an overflow condition (OFL).

d. Branch if a comparison reflects a Greater-than condition (GTR).

e. Branch if a comparison reflects a Less-than condition (LSS).

f. Branch if a comparison reflects an Equal-to condition (EQL).

g. Branch if Less-than or Equal-to condition (LEQ).

h. Branch if a comparison reflects a Greater-than or Equal-to (GEQ).

i. Branch if a comparison reflects a Not Equal condition (NEQ).

j. Halt and Branch (HBR).

Any conditional branch is effectively a NOP if the condition of the branch is not met.

SUBROUTINE OPERATION.

It is often necessary to execute a given routine in several different places within a program. It is ideal in these cases to write the routine once, and each time it is needed to pass the necessary parameters to the routine, branch to and execute the routine, and then return to the main program in sequence.

While this is the ideal way to process subroutines, it has often developed that in conventional computers it was more trouble than it was worth as it required more instructions to pass the parameters and create return linkage than to duplicate the routine each time it was needed. Two instructions have been implemented within the B 2500 and B 3500 Systems which provide automatic recursive subroutine linkages which are called Enter and Exit. The Enter instruction will store the indicator settings and clear the indicators, store the location of the next instruction, pass parameters of from 1 to 9,999 characters, and branch to the subroutine. Upon completion of the subroutine, the Exit (EXT) instruction will restore the setting of the indicators (to the settings which were originally stored) and branch to the address which was stored by the Enter (NTR) command. This return address branch may be indexed, thus providing for variable return points if desired.

INTERRUPT SYSTEM.

The term "interrupt condition" means that a transfer of control takes place from the object program to the MCP so that it may initiate certain types of operations, or automatically handle errors which would cause time loss and operator intervention in conventional systems. The interrupt system also furnishes a means for continuous automatic recognition of exception conditions which otherwise would have to be checked programmatically at intervals. The conditions which cause storage of information and return points within a program and exit to the MCP are as follows:

a. Memory Parity Error. The instruction is terminated immediately. The address of the field containing the parity error is stored and a branch to the appropriate MCP routine is executed. The MCP will remove the affected program and notify the operator of the location of the parity error.

b. Address Error. All memory write cycles are inhibited. The MCP will remove the program, notify the operator of the program error and its location, and resume processing of other programs.

c. Instruction Time Out. An instruction timer is started with the fetch of an instruction and should time run out (250 ms.) without the completion of the instruction, an interrupt occurs and the MCP is invoked.

d. Invalid Instruction. The MCP will remove the program and notify the operator of the location of the invalid instruction.

e. Privileged Instruction. If a privileged instruction is encountered while the processor is in the normal state, it will be treated as an invalid instruction.

f. Receipt of Result Descriptor. When a peripheral control unit completes an I/O operation, an interrupt occurs to allow the MCP to handle the I/O Complete condition and check for any errors which may have occurred, or to start the next I/O operation on that peripheral control unit.

SECTION 5
MASTER CONTROL PROGRAM

MULTIPROGRAMING

One of the major advantages of comprehensive operating system control is the ability to simultaneously process several programs. The Master Control Program (MCP) supplied with B 2500 and B 3500 disk file system configurations directs a group of programs through the system according to scheduling priority, automatically allocating memory areas for each, assigning peripheral equipment to meet I/O requirements, and tending to all processing details (commonly referred to as "housekeeping"). Independently written programs may be entered into the system in any combination of B 2500 and B 3500 languages and in any sequence. The MCP continues to assign hardware components to meet the requirements of incoming programs until the entire system is in operation. As soon as a program is terminated, the MCP allows another program or group of programs to refill the system. In this way, the MCP keeps the system at peak utilization as long as there is work to be done.

A job may be entered into the system to meet an unexpected demand, given precedence, and, if desired, allowed to displace lower priority work currently being processed. As soon as the rush job is completed, regular work is resumed at the exact point where it was interrupted.

Some of the major MCP functions which have made multiprograming so successful are explained below.

Program Loading and Scheduling

Programs may be requested by an operator or by another program. When a program is requested, the MCP analyzes the job for memory requirements and priority, and schedules it for execution.

Jobs are queued by priority and executed as core becomes available. Any job may be given an immediate action priority causing the MCP to displace lower priority work until the rush job has been completed. When this job is completed, displaced programs will be automatically reinstated.

This job scheduling capability and flexibility taxes little of the system's power yet results in much higher productivity by assuring maximum utilization of systems resources in a multiprograming environment.

I/O Scheduling

The MCP maintains tables indicating the status of each I/O device as well as each I/O request being processed or awaiting an opportunity to be processed. Each time an I/O operation is initiated or terminated, the MCP compares the two lists and attempts to initiate all possible I/O operations. As a result, the system's I/O devices are kept fully active as long as there is work for them to do.

File Handling

On an MCP controlled B 2500 or B 3500 system, the operator puts tape files for scheduled programs on any convenient tape unit. When the unit is activated, the MCP reads the file label, associates it with the unit, and thus is able to identify it when the program is initiated. The MCP also automatically assigns scratch tapes as needed. If a needed file or scratch tape is not mounted when the program is started, the MCP will advise the operator and start the program as soon as the requirements are met. Files from random access disk storage are allocated space automatically by the MCP and accessed by symbolic relative addresses in the object program. When a program is terminated, the MCP automatically closes all files as part of its routine housekeeping function.

Files may consist of fixed or variable length records arranged in fixed or variable length blocks within core memory size limitations. The MCP automatically blocks and unblocks records according to specifications of the program.

Printer Back-Up

In a multiprograming environment it is quite possible for several programs to require printer or punch output simultaneously. The MCP handles this situation by diverting printer or punch output to a scratch tape or disk area while the printer or punch is busy and then reading the captured data back to the printer or punch when it is free. This allows processing to continue on programs which would otherwise have been suspended awaiting availability.

Pseudo Card Readers

To allow for maximum utilization of the card reader(s) on the system, the MCP provides the capability of creating pseudo card decks on disk. The cards are read at full card reader speed while the data is stored on disk. Subsequently, data is read from disk by an object program exactly as if the program were reading the data from a card reader. In this manner the MCP provides up to ten "card readers" for object program utilization with only one physical card reader present on the system.

Memory Allocation

At load time the MCP automatically reserves and loads all disk file and core memory space necessary for each program, its associated data, and its I/O buffers. The MCP takes advantage of base relative addressing to "float" programs and data in memory to optimize the use of available core space. When a program is terminated, the MCP immediately carries out all housekeeping necessary to make memory space available to the next program and may relocate programs currently in memory to prevent "checkerboarding" and further maximize space availability.

Automatic Overlay

Compiler and Assembler programs may be segmented to allow most of the program to remain in disk storage during execution until needed in core memory. The MCP automatically handles overlay routines, shuttling program segments from disk to core memory. This technique allows extremely large programs to be run on systems with modest core memory size, reduces core storage requirements for multiprograming, and lowers overall system cost without sacrificing performance.

Operator-System Communications

Most communications are from the MCP to the operator. The MCP informs the operator whenever it initiates or terminates a program. It advises him when a file is missing or a new scratch tape must be mounted and it warns of error conditions. The operator at any time may request a listing of all programs in the library or in process, and can start or suspend programs, change priorities, purge files, rewind tapes, and perform other operations through commands to the MCP.

Logging

Many difficult cost accounting problems are completely solved through use of the MCP-generated log. On operator command, the MCP interrogates its files and prints out a distribution of processor and peripheral time used by each program, plus a proration of system overhead time incurred. This log provides an excellent base for job costing and, as a complete and accurate picture of system utilization, greatly enhances management control over EDP operations.

Library Maintenance

The MCP allocates and maintains the system's library. It automatically assigns data and programs to the library, fetches them as required, and keeps track of their usage. The library may be maintained on magnetic tape or disk file or both and may be in source language, machine language, or both, thus affording complete flexibility at the user's option.

System Clock

The MCP maintains a 24-hour clock. It is used to supply time to object programs, to record timings in the log, and to trigger environment checking at regular intervals. Each time the MCP polls the equipment environment, it immediately takes action to make use of newly activated devices.

Major Priority Programs

Many computer users have several long jobs which must be run at regular intervals on a high priority basis. On conventional equipment these jobs tie up the entire system over a long period of time. Multiprograming gives the user access to his system during these long runs and allows him to process a number of other programs without disrupting the major run.

Data Communications

The multiprograming capabilities of the B 2500 and B 3500 lend great power to these systems for profitable data communications applications.

Rather than remaining idle while waiting for a message or a request from the data communications network, these systems process the installation's normal work. When a request is received, it is processed simultaneously with the current workload. Priorities are maintained. By eliminating wasted time between requests and spreading system cost among communications and normal processing, the B 2500 and B 3500 help break the economic barriers to many data communications and time sharing applications.

Expansion Without Reprograming

When the B 2500 or B 3500 configuration is expanded to include more memory or additional peripheral devices, a control card is entered to advise the MCP of the change. This simple action ordinarily represents the total reprograming required. The MCP automatically reorganizes the in-process program mix to take full advantage of the system's new capabilities. Changing from a B 2500 to a B 3500 is just as simple and usually requires no reprograming whatsoever.

Graceful Degradation

If certain types of equipment must be deactivated or removed from the system for maintenance, the MCP may be advised with a control card. The MCP will then dynamically reorganize processing and allow the system to continue to operate efficiently, although possibly with fewer programs in the mix.

Programing Simplicity

Many of the tedious, repetitious operations inherent in programing have been absorbed by the MCP. B 2500 and B 3500 programmers avoid nearly all detail work involved in memory allocation, loading routines, file opening and closing, record blocking and unblocking, I/O procedures, program overlays, library calls and other computer housekeeping. They can concentrate their talents on solving their problems rather than those of the machine.

Ease of Operation

The self-scheduling B 2500 or B 3500 system takes over many operating tasks, greatly easing the responsibilities of both the operator and the computer department manager. At set-up time, the operator may mount tape files on any convenient, unused, tape unit. The MCP will record the file identification and will assign it to the program at initiation time. The MCP also automatically assigns scratch tapes. The combination of MCP control and extensive disk file storage capabilities makes it possible to store most programs and many files in random access storage. This greatly reduces manual tape reel and card deck filing and retrieval operations.

Nearly all bookkeeping is done by the system, as is most scheduling. With the scheduling problems eliminated and with a detailed accounting of system use and utilization at his fingertips, the DP Manager can work with greater efficiency and ease on new applications and on further improvements of current operations.

SUMMARY

The Master Control Program is more than a computer program. It is a collection of programs designed around special hardware features to effect efficient and orderly use of the B 2500 and B 3500 Systems, associated facilities, and personnel. Thus, many of the necessary costly tasks of operating a computer system and running a computer installation are assumed by the Burroughs B 2500 and B 3500.

SECTION 6
PERIPHERAL COMPONENTS

GENERAL

This section discusses in general each of the peripheral units used in the B 2500 and B 3500 Systems.

SYSTEM CONSOLE

Communications between the operator and the system are made possible by the system console. The console contains NIXIE® tube and light displays which inform the operator of the contents of the various registers and the settings of the logical indicators. There is a set of operating keys used in conjunction with the numeric keyboard which allow the operator to read or alter internal memory or system control functions in any manner he desires. Either a standing or desk level console is available.

SUPERVISORY PRINTER

Next to, and considered a part of, the console is the supervisory printer (if a part of the system). This is an input or output device which allows direct communication between the system and the operator under program control. For systems using the Master Control Program, the supervisory printer is required and will be the only device used for normal system communications by the operator.

SYSTEM MEMORY

The system memory is a single magnetic disk device capable of storing up to two million characters of information. This information may be retained indefinitely without regeneration. The primary use of system memory is to house the software package and the user program library. In addition, it may be used for any type of working or general storage function.

DISK FILE

Bulk disk-file storage is available with the B 2500 and B 3500 Systems and features head-per-track construction.

® Registered Burroughs trademark.

MAGNETIC TAPE

Considerable flexibility in the area of magnetic tape handling devices is offered. The number of units available on a given system is limited only by the number of I/O channels used. Each I/O channel is capable of handling up to 10 magnetic tape units. The user is given the choice of using 7-channel tape or 9-channel tape. These may be intermixed if desired, but not on the same channel. The user may also select any of four packing densities desired up to 1600 bits per inch (BPI).

A choice of physical construction is also furnished. The user may choose a free-standing device which houses one magnetic tape unit within one cabinet, or the cluster device which provides four tape-handling devices in the same cabinet. A wide range of packing densities and transfer rates are available with either type of tape unit.

The magnetic tape units are capable of reading and spacing in either a forward or reverse direction.

CARD READERS

A full range of punched card readers with input speeds of from 200 to 1400 cards per minute are provided with the B 2500 and B 3500 Systems. These readers will accept 51, 60, 66 or 80 column cards. Cards punched in EBCDIC are read as a standard function. In addition, cards punched in BCL may be read as an option, but must be specified at the time of program execution. Automatic BCL code translation to EBCDIC is provided, if needed. The number of card readers on a given system is limited only by the number of input/output channels available. The readers feature photoelectric reading with character validity and read checking performed according to the character set of the code being read.

LINE PRINTERS

A wide selection of line printers is available. The user may desire a printer with speeds ranging from

315 to 1100 lines per minute. All printers may have either 120 or 132 print positions per line. All have vertical skipping and end-of-page formatting controlled by a punched paper tape and feature printing units which contain a 64-character set. Special purpose character sets are also available optionally. Again, the number and combinations of printers are limited only by I/O channels available.

CARD PUNCHES

The user may select card punches rated at either 150 or 300 cards per minute. The punches will create EBCDIC, BCL, or binary output under program control. The 300 card per minute model features programmatic stacker-select capabilities.

MICR READER SORTER

Reader sorters capable of reading and sorting documents encoded with magnetic ink at speeds up to 1565 items per minute are provided for use with the B 2500 and B 3500 Systems. When under program control, the reader sorter can operate in two modes: demand and flow. In demand mode, documents are fed one at a time as required by the program, at a maximum rate of 400 items per minute. In flow mode, documents are read and sorted at the free-flow rate of the reader sorter which is up to 1565 items per minute.

PAPER TAPE READERS

The paper tape readers used with the B 2500 and B 3500 Systems are capable of reading punched paper tape at speeds of 1000 characters per second. They are also capable of reading metalized Mylar or fanfold tape at a maximum rate of 500 characters per second. Baudot and BCL to EBCDIC code translation is automatic, but all other codes are read directly into memory and are programmatically translated. The readers accommodate 5, 6, 7, or 8-channel tape as selected by the system operator. Tape widths of 11/16, 7/8, or 1 inch are interchangeable. The number of paper tape readers is controlled by the number of I/O channels available.

PAPER TAPE PUNCH

One or more paper tape punches, depending on the I/O channels available, may be used as output devices for the B 2500 and B 3500 Systems. The paper tape punch is basically a teletype paper tape punch which is capable of punching standard paper tape format in BCL or Baudot code. The punch will accommodate 5, 6, 7, or 8-level tape at a minimum rate of 100 characters per second, punching ten characters per inch. Standard tape widths of 11/16, 7/8, and 1 inch may be punched. Either oiled paper tape, dry paper tape, metalized Mylar tape, or laminated Mylar tape may be used on the punch.

DATA COMMUNICATIONS EQUIPMENT

The following partial list of remote devices may communicate with the B 2500 and B 3500 Systems using either single-line or multiple-line controls:

a. B 300 and B 500 Systems.

b. Other B 2500 or B 3500 Systems.

c. B 9350 Typewriter Inquiry Station.

d. IBM 1030.

e. IBM 1050.

f. Burroughs Input and Display Unit.

g. TWX/Remote Typewriter.

h. Burroughs Audio Response System.

i. UNIVAC DCT 2000.

j. AT&T-85A1 Selective Calling Service.

k. TTY-28.

l. TC 500/TC 700

m. Burroughs On-Line Teller Consoles.

SECTION 7

SOFTWARE SYSTEMS

GENERAL

Today's EDP Manager often finds he can no longer meet increased demand on his department merely by installing more equipment or a faster system. This hardware approach often provides only a temporary solution to his workload problem. It fails to provide the responsiveness to unscheduled demands, applicational changes, and new applications required in most installations. Even excellent hardware is not enough to solve these problems, or to provide the high equipment utilization required for profitable EDP operation.

Software/hardware integration makes the difference. Burroughs B 2500 and B 3500 Systems are a blend of advanced electronic technology and unparalleled systems programing. They are designed to allow fast response to changes in workload or application. Multiprograming facilities (provided by the MCP) allow these machines to serve many programs simultaneously and to accept unscheduled jobs as easily as more routine processing.

In addition to this multiprograming capability, a complete software package is available which includes COBOL, FORTRAN, and the Assembler Programing System to meet his major programing requirements. Generated software is also available to meet day-to-day report, sort, and utility program needs.

COBOL

Burroughs B 2500/B 3500 COBOL offers all the advantages inherent in U.S.A.S.I. (United States of America Standards Institute) COBOL plus the unique benefits of B 2500 and B 3500 hardware design and MCP guided operation. COBOL consists of an English-like, business oriented language and a compiler program which translates this language into computer code. It has been implemented in installations across the country for nearly a decade and is the most widely used business oriented compiler language in existence.

COBOL'S popularity has been based on its distinct advantages to the business data processing installation. Its easy-to-read language allows management and supervisory personnel to see just what steps are taken in a program and to understand the process. Debugging and program modification are simplified and documentation is nearly automatic. Since a COBOL statement may represent an entire string of machine instructions, programs written in COBOL tend to be shorter and simpler; they require less writing time; and they are easier to test. In addition, COBOL programs written for one computer system may, with minor modifications, be run on another system and similarly, programmers are easily retrained to write COBOL programs for different computer systems. All of these advantages are inherent in COBOL usage.

B 2500/B 3500 COBOL meets or exceeds all the standards proposed by the United States of America Standards Institute X3,4.4 task group in May, 1965. This recently standardized version of COBOL has been carefully integrated with B 2500/B 3500 hardware and MCP design to ensure optimum operation on these systems.

COBOL CROSS-REFERENCING AND FLOW CHART SYSTEM

COBOL programs are normally very lengthy due to the inherent nature of the language itself. They normally contain data elements by the hundreds, plus many procedure-names, thus causing time consuming, manual effort when a programmer is required to make a simple change in a program. A single programmatic change in the source language may affect many areas of a program with the possibility that an affected area will be far removed from that which was immediately changed and in turn can be easily missed by the most experienced programmer. The Cross-Referencing and Flow Chart System allows a programmer to find all areas that a change will affect, at a glance.

FORTRAN

FORTRAN is offered as the major computational programing system for the B 2500 and B 3500. It is the most widely used compiler system in existence. FORTRAN is similar in concept to COBOL, but is tailored to scientific and engineering applications. Burroughs B 2500/B 3500 FORTRAN is adopted from U.S.A.S.I. standards set by the X3.9 task group and accepted March 7, 1966.

FORTRAN offers two main advantages to the scientific/engineering computer installation: improved communications and greater programing speed. Its language is expressed much like mathematical formulae and is easily learned by most scientists and engineers. Once learned, FORTRAN allows these people to communicate directly with the computer system for many of their processing requirements and to work closely with the programing staff on more complicated applications. The second major FORTRAN advantage, speed, results from its simplified condensed nature. Each FORTRAN statement represents a string of machine instructions.

The programmer does not have to concern himself with these machine instructions but instead uses the FORTRAN language as a kind of shorthand. The result is shorter, more straight forward programing.

Statements written in the FORTRAN language are translated into machine language by the FORTRAN compiler. On the B 2500 and B 3500 Systems, this translation, or compilation, is done under the guidance of the MCP and may be multiprogramed. The resulting program is usually about equal in processing efficiency to one devised in machine language by a competent programmer. The time and cost involved in producing a FORTRAN program, however, is far less than that of machine coding.

The standardization inherent in FORTRAN programing has several major advantages. A program does not have to be returned to its author to be revised. Any programmer with FORTRAN training can understand another programmer's FORTRAN program and expand it, revise it, or adapt it to another computer system. This flexibility can drastically cut costs during normal operating conditions and offers even greater savings when a new system is installed and software conversions must be made. Little or no reprograming is required when changing B 2500 or B 3500 configurations or moving to another computer system.

ASSEMBLER PROGRAMING SYSTEM

Assembler Programing Systems offer B 2500 and B 3500 users the flexibility inherent in symbolic languages plus the advantages of many compiler-like macro-instructions to further ease the programing task. The languages include facilities for symbolic addressing, diagnostics, and operation with the control programs.

While coding in the symbolic Assembler Language, the programmer has complete flexibility of instruction modification, indexing, character or bit manipulation, and program segmentation.

Programs coded in the Assembler Language are translated into machine language by the Assembler Program. The time required for this assembly process is minimal, and on MCP controlled systems, assembly operations may be multiprogramed. The resulting object programs operate in concert with the control programs to take full advantage of the hardware system capabilities and operate at optimum speed.

REPORT PROGRAM GENERATOR

The Report Program Generator available with the B 2500 and B 3500 Systems allows the user to produce many programs of the simpler variety in minimum time and at optimum efficiency. The Report Program Generator is a generative program which compiles relatively complex symbolic programs from a brief, simplified, problem-oriented language. Program generation is fast and the resulting programs will normally run at the rated speeds of the designated peripheral equipment. The Report Program Generator is designed for use with the Master Control Program. The generative program and the resulting object program may be multiprogramed on an MCP controlled system.

SORT PROGRAM GENERATOR

Sorting procedures, which take 30 to 50 percent of total run time in many installations, are simplified and made more efficient by the generative programs available with the B 2500 and B 3500.

The Sort Program Generator is designed to reduce programing time and to quickly generate programs tailored to the user's requirements.

An Advanced Sort Program Generator is available in both magnetic tape and disk file versions and operates with the Master Control Program (MCP).

MEDIA CONVERSION PROGRAM GENERATOR

The Burroughs B 2500 and B 3500 Information

Processing Systems user has the advantage of generative data conversion software. Generation eliminates the cost of individually programing conversion routines - yet provides better flexibility and greater efficiency than is possible with standard routines. The Media Conversion Program Generator, with a single specification card input, provides complete media conversion routines tailored to individual requirements. Both the generative programs and the generated media conversion routines take full advantage of the computer system's speed and capabilities.

B 100/B 200/B 300/B 500 SIMULATOR

The B 2500/B 3500 Simulator of B 100/B 200/B 300/B 500 Systems was designed to provide the means to execute object programs written for B 100/ B 200/B 300/B 500 Systems on the B 2500/B 3500 Systems. The Simulator will create a B 100/B 200/B 300/B 500 environment within B 2500/B 3500 core to perform the originally intended functions, and can be multiprogramed with itself or with any other advanced systems software package under MCP control.

The intent of this software package is to provide every B 100/B 200/B 300/B 500 user with the ability to take an auto-load deck, data, and pertinent operating instructions to a B 2500/B 3500 System and execute the program as if the system was a B 100/B 200/B 300/B 500.

MCP SORT INTRINSIC

The MCP Sort Intrinsic allows all advanced system languages available for the B 2500/B 3500 to contain a disk sort capability, but does not demand the user to provide disk storage space (library) for internal program sort coding. Users will only find a simple Branch Communicate linkage to the MCP at the sorting point in his program which will cause the MCP to generate a sort program (in approximately three seconds), roll-out and save the object program, do the sort in the rolled-out-of-space, deliver a sorted output file, roll-in the suspended program to its original place in memory, and proceed with the program. Error procedures, as well as special file handling before and after the sort is executed, are provided. Programs can contain an infinite number of sorts without affecting memory or disk storage, or the program's physical operation, to any degree.

BURROUGHS CORPORATION
DATA PROCESSING PUBLICATIONS
REMARKS FORM

TITLE: ______________________
____
____

FORM: ______________
DATE: ______________

CHECK TYPE OF SUGGESTION:

☐ ADDITION ☐ DELETION ☐ REVISION ☐ ERROR

GENERAL COMMENTS AND/OR SUGGESTIONS FOR IMPROVEMENT OF PUBLICATION:

FROM: NAME ______________________
TITLE ______________________
COMPANY ______________________
ADDRESS ______________________
____

DATE ______________

tear along dotted line

STAPLE

FOLD DOWN SECOND FOLD DOWN

Postage Will Be Paid by Addressee

No Postage Stamp Necessary If Mailed in the United States

BUSINESS REPLY MAIL
First Class Permit No. 817, Detroit, Mich. 48232

Burroughs Corporation
6071 Second Avenue
Detroit, Michigan 48232

attn: Sales Technical Services
Systems Documentation

FOLD UP FIRST FOLD UP