BUSINESS PACKAGE II
(BUS-PAK II)

REFERENCE MANUAL

PRELIMINARY

DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS

BUSINESS PACKAGE II

(BUS-PAK II)

REFERENCE MANUAL

G B Colicelli
Digital Equipment Corporation
January 31, 1965

CONTENTS

CONTENTS (Cont'd)

CONTENTS (Cont'd)

SECTION I

SYSTEM DESCRIPTION

Bus-Pak II, in essence, is a new computer designed for data processing operations. It operates on a character by character basis and its instructions are powerful and easy to learn and understand. Bus-Pak II offers a variety of powerful programming features such as Editing, Two Modes of Indexing and Complete Input/Output Control. The Bus-Pak II programming system was developed so that many of the manual record keeping and updating operations could easily be converted to make use of the PDP-4 or PDP-7 computing system. Bus-Pak II users need not be aware of all the computer intricacies. Through the use of the pseudo-language, one can accomplish most of the functions of a business oriented computer including the handling of the peripheral in-out equipment.

This manual has been written so that programmers with minimum experience will be able to learn and understand the Bus-Pak II Programming Language.

Reference may also be made to either the PDP-4 or PDP-7 Assembler writeups but is not a necessity.

Subroutines and other programs written in machine Assembler language may be used within a Bus-Pak II program provided that the programmer saves the accumulator before execution of a Bus-Pak II instruction, and restores it when re-entering the machine language program where necessary.

SYSTEM CONFIGURATION REQUIREMENTS

The Bus-Pak II Programming System will operate on either a PDP-4 or PDP-7 with the following configurations:

Standard Equipment

8K core storage

Paper tape reader-punch

Teletype input-output

And at least 1 input and 1 output unit shown below

Optional Equipment

Card reader

Card punch

Magnetic tape

DECtape

High speed printer

AVAILABLE STORAGE

The Bus-Pak II operating system occupies all of upper memory (10000_8 to 17777_8) and locations 0_8 to 21_8 of lower memory. The balance of lower memory is available to the user minus the sum of all the buffer areas assigned to Magnetic Tape and Micro-Disk units being used.

7777_8 - total sum of all buffer areas -1 = last available storage location.

The following formula may be used to determine the additional non-available storage areas.

Magnetic Tape (for each unit used)

Maximum length tape record in characters ÷ 3 = non-available storage.

Micro-Disk (for each unit used)(1)

((768 - record length) x record length) ÷ 3 = non-available storage

(1) Do not use any remainders in calculations

MODES OF OPERATION

Bus-Pak II has two (2) modes of operation. A "Run" mode which is used for normal execution of the users program and a "Single Instruction" mode for use in debugging Bus-Pak II programs. The control of the mode of operation is by the AC switch Zero (Ø) on the console. When in the down position, Bus-Pak II operates in the "Run" mode. When in the up position, Bus-Pak II operates in the "single instruction" mode.

In the single instruction mode of operation, Bus-Pak II halts after the execution of each Bus-Pak II instruction and indicates in the AC lights on the console, the address of the next Bus-Pak II instruction to be executed. When a "GOTO" (transfer) instruction is executed, Bus-Pak II will not stop until the instruction at the location indicated by the GOTO instruction is executed.

ADDRESSING

Both instructions and data essential for processing are contained in core storage. Each core storage location is completely addressable.

Bus-Pak II instructions are variable length type instructions in that not all the instructions take up the same number of core storage locations.

Data fields being processed are also of the variable length type. A data field length is determined by the "N" (number of characters) field in a specific instruction.

All data is processed from left to right, for as many characters specified by the instruction being executed.

Both instructions and data may be intermixed as long as the data does not interfere with the normal flow of the program.

INPUT-OUTPUT STORAGE ASSIGNMENTS

No specific input-output areas have been assigned to any input-output device in the Bus-Pak II system. The assignment of these areas has been left entirely up to the programmer. In this way, more efficient and less core consuming programs may be written. Care, though, must be taken so that an area defined for a specific input-output device is large enough for that particular device.

EDITING

In the printing of reports, it is sometimes necessary to punctuate numeric data by the addition of dollar signs, commas, and decimal points. This punctuation would take many, many instructions of testing and shifting the data, and inserting the correct punctuation characters. The editing feature provides this punctuation of data automatically based on a control word specified by the user.

Floating dollar sign and asterisk protection is also available for check writing. Multiple sequential data fields may be edited in one editing operation.

INDEXING

Indexing is a means of address modification without disturbing the

original data address in an instruction. Bus-Pak II makes available two modes of indexing, single indexing and double indexing.

An effective address is calculated for every "TO", "FROM", and "BY" address field specified by an instruction.

In single indexing the contents of the index register specified by an address field is added to the data address and this new effective address is used in the execution of the instruction.

In double indexing, the contents of the index register specified by the double index register is also added to the data address and this new address used in the execution of the instruction.

INDIRECT ADDRESSING

When indirect addressing is specified, the address of the instruction is interpreted as the address of a register which contains the actual address of the data to be processed. Multiple levels of indirect addressing are available and each level of a "TO" or "FROM" address field may use single and/or double indexing.

DOUBLE PRECISION ARITHMETIC

All arithmetic operations on numeric data must be done by the use of one of the fifteen (15) double precision accumulators available in Bus-Pak II. Each accumulator is capable of containing a magnitude not exceeding

$$\pm 3\ 4\ 3\ 5\ 9\ 7\ 3\ 8\ 3\ 6\ 7.$$

An overflow indicator (set when any calculated value exceeds that limit) is associated with each of the 15 available accumulators. The signs of the accumulators are computed algebraically depending on the signs of the data being calculated.

PROGRAM COUNTERS

Fifteen (15) programmed counters are available for control of multiple execution of a particular sequence of instructions.

SENSE SWITCHES

Fifteen (15) sense switches are available through the use of the AC switches on the console for manual control of program execution.

PROGRAM SWITCHES

Fifteen (15) programmed switches are available for internal control of program execution.

SECTION II

INSTRUCTION DESCRIPTION

For ease in learning and understanding all the instructions, their descriptions will follow the format indicated below:

INSTRUCTION TITLE:	Indicates the operation of the instruction.
INSTRUCTION FORMAT:	Indicates the exact format in which the instruction is to be written.
op-code	The actual machine code assembled for the instruction (for reference only).
mnemonic	The mnemonic code for the instruction.
operands	Indicates the variable operands necessary for the instruction in the sequence in which they must appear.
INSTRUCTION DESCRIPTION:	Describes, in detail, the operation of the instruction.
INSTRUCTION NOTE:	Indicates restrictions, exceptions, and requirements for that instruction.
INSTRUCTION EXAMPLE:	Indicates the instruction's effect on the data fields showing what the applicable fields look like both before and after the instruction is executed.

INSTRUCTION FORMAT

Bus-Pak II instructions are divided into four distinct fields of information. Each field must be separated from the other by either a carriage return (↵) or a tabulation (→). The Bus-Pak II coding sheet has been developed for ease of coding and punching of Bus-Pak II programs. (See Appendix V)

FIELD	USE
LABEL	Used to assign a symbolic name to an instruction or data for reference by instructions located elsewhere in the program. A label may vary from one to six alphanumeric characters in length, beginning with an alphabetic character. Only the normal alphabetic characters and the numbers 0-9 may be used in a label. A comma (,) must immediately follow the label.
OPERATION	Indicates the operation to be performed by Bus-Pak II. An instruction mnemonic will normally appear in this field.
OPERANDS	This field is sub-divided into up to four sub-fields. These fields are used to give Bus-Pak II the information needed to execute the instruction. Each sub-field must be separated by a tabulation (→). All four fields need not be used on all instructions. See "Detailed Operand Description".

COMMENT	This field may be used to place textual information about the instruction on the listing of the program for easier understanding of the program coding. A slash (/) must immediately precede this field. A comment may contain any character and is only terminated by a carriage return (↵).

DETAILED OPERAND DESCRIPTION

Following is a detailed description of the different types of information that may be contained in the operand field of an instruction. A name has been assigned to each type of information for ease in recognition. These names will be used in the description of the instructions, and should be referred to wherever necessary.

NAME	MEANING AND USE
AC	Indicates the number of the Accumulator being used.
ALT	Indicates the alternate tape unit specified for use in switching tape units on End-of-Tape conditions.
B	Indicates the blocking factor for magnetic tape initialization.
BY	Indicates the address of the control word to be used in an "Editing Operation". It should be replaced by the label assigned to the control word. This field may be indexed, double indexed, and indirectly addressed. (See indexing feature).
CHAR R?	Used in the character testing and forms control instructions. Must be written exactly as is with the

	question mark (?) replaced by the character desired.
CTR	Indicates the program counter being used. Should be replaced by the actual program counter number or label.
FROM	Indicates the address of the data field being worked with. Should be replaced by the label assigned to the data field. This field may be indexed, double indexed, and indirectly addressed (see indexing feature).
IDX	Indicates the specific index register being used. It should be replaced by the actual index number or label.
INST	Indicates the address or label of an instruction in a "GOTO" instruction. This address may be indirectly addressed, but not indexed. It is separated from the "GOTO" command by a space.
L	Indicates the length of a given record for an input-output device initialization.
N	Indicates the number of consecutive character locations that a particular instruction is to process.
SW	Indicates either the sense switch or program switch being used. Should be replaced by the actual switch number or label.
T	Indicates the number of times a particular instruction is to be executed. Can never be negative.

TO	Indicates the address of the data field receiving information. Should be replaced by the label assigned to the data field. This field may be indexed, double indexed, and indirectly addressed. (See indexing feature).
U	Indicates the particular input-output unit being used. Should be replaced by the actual unit number or label.
V	Indicates a value for use with the indexing and program counter instructions. Should be replaced by an actual value.

INDEXING

SINGLE INDEXING

A program sometimes requires that a particular sequence of instructions be executed repetitively with a change only to the data addresses specified by the instructions. The act of changing the data addresses prior to the execution of the instruction is referred to as address modification.

The indexing feature of Bus-Pak II performs this address modification automatically. Making use of this indexing feature reduces the core storage requirements of a program and provides for faster program execution and simplification of the programming effort.

Bus-Pak II makes available fifteen (15) index registers numbered 1 through 15. These index registers may be cleared, initially loaded with a given value, incremented or decremented by a given value, or deposited for the purpose of saving the contents of the index register.

To make use of the indexing feature, the programmer selects those instructions which use indexing. He then indicates the indexing operation by placing the symbol (+Xn), where n is the index register to be used, in the address field to be indexed:

ADDRESS + Xn

Before an instruction is executed, the data addresses specified by that instruction are examined for indexing. If indexing is required, the contents of the proper index register are added to the data address to develop an effective address. This effective address is the actual address used in the execution of the instruction.

DOUBLE INDEXING

Indexing is the means of modifying a specific data address by adding the contents of a specified index register. Double indexing therefore is the means by which the contents of a second index register may also be added to the data address.

The double index register is loaded with the index register number whose contents are to be added to the data address when double indexing is specified.

Double indexing is specified by the symbol (+D) in the data address as follows:

```
TO+D+X1

TO+D
```

INDIRECT ADDRESSING

When indirect addressing is specified, the register indicated by the address in the instruction contains the address to be used as the final data address.

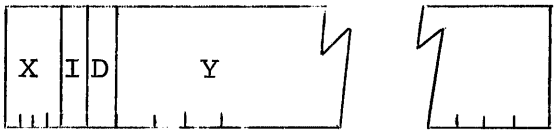
Indirect addressing is specified by placing the symbol +I following the address of the data address to be used.

```
FROM+I

TO+X1+I

BY+X1Ø+D+I
```

EFFECTIVE ADDRESS CALCULATION

All "FROM", "TO", and "BY" address fields of instructions calculate an effective address in the following sequence.

FORMAT OF ADDRESS FIELD



1. Obtain the number in the address part, Y, bits 6-17.

2. If the index field, X, bits 0-3 is non-zero, add the contents of the specified index register to the number obtained in step 1.

3. Obtain the double indexing bit, D, bit 5. If it is a zero, go to step 5. If it is a 1, go to step 4.

4. Add the contents of the index register specified by the double index register to the result of steps 1 and 2.

5. Obtain the indirect bit, I, bit 4. If it is a zero, the calculation is done and the result of steps 1, 2, 3, and 4 is the effective address. If it is a 1, go to step 6.

6. Use the address calculated by steps 1, 2, 3, and 4 to obtain a new word from memory, and go back to step 1.

The effective address calculation continues until a word is encountered with a Ø in bit 4. At this point the result of steps 1, 2, 3, and 4 is taken as the effective address for that instruction.

SECTION III

INSTRUCTION SET

The Bus-Pak II instruction set has been divided into specific types of instructions. They will be described in the following sequence:

1. Basic Control Instructions
2. Data Manipulation Instructions
3. Logical Control Instructions
4. Arithmetic Instructions
5. Accumulator Shift Instructions
6. Editing Instructions
7. Alphanumeric and Numeric Comparison Instructions
8. Indexing Control Instructions
9. Input-Output Instructions
    a. Teleprinter
    b. Punched Cards
    c. High Speed Printer
    d. Magnetic Tape
    e. Micro-Disk
    f. Storage and Retrieval Instructions
    g. Checkpoint and Restart
10. Subroutine Control Instructions

BASIC CONTROL INSTRUCTIONS

The following instructions are used to control the Bus-Pak II operating system:

DECIMAL RADIX CONTROL

op-code	mnemonic	variable operands
	DECIMAL	

This is an instruction to the assembler to accept all numeric data as decimal numbers.

NOTE:
1. This instruction must appear after the program "Title" and before any Bus-Pak II instructions.
2. This instruction does not occupy any core storage locations.

ANELEX CHARACTER MODE

op-code	mnemonic	variable operands
	ANELEX	

This is an instruction to the assembler to accept all textual or character information of the program in "ANELEX" code.

NOTE:

1. This instruction must appear after the program "Title" and before any Bus-Pak II instruction.

2. This instruction does not occupy any core storage location.

3. All textual or character information used either in the Bus-Pak II instructions themselves or as constants necessary in the processing of the data must be in "ANELEX" code.

4. Textual information (see "TEXT" pseudo instruction) will be stored 3 characters per word. When expanded to one character per word Bus-Pak II assumes the data is in "ANELEX" code.

TELETYPE CHARACTER MODE

op-code	mnemonic	variable operands
	TELETYPE	

This is an instruction to the assembler to accept all textual or character information of the program in "TELETYPE" code.

NOTE:

1. This instruction is to be used prior to the defining of textual information which will only be typed out on the on-line teleprinter.

2. Textual information used for message printing is usually placed at the end of the program. If it is inserted within the program itself, "ANELEX" must again be specified to insure correct character conversion for the remaining part of the program.

INITIALIZE SYSTEM

op-code	mnemonic	variable operands
1174Ø1	BEGIN	

This instruction causes the Bus-Pak II operating system to be initialized.

NOTE:

1. This instruction must be the first instruction executed by the users program.

2. Initialized the on-line teleprinter by setting it to lower case and typing a line feed and carriage return.

3. Clears all the interrupt flags in the computer and turns the interrupt to the "ON" state.

4. Resets certain controls in the Bus-Pak II operating system.

5. When the Bus-Pak II system is operating in the single instruction mode, the system will not stop until the instruction immediately following this instruction has been executed.

UNCONDITIONAL TRANSFER

op-code	mnemonic	variable operands
6ØØØØØ	GOTO INST	

This instruction causes a transfer of the program control from one sequence of instructions to another.

NOTE:

1. The "GOTO" and the instruction address "INST" must only be separated by a space.
2. Indirect Addressing may be used with this instruction. (See Indirect Addressing).
3. Indexing and Double Indexing may not be used with this instruction.
4. When the Bus-Pak II system is operating in the Single Instruction Mode, the system will not stop until the instruction at location "INST" has been executed.

NO OPERATION

op-code	mnemonic	variable operands
74ØØØØ	NOP	

This instruction performs no operation.

NOTE:

1. This instruction can be substituted for the operation code and operand fields of instructions to make the instruction ineffective.
2. When the Bus-Pak II system is operating in the Single Instruction Mode, the system will not stop until the next Bus-Pak II instruction immediately following this instruction has been executed.

STOP

op-code	mnemonic	variable operands
175ØØ	STOP	

This instruction will cause the Bus-Pak II system to stop operations.

NOTE:

1. The system will first wait until all input-output operations have been completed.

2. The address of the next instruction following the STOP command is displayed on the console AC lights.

3. When the Bus-Pak II system is operating in the Single Instruction Mode, the system will again stop after the CONTINUE key on the console is depressed before executing the next instruction.

DATA MANIPULATION INSTRUCTIONS

The following instructions are used to manipulate and position input and calculated data for eventual output.

CLEAR STORAGE

op-code	mnemonic	variable operands
17423	CLRSTR	N FROM

The "N" consecutive locations starting at address "FROM" are cleared to blanks (2Ø octal).

NOTE: 1. The original "N" consecutive locations at "FROM" are lost.

EXAMPLE:

CLRSTR 3 5Ø1

CONTENTS OF CORE STORAGE	before	1	2	5	7	3	Ø
	after	1	b	b	b	3	Ø
CORE STORAGE ADDRESSES		5 Ø Ø	5 Ø 1				5 Ø 5

MOVE CHARACTERS

op-code	mnemonic	variable operands
17411	MV	N FROM TO

The "N" consecutive characters starting at address "FROM" are moved from left to right to the "N" consecutive character positions starting at address "TO".

NOTE: 1. The original "N" consecutive characters at "TO" are replaced by the "N" consecutive characters at "FROM". The "N" consecutive characters at "FROM" are left undisturbed.

EXAMPLE:

MV 3 47Ø 473

CONTENTS OF CORE STORAGE		A	B	C	D	E	F
	before	A	B	C	A	B	C
	after						
CORE STORAGE ADDRESSES		4			4		
		7			7		
		Ø			3		

MOVE ZONE

op-code	mnemonic	variable operands
17412	MVZ	N FROM TO

The zone portion only (bits B and A of a character) of the "N" consecutive characters with the starting address "FROM" are moved from left to right to the zone portion only of the "N" consecutive characters with the starting address "TO".

NOTE: 1. The original zone portions only of the "N" consecutive characters at "TO" are replaced by the zone portions only of "N" consecutive characters at "FROM". The "N" consecutive characters at "FROM" and the numeric portion only of the "N" consecutive characters at "TO" are left undisturbed.

EXAMPLE:

MVZ 1 5Ø2 5Ø5

CONTENTS OF	before	5	7	+6	3	9	-1
CORE STORAGE	after	5	7	+6	3	9	+1
CORE STORAGE ADDRESSES				5 Ø 2			5 Ø 5

MOVE NUMERIC

op-code	mnemonic	variable operands
17413	MVN	N FROM TO

The numeric portion only of the "N" consecutive characters with the starting address "FROM" are moved to the numeric portion only of the "N" consecutive characters with the starting address "TO".

NOTE: 1. The original numeric portion only of the "N" consecutive characters at "TO" are replaced with the numeric portion only of the "N" consecutive characters at "FROM". The "N" consecutive characters at "FROM" and the zone portion only of the "N" consecutive characters at "TO" are left undisturbed.

EXAMPLE:

MVN 3 3ØØ 3Ø3

CONTENTS OF CORE STORAGE	before	5	7	+3	5	2	-1
	after	5	7	+3	5	7	-3
CORE STORAGE ADDRESSES		3			3		
		Ø			Ø		
		Ø			3		

MOVE AND SUPPRESS ZEROS

op-code	mnemonic	variable operands
17414	MVS	N FROM TO

The "N" consecutive characters with the starting address "FROM" are moved to the "N" consecutive character positions with the starting address "TO". As each character is moved, it is tested for zero. If it is a zero and it is to the left of the most significant digit (other than zero), the zero is replaced by a blank (2Ø octal). When the most significant digit is found, zero suppressing is terminated.

NOTE:

1. The original "N" consecutive characters at "TO" are replaced by the "N" consecutive characters at "FROM".
2. All leading zeros up to the most significant digit are replaced by blanks.
3. All zones of the receiving field are lost.
4. The "N" consecutive characters at "FROM" are left undisturbed.
5. If a field contains all zeros, the resulting field will be entirely blank.

EXAMPLE:

MVS 3 5ØØ 5Ø3

CONTENTS OF	before	Ø	Ø	+5	A	B	C
CORE STORAGE	after	Ø	Ø	+5	b	b	5
CORE STORAGE ADDRESSES		5Ø Ø			5Ø 3		

MOVE AND EXPAND

op-code	mnemonic	variable operands
17415	MVX	N FROM TO

The "N" consecutive characters (packed three characters per storage word) with the starting address "FROM" are moved to the "N" consecutive character positions (one character per storage word) with the starting address "TO".

NOTE:

1. The original "N" consecutive characters at "TO" are replaced by the "N" consecutive characters with the starting address "FROM". The "N" consecutive characters (packed three characters per storage word) at "FROM" are left undisturbed.

2. The "N" consecutive characters at "FROM" must be in "ANELEX" code and packed three characters per storage word.

3. The pseudo-instruction "TEXT" under "ANELEX" code control is used to input the "N" consecutive characters, packed three characters per storage word at "FROM".

4. This instruction is normally used to expand "CONSTANT" information used as input to the assembly process so that it can be used by the Bus-Pak II data manipulation instructions.

CHARACTER ZONE MANIPULATION

op-code	mnemonic	variable operands
		TO

These instructions operate on the zone portion only of the character at location "TO". Following are the instruction mnemonics and the effect each has on the character zone.

op-code	mnemonic	effect
1742Ø	SETX	sets zone to 6Ø octal (plus)
17421	SETY	sets zone to 4Ø octal (minus)
17422	CLZ	clears the zone to ØØ octal

NOTE: 1. The numeric portion of the character is left undisturbed.

LOGICAL CONTROL INSTRUCTIONS

The following instructions may be used to control the logical flow of the program. Both internal and external control are available.

Internal control may be exercised by the use of program switches, program counters, or the actual data being processed.

External control can be exercised by the use of sense switches on the computer console.

TEST CHARACTER EQUAL

op-code	mnemonic	variable operands
17456	TCE	CHAR R? TO GOTO INST

The single character at address "TO" is compared to the character "CHAR R?"

If the single character at address "TO" is identical to the character desired the instruction "GOTO INST" will be executed. Otherwise, the program continues in sequence.

NOTE:

1. Core storage is left undisturbed.
2. The character at address "TO" must be in BCD mode, one character per storage location.
3. The character being tested for by "CHAR R?" must be under the ANELEX mode when being assembled.
4. Bus-Pak II will automatically convert the character desired from ANELEX to BCD code before the actual test is made.
5. Indexing, double indexing, and/or indirect addressing may be used with the "TO" address.

SEARCH FOR CHARACTER

op-code	mnemonic	variable operands
175Ø5	SEARCH	N CHAR R? FROM GOTO INST

This instruction will cause a search, character by character, of the "N" consecutive characters with the starting address "FROM" for the character "CHAR R?".

If no such character is found within the "N" characters being tested the instruction "GOTO INST" will be executed. If the character is found, the program continues in sequence.

NOTE:

1. Core storage is left undisturbed.
2. The count of the number of characters tested before the character was found will be found in index register #15. (For example, if the character desired was in the first location searched, index register #15 would contain zero).
3. The previous contents of index register #15 are lost.

TEST SENSE SWITCH

op-code	mnemonic	variable operands
17457	TSS	SW GOTO INST

This instruction enables the user to test one of the 15 sense switches on the computer console. If the switch being tested is in the up position, the "GOTO INST" will be executed. Otherwise the program continues in sequence.

NOTE:

1. The right most 15 accumulator switches on the console are defined as sense switches, numbered 1 through 15 respectively.

CHARACTER ZONE TESTS

op-code	mnemonic	variable operands
		FROM GOTO INST

These instructions test the two zone bits of the character at location "FROM". The following table defines the instruction mnemonic and the condition under which the "GOTO INST" will be executed.

op-code	mnemonic	condition
17462	IFX	Zone portion equal to 60 octal (plus)
17463	IFY	Zone portion equal to 40 octal (minus)

Note: 1. Core storage is left undisturbed.

LOAD PROGRAM COUNTER

op-code	mnemonic	variable operands
17454	LDCTR	CTR V

This instruction causes program counter "CTR" to be set to the value "V".

NOTE:
1. Core storage is left undisturbed.
2. "V" must not be a negative value.
3. If the "TSTCTR" instruction is to be executed prior to the instructions to be repeated, the value "V" must be the number of repeats +1.
4. Fifteen (15) program counters are available in the Bus-Pak II system.

TEST PROGRAM COUNTER

op-code	mnemonic	variable operands
17455	TSTCTR	CTR GOTO INST

This instruction decrements the contents of program counter "CTR" and tests the results. If the result is not zero, the instruction "GOTO INST" will be executed. Otherwise, the program continues in sequence.

NOTE:
1. Core storage is left undisturbed.
2. This instruction is normally executed immediately following the instructions being repeated with the "GOTO" instruction transferring to the first instruction being repeated.

PROGRAM SWITCH CONTROL INSTRUCTIONS

op-code	mnemonic	variable operands
		SW

These instructions are used to control the status of the "SW" internal program switch. Fifteen (15) program switches are available in the Bus-Pak II system. The following table defines the instructions mnemonic to be used and its effect on the program switches.

op-code	mnemonic	effect
17464	SET	Sets the switch "SW" to the ON state
17465	CLEAR	Sets the switch "SW" to the OFF state

NOTE:
1. If "SW" is zero (Ø), the instruction affects all fifteen (15) program switches.
2. Core storage is left undisturbed.

TEST PROGRAM SWITCH

op-code	mnemonic	variable operands
1746Ø	TPS	SW GOTO INST

This instruction tests program switch "SW" and if switch "SW" is set, the instruction "GOTO INST" is executed. Otherwise, the program continues in sequence.

NOTE: 1. Core storage is left undisturbed.

TEST ACCUMULATOR OVERFLOW

op-code	mnemonic	variable operands
17461	TAO	AC GOTO INST

The overflow indicator of the accumulator specified by "AC" is tested. If an overflow has occurred in the specified accumulator, the instruction "GOTO INST" is executed. Otherwise the program continues in sequence.

NOTE:
1. The overflow indicator of the specified accumulator is cleared by this instruction.
2. The overflow indicator is also cleared when a "CLRAC" instruction is executed.
3. If an overflow has occurred on an accumulator operation, the result of the operation is indeterminate.

ARITHMETIC INSTRUCTIONS

All arithmetic operations must be done in one of the 15 accumulators available in the Bus-Pak II programming system. These accumulators are numbered 1 through 15, each of which can contain a positive or negative value whose magnitude does not exceed ± 3 4 3 5 9 7 3 8 3 6 7. If the magnitude exceeds the maximum, overflow occurs.

An overflow indicator is associated with each of the 15 accumulators.

The signs of the contents of the accumulators are computed algebraically consistent with the sign of the data in the accumulator itself and the sign of the data being used in the computation.

The sign of the data stored in core storage may be found over the units position of the data being computed. If the zone of the units position is equal to 40 octal, the value is negative. Otherwise the sign is assumed positive.

The sign when stored into core storage after an accumulation will always be placed over the units position of the data being stored. A negative sign will be equal to 40 octal and a positive sign will be equal to 60 octal.

Signs over other than the units position of data being used will be ignored.

For decimal point alignment see "Accumulator Shift Instructions". The imaginary decimal point location must be taken into consideration in all arithmetic operations, especially in the multiplication and division operations. The following rules apply:

Multiplication

The number of decimal places in the product will be the sum of the decimal places of the multiplicand and the multiplier.

Division

The number of decimal places in the quotient will be the number of decimal places of the dividend minus the number of decimal places of the divisor.

Addition and Subtraction

When adding or subtracting different values, the decimal point must be aligned correctly or incorrect results will occur.

CLEAR ACCUMULATOR

op-code	mnemonic	variable operands
1741Ø	CLRAC	AC

The contents of accumulator "AC" are set to plus zero (+Ø).

NOTE:
1. The original contents of accumulator "AC" are lost.
2. The overflow indicator associated with accumulator "AC" is cleared.
3. If "AC" is made zero (Ø), all accumulators are set to plus zero (+0) and all their associated overflow indicators are cleared.

LOAD ACCUMULATOR

op-code	mnemonic	variable operands
175Ø1	LOADAC	AC N FROM

The "N" consecutive characters with the starting address "FROM" are placed into the accumulator "AC".

NOTE:
1. The original contents of accumulator "AC" are lost.
2. The sign of accumulator "AC" is made equal to the sign over the units position of the "N" consecutive characters at "FROM".
3. Core storage is left undisturbed.
4. The overflow indicator associated with accumulator "AC" is left undisturbed.

EXAMPLE;

LOADAC 3 3 5Ø1

CONTENTS OF CORE STORAGE	before	2	3	$\bar{4}$
	after	2	3	$\bar{4}$
CORE STORAGE ADDRESSES		5		
		Ø		
		1		

CONTENTS OF ACCUMULATOR 3	before	+35291
	after	-234

DEPOSIT ACCUMULATOR

op-code	mnemonic	variable operands
174Ø6	DEPAC	AC N TO

The right most "N" consecutive characters of the contents of accumulator "AC" are deposited into the "N" consecutive characters with the starting address "TO".

NOTE:
1. The contents of accumulator "AC" are left undisturbed.
2. The original "N" consecutive characters at "TO" are lost.
3. The sign of the accumulator is placed in the zone portion of the units position of the data deposited.
4. The overflow indicator associated with accumulator "AC" is left undisturbed.

EXAMPLE:

DEPAC 1Ø 3 5Ø1

CONTENTS OF CORE STORAGE	before	3	7	9
	after	5	2	$\overline{1}$
CORE STORAGE ADDRESSES		5Ø		
		1		

CONTENTS OF ACCUMULATOR 1Ø	before	-39521
	after	-39521

ADD TO ACCUMULATOR

op-code	mnemonic	variable operands
175Ø3	ADDAC	AC N FROM

The "N" consecutive characters with the starting address "FROM" are algebraically added to the contents of accumulator "AC" and the result placed in accumulator "AC".

NOTE:

1. Core storage is left undisturbed.
2. The sign over the units position of the "N" consecutive characters at "FROM" is taken into consideration.
3. The original contents of accumulator "AC" are lost.
4. If the result of the addition produced a value whose magnitude exceeds the capacity of the accumulator, the associated overflow indicator will be set. The result itself is indeterminate.
5. Zero answers will always be made +Ø.

EXAMPLE:

ADDAC 5 3 5Ø1

CONTENTS OF CORE STORAGE	before	Ø	5	$\overline{3}$
	after	Ø	5	$\overline{3}$
CORE STORAGE ADDRESSES		5 Ø 1		

CONTENTS OF ACCUMULATOR 5	before	+2ØØ
	after	+147

SUBTRACT FROM ACCUMULATOR

op-code	mnemonic	variable operands
174Ø3	SUBAC	AC N FROM

The "N" consecutive characters with the starting address "FROM" are algebraically subtracted from the contents of accumulator "AC" and the result placed into accumulator "AC".

NOTE:
1. The original contents of accumulator "AC" are lost.
2. Core storage is left undisturbed.
3. The sign over the units position of the "N" consecutive characters being added is taken into consideration.
4. If the result of the subtraction produces a value whose magnitude exceeds the capacity of the accumulator, the associated overflow indicator will be set. The result itself is indeterminate.

EXAMPLE:

SUBAC 5 3 5Ø1

CONTENTS OF CORE STORAGE	before	Ø	2	$\overline{5}$
	after	Ø	2	$\overline{5}$
CORE STORAGE ADDRESSES		5		
		Ø		
		1		

CONTENTS OF ACCUMULATOR 5	before	-5ØØ
	after	-475

MULTIPLY ACCUMULATOR

op-code	mnemonic	variable operands
174Ø4	MULAC	AC N FROM

The contents of accumulator "AC" are multiplied by the "N" consecutive characters with the starting address "FROM" and the result placed in accumulator "AC".

NOTE:

1. The original contents of accumulator "AC" are lost.
2. Core storage is left undisturbed.
3. If the result of the multiplication produces a value whose magnitude exceeds the capacity of the accumulator, the associated overflow indicator will be set. The result itself is indeterminate.

EXAMPLE:

MULAC 3 3 5Ø1

CONTENTS OF CORE STORAGE	before	Ø	1	+2
	after	Ø	1	+2
CORE STORAGE ADDRESSES		5Ø 1		

CONTENTS OF ACCUMULATOR 3	before	+12
	after	+144

DIVIDE INTO ACCUMULATOR

op-code	mnemonic	variable operands
174Ø5	DIVAC	AC N FROM

The contents of accumulator "AC" are divided by the "N" consecutive characters with the starting address "FROM". The results are placed in accumulator "AC".

NOTE:

1. If the "N" consecutive characters with the starting address "FROM" are greater than the contents of accumulator "AC", the result of the division is zero.
2. The integral part of the quotient is taken as the result and the fractional part (the remainder) is discarded.
3. The original contents of accumulator "AC" are lost.
4. Core storage is left undisturbed.
5. Overflow may not be set because of a division.

EXAMPLE:

DIVAC 5 3 5Ø1

CONTENTS OF CORE STORAGE	before	Ø	Ø	5
	after	Ø	Ø	5
CORE STORAGE ADDRESSES		5 Ø 1		

CONTENTS OF ACCUMULATOR 5	before	+23
	after	+4

ADD TO MEMORY

op-code	mnemonic	variable operands
17523	ADDMEM	AC N TO

The contents of accumulator "AC" are algebraically added to the "N" consecutive characters with the starting address "TO". The results are placed into the "N" consecutive character positions with the starting address "TO".

NOTE:

1. The contents of accumulator "AC" are left undisturbed.
2. The sign over the units position of the "N" consecutive characters at "TO" is taken into consideration.
3. The original "N" consecutive characters at "TO" are lost.
4. If the result of the addition produced a value whose magnitude exceeded the capacity of the accumulator, the associated overflow indicator will be set. The result itself is indeterminate.
5. The sign of the result is placed over the units position of the "N" consecutive characters at "TO".

EXAMPLE:

ADDMEM 15 3 5Ø1

CONTENTS OF CORE STORAGE	before	Ø	7	+5	3	9
	after	1	1	+Ø	3	9
CORE STORAGE ADDRESSES		5Ø 51				5Ø 55

CONTENTS OF ACCUMULATOR 5	before	+35
	after	+35

SUBTRACT FROM MEMORY

op-code	mnemonic	variable operands
17522	SUBMEM	AC N TO

The contents of accumulator "AC" are algebraically subtracted from the "N" consecutive characters with the starting address "TO". The results are placed into the "N" consecutive character positions with the starting address "TO".

NOTE:

1. The contents of accumulator "AC" are left undisturbed.
2. The sign over the units position of the "N" consecutive characters at "TO" is taken into consideration.
3. The original "N" consecutive characters at "TO" are lost.
4. If the result of the subtraction produced a value whose magnitude exceeds the capacity of the accumulator, the associated overflow indicator will be set. The result itself is indeterminate.
5. The sign of the result is placed over the units position of the "N" consecutive characters at "TO".

EXAMPLE:

SUBMEM 1Ø 3 5Ø1

CONTENTS OF CORE STORAGE	before	Z	Ø	5	$\bar{Ø}$	9	9
	after	Z	3	5	$\bar{Ø}$	9	9
CORE STORAGE ADDRESSES		5 Ø Ø	5 Ø 1				5 Ø 5

CONTENTS OF ACCUMULATOR 1Ø	before	+3ØØ
	after	+3ØØ

MULTIPLY MEMORY

op-code	mnemonic	variable operands
17524	MULMEM	AC N TO

The contents of the "N" consecutive characters with the starting address "TO" are multiplied by the contents of accumulator "AC". The results are placed into the "N" consecutive character positions with the starting address "TO".

NOTE:

1. The contents of accumulator "AC" are left undisturbed.
2. The original "N" consecutive characters at "TO" are lost.
3. If the result of the multiplication produced a value whose magnitude exceeded the capacity of an accumulator, the associated overflow indicator will be set. The result itself is indeterminate.
4. The sign of the result is placed over the units position of the "N" consecutive characters with the starting address "TO".

EXAMPLE:

MULMEM 1Ø 4 5ØØ

CONTENTS OF CORE STORAGE	before	Ø	Ø	1	2
	after	Ø	1	4	4
CORE STORAGE ADDRESSES		5 Ø Ø			

CONTENTS OF ACCUMULATOR 1Ø	before	+12
	after	+12

DIVIDE INTO MEMORY

op-code	mnemonic	variable operands
17525	DIVMEM	AC N TO

The "N" consecutive characters with the starting address "TO" are divided by the contents of accumulator "AC". The result is placed into the "N" consecutive character positions with the starting address "TO".

NOTE:

1. The contents of accumulator "AC" are left undisturbed.
2. The original "N" consecutive characters at "TO" are lost.
3. The integral part of the quotient is taken as the result and the fractional part (the remainder) is discarded.
4. If the contents of accumulator "AC" are greater than the "N" consecutive characters with the starting address "TO", the result of the division is plus zero (+Ø).
5. Overflow may not be set because of division.

EXAMPLE:

DIVMEM 1Ø 4 5ØØ

CONTENTS OF CORE STORAGE	before	Ø	1	4	4
	after	Ø	Ø	1	2
CORE STORAGE ADDRESSES		5 Ø Ø			

CONTENTS OF ACCUMULATOR 1Ø	before	+12
	after	+12

ACCUMULATOR SHIFT INSTRUCTIONS

The following instructions may be used for the alignment of decimal positions in arithmetic operations.

The imaginary decimal point location must be taken into consideration for all arithmetic operations. (See Arithmetic Instructions).

SHIFT ACCUMULATOR LEFT

op-code	mnemonic	variable operands
175Ø6	SHFTL	AC T

This instruction causes the contents of accumulator "AC" to be shifted left "T" character positions.

NOTE:
1. The contents of accumulator "AC" are in essence multiplied by $1Ø^{T}$.
2. In shifting an accumulator left, the resulting contents of the accumulator may not exceed the capacity of the accumulator.
3. The sign of the accumulator is left undisturbed.
4. Overflow may be set.

EXAMPLE:

SHFTL 13 3

CONTENTS OF ACCUMULATOR 13	before	+321
	after	+321ØØØ

SHIFT ACCUMULATOR RIGHT

op-code	mnemonic	variable operands
175Ø7	SHFTR	AC T

This instruction causes the contents of the accumulator "AC" to be shifted right "T" character positions.

NOTE:

1. The contents of accumulator "AC" are in essence divided by $1Ø^{T}$.
2. If the contents of the accumulator being shifted are less in value than $1Ø^{T}$, the result of the shift will be plus zero (+Ø).
3. The sign of the accumulator is left undisturbed.

EXAMPLE:

SHFTR 1Ø 2

CONTENTS OF ACCUMULATOR 1Ø	before	+52173
	after	+521

SHIFT ACCUMULATOR RIGHT AND ROUND

op-code	mnemonic	variable operands
1751Ø	SHFTRR	AC T

The contents of accumulator "AC" are shifted right "T" character positions. A five (5) is added to the last character being shifted to increment the resulting contents of the accumulator by one (1) if the last character was a value of five or more.

NOTE:

1. The contents of accumulator "AC" are in essence divided by $1Ø^{T}$.
2. If the last character being shifted is equal to or greater than 5, a one is added to the units position of the resulting contents of the accumulator.
3. If the contents of the accumulator being shifted are less than $5 \times 1Ø^{T-1}$, the result of the shift will be plus zero (+Ø).
4. The sign of the accumulator is left undisturbed.

EXAMPLE:

SHFTRR 3 3

CONTENTS OF ACCUMULATOR 3	before	+725731
	after	+726

EDITING FEATURE

Bus-Pak II has the ability to automatically punctuate numeric data for eventual output on printed reports. Through the use of a control word, any format of punctuation may be obtained.

In editing, certain laws must be followed to insure correct punctuation of data. Some of these laws are common to both editing instructions and will be defined below. The laws not common to both editing instructions will be defined in the description of their respective edit instruction.

COMMON LAWS

1. The "N" field of the edit instruction must indicate the length of the control word being used, not the number of characters being edited.
2. The number of blanks and zeroes contained in a control word must equal the total number of characters to be edited.
3. The "FROM" address field must always address the first character to be edited.
4. The "TO" address field must always address the first character of the control word being used in the output field.

CONTROL WORD FORMAT

The control word is made up of certain characters which govern the editing operation. Following is a list of these characters and their effect on the editing operation:

Character	Effect
b (blank)	Is replaced by its corresponding digit of the "N" consecutive characters being edited. If its corresponding digit is zero (12-octal) and zero suppressing is active, the current filler code will replace the blank.
Ø (zero)	Is replaced by its corresponding digit of the "N" consecutive characters being edited. If its corresponding digit is zero (12-octal) and zero suppressing is active, the current filler code will replace the zero. Zero suppressing is then made inactive.
, (comma)	Remains in the edited character position where it was originally placed. Is replaced by the current filler code only if zero suppressing is active.
. (decimal point)	Remains in the edited character position where it was originally placed. Is replaced by the current filler code only if zero suppressing is active.
$ (dollar sign)	Remains in the edited character position where it was originally placed.
* (asterisk)	Indicates the asterisk protection feature is active. Asterisk (*) is made the current filler code and it will replace all characters being suppressed. Normally the filler code is blank (20-octal).

—→ (floating dollar sign)	Indicates the floating dollar sign feature is active. Dollar sign ($) is made the current filler code and it will replace all characters being suppressed. As each suppressed character is replaced by a dollar sign, the preceding character is made blank (20-octal). Normally the filler code is blank (20-octal).
CR (Credit Symbol)	Remains in the edited character positions where it was originally placed only if the sign of the data being edited is minus. If the sign is plus, the "CR" is replaced by blanks. Has this effect only if CR are the right most two characters in the control word.
- (minus sign)	Remains in the edited character position where it was originally placed only if the sign of the data being edited is minus. If the sign is plus, the "-" is replaced by a blank. Has this effect only if the symbol "-" is the right most character of the control word.

EDITING EXAMPLES

DATA	EDIT CONTROL WORD	RESULTS
ØØ593ØØ	$ b b , b b Ø . b b	$ b b b 5 9 3 . Ø Ø
ØØ593ØØ	$ b b , b b Ø . b b -	$ b b b 5 9 3 . Ø Ø b
Ø132599	$ b b , b b Ø . b b CR	$ b 1 , 3 2 5 . 9 9 CR
ØØØØØØØ	$ b b , b b Ø . b b	$ b b b b b b . Ø Ø
ØØØ5278	$ * b , b b Ø . b b -	$ * * * * 5 2 . 7 8 b
ØØØØØØ5	$ * b , b b Ø . b b CR	$ * * * * * * . Ø 5 b b
ØØØ7563	→b b , b Ø b . b b CR	$ 7 5 . 6 3 CR
ØØØØ325	→b b , b Ø b . b b -	$ Ø 3 . 2 5 -

EDIT

op-code	mnemonic	variable operands
17416	EDIT	N FROM TO

This instruction causes the numeric data with the starting address "FROM" to be moved to and edited by the control word contained in the "N" consecutive characters with the starting address "TO".

NOTE:
1. The numeric data at "FROM" are left undisturbed.
2. The control word to be used in the editing operation must first be moved to the "N" consecutive characters at "TO".
3. "N" must specify the length of the control word being used.

MOVE AND EDIT

op-code	mnemonic	variable operands
17417	MVEDIT	N FROM TO BY

This instruction operates similarly to the "EDIT" instruction except that the control word need not be previously placed into the "N" consecutive characters with the starting address "TO". This instruction operates in the following sequence:

1. The "N" consecutive characters at "BY" are moved and expanded to the "N" consecutive characters with the starting address "TO".
2. Then the numeric data at "FROM" is moved to and edited by the "N" consecutive characters at "TO".

NOTE:
1. The numeric data at "FROM" is left undisturbed.
2. The control word specified by the "BY" address must be defined using the "TEXT" pseudo instruction and must also be in "ANELEX" code.
3. The editing operation is affected by the characters previously defined.

ALPHANUMERIC AND NUMERIC COMPARISON INSTRUCTIONS

The following instructions are divided into two distinct types of comparisons. Alphanumeric, which may be used to compare data internally in core storage and numeric which is used to compare the contents of an accumulator to data in core storage.

ALPHANUMERICAL COMPARISONS

op-code	mnemonic	variable operands
		N FROM TO GOTO INST

With all the alphanumeric comparison instructions, the "N" consecutive characters at the starting address "FROM" are compared alphanumerically to the "N" consecutive characters with the starting address "TO". The execution of the "GOTO INST" is dependent on the type of comparison made and its results. The following table indicates the types of comparisons that can be made and the conditions checked for. The "GOTO INST" is executed only if the condition is met.

op-code	mnemonic	"GOTO INST" executed if
17466	CMPEQU	"FROM" equal to "TO"
17467	CMPUEQ	"FROM" unequal to "TO"
17471	CMPGRT	"FROM" greater than "TO"
1747Ø	CMPLES	"FROM" less than "TO"
17424	CMPGEQ	"FROM" greater or equal to "TO"
174Ø7	CMPLEQ	"FROM" less or equal to "TO"

NOTE: 1. See the collating sequence specified in Appendix I.

NUMERICAL COMPARISON

op-code	mnemonic	variable operands
		AC N TO GOTO INST

With all the numeric comparison instructions the contents of accumulator "AC" are compared numerically to the "N" consecutive characters starting at address "TO". The following table defines the types of comparisons that can be made. The table indicates the instruction mnemonic and the condition that is to be met for the "GOTO INST" instruction to be executed.

op-code	mnemonic	"GOTO INST" executed if
17472	EQUAL	"AC" equal to "TO"
17473	UNEQUAL	"AC" unequal to "TO"
17475	GREATER	"AC" greater than "TO"
17474	LESS	"AC" less than "TO"
17477	GRTEQU	"AC" greater or equal to "TO"
17476	LESEQU	"AC" less or equal to "TO"

NOTE:

1. Core storage is left undisturbed.
2. The sign (zone) of the least significant digit of the "N" consecutive characters at "TO" are taken into consideration.
3. All other zones of the "N" consecutive characters at "TO" are ignored.
4. The contents of accumulator "AC" are left undisturbed.

INDEX CONTROL INSTRUCTIONS

Fifteen (15) index registers are available for address modification in the Bus-Pak II system. Double indexing is also available. Normal indexing is not required to make use of double indexing.

LOAD INDEX WITH VALUE

op-code	mnemonic	variable operands
1745Ø	LDIDX	IDX V

This instruction sets the contents of index register "IDX" to the value "V".

NOTE:
1. Core storage is left undisturbed.
2. To clear an index register, "V" would be made equal to zero (Ø).
3. "V" may be an address or a positive or negative number.
4. The previous contents of index register "IDX" are lost.

ADD VALUE TO INDEX

op-code	mnemonic	variable operands
17451	ADDIDX	IDX V

This instruction causes the value "V" to be algebraically added to the contents of index register "IDX".

NOTE:
1. Core storage is left undisturbed.
2. The value "V" may be a positive or negative number, or address.
3. The previous contents of index register "IDX" are lost.

DEPOSIT INDEX

op-code	mnemonic	variable operands
17453	DEPIDX	IDX TO

This instruction deposits the contents of index register "IDX" in the single storage location "TO".

NOTE:
1. The contents of the index register are left undisturbed.
2. An index register is normally deposited into the "N", "V", or address fields of instructions.

LOAD INDEX WITH DECIMAL VALUE

op-code	mnemonic	variable operands
17452	LDIDEC	IDX N FROM

This instruction loads index register "IDX" with the numeric value of the "N" consecutive characters with the starting address "FROM".

NOTE:
1. Core storage is left undisturbed.
2. The value of the "N" consecutive character at "FROM" may be positive or negative.
3. The previous contents of index register "IDX" are lost.

LOAD DOUBLE INDEX REGISTER

op-code	mnemonic	variable operands
175Ø2	LDBLIDX	IDX

The index register number "IDX" is loaded into the double index register.

NOTE:
1. Core storage is ledt undisturbed.
2. "IDX" will be the index register number of the index register to be used whenever double indexing is specified.

INPUT-OUTPUT INSTRUCTIONS

The following pages contain the input-output instructions to input data to, or output data from, the Bus-Pak II operating system.

TELEPRINTER INPUT-OUTPUT INSTRUCTIONS

The following instructions permit input from or output to the on-line teleprinter with complete forms control.

TYPE A LINE

op-code	mnemonic	variable operands
17431	TYPE	N FROM

The "N" consecutive characters with the starting address "FROM" are typed on the on-line teleprinter.

NOTE:
1. Core storage is left undisturbed.
2. The character information to be typed must always be in BCD code one character per storage location.
3. Indexing, double indexing, and/or indirect addressing may be used with the address "FROM".

TYPE TEXT INFORMATION

op-code	mnemonic	variable operands
17434	TYPTXT	FROM

The textual information with the starting address "FROM" is typed on the on-line teleprinter.

NOTE:
1. Core storage is left undisturbed.
2. The textual information must be defined using the "TEXT" pseudo instructions.
3. The "TELETYPE" instructions must precede the definition of the "TEXT" information. (See also "TEXT" instruction).
4. Indexing, double indexing, and/or indirect addressing may be used with the address "FROM".

TYPE TABULATION

op-code	mnemonic	variable operands
17433	TAB	

This instruction causes the on-line teleprinter to be spaced to the next tab location.

NOTE:
1. Tabs are assumed 1Ø character positions apart.
2. The number of spaces spaced will be the remaining number of spaces needed to position the on-line teleprinter at the next tab location.

TYPE CARRIAGE RETURN

op-code	mnemonic	variable operands
17432	TCR	

This instruction causes a line feed and carriage return to be typed on the on-line teleprinter.

NOTE: 1. Resets the tabulation control to the first character position on the on-line teleprinter.

TYPE-IN INFORMATION

op-code	mnemonic	variable operands
175Ø4	TYPEIN	N TO

This instruction reads in "N" consecutive characters being typed on the on-line teleprinter and places these characters from left to right at address "TO".

NOTE:

1. All characters typed will be stored one character per storage location, in BCD code.
2. All initial carriage returns typed will be ignored and do not occupy core storage locations.
3. The "FIGS" and "LTRS" keys may be used whenever necessary and do not occupy core storage locations.
4. Typing is terminated either when "N" characters have been typed or when a carriage return is typed.
5. Carriage return does not occupy a core storage location.
6. The "BELL" key is used for tabulations. Tabs are assumed set 1Ø character positions apart. The actual spaces spaced will be the remaining number of spaces to the next tab locations.
7. The "BLANK" key is used as a "backspace" key. When depressed, the character previously typed will be ignored and the next character typed will replace it. If no character is typed, the character backspaced will remain in the area.
8. Upon completion of the "TYPEIN" instruction, a count of the number of characters typed in may be found in index register #15.
9. The previous contents of index register #15 are lost.
10. Indexing, double indexing, and/or indirect addressing may be used with the address "TO".

INQUIRY

op-code	mnemonic	variable operands
1753Ø	INQUIRY	GOTO INST

Bus-Pak II has available an inquiry feature which provides a direct and immediate means of communication between the operator and the users program.

The inquiry feature is especially valuable when used with the Micro-Disk system. It can be used to retrieve data stored on a Micro-Disk record. A personnel record or inventory stock-status record needed by management can be requested by the operator and made available to management in a short time.

Inquiries though, may only be made if the "INQUIRY" instruction is executed by the operating program and the carriage return key on the on-line teleprinter was depressed, which sets the inquiry indicator in Bus-Pak II.

Every program written reaches a point in execution when all the data currently in the machine has been processed and the program is ready to accept more data. At this point the "INQUIRY" instruction is executed. If the carriage return key was depressed, the "GOTO INST" will be executed. Otherwise the program continues in sequence.

PUNCHED CARD INPUT-OUTPUT INSTRUCTIONS

The following instructions permit input and output from punched card equipment.

READ A CARD

op-code	mnemonic	variable operands
17425	RDCRD	TO GOTO INST

The 80 columns of information punched on the card in the card reader hopper will be read and stored at the 80 consecutive locations with the starting address "TO".

After this information has been stored the reading of the next card in the card reader hopper is initialized and control is returned to the users program. This is done so that the users program can process the information on the card just read while the next card is being read. The overlapping of these two functions saves a considerable amount of time in that the user need not wait the entire three (3) milliseconds necessary to read a card.

If the End-of-File button on the card reader has been pressed after the last card was read, the instruction "GOTO INST" will be executed. Otherwise the program continues in sequence.

NOTE:

1. As there are 80 columns on a punched card, 80 columns of information will always be transferred to the 80 consecutive locations, from left to right at address "TO".
2. See also "Card Reader Operating Features".
3. The character information stored will be in BCD code, one character per storage location.
4. If, when processing card files, it is necessary to stop the operation of the program in order to make corrections to erroneously punched cards, normally the card to be corrected will be the second card from the top in the card reader stacker. When the correction is made, that card and the last card in the card reader stacker must be placed in the card reader hopper, accumulator switch one (1) must be set to the up condition and then the continue key depressed.
5. Indexing, double-indexing, and indirect addressing may be used with the address "TO".

PUNCH A CARD

op-code	mnemonic	variable operands
17426	PUNCRD	FROM

The 80 consecutive characters with the starting address "FROM" are punched on the next blank card in the card punch hopper. Control is immediately returned to the users program so that processing can continue. The accumulation of the next card image may be started immediately in the same output area.

NOTE:

1. Core storage is left undisturbed.
2. Bus-Pak II will always punch 80 consecutive characters.
3. The character information to be punched must always be in BCD code, one character per storage location.
4. Indexing, double indexing, and/or indirect addressing may be used with the address "FROM".
5. See also "Card Punch Operating Features".
6. If a card feed check, card jam, or empty hopper occurs, the system will loop until it has been corrected.

HIGH SPEED PRINTER OUT-PUT INSTRUCTIONS

The following instruction permits data located at any location in core storage to be printed on the high speed printer and also permits complete forms control (spacing) of the printed data.

PRINT A LINE

op-code	mnemonic	variable operands
17427	PRNLIN	FROM

The 120 consecutive characters with the starting address "FROM" will be printed.

NOTE:

1. Core storage is left undisturbed.
2. Bus-Pak II will always print 120 consecutive characters.
3. The character information to be printed must always be in BCD code, one character per storage location.
4. Indexing, double indexing, and/or indirect addressing may be used with the address "FROM".
5. See also "High Speed Printer Operating Features".

SPACE PRINTER

op-code	mnemonic	variable operands
1743Ø	SPACE	CHAR R?

This instruction causes the high speed printer to space the paper as specified by "CHAR R?" (when "?" is one of the following characters):

CHAR	SPACING	STANDARD TAPE	TIMING
A	Double space	2 lines	2 x 16 MS
B	Triple space	3 lines	3 x 16 MS
C	Four spaces	4 lines	4 x 16 MS
D	Five spaces	5 lines	5 x 16 MS
E	Six spaces	6 lines	6 x 16 MS
F	Seven spaces	7 lines	7 x 16 MS
G	Eight spaces	8 lines	8 x 16 MS
H	Nine spaces	9 lines	9 x 16 MS
I	Ten spaces	10 lines	10 x 16 MS
1	Skip to Channel 1	2 lines	2 x 16 MS
2	Skip to Channel 2	3 lines	3 x 16 MS
3	Skip to Channel 3	6 lines	6 x 16 MS
4	Skip to Channel 4	11 lines	11 x 16 MS
5	Skip to Channel 5	22 lines	22 x 16 MS
6	Skip to Channel 6	33 lines	33 x 16 MS
7	Skip to Channel 7	Top of form	520 MS for 66 lines

NOTE:

1. Spacing is always immediate.
2. If spacing to a channel is specified, the channel need not be specified by "CHAR R?", but may be represented by the channel number only.
3. The timings for channel skipping refer to the use of a standard carriage control tape.

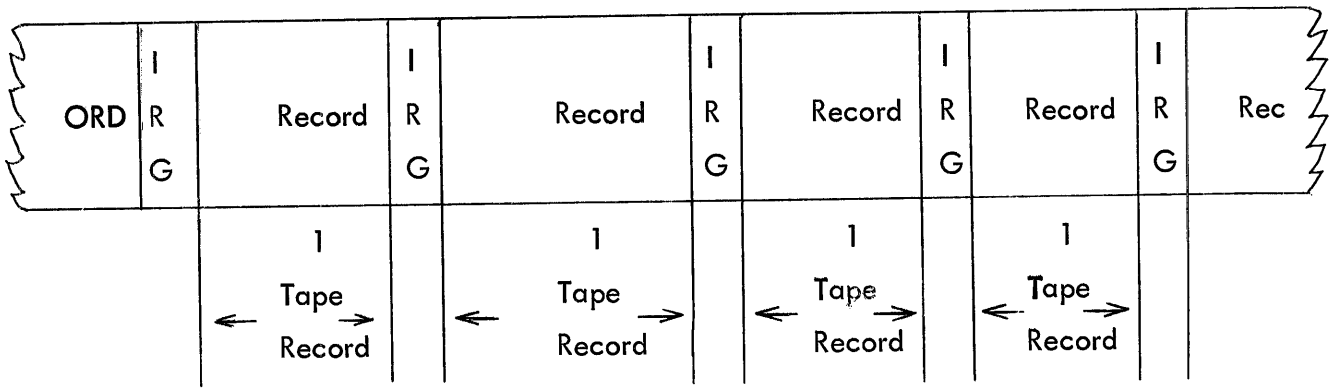
MAGNETIC TAPE

ORGANIZATION OF DATA

Information is recorded on magnetic tape character by character in the binary-coded-decimal (BCD) mode. Even parity is maintained for all characters, that is, if the character being written contains an odd number of bits, the parity bit is recorded to make the number of bits even. When the character is read and transferred to core storage, the parity bit is not transferred.

TAPE RECORDS, INTER-RECORD GAPS

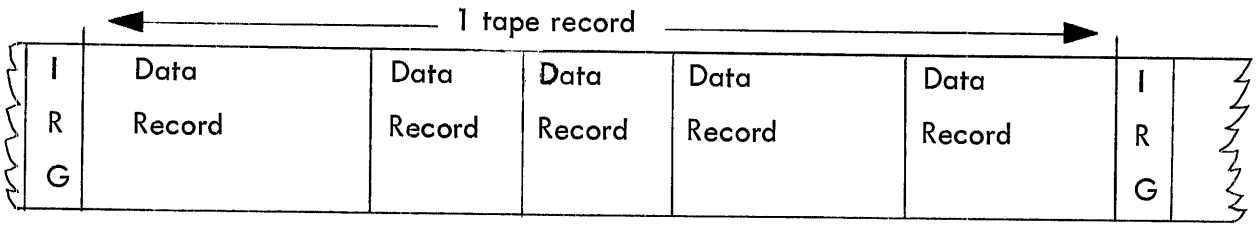
A tape record is a sequence of characters physically separated from other sequences of characters by an inter-record gap. This inter-record gap is approximately 3/4 inches of erased tape. During reading, the sequence of characters starting with the first character sensed after an inter-record gap to the next inter-record gap is read and placed into core storage where specified.



BLOCKING

Data records and tape records are quite different. Many data records may be contained within a tape record. This is done to conserve space and increase storage capacity on magnetic tape reels.

Blocking is the process of writing two or more data records as a single tape record.



TAPE MARK (END OF FILE MARK)

A tape mark is written on tape to indicate the logical end of a file of information. The tape mark is a single character, but is considered a tape record because it is preceded and followed by an inter-record gap. It is also considered a data record when read by the program.

TAPE FILE

A tape file is a series of tape records related to one another and followed by a tape mark.

END-OF-FILE

A single tape-reel may contain any number of files of information. A tape mark would then separate each file of information from another file of information. When reading, the program is notified of an End-of-File condition so that it may choose a pre-determined sequence of instructions to process such a condition.

DENSITY

The density of a magnetic tape is a measure of the number of consecutive characters which may be written on one inch of tape. Two densities are available, low density (200 characters to the inch) and high density (556 characters to the inch). Instructions are available to set the desired density. If no density is specified, high density (556 characters to the inch) is assumed.

ALTERNATE TAPES

When alternate tapes are specified, encountering End Point causes Bus-Pak II to automatically reference the alternate tape specified. When End Point on that tape is encountered, reference to the original tape will be automatic.

OPEN MAGNETIC TAPE UNIT

op-code	mnemonic	variable operands
17435	OPEN	U B L

This instruction causes the magnetic tape controls for tape unit "U" to be initialized to read or write records which contain "B" number of data records "L" characters in length as one tape record.

NOTE:

1. "L" must be a multiple of three (3).
2. "B" must be at least the value of one (1).
3. This instruction does not rewind the tape.
4. This instruction initializes the tape density to "High Density" (556 BPI).
5. On the initial OPENing of tape unit "U", an internal buffer is assigned. It is therefore necessary to originally initialize tape unit "U" to the largest tape record that may be read or written on that unit.
6. This instruction may again be executed to indicate a different blocking factor and record length next to be read or written.
7. The internal buffers assigned begin at location 7777_8 and project down into users memory. The size of each buffer area may be calculated by the following formula:

$$\frac{B \times L}{3} = \text{size of buffer assigned}$$

8. The last location available to the user may be calculated by using the following formula:

$$\left(4095_{10} - \left(\begin{array}{c}\text{the sum of all the magnetic tape}\\ \text{and micro-disk buffers assigned}\end{array}\right)\right) - 1 = \text{the last available users storage location}_{10}$$

SET HIGH DENSITY

op-code	mnemonic	variable operands
17446	HIDEN	U

This instruction will cause the input-output controls for magnetic tape unit "U" to be set to read or write in high density (556 char/in).

NOTE: 1. If no density is specified, high density (556 char/in) is assumed.

SET LOW DENSITY

op-code	mnemonic	variable operands
17447	LODEN	U

This instruction will cause the input-output controls for magnetic tape unit "U" to be set to read or write in low density (2ØØ char/in).

NOTE: 1. If no density is specified, high density (556 char/in) is assumed.

ASSIGN ALTERNATE TAPE

op-code	mnemonic	variable operands
17526	ALTTAPE	U ALT

This instruction will assign tape unit "ALT" as an alternate tape unit for unit "U".

NOTE: 1. The alternate tape feature may only be eliminated by the re-execution of a OPEN or BEGIN instruction.

REWIND TAPE REEL

op-code	mnemonic	variable operands
1744Ø	REWIND	U

This instruction causes the tape reel on tape unit "U" to rewind to load point.

NOTE: 1. Rewinding of tape reels must be done either before any or after all data records and end of files have been processed. Records may be lost or extra records read if a REWIND operation is performed other than as stated above.

READ MAGNETIC TAPE

op-code	mnemonic	variable operands
17436	RDTAPE	U TO GOTO INST

This instruction causes one data record from tape unit "U" to be placed in the storage locations with the starting address "TO". If an End-of-File or end-point on tape is encountered, the "GOTO INST" instruction will be executed. Otherwise, the program continues in sequence.

NOTE:

1. If end-point is encountered during a read operation, and an alternate tape unit was specified, Bus-Pak II will automatically reference the alternate tape unit whenever unit "U" is referenced.
2. When end-point is encountered on the alternate tape, Bus-Pak II will automatically re-reference tape unit "U".
3. The instruction "GOTO INST" will be executed when end-point is encountered even though an alternate tape was specified so that header labels may be checked on the alternate tape unit.
4. An internal buffer is assigned to tape unit "U" and each data record read will be extracted from this buffer area. Only when the buffer is empty will tape be moved to read the next tape record into the buffer area.
5. Indexing, double indexing and indirect addressing may be used with the "TO" address.
6. The detection of octal 17's on reading the tape indicates an End-of-File. The tape will be positioned after the "EOF".

WRITE MAGNETIC TAPE

op-code	mnemonic	variable operands
17437	WRTAPE	U FROM GOTO INST

This instruction will cause one data record to be taken from core storage at the starting address "FROM" and place it in the internal buffer for tape unit "U". When the buffer is full, a tape record will be written. If end point is encountered during the writing of a tape record, the "GOTO INST" instruction will be executed. Otherwise, the program continues in sequence.

NOTE:

1. If end-point is encountered during a write operation and an alternate tape unit was specified, Bus-Pak II will automatically reference the alternate tape whenever unit "U" is referenced.
2. When end-point is encountered on the alternate tape, Bus-Pak II will automatically re-reference tape unit "U".
3. The instruction "GOTO INST" will be executed when end-point is encountered even though an alternate tape was specified so that heading records may be written on the alternate tape.
4. An internal buffer is assigned to tape unit "U" and each data record written will be accumulated in this buffer area. When the buffer is full, one tape record is written onto tape.
5. Indexing, double indexing, and indirect addressing may be used with the "FROM" address.

WRITE AND COMPARE MAGNETIC TAPE

op-code	mnemonic	variable operands
17533	WRTCMP	U FROM GOTO INST

This instruction will cause one data record to be taken from core storage at the starting address "FROM" and place it in the internal buffer for tape unit "U". When the buffer is full, a tape record will be written, backspaced, and then compared to insure that the tape record was written correctly. If an end-point is encountered during the writing of a tape record, the instruction "GOTO INST" will be executed. Otherwise, (however the record has been written successfully) the program continues in sequence.

NOTE:

1. If end-point is encountered during a write operation and an alternate tape unit was specified, Bus-Pak II will automatically reference the alternate tape unit whenever unit "U" is referenced.
2. When end-point is encountered on an alternate tape, Bus-Pak II will automatically re-reference tape unit "U".
3. The instruction "GOTO INST" will be executed when end-point is encountered even though an alternate tape was specified so that heading records may be written on the alternate tape.
4. An internal buffer is assigned to tape unit "U" and each data record written will be accumulated in this buffer area. When the buffer is full, one tape record is written. This tape record will then be backspaced and read for comparison to insure correct writing of data on tape.
5. Indexing, double indexing, and indirect addressing may be used with the "FROM" address.

SPACE RECORD

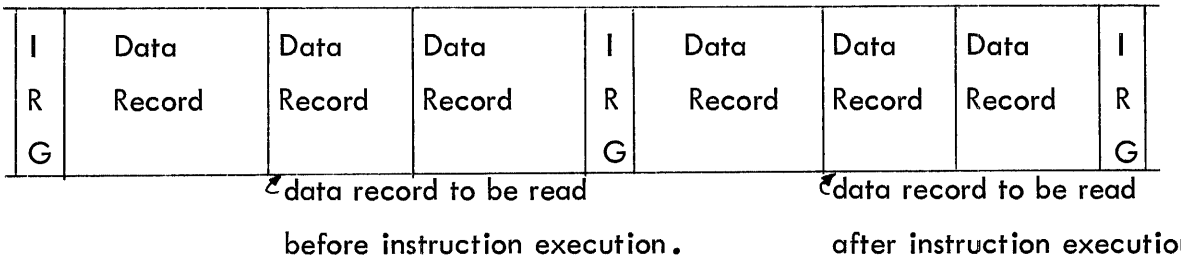
op-code	mnemonic	variable operands
17444	SPCREC	U T

This instruction spaces "T" data records on tape unit "U".

NOTE:
1. Magnetic tape may not move if the data records being spaced are contained in the internal buffer at the time the instruction is executed.
2. An end-of-file (tape mark) is counted as one data record.
3. Tape records in the new file must be in the current format if spacing over an End-of-File mark. If not, the data records after the End-of-File will be spaced incorrectly.
4. If no tape records are contained after the End-of-File mark, spacing continues indefinitely.

EXAMPLE:

```
SPCREC          U          3
```



SPACE FILE

op-code	mnemonic	variable operands
17445	SPCFILE	U

This instruction will space over one file of information on tape unit "U".

NOTE:
1. Bus-Pak II will position itself at the inter-record gap immediately following the first End-of-File encountered.
2. Spacing a file which will encounter the tape end-point is not recommended as the next operation to be performed will cause the tape to run off the reel or not executed.

BACKSPACE RECORD

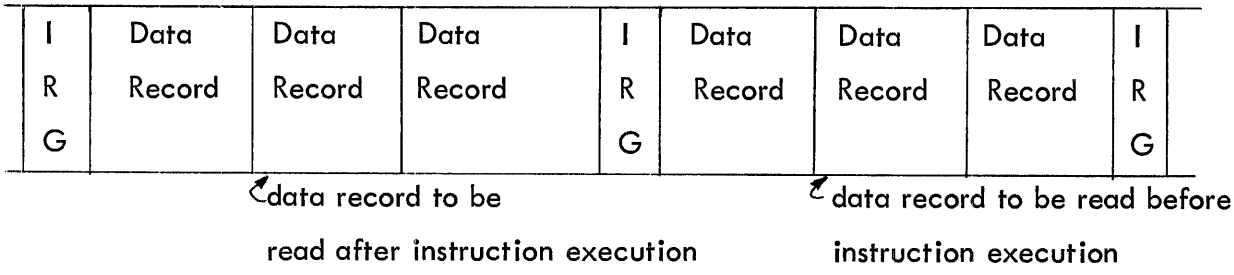
op-code	mnemonic	variable operands
17442	BKSREC	U T

This instruction backspaces "T" data records on tape unit "U".

NOTE:
1. Magnetic tape may not move if the data records backspaced are contained in the internal buffer at the time the instruction is executed.
2. An End-of-File (tape mark) is counted as one data record.
3. If an End-of-File is encountered during a backspace operation, Bus-Pak II will position itself at the next available data record location of the previous tape-record if that tape record was not completely full when written.
4. If a backspacing operation is attempted which will encounter load point, the backspacing operation will not position itself correctly.

EXAMPLE:

BKSREC U 3



BACKSPACE FILE

op-code	mnemonic	variable operands
17443	BKSFILE	U

This instruction will backspace one file of information on tape unit "U".

NOTE: 1. When an End-of-File (tape mark) is encountered, Bus-Pak II will position itself at the first data record location in the tape record preceding the End-of-File that has not been used.

2. If load point is encountered during the backspace file operation, Bus-Pak II will be positioned at the first data record on the tape.

TEST END POINT

op-code	mnemonic	variable operands
17527	ENDPOINT	U GOTO INST

On reading magnetic tape, the "GOTO INST" instruction associated with the "RDTAPE" instruction will be executed if either an End-of-File or end-point is encountered on tape unit "U".

This instruction when executed, tests to see if the "GOTO INST" instruction was executed because of an end-point on tape unit "U". If it was, the "GOTO INST" above will be executed. Otherwise, the program continues in sequence.

WRITE END-OF-FILE

op-code	mnemonic	variable operands
17441	WREOF	U

This instruction accompl ishes two things - it writes the data records (if any) still left in the internal buffer on tape unit "U" and then writes an End-of-File (tape mark).

NOTE:

1. This instruction must be executed after all data records are written or records may be lost.
2. If a record is to be written onto tape, the "WRTCMP" instruction is used.
3. If the internal buffer is not full when "WREOF" is executed, the remaining locations are filled with octal 17's.
4. The detection of octal 17's on reading the tape indicates an End-of-File. The tape will be positioned after the "EOF".

MICRO-DISK

By the use of DECtape, Bus-Pak II is able to simulate a random access disk file. This means that files of information contained on Micro-Disk may be processed randomly by input information as it is received. Pre-sorting of transactions affecting the Micro-Disk file has been eliminated.

INITIALIZE MICRO-DISK

op-code	mnemonic	variable operands
17515	OPENDISK	U L

This instruction causes the input-output controls for Micro-Disk unit "U" to be initialized to accept"L" consecutive characters as one record of information.

One DECtape reel has the capacity of 576 blocks, each of which contains 768 alphanumeric characters. Bus-Pak II automatically packs as many records ("L" characters long) as possible into one block. Each record is then assigned an address relative to its sequential location on the Micro-Disk. The first record on Micro-Disk is always assigned address 1.

When a record is requested, Bus-Pak II automatically calculates the block address in which the record may be found and also which record it is in that block. As there may be more than one record per block on Micro-Disk, Micro-Disk may not always be moved when a "SEEK" "RDDISK", or "WRDISK" instruction is executed because the block previously requested which is stored in core storage, may contain the record desired.

NOTE:

1. "L" must be defined as a multiple of three (3).
2. This instruction must be executed before any processing may be done with Micro-Disk unit "U".
3. The mark track on the DECtape reel being used must be set for 256-3 character words.
4. The following formula determines the maximum number of records that may be stored on one Micro-Disk.

$$\left(\frac{768}{L}\right) X\ 576 = \text{number of records per micro-disk reel.}$$

5. Records may only be referenced by the addresses assigned to them.
6. This instruction must never be executed a second time unless the "BEGIN" instruction also is executed.

SEEK A RECORD

op-code	mnemonic	variable operands
17516	SEEK	U

This instruction will cause Bus-Pak II to search Micro-Disk unit "U" for the record whose address was previously loaded into accumulator 15. Control is then returned to the users program so that the processing of data already available to the program may be overlapped with the Micro-Disk search operation.

If the record being searched for is in the block of information already read into core storage, Micro-Disk will not move.

NOTE:

1. The address of the record desired must have previously been loaded into accumulator 15.
2. No other input-output device can be running, during a search operation.
3. If an attempt is made to input or output data the program will hang up until the search operation is over.

READ A RECORD

op-code	mnemonic	variable operands
17517	RDDISK	U TO

This instruction will read a record, whose address is specified in accumulator 15, from Micro-Disk unit "U" and place it into the "L" (defined in the "OPENDISK" instruction for Micro-Disk unit "U") consecutive characters with the starting address "TO".

NOTE:

1. The address of the record desired must have previously been loaded into accumulator 15.
2. No processing can be done during a read Micro-Disk operation.
3. If the record desired is contained in the block previously read, Micro-Disk will not be moved.
4. If an attempt is made to input or output data the program will hang up until the search operation is over.

WRITE A RECORD

op-code	mnemonic	variable operands
1752Ø	WRDISK	U FROM

This instruction will write the "L" (defined in the "OPENDISK" instruction for Micro-Disk unit "U") consecutive characters with the starting address "FROM" as a record, whose address is specified in accumulator 15, onto Micro-Disk unit "U".

NOTE:

1. The address of the record being written must have previously been loaded into accumulator 15.
2. No processing can be done during a write Micro-Disk operation.
3. If the record being written is contained in the block previously read or written, Micro-Disk will not be moved.
4. The block of information contained in core storage will not be written onto Micro-Disk until either a "SEEK", "RDDISK", "WRDISK", or "CLOSE DISK" instruction is executed in which the record being read or written is not contained in the block in core storage.
5. If an attempt is made to input or output data the program will hang up until the search operation is over.

CLOSE MICRO-DISK

op-code	mnemonic	variable operands
17521	CLOSEDISK	U

This instruction causes the input-output controls for Micro-Disk unit "U" to check the block of information contained in core storage pertaining to Micro-Disk unit "U" to see if a "WRDISK" operation was performed. If so, that block of information is written on Micro-Disk. If not, it is disregarded.

NOTE:

1. This instruction need only be executed if a writing operation was performed in a Micro-Disk unit.
2. No processing can be done during a "CLOSEDISK" operation.
3. The buffer area assignment for Micro-Disk unit "U" is not reset.
4. If an attempt is made to input or output data the program will hang up until the search operation is over.

STORAGE AND RETRIEVAL

The following instructions were designed to extend the available storage of the computer. By making use of DECtape, data and complete subroutines may be stored on DECtape and retrieved when necessary for their execution.

In this way, more than one subroutine may occupy the same storage location. Each of the subroutines occupying the same locations would first be written on DECtape with the "DUMP" instruction; then, as each subroutine is needed for execution, it would be reloaded into those locations by the "RETRIEVE" instruction. This will help preserve core storage whenever necessary.

DUMP DATA

op-code	mnemonic	variable operands
17532	DUMP	FROM TO

This instruction writes out on DECtape Unit 8 the storage locations between address "FROM" and address "TO", inclusive.

NOTE:

1. The starting block number on which to write the data must have been previously loaded into index register 15.
2. No indexing, double indexing, or indirect addressing may be used with either the "FROM" or "TO" addresses.
3. Bus-Pak II will store 256 locations on one block on tape. Successive blocks will be used if necessary to store all the data.
4. This instruction does not require any of the Micro-Disk instructions to be executed.

RETRIEVE DATA

op-code	mnemonic	variable operands
17531	RETRIEVE	FROM TO

This instruction reads from DECtape Unit 8 and places the data read into the storage locations between address "FROM" and address "TO", inclusive.

NOTE:
1. The starting block number on which the data is contained must have been previously loaded into index register 15.
2. No indexing, double indexing, or indirect addressing may be used on either the "FROM" or "TO" addresses.
3. This instruction does not require any of the Micro-Disk instructions to be executed.

LOAD PROGRAM

op-code	mnemonic	variable operands
17536	LDPROG	

This instruction causes a transfer of control to the RIM loader which will load the program tape in the paper tape reader.

This may be used to control the loading of subroutines for storing them on DECtape.

CHECKPOINT AND RESTART

The Checkpoint and Restart feature permits a job to be restarted at an intermediate point after an interruption in processing.

The use of this feature requires that at lease one DECtape and control be part of the input output equipment available on the operating system.

The use of this feature would depend greatly on the complexity and running time of the particular job. It would reduce the time and data lost due to machine failure, power failure, and operator error. It would also give the ability to stop the running of a program, take it off the machine and resume it later.

WRITE CHECKPOINT RECORD

op-code	mnemonic	variable operands
17534	CHECKPOINT	

This instruction causes a checkpoint record to be written onto DECtape unit 8.

The contents of the checkpoint record would be as follows:

1. The users program and its status at the time the checkpoint was written.
2. The status of all program switches, program counters, double index register, and index registers.
3. The contents of all accumulators and the status of their respective overflow indicators.
4. The input/output controls for all magnetic tape and DECtape files being used.
5. The transfer location to restart the program.

NOTE:

1. Each time a checkpoint record is written, "CKP" will be typed on the on-line teleprinter. If punched cards are either being read or punched, the system will stop to permit the operator to indicate on the card decks the position of the last checkpoint. If no cards are being processed, the system will not stop.
2. The transfer location for restarting the program stored on the checkpoint record will be the location of the instruction immediately following the "CHECKPOINT" instruction being executed.
3. The checkpoint record written on unit 8 will occupy blocks 1 - 18 of the tape.

RESTART PROGRAM FROM CHECKPOINT

op-code	mnemonic	variable operands
17535	RESTART	

This instruction will cause the checkpoint record written on DECtape unit 8 to be read and the following operations performed:

1. Restore the status of the users program to what it was at the time the checkpoint record was written.
2. Restore all program switches, program counters, double index register, and index registers.
3. Restore all accumulators and their respective overflow indicators.
4. Restore the input/output controls for all magnetic and DECtape files.
5. Re-position all tape files (magnetic and DECtape) on the basis of record counts and other identifying information contained in the checkpoint record.
6. Transfer control to the appropriate instruction to resume the program.

NOTE:

1. If punched cards were being read at the time the checkpoint record was written, the cards from the time of the checkpoint must be inserted into the card reader. If the wrong card deck is placed in the card reader the restart may be in error.
2. If an error in restart occurs, the program may again be restarted providing another checkpoint was not written before the error was detected.

SUBROUTINE CONTROL

Subroutines are used to code a special sequence of instructions that would normally be repeated many times in different locations in a program. The use of subroutines reduces the over-all coding effort involved in coding a solution to a given problem and also conserves core storage.

There are two methods of defining a subroutine in the Bus-Pak II system.

METHOD NO 1

The following instructions may be used to transfer control to a given subroutine and, upon completion of that subroutine, return to the next storage location following the transfer.

TRANSFER AND SAVE RETURN ADDRESS

op-code	mnemonic	variable operands
1ØØØØØ	GOTOSV	INST

This instruction transfers the control of the program to the subroutine at location "INST". It requires that the subroutine be defined in the following manner:

INST,	Ø
(INST +1)	First instruction of subroutine

The address of the location following the "GOTOSV" instruction is saved in location "INST" and control is transferred to location "INST +1".

RETURN FROM SUBROUTINE

op-code	mnemonic	variable operands
62ØØØØ	RETFROM INST	

This instruction is used to return control of the program to the instruction following the "GOTOSV" instruction.

NOTE: 1. "INST" must be the same subroutine name used in the "GOTOSV INST" used to enter the subroutine.

METHOD NO 2

The following instructions provide a means of transferring to a subroutine with optional return locations upon completion of the execution of the subroutine.

SUBROUTINE

op-code	mnemonic	variable operands
	INST	

By placing the name of the subroutine being called in the operation portion of the instruction, a transfer will be made to that subroutine, enabling optional return locations.

NOTE:
1. No indexing, double indexing, or indirect addressing may be used in the address "INST".
2. "GOTO" instructions normally follow the name of the subroutine being transferred to. These "GOTO" instructions are placed in such a sequence so as to transfer control of the program to other routines, depending on the return location of the subroutine.

EXAMPLE:

```
TSTCHR          /transfer to routine to test character
GOTO LAB        /return here if "A" (normal return)
GOTO LAB 1      /return here if "B"
GOTO LAB 2      /return here if "C"
GOTO LAB 3      /return here if other etc
```

EXIT FROM SUBROUTINE

op-code	mnemonic	variable operands

The table below lists the instruction mnemonics used to exit from the subroutine and the location to which control is returned.

op-code	mnemonic	return location
637511	EXIT	normal return
637512	EXIT 1	normal return +1
637513	EXIT 2	normal return +2
637514	EXIT 3	normal return +3

SECTION IV

PROGRAM PREPARATION

A Bus-Pak II program is prepared in FIO-DEC code on 8 channel punched paper tape, using either a FIO-DEC flexowriter or the tape editor CANUTE on the computer. The mechanics of using a flexowriter or CANUTE for paper tape preparation are described elsewhere and only the formats which apply to Bus-Pak II will be described here.

In general, a program should begin with about two feet of tape feed, to allow easy placement in the reader. Deletes and tape feed may be used freely throughout the tape, and will always be ignored.

The program tape itself consists of three sections, described below.

TITLE

All text between the first character other than a carriage return, and the second carriage return is taken as the title of the tape. Each tape must have a title. This title will be printed on the printer or teleprinter at assembly time, as well as punched in readable format on the front of the object (binary) tape.

PROGRAM BODY

The text consisting of the program itself follows the title. Redundant carriage returns and tabs are ignored, and may be used for formatting.

A suggested format is to place address tags (labels) at the left margin, indent the instruction mnemonic to the first tab stop, and indent each instruction field and/or comments further to subsequent tab stops.

The character "stop code" may be used as a page separator (both with the tape editor, and flexowriter), although pages have no meaning to the assembler. New pages should begin with a carriage return.

Deleted characters, tape feed, and stop codes are always ignored by the assembler during processing.

Except in text strings, the characters upper case and lower case are filtered out of the input string, and are used only to inform the assembler of the case of the characters on input, therefore except for special characters or comments, all typing is normally done in lower case.

START BLOCK

The last section of a program is the start block, consisting of the pseudo-instruction start, or pause, followed by either the starting address of the program, or by a carriage return. In either case, a carriage return must appear on the tape after the start block.

The pseudo-instruction start indicates the end of the symbolic program, and causes an instruction to be punched which will be executed when the loading of the object program is completed. If start is followed by an address, then a jmp to that address will be assembled, causing execution of the program to begin when the program is loaded.

If start is not followed by an address, then an hlt instruction will be assembled, halting the loader after the object program is loaded.

Constants will be stored starting at the current address location at the time start was encountered on the last source tape.

Pause performs the same function as start, but will always cause the loader to halt before executing the instruction "indicated by pause".

SECTION V

SYMBOLIC LANGUAGE

PURPOSE

The use of symbolic languages has become a standard practice in the programming of computers. A symbolic language permits a programmer to code in a more convenient language than the language that the machine understands. A processor (Assembly Program) translates the programmer's source language to the machine language. The advantages are widely recognized. Instruction codes with high mnemonic value are used instead of numeric codes. Instructions or data may be referred to by symbolic name (label), without knowing or even caring about the actual machine address. Decimal or alphabetic data may be expressed in a more convenient form than in a binary number system. Programs may be altered without extensive changes in the source language, and debugging is considerably simplified.

DESCRIPTION OF THE ASSEMBLER SYSTEM

The Assembly program accepts a symbolic source language program on punched paper tape, translates it and produces a binary tape. (See also Assembler writeups for both the PDP-4 or PDP-7 computers).

The Assembler performs this function in one pass. That is, the source language tape is processed only once to produce the object language tape.

The object language tape consists of the binary version of the source program and a binary loader.

NOTATIONS

Special Characters

Character		Meaning
→|	(tab)	field delimiter
↵	(carriage return)	field delimiter
b	(space)	punctuation character
+	(space)	punctuation character
-	(minus)	punctuation character
,	(comma)	label delimiter

Syllables

Number

Any sequence of digits delimited by field delimiters or punctuation characters.

Examples

< ↵ 123→| >
<→| 567 ↵ >
< ↵ 56+ >

Labels

Any sequence of alphanumeric characters delimited by a comma and field delimiters or punctuation characters. The first character must be an alphabetic character, the label may be any length, but all characters after six are ignored.

Examples

< ↵ A,→| >
<→| Harry, ↵ >
< ↵ AB12,→| >

Expressions

An expression is a sequence of labels and/or numbers connected by punctuation characters and delimited by field delimiters.

Examples

<→ A , >

<→ A + 3→ >

< , ABLE + 21 , >

PSEUDO INSTRUCTIONS

variables

A label used in the program, with one of its characters overbarred (the overbar needs to be indicated only once in any of its appearances) and not explicitly defined elsewhere, is a variable. Variables will be assigned a register location following the occurrence of the pseudo-instruction "variable". In addition, should any variable remain undefined at the termination of the assembly, these will be automatically defined following the program.

Example:

```
< ↓ WHAT →↓ >
<→↓ A123 ↓  >
```

text

TEXT b Z ABC.....CZ

The pseudo-instruction "text" indicates the characters between the first occurrence of the character Z and the next occurrence of the character Z is to be converted to character codes and stored three to a word in successive registers. The character Z may be any single character desired. If the text information is to be typed on the on-line teletype by the TYPTXT instruction "TELETYPE" must precede the "TEXT" pseudo instruction. After the text has been written, ANELEX must follow.

The "TEXT" pseudo instruction should always be used to define constant information and the program should move and expand this information in the initialization of the program.

ASSEMBLER FEATURES (See also Assembler Program Writeup)

Current Address Indicator

The character period (.) has the value of the current address when standing alone between field delimiters and/or punctuation characters, otherwise it is regarded as an alphabetic character.

Address Assignment

The expression preceding a slash (/) will be taken as a new current address. If no expression precedes the slash, then the slash initiates a comment statement. This may be used to assign consecutive core locations to a given label.

Example:

```
< ABLE,          ABLE +10/>
This assigns 1Ø consecutive locations
to the label ABLE.
```

Parameter Assignments

A parameter may be defined by use of the equal (=) sign. The label to the left of the equal sign will be assigned the value of the expression to the right of the equal sign. If the expression to the right of the equal sign is not terminated by a tab or carriage return, the error message, IFP (Illegal Format in Parameter Assignment) will occur. An undefined label appearing in the expression to the right of the equal sign will cause the error print UPA (Undefined Parameter Assignment).

Example:

```
AREA,       AREA +80/
TAX = AREA + 15
Whenever TAX is referenced, the address
AREA + 15 will be used.
```

SECTION VI

ASSEMBLING A BUS-PAK II PROGRAM

The following steps are those to follow to assemble a Bus-Pak II Program on the computer. Each statement is followed by the number of the next step to be executed. Normally, the sequence will continue to the next step.

I. LOADING THE ASSEMBLER

1. Obtain the binary tape of the Assembler from the files.
2. Place the binary tape to be loaded into the paper tape reader.
3. Set the address switches on the console to 17770 and depress the "start" switch. The read-in mode loader (see Appendix I) must be in core at this time.
4. When the loading of the binary tape is completed remove the tape from the paper tape reader.
5. Replace the Assembler tape into the file.
6. Go to Step II.

II. LOADING A SYMBOL PUNCH DEFINITION TAPE

1. Obtain the Bus-Pak II Symbol Punch definition tape from the files.
2. Place the symbol punch definition tape into the paper tape reader.
3. Set the address switches on the console to 4 and depress the "start" switch.
4. When the loading of the Symbol Punch definition tape is completed, remove the tape from the paper tape reader.
5. Replace the Bus-Pak II Symbol Punch definition tape in the files.
6. If other Symbol Punch definition tapes are to be loaded, execute Steps 2, 3 and 4 above for each Symbol Punch tape.
7. Go to Step III.

III. ASSEMBLING A SYMBOLIC LANGUAGE TAPE

1. Place the Symbolic source language tape of the program to be assembled into the paper tape reader. Make sure paper tape punch is loaded as well as printer if symbols are to be printed.

2. Set the address switches on the console to 2Ø.

3. At this point the operator may choose to command the assembler by use of the AC switches (see IV) or begin a normal assembly.

3A To begin a normal assembly, depress the "Continue" switch.

3B To give the assembler an AC switch command, set the AC switches desired and depress the "start" switch.

4. When the pseudo-instruction START is encountered at the end of the source language tape, the assembler will stop with -Ø (777777) in the AC.

4A If more tapes are to be processed at this time, place each tape in the paper tape reader and depress "start".

4B To complete the assembly of the program depress the "continue" switch. The assembler will punch the variable definitions, punch the undefined symbol definitions listing these symbols on the on-line teletype or line printer, punch the starting block, the loader, and the title in readable form.

5. When the assembly is complete, the assembler will stop with -Ø (777777) in the AC.

5A To print symbol definitions, set the AC switches and depress the "continue" switch. The AC switches have the following meanings:

AC Switch	Meaning
11	Print on line printer
15	Restore permanent Symbol table
16	Numeric Print
17	Alphabetic Print

5B To assemble another program, saving the present symbol table, put the new tape in the paper tape reader and depress the "start" switch and go to III-4.

6. When the symbol print is complete the assembler will stop with +Ø (ØØØØØØ) in the AC.

6A To restore the symbol table and start a new assembly, go to III-1.

6B To start a new assembly saving the present symbol table, place the new symbolic language tape into the paper tape reader, depress the "start" switch and go to III-4.

IV. AC SWITCH CONTROL

Whenever the "start" switch is depressed with 2Ø in the address switches, the AC switches will be examined. If bit Ø is a zero, then the AC switches are ignored. However, if bit Ø is a one, the remaining AC switches have the following meanings.

Bit a "One"	Meaning
1	Suppress punching
2	Punch symbols for DDT-4
3	Take this title
4	Restore the assembler
5	Take first address from AC switches 6-17

V. ASSEMBLER OUTPUT

The assembler punches the object tape as it reads the symbolic source language tape. During assembly, error messages may be typed on-line. (See Error Messages)

VI. STOPS DURING ASSEMBLY

The following is a list of all possible stops during Assembly, the cause, and the action which may be taken.

AC	Cause
-0	start or pause encountered
-0	assembly complete
+0	symbol print request satisfied
character	illegal parity
status register	offensive interrupt

When a device other than the reader, punch or teletype causes a program interrupt, the Assembler will halt with the status register displayed in the AC. "Continue" will clear some standard device flags not including those of the devices used by the Assembler, and proceed.

If this fails to clear the offending device's flag, the other action must be taken. When the device is disabled, pressing "continue" will permit Assembly to continue correctly.

VII. ERROR MESSAGES

A list of the error messages may be found below. With the exception of sce (storage capacity exceeded) and ilp (illegal parity), assembly continues automatically after the error message has been printed. An error message may occur in one of three formats.

Format A

The appearance of a diagnostic printed in format A:

ERROR PREVIOUS VALUE SYMBOL NEW VALUE

Whether the new value was actually incorporated into the symbol table depends upon the particular error.

ERR	Meaning
mdt	the symbol was redefined with a comma
rps	a permanent symbol was redefined
rda	an attempt to redefine a symbol was made. The symbol was not redefined.

Format B

The appearance of a format B diagnostic is:

ERROR OCTAL ADDRESS SYMBOLIC ADDRESS

The general error message is printed in Format B.

ERR	Meaning
ifp	illegal format in parameter assignment
ifc	illegal format in a comma assignment
mdt	the value and address disagree in an address assignment
tua	too many undefined symbols in an address assignment
lit	illegal terminator in a pundef list
ifl	illegal format in a pundef list
ifs	illegal format in a start (or pause)
ifi	illegal format in an input pseudo-instruction
sce	storage capacity exceeded

Format C

The appearance of a format C diagnostic is:

ERROR OCTAL ADDRESS SYMBOLIC ADDRESS CAUSE

Format C is an expanded version of Format B. CAUSE is additional information to help the programmer ascertain the cause of the error. For example, in the case of an error caused by an undefined symbol, the symbol will be printed.

ERR	CAUSE	MEANING
ilp	character	illegal parity (place correct character in ACS and 'continue').
ust	symbol	undefined symbol in a start or pause
uaa	symbol	undefined symbol in an absolute address assignment
upa	symbol	undefined symbol in a parameter assignment
ich	character	illegal character
lus	symbol	undefined symbol in a pundef list
ubr	symbol	undefined symbol in a bar pseudo-instruction

Undefined Symbol Assignments

At the end of assembly, before the loader is punched, any undefined symbols will be automatically defined. Each undefined symbol which was used in a storage word will be defined as the address of a register at the end of the program, and the definition printed. If the symbol was not used in a storage word, then the symbol will be printed but not defined.

VIII. LOADING THE OBJECT PROGRAM

1. Obtain the object tape of the Bus-Pak II program from the file.
2. Place the object tape into the paper tape reader.
3. Set the address switches on the console to 17770 and depress the "start" switch. The read-in mode loader (see Appendix I) must be in core at this time.
4. When loading is completed, the loader will stop.
5. Remove the object tape from the paper tape reader and replace it into the files.
6. Place the object tape of your program into the paper tape reader.
7. Execute 3 above.
8. When loading is completed, the machine is under control of your program.

SECTION VII

EQUIPMENT OPERATING FEATURES

CARD READER

Keys and Lights

Power ON	Depressing this key supplies power to the card reader and lights the power ON key.
Power OFF	Depressing this key turns off the power supply to the card reader.
Not Ready	This light is lit whenever the card reader is in a not ready condition.
Read Check	This light is lit when a read check occurs.
Feed Check	This light is lit when a feed check occurs.
Validity Check	This light is lit when a validity error has occurred and the Validity ON key is lit.
Validity ON	Depressing this key turns on the Validity checking feature for alphanumeric input and lights this key.
Reset	This key is depressed to reset a feed check, read check, or validity check condition.
End of File	When all the cards have been read, depressing this key informs the computer of an End-of-File condition on the card. reader and lights.
Start	Depressing this key, when all conditions are right, makes the card reader ready to read cards.
Stop	Depressing this key puts the card reader in a not ready condition.

Loading the Card Reader

1. Depress the power ON key.
2. Place the cards to be read into the card read hopper face down - 12 edge toward the operator.
3. Depress the Validity ON key to the ON condition (when on, it is lit).
4. If either the Read check, Feed check, Validity check, or End-of-File indicators are lit, depress the reset key.
5. Depress the Start key.

Restart Procedures

Not Ready Condition Only.

1. Remove the cards from the card read stacker.
2. Be sure to place these cards in back of previously removed cards.
3. If there are more cards to read go to Step 5 below.
4. If no more cards are to be read, depress the End-of-File Key. This will signal the computer of this fact. The End-of-File key is lit. Proceed no more.
5. Place the cards to be read into the card read hopper, face down - 12 edge toward the operator.
6. Depress the start key.

Feed Check

1. Remove the cards in the card read hopper.
2. Repunch the first few cards that caused the feed check.
3. Replace these cards in front of the cards removed from the read hopper.
4. Place these cards into the card feed hopper; face down - 12 edge toward the operator.
5. Set AC switch 1 in the up position.
6. Depress the Reset Key.
7. Depress the Start Key.

Read Check

1. Remove the cards from the card feed hopper.
2. Remove the top card in the card read stacker.
3. Check card and re-punch if necessary.
4. Place this card in front of the cards removed from the card read hopper.
5. Place these cards into the card feed hopper, face down - 12 edge toward the operator.
6. Set AC switch 1 in the up position.
7. Depress the Reset Key.
8. Depress the Start Key.

Validity Check

Do the same operations as for the Read Check above.

CARD PUNCH

Keys and Lights

Power ON	Placing this switch in the ON position supplies power to the punch.
Start	Depressing this key initiates a 2-card read-in cycle.
Stop	Depressing this key will cause the card mechanism to be shut off after a delay of approximatly 3 seconds.
Continue	Depressing this key turns the card mechanism ON but does not initiate a 2-card read-in cycle.

Loading the Punch Hopper

1. Set the Power switch to the ON position.
2. Depress the Start key to insure that no cards were left in the punch mechanism from a previous run.
3. Insert blank cards, face down - 12 edge first into the punch feed hopper.
4. Depress the Start key.
5. When punch idles, release key. It is ready to receive information from the computer.

Servicing the Punch

1. Remove punched cards from the card punch stacker.
2. Be sure to place these cards in back of previously removed cards.
3. Insert blank cards, face down - 12 edge first into the punch feed hopper.
4. Depress the Continue key.

Emptying the Punch

1. Remove blank cards from the punch feed hopper.
2. Depress the Start key to allow the 2 cards in the card mechanism to be fed into the punch stacker.
3. Remove punched cards from the card punch stacker.
4. Be sure to place these cards in back of previously removed cards.
5. Place the power ON switch in the OFF position.

HIGH SPEED PRINTER

Operating Controls

Four operating controls, mounted on the top and front of the printer, are provided as follows:

Paper Tension knob - controls the distance between corresponding sprockets on the upper and lower tractors to increase or decrease the tension on the paper.

Form Positioning knob - provides a means of adjusting the paper tractor sprock up and down to vary the vertical position of the printed line on the paper.

Character Phasing knob - controls the time interval between the occurrence o the character pulse and the appearance of the corresponding row of character directly opposite the print hammers. When this knob is properly positioned, the hammers strike directly over the characters on the print wheel, and unifo printing results.

Penetration Control knob - provides fine adjustment of the spacing between print hammers and print wheel to vary the density of printing and accommodate different paper thicknesses.

Loading the Paper

Thread the paper over the lower paper feed tractors, under the paper hold-dov and ribbon, and over the upper paper feed tractors. Use the penetration control cra to lower the print hammer module assembly if necessary. Return the hammer module sembly to printing position. Use the scales and individual adjustment screws provide accomplish proper positioning of the paper feed tractors. The paper should press fir against the platen but should not be taut enough to cause elongation of the perforati

Replacing Ribbon

To replace the ribbon, proceed as follows:

1. Remove ribbon cover.
2. See that ribbon is feeding onto inner roll. Switch power OFF as soon as direction of ribbon travel reverses.
3. Slacken ribbon by manually turning outer roll several forward revolutions.
4. Grasp outer roll and push toward right side of printer. The left end of the roll will disengage its drive cap, permitting removal of outer roll.
5. Unwind remaining ribbon from outer roll.
6. Remove inner roll by pushing to the right until left end of roll disengages its drive cap. Pull inner roll toward front of printer, drawing free end of ribbon through print aperture. Discard worn ribbon and roll.
7. Place empty roll in the inner position. Be sure slot on left end of roll engages drive cap pin.
8. Insert two sheets of paper through the print aperture. Attach the leader of a fresh roll of ribbon between these two sheets. Draw paper and ribbon through the aperture toward front of printer.
9. Detach ribbon leader from paper. Pull ribbon leader under inner roll and fasten it to the roll. Wind a few turns onto the roll.
10. Place new roll of ribbon in outer roll position, taking up slack by rotating roll several revolutions. Be sure slot in left end of roll engages drive cap pin.
11. Turn on power. Observe that ribbon moves at a steady rate and winds evenly.

Control of Vertical Format

a. Preparing Format Tape - the following procedure is suggested for the preparation of a printer format tape. Since formats differ from application to application, this procedure is intended to serve only as a guide for those lacking previous experience in tape preparation.

The following special tools and materials are recommended for the preparation of a format tape:

1. Roll of Anelex (or equivalent) format tape - laminated
2. Tape punch. Anelex No 52026
3. Pliobond cement (or equivalent)

b. Procedure -

1. Determine the total number of lines contained on the entire form. This is done by multiplying the number of inches (length) of the form by the number of lines of paper feed per inch. Most Anelex printers have vertical spacing of 6 lines per inch. An 11-inch form will have 66 lines (6 x 11); a 17-inch form 102 lines, etc.

2. Take a sample form and rule in all the lines. Number each line.

3. Cut off a strip of format tape containing three more sprocket holes than the total number of lines on the form (always cut at the midpoint between the holes). Each sprocket hole on the tape corresponds to a line on the form, so a tape for a 66 line form should have 69 sprocket holes. The extra holes will be used for splicing purposes. In cases of small forms, such as a 3-inch, 18-line form, it is a common practice to format the form a multiple number of times onto a single tape. For example, an 18-line form may be formatted three times on a tape containing 57 sprocket holes. In any case, the tape must be long enough to loop around the format drum loosely.

4. Hold the tape in a vertical position with the column of sprocket holes appearing towards the right. Visualize the location of the eight possible channels which may be punched onto the tape.

5. Determine the format(s) desired and indicate on the sample form by placing a mark on every line where printing is desired. The first line of print is generally considered as the top of form.

6. Insert the tape into the puncher, aligning the first hole to be punched beneath the punch guide holes

NOTE

When inserting the tape into the puncher, you will note that the sprocket pins are spaced to engage every 8th sprocket hole on the tape, and that the eight punch guide holes are exactly eight sprocket hole spaces from the nearest engaged sprocket hole (counting the engaged hole also). Knowing this, the alignment of the punching position of the tape beneath the punch guide holes is readily obtained. If, for example, top of form (first line of printing) is to appear on the third line of the form, count down to the 11th sprocket hole on the tape (3 + 8) and engage that hole with the nearest sprocket pin to the punch guide holes. The third hole in the tape, which as noted previously corresponds to the third line of the form, will fall exactly beneath the punch guide holes.

7. Punch the format hole(s) into the tape by pushing the punching pin through the desired guide hole.

8. Carefully lift the tape off the sprocket pins and advance it through the puncher until the next punching position is reached. Small grease pencil markings made at strategic places on the tape, may help prevent losing hole count during tape advancement. Markings should be wiped

off after the entire tape is punched.

9. Repeat subparagraphs (7) and (8) until all channels have been punched. If format holes have been punched adjacent to the first sprocket hole, they should be duplicated adjacent to the last or 67th sprocket hole.

10. Carefully remove the tape from the puncher and turn the end of the tape containing the 67th hole in so that the bottom side of the tape becomes the top side. Channel No 1 should still appear on the right side of the sprocket hole column. With the tape reversed in this manner, engage the 67th sprocket hole with the center sprocket pin of the puncher. Also engage the first sprocket pin with a corresponding hole. This will keep the tape even and aligned for the next step.

11. Apply a thin coat of pliobond cement (or equivalent) across the width of the tape adjacent to the 67th sprocket hole. Application may be made with a toothpick.

12. Holding the middle section of the tape firm against the flat portion of the puncher, bring the free end of the tape with the first sprocket hole over and place it above the cemented end. Engage the first sprocket hole with the same sprocket pin protruding through the 67th hole. Align any holes which may have been punched adjacent to the first and 67th sprocket holes so that they complement each other perfectly.

13. Apply pressure to the union until it is held firm.

14. Clean excess cement from tape punch.

MAGNETIC TAPE TRANSPORT

Manual Control Panel

This panel includes four switches with three associated indicators and eight independent lights that indicate various tape unit conditions when lit.

Transport Power

This switch has two stable positions, OFF and REMOTE, and a momentary contact position, ON. When OFF, power cannot be applied to the transport. Pushing the switch to the momentary ON position turns on the transport; it then stays on even if the switch is returned to REMOTE. However, under no circumstances can power be applied to the transport (by either the TRANSPORT POWER switch or the 822 power control) unless the transport interlock is closed.

The indicator above TRANSPORT POWER does not indicate a switch position, but instead lights whenever power is actually applied to the transport.

Mode

Two-position switch with associated AUTO and MAN indicators. In AUTO the tape unit is on line and all operations are initiated from the tape control. In MAN the unit is off line; reading and writing are disabled and transport motion signals are generated from the rewind and direction switches on the control panel.

Unit

Eight-position thumbwheel which determines the address of the tape unit. For example, if the tape control addresses unit 5, only that unit whose UNIT thumbwheel is set to 5 responds. If desired, the operator can cause the tape control to write on two tapes simultaneously by giving both units the same address.

Manual Rewind

Three-position momentary contact switch with a stable center off position. Pushing this switch to START RWD sets the rewind flip-flop, causing the tape to run in reverse at high speed. Pushing the switch to STOP RWD clears the rewind flip-flop, halting the tape.

Manual Direction

Three-position momentary contact switch with a center off position. Forward tape motion is produced while this switch is held in FORWARD; reverse motion while held in REVERSE.

Selected

With the Type 52 Control, this light indicates that the tape unit has been selected. However, with the Type 51 Control, the light indicates that the tape unit has been selected and commanded to operate.

Ready

Indicates that the unit is ready for on-line operation. This requires that the unit be in AUTO mode, that transport power be on and that the tape be stationary.

Write Lock

Indicates that the supply reel does not contain a write enable ring. Leaving the ring out of the reel protects the information contained on the tape.

Rewind

Indicates that the tape is rewinding.

Load Point

Indicates that the beginning-of-tape reflective strip is at the photosensor.

Full Reel

Indicates that less than 100 feet of tape are on the takeup reel.

Low Reel

Indicates that less than 100 feet are left on the supply reel.

End Point

Indicates that the end-of-tape reflective strip is at or has passed the photosensor.

Tape Loading

1. Turn TRANSPORT POWER switch to OFF.
2. Take the unit off line by turning the mode switch to MAN.
3. Rotate tape load handle 180° clockwise. This brings the tension arms inside the bridge rollers.
4. Lock low tape sensors by pushing them against the stop blocks.
5. Mount an empty reel on the lower hub.
6. Mount a full reel of tape on the upper hub so that the free end of the tape hangs down from the right side of the reel, shiny side out. The groove for the write enable lockout ring should be on the back of the reel. If the tape contains data that must be protected, make certain there is no write enable ring in the reel.
7. Unwind about 6 feet of tape from the reel.
8. Open buffer cover and head cover.
9. Thread tape.
10. Close buffer cover but not head cover.
11. Wind one turn of tape around the takeup reel in a clockwise direction.

CAUTION

Do not slip free end of tape into slot in reel core and do not secure the free end to the reel in any manner.

12. Hold free end of tape to core of takeup reel with finger.

13. Turn supply reel until tape slack is taken up.

14. Wind about four turns of tape on takeup reel by rotating both reels manually. Make sure the tape is not slipping on the takeup reel.

15. Unwind about 2 feet of tape from each reel to provide slack for the tension arms.

16. Release tape load handle by rotating 180° counterclockwise. The tension arms swing toward normal position, taking up tape slack.

17. Inspect to see that tape is properly positioned within all guiding rollers and the guide trough.

18. Close head cover.

19. Release low tape sensors.

20. Make sure the tape is seated properly by running the transport from the manual control panel.

Manual Control

After loading a tape, the operator should check out the tape unit from the manual control panel to make sure that the tape is seated correctly and that the transport is running properly. Apply power to the transport by turning ON the TRANSPORT POWER switch. It is not necessary to put the 822 power control into LOCAL. It can be left in REMOTE provided the tape unit is off line (i.e. the mode switch is in MAN). However, transport power cannot go on unless the transport interlock is closed.

The interlock opens whenever the tape load handle is turned to the load position bringing the tension arms inside the bridge rollers. It also opens whenever the tension arms swing all the way out to the tension arm bumpers. The former condition prevents the transport from being turned on while tape is being loaded; the latter turns off the transport if the tape should break or run off the reel.

With power on and the unit off line, check the effect of the direction and rewind switches at the right of the control panel. These switches function only when the unit is off line. Check the forward motion of the tape by holding the direction switch in FORWARD. Wind enough tape on the takeup reel so that both reverse and rewind can be checked. Run the tape in reverse for a short distance. Then push the rewind switch to START RWD and let the rewind go to the beginning of the tape. The tape should halt when the LOAD POINT indicator goes on.

Operator's Check List

After loading a tape and checking tape motion, the unit should be readied for on-line operation. Check the following:

1. If computer power is off, turn TRANSPORT POWER to REMOTE.
2. If computer power is on, is the transport also on? If not, push TRANSPORT POWER to ON, then return it to REMOTE. The indicator at the upper left should light.

3. Is the tape unit on-line? The AUTO indicator should be lit.

4. Is the READY indicator lit? The LOAD POINT and FULL REEL indicators should also be on.

5. If the tape contains data that must not be destroyed, make sure the write enable ring is not in the reel. The WRITE LOCK indicator should be lit.

6. Is the UNIT selector set to the correct address? If the computer is to read the tape, make sure no other unit is set to the same address.

APPENDICES

APPENDIX I

CHARACTER CODE CHART

COLLATING SEQUENCE

BCD	ANSI	TELE	CARD	DEFINED AS	CODE
				Space	010000
A	A	A	12-1		110001
B	B	B	12-2		110010
C	C	C	12-3		110011
D	D	D	12-4		110100
E	E	E	12-5		110101
F	F	F	12-6		110110
G	G	G	12-7		110111
H	H	H	12-8		111000
I	I	I	12-9		111001
&	?		12-Ø	Plus Zero	111010
.	.	.	12-3-8	Period	111011
)	)	)	12-4-8	Right Paren.	111100
[	[	$(	12-5-8	Left Bracket	111101
<	<	$-	12-6-8	Less Than	111110
→	→	$;	12-7-8	Arrow	111111
-	-	-	11	Minus	100000
J	J	J	11-1		100001
K	K	K	11-2		100010
L	L	L	11-3		100011
M	M	M	11-4		100100
N	N	N	11-5		100101
O	O	O	11-6		100110
P	P	P	11-7		100111
Q	Q	Q	11-8		101000
R	R	R	11-9		101001
!	↑	!	11-0	Minus Zero	101010
$	$	$S	11-5-8	Dollar Sign	101011
*	x	#	11-4-8	Asterisk	101100
]	]	$)	11-5-8	Right Bracket	101101
;	|	;	11-6-8	Semicolon	101110
Δ	"	"	11-7-8	Delta	101111

BCD	ANSI	TELE	CARD	DEFINED AS	CODE
+	+	&	12	Plus	110000
/	/	/	0-1	Slash	010001
S	S	S	0-2		010010
T	T	T	0-3		010011
U	U	U	0-4		010100
V	V	V	0-5		010101
W	W	W	0-6		010110
X	X	X	0-7		010111
Y	Y	Y	0-8		011000
Z	Z	Z	0-9		011001
~	~	$?	0-2-8	Negation	011010
,	,	,	0-3-8	Comma	011011
(	(	(	0-4-8	Left Paren.	011100
∨	∨	$/	0-5-8	Or	011101
∧	∧	$#	0-6-8	And	011110
⊃	⊃	$,	0-7-8	Implies	011111
Ø	Ø	Ø	0		001010
1	1	1	1		000001
2	2	2	2		000010
3	3	3	3		000011
4	4	4	4		000100
5	5	5	5		000101
6	6	6	6		000110
7	7	7	7		000111
8	8	8	8		001000
9	9	9	9		001001
				N/A	000000
=	=	$:	3-8	Equal	001011
@	'	'	4-8	At	001100
:	-	:	5-8	Colon	001101
>	>	$&	6-8	Greater than	001110
			7-8	EOF	001111

APPENDIX II

BUS-PAK II INSTRUCTION LIST

APPENDIX III

BUS PAK II ERROR MESSAGES

MESSAGE	CAUSE	ACTION
BOC	Bad op-code	Cannot be restarted
CKP	Checkpoint record has been written	Indicate location of checkpoint on cards (input or output) and press continue
CMP	Compare error on Magnetic tape	Does not stop
NSR	No such record on disk	Press continue to bypass error record and continue processing
PAR	Parity during read of Magnetic Tape	Does not stop
PAW	Parity during write of Magnetic tape	Does not stop
PNR	Card punch not ready	Fix error condition, set switch 2 up if card is to be repunched, press continue
PRC	Parity during read-compare on Magnetic tape	Does not stop
RIC	Read incorrect card in card reader	Fix card and replace in card hopper, set AC switch 1 in the up position and press continue.
RWD	Error in reading or writing of Micro-Disk	Check unit and fix, press continue to try again
SRR	Short record read on Magnetic tape	Press continue to back space and try again.
SRW	Short record written on Magnetic tape	Press continue to back space and try again
TEP	Tape at end-point, no end of file written	Does not stop

APPENDIX III Cont'd

BUS-PAK II ERROR MESSAGES

MESSAGE	CAUSE	ACTION
TME	Too many EXIT's	Cannot be restarted
TNI	Too many tape units defined	Cannot be restarted
TP	Tape unit is write protected	Correct and press continue
TUD	Either tape unit or Micro-Disk unit is not defined	Cannot be restarted

APPENDIX IV

FORBIDDEN LABELS

ADD	D	ION	MMEF	OPENDI	SKP
ADDAC	DAC	IORS	MMLC	OPR	SKR
ADDIDX	DCF	IOT	MMRD	PAUSE	SMA
ADDMEM	DECIMA	ISZ	MMRS	PCF	SML
AND	DEPAC	JMP	MMSE	PLS	SNA
ANELEX	DEPIDX	JMS	MMWR	PRNLIN	SNL
BAR	DIVAC	KRB	MNC	PSF	SPA
BEGIN	DIVMEM	KSF	MRC	PUNCH	SPACE
BKSFIL	D3B	LAC	MRCA	PUNCRD	SPCFIL
BKSPC	DSF	LAM	MRD	PUNDEF	SPCREC
BKSREC	DUMP	LAS	MRL	RAL	SPL
BSR	DXL	LAT	MRM	RAR	START
CAL	DXS	LAW	MRR	RCBH	STL
CCL	DYL	LDBLID	MRS	RCBL	STOP
CHAR	DYS	LDCTR	MSC	RCDH	SUBAC
CHECKP	DZM	LDIDEC	MSCR	RCDL	SUBMEM
CLA	ENDPOI	LDIDX	MSEF	RCL	SYMBOL
CLC	EQUAL	LESEQU	MSF	RCR	SZA
CLEAR	EXIT	LESS	MSI	RDCRD	SZL
CLL	EXITI	LOADAC	MSUR	RDTAPE	TAB
CLOF	EXIT2	LODEN	MSWF	RESTAR	TAD
CLON	EXIT3	LPCF	MTC	RETFRO	TAO
CLOSED	EXPUNG	LPLD	MTRS	RETRIE	TCE
CLRAC	FIODEC	LPSE	MTS	REW	TCF
CLRSTR	FIX	L7SF	MULAC	REWIND	TCR
CLSF	FLEX	LSCF	MULMEN	RRB	TELETY
CLZ	GCL	3S3S	MV	RSA	TEXT
CMA	GLF	LSSF	MVE	RSB	TLS
CML	GLK	4CA	MVN	RSF	TPS
CMPEQU	GOTO	MCC	MVS	RTBH	TSF
CMPGEQ	GOTOSV	MCD	MVX	RTBL	TSS
CMPGRT	GPL	MCEF	MVZ	RTDH	TSTCTR
CMPLEQ	GPR	MCI	MWC	RTDL	TYPE
CMPLES	GREATE	MCWF	MWL	RTR	TYPEIN
CMPUEQ	GRTEQU	MDEF	MWM	SAD	TYPTXT
CPCF	GSF	MDWF	MWR	SEARCH	UNEQUA
CPLR	GSP	MEEF	NOINPU	SET	VARIAB
CPSE	HIDEN	MEWF	NOP	SETX	WEF
CPSF	HLT	MIEF	NOSYMB	SETY	WREOF
CRRB	I	MIWF	NOT	SHFTL	WRTAPE
CRSA	IFX	MLI	OAS	SHFTR	WRTCMP
CRSB	IFY	MMBF	OCTAL	SHFTRR	WTBH
CRSF	IOF	MMDF	OPEN	SHIFT	WTBL

APPENDIX IV Cont'd.

FORBIDDEN LABELS

WTDH	XX	XII	XI4	X3	X6
WTDL	XI	XI2	XI5	X4	X7
XCT	XI	XI3	X2	X5	X8
XOR					

BUS-PAK II CODING SHEET

Sheet___of___

Program Name_____ No._____ Author_____ Date__/__/__

Label	Operation	Variable Operand Field	Comments

APPENDIX V

APPENDIX VI

PROGRAMMING EXAMPLES

Example 1:

```
card-to-tape
        anelex                                          / set anelex code input
        decimal                                         / set decimal radix
strt,   begin                                           / initialize Bus-Pak II
        open        1       10      81                  / open unit 1 w/10 x 81 rec
        clrstr      81      cda                         / clear card area
        stop                                            / stop machine
        rewind      1                                   / rewind unit 1
rdcd,   rdcrd       cda     goto ceof                   / get card info cda
        wrtcmp      1       cda     goto tep            / write card from cda
        goto rdcd                                       / go back and read next card
ceof,   wreof       1                                   / write end-of-file on unit 1
        rewind      1                                   / rewind unit 1
        goto strt+8                                     / go stop machine
tep,    typtxt      msl                                 / type end-point message
        stop                                            / stop machine
        goto rdcd-2                                     / go rewind new tape
cda,    cda+81/                                         / define card area
        teletype                                        / set teletype code input
msl,    text /
tape is at end point
load next tape, press continue
/
start strt
```

Example 2:

```
tape-to-card
        anelex                                          / set anelex code input
        decimal                                         /set decimal radix
strt,   begin                                           / initialize Bus-Pak II
        open        1       1Ø      81                  / open unit 1 w/1Ø x 81 rec
        stop                                            / stop machine
        rewind      1                                   / rewind unit 1
rdt,    rdtape      1       cda     goto teof           / read record from tape
        puncrd      cda                                 / punch card
        goto rdt                                        / go read next record
teof,   endpoint    goto tep                            / test if eof or end-point
        rewind      1                                   / eof - rewind unit 1
        goto strt + 5                                   / go stop machine
tep,    typtxt      msl                                 / type end- point message
        stop                                            / stop machine
        goto rdt-2                                      / go rewind new tape

cda,    cda + 81/                                       / define card area

        teletype                                        / set teletype code in put
msl,    text /
tape at end-point.
load next tape, press continue.
/

start strt
```

Example 3:

```
tape-to-printer
        anelex                                          / set anelex code input
        decimal                                         / set decimal radix
strt,   begin                                           / initialize Bus-Pak II
        open        1       1Ø      81                  / open unit 1 w/ 1Ø x 81 rec
        clrstr      12Ø     pta                         / clear print area
        stop                                            / stop machine
        rewind      1                                   / rewind unit 1
rdt,    rdtape      1       pta     goto teof           / read rec from unit 1
        prnlin      pta                                 / print record
        goto rdt                                        / go read next record
teof,   endpoint    goto tep                            / test if end-point or eof
        rewind      1                                   / eof - rewind unit 1
        goto strt + 8                                   / go stop machine
tep,    typtxt      msi                                 / type end-point message
        stop                                            / stop machine
        goto rdt - 2                                    / go rewind new tape

pta,    pta + 12Ø /                                     / set up print area

        teletype                                        / set teletype code input
msi,    text /
tape at end-point.
load next tape, press continue.
/

start strt
```

Example 4:

card-to-tape, tape-to-card, tap-to-printer

```
        anelex
        decimal
strt,   begin                                   / initialize Bus-Pak II
        clear       Ø                           / clear all prog sw
tstssw, stop                                    / stop machine
        tss         1       goto ctt            / card-to-tape op
        tss         2       goto ttc            / tape-to-card op
        tss         3       goto ttp            / tape-to-printer op
        goto tstssw                             / test sw's again
/ set sense switch 1 up for cart-to-tape operation
ctt,    open        1       1Ø      8l          / card-to-tape open
        set         1                           / set prog sw 1
loop1,  rdcrd       cda     goto deof           / rd card
        wrtcmp      1       cda     goto tieof  / write card
        tss         1       goto .+3            / test ssw 1
        clear       1                           / no-clear prog sw 1
        tps         2       goto loop2          / test for tape-to- card
        tss         2       goto ttc            / test start tape-to-card
        tps         3       goto loop 3         / test for tape-to-printer
        tss         3       goto ttp            / test start tape-to-printer
        tps         1       goto loop 1         / test still card-to-tape
        goto tstssw-2                           / terminate operations
/ set sense switch 1 down to stop card-to  tape operation
ceof,   wreof       1                           / write eof on unit 1
        stop                                    / stop machine
        tss         1       goto ctt            / test continue cit
```

```
        goto loop l + 7                                 / no

tieof,  typtxt      msl                                 / type end-point mess
        stop                                            / stop machine
        goto loop l + 7                                 / continue
/ set sense switch 2 up for tape- to-card operation
ttc,    open        2       lØ          8l              / open tape-to-card
        set         2                                   / set prog sw 2
loop 2, rdtape      2       pcha        goto t2eof      / read tape
        puncrd      pcha                                / punch card
        tss         2       goto .+3                    / test ssw 2
        clear       2                                   / clear prog sw 2
        tps         3       goto loop 3                 / test tape-to-printer
        tss         3       goto ttp                    / test start ttp
        tps         l       goto loop l                 / test card-to-tape
        tss         l       goto ctt                    / test start ctt
        tps         2       goto loop 2                 / test continue ttc
        goto tstssw - 2
/ set sense switch 2 down to stop tape-to-card operation
t2eof,  endpoint    goto t2ep                           / test if end-point
        stop                                            / no-stop machine
        tss         2       goto ttc                    / test continue ttc
        gon loop 2 + 6                                  / no
/ type end-point message
t2ep,   typtxt      msl
        stop
        goto loop 2
/ set sense switch 3 up for tape-to-printer operation
ttp,    open        3       lØ          8l              / tape-to-printer open
        clrstr      l2Ø     pta                         / clear print area
```

```
        set         3                                   / set prog sw 3
loop 3, rdtape      3       pta         goto t3eof      / read tape
        prnlin      pta                                 / print line
        tss         3       goto .+3                    / test continue ttp
        clear       3                                   / no - clear prog sw 3
        tps         1       goto loop 1                 / test card-to-tape
        tss         1       goto ctt                    / test start ctt
        tps         2       goto loop 2                 / test tape-to-card
        tss         2       goto ttc                    / test start ttc
        tps         3       goto loop 3                 / test continue ttp
        goto tstssw                                     / no
/ set sense switch 3 down to stop tape-to-printer operation
t3eof,  endpoint            goto t3ep
        stop
        tss         3       goto ttp
        goto loop 3 + 6
/ type end-point message
t3ep,   typtxt              msl
        stop
        goto loop 3
/ in put - out put areas
cdq,    cda + 81/
pcha,   pcha + 81/
prt,    prt + 120/
/ teletype out put message
        teletype
msl,    text /
tape at end-point
load next tape, press continue.
/
start strt
```