HEWLETT hp PACKARD

SOFTWARE OPERATING PROCEDURES

SOFTWARE OPERATING PROCEDURES

INTRODUCTION

HEWLETT PACKARD

11000 Wolfe Road
Cupertino, California 95014

5951-1369

April, 1971

INTRODUCTION

SOFTWARE OPERATING PROCEDURES is a collection of small publications--called modules--that contain operating instructions for HP software products. Each user has a customized set of these modules that covers the software relevant to his system.

The SOP (*SOFTWARE OPERATING PROCEDURES*) is a structured set of modules organized into five levels:

Level I:	The Introduction
Level II:	Hardware Considerations
Level III:	Utility Software
Level IV:	Software Sub-systems
Level V:	Software Operating Systems

The modules in each level of the SOP refer to general information in modules of the preceding levels. In this way, information need not be duplicated.

The title page of each module contains the following information:

Title.
Other modules required.
Reference manuals required.
Module document number.
Module publication date.

Each user of HP software should have the following modules of the SOP:

Level I:	Introduction
Level II:	A module describing his computer and one describing peripherals.
Level III:	Modules describing BBL, BBDL, and SIO, plus any other utility software included in his system.
Level IV:	A module containing language error messages and modules for any sub-systems included in his system.
Level V:	One module describing creation and use of his operating system (BCS, DOS-M, etc.)

In addition, the user should have the following two books if he plans to use the related software:

SYMBOLIC EDITOR (02116-9016)

PREPARE TAPE SYSTEM (02116-91751)

CONVENTIONS USED IN THE SOP

Information printed on the terminal by the computer appears in the text as

OUTPUT EXAMPLE

Information typed on the teleprinter by the user appears in the next as

INPUT EXAMPLE

Items within input or output that appear as

VARIABLE

are variable items and stand for a class of possible entries.

The contents of the registers of the computers (i.e., switch registers, memory data registers, etc.) may appear as a series of 16 binary digits (bits) organized into octal digits:

```
0/000/000/000/000/000
↑                   ↑
Bit 15          Bit 0
```

0 means the bit is off or down (equal to binary 0).

● means the bit is on or up (equal to binary 1).

/000 represents a octal digit (e.g., /0●0 = 2_8).

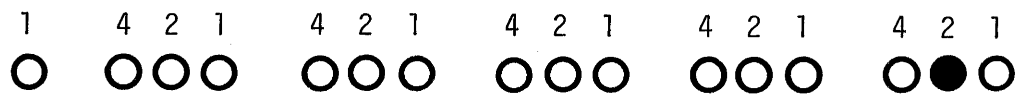
For example: ●/000/0●0/000/●●●/●●● = 102077_8.

USING THE HEWLETT-PACKARD COMPUTER REGISTERS

Digital computers use the binary number system. The Hewlett-Packard digital computers work with a computer word containing 16 binary digits or bits. Since the registers can represent 16 binary digits and an octal digit is three bits, they can be divided into five full octal digits and one partial digit (i.e., one bit left over). The 16 register display lights or switches are numbered 0 through 15 (from right to left).

How To Read The Octal Contents In The Register

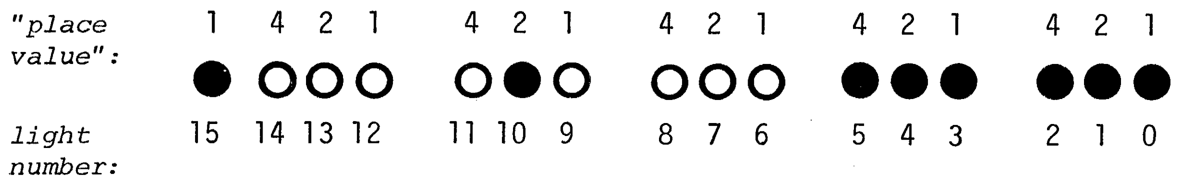
Since each octal number is made up of a series of binary digits, 2_8 appears on the register as:



The binary number is 0000000000000010_2, and the octal equivalent is 000002_8.

Each light on the register display has a "place value" of 1,2, or 4.

To read the octal contents, add the "place values" for each series of three register bits, for example:



is 102077_8.

To interpret each digit in 102077_8, add the "place values" (in groups of three), from left to right:

1, is represented by bit 15. Since the computer word is 16 bits long, bit 15 is the only one in the first octal digit.

0, (the first series of bits) is: 0 + 0 + 0.

2, (the second series of bits) is: 0 + 2 + 0.

0, (the third series of bits) is: 0 + 0 + 0.

7, (the fourth series of bits) is: 1 + 2 + 4.

7, (the fifth series of bits) is: 1 + 2 + 4.

SOME EXAMPLES

1 4 2 1 4 2 1 4 2 1 4 2 1 4 2 1

○ ○○○ ○○○ ○○○ ○○○ ●●● = 7_8

NOTE: 7 is the largest octal digit; there is no 8 or 9 digit.

1 4 2 1 4 2 1 4 2 1 4 2 1 4 2 1

○ ○○○ ○○○ ○○○ ○○● ○○○ = 10_8

NOTE: 10_8 *is a combination of the octal digits 0 and 1.*

1 4 2 1 4 2 1 4 2 1 4 2 1 4 2 1

○ ○○○ ○○○ ○○○ ○○● ●●● = 17_8

1 4 2 1 4 2 1 4 2 1 4 2 1 4 2 1

○ ○○○ ○○○ ○●○ ○○○ ○○○ = 200_8

1 4 2 1 4 2 1 4 2 1 4 2 1 4 2 1

○ ○○○ ○○● ●●● ○○○ ○○○ = 1700_8

1 4 2 1 4 2 1 4 2 1 4 2 1 4 2 1

○ ●●○ ●●● ○○○ ○○○ ○○○ = 67000_8

1 4 2 1 4 2 1 4 2 1 4 2 1 4 2 1

● ●●● ●●● ●●● ●●● ●●● = 177777_8

SOFTWARE OPERATING PROCEDURES

HP 2100A FRONT PANEL PROCEDURES

PREREQUISITE SOP MODULES:

Introduction (5951-1369)

Basic Binary Loader -
Basic Binary Disc Loader (5951-1376)

HEWLETT hp PACKARD

11000 Wolfe Road
Cupertino, California 95014

5951-1371 August 1971

HP 2100A FRONT PANEL PROCEDURES

This SOP module consists of several procedures for the operation of the front panel pushbuttons and DISPLAY REGISTER for the HP 2100A computer:

PROCEDURE 1
POWER ON AND OFF

If the computer is mounted in a cabinet or rack that includes a master power switch, make sure that switch is on before turning on the computer POWER key.

To turn on the power to the computer, turn the power key to either the POWER ON or the LOCK ON position. The POWER ON position allows full use of the front panel buttons and display register; the LOCK ON position disables the front panel buttons.

To turn off computer power, turn off all peripheral I/O devices, then turn the power key to the POWER OFF position. The front panel display lights go out.

PROCEDURE 2

SETTING THE SWITCH REGISTER

The switch register consists of 16 binary bits numbered from 15 to 0. When the S button is lit, each button of the DISPLAY REGISTER represents a corresponding binary bit of the switch register.

Computer Halted

1. Press the S button to light it. A copy of the switch register contents are displayed in the front panel DISPLAY REGISTER.
2. To set a switch register bit on (to the "1" state), press the comparably numbered DISPLAY REGISTER button to light it.
3. To set a switch register bit off (to the "0" state) press the comparably numbered DISPLAY REGISTER button to turn it off. To clear the switch register (set all bits off), press the CLEAR DISPLAY button.
4. Press the RUN button, or any other register button (A,B,M,P,MEMORY DATA) other than the S button. The contents of the DISPLAY REGISTER are transferred into the switch register, replacing the original switch register contents.

   NOTE: If the S button is pressed again before step 4, the original contents of the switch register are again loaded into the DISPLAY REGISTER. The original switch register contents are not modified.

Computer Running

When the computer is executing a program, the S button remains lit and the current contents of the switch register are displayed continuously in the DISPLAY REGISTER. Any modifications to the DISPLAY REGISTER contents are transferred to the switch register right away. (See Computer Halted, steps 2 and 3.)

PROCEDURE 3
SETTING A STARTING ADDRESS

1. Press the P button to light it. The current contents of the P-Register (Program Counter) are displayed in the DISPLAY REGISTER.

2. If the DISPLAY REGISTER does not show the octal starting address desired, modify the contents to the correct address by pressing the DISPLAY REGISTER buttons.

3. Press the RUN button, or any other register button (A,B,M,S, MEMORY DATA) other than the P button. The contents of the DISPLAY REGISTER are transferred into the P-Register, replacing the original P-Register contents.

   NOTE: If the P button is pressed again before step 3, the original contents of the P-Register are again loaded into the DISPLAY REGISTER. The original P-Register contents are not modified.

PROCEDURE 4

EXAMINING THE CONTENTS OF A MEMORY LOCATION

1. Press the M button to light it. The current contents of the M-register (Memory Address Register) are displayed in the DISPLAY REGISTER.

2. If the DISPLAY REGISTER does not show the desired octal address, modify the contents to the correct address by pressing the DISPLAY REGISTER buttons.*

3. Press the MEMORY DATA button. The address set in the DISPLAY REGISTER is transferred into the M-register. The contents of the memory location addressed in the M-register are copied into the DISPLAY REGISTER.

4. To examine the contents of the next higher (or lower) adjacent memory location press the INCREMENT M (or the DECREMENT M) button. The contents of the DISPLAY REGISTER are transferred into the location addressed in step 3. The contents of the adjacent memory location are copied into the DISPLAY REGISTER.

5. To examine the contents of a non-adjacent location, repeat steps 1 through 3 instead of step 4.

*If the address shown in the DISPLAY REGISTER points to a memory location reserved for the Basic Binary Loader or Basic Binary Disc Loader (last 64_{10} words of memory), press to light the LOADER ENABLE button before executing step 3.

PROCEDURE 5

CHANGING THE CONTENTS OF A MEMORY LOCATION

1. Press the M button to light it. The current contents of the M-register (Memory Address Register) are displayed in the DISPLAY REGISTER.

2. If the DISPLAY REGISTER does not show the desired octal address, modify the contents to the correct address by pressing the DISPLAY REGISTER buttons.*

3. Press the MEMORY DATA button. The address set in the DISPLAY REGISTER is transferred into the M-register. The contents of the memory location addressed in the M-register are copied into the DISPLAY REGISTER.

4. Set the DISPLAY REGISTER to the desired contents.

5. To examine or modify the next higher (or lower) adjacent location in memory press the INCREMENT M (or the DECREMENT M) button. The contents of the DISPLAY REGISTER are transferred into the location addressed in step 3. The contents of the adjacent memory location are copied into the DISPLAY REGISTER; return to step 4.

6. To examine and modify the contents of a non-adjacent location, repeat steps 1 through 4 instead of step 5.

*If the address shown in the DISPLAY REGISTER points to a memory location reserved for the Basic Binary Loader or Basic Binary Disc Loader (last 64_{10} words of memory), press to light the LOADER ENABLE button before executing step 3.

PROCEDURE 6
STARTING PROGRAM EXECUTION

1. Set the required program starting address by using PROCEDURE 3.

2. If the program is being run for the first time (the program has just been loaded into memory using the Basic Binary Loader or the Basic Binary Disc Loader), press the INTERNAL PRESET button then press the EXTERNAL PRESET button.

3. If the program is in transition from one logical sequence to another, it is unnecessary to press either PRESET button.

4. Press the RUN button.

PROCEDURE 7

CHANGING THE CONTENTS OF THE A-REGISTER OR B-REGISTER

1. Press the A button (to display the A-register) or the B button (to display the B-register). The A- (or B-) register contents are displayed in the DISPLAY REGISTER.

2. Set the DISPLAY REGISTER to the desired contents.

3. Press the RUN button, or any other register button (A,B,P,M,S, MEMORY DATA) other than the A (or B) button. The DISPLAY REGISTER contents are transferred to the A- (or B-) register, replacing the original A- (or B-) register contents.

   NOTE: If the A (or B) button is pressed again before step 3, the original contents of the A- (or B-) register are reloaded into the DISPLAY REGISTER. The original A- (or B-) register contents are not modified.

PROCEDURE 8
INTERPRETING HALT CODES

Whenever a program executes a halt instruction, the MEMORY DATA button lights and an octal number called a halt code is displayed in the DISPLAY REGISTER. Refer to the appropriate software documentation for the meaning of the halt code; a sample of halt code listings appears in "USING THE BASIC BINARY LOADER or THE BASIC BINARY DISC LOADER." Do not modify the DISPLAY REGISTER contents until another button (A,B,P,M,S) is pressed.

PROCEDURE 9

USING THE BASIC BINARY LOADER OR BASIC BINARY DISC LOADER

The Basic Binary Loader (BBL) or the Basic Binary Disc Loader (BBDL) loads absolute binary program tapes into core memory. Either the BBL or BBDL resides in the last 64_{10} words of core.

BBL or BBDL Paper Tape Loading

1. Press the HALT/CYCLE button to light it.
2. Place the tape to be loaded in the tape input device and ready that device.
3. Set the BBL (or BBDL) starting address (see PROCEDURE 3) according to the memory size of the computer:

MEMORY SIZE	STARTING ADDRESS
4K	007700 (octal)
8K	017700
12K	027700
16K	037700
24K	057700
32K	077700

4. Clear the switch register (set all bits off). (Press S then press CLEAR DISPLAY.)
5. Press the LOADER ENABLE button to light it.
6. Press the INTERNAL PRESET button, then press the EXTERNAL PRESET button.
7. Press the RUN button.
8. After all or part of the tape is read, the computer halts with $1020xx_8$ in the DISPLAY REGISTER. The LOADER ENABLE button light goes out (Loader now protected).

   If *xx* = 11, a checksum error was detected. Check for torn tape or dust in reader, check the tape for ragged edges or torn holes, then return to step 2.

8. (cont.)

If *xx* = 55, an address error was detected. A program being loaded attempted to enter a location reserved for the BBL (or BBDL), or a location not available in the computer. Check that an absolute binary tape was used, and that it was placed properly in the reader.

If *xx* = 77, the tape loaded correctly.

BBL Checksum Verification

NOTE: A checksum verification is performed by the BBL only-BBDL cannot perform this operation.

1. Press the HALT/CYCLE button to light it.
2. Place the absolute binary tape in the tape input device and ready that device.
3. Set the BBL starting address (see PROCEDURE 3) according to the memory size of the computer:

MEMORY SIZE	STARTING ADDRESS
4K	007700 (octal)
8K	017700
12K	027700
16K	037700
24K	057700
32K	077700

4. Set switch register bit 0 on (all others off) (PROCEDURE 2).
5. Press the LOADER ENABLE button to light it.
6. Press the INTERNAL PRESET button then press the EXTERNAL PRESET button.
7. Press the RUN button.
8. After all or part of the tape is read, the computer halts with $1020xx_8$ in the DISPLAY REGISTER. The LOADER ENABLE button light goes out (Loader now protected).

8. (cont.)

If *xx* = 11, a checksum error was detected. The A-register (press A to display it) contains the checksum from the tape; the B-register (press B to display it) contains the computed checksum. Check for torn tape or dust, then return to step 2.

If *xx* = 77, the tape checksum verified.

NOTE: The tape is not read into memory during a checksum verification. To load the tape into memory, refer to "BBL OR BBDL PAPER TAPE LOADING."

BBL Comparing a Paper Tape With Memory

NOTE: A tape comparison with memory is performed by the BBL only - the BBDL cannot perform this operation.

1. Place the tape to be compared into the tape input device and ready that device.

2. Set the BBL starting address (see PROCEDURE 3) according to the memory size of the computer:

MEMORY SIZE	STARTING ADDRESS
4K	007700 (octal)
8K	017700
12K	027700
16K	037700
24K	057700
32K	077700

3. Set switch register bit 15 on (all others off) (PROCEDURE 2).

4. Press the LOADER ENABLE button to light it.

5. Press the INTERNAL PRESET button then press the EXTERNAL PRESET button.

6. Press the RUN button.

7. After all or part of the tape is read, the computer halts with $1020xx_8$ in the DISPLAY REGISTER. The LOADER ENABLE button light goes out (Loader now protected).

   If *xx* = 00, the tape contents do not compare with memory contents. The A-register (press A to display it) contains the tape data that does not agree.

   To find the address of the memory contents which did not agree with the tape, press to light the LOADER ENABLE button (enable the Loader), press the INSTR STEP (instruction step) button, then press the MEMORY DATA button. The DISPLAY REGISTER contents minus one is the address of the contents that did not agree. To examine those contents, follow the steps outlined in "EXAMINING THE CONTENTS OF A MEMORY LOCATION" for the address (minus one) shown in the DISPLAY REGISTER.

   To restart the comparison process return to step 1. If the tape fails to compare a second time, inspect the BBL (or BBDL) to ensure that it is intact. (See PROTECTING AND ENABLING THE BBL OR THE BBDL.")

   If *xx* = 77, the tape successfully compared.

Protecting and Enabling the BBL or the BBDL

To use the BBL or BBDL in any way, or to inspect it and make any modifications to it, the LOADER ENABLE button must be lit.

If the button is not lit, the last 64 words of core (where the BBL or BBDL resides) are protected from examination, modification, or illegal entry by a program.

Once the LOADER ENABLE button is pressed, it stays lit (the Loader is enabled) until one of three actions occurs:

1. The LOADER ENABLE button is pressed to turn it off.
2. An executing program halts because of hardware error or because a HALT instruction is executed.
3. The HALT/CYCLE button is pressed.

SOFTWARE OPERATING PROCEDURES

SOFTWARE INPUT/OUTPUT SYSTEM CONFIGURATION

PREREQUISITE SOP MODULES:

Front Panel Procedures Module
Peripheral Equipment

HEWLETT hp PACKARD

11000 Wolfe Road
Cupertino, California 95014

5951-1374 April, 1971

SOFTWARE INPUT/OUTPUT SYSTEM CONFIGURATION

A peripheral device cannot operate without a utility program in core to activate and control that device. Such a program is called an input/output (I/O) driver.

Each device is identified by an octal number--its select code--that specifies the physical I/O channel into which the device is connected; its driver uses the select code in its instructions to reference the device. The driver is stored in core and when the instructions reference the select code, the computer produces a signal that activates the device. The process of loading a device driver into core locations and assigning a device to the driver is called configuration.

The procedures covered in this SOP module are:

SOFTWARE INPUT/OUTPUT (SIO) DRIVERS

SIO drivers are the minimum software environment for the HP Assembler, HP FORTRAN, HP ALGOL, Punch-Verify, Symbolic Editor, Cross-Reference Symbol Table Generator, Prepare Tape System, SIO System Dump, System Dump, DOS System Generator, RTE System Generator, and DOS-M System Generator.

These drivers are designed to make efficient use of core. They are not necessarily time-efficient.

SIO drivers are absolute programs that reside in high core, just below the BBL or BBDL. Therefore, there are different sizes of SIO drivers: 4K for use with 4K memory size, 8K for use with 8K memory size, and 16K for use with either 16K, 24K, or 32K memory size.

Before using a computer system, make the device assignments for the SIO drivers. SIO drivers must be configured in the order listed below:

SIO Teleprinter driver
SIO Line Printer driver
SIO Tape Reader driver or SIO Card Reader driver
SIO Tape Punch driver
SIO Magnetic Tape driver or SIO Disc/Drum driver

The SIO Teleprinter driver is always required unless program operating instructions state otherwise.

PROCEDURE 1
SIO MODULES

An SIO module consists of one to five configured SIO drivers (always containing the SIO Teleprinter driver) on one roll of tape. Since SIO drivers are distributed separately, the user must combine the desired drivers into an SIO module. If configuring SIO modules for the Magnetic Tape System, do not include the Magnetic Tape Driver in any of the modules. Make all the SIO modules that are appropriate to your system using this procedure:

1. Use the BBL or BBDL to load the SIO driver.
2. Set up a starting address of 2_8.
3. Set the high priority I/O address (select code) of the device whose driver is being configured into switch register bits 5 through 0. Bits 15 through 6 are "OFF."

   NOTE: For the SIO Teleprinter driver, set switch register bit 15 equal to one if the teleprinter is a 2754B. Bits 14 through 6 of the SWITCH REGISTER are zero.

4. Press RUN.
5. Repeat steps 1 through 4 for each I/O device driver in this order:

   SIO Teleprinter driver

   Optional {
   - SIO Line Printer driver (see Note 1)
   - SIO Card Reader driver or SIO Tape Reader driver
   - SIO High-Speed Tape Punch driver
   - SIO Magnetic Tape driver or SIO Disc/Drum driver (see Note 2).

NOTE 1: If the SIO Line Printer driver is loaded, the High-Speed Tape Punch driver must also be loaded in order to punch paper tape output.

NOTE 2: SIO Disc/Drum drivers require special device assignment procedures. Refer to Procedure 3 in this section.

6. Load the SIO System Dump with BBL or BBDL.

7. Set up a starting address of 2_8.

8. Set all switch register bits to zero.

9. Turn on the teleprinter punch or tape punch (requires Tape Punch Driver).

10. Press RUN. The configured drivers are punched and the computer halts.

11. Additional copies of the SIO module can be obtained by pressing RUN.

PROCEDURE 2
RECOMMENDED SIO PROGRAM CONFIGURATION

After completing the necessary procedures to produce modules configured with one or more drivers, choose one of the programs listed below:

Assembler
Extended Assembler
HP FORTRAN
4K FORTRAN
ALGOL
Cross Reference Symbol Table Generator
Symbolic Editor
Prepare Tape System
System Dump
Any user program that has properly preset the first word of available memory (105_8) and allocated driver link words 100_8-107_8.

If any one of these programs is to be used, it is more convenient to combine the SIO drivers and the program onto one tape than to load the two tapes separately each time. Use the following procedure:

1. Load the program tape with BBL or BBDL.
2. Choose one of the SIO module tapes containing the desired SIO drivers. If module not ready, create SIO module now. (Procedure 1)
3. Load SIO module tape with BBL or BBDL.
4. Load SIO System Dump with BBL or BBDL. (Must be same core size as drivers.)
5. Set up a starting address of 2_8.
6. Set all switch register bits to zero.
7. Set switch register bit 15 to one for a tape containing both the program and the drivers.

8. Turn on the tape punch or teleprinter punch.

9. Press RUN.

10. Additional copies of the "configured" program can be obtained by pressing RUN.

This tape is a configured SIO program ready to be loaded with BBL and run; i.e., a "configured" Assembler. Now repeat the process for all other programs to be used.

PROCEDURE 3

SIMULATION OF A MAGNETIC TAPE ENVIRONMENT BY AN SIO DISC DRUM DRIVER

The SIO Disc/Drum driver simulates a magnetic tape on the fixed-head disc or drum, enabling a disc to be used instead of a magnetic tape in a magnetic tape environment. This saves time in the generation of the Disc Operating System and Real Time Executive System.

The SIO Disc/Drum driver takes up the same locations in core as the SIO Magnetic Tape drivers; therefore, this driver cannot be used in a Magnetic Tape System.

HOW TO CONFIGURE AN SIO DISC/DRUM DRIVER

NOTE: The SIO Disc/Drum Driver must be the last SIO driver loaded with BBL or BBDL.

1. Load the SIO Disc/Drum driver with BBL or BBDL.
2. Set up a starting address of 2_8.
3. Set the high priority I/O address (select code) of the disc/drum DATA channel into switch register bits 5 through 0.
4. Set switch register bit 15 to zero, if there are 90 sectors/programming track.
   Set switch register bit 15 to one, if there are 128 sectors/programming track.
5. Press RUN. The computer halts with a code of 102001_8.

6. Set the first octal track address to be made available to the driver into switch register bits 7 through 0. Set bits 14 through 12 to the logical unit number of the disc. Set bit 15 on if this is the last logical unit to be configured.

7. Press RUN. The computer halts with a halt code of 102002_8.

8. Set the last octal track address to be made available to the driver into bits 7 through 0.

9. Press RUN. If the computer halts with a code of 102000_8, the last track for the current logical unit is less than that of the first track address given. Restart driver configuration at step 3.

   NOTE: Configuration of the first logical unit sets the other seven logical units to the same configuration in case of premature termination.

10. If bit 15 (step 6) is set to zero, the computer halts with a code of 102001_8 and the user can configure the next logical unit by returning to step 6.

11. If bit 15 (step 6) is set to one or this is the eighth logical unit to be configured, the computer halts with a code of 102077_8. Configuration is complete. To reconfigure, start at step 2.

SOFTWARE OPERATING PROCEDURES

ASSEMBLER, FORTRAN AND ALGOL ERROR MESSAGES

PREREQUISITE SOP MODULES:
Introduction (5951-1369)

REFERENCE MANUALS:
HP Assembler (02116-9014)
HP FORTRAN (02116-9015)
HP ALGOL (02116-9072)

HEWLETT hp PACKARD
11000 Wolfe Road
Cupertino, California 95014

5951-1377 April, 1971

ASSEMBLER, FORTRAN AND ALGOL ERROR MESSAGES

During the compilation or assembly of programs, error messages are typed on the list output device to aid the programmer in debugging programs. This SOP module consists of three procedures as follows:

PROCEDURE 1
ASSEMBLER ERROR MESSAGES

Errors detected in the source program are indicated by a 1- or 2- letter mnemonic followed by the sequence number and the first 62 characters of the statement in error. The messages are printed on the list output device during the passes indicated:

For Extended Assembler, error listings produced during Pass 1 are preceded by a number which identifies the source input file where the error was found. Pass 2 and 3 error messages are preceded by a reference to the previous page of the listing where an error message was written. The first error will refer to page "O".

Error Code	Pass	Description
CS	1	Control statement error: a) The control statement contained a parameter other than the legal set. b) Neither A nor R, or both A and R were specified. c) There was no output parameter (B, T, or L.)
DD	1	Doubly defined symbol: A name defined in the symbol table appears more than once as: a) A label of a machine instruction. b) A label of one of the pseudo operations: BSS EQU ASC ABS DEC OCT DEF Arithmetic subroutine call DEX c) A name in the Operand field of a COM or EXT statement. d) A label in an instruction following a REP pseudo operation.

Error Code	Pass	Description
		e) Any combination of the above.
		An arithmetic subroutine call symbol appears in a program both as a pseudo instruction and as a label.
EN	1	The symbol specified in an ENT statement has already been defined in an EXT or COM statement.
ENØØØ <symbol>	2	The entry point specified in an ENT statement does not appear in the label field of a machine or BSS instruction. The entry point has been defined in the Operand field of an EXT or COM statement, or has been equated to an absolute value.
IF	1	An IFZ or an IFN follows either an IFZ or an IFN without an intervening XIF. The second pseudo instruction is ignored.
IL	1	Illegal instruction: a) Instruction mnemonic cannot be used with type of assembly requested in control statement. The following are illegal in an absolute assembly: NAM, EXT, ENT, COM, ORB, Arithmetic subroutine calls. b) The ASMB statement has an R parameter, and NAM has been detected after the first valid Opcode.
IL	2 or 3	Illegal character: A numeric term used in the Operand field contains an illegal character (e.g. an octal constant contains other than +, -, or Ø-7).
		Illegal instruction: ORB in an absolute assembly.
M	1, 2 or 3	Illegal operand: a) Operand is missing for an Opcode requiring one. b) Operands are optional and omitted but comments are included for: END, HLT

Error Code	Pass	Description
M	1, 2 or 3	c) An absolute expression in one of the following instructions from a relocatable program is greater than 77_8. Memory Reference DEF Arithmetic subroutine calls
		d) A negative operand is used with an Opcode field other than ABS, DEX, DEC, and OCT.
		e) A character other than I follows a comma in one of the following statements: ISZ ADA AND DEF JMP ADB XOR Arithmetic JSB LDA IOR subroutine LDB CPA calls STA CPB STB
		f) A character other than C follows a comma in one of the following statements: STC MIB CLC OTA LIA OTB LIB HLT MIA
		g) A relocatable expression in the operand field of one of the following: ABS ASR RRL REP ASL LSR SPC RRR LSL
		h) An illegal operator appears in an Operand field (e. g. + or - as the last character).
		i) An ORG statement appearing in a relocatable program includes an expression that is base page or common relocatable or absolute.
		j) A relocatable expression contains a mixture of program, base page, and common relocatable terms.

Error Code	Pass	Description
		k) An external symbol appears in an operand expression or is followed by a common and the letter I.
		l) The literal or type of literal is illegal for the operation code used (e.g., STA = B7).
		m) An illegal literal code has been used (e.g., LDA = 077).
		n) An integer expression in one of the following instructions does not meet the condition $1 \leq n \leq 16$. The integer is evaluated modulo 2^4. ASR RRR LSR ASL RRL LSL
		o) The value of an 'L' type literal is relocatable.
NO	1, 2, 3	No origin definition: The first statement in the assembly containing a valid opcode following the ASMB control statement (and remarks and/or HED, if present) is neither an ORG nor a NAM statement. If the A parameter was given on the ASMB statement, the program is assembled starting at 2000; if an R parameter was given, the program is assembled starting at zero.
OP	1, 2, 3	Illegal Opcode preceding first valid Opcode. The statement being processed does not contain an asterisk in position one. The statement is assumed to contain an illegal Opcode; it is treated as a remarks statement.
OP	1,2, or 3	Illegal Opcode: A mnemonic appears in the Opcode field which is not valid for the hardware configuration or assembler being used. A word is generated in the object program.
OV	1,2, or 3	Numeric operand overflow: The numeric value of a term or expression has overflowed its limit:

Error Code	Pass	Description
		$1 \geq N \geq 16$ Shift-Rotate Set
		2^6-1 Input/Output, Overflow, Halt
		$2^{10}-1$ Memory Reference (in absolute assembly)
		$2^{15}-1$ DEF and ABS operands; data generated by DEC; or DEX: expressions concerned with program location counter.
		$2^{16}-1$ OCT
R?	Before 1	An attempt is made to assemble a relocatable program following the assembly of an absolute program.
SO		There are more symbols defined in the program than the symbol table can handle.
SY	1,2,3	Illegal Symbol: A Label field contains an illegal character or is greater than 5 characters. A label with illegal characters may result in an erroneous assembly if not corrected. A long label is truncated on the right to 5 characters.
SY	2 or 3	Illegal Symbol: A symbolic term in the Operand field is greater than five characters; the symbol is truncated on the right to 5 characters. Too many control statements: A control statement has been input both on the teleprinter and the source tape or the source tape contains more than one control statement. The Assembler assumes that the source tape control statement is a label, since it begins in column 1. Thus, the commas are considered as illegal characters and the "label" is too long. The binary object tape is not affected by this error, and the control statement entered via the teleprinter is the one used by the Assembler.
TP	1,2, or 3	An error has occurred while reading magnetic tape.
UN	1,2, or 3	Undefined Symbol:

Error Code	Pass	Description
		a) A symbolic term in an Operand field is not defined in the Label field of an instruction or is not defined in the Operand field of a COM or EXT statement.
		b) A symbol appearing in the Operand field of one of the following pseudo operations was not defined previously in the source program: BSS ASC EQU ORG END

PROCEDURE 2
FORTRAN ERROR MESSAGES

Errors detected in the source program are indicated by a nemeric code inserted before or after the statement in the List Output.

The format is as follows:

E-eeee:	ssss + nnnn
eeee	The error diagnostic code shown below.
ssss	The statement label of the statement in which the error was detected. If unlabeled, 0000 is typed.
nnnn	Ordinal number of the erroneous statement following the last labeled statement. (Comment statements are not included in this count.)

Error Code	Description
0001	Statement label error: a) The label is in positions other than 1-5. b) A character in the label is not numeric. c) The label is not in the range 1-9999. d) The label is doubly defined. e) The label indicated is used in a GO TO, DO, or IF statement or in an I/O operation to name a FORMAT statement, but it does not appear in the label field for any statement in the program (printed after END).
0002	Unrecognized Statement a) The statement being processed is not recognized as a valid statement. b) A specifications statement follows an executable statement.

Error Code	Description
	c) The specification statements are not in the following order: DIMENSION COMMON EQUIVALENCE
	d) A statement function precedes a specification statement.
0003	Parenthesis error: There are an unequal number of left and right parentheses in a statement.
0004	Illegal character or format:
	a) A statement contains a character other than A through Z, 0 through 9, or space =+-/(),.$".
	b) A statement does not have the proper format.
	c) A control statement is missing, misspelled, or does not have the proper format.
	d) An indexing parameter of a DO-loop is not an unsigned integer constant or simple integer variable or is specified as zero.
0005	Adjacent operators: An arithmetic expression contains adjacent arithmetic operators.
0006	Illegal subscript: A variable name is used both as a simple variable and a subscripted variable.
0007	Doubly defined variable:
	a) A variable name appears more than once in a COMMON statement.
	b) A variable name appears more than once in a DIMENSION statement.
	c) A variable name appears more than once as a dummy argument in a statement function.

Error Code	Description
	d) A program subroutine, or function name appears as a dummy parameter; in a specifications statement of the subroutine or function; or as a simple variable in a program or subroutine.
0008	Invalid parameter list:
	a) The dummy parameter list for a subroutine or function exceeds 63.
	b) Duplicate parameters appear in a statement function.
0009	Invalid arithmetic expression:
	a) Missing operator
	b) Illegal replacement
0010	Mixed mode expression: integer constants or variables appear in an arithmetic expression with real constants or variables.
0011	Invalid subscript:
	a) Subscript is not an integer constant, integer variable, or legal subscript expression.
	b) There are more than two subscripts (i.e., more than two dimensions.)
	c) Two subscripts appear for a variable which has been defined with one dimension only.
0012	Invalid constant:
	a) An integer constant is not in the range of -2^{15} to 2^{15} -1.
	b) A real constant is not in the approximate range of 10^{38} to 10^{-38}.
	c) A constant contains an illegal character.

Error Code	Description
0013	Invalid EQUIVALENCE statement: a) Two or more of the variables appearing in an EQUIVALENCE statement are also defined in the COMMON block. b) The variables contained in an EQUIVALENCE cause the origin of COMMON to be altered. c) Contradictory equivalence; or equivalence between two or more arrays conflicts with a previously established equivalence.
0014	Table overflow: Too many variables and statement labels appear in the program.
0015	Invalid DO loop: a) The terminal statement of a DO loop does not appear in the program or appears prior to the DO statement. b) The terminal statement of a nested DO loop is not within the range of the outer DO loop. c) DO loops are nested more than 10 deep. d) Last statement in a loop is a GO TO, arithmetic IF, RETURN, STOP, PAUSE, or DO.
0016	Statement function name is doubly defined.

PROCEDURE 3
ALGOL ERROR MESSAGES

Source Program Diagnostic Message

Errors detected in the source program are indicated by a code number and an "↑" below the symbol which caused the error.

Error Code	Description
1	More than two characters used in an ASCII constant
2	@ not followed by an octal digit
3	Octal constant greater than 177777
4	Two decimal points in one number
5	Non-integer following apostrophe
6	Label declared but not defined in program
7	Number required but not present
8	Missing END
10	Undefined identifier
11	Illegal symbol
12	Procedure designator must be followed by left parenthesis
13	Parameter types disagree
14	Name parameter may not be an expression
15	Parameter must be followed by a comma or right parenthesis
16	Too many parameters
17	Too few parameters
18	Array variable not followed by a left bracket
19	Subscript must be followed by a comma or right bracket
20	Missing THEN
21	Missing ELSE
22	Illegal Assignment
23	Missing Right Parenthesis
24	Proper procedure not legal in arithmetic expression
25	Primary may not begin with this type quantity

Error Code	Description
26	Too many subscripts
27	Too few subscripts
28	Variable required
4Ø	Too many external symbols
41	Declarative following statement
42	No parameters declared after left parenthesis
43	REAL, INTEGER, or BOOLEAN illegal with this declaration.
44	Doubly defined identifier or reserved word found
45	Illegal symbol in declaration
46	Statement started with illegal symbol
47	Label not followed by colon
48	Label is previously defined
49	Semicolon expected as terminator
5Ø	Left arrow or := expected in SWITCH declaration
51	Label entry expected in SWITCH declaration
52	Real number assigned to integer
53	Constant expected following left arrow or :=
54	Left arrow or := expected in EQUATE declaration
55	Left bracket expected in array declaration
56	Integer expected in array dimension
57	Colon expected in array dimension
58	Upper bound less than lower bound in array
59	Right bracket expected at end of array dimensions
6Ø	Too many values for array initialization
61	Array size excessive (set to 2Ø47)
62	Constant expected in array initialization
63	Too many parameters for procedure
64	Right parenthesis expected at end of procedure parameter list
65	Procedure parameter descriptor missing
66	VALUE parameter for procedure not in list
67	Illegal TYPE found in procedure declaration
68	Illegal description in procedure declaratives
69	Identifier not listed as procedure parameter
7Ø	No type FOR variable in procedure parameter list
71	Semicolon found in a format declaration
72	Left parenthesis expected after I/O declaration name

Error Code	Description
73	Right parenthesis expected after I/O name parameters
74	Undefined label reference
75	Switch identifier not followed by a left bracket
76	Missing right bracket in switch designator
77	THEN missing in IF statement
78	DO missing in WHILE statement
79	FOR variable must be of type INTEGER
8Ø	FOR variable must be followed by an assign symbol
81	STEP symbol missing in FOR clause
82	UNTIL symbol missing in FOR clause or DO statement
83	DO symbol missing in FOR clause
84	Parenthesis expected in READ/WRITE statement
85	Comma expected in READ/WRITE statement
86	Free field format (*) illegal with WRITE
87	Unmatched [in I/O statement list
88	Missing BEGIN in case statement
89	Missing END in case statement
1ØØ	Program must start with BEGIN, REAL, INTEGER or PROCEDURE. Computer halts with 102077_8 in MEMORY DATA Register.
999	Table areas have overflowed, program halts with 102077_8 in MEMORY DATA Register.

SOFTWARE OPERATING PROCEDURES

SIO SUBSYSTEM OPERATION

PREREQUISITE SOP MODULES:

Introduction (5951-1369)
Front Panel Procedures Module
Assembler, FORTRAN and ALGOL Error Messages (5951-1377)
Software Input/Output System Configuration (5951-1374)

RELATED MANUALS:

HP Assembler (02116-9014)
HP FORTRAN (02116-9015)
HP ALGOL (02116-9072)

HEWLETT *hp* *PACKARD*

11000 Wolfe Road

Cupertino, California 95014

5951-1390 September 1971

SIO SUBSYSTEM OPERATION
INTRODUCTION

The HP Assembler and the HP FORTRAN and HP ALGOL compilers are computer language processors which translate source programs into machine instructions for computer execution. The Cross-Reference Symbol Table Generator, Punch/Verify Routine and SDUMP are utility programs. All are subsystems of the SIO (Software Input/Output) operating system, and are run under the control of SIO.

Data transfer between the subsystems and peripheral devices is handled through noninterrupt Input/Output routines called SIO drivers. These drivers are loaded into memory before subsystem execution begins.

The following procedures give explicit instructions for loading, configuring and running the subsystems mentioned above in the SIO operating system.

The procedures are:

PROCEDURE 1

HP ASSEMBLER AND EXTENDED ASSEMBLER

The Assembler or Extended Assembler converts assembly language source programs into object programs by translating each assembly instruction into one machine language instruction.

One reading of the source program by the assembler is a pass. If a teleprinter is used without another output device such as a high-speed punch or lineprinter, then assembly is completed in three passes (if both binary output and a listing are requested). If the teleprinter punches tape and lists separately, or if an extra output device (list or punch) is used, the assembler makes only two passes.

Assembler output (object program tapes, program listings) is controlled through the assembly control statement. The control statement is input to the Assembler as the first statement of the source program (either on the same tape or on a separate tape) or through the terminal keyboard.

The control statement starts with the word "ASMB" followed by one or more of these parameters (in any order), separated by commas.

Parameter	Meaning
A	Process the source program in absolute format.
R	Process the source program in relocatable format.
B	Binary output in the "A" or "R" format.
L	List the source program during pass 2 (or during pass 3 if binary output was selected in a three-pass configuration).
T	List the symbol table at the end of pass 1.
N	Assemble all instructions between the IFN and XIF instructions.
Z	Assemble all instructions between the IFZ and XIF instructions.

Both IFN-XIF and IFZ-XIF pseudo instructions may be used in the program; however, only one type is selected in a single assembly. If both "N" and "Z" appear in the control statement, the character last listed controls which block of statements are assembled.

Sample Control Statement:	ASMB, R, B, L, T produces a symbol table listing after pass 1, a relocatable program object tape after pass 2, and a source listing at the end of pass 2 or 3, depending upon the hardware configuration.

Using the Assembler or Extended Assembler

1. Turn on all necessary peripheral devices.
2. If an appropriately configured Assembler or Extended Assembly tape does not exist, configure the Assembler or Extended Assembler with the appropriate SIO drivers by following the steps outlined in the *SOFTWARE INPUT/OUTPUT SYSTEM CONFIGURATION* module (HP 5951-1374).
3. Load the configured Assembler or Extended Assembler tape into memory using the Basic Binary Loader (BBL) or the Basic Binary Disc Loader (BBDL).
4. Place the source tape in the tape input device and ready that device.
5. If the assembler control statement is to be entered on the source tape (either on separate tape or on the same tape as the source program) set a starting address of 100_8.

   If the assembler control statement is to be entered on the terminal keyboard, set a starting address of 120_8.
6. Start program execution.
7. If the starting address set in step 5 is 100_8, the Assembler reads the source tape.

   If the starting address set in step 5 is 120_8, enter the assembler control statement on the terminal keyboard and press the <u>*return*</u>, then <u>*linefeed*</u> keys. The assembler reads the source tape (pass 1).

8. The assembler halts with halt code $1020xx_8$ displayed. Consult Table SSO-1 for the meaning of the halt and follow the procedures outlined for that code.

9. Replace the source tape in the input device. Ready that device and any other needed output devices.

   Set switch register bit 0 on, if desired, to suppress leader and trailer which the Assembler automatically provides for the object tape. Otherwise, set bit 0 off.

   If both listing and punching are specified on the control statement, and the hardware configuration does not allow listing and punching separately (3-pass operation), set bit 15 on.

   If the hardware configuration allows listing and punching separately (2-pass operation), set switch register bit 15 off.

10. Press RUN. To halt pass 2 at the end of the current source line, set switch register bit 1 on. To continue pass 2 without halting at the end of the next source line, set bit 1 off and press RUN. To halt pass 2 at the end of the next source line, leave bit 1 on and press RUN.

11. After the assembler starts reading the source tape, it halts with halt code $1020xx_8$ displayed. Consult Table SSO-1 for the meaning of the halt and follow the procedures outlined for that code.

Table SSO-1

Halt Code Interpretation

Halt Code	Meaning
102011 (*xx* = 11)	The first pass is complete. The assembler types the symbol table before halting, if specified in the control statement. To begin pass 2, return to step 9.
102023 (*xx* = 23)	The second of three passes has ended. To continue with pass 3, replace the source tape in the input device, and press RUN. To omit pass 3 and assemble another source program, return to step 4.

Table SSO-1. Halt Code Interpretation (cont.)

Halt Code	Meaning
102054 (*xx* = 54)	The assembler has halted because switch register bit 1 was set on. To continue without halting at the end of the next source line, set bit 1 off and press RUN. To continue and halt at the end of the next source line, leave bit 1 on and press RUN.
102055 (*xx* = 55)	Switch register bit 15 is set on. The assembler halted before printing a diagnostic message. Turn off the tape punch and press RUN. The message is typed and the assembler again halts with 102055_8 displayed. Turn the tape punch back on and press RUN. At the end of pass 2, set switch register bit 15 off.
102057 (*xx* = 57)	The assembler has read to the end-of-tape but still expects more source program. Put the next source program segment tape into the input device and press RUN.
102066 (*xx* = 66)	The control statement is either missing or illegal. Correct the control statement and return to step 4.
102077 (*xx* = 77)	End of assembly. Return to step 4 to assemble another source program.

PROCEDURE 2

CROSS REFERENCE SYMBOL TABLE GENERATOR

The Cross Reference Symbol Table Generator (X-REF) processes assembly language source program tapes and lists specified symbols along with the sequence numbers of all statements defining or referring to each symbol. Programs may be on more than one tape. X-REF processes all of the symbols, or a range of symbols (such as symbols starting with the letters A through C only).

If the source program uses the IFN or IFZ psuedo-instructions, doubly defined symbols may appear in the table. X-REF also processes literals. The statement number is always 0000 00 for literals; literal definition is not assigned a statement number. Only the first five characters of a literal, including the =, are cross-referenced. Therefore, = D3156 and = D3157 are listed under = D315. Negative literals are all listed under =D.

X-REF reads source programs from paper or magnetic tape. X-REF checks location 107_8 to see if an SIO magnetic tape driver is resident. If so, X-REF reads the source program from magnetic tape.

Using X-REF

1. Turn on all necessary peripheral devices.
2. If an appropriately configured Cross Reference Symbol Table Generator (X-REF) does not exist, configure X-REF with the appropriate SIO drivers* by following the steps outlined in the *SOFTWARE INPUT/OUTPUT SYSTEM CONFIGURATION* module (HP 5951-1374).
3. Load the configured X-REF tape using the Basic Binary Loader (BBL) or Basic Binary Disc Loader (BBDL).
4. Set a starting address of 100_8.

*For paper tape operation, use the SIO teleprinter driver, and the tape input device driver. For magnetic tape operation, use the SIO teleprinter driver and the magnetic tape unit driver.

5. Place the source program paper tape in the tape input device, or mount the magnetic tape in the magnetic tape device. The source program must be on the third file of the magnetic tape, if it is used.

   NOTE: If the magnetic tape driver is in core, but X-REF is to read from paper tape, memory location 107_8 must be set to 000000_8 and the paper tape input device driver must be loaded into memory. To use the SIO magnetic tape driver at a later time, location 107_8 must be restored or the driver reloaded into memory.

6. Set all switch register bits off.

7. If the entire symbol table is to be printed, set switch register bit 15 off. If only a selected range of symbols is to be processed, set bit 15 on.

8. Start program execution.

9. If switch register bit 15 was set off, the X-REF starts reading the source tape.

   If bit 15 was set on, X-REF types................**ENTER CHARACTER RANGE

   Type the range of starting characters for the symbols to be processed by X-REF on the teleprinter keyboard.

   Example: If labels starting with the letters A, B, and C only are to be listed, type AC *return* *linefeed* on the teleprinter.

   If labels starting with the character = only are to be listed, type = = *return* *linefeed* on the keyboard.

   After the range is typed, X-REF starts reading the tape.

10. Once X-REF starts reading the tape, it halts with the halt code 1020*xx* displayed, where:

xx = 57	End-of-tape or end-of-file has occurred, but the tape does not contain an END statement (signifying that the source program continues on one or more tapes). Place the next tape in the input device and ready that device, then press RUN (without modifying the starting address).

xx = 77 The last tape (containing the END statement to signify the end of source program) has been read. Before halting, X-REF prints out the symbol table according to the character range specified. If no symbol exists in the character range specified, X-REF types E before halting.

11. To list the symbol table for the same (or different) tape(s) (any character range desired), return to step 5. When X-REF halts with halt code 102077_8 displayed it returns to a starting address of 100_8. Resetting the starting address is unnecessary.

PROCEDURE 3

HP FORTRAN

The FORTRAN Compiler or 4K FORTRAN Compiler translates a source HP FORTRAN program into a machine language object program. The FORTRAN Compiler is used with a minimum 8K words of memory, and is composed of two tapes labeled FORTRAN pass 1, and FORTRAN pass 2. The 4K FORTRAN Compiler is used with 4K words of memory only, and consists of four tapes labeled 4K FORTRAN pass 1, 4K FORTRAN pass 2, 4K FORTRAN pass 3 and 4K FORTRAN pass 4.

Compiler output (object program tapes, program listings) is controlled through the compiler control statement. The control statement is input to the compiler as the first statement of the source program (either on the same tape or a separate one). The FORTRAN Compiler (but not the 4K FORTRAN Compiler) also reads the control statement from the teleprinter keyboard, if desired.

The control statement starts with the word "FTN" followed by one or more of these parameters (in any order), separated by commas.

Parameter	Meaning
B	Binary output; punch the program in relocatable binary format suitable for loading by the Basic Control System Relocating Loader.
L	List output; list the source program during execution of pass 1.
A	Assembly listing; list the object program in assembly level language during the last pass.
T	Symbol table; list the symbol table. If both T and A are specified, the first of the two is ignored by the compiler.

PROCEDURE 3

Using FORTRAN Compiler

1. Turn on all necessary peripheral devices.

2. If an appropriately configured FORTRAN pass 1 tape does not exist, configure the FORTRAN pass 1 tape with the appropriate SIO drivers by following the steps outlined in the *SOFTWARE INPUT/OUTPUT SYSTEM CONFIGURATION* module (HP 5951-1374). FORTRAN pass 2 is not configured.

3. Load the configured FORTRAN pass 1 tape into memory using the Basic Binary Loader (BBL) or the Basic Binary Disc Loader (BBDL).

4. Place the source tape in the tape input device and ready that device.

5. To enter the control statement as part of the source program (either on the same tape or a separate one), set a starting address of 100_8.

   To enter the control statement on the teleprinter keyboard, set a starting address of 50_8.

6. Start program execution.* If starting address 50_8 was selected in step 5, type in the compiler control statement. The first intermediate binary tape is punched. Pass 1 halts with halt code $1020xx_8$ displayed.

   where:

xx = 57	End-of-tape has occurred, but more source input is required. Place the next source program segment tape in the tape input device and ready that device. Continue processing by starting program execution.
xx = 77	Pass 1 has found the END$ statement, or has compiled five programs. Pass 1 is complete. Rewind the intermediate binary tape.

7. Load the FORTRAN pass 2 tape using the BBL or BBDL.

8. Place the intermediate binary tape in the input device and ready that device.

*See SOFTWARE OPERATING PROCEDURES -- FRONT PANEL PROCEDURES Module for the computer in use.

9. Set a starting address of 100_8.

10. Start program execution. If requested in the compiler control statement, pass 2 punches a final binary object tape. Pass 2 halts with halt code $1020xx_8$ displayed

    where:

xx = 01	Produce an assembly listing or symbol table listing at this time by placing the intermediate binary tape in the tape input device, readying that device then pressing RUN.
xx = 77	Compilation is complete. Rewind the final binary object tape. To compile another source tape, return to step 3.

Using 4K FORTRAN Compiler

1. Turn on all necessary peripheral devices.

2. If an appropriately configured 4K FORTRAN pass 1 tape does not exist, configure the 4K FORTRAN pass 1 tape with the appropriate SIO drivers by following the steps outlined in the *SOFTWARE INPUT/OUTPUT SYSTEM CONFIGURATION* module (HP 5951-1374). 4K FORTRAN pass 2, pass 3 or pass 4 are not configured.

3. Load the configured 4K FORTRAN pass 1 tape into memory using the Basic Binary Loader (BBL) or the Basic Binary Disc Loader (BBDL).

4. Place the source tape in the tape input device and ready that device.

5. Set a starting address of 100_8.

6. Start program execution. The first intermediate binary tape is punched. Pass 1 halts with halt code $1020xx_8$ displayed.

   where:

xx = 02	End-of-tape has occurred, but more source input is required. Place the next source program segment tape in the tape input device and ready that device. Press RUN.
xx = 77	Pass 1 is complete. Rewind the first intermediate binary tape.

7. Load the 4K FORTRAN pass 2 tape into memory using the BBL or BBDL.

8. Place the first intermediate binary tape in the input device and ready that device.

9. Set a starting address of 100_8.

10. Start program execution. Pass 2 punches the second intermediate binary tape and halts with halt code 102077_8 displayed. Rewind the second intermediate binary tape.

11. Load the 4K FORTRAN pass 3 tape into memory using the BBL or BBDL.

12. Place the second intermediate binary tape in the input device and ready that device.

13. Set a starting address of 100_8.

14. Start program execution. Pass 3 punches the third intermediate binary tape and halts with 102077_8 displayed. Rewind the third intermediate binary tape.

15. Load the 4K FORTRAN pass 4 tape into memory, using the BBL or BBDL.

16. Place the third intermediate binary tape in the input device and ready that device.

17. Set a starting address of 100_8.

18. Start program execution. If requested in the control statement, pass 4 punches the final binary object tape. Pass 4 halts with halt code $1020xx_8$ displayed.

    where:

xx = 01	Produce an assembly listing or symbol table listing at this time by placing the third intermediate binary tape in the input device, readying that device then pressing RUN.
xx = 77	Compilation is complete. Rewind the final binary object tape. To compile another source tape, return to step 3.

PROCEDURE 4
HP ALGOL

The HP ALGOL Compiler translates an HP ALGOL source program into a machine language object program.

One reading of the source program by the compiler is a pass. If a teleprinter is used without another output device such as a high-speed punch or lineprinter, then compilation is completed in two passes (if both binary output and a listing are desired). If the teleprinter punches tape and lists separately, or if an extra output device (list or punch) is used, compilation requires only one pass (listing and punching occur concurrently).

Compiler output (object program tapes, program listings) is controlled through the compiler control statement. The control statement is input to the compiler as the first statement of the source program (either on the same or on a separate tape).

The control statement starts with the word "HPAL" followed by one or more of these parameters (in any order), separated by commas.

Parameter	Meaning
B	Binary output; punch the program in relocatable binary format suitable for loading by the Basic Control System Relocating Loader.
L	List output; list the source program.
A	Object output; list the assembly level program.
P	The source program is a procedure only, not a main program.
S	Sense switch control; read the compiler options through the switch register (ignore the B, L and A parameters following HPAL):

Switch Register Bit	Control Parameter Equivalent
15	B - punch binary object tape
14	L - list source program
13	A - produce object program assembly - level listing.

If the S parameter is included in the control statement, the B, L, and A options are read by the compiler from the switch register. B, L, and A parameters in the control statement are ignored. The switches are read at the beginning of each program line so that any option can be altered during compilation.

Using HP ALGOL Compiler

1. Turn on all necessary peripheral devices.
2. If an appropriately configured HP ALGOL tape does not exist, configure the HP ALGOL tape with the appropriate SIO drivers by following the steps outlined in the *SOFTWARE INPUT/OUTPUT SYSTEM CONFIGURATION* module (HP 5951-1374).
3. Load the configured HP ALGOL tape into memory using the Basic Binary Loader (BBL) or Basic Binary Disc Loader (BBDL).
4. Place the source tape in the tape input device and ready that device.
5. Set a starting address of 100_8.
6. Set switch register bits 15, 14 and 13 for the desired options, if the S parameter is included in the control statement.
7. Start program execution. If ALGOL types "HPAL??" and then halts with halt code 102077 displayed, the control statement is missing or incorrect. Place the corrected source tape in the input device and press RUN.
8. The relocatable binary tape is punched, if B is specified in the control statement.

   If L is specified, a source listing is printed on the list output device.

If both A and L are specified, the source listing, interleaved with the object code listing, is printed on the list output device.

Error messages are included in any listing printed. If only B (punch binary tape) is specified or if S is specified but no option switches are set on, error messages and the incorrect source instructions are printed.

9. When the compiler halts with halt code 102077_8 displayed, compilation is complete. The starting address is set to 100_8 (by the program). To recompile the same source program (with different compiler control statement parameters) or another source program, place the desired source tape in the tape input device and ready that device. Return to step 6.

NOTE: If a teletype only is used without another punch or list device, then only the punch parameter or the list parameters (but not both) may be indicated at one time. To both punch and list in the teletype only environment, make one pass specifying punch only, and one pass specifying list only.

PROCEDURE 5

PUNCH/VERIFY ROUTINE

Punch/Verify Routine (P-V) copies a master tape of any size, frame for frame, regardless of content. Punch/Verify also verifies the copy tape contents against the master tape contents.

This routine requires a tape punch device other than a teleprinter tape punch and also a tape input device. However, separate SIO drivers for those devices are not required; the necessary code for driving the punch and reader is included within the Punch/Verify Routine.

P-V creates a tape copy of a master tape by alternately reading 30 master tape frames (tape characters) and then punching a copy of those frames on tape. To verify the copy against the master, P-V first stores the entire master tape contents in memory. The total number of frames that can be stored depends upon the memory size of the computer. P-V is adjusted by the user for the memory size of the computer used. The following list shows the capacity for various memory sizes.

Memory Size	Frame Capacity	Memory Size	Frame Capacity
4K	7560	20K	40328
8K	15752	24K	48520
12K	23944	28K	56712
16K	32136	32K	64904

Loading Punch/Verify

1. Load the Punch/Verify Routine into memory using the Basic Binary Loader (BBL) or Basic Binary Disc Loader (BBDL).

2. If the computer memory size is other than 8K, modify memory location 364_8 according to the memory size of the computer. The number to be loaded (through the computer front panel) is shown in the table below:

Memory Size	Octal Number (in location 364_8)
4K	7700_8
12K	27700_8
16K	37700_8
20K	47700_8
24K	57700_8
28K	67700_8
32K	77700_8

3. Proceed to either "Punching a Tape Copy" or "Verifying a Tape Copy."

Punching a Tape Copy

1. If Punch/Verify is not already loaded, perform the steps outlined in "LOADING PUNCH/VERIFY."
2. Punch a length of leader on the tape copy.
3. Set a starting address of 100_8.*
4. Start program execution. The computer halts with halt code 102001_8 displayed.
5. Set switch register bits 5 through 0 to the select code of the tape input device.
6. Press RUN. The computer halts with halt code 102002_8 displayed.
7. Set switch register bits 5 through 0 to the select code of the tape punch.
8. Place the master tape to be copied in the tape input device and ready that device.

*See the *SOFTWARE OPERATING PROCEDURES - FRONT PANEL PROCEDURES* module for the computer in use.

9. Press RUN. P-V reads 30 frames (tape characters) of the master tape. Then the routine punches an exact copy of those 30 frames. If one or more of the 30 frames just copied contains any code (other than just the feed-frames), P-V reads another 30 frames and punches a copy of those. If none of the 30 frames just read contained any code (other than the feed-frame holes), the routine halts with halt code 102002_8 displayed.

10. If the routine halts with 102002_8 displayed, examine the master tape to see if the end of the tape has been reached.

11. If the end of the tape has not been reached, return to step 9. If the end of the tape has been reached, remove the master tape from the reader. Then punch a length of trailer on the copy tape.

12. To verify the tape copy just punched against the master tape, follow the steps outlined in "Verifying a Tape Copy." To punch a copy of another (or the same) master tape, set switch register bit 0 on and go to step 8.

Verifying a Tape Copy

1. If Punch/Verify has not been loaded, perform the steps outlined in "Loading Punch/Verify."

2. Set a starting address of 200_8.

3. If Punch/Verify has been used since it was loaded, set all switch register bits off and skip to step 6. Otherwise, configure Punch/Verify by setting switch register bit 0 on and continuing with step 4.

4. Start program execution. The program halts with halt code 102001_8 displayed.

5. Set switch register bits 5-0 to the punched tape reader select code.

6. Place the master tape in the reader.

7. Press RUN. The program reads the master tape until a halt occurs with halt code $1020xx_8$ displayed:

   xx = 01 Examine the master tape. If all of the tape has been read, remove the master tape from the reader and go to step 8. If *xx* = 01 (cont.) the tape has not been entirely read, return to the beginning of step 7.

xx = 77 Go to "Master Tape Memory Overflow."

NOTE: If all of the master tape is read, but the program does not halt, the tape does not contain enough trailer. Press HALT (or HALT/CYCLE) and place at least eight inches of blank (feed-frames only) tape into the reader. Ready the reader and press RUN. Continue with step 8.

8. Place the copy tape in the reader and ready the reader.
9. Set switch register bit 15 on.
10. Press RUN. Punch/Verify reads the copy tape until a halt occurs with halt code $1020xx_8$ displayed:

xx = 02 The copy tape is correct. To verify another copy tape against the same master tape, return to step 8. To read another master tape and verify copies of it, return to step 2.

xx = 55 The copy tape did not verify. Refer to "Comparing Master Tape With Copy Tape Data" to observe the discrepancy.

Master Tape Memory Overflow

1. The master tape being loaded has overflowed memory. Either use a computer with larger memory size, or follow the steps outlined below.
2. Mark the master tape in the reader so that it can be removed and placed back at a later time. Remove the tape from the reader.
3. Place the copy tape in the reader (either at the beginning or the last point marked on the tape) and ready the reader.
4. Set switch register bit 15 on.
5. Press RUN. Punch/Verify reads the copy tape until a halt occurs with halt code $1020xx_8$ displayed:

xx = 02 The copy tape section verified with the section of the master tape currently in memory. If the copy tape was read to the

xx = 02 (cont.)	end, then Punch/Verify is finished with that tape. If the copy tape was not read to the end, mark the tape in the reader and remove it. To verify another copy section with the master tape section currently in memory, return to step 3; otherwise, go to step 6.
xx = 55	The copy tape did not verify. Refer to "Comparing Master Tape With Copy Tape Data" to observe the discrepancy.

6. Place the master tape back in the reader at the last point marked on the tape before it was last removed.
7. Set switch register bit 15 off.
8. Ready the reader and press RUN. Punch/Verify reads the master tape until a halt occurs with halt code $1020xx_8$ displayed:

xx = 01	Examine the master tape. If all of the tape has been read, remove the master tape from the reader. Place the copy tape in the reader at the last point marked and return to step 4. If the tape has not been entirely read, return to the beginning of step 8.
xx = 77	Mark the master tape in the reader so that it can be removed and placed back later. Remove the tape from the reader. Place the copy tape in the reader at the last point marked and return to step 4.

Comparing Master Tape With Copy Tape Data

1. The copy tape did not verify with the master tape.
2. Write down the contents of the computer memory address register.
3. Write down the two characters (feed-frames) just read from the copy tape by copying the A-register contents.
4. Set a starting address of 153_8.
5. Set switch register bit 15 off.
6. Press the SINGLE CYCLE (or HALT/CYCLE) button three times.

7. The two characters (feed-frames) from the master tape are loaded into the A-register. Copy the A-register contents and compare them with the contents copied in step 3.

8. Set a starting address equal to the one copied from the memory address register in step 2.

9. To continue verifying the tape, return to either "Verifying a Tape Copy" step 10, or "Master Tape Memory Overflow," step 5.

PROCEDURE 6

SDUMP

SDUMP is a utility stand-alone routine which dumps information from disc to either paper or magnetic tape. SDUMP also verifies the copy dump against the disc contents or loads the copy information back onto the disc, if desired.

Using SDUMP

1. Turn on all necessary peripheral devices.
2. If an appropriately configured SDUMP tape does not exist, configure SDUMP with the appropriate SIO drivers (teleprinter and magnetic tape drivers for magnetic tape; teleprinter, tape reader, tape punch drivers for paper tape operation), by following the steps outlined in the *SOFTWARE INPUT/OUTPUT SYSTEM CONFIGURATION* module (HP 5951-1374).
3. Load the configured SDUMP into memory using the Basic Binary Loader (BBL) or the Basic Binary Disc Loader (BBDL).
4. Set a starting address of 100_8.
5. Start program execution. SDUMP types a request guide on the teleprinter......................DUMP=D,T[-S][,T[-S]]([]=OPTIONAL)

   ```
   VERIFY = V
   LOAD = L
   TERMINATE = T
   ```
6. SDUMP requests the lower-numbered octal select code (I/O address) for the disc.................................DISC CHNL?

   Type the disc select code, followed by pressing <u>*return*</u> then <u>*linefeed*</u>.
7. SDUMP then types..COMMAND

   Respond according to the option desired:

Option 1 Dump all or part of the disc contents by typing "D," followed by the first, then last disc track-sector numbers to be dumped (step 8).

Option 2 Verify that the copy created by SDUMP (step 8) is correct by comparing the copy with the disc contents (step 9).

Option 3 Load the copy (created in step 8) back onto the disc (step 10).

Option 4 Terminate SDUMP execution (step 11).

NOTE: If magnetic tape is used, SDUMP issues a rewind command to the tape unit during initialization, before and after a verify or load operation, and rewind/standby after T for termination.

8. a. To create a copy of disc contents type... D,T_1-S_1,T_2-S_2 return linefeed

where:

T_1 = starting octal track number for the disc.

S_1 = starting octal sector number. This number may be omitted if the entire track (T_1) is to be dumped.

T_2 = ending octal track number for the disc.

S_2 = ending octal sector number. This number may be omitted if the entire track (T_2) is to be dumped.

b. For magnetic tape storage, SDUMP dumps the requested tracks and sectors onto the magnetic tape, followed by an end-of-file mark. Return to step 7.

c. For paper tape storage, SDUMP punches the first two of the requested disc tracks on paper tape, then halts after typing the message............................CHANGE OUTPUT TAPE, HIT RUN.

Tear off the paper tape from the punch and rewind it. Check the punch for adequate paper tape supply. Start program execution.

SDUMP dumps the next two disc tracks onto paper tape. This cycle repeats until the last disc track - sector is dumped. Return to step 7.

9. a. To verify a magnetic tape copy created by SDUMP, place the copy tape in the tape unit and ready the unit. Then type V <u>*return*</u> <u>*linefeed*</u>. One file of the magnetic tape is read and verified against the contents of the disc. When SDUMP reads the end-of-tape marker, the program types...EOT
and rewinds the magnetic tape. Return to step 7.

NOTE: SDUMP rewinds the magnetic tape before and after a V command.

b. To verify a paper tape copy created by SDUMP, place the copy tape in the tape reader and ready the reader. Type V <u>*return*</u> <u>*linefeed*</u>. SDUMP reads the tape and verifies it against the contents of the disc. Discrepancies are reported on the teleprinter. After the tape is verified, SDUMP types..EOT
and then asks for another command.

If the dump consists of more than one tape, insert the next tape in the reader and type V <u>*return*</u> <u>*linefeed*</u>. Continue to read and verify tapes until the last tape is verified. Return to step 7.

10. a. To load a magnetic tape copy back onto the disc using SDUMP, place the copy tape in the tape unit and ready the unit. Type L <u>*return*</u> <u>*linefeed*</u>. SDUMP loads one file of the magnetic tape onto the disc, then types...EOT
and rewinds the magnetic tape. Return to step 7.

NOTE: SDUMP rewinds the magnetic tape before and after an L command is executed.

b. To load a paper tape copy back onto the disc using SDUMP, place the copy tape in the tape reader and ready the reader. Type L <u>*return*</u> <u>*linefeed*</u>. SDUMP loads the entire tape contents onto the disc, then types...EOT
and asks for a command.

10. b. (cont.)

If the dump consists of more than one tape, place the next tape in the reader and type L *return* *linefeed*. Continue to load tapes until the last tape is loaded. Return to step 7.

11. To terminate SDUMP execution, type T *return* *linefeed*. SDUMP halts after issuing a command to rewind the magnetic tape (if magnetic tape is used).

SDUMP ERROR MESSAGES

The following messages may be printed on the teleprinter by SDUMP:

Statement	Action
STATEMENT ERROR	Retype input statement in correct format.
EOT	The end of the input tape being read has been reached; either load the next tape or go on to the next phase.
CHANGE OUTPUT TAPE, HIT RUN	Two full tracks have been dumped onto paper tape; perform the requested action.
TURN OFF DISC PROTECT, HIT RUN	Set the DISC TRACK PROTECT switch off, then press RUN.
DISC INPUT ERROR	Disc Error Diagnostic, for a Parity, Decode or Abort status after 10 retrys. Input sequence repeated on restart.
DISC WRITE ABORT	Disc Error Diagnostic, for an Abort status after a write attempt. Sequence is repeated if restarted.
TRACK *nnn* (8) SECTOR *mmm* (8)	Identification information for the Disc Error Diagnostic messages described above. *nnn* is the octal track number and *mmm* is the octal sector number where the error occurred.
TAPE/DISC VERIFY ERROR	Disc and tape records do not agree. Disc record is rewritten on restart.

SDUMP ERROR MESSAGES (cont.)

Statement	Action
TAPE CHECKSUM ERROR	The checksum in the tape record does not match the sum computed by SDUMP. Current record is ignored if restarted.
MT ERROR - READ PARITY MT ERROR - EOT, RESTART	Magnetic Tape Errors. Error recovery procedures are completed by driver. Restart to retry sequence.
PARAM ERROR NON-NUMERIC or NON-OCTAL	The Disc Channel Number has been incorrectly specified. Re-enter correct number.

SOFTWARE OPERATING PROCEDURES

GENERATING HP BASIC

PREREQUISITE SOP MODULES:
- Introduction (5951-1369)
- Front Panel Procedures Module
- Peripheral Equipment Manual Functions (5951-1373)

REFERENCE MANUALS:
- HP BASIC (02116-9077)
- Magnetic Tape System (02116-91752)

HEWLETT hp PACKARD

11000 Wolfe Road
Cupertino, California 95014

5951-1388 May, 1971

INTRODUCTION

An HP BASIC system consists of the HP BASIC interpreter and the Prepare BASIC System (PBS) programs. Assembly language subroutines written by the user may be included.

The HP BASIC tape consists of the HP BASIC interpreter. The PBS tape contains drivers for the terminal, photoreader, high-speed punch, and the routines necessary to configure these drivers into an HP BASIC system.

An HP BASIC system is generated by:

- Loading the configuration program (PBS) into memory.
- Loading other tapes (HP BASIC, user subroutines) to be included on the system tape.
- Using PBS to configure the HP BASIC System and to dump it onto a single tape.
- Loading the configured HP BASIC System tape into memory along with any separate programs (HP BASIC, user subroutines) included in the system.

PROCEDURE 1

CONFIGURING AN HP BASIC SYSTEM

1. Decide which elements the configured HP BASIC system tape will contain. The three choices are:
   a. I/O drivers, BASIC, User subroutines
   b. I/O drivers, BASIC
   c. I/O drivers.
2. Turn on all necessary peripheral devices (teleprinter, tape punch, etc).
3. Make sure the computer has halted.
4. Use the Basic Binary Loader (BBL) or the Basic Binary Disc Loader (BBDL) to load the PBS tape into memory.*
5. If option a or b was chosen in step 1, use the BBL or the BBDL to load the HP BASIC tape into memory. If option a or b was not chosen, skip to step 7.
6. If option a was chosen in step 1, then use the BBL or BBDL to load the user-subroutines tapes into memory. If option a was not chosen, skip to step 7.
7. Set a starting address of 2_8.
8. Set the switch register to the octal select code of the terminal. (Set bit 15 OFF.)

* *If an operator error is made or if any tape does not load properly, return to step 3 to reload PBS.*

9. Start PBS execution.

10. The PBS program types:

    PHOTOREADER I/O ADDRESS?

    Type the photoreader octal select code on the teleprinter keyboard, then press the *return* key. If there is no photoreader, then press *return* key only.

11. PBS then types:

    PUNCH I/O ADDRESS?

    Type the high-speed punch octal select code on the teleprinter keyboard, then press the *return* key. If there is no high-speed punch, press the *return* key only.

12. PBS then asks:

    SYSTEM DUMP I/O ADDRESS?

    Type the high-speed punch octal select code on the teleprinter keyboard, then press the *return* key. If no high-speed punch exists, press the *return* key only.

13. PBS then asks:

    CORE SIZE?

    Enter the computer core size (8, 16, 24 or 32), then press the *return* key. (Pressing *return* only indicates an 8K memory size.)

14. If a high-speed punch is available, a configured HP BASIC system tape is punched. If a high-speed punch is not available, the message:

    TURN ON TTY PUNCH, PRESS RUN

    is printed, and the computer halts.

15. Turn the teleprinter punch on and start the computer, without modifying the contents of any computer register.

    The configured HP BASIC system tape is punched on the teleprinter punch and the computer halts.

16. To punch another copy of the system tape, merely restart the computer without modifying any register contents.

NOTE: After the configured system tape is punched (Steps 14, 15 and 16), the configured system remains intact in memory. To run the system right away on the same computer that configured it, start at Step 4 when using PROCEDURE 2 (to avoid loading in the configured system tape). If the system is to run on a computer different from the one that configured it, or on the same computer at a later time, start at Step 1 when using PROCEDURE 2.

PROCEDURE 2
LOADING THE CONFIGURED HP BASIC SYSTEM

1. Turn on all necessary peripheral equipment (teleprinter, tape input device, etc.).

2. Make sure that the computer has halted.

3. Load the configured HP BASIC system tape using the BBL or BBDL.

4. If the HP BASIC interpreter was not included as part of the configured HP BASIC system tape, load the HP BASIC interpreter tape into memory using the BBL or BBDL.

5. If any user subroutines are to be included in the system and if they are not part of the HP BASIC system tape previously loaded, load the user-subroutine tapes using the BBL or BBDL.

6. Set a starting address of 100_8.

7. Start program execution. The message:

   ```
   READY
   ```

   is typed. HP BASIC is ready for use.

SOFTWARE OPERATING PROCEDURES

BASIC CONTROL SYSTEM

PREREQUISITE SOP MODULES:

Front Panel Procedures

Peripheral Equipment Manual Functions (5951-1373)

RELATED MANUALS:

Basic Control System (02116-9017)

Magnetic Tape System (02116-91752)

HEWLETT hp PACKARD

11000 Wolfe Road

Cupertino, California 95014

5951-1391 August 1971

BASIC CONTROL SYSTEM

This module covers procedures for creating and using a Basic Control System (BCS) on an HP computer. BCS is a core-based operating system that requires at least 4K of memory and a teleprinter.

PROCEDURE 1
HOW TO GENERATE A BASIC CONTROL SYSTEM

The stand-alone program Prepare Control System (PCS) is used to generate BCS. The following parameters must be specified during generation (all numbers typed in octal):

1. First Word of Available Memory (FWA MEM)

   This is the lowest memory location that is available to PCS for BCS construction. It should be higher than the last linkage location used in the Interrupt Table ar d if the BCS is to be used within MTS (Magnetic Tape System) it must be set to exactly 110_8 (to allow for MTS linkage locations). (Interrupt Table must be pre-planned before running PCS, since (FWA MEM) depends upon Interrupt Table length.)

2. Last Word of Available Memory (LWA MEM)

   This is the highest memory location available to BCS. This value depends on the core size and the context as follows:

Core Size	Last Word BCS	BCS in MTS
4K	7677	
8K	17677	15777
16K	37677	35777
24K	57677	
32K	77677	

3. Equipment Table (EQT)

   A table of varying size whose entries are numbered sequentially starting with 7. The user relates each entry to a specific I/O device and to an I/O driver. There must be at least one EQT entry per device to be used in BCS.

4. Standard Unit Table (SQT)

   A set of 6 numbers (chosen from the EQT) that specify devices for standard functions (i.e., keyboard, list output, etc.).

5. Interrupt Table

The set of memory locations where interrupts may occur and a matching set of linkage locations (one per interrupt location). Also, an entry point into a driver is associated with each interrupt.

Each interrupt location corresponds directly to the select code of the device, i.e., if the teleprinter select code is 10_8, the interrupt location in memory is 10_8. The linkage location associated with the device must be higher than the highest select code (interrupt location) used.

6. Driver Identification Codes

Driver Identification codes are required when creating the EQT and the Interrupt Table. These are the currently defined driver codes:

Table BCS-1. Driver Codes

ØØ to Ø7	Paper Tape Devices:
	ØØ Teleprinter
	Ø1 Tape Reader
	Ø2 Tape Punch
1Ø to 17	Unit Record Devices:
	1Ø Calcomp Plotter
	11 Card Reader
	12 Line Printer
	15 Mark Sense Card Reader (uses DMA)
	16 80-Column Line Printer
2Ø to 37	Magnetic Tape/Mass Storage Devices:
	21 HP 2Ø2Ø (A or B) Magnetic Tape (7-Track)
	22 HP 3Ø3Ø G Magnetic Tape (9-Track) (uses DMA with character packing)
	23 HP 797Ø (A or B) Magnetic Tape (9-Track)
4Ø to 77	Instruments

OPERATING INSTRUCTIONS

1. Turn on all desired equipment.
2. Load PREPARE CONTROL SYSTEM (PCS) using the Basic Binary Loader (BBL) or Basic Binary Disc Loader (BBDL).
3. Set starting address 2000_8.
4. Set all switch register bits off; then set switches 5 through 0 to the octal select code (I/O channel) of the teleprinter.
5. Start program execution.
6. Set all switch register bits off.
7. PCS asks for the high-speed input device. (Remember to terminate each reply with a *return* and *linefeed*.)*.................................... HS INP?

   Reply with the select code of the high-speed input unit for PCS (either tape reader or teleprinter)...........

8. PCS asks for high-speed punch.....................................HS PUN?

   Type the select code of the tape punch or teleprinter to be used by PCS...........

9. PCS asks for the first word of available memory..................FWA MEM?

   Type the octal address beyond the last address necessary for interrupt linkages...........

10. PCS asks for the last word of available memory...................LWA MEM?

    Type the octal address of last available memory address (first digit must be non-zero)...........

11. PCS prints...*LOAD
    and halts.

    At this point, load the appropriate BCS drivers (Magnetic tape first, if present)

**Terminate any reply typed on the keyboard throughout PCS execution with return linefeed. If an error occurs while typing a response, press RUBOUT, return linefeed, then retype the response.*

one at a time. Place the driver tape in the reader and press RUN.

PCS prints the driver name and absolute memory bounds, then prints *LOAD and halts for the next tape.

Keep loading driver tapes until all are loaded. Then load the Input/Output Control routine (IOC), either buffered or non-buffered.

NOTE: If driver D.21 (HP 2020 (A or B) Magnetic Tape Unit) is loaded, only non-buffered IOC can be used; D.21 turns off the interrupt system. D.11 (Card Reader Driver) and D.23 (HP 7970 (A or B) Magnetic Tape Unit) also require non-buffered IOC when used without DMA.

12. PCS prints IOC and the memory bounds and then asks for Equipment Table entries by printing....................*TABLE ENTRY EQT? and halts.

Press RUN. Then type in the required EQT entries, one per line (each entry followed by *return* and *linefeed*). Remember that the entries are implicitly assigned octal numbers, starting with 7_8, as they are entered.. *xx*,D.*yy*[,D[,U1]]

NOTE: Elements in brackets "[ ]" are omitted according to the driver requirements.

where

xx = high priority select code of the device
D.yy = driver identification number (see chart).
D = uses DMA; omit if device does not use DMA.
U1 = file protect mode for mass storage device; omit if file protect is not desired.

Terminate the EQT by typing.../E

13. PCS asks for the Standard Unit Table.................................SQT?
and requests octal EQT entry numbers (7,10,11,...) for the following standard functions:

1. Keyboard input.. -KYBD?
2. Teleprinter... -TTY?
3. Library subroutine input at load-time.................. -LIB?
4. Punch output...-PUNCH?
5. Standard input...-INPUT?
6. Standard list output................................... -LIST?

Respond to each request by typing the EQT entry number of the device that is most appropriate for the specific function.

14. PCS requests information about the availability of Direct Memory Access...DMA?

Respond by typing 0 (no DMA), 6 (one channel DMA), or 6,7 (two channel DMA)

15. PCS halts after typing...*LOAD

Place the BCS Relocating Loader in the reader and press RUN.

16. PCS loads the Relocating Loader, then prints.......................LOADR
and the loader's memory bounds.................................*xxxxx yyyyy*
PCS then asks for the Interrupt Linkage Table
by printing.......................................INTERRUPT LINKAGE?
and halts.

Press RUN. Type the Interrupt Linkage Table entries for each device, one per line, in order of ascending select codes.

For a device using only one select code (I/O channel) type.....*xx*,*yy*,I.*zz*

where

> *xx* = select code of the device. (Lower numbered of two select codes if device is mass storage.)
>
> *yy* = octal address of interrupt linkage memory word for the device.
>
> *zz* = driver identification number (see Table BCS-1).

Example: 10,16,I.00

For a mass storage device using two select codes (I/O channels)
type..*xx*,*yy*,I.*zz*
qq,*rr*,C.*zz*

where

> *qq* = the lower priority (higher numbered) select code (*xx* = higher priority, lower numbered select code).
>
> *rr* = octal address of the interrupt linkage memory word for the device (different from *yy*).
>
> *zz* = driver identification number (same as for I.*zz*)

Example: 11,17,I.21
12,20,C.21

To put an octal instruction (i.e., a precautionary halt instruction) in an unused interrupt location (select code) type.............*xx*,*bbbbbb*

where

> *xx* = select code
>
> *bbbbbb* = an octal instruction (*b* = 0-7).

Example: $15,\underbrace{1020}_{\text{halt instruction}}\overbrace{55}^{\text{halt number}}$

PCS checks each entry after it is typed. If the driver name was typed incorrectly, PCS types.................................*ERROR

If the driver was not loaded earlier (step 11) then PCS types...*UN *name*

In either of the above cases, refer to Procedure 2 to continue.

Terminate the Equipment Table by typing................................/E

17. PCS determines whether there are any undefined references (e.g., to drivers that were not loaded). If none, PCS goes on to the next step.

    If some symbols are undefined, PCS prints............*UNDEFINED SYMBOLS: followed by a list of entry points for drivers which have been referenced in tables but not loaded..................... I.*xx*
    :

    If the drivers were not loaded during step 11 but should have been, restart PCS from step 1. To leave the references unresolved and load in the driver tapes at load-time, (Procedure 3) continue PCS processing with step 18.

    NOTE: Drivers that use DMA or entry point IOERR in the loader cannot be left undefined (must be loaded during step 11).

18. PCS lists the entry points of BCS and prints the system linkage area...*SYSTEM LINK
    xxxxx yyyyy

19. PCS then prints.....................................*BCS ABSOLUTE OUTPUT and halts.
    Check that the tape punch is operable and press RUN.
    PCS punches a configured BCS tape and halts.
    To punch additional copies, set switch register bit 15 on and press RUN.

20. Terminate PCS by setting all switch register bits to zero and pressing RUN. PCS prints...*END and halts.

PROCEDURE 2
PREPARE CONTROL SYSTEM ERROR HALTS AND MESSAGES

Message	Meaning	Action
*EOT	End-of-tape	Place next tape in tape reader and press RUN to continue loading.
*ERROR	A non-numeric parameter or illegal numeric parameter has been entered.	Retype the entire entry correctly.
I/O DRIVER? D.*rr*	A driver has been named in EQT entry but has not been loaded.	1. If the driver is to be loaded with user's program at load-time, type an exclamation mark (!). The driver name is added to the loader's LST. 2. If the driver should have been loaded, restart PCS.
*LØ1	Checksum error	To reread record, reposition tape to beginning of record and press RUN. If computer halts again, tape must be replaced.
*LØ2	Illegal record read: The last record read was not recognized as a valid relocatable format record.	To reread record, reposition tape to beginning of record and press RUN. If computer halts again, tape must be replaced.
*LØ3	Memory overflow: The length of BCS exceeds available memory.	Abort PCS. Reduce the number of core resident I/O drivers or increase memory.
*LØ4	System linkage area overflow in Base Page.	Abort PCS. Reduce the number of, or reorder the core resident I/O drivers.
*LØ5	Loader symbol table overflow: The number of EXT/ENT symbols exceeds available memory.	Abort PCS. Reduce the number of, or reorder the core resident I/O drivers.
*LØ6	PCS interprets the program length of BCS to be zero.	Abort PCS.
*LØ7	Duplicate entry points; an entry point in the current program matches a previously loaded entry point.	Eliminate an entry point. Check to see if the same program was loaded twice.

Message	Meaning	Action
*UNDEFINED SYMBOL: *symbol*	An entry point in a BCS module cannot be located.	If the subroutine should have been loaded, rerun PCS.
*UN *name*	The name I.*ee* is not defined as an entry point in any I/O driver previously loaded.	1. If the driver name was typed incorrectly, retype the entire entry correctly. 2. If the driver is to be loaded with the user's program at load-time, type an exclamation mark (!).

Halt Code		
1Ø2Ø55	A line is about to be printed on the teleprinter.	Turn punch unit OFF. Press RUN.
1Ø2Ø56	A line has been printed while the teleprinter punch unit was off.	Turn punch unit ON. Press RUN.
1Ø2Ø66	Tape supply low on tape punch which is producing absolute binary output. Trailer follows last valid output.	Place new reel of tape in unit. Press RUN. Leader is punched.
1Ø2Ø77	BCS tape is punched.	To produce additional copies, set switch 15 on.

PROCEDURE 3
HOW TO USE BCS TO RELOCATE AND RUN PROGRAMS

BCS performs two main functions: 1) relocates and links subroutines to main programs, and 2) executes programs.

Starting with relocatable code produced by an assembler or compiler, there are two possible methods to accomplish function 1 and reach function 2:

a. BCS relocates the code (including subroutines) into core memory directly and then executes it.

b. BCS relocates the code (including subroutines) and punches it onto an absolute tape along with the necessary system routines, drivers, tables, etc. This absolute tape can then be loaded into core through BBL or BBDL and executed.

Method (a) is faster, but does not provide a permanent, runnable copy of the program. Not only does the program code have to be relocated each time the program is to be run, but less core is available because the Relocating Loader occupies a part of memory.

Method (b) takes longer the first time, but provides a permanent copy of the program that can be executed. Also, more core is available since the program can (at run-time) use the space occupied by the Relocating Loader at load-time.

OPERATING INSTRUCTIONS

1. Load a configured BCS into core with BBL or BBDL. (See Procedure 1 for generation of a configured BCS.)
2. Set a starting address of 2_8.

3. Set all switch register bits off, then select the following options:

   Bit 15 on (suppress memory allocation listing)
   off (include memory allocation listing)

   Bit 14 on (punch absolute tape copy of program)
   off (relocate into core, do not punch tape)

   If Bit 14 on and a teleprinter is to be used for punching, then

   Bit 13 on (teleprinter is a 2754B and can print and punch separately; set teleprinter mode to KT)

   off (teleprinter cannot print and punch separately; BCS halts before and after each line of printing so that the operator can turn on/off punch unit to avoid punching list output, then punch the absolute binary output).

4. Place the first relocatable program tape into the reader. Press PRESET and RUN. BCS reads and relocates the binary code on the tape. If switch register bit 14 is on, an absolute binary tape is punched. (Otherwise, BCS relocates the program in memory.)

5. BCS halts after typing...*LOAD

   Load the user relocatable tapes as follows: Set switch register bits 2-0 off.

   Place the tape in the reader. Set switch register bit 15 on (if desired) to suppress memory allocation listing. Press RUN. When tape has been read, BCS halts after printing...*LOAD

   If there are more user tapes to load, repeat step 5.

6. After all user program tapes have been loaded, there are several options:

   To read a library subroutine tape (and load only those subroutines which are necessary to resolve externals). (Step 7)

To list undefined externals (or bypass further loading if there are no undefined externals). (Step 8)

To bypass further loading even if undefined externals remain. (Step 9)

7. Set switch register bit 2 on (bits 1 and 0 off). Place the relocatable library tape in the reader (FORTRAN IV library must be loaded first). Set switch register bit 15 on to suppress the memory allocation listing, if desired. Press RUN.

When the tape has been read, BCS halts after indicating:

```
No undefined externals.......................................*LST
(Set switch register bit 2 off
and go to step 10.)

or
                                                              ⎧ symbol
Undefined externals.................................         ⎨ symbol
                                                              ⎪    :
                                                              ⎩ symbol
                                                                 *LOAD
```

Return to Step 6 and select an option.

8. Set switch register bit 0 on (bits 1 and 2 off). Press RUN. BCS indicates whether undefined externals exist by printing either:

```
No undefined externals.......................................*LST
(Set switch register bit 2 off
and go to Step 10)

or
                                                              ⎧ symbol
Undefined externals.................................         ⎨ symbol
                                                              ⎪    :
                                                              ⎩ symbol
                                                                 *LOAD
```

Return to Step 6.

9. Set switch register bit 1 on (bits 2 and 0 off). Press RUN. BCS goes on to Step 10, even though undefined externals may still exist.

10. BCS has completed loading and is ready to print the Loader Symbol Table (LST), common bounds, and linkage area bounds. Set switch register bit 15 on to suppress listing of these items. Set bit 15 off to list them.

    If a 2754B Teleprinter is used, set the mode switch to "T" to enable the tape punch.

    Press RUN.

11. BCS completes listing (if requested by bit 15).

    If the program was relocated into core (bit 14 off),
    BCS prints..*RUN.
    and halts.

    Press RUN to execute the program.

12. If the program was punched onto paper tape (bit 14 on),
    BCS prints..*END
    and halts.

13. Tear off the absolute tape output and wind. To execute the program:

    Load the tape with BBL or BBDL.

    Start the program at location 2_8.

PROCEDURE 4
BCS LOAD-TIME ERROR HALTS AND MESSAGES

Message	Meaning	Action
*LØ1	Checksum error.	To reread record, reposition tape to beginning of record and press RUN. If computer halts again, tape must be replaced.
*LØ2	Illegal record read: The last record read was not recognized as a valid relocatable record tape.	To reread record, reposition tape to beginning of record and press RUN. If computer halts again, tape must be replaced.
*LØ3	Memory overflow: The length of BCS exceeds available memory.	Abort load. Reduce program size or increase memory.
*LØ4	System linkage area overflow in Base Page.	Abort load. Reduce program size or alter subprogram loading sequence.
*LØ5	Loader symbol table overflow: The number of EXT/ENT symbols exceeds available memory.	Abort load. Reduce program size or increase memory.
*LØ6	Common block error: The length of the common block in the current program is greater than the length of the first common block allocated.	Abort load. Reorder the programs during loading or make the common blocks the same length.
*LØ7	Duplicate entry points: An entry point in the current program matches a previously declared entry point.	Abort load. Eliminate an entry point. Check to see if the same program was loaded twice.
*LØ8	No transfer address: The initial starting location was not present in any of the programs which were loaded.	Load the absolute starting address into the A-register. Start program execution.

Message	Meaning	Action
*LØ9	Record out of sequence: A NAM record was encountered before the previous program was terminated with an END record.	1. Reload the program. 2. If program does not load properly, replace the binary tape for the program being loaded.

Error Code	Meaning	Action
1Ø2Ø55	A line is about to be printed on the teleprinter.	Turn punch unit OFF. Press RUN.
1Ø2Ø56	A line has just been printed on the teleprinter with the tape punch OFF.	Turn punch unit ON. Press RUN.
1Ø2Ø66	Tape supply low on tape punch which is producing absolute binary output. Trailer follows last valid output.	Place new reel of tape in unit. Press RUN. Leader is punched.

PROCEDURE 5

BCS RUN-TIME HALTS AND MESSAGES

Certain library routines, including the Formatter, produce error messages at run-time.

Halt Code	Meaning
1Ø6Ø55	Program has attempted to execute a non-program area of core. Warning-only. Program can be restarted.

SOFTWARE MANUAL CHANGES

BASIC CONTROL SYSTEM

(5951-1391)
Dated August 1971

Updates to this publication as well as changes to the Manual Change Sheet itself are listed below.

November 1972

Change Number	Page	Instructions
1	BCS-4	In step 7, change the reply to HS INP? to "Reply with the select code of the high-speed input unit for PCS or if the teleprinter reader is used, enter zero."
2	BCS-4	In step 8, change the reply to HS PUN? to "Reply with the select code of the high-speed punch or if the teleprinter punch is used, enter zero."