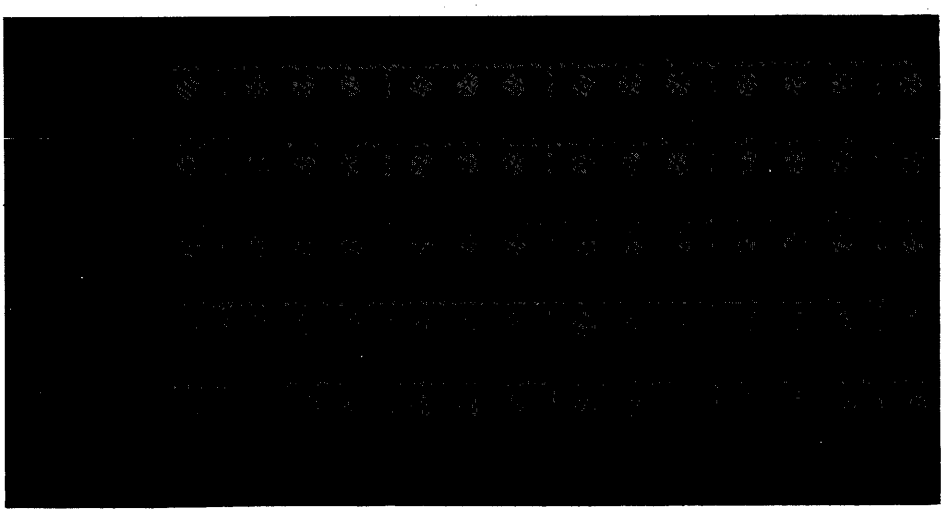
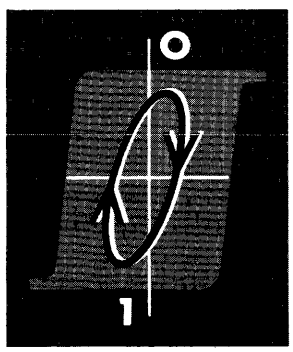
2116C COMPUTER

HEWLETT-PACKARD





SPECIFICATIONS AND BASIC OPERATION

VOLUME 1

CERTIFICATION

The Hewlett-Packard Company certifies that this instrument was thoroughly tested and inspected and found to meet its published specifications when it was shipped from the factory. The Hewlett-Packard Company further certifies that its calibration measurements are traceable to the U.S. National Bureau of Standards to the extent allowed by the Bureau's calibration facility.

WARRANTY AND ASSISTANCE

All Hewlett-Packard products are warranted against defects in materials and workmanship. This warranty applies for one year from the date of delivery, or, in the case of certain major components listed in the operating manual, for the specified period. We will repair or replace products which prove to be defective during the warranty period provided they are returned to Hewlett-Packard. No other warranty is expressed or implied. We are not liable for consequential damages.

Service contracts or customer assistance agreements are available for Hewlett-Packard products that require maintenance and repair on-site.

For any assistance, contact your nearest Hewlett-Packard Sales and Service Office. Addresses are provided at the back of this manual.

HEWLETT PACKARD

VOLUME ONE

SPECIFICATIONS AND BASIC OPERATION MANUAL

MODEL 2116C COMPUTER

All Serial Numbers

Note

This volume applies to all HP Model 2116C Computers. Any differences for specific serial numbered computers are described in Volume Two, Installation and Maintenance manual, or in supplements to that manual. To order additional copies of this manual, specify part number 02116-91755.

02116-91755

Printed: JULY 1971

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

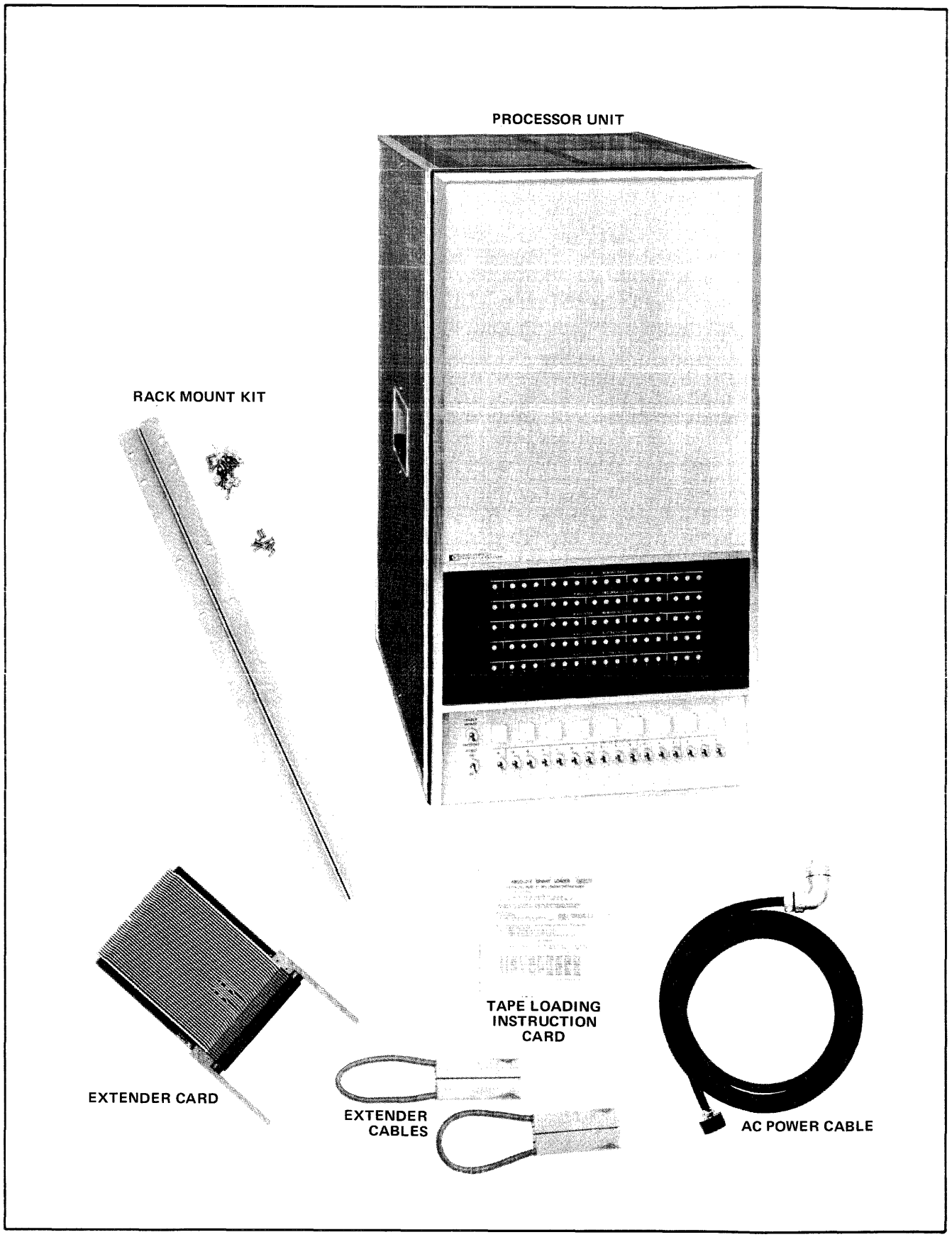


Figure 1-1. Basic HP 2116C Computer and Accessories

SECTION I

DOCUMENTATION

1-1. BASIC COMPUTER MANUALS.

1-2. Documentation supplied with the Hewlett-Packard 2116C Computer (shown in figure 1-1) consists of four volumes which are described briefly in paragraphs 1-3 through 1-10. The four volumes and other system manuals are placed in binders as illustrated in figure 1-2. A manual index is supplied at the front of the first binder; refer to this index for locations of all manuals within the binders. Immediately following the index is a hardware system installation record that defines the system configuration as originally shipped.

1-3. SPECIFICATIONS AND BASIC OPERATION MANUAL.

1-4. Volume One gives specifications and basic operating information for the computer. The following sections of information are included:

a. Specifications: Descriptions and capabilities of computer hardware and software and a table of specifications are given in this section.

b. Fundamental Theory of Operation: This section provides basic theory of operation for the computer. For detailed theory of operation, refer to the Installation and Maintenance Manual, Volume Two.

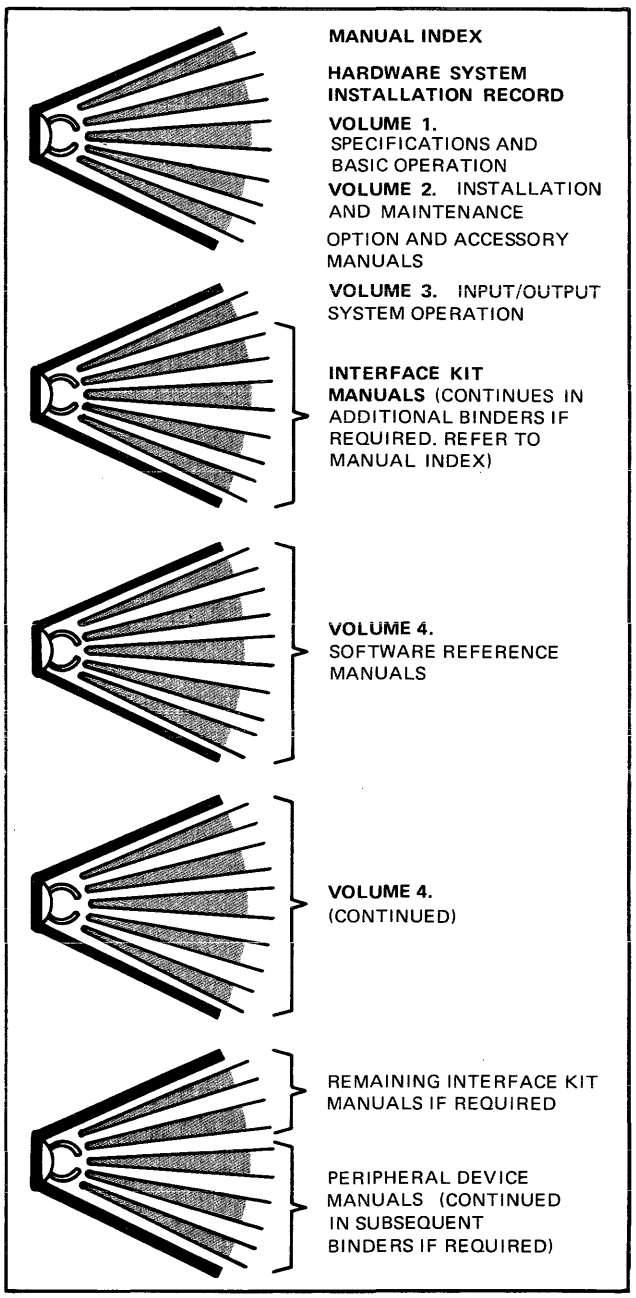
c. Operating the Computer: This section provides front-panel operating information and basic machine language programming information for the computer.

1-5. INSTALLATION AND MAINTENANCE MANUAL.

1-6. Volume Two gives instructions for installation and maintenance of the computer. Contents of this volume are as follows:

a. General Information: This section contains information for users who require a knowledge of the physical makeup of the computer and an understanding of its maintenance features. Included is a description of the various electronic assemblies which comprise the computer, an explanation of controls and indicators, a description of identification numbers used in the computer, a description of standard accessory equipment supplied with the computer, an explanation of the principal built-in maintenance features, and a list of recommended servicing equipment.

b. Installation: This section describes unpacking procedures, provides primary power data, explains initial inspection procedures, and gives instructions for installing the computer.



2123-10

Figure 1-2. Typical Computer System Documentation

c. Theory of Operation: This section describes the circuit theory of the control, arithmetic, memory, input/output, and power supply sections.

d. Troubleshooting: This section gives step-by-step procedures for testing the computer.

e. Maintenance: This section provides preventive maintenance instructions and adjustment information.

f. Replaceable Parts: This section provides lists and illustrations of replaceable parts.

g. Diagrams: This section consists of logic diagrams, parts location diagrams, wiring information and logic equations. Also included are definitions of the abbreviations used for signal names and information about integrated circuits.

1-7. INPUT/OUTPUT SYSTEM OPERATION MANUAL.

1-8. Volume Three describes the structure of the computer input/output system. Operation of the priority interrupt system and typical interface card circuits are included. Volume Three contains the following sections of information:

a. General Information: This section defines the input/output section of the computer system. Information about computer power supply capabilities and current requirements of options and accessories is provided.

b. The Input/Output System: This section contains principles of circuit operation for the input/output system. Input/output extender units and interface kits are also described.

c. Priority Interrupt System: Operation of the priority interrupt system, including examples of typical interface circuits, is described in this section. A timing diagram shows relationships of signals discussed in the text.

1-9. SOFTWARE REFERENCE MANUALS.

1-10. Volume Four consists of one or more three-ring binders containing documentation for each item of software supplied with the computer. Both standard software programs and software specially originated for an individual user are fully described as to specifications and usage. A Software System Installation Record at the front of the Volume Four binder lists all software furnished with the original shipment, and provides an index to the supporting documents in Volume Four. Space is provided for noting changes and additions, so that an up-to-date record can be maintained by the user. Printed listings for standard programs furnished with the system are supplied as manual supplements in this volume. Reference manuals normally included in volume four are:

a. HP Assembler
b. HP Symbolic Editor
c. HP Basic Control System
d. HP FORTRAN
e. HP Relocatable Subroutines
f. HP ALGOL
g. HP BASIC
h. HP Software Operating Procedures
i. HP Manual of Diagnostics
j. HP Small Programs

1-11. IDENTIFICATION.

1-12. Each computer is identified by a serial number on the rear panel (for example 1001A00600 or 949-00599). The first group of digits makes up a serial number prefix used to document equipment changes. This prefix does not change unless changes to the equipment have been made. The last five digits form a serial number to identify each piece of equipment. The serial number prefix may be either three or four digits in length. If the prefix contains four digits, a code letter will be stamped between the prefix and the serial number indicating the country in which the equipment was manufactured. If the serial number prefix on your equipment does not agree with that shown on the title page of the hardware manuals, there are differences between your equipment and the equipment described in the manuals. These differences are described in manual supplements included with the manual or available at the nearest HP Sales and Service Office.

SECTION II

SPECIFICATIONS

2-1. DEFINITION OF COMPUTER SYSTEM.

2-2. BASIC UNIT DESCRIPTION. The Hewlett-Packard Model 2116C Computer is a general-purpose digital computer designed to add computation capability to Hewlett-Packard data measuring and recording systems. The logical design and software follow conventional standards of computer usage and notation so that the HP 2116C may also be used as a free-standing unit or in other types of systems, such as process control, media conversion, data reduction, or communication systems. The hardware and software are specially designed to permit interfacing of real-time devices (i.e., devices running asynchronously with respect to a program being run). The word length is 16 bits. The basic computer includes the processor unit (main frame) with an 8192-word memory. All specifications in this section apply to the basic unit only, unless specifically denoted as an option specification.

2-3. OPTIONS. Options for the HP 2116C Computer are of two general types:

a. Processor Options. These options extend the memory and computation capabilities of the basic unit.

b. Input/Output Options. These options add input and/or output facilities to the basic computer. The option, identified by an interface kit number (e.g., HP 12531C), provides the circuitry, cabling, and software to enable the computer to operate with a specific input or output instrument (measuring, reading, or recording device), or with a series of instruments. Compatible instruments, not included in the interface kit, are separately available from Hewlett-Packard. When external devices are connected to the computer, the computer becomes part of a system.

2-4. SYSTEMS. Two general types of computer systems are available from Hewlett-Packard:

a. HP 2116C Computer Systems. Systems may be configured to individual requirements using combinations of standard input/output options. The software packages which are hardware dependent (Basic Control System and System Input/Output, see table 2-3) will be made up in accordance with the hardware system configuration.

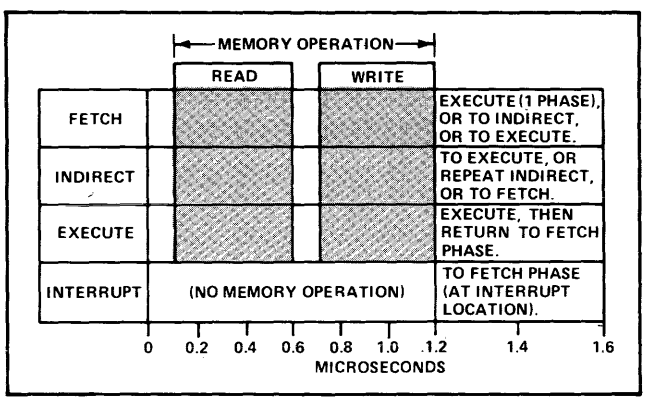
b. Data Acquisition Systems. Systems are available in standard configurations which combine Hewlett-Packard digital scanning, measuring, and recording equipment with the computer. In these systems the computer is programmed to exercise partial or complete control over the data taking process and to perform computations on data measured by the system. A data acquisition program is furnished with these systems. Capabilities of available instruments include measurements of ac or dc voltages, resistances, frequencies, time periods, temperatures, gas pressures, nuclear events, etc., from multiple inputs. (The functions of some instruments such as linearizers, comparators, scanner programmers, and output couplers are present in the basic computer, or may be accomplished with options or programming.)

2-5. TECHNICAL SPECIFICATIONS.

2-6. Table 2-1 lists technical specifications for the computer.

2-7. MACHINE TIMING.

2-8. An internal 10-MHz timing generator automatically generates read/write memory cycles every 1.6 microseconds (see figure 2-1). The basic computer has four machine phases (fetch, indirect, execute, interrupt), of which the first three include a memory cycle. Phases do not occur in a fixed sequence, but rather are determined by conditions which occur during operation. The computer can go directly from one of the first three phases to certain others in the manner indicated in figure 2-1, and an external device can cause the computer to go into the interrupt phase on completion of any current phase. The fetch phase may be thought of as the normal or "home" condition; the processing of each instruction begins with a fetch phase, and in many cases is fully executed within that phase. Each phase takes 1.6 microseconds with one exception: the execute phase of the ISZ instruction (Increment, and Skip if Zero) takes 2.0 microseconds.



2106-3

Figure 2-1. Machine Timing

2-9. FETCH PHASE. The contents of the currently-addressed memory cell is read into the T-register during the read portion of the memory cycle, and written back into the memory cell during the write portion of the memory cycle. The information left in the T-register is taken as an instruction when read during the fetch phase. If the instruc-

Table 2-1. Specifications

MEMORY

Type: Magnetic core
Size: 8192 16-bit words (expandable to 16, 24, or 32K)
Page Size: 1024 words
Direct Addressing: Current page and Base page
Indirect Addressing: All pages
Speed: 1.6 microsecond cycle time
Loader Protection: "Protected" switch disables last 64 locations

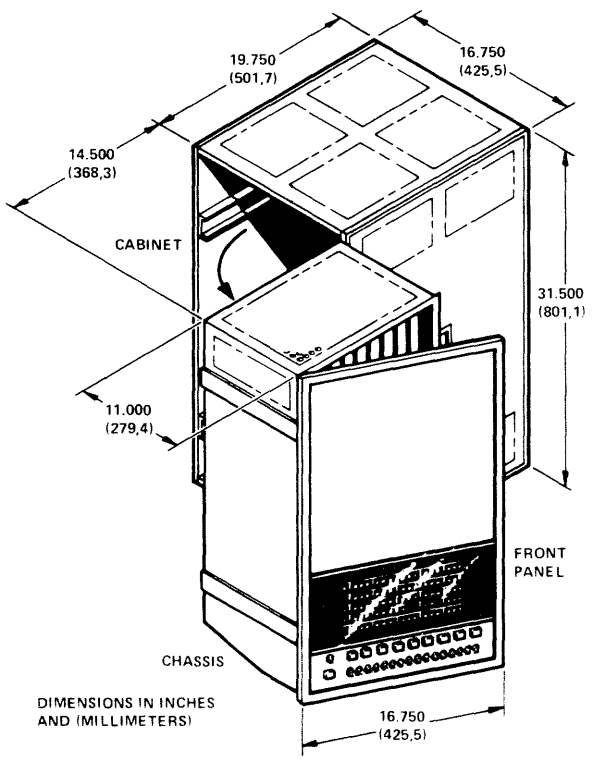
ARITHMETIC

Parallel, two's complement binary

COMPUTE SPEED

	Microseconds	With Extended Arithmetic Unit	
Add	3.2	3.2	(1 instruction)
Subtract	4.8	4.8	(2 instructions)
Multiply	112*	19.2	(1 instruction)
Divide	350*	20.8	(1 instruction)
Floating Point Add	487*	302*	
Floating Point Subtract	468*	286*	
Floating Point Multiply	803*	326*	
Floating Point Divide	1590*	350*	

*Subroutine — time approximate



INPUT/OUTPUT

Number of mainframe channels: 16
Total channels with extender: 32 or 48
Channel capacity: 16 bits, parallel transfer
Service Method: priority interrupt hardware, standard
Priority assignment: by position of interface card

VENTILATION

Intake on sides and back at bottom, exhuast at top. Air flow 600 cfm. Heat dissipation 5500 BTU/hr.

INSTALLATION

For use on table, or mounted in standard 19-inch rack, using adapters furnished (panel height 31-1/2 inches). Requires no special wiring, subflooring, or other special installation preparations.

WEIGHT

Net 230 lb (104 kg)
Shipping 330 lb (150 kg)

POWER REQUIRED

Source: 115V (230V with option 015) ±10%, 50 to 60 Hz
Consumption: 1000W min to 1600W max, depending on number of I/O interfaces installed.

ENVIRONMENTAL CONDITIONS

From 0° to +55°C (+32° to +131°F)
Relative humidity to 95% at 40°C

OPTIONS

Power Fail: Has highest interrupt priority. Can be used to automatically save register contents and program location if power fails. Optionally halts or resumes execution of the interrupted program when power is restored.

Direct Memory Access: Provides two 16-bit channels, program assignable to any mainframe channel. Maximum transfer rate is 263,000 words per second. Uses one memory cycle per transfer. Block transfers up to 16,384 words.

Parity Check: Verifies parity for all words transferred in or out of memory. Provides front panel indication of error and transfers control to interrupt routine.

Memory Protect: Permits any amount of memory to be "fenced". Inhibits any instruction that attempts to change a protected location and makes address of offending instruction available for software interrogation. Inhibits I/O instructions outside protected area.

Extended Arithmetic Unit: Permits integer multiply, integer divide, double load, double store, long shifts, and long rotations to be performed by hardware rather than program subroutines.

tion includes an indirect address bit (see paragraph 2-19), the computer sets the indirect phase condition; if the instruction does not have an indirect address bit but does include a memory reference (two-phase instruction), the computer sets the execute phase condition. Otherwise the current instruction is fully executed at the end of the fetch phase, and the computer remains in the fetch state for the next memory cycle. An exception to these conditions is the JMP (jump) instruction, which is a memory reference instruction but does not require an execute phase; the computer executes the instruction at the end of the fetch phase or the indirect phase, and then sets the fetch phase again for the next memory cycle.

2-10. INDIRECT PHASE. The contents of the memory cell referenced during the fetch phase is read into the T-register and the entire 16-bit word (15 bits of address, plus a new direct/indirect bit) is taken as a new memory reference for the same instruction. Using 15 bits for an address permits addressing any location in memory. If the direct/indirect bit again specifies indirect addressing, the computer remains in the indirect state and reads another 16-bit address word out of memory as a continuation of multiple-step indirect addressing. If the direct/indirect bit specifies direct addressing, the computer sets the execute phase (or, in the case of a jump indirect, the fetch phase).

2-11. EXECUTE PHASE. The 16-bit data word in the memory cell referenced during a fetch phase or an indirect phase is read into the T-register and is operated on by the current instruction (retained from the fetch phase) at the end of the execute phase. At the end of this phase, the computer sets the fetch phase again to read the next instruction.

2-12. INTERRUPT PHASE. An input/output device requesting service at any time during one of the phases is acknowledged at the end of that phase, unless the interrupt is inhibited for any reason by the program being run. The computer then goes into the interrupt phase, which does not have a memory cycle. During this phase the P-register is decremented, so that no instruction in the main program will be skipped or executed twice. At the end of this phase, the interrupt address of the interrupting device is transferred into the M-register, and the fetch phase is set, causing the instruction contained in the interrupt address location to be read. The interrupt phase cannot occur again until at least this instruction is completed.

2-13. MEMORY.

2-14. TYPE.

2-15. The computer uses a ferrite core storage module capable of storing 8196 (8K) 17-bit words. A computer word requires 16 bits and one bit is reserved for the memory parity option available for the computer. The memory module is mounted on a plug-in card that is inserted in the memory section of the computer card cage. Plug-in card slots are provided in the computer main frame for up to three additional memory modules which will expand the memory storage capacity to 32,768 (32K) words.

2-16. ADDRESSING.

2-17. GENERAL. The 8K-word module is logically divided into two 4K sections. These sections are referred to as the lower 4K and the upper 4K. Each 4K section is further divided into four pages. A page is defined as the largest block of memory which can be addressed by the memory address bits of a memory reference instruction. Memory reference instructions have ten bits to specify a memory address; thus, the page size is 1024 (2000 octal) locations. Octal addresses of the eight pages of the basic 8K module are therefore:

Lower 4K	page 0	00000 to 01777
	page 1	02000 to 03777
	page 2	04000 to 05777
	page 3	06000 to 07777
Upper 4K	page 4	10000 to 11777
	page 5	12000 to 13777
	page 6	14000 to 15777
	page 7	16000 to 17777

Dividing the memory into pages is useful for programming purposes only. Page divisions bear no relationship to the physical makeup of the computer core memory.

2-18. ZERO/CURRENT PAGE. For direct addressing purposes, generally only two pages are of interest: page zero (the base page, consisting of locations 00000 through 01777), and the current page (the page in which the instruction itself is located). All memory reference instructions include a bit (bit 10) reserved to specify one or the other of these two pages. To address locations in any other page, indirect addressing is used. Page references for direct addressing of memory reference instructions are specified by bit 10 as follows: 0 = page zero (Z), 1 = current page (C).

2-19. DIRECT/INDIRECT. All memory reference instructions use bit 15 to specify direct or indirect addressing. Direct addressing combines the instruction code and the effective address into one word, permitting a memory reference instruction to be executed in two machine phases (fetch and execute). Indirect addressing uses the address part of the instruction word to access another word in memory which is taken as a new memory reference for the same instruction. This new address word is a full 16 bits long, 15 bits of address plus another direct/indirect bit. The 15-bit length of the address permits access to any location in any module. If bit 15 again specifies indirect addressing, still another address is obtained; this multiple-step indirect addressing may be done to any number of levels. The first address obtained that does not specify another indirect level becomes the effective address for the instruction. Instructions with indirect addresses are therefore executed in a minimum of three machine phases (fetch, indirect,

execute). Direct or indirect addressing is specified by bit 15 as follows: 0 = direct, 1 = indirect.

2-20. RESERVED LOCATIONS. The first 64 memory locations of the base page (octal addresses 00000 through 00077) are reserved as listed below. The first two addresses are A and B flip-flop registers and not core storage locations. Locations 4 through 77 are reserved in the sense that interrupt wiring is present for the priority order given. As long as the locations do not have actual interrupt assignments (as determined by the input/output devices included in the user's system), these locations may be used for normal program purposes.

00000	Address of A-register
00001	Address of B-register
00002 00003	For exit sequence if A and B contents are used as executable words
00004	Interrupt location, highest priority (reserved for power fail interrupt)
00005	Reserved for memory parity interrupt and memory protect interrupt
00006 00007	(Reserved for DMA interrupt)
00010 thru 00077	Interrupt locations in decreasing order of priority

2-21. LOADER PROTECTION.

2-22. The last 64 locations of memory (octal addresses 17700 through 17777 in the standard computer) are reserved for the Basic Binary Loader. The Basic Binary Loader is a manually entered program that permits reading and storage of binary information from punched paper tape, as read by a punched tape reader or a teleprinter. Absolute addresses are required in the loaded data. A front-panel switch (LOADER), when set to PROTECTED, disables the Basic Binary Loader locations so that they can neither be used nor altered in any way. For entering the Basic Binary Loader manually into the computer and for actual loading of tapes, this switch must be set to ENABLED. The LOADER switch is effective for the last 64 locations of memory, regardless of memory size. Plug-in options which expand memory relocate the protected area automatically to the 64 highest numbered locations.

2-23. WORKING REGISTERS.

2-24. The computer has seven working registers and gives continuous display of the register contents by lights on the computer front panel. Five of these are 16-bit flip-flop registers, and two are 1-bit flip-flop registers indicated by panel lighting (on or off) of the register name.

2-25. [redacted] (MEMORY DATA). All data transferred into or out of memory is routed through the 16-bit T-register (Transfer Register). The T-register display therefore indicates exactly what information went into or out of a memory cell during the preceding memory cycle.

2-26. [redacted] (PROGRAM COUNTER). On completion of each instruction, the P-register indicates the address of the next instruction to be fetched from memory. The P-register automatically increments by one (or two, when executing a skip instruction) after the execution of each instruction. A jump instruction (JMP or JSB) can set the P-register to any core location number.

2-27. [redacted] (MEMORY ADDRESS). The M-register holds the address of the memory cell being read or written into. The M-register indication will differ from the P-register indication when multi-phase instructions are being processed, since the M-register will be changed by memory references in the instruction (which may be several in the case of indirect addressing) or by an interrupt, whereas the P-register remains constant until completion of the instruction.

2-28. [redacted] (ACCUMULATOR). The A-register is an accumulator, holding the results of arithmetic and logical operations performed by programmed instructions. This register may be addressed by any memory reference instruction as location 00000, thus permitting inter-register operations such as "add B to A", "compare B with A", etc., using a single-word instruction.

2-29. [redacted] (ACCUMULATOR). The B-register is a second accumulator, which can hold the results of arithmetic and logical operations completely independent of the A-register. The B-register may be addressed by any memory reference instruction as location 00001 for inter-register operations with A.

2-30. EXTEND. The Extend bit is a one-bit register, used to link the A- and B-registers by rotate instructions or to indicate a carry from bit 15 of the A- or B-registers by an add instruction (ADA, ADB) or increment instruction (INA or INB, but not ISZ). This is of significance primarily for multiple-precision arithmetic. The Extend bit is not complemented by a carry if already set, but may be cleared, complemented, or tested by program instruction. The Extend bit is set when the EXTEND panel light is on ("1") and clear when off ("0").

2-31. OVERFLOW. The Overflow bit is a one-bit register which indicates, if on, that an add instruction (ADA, ADB) or an increment instruction (INA or INB, but not ISZ) referencing the A- or B-registers has caused one of these accumulators to exceed the maximum positive or negative number which can be contained (+32767 or -32768, decimal). This condition is implied by a carry (or lack of carry) from bit 14 to bit 15. By program instructions, the Overflow bit may be cleared, set, or tested. The OVERFLOW panel light remains on until the bit is cleared by an instruction and is not complemented if a second overflow occurs before being cleared. It will not be set by shift or rotate instructions.

2-32. PANEL CONTROLS.

2-33. SWITCH REGISTER. Sixteen toggle switches to enter manually set information into the computer. The setting of the Switch Register (up is "one", down is "zero") may be transferred into the computer in the following ways.

a. By program, may be loaded into the A- or B-register using LIA or LIB instructions.

b. By program, may be merged (inclusive "or") into the A- or B-register using MIA or MIB respectively.

c. Manually, using LOAD ADDRESS switch, may be loaded into the P- and M-registers (simultaneously), thus directing the computer to a specific memory cell.

d. Manually, using LOAD MEMORY switch, may be entered into the memory cell specified by the M-register, thus permitting the user to change the contents of any memory cell.

e. Manually, using LOAD A or LOAD B switches, may be loaded into the A- or B-registers.

2-34. POWER. Toggle switch to turn computer power on or off. Regulated power automatically goes off in case of abnormal changes in internal power supplies. Contents of memory are not affected by switching power off and on; contents of working registers, however, are lost when power goes off (contents random following turn-on). Note: Some units are supplied with a push on/push off switch; switch is lit when regulated power is on.

2-35. PROTECT. Toggle switch associated with the last 64 locations of memory; for example, octal addresses 17700 through 17777 in the basic computer. These locations are normally occupied by the Basic Binary Loader. In the ENABLED position, this block of memory can be read or loaded; in the PROTECTED position, this block is disabled.

2-36. PRESET. Momentary switch to preset the computer to the fetch phase, to turn off the interrupt system and all input/output control bits, to set all input/output flag bits, and to reset the parity halt light located on the computer front panel. It also clears the power fail interrupt circuits. An internal preset pulse accomplishing the same functions is generated each time POWER is switched on.

2-37. RUN. Momentary switch to start operation at the current state of the computer. Switch is lit when a program is running, and goes off when HALT is pressed, when HLT instruction is executed, when parity error occurs, or when an abnormal change occurs in the internal power supplies. When the RUN light is on, all front-panel control switches except HALT, POWER and LOADER are disabled.

2-38. HALT. Momentary switch to stop computer operation at the end of the current phase. When the computer is halted, the HALT switch is lit, and all front-panel control switches are enabled.

2-39. LOAD MEMORY. Momentary switch to store the contents of the Switch Register into the memory location specified by the address in the M-register. P- and M-registers are automatically incremented after operation of the LOAD MEMORY switch, to simplify storing data into consecutive memory locations. The stored data remains displayed in the T-register, and the fetch phase is set at the end of the load operation.

2-40. LOAD A. Momentary switch to transfer the contents of the Switch Register into the A-register. The computer's phase status is not altered.

2-41. LOAD B. Momentary switch to transfer the contents of the Switch Register into the B-register. The computer's phase status is not altered.

2-42. LOAD ADDRESS. Momentary switch to transfer the contents of the Switch Register into both the P- and M-registers, thus directing the computer to the desired address. The fetch phase is set at the end of the load operation.

2-43. DISPLAY MEMORY. Momentary switch to display, in the T-register, the contents of the location specified by the address in the M-register. P- and M-registers are automatically incremented after operation of the DISPLAY MEMORY switch, so that consecutive memory locations may be displayed simply by repeated operation of this switch. (P- and M-registers are therefore one step ahead of the T-register display.) The fetch phase is set after incrementing of the P- and M-registers.

2-44. SINGLE CYCLE. Momentary switch to execute one machine phase each time the switch is pressed.

2-45. INSTRUCTIONS.

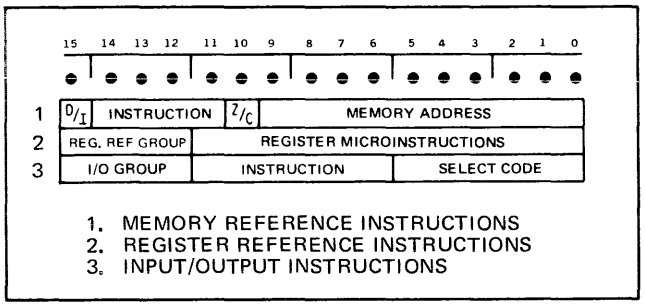
2-46. NUMBER.

2-47. The computer has 70 basic one-word instructions, all executable in 1.6 or 3.2 microseconds (except for ISZ, which is executable in 3.6 microseconds). These instructions are grouped in three types, described in paragraphs 2-53 through 2-97. Combinations of the register reference microinstructions, which are all one-word instructions executable in 1.6 microseconds, extend the total of different one-word instructions to over 1000.

2-48. FORMATS.

2-49. The three types of basic instructions are grouped according to the bit format of the instruction word. These types are: memory reference, register reference, and input/output instructions. A comparison of the three formats is given in figure 2-2, and detailed binary coding is included

with the instruction descriptions following. A Consolidated Coding Table appears in the appendix of this manual.



02116-A-3

Figure 2-2. Basic Instruction Formats

2-50. The first type comprises the memory reference instructions, using 10 bits (0 through 9) for a memory address, bit 10 to specify zero or current page, and bit 15 for direct or indirect addressing. This leaves four bits (14,13,12,11) to encode the 14 instruction commands in this group.

2-51. The other two types use four bits (15, 14, 13, 12) to distinguish the register reference and the input/output instructions. The register reference type uses bits 11 through 0 to combine up to eight "microinstructions" (i.e., instructions formed by only 1, 2, or 3 bits), with the resulting multiple instruction operating on the A-, B-, or E-registers as a single-word instruction. The input/output type uses bits 11 through 6 for a variety of input/output instructions, and bits 5 through 0 to make the instruction apply directly to one of 64 possible input/output devices or functions.

2-52. The following paragraphs, through 2-97, describe in detail each of the instructions in the three type groups.

Note

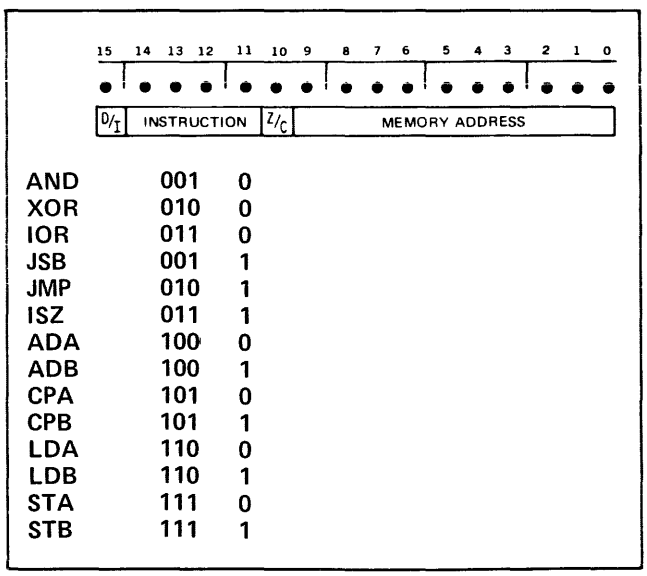
Functions of bits appearing in the form A/B, D/I, D/E, Z/C, or H/C throughout these specifications are invariably obtained by coding a 0 or 1 respectively (0/1). Thus, for example, A is specified by a zero-bit, and B by a one-bit.

2-53. MEMORY REFERENCE INSTRUCTIONS.

2-54. The 14 memory reference instructions execute some operation involving memory locations, such as transferring information into or out of a memory cell or checking the memory cell contents. The address of the referenced cell is determined by a combination of the ten memory address bits in the instruction word (0 through 9) and five bits (10 through 14) from the current indication of the P-register. This means that memory reference instructions can directly address any word in the current page. Additionally, if the instruction is not located on the base page (page zero), bit 10 of the instruction word doubles the addressing range to 2048 words by allowing selection of either page zero or current page (i.e., bits 10 through 14 of the address in the M-register can be reset to zero, instead of assuming the current indication of the P-register). This feature provides a convenient linkage between all pages of memory, since page zero can be reached directly from any other page.

2-55. Note that since the A- and B-registers can be addressed, any memory reference instruction can apply to either of these registers as well as to memory cells. For example, ADA 0001 means add the contents of the B-register (its address being 0001) to the A-register; specify page zero for this operation, since the B-register address is on page zero.

2-56. Figure 2-3 gives instruction codes and mnemonics for all 14 memory reference instructions. All memory reference instructions take a minimum of two machine phases (one to read the instruction word, and one to read the referenced memory cell), except JMP, which is a one-phase instruction. Logic truth tables, relating to the first three instructions described below, are given in table 2-2. Note that logic operations are performed on a bit-for-bit basis (i.e., no carries).



02116-A-4

Figure 2-3. Memory Reference Instructions

2-57. AND. "And" to A. The contents of the addressed location are logically "anded" to the contents of the A-register. The contents of the memory cell are left unaltered.

2-58. XOR. "Exclusive or" to A. The contents of the addressed location are combined with the contents of the A-register as an "exclusive or" logic operation. The contents of the memory cell are left unaltered.

Table 2-2. Logic Truth Tables

	AND	XOR	IOR
A Contents	0 0 1 1	0 0 1 1	0 0 1 1
Memory	0 1 0 1	0 1 0 1	0 1 0 1
Result (in A)	0 0 0 1	0 1 1 0	0 1 1 1
1 = True, 0 = False			

2-59. IOR. "Inclusive or" to A. The contents of the addressed location are combined with the contents of the A-register as an "inclusive or" logic operation. The contents of the memory cell are left unaltered.

2-60. JSB. Jump to Subroutine. This instruction, executed in location P, causes computer control to jump unconditionally to the memory location (X) specified in the address portion of the JSB instruction word. The contents of the P-register plus one (return address) is stored in location X, and the next instruction to be executed will be that contained in the next location (X + 1). A return to the main program sequence at P + 1 may be effected by a jump indirect through location X.

2-61. JMP. Jump. This instruction transfers control to the contents of the addressed location. That is, JMP causes the P- and M-registers to be set according to the memory address portion of the instruction word, thus addressing memory cell X, so that the next instruction will be read from location X.

2-62. ISZ. Increment, and Skip if Zero. An ISZ instruction adds one to the contents of the addressed memory location. If the result of this operation is zero, the next instruction is skipped; i.e., the P- and M-registers are advanced by two instead of one. Otherwise, the program proceeds normally to the next instruction in sequence. The incremented value is written back into the memory cell in either case, An ISZ instruction referencing locations zero or one (A- or B-registers) can not cause setting of the Extend or Overflow bits (unlike INA and INB).

2-63. ADA. Add to A. The contents of the addressed memory location are added to the contents of the A-register, and the sum remains in the A-register. The result of the addition may set the Extend or Overflow bits. The contents of the memory cell are unaltered.

2-64. ADB. Add to B. The contents of the addressed memory location are added to the contents of the B-register, and the sum remains in the B-register. Extend or Overflow bits may be set, as for ADA. The contents of the memory cell are unaltered.

2-65. CPA. Compare to A, skip if unequal. The contents of the addressed location are compared with the contents of the A-register. If the two 16-bit words are different, the next instruction is skipped; i.e., the P- and M-registers are advanced by two instead of one. If the words are identical, the program proceeds normally to the next instruction in sequence. The contents of neither the A-register nor the memory cell are altered.

2-66. CPB. Compare to B, and skip if unequal. Same as CPA, except comparison is made with B-register.

2-67. LDA. Load into A. The A-register is cleared and loaded with the contents of the addressed location. The contents of the memory cell are unaltered.

2-68. LDB. Load into B. The B-register is cleared and loaded with the contents of the addressed location. The contents of the memory cell are unaltered.

2-69. STA. Store A. The contents of the A-register are stored in the addressed location. The previous contents of the memory cell are lost; the A-register is unaltered.

2-70. STB. Store B. The contents of the B-register are stored in the addressed loccation. The previous contents of the memory cell are lost; the B-register is unaltered.

2-71. REGISTER REFERENCE INSTRUCTIONS.

2-72. The register reference instructions, in general, manipulate bits in the A-, B-, and E-registers. There is no reference to memory; thus these instructions are executed in only one machine phase. This type includes 39 basic instructions, which are combinable to form one-word multiple instructions that can operate in various ways on the contents of the A-, B-, or E-registers. These "microinstructions" are divided into two sub-groups, the shift-rotate group (SRG) and the alter-skip group (ASG). Three instructions (SLA, SLB, and CLE) appear in both groups and, being combinable in these different contexts, are counted twice in the total of basic instructions. Microinstructions may be combined under the following general rules:

a. Instructions from the two groups cannot be mixed.

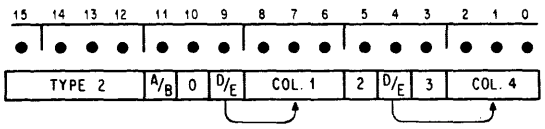
b. References to both A- and B-registers cannot be mixed.

c. Only one microinstruction can be chosen from each column of the selection tables in figures 2-4 and 2-5.

d. Use zeros to exclude unwanted microinstruction bits.

e. The sequence of execution is left to right in the selection tables (column 1, then column 2, etc.).

f. If two (or more) skip functions are combined, the skip will occur if either or both conditions are met. One exception exists: see RSS under paragraph 2-75.



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOP	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
CLE	0	0	0	0		0					1					
SLA	0	0	0	0	0	0							1			
SLB	0	0	0	0	1	0							1			
ALS	0	0	0	0	0	0	1	0	0	0		X		X	X	X
BLS	0	0	0	0	1	0	1	0	0	0		X		X	X	X
ARS	0	0	0	0	0	0	1	0	0	1		X		X	X	X
BRS	0	0	0	0	1	0	1	0	0	1		X		X	X	X
RAL	0	0	0	0	0	0	1	0	1	0		X		X	X	X
RBL	0	0	0	0	1	0	1	0	1	0		X		X	X	X
RAR	0	0	0	0	0	0	1	0	1	1		X		X	X	X
RBR	0	0	0	0	1	0	1	0	1	1		X		X	X	X
ALR	0	0	0	0	0	0	1	1	0	0		X		X	X	X
BLR	0	0	0	0	1	0	1	1	0	0		X		X	X	X
ERA	0	0	0	0	0	0	1	1	0	1		X		X	X	X
ERB	0	0	0	0	1	0	1	1	0	1		X		X	X	X
ELA	0	0	0	0	0	0	1	1	1	0		X		X	X	X
ELB	0	0	0	0	1	0	1	1	1	0		X		X	X	X
ALF	0	0	0	0	0	0	1	1	1	1		X		X	X	X
BLF	0	0	0	0	1	0	1	1	1	1		X		X	X	X

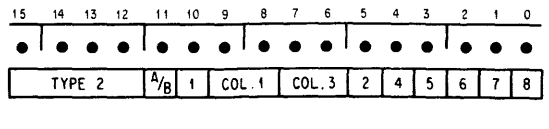
SELECTION TABLE

1	2	3	4
ALS ARS RAL RAR ALR ERA ELA ALF	CLE	SLA	ALS ARS RAL RAR ALR ERA ELA ALF
BLS BRS RBL RBR BLR ERB ELB BLF	CLE	SLB	BLS BRS RBL RBR BLR ERB ELB BLF

COMBINING GUIDE

1. Choose up to 4 instructions, one from each column of the Selection Table.
2. Use a one-bit for bit 9 to enable column 1 instructions, and a one-bit for bit 4 to enable column 4 instructions. Figure above shows column 1 enabled (executed first) with duplicate column 4 pattern (executed last) indicated by X's.
3. Use a one-bit for bit 5 to select column 2 (CLE), or a zero-bit to exclude CLE.
4. Use a one-bit for bit 3 to select column 3 (SLA/B), or a zero-bit to exclude SLA/B.

Figure 2-4. Shift-Rotate Instructions



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLA	0	0	0	0	0	1	0	1								
CLB	0	0	0	0	1	1	0	1								
CMA	0	0	0	0	0	1	1	0								
CMB	0	0	0	0	1	1	1	0								
CCA	0	0	0	0	0	1	1	1								
CCB	0	0	0	0	1	1	1	1								
SEZ	0	0	0	0		1					1					
CLE	0	0	0	0		1			0	1						
CME	0	0	0	0		1			1	0						
CCE	0	0	0	0		1			1	1						
SSA	0	0	0	0	0	1						1				
SSB	0	0	0	0	1	1						1				
SLA	0	0	0	0	0	1							1			
SLB	0	0	0	0	1	1							1			
INA	0	0	0	0	0	1								1		
INB	0	0	0	0	1	1								1		
SZA	0	0	0	0	0	1									1	
SZB	0	0	0	0	1	1									1	
RSS	0	0	0	0		1										1

SELECTION TABLE

1	2	3	4	5	6	7	8
CLA CMA CCA	SEZ	CLE CME CCE	SSA	SLA	INA	SZA	RSS
CLB CMB CCB	SEZ	CLE CME CCE	SSB	SLB	INB	SZB	RSS

COMBINING GUIDE

1. Choose up to 8 instructions, one from each column of the Selection Table.
2. Use the specified two-bit combinations of bits 9 and 8, plus A/B bit 11, to encode column 1 instructions.
3. Use the specified two-bit combinations of bits 7 and 6 to encode column 3 instructions.
4. Use a one-bit in bits 5, 4, 3, 2, 1, plus A/B bit 11, to encode column 2, 4, 5, 6, 7 instructions respectively.
5. Use a one-bit for bit 0 to encode column 8.

Figure 2-5. Alter-Skip Instructions

2-73. Register reference instructions are recognized by the computer when the four most significant bits of the instruction word are zeros; the general format for this type of instruction (the dots representing variable microinstruction bits) is therefore:

0 000 ... ...

2-74. SHIFT-ROTATE GROUP. The SRG instructions are specified by a zero for bit 10 (compare figures 2-4 and 2-5). Figure 2-4 gives both the bit format and the selection table for using these instructions. Definitions for the mnemonics used are listed below. Note that the Extend bit is not affected by shifts or rotates unless specifically stated. All of the shift and rotate instructions can be executed either first or last in a combined instruction, or both times. This permits sequencing of CLE and SLA/B either before or after shifts and rotates.

NOP No Operation. One timing cycle elapses.

CLE Clear E-register.

SLA Skip next instruction if Least significant bit of A-register is zero (i.e., skip if an even number is in A).

SLB Skip next instruction if Least significant bit of B-register is zero (i.e., skip if an even number is in B).

ALS A-register Left Shift one place, arithmetically (15 bits only). A zero replaces vacated bit 0; bit shifted out of bit 14 is lost; bit 15 (sign bit) is not affected.

BLS B-register Left Shift one place, arithmetically (15 bits only). A zero replaces vacated bit 0; bit shifted out of bit 14 is lost; bit 15 (sign bit) is not affected.

ARS A-register Right Shift one place, arithmetically. Bit shifted out of bit 0 is lost; copy of sign bit (bit 15) shifted into bit 14; bit 15 is not affected.

BRS B-register Right Shift one place, arithmetically. Bit shifted out of bit 0 is lost; copy of sign bit (bit 15) shifted into bit 14; bit 15 is not affected.

RAL Rotate A-register Left one place, all 16 bits. Bit 15 is rotated around to bit 0.

RBL Rotate B-register Left one place, all 16 bits. Bit 15 is rotated around to bit 0.

RAR Rotate A-register Right one place, all 16 bits. Bit 0 is rotated around to bit 15.

RBR Rotate B-register Right one place, all 16 bits. Bit 0 is rotated around to bit 15.

ALR A-register Left shift one place, same as ALS, but clear sign bit after shift.

BLR B-register Left shift one place, same as BLS, but clear sign bit after shift.

ERA Rotate E-register Right with A-register, one place (17 bits). Bit 0 is rotated into Extend Register; Extend content is rotated into bit 15.

ERB Rotate E-register Right with B-register, one place (17 bits). Bit 0 is rotated into Extend Register; Extend content is rotated into bit 15.

ELA Rotate E-register Left with A-register, one place (17 bits). Bit 15 is rotated into Extend Register; Extend content is rotated into bit 0.

ELB Rotate E-register Left with B-register, one place (17 bits). Bit 15 is rotated into Extend Register; Extend content is rotated into bit 0.

ALF Rotate A-register Left Four places, all 16 bits. Bits 15, 14, 13, 12 are rotated around to bits 3, 2, 1, 0 respectively. Equivalent to four successive RAL instructions.

BLF Rotate B-register Left Four places, all 16 bits. Bits 15, 14, 13, 12 are rotated around to bits 3, 2, 1, 0 respectively. Equivalent to four successive RBL instructions.

2-75. ALTER-SKIP GROUP. The ASG instructions are specified by a one in bit 10. Figure 2-5 gives both the bit format and the selection table for using these instructions. Definitions for the mnemonics used are as follows:

CLA Clear A-register.

CLB Clear B-register.

CMA Complement A-register. One's complement, reversing the state of all 16 bits.

CMB Complement B-register. Reverses state of all 16 bits.

CCA Clear, then Complement A-register. Puts 16 ones in the A-register; this is the two's complement form of -1.

CCB Clear, then Complement B-register. Puts 16 ones in the B-register; this is the two's complement form of -1.

CLE Clear E-register.

CME Complement E-register. Reverses state of the Extend bit.

CCE Clear, then Complement E-register. Sets the Extend bit.

SEZ Skip the next instruction if E-register is Zero.

SSA Skip next instruction if Sign bit (bit 15) of A-register is zero; i.e., skip if the content of A is positive.

SSB Skip next instruction if Sign bit (bit 15) of B-register is zero; i.e., skip if the content of B is positive.

SLA Skip next instruction if Least significant bit of A-register is zero (i.e., skip if an even number is in A).

SLB Skip next instruction if Least significant bit of B-register is zero (i.e., skip if an even number is in B).

INA Increment A-register by one. Can cause setting of Extend or Overflow bits.

INB Increment B-register by one. Can cause setting of Extend or Overflow bits.

SZA Skip next instruction if A-register is Zero (16 zeros).

SZB Skip next instruction if B-register is Zero (16 zeros).

RSS Reverse Skip Sense. Skip occurs for any of the preceding skip instructions, if present, when the non-zero condition is met. RSS without a skip instruction in the word causes an unconditional skip. If a word with RSS also includes both SSA/B and SLA/B, both bits (bit 15 and bit 0) must be ones for skip to occur; in all other cases the skip occurs if either or both conditions are met.

2-76. INPUT/OUTPUT INSTRUCTIONS.

2-77. The computer has 17 basic input/output instructions, which provide the following general capabilities.

a. Fix the state of the Flag, Control, and Overflow bit.

b. Test the state of the Flag and Overflow bits (i.e., skip if set or clear, as specified).

c. Enter data from a specific device into the A- or B-registers.

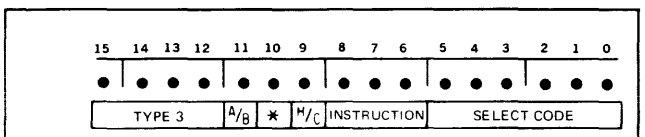
d. Output data to a specific device from the A- or B-registers.

e. Halt the program.

2-78. Input/output instructions are recognized by the computer when the four most significant bits of the instruction word are 1000 and bit 10 is a one. The codes and mnemonics for all 17 instructions are given in figure 2-6 (the MAC instruction is not counted as a basic instruction; see paragraph 2-80). All input/output instructions are executed in one phase.

2-79. Note that bit 11, where relevant, specifies A- or B-register; otherwise it may be one or zero without affecting the instruction, although the Assembler will assign zeros (as shown). Bit 9, where not specified, offers the choice of holding (0) or clearing (1) the device Flag after execution of the instruction. (Exception: the H/C bit associated with the last two instructions in this list holds or clears the Overflow bit instead of the Flag bit.) Bits 8, 7, and 6 identify the instruction; some of the instructions, however, require additional specific bits for the complete code. Bits 5 through 0 form select codes to make the instruction apply to one of up to 64 input/output devices or functions (see paragraph 2-100.)

2-80. The MAC instruction listed in figure 2-6 is available to provide up to 2048 entries to macroinstruction subroutines. Since it is used only by special options and special software, MAC is not counted as one of the 70 basic machine instructions. The basic computer will treat MAC as a No-Operation (NOP) instruction.



MAC	1	000	0			
HLT	1	000	1	000		
STF	1	000	10	001		
CLF	1	000	11	001		
SFC	1	000	10	010		
SFS	1	000	10	011		
MIA	1	000	01	100		
MIB	1	000	11	100		
LIA	1	000	01	101		
LIB	1	000	11	101		
OTA	1	000	01	110		
OTB	1	000	11	110		
STC	1	000	01	111		
CLC	1	000	11	111		
STO	1	000	10	001	000	001
CLO	1	000	11	001	000	001
SOC	1	000	1	010	000	001
SOS	1	000	1	011	000	001

* IDENTIFIES MACROINSTRUCTIONS (0) OR STANDARD INPUT/OUTPUT INSTRUCTIONS(1).

02116-A-5

Figure 2-6. Input/Output Instructions

2-81. HLT. Halt. Stops the computer, and holds or clears the Flag (according to bit 9) of any desired input/output device (bits 5 through 0). The HLT instruction has the same effects as the HALT pushbutton: the HALT switch lights, all front-panel control switches are enabled, and no interrupts may occur. The HLT instruction will be displayed in the T-register, and the P-register will normally indicate the halt location plus one.

2-82. STF. Set Flag. Sets the input/output Flag of the selected device, thus causing an interrupt during the next machine phase (if the interrupt system is enabled and the corresponding Control bit is set). The interrupt system itself is enabled by an STF instruction with a select code of six zeros (octal 00).

2-83. CLF. Clear Flag. Clears the Flag of the selected device, thus permitting the device to present another Flag when ready again. A CLF with a select code of six zeros (octal 00) disables the entire interrupt system; this does not affect the status of individual input/output Flags.

2-84. SFC. Skip if Flag Clear. Causes the computer to skip the next instruction if the Flag bit of the selected device is zero (i.e., the device is not ready).

2-85. SFS. Skip if Flag Set. The next instruction is skipped if the Flag bit of the selected device is one (i.e., the device is ready).

2-86. MIA. Merge Input into A. The contents of the input/output buffer associated with the selected device are merged ("inclusive or") into the A-register.

2-87. MIB. Merge Input into B. The contents of the input/output buffer associated with the selected device are merged ("inclusive or") into the B-register.

2-88. LIA. Load Input into A. The contents of the input/output buffer associated with the selected device are loaded into the A-register. Previous contents of the A-register are lost.

2-89. LIB. Load Input into B. The contents of the input/output buffer associated with the selected device are loaded into the B-register. Previous contents of the B-register are lost.

2-90. OTA. Output from A. The contents of the A-register are loaded into the input/output buffer associated with the selected device. If the buffer is less than 16 bits in length, the least significant bits of the A-register normally are loaded. (Some exceptions exist, depending on the type of output device.) A-register contents are not altered.

2-91. OTB. Output from B. The contents of the B-register are loaded into the input/output buffer associated with the selected device.

2-92. STC. Set Control bit of the selected device. This commands or prepares the device to perform its input or output function, and enables its Flag bit to interrupt the program being run (provided the program is not disabling the interrupt system).

2-93. CLC. Clear Control bit of the selected device. This prevents the device from interrupting. A CLC instruction with a select code of 00 (octal) clears all Control bits, effectively turning off all input/output devices. CLF 00 may be combined with this to additionally turn off the interrupt system.

2-94. STO. Set Overflow. The Overflow bit remains set until cleared by one of the following three instructions.

2-95. CLO. Clear Overflow. Clears the Overflow register.

2-96. SOS. Skip if Overflow Set. If the Overflow register is set, the next instruction of the program is skipped. Use of the H/C bit will hold or clear the Overflow bit following execution of this instruction (whether the skip is taken or not).

2-97. SOC. Skip if Overflow Clear. If the Overflow register is clear, the next instruction of the program is skipped. Use of the H/C bit will hold or clear the Overflow bit following execution of this instruction.

2-98. DATA FORMATS.

2-99. The basic format for arithmetic operations on numerical data is defined in figure 2-7. The data is assumed to be an integer (binary point to the right of bit 0), and is positive if the sign bit is zero, or negative if one. Negative data is represented in two's complement form in the computer. The largest possible positive number (in octal) is +77777, or (in decimal) +32767; the largest possible negative number is -100000 (octal) or -32768 (decimal). Other possible formats, including packed data words, double-length fixed point, and floating point representations, are defined in standard software packages.

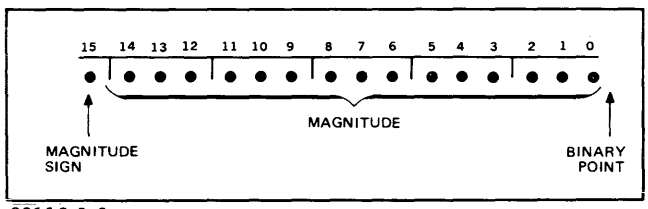


Figure 2-7. Basic Data Format

2-100. INPUT/OUTPUT SYSTEM.

2-101. Information is transferred into the computer from an external device, or out of the computer to an external device, by way of its input/output capability, termed the input/output system. A transfer of information is initiated by a signal from a device indicating that it is ready for input or output. The transfer occurs by the process of interrupting a running program (which could be either a problem-solving program, or a program specifically designed to transfer data). The interrupt directs the computer to a location in memory uniquely associated with the interrupting device. This location in turn directs the computer to a program routine (a service routine, which must previously have been stored in memory) containing instructions which effect the actual transfer of information. Since interrupts can occur at almost any time, including during the service routine of an earlier interrupt, a priority network is present in the computer to establish the sequence in which interrupts are serviced.

2-102. Interface cards may be plugged into any of the identical input/output slots, depending on the desired priority rating. Each combination of interface card and device, when plugged into the computer, constitutes an input/output channel. For further information about priority assignments and the input/output system in general, refer to Volume Three of the computer manual.

2-103. INPUT/OUTPUT OPTIONS.

2-104. Input/output options for the computer, identified by interface kit accessory numbers, consist of a combination of plug-in cards, interconnecting cables, and appropriate software. The following paragraphs briefly describe the capabilities added by several of these options. More detailed information will be found in Volume Three of the computer manual.

2-105. TELEPRINTER INPUT/OUTPUT. The simplest configuration of an HP computer system is provided by a combination of the HP 2752A Teleprinter (modified Teletype ASR-33) and the HP 12531C interface kit. The teleprinter combines a typewriter, punched tape reader and tape punch. Data and instructions may be entered from punched tape or the keyboard. Output information is recorded on the typewriter, and may be recorded simultaneously on punched tape. The teleprinter operates at 10 characters/second for both data entry and data recording. Where heavy use of the teleprinter is anticipated, 5 hours per day or 30 hours per week, a heavy duty HP 2754B Teleprinter (modified Teletype ASR-35) is recommended. This device uses the same interface. The HP 2752A and HP 2754B Teleprinters perform the same functions and operate at the same speed.

2-106. HIGH-SPEED PUNCHED TAPE INPUT. For rapid entry of punched tape programs and data into the computer, a 500 character-per-second HP 2748A Tape Reader, with its HP 12597-002 interface kit, is available.

2-107. HIGH-SPEED PUNCHED TAPE OUTPUT. Data output from the computer can be recorded (asynchronously) on punched tape at 120 characters/second with an HP 2753A Tape Punch and HP 12597A-003 interface kit. This device includes a tape spooler, which accepts approximately 1000 feet of tape.

2-108. PROCESSOR OPTIONS.

2-109. The following processor options are available for the computer, and all may be installed in the field. For further information about these options, refer to the technical data sheet for the computer or consult the nearest Hewlett-Packard Sales and Service Office.

2-110. POWER FAILURE INTERRUPT WITH AUTOMATIC RESTART. This option has the highest interrupt priority. If power fails with this option installed, a user designed subroutine automatically stores register contents and program location address in core locations. When power is restored, program execution continues.

2-111. DIRECT MEMORY ACCESS. This option enables the computer to transfer data directly between memory and external devices. It provides two 16-bit channels that are program assignable to any mainframe I/O channels. The option uses one timing cycle for each transfer; maximum transfer rate is 263,000 words per second.

2-112. MEMORY PARITY CHECK. This option provides each computer word in memory with odd parity, and monitors the parity of all words transferred from memory. The option allows for either a program interrupt or a computer halt if a parity error is detected.

2-113. MEMORY PROTECT. This option protects a selected block of memory of any size from alteration by memory reference instructions. It also limits input/output instructions to one-phase instructions in interrupt locations and instructions that reference the Switch or Overflow Registers.

2-114. EXTENDED ARITHMETIC UNIT. This option allows integer multiply, integer divide, double load, double store, long shifts, and long rotations to be performed by hardware means instead of by programmed subroutines.

2-115. MEMORY EXPANSION (16K, 24K, or 32K). These options provide for expansion of the basic 8K memory to 32K in steps of 8K.

2-116. SOFTWARE.

2-117. GENERAL. The computer is supported by a full range of software, normally furnished in the form of punched paper tape. As standard accessories, the following software packages are supplied with the computer, unless additions or deletions are otherwise specified. All are operable with the minimum system configuration; i.e., 8K memory and teleprinter input/output.

HP Basic Control System
HP Symbolic Editor
HP Assembler
HP ALGOL Compiler
HP FORTRAN Compiler
HP Relocatable Library
HP BASIC
HP System Input/Output
HP Hardware Diagnostics

2-118. Each of the software packages listed above consists in most cases of a number of individual tapes. The number of tapes furnished depends on the options purchased with a system; driver tapes and test tapes are furnished as accessories to interface options when purchased, either with the initial order or with field installation. Table 2-3 lists all the standard tapes furnished with a typical system, consisting of an HP 2752A Teleprinter, HP 2753A Tape Punch, and HP 2748A Tape Reader. In this case, 28 tapes would be furnished for computers having 8K memories. For 16K or larger memory computers, 26 tapes

would be furnished, since the FORTRAN compiler requires only two pass tapes instead of four. (Note: the list of standard software given in table 2-3 may change from time to time; check the HP Software Catalog, available from Hewlett-Packard Sales and Service Offices, for latest information.) In addition to these standard tapes, two configured tapes, incorporating actual system device assignments, are furnished with the initial shipment, one for the System Input/Output Drivers and one for the Basic Control System. The System Input/Output (SIO) Drivers primarily provide input/output capability for the Assembler, Symbolic Editor, and FORTRAN compiler, but may also be used as desired in user's programs. The Basic Control System, on the other hand, is primarily intended to provide a complete software input/output system for user's programs. These two tapes are unique to each system, and do not have HP accessory numbers and are not listed in the software catalog. Subsequent reconfiguring of System Input/Output and the Basic Control System, if desired, is easily accomplished by the user, with the aid of supplied software (System Input/Output Dump, and Prepare Control System).

2-119. TAPE IDENTIFICATION. Each software tape is separately identifiable by description and HP accessory number, labeled on both the tape container and the tape itself. The letter at the end of the number identifies a particular version of the tape (e.g., B supersedes A). A detailed list of the software packed with the system is given in the software installation record, supplied with the system documentation at the front of Volume Four. When ordering new or duplicate tapes (or documentation), the latest applicable version will automatically be furnished. Software is ordered through Hewlett-Packard Sales and Service Offices.

2-120. SOFTWARE CATALOG. A fee is charged for all software except for the one set of standard software tapes included with the computer (Mylar tapes are extra cost). The HP Software Catalog lists prices for all available software, including binary tapes, source tapes, Mylar tapes, and documentation. In addition to the standard software packages, Hewlett-Packard maintains a constantly growing library of HP programs, and new additions will be added to the software catalog.

Table 2-3. Standard Software

TAPE DESCRIPTION
*Basic Control System Input/Output Control Relocating Loader Debug Routines Prepare Control System **BCS Teleprinter Driver **BCS Tape Reader Driver **BCS Tape Punch Driver
Symbolic Editor Assembler FORTRAN Compiler: Pass 1 Pass 2 Pass 3 Pass 4 Relocatable Library (Math, Floating Point, Formatter) ALGOL
*System Input/Output System Input/Output Dump **SIO Teleprinter Driver **SIO Tape Reader Driver **SIO Tape Punch Driver
Hardware Diagnostics Alter-Skip Instruction Test Memory Reference Instruction Test Shift-Rotate Instruction Test Memory Address Test (Low Core) Memory Address Test (High Core) Memory Checkerboard Test (Low Core) Memory Checkerboard Test (High Core) **Teleprinter Test **General Purpose Register Diagnostic
*A configured tape is furnished with the initial shipment both for the System Input/Output and for the Basic Control System, in addition to the individual tapes listed above. These two additional tapes, unique to each system, do not have HP accessory numbers; they are identified only by system serial number.
**Driver tapes and test tapes are furnished for each type of device in a system. The tapes listed above are for a typical system.

SECTION III

FUNDAMENTAL THEORY OF OPERATION

3-1. INTRODUCTION.

3-2. This section describes how the computer manipulates information internally to execute the basic instructions defined in the preceding specifications section. In the interest of users without previous computer experience, the material in this and the following section is organized to begin at an elementary level, and to progress on the basis of previously given information, in the form of a training course.

3-3. The fundamental operations described in this section (and the following section) are in practice nearly always accomplished with the aid of software and input/output devices. However, for simplicity it will be assumed that the computer is an independent instrument and will be operated only by front panel controls. Additionally, it will be assumed for descriptive purposes that the computer runs slowly enough to observe the operations step by step. When running, the computer reads and executes each instruction usually in 1.6 or 3.2 microseconds. Thus only the beginning and ending conditions are normally readable on the front panel display. (Note: it is possible to single-step the computer through each instruction, one phase at a time, by using the SINGLE CYCLE pushbutton. This technique will be used in section IV.)

3-4. The computer performs its operations solely by instructions inserted into its memory by the user. The front panel controls therefore do not "operate" the computer, but rather are used for entering instructions and data into memory, and for initiating operation at the starting instruction. Very basically, the overall operation is:

a. The user enters instructions and data (all manually set in binary coded numbers on the 16 switches of the SWITCH REGISTER) into the computer's memory, using the LOAD ADDRESS and LOAD MEMORY pushbuttons.

b. When the program of instructions is complete in memory and is ready to be run, the user enters the address of the starting instruction, which points the computer to the location in memory where this first instruction has been stored. The SWITCH REGISTER and LOAD ADDRESS switches are used for this purpose.

c. The user presses the RUN pushbutton.

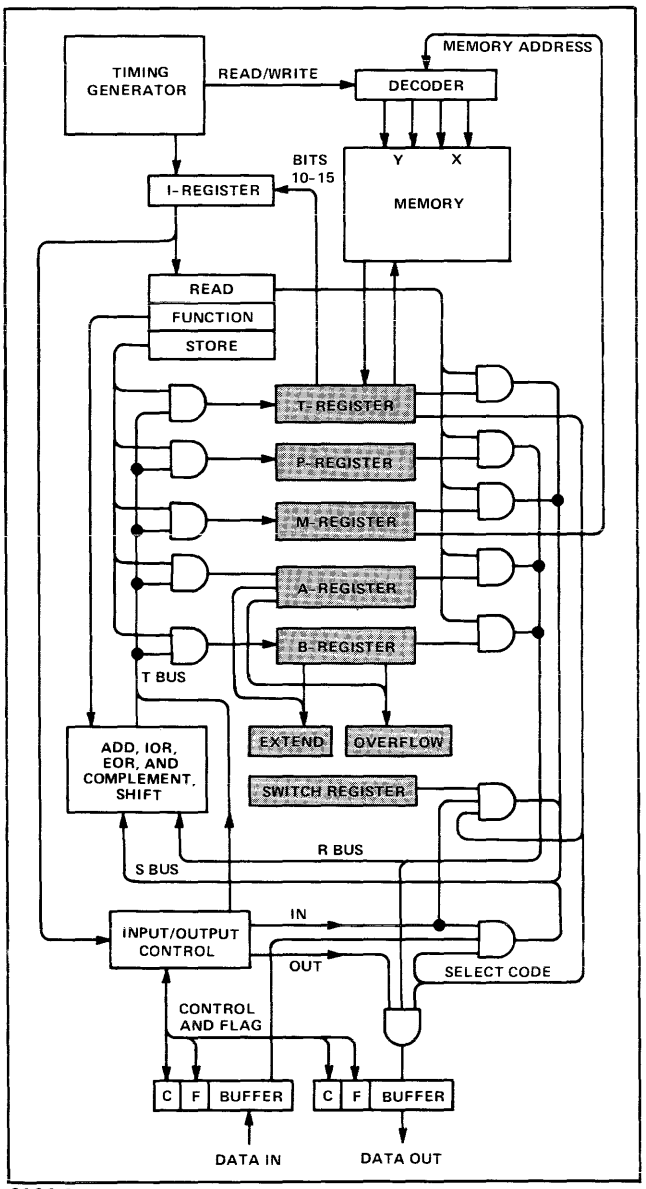
d. The computer reads and executes the instruction contained in the memory cell designated by the starting address.

e. The computer automatically continues to the next and all succeeding instructions, operating on the internally stored data, until reaching a halt instruction.

f. The user, having prepared the instructions and knowing where the computed answer is stored, reads the result. (The LOAD ADDRESS and DISPLAY MEMORY pushbuttons may be used to display the answer on the front panel.)

3-5. FRONT PANEL PRESENTATION.

3-6. To present the material of this section in the most practical form from the user's point of view, the descriptions will relate to the front-panel view of the computer. Figure 3-1 is a simplified block diagram of the computer,



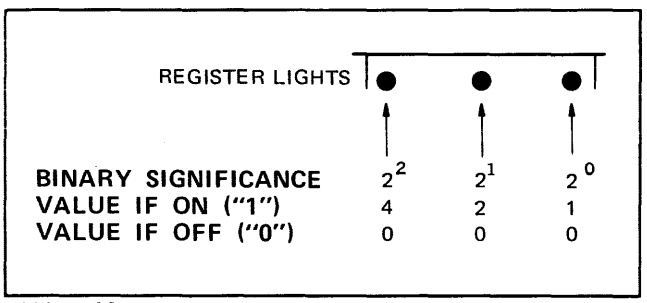
2106-6

Figure 3-1. Simplified Computer Block Diagram

showing the relationship of the display registers. The block diagram, which corresponds to the physical layout of the panel (shaded blocks), will be used for descriptions of register operations later in this section.

3-7. Information is displayed in rows of 16 lights, numbered 0 through 15, and the Switch Register consists of 16 switches similarly numbered. Each light or switch represents a bit (condensed from "binary digit") in the binary numbering system, where a light off or a switch down is a "zero", and a light on or a switch up is a "one". In the binary system, there are only two digits, 0 and 1, which are easily stored and manipulated by a computer using bistable devices. Thus input information which is applied to the computer in binary form (such as by the Switch Register) is said to be in "machine language" since the computer can handle these numbers directly without conversions of any kind. For the user, however, binary numbers are difficult to read and use, so the bits are grouped in threes for convenient notation in the "octal" numbering system.

3-8. There are five three-bit groups in each row of panel lights and the Switch Register, with one bit remaining at the left end. This last bit, bit 15, is normally used for special purposes (e.g., to indicate direct/indirect addressing or +/- numbers). In converting each group of three bits to an octal digit, the binary significance of each bit is converted to its absolute value, which is then considered to be absent or present, depending on whether the bit is a "zero" (light off) or a "one" (light on) respectively. This is shown in figure 3-2.



02116-A-10

Figure 3-2. Composition of Octal Digit

3-9. By various combinations of on and off states,eight digits are possible, 0 through 7. The digits 8 and 9 never appear in the octal numbering system. Figure 3-3 lists all eight binary/octal equivalents, along with some examples of numbers as might be read from a computer display register.

3-10. As can be seen from the last example in figure 3-3, the largest possible number which can be displayed by a register is 77777 (all lights on). Since there are no 8's or 9's in the octal system, this number must correspond to some lower value in the decimal system (specifically 32767). To avoid confusion when numbers are written in more than one numbering system, a subscripted digit is attached to the number to identify the system used. Thus:

$$111111111111111_2 = 77777_8 = 32767_{10}.$$

The computer manuals will use these subscripts or the word binary, octal, or decimal whenever such confusion may occur.

BINARY		OCTAL INTERPRETATION		OCTAL
000	=	0	=	0
001	=	1	=	1
010	=	2	=	2
011	=	2+1	=	3
100	=	4	=	4
101	=	4+1	=	5
110	=	4+2	=	6
111	=	4+2+1	=	7

EXAMPLES

5	2	6	0	1
101	010	110	000	001
7	4	3	5	0
111	100	011	101	000
7	7	7	7	7
111	111	111	111	111

02116-A-11

Figure 3-3. Binary/Octal Conversions

3-11. COMPUTER STRUCTURE.

3-12. Figure 3-1, the simplified block diagram of the computer, is the basis for the partial versions used to illustrate descriptions in this section. This figure will be reconstructed step by step as the explanations progress. The first step is figure 3-4, which outlines the blocks and signal routes mentioned in the following discussion of memory.

3-13. THE COMPUTER MEMORY.

3-14. The memory section of the computer is its information storage area. "Information" is a broad term intended to cover anything which can be represented as a binary number; this includes instruction codes, memory addresses and alphabetic codes, as well as pure numeric data. The primary storage section of the computer is a core memory that is internal to the computer. Auxiliary storage is available in the form of disc and magnetic tape; however, these units are accessed through the computer input/output

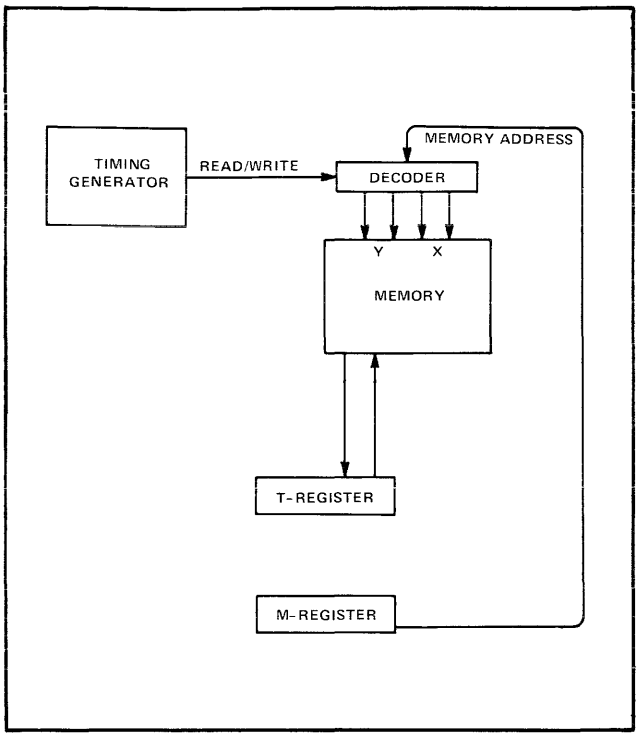


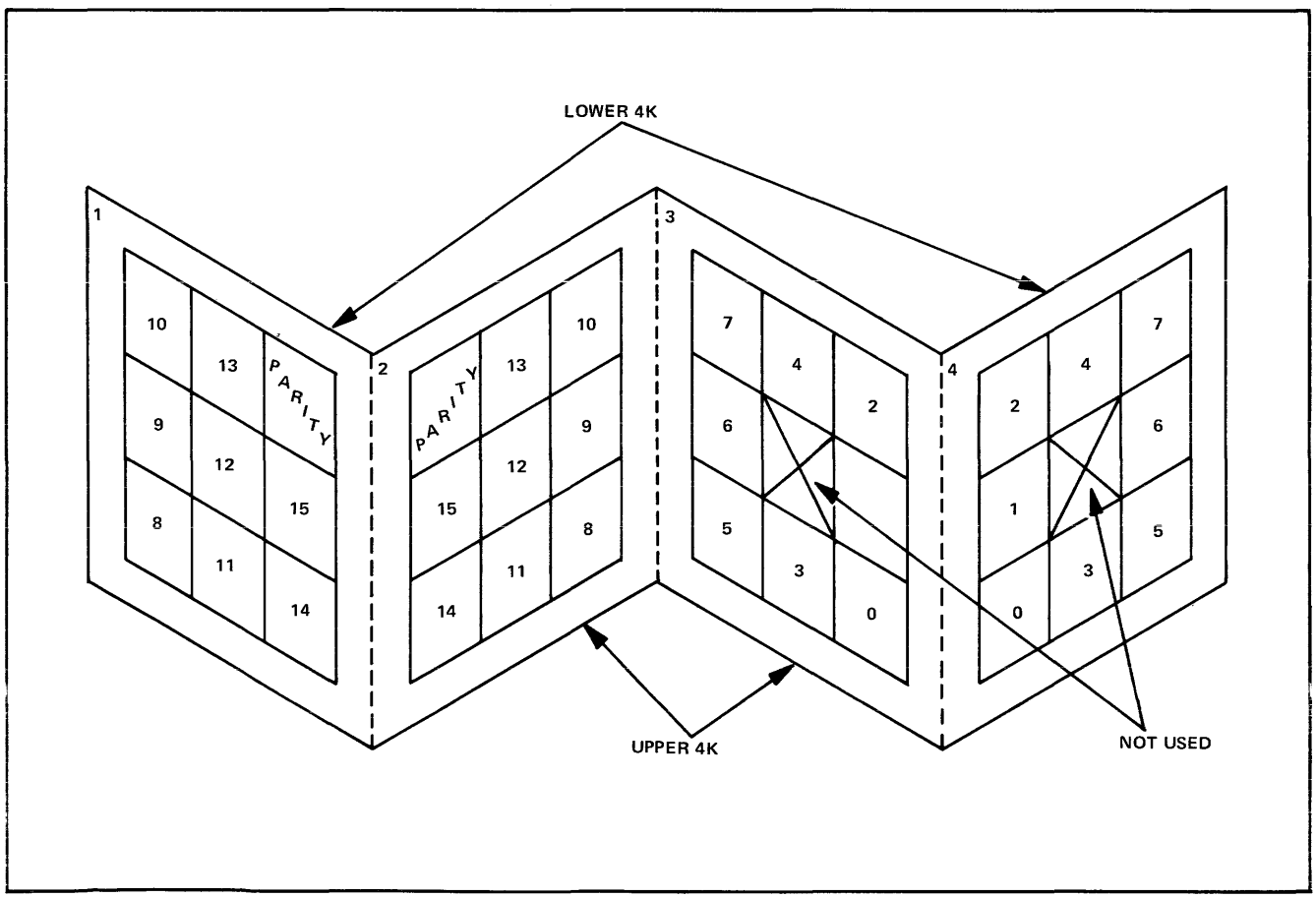
Figure 3-4. Memory Block Diagram

system and are not treated as an extension of the memory in this discussion.

3-15. The portion of the computer block diagram that describes the operation of the memory section is shown in figure 3-4. The M-register (further defined in paragraph 3-31) supplies a 16-bit memory address to the decoder circuits. The decoder locates this address in the memory and supplies the required signals for a read or write operation as directed by the timing generator. If a read operation is taking place, 16 bits of information are transferred from the addressed memory location to the T-register (further defined in paragraph 3-29). During a write operation, the information is transferred from the T-register to the addressed memory location.

3-16. The following discussion provides general information on the physical structure of the core memory, the core storage technique, and the operation of the memory including addressing the memory, reading information from the memory, and writing information into the memory.

3-17. PHYSICAL STRUCTURE. Figure 3-5 shows the physical structure of the 8K memory module used in the computer. The entire 8K of memory is mounted on four boards (frames) which are hinged together as shown in figure 3-5. After construction, the four frames are folded to



2106-7

Figure 3-5. Core Stack Physical Details

form a small package and mounted on a plug-in card for insertion into the computer card cage. The memory module is shown unfolded in figure 3-5 for illustration purposes only.

3-18. Each of the four frames contains nine bit-planes. A "bit-plane" is defined as a grid of wires containing 4096 separately addressable cores. All cores on a bit-plane correspond to the same bit number of the computer word format. Two of the frames (numbered 1 and 4 in figure 3-5) provide the cores for storing the 4096, 16-bit words addressed as 00000 through 07777 octal (lower 4K). The remaining two frames (numbered 2 and 3) provide core storage for addresses 10000 through 17777 octal (upper 4K). Frame 1 contains the parity bit-plane for the lower 4K and frame 2, the parity bit-plane for the upper 4K. These bit-planes are used only if the computer is equipped with the memory parity option. The center bit-planes on frames 3 and 4 are not used.

3-19. THE FERRITE CORE. As previously explained, the computer handles all information in binary form; that is, by a combination of "zeros" and "ones". The ferrite core is a small ring of soft iron that can be magnetized in either of two possible directions, clockwise or counterclockwise. One direction is assigned the binary value of one and the other the value of zero. If a current carrying wire is threaded through the core, the direction of magnetization of the core can be changed by reversing the direction of the current in the wire. Since the mass of the core is very small (diameter of .03 inch), little current is required to switch the binary state; this allows fast switching speeds (about 400 nanoseconds). The magnetic state of the core remains indefinately after the current is removed, so switching can be accomplished with bidirectional current pulses.

3-20. ADDRESSING THE CORE. The basic memory of the computer has a storage capacity of 8192, 16-bit words; this requires 131,072 ferrite cores. It is apparent that some means must exist to address only a specific 16 cores to store a computer word. Also, it must be possible to locate those same 16 cores to read the stored word. To provide this means of addressing, the single current carrying wire mentioned in the above paragraph is replaced with two wires; one labeled X and the other labeled Y. The X and Y wires are threaded through the core as shown in figure 3-6. Each wire supplies one-half of the magnetizing force required to change the magnetic state (clockwise or counterclockwise) of the core. Both currents must be present at the core and must be passing through the plane of the core in the same direction to affect the magnetic state of the core. If both currents are reversed, the magnetic state of the core will reverse.

3-21. Figure 3-7 shows the X and Y wiring of a small segment of one bit-plane. Current pulses are applied to the X wire labeled X1 and the Y wire labeled Y4. The core at the intersection of wires X1 and Y4 (circled on figure 3-7) is the only core on the bit-plane that will be affected. To address this core, the memory address decoding circuits must cause current pulses to be applied to this combination of X and Y wires.

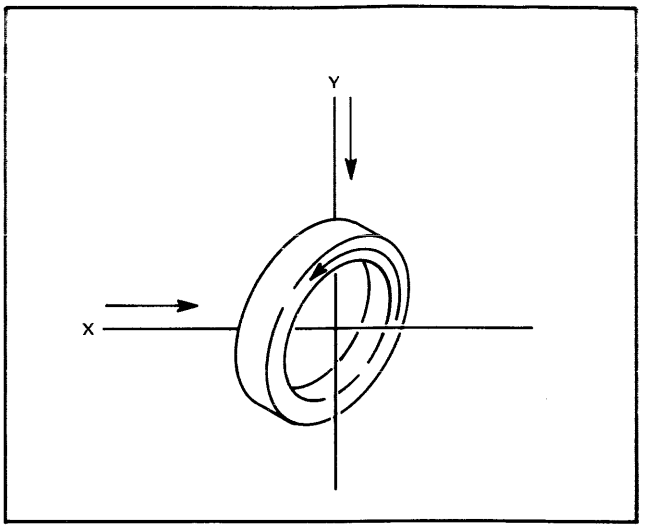


Figure 3-6. Magnetizing a Core

3-22. When addressing a computer word, all 16 bits of that word must be affected simultaneously. Figure 3-8 illustrates this with one X and one Y wire shown in relation to the entire 8K core stack. The X and Y wires cross at one core in each of the 32 (34 if memory parity is used) bit-planes used in the computer memory. This corresponds to one 16-bit word address in the lower 4K and one in the upper 4K. However, if the directions of the X and Y currents are as indicated by the arrows in figure 3-8, only the cores in the lower 4K will be affected. The X and Y currents pass through the cores in the upper 4K in opposing directions and will not supply the magnetizing force required to change the magnetic state of the core. Thus, only the cores in the lower 4K have been addressed. If the

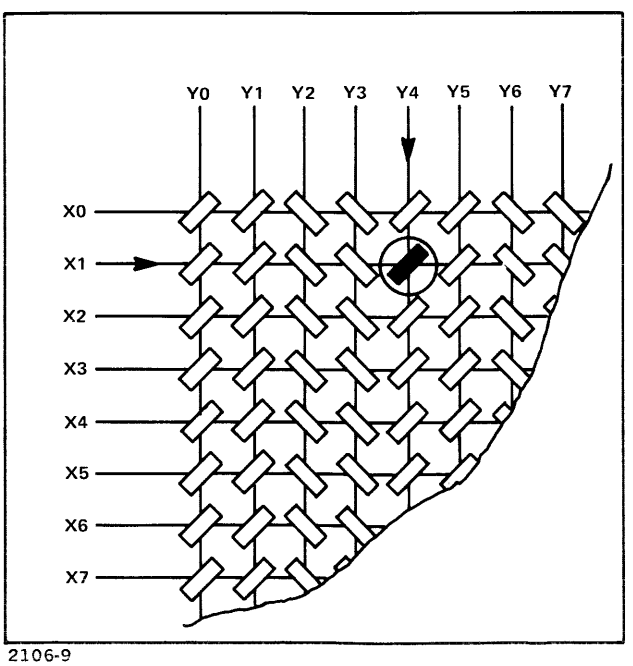
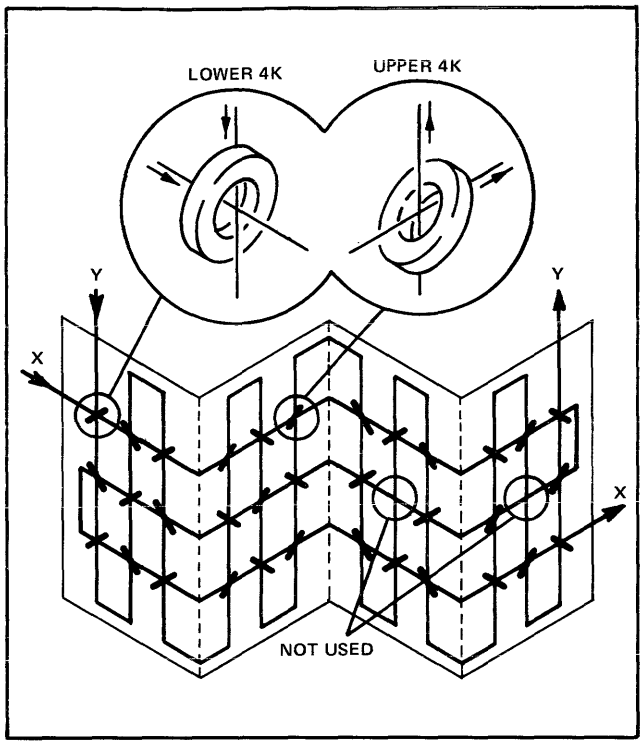


Figure 3-7. Addressing One Bit

X current direction is reversed and the Y current remains the same, the cores in the upper 4K will be addressed. The following paragraphs describe how read and write operations are performed on the addressed cores.



2106-10

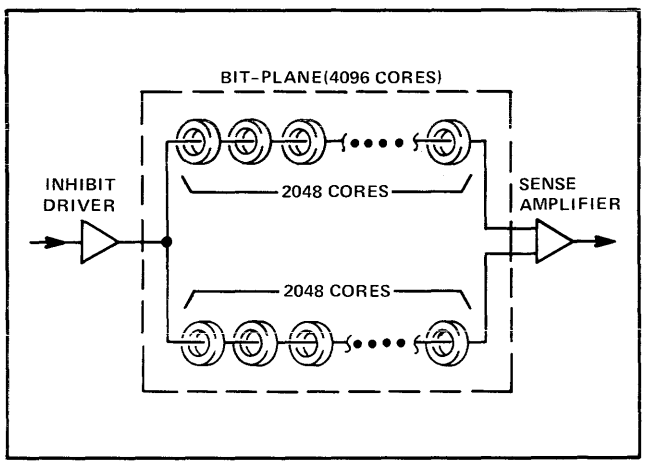
Figure 3-8. Addressing One Word

3-23. READING. During the read cycle, the direction of the X and Y currents is selected to drive the addressed cores to the binary zero state. If the core is already storing a zero, it will not be affected; if it is storing a one, it will switch to a zero. This change in magnetic state is detected as an induced voltage pulse in a third (sense) wire threaded through the core. Since this voltage is induced in the sense wire only when the core changes state, its presence is interpreted by the computer as a binary one. Conversely, the absence of a pulse on the sense wire during the read cycle is interpreted as a binary zero. At the end of the read cycle, all of the addressed cores will be in the zero state; a write cycle must follow to restore the original data word or to replace it with a new data word.

3-24. WRITING. To write information into the memory, the X and Y currents are reversed so that the addressed core is driven toward the binary one state. Since the addressed core is at the binary zero state after a read cycle (a read cycle always precedes a write cycle), writing a zero requires that the computer inhibit the writing of a one. This is accomplished by applying a current to an inhibit wire that will cancel the magnetizing effect of the X and Y currents. The inhibit wire is threaded through the core parallel to the Y wire. If the direction of the inhibit current is opposite that of the Y current, the magnetizing force applied to the core will not be strong enough to change the core to the binary one state. The core is then storing a binary zero. If the core is to store a binary one, the inhibit current is omitted and the X and Y currents drive the core to its binary one state.

3-25. In the above discussion, only one core is used to illustrate the read and write cycles. During the actual reading and writing of a computer word, 16 cores (one for each bit) are affected simultaneously. All of the 16 cores are addressed by the same X and Y currents; each of the cores has individual sense voltages and inhibit currents.

3-26. COMMON SENSE/INHIBIT WIRING. The preceding paragraphs describe a need for a sense wire for read operations and an inhibit wire for write operations. The core memory of the computer uses a single wire to perform both of these operations. Figure 3-9 shows the common sense/inhibit wiring for one bit-plane of the core stack. Notice that the inhibit current actually flows through two wires during the write cycle. Each wire passes through half of the 4096 cores on the bit plane. The inhibit current is returned to ground through the input circuitry of the sense amplifier. The sense amplifier is of the symmetrical type and will deliver an output for either positive or negative sense voltage pulses.



2106-11

Figure 3-9. Common Inhibit/Sense Wiring

3-27. THE REGISTERS.

3-28. Figure 3-10 shows the seven working registers of the computer. The five principal registers (T, P, M, A, B) are purposely shown as being independent of each other since, in fact, information is not transferred directly from register to register. Rather, information is transmitted via the bus system (described later under paragraph 3-36) under command of the instruction logic (paragraph 3-39). The following paragraphs explain why the registers are needed, not how they are operated. In essence, these registers are short-term information storage devices consisting of flip-flop circuits. Front-panel indicator lamps indicate the status of each bit.

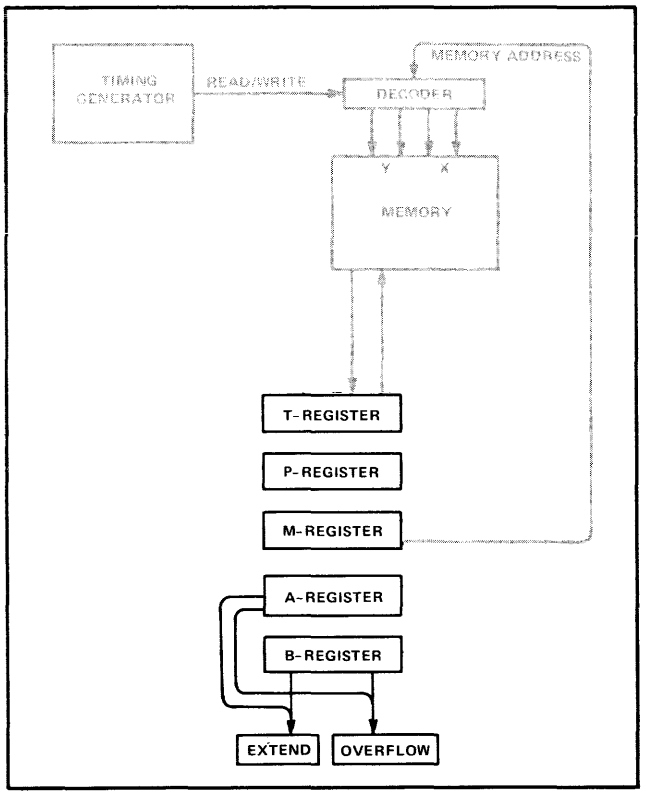


Figure 3-10. Register Block Diagram

3-29. T-REGISTER. The T-register was briefly mentioned in the description of how memory operates. As can be assumed from that description and from the front-panel engraving (MEMORY DATA), the T-register holds data that is read out of and written into memory. For the majority of operations when a computer is running, the principal concern is with the data read out of a memory cell; once a word of information is in the T-register, it is accessible for arithmetic operations and for transfers to other registers via the bus system. For the reverse (write) operation, the T-register is loaded by transfers from other registers, and the information is stored in memory during the latter half of the memory cycle.

3-30. P-REGISTER. The P-register is the computer's program counter. This means that this register goes through a step-by-step counting sequence and causes the computer to read successive memory locations, corresponding to the existing count. In the simplest case, the P-register would start at zero when the RUN pushbutton is pressed, causing memory location 00000 to be read into the T-register. The computer would act on the instruction code in location 00000, then advance the P-register one location (memory location 00001_8). This process of stepping through memory locations (at a rate of 1.6 or 3.2 microseconds per step for most instructions) continues until one of the instructions read out is a halt, which terminates the program. Of necessity, this simple case is not typical. First, programs do not normally begin at locations lower than 00100_8, since these locations are reserved for special purposes. Therefore the starting address of a program must be manually set into the P-register before pressing RUN. Second, the strict sequential stepping can be altered in the course of a program, either by a skip instruction (which causes the P-register to increment by two instead of one, thus skipping one memory location) or by a jump instruction (which transfers numbers from another register into the P-register, thus causing the program to continue at a different point in memory).

3-31. M-REGISTER. As implied by figure 3-1, the M-register (MEMORY ADDRESS) is the only means of addressing specific memory locations. The setting of the M-register can occur from any of the other registers, depending on the effects of instructions. In the preceding paragraph, it could be assumed that the P-register directly addresses memory; in actual fact, however, the computer must transfer the desired address from the P-register to the M-register, which in turn addresses the desired memory location. Thus it is seen that these two registers will frequently contain the same number. The reason why both registers are needed is that it is necessary for one register (the P-register) to keep track of the location of the current instruction in case the instruction is a multiple phase type. In this case, the M-register may have to be changed several times in the course of executing an instruction. A common example would be when the instruction is to "add the contents of location 100_8 to the A-register" (ADA 100). The P- and M-registers would be identical while reading this instruction out of memory (say the instruction is in location 500_8; both registers indicate this value). Then the M-register would have to change to 100_8 to get the contents of this location for the addition. After the addition has been executed, the contents of the P-register are incremented by one (501_8). The P- and M-registers are then both set to this new value, and the computer is then ready to read the next instruction.

3-32. A-REGISTER. The A-register is one of the two accumulators. An accumulator in a computer accumulates the results of arithmetic operations. A simple example was given in the preceding paragraph, where one number from memory was added to the existing contents of the A-register. Assuming that the A-register previously held the number 1000_8, and the number in location 100_8 was 22_8, the number left in the A-register after execution of the instruction would be 1022_8. Other types of operations which may be done with the A-register are: boolean logic operations ("and", "exclusive or", "inclusive or"), comparison for equality with a memory word, shifting or rotating of bits left or right, testing the status of individual bits, complementing of bits, and accepting or holding data for transfer to and from external devices. All of these operations are accomplished by the instruction logic (paragraph 3-39).

3-33. B-REGISTER. The B-register is the second of the two accumulators. It has the same capabilities as the A-register, except that the three boolean logic instructions (AND, XOR, IOR) can apply only to the A-register. The main reason for having two accumulators is to provide faster, more flexible arithmetic than can be accomplished with one accumulator. This advantage will be seen later in programming of the computer.

3-34. EXTEND. The Extend Register is shown connected to bit 15 (left end bit) of both A- and B-registers. This is to indicate that this one-bit register becomes set whenever there is a carry out of bit 15 of either accumulator; i.e., whenever the quantity accumulated exceeds 16 ones. This fact is frequently of significance. For example, if the quantity in an accumulator is 16 ones and an ADD instruction adds one, the result in the accumulator will be 16 zeros. This answer is obviously incorrect; it is correct if the Extend bit, which is now in the set state ("1") is temporarily assumed to be "bit 16". The program can be written to make this assumption, and it can proceed without error on the basis of the resulting information. To be certain that the Extend information is valid, the Extend Register is normally cleared by an instruction (CLE) before the addition is done. Another valuable feature of the Extend Register is its ability to link the two accumulators (effectively providing a single 32-bit accumulator).

3-35. OVERFLOW. The Overflow Register is similar in purpose to the Extend Register. The difference is that, whereas the Extend Register indicates that the largest 16-bit quantity has been exceeded, the Overflow Register indicates that the largest "signed" quantity has been exceeded. (A program may work with both signed and unsigned numbers.) Since bit 15 is the sign bit, bit 14 (as shown in figure 3-10) is the source of the significant carry. Having two possible signals (+ and -) means that detection of overflow requires two different sets of conditions. For addition of two positive numbers, overflow occurs if there is a carry from bit 14 to bit 15 in one of the accumulators. For addition of two negative numbers (which are represented in two's complement form), overflow occurs if there is not a carry from bit 14 to bit 15. Obviously, overflow cannot occur when adding numbers of opposing signs, since the resulting quantity cannot be greater than the larger of the two numbers. As with the Extend Register, the Overflow Register should be cleared before an addition.

3-36. THE BUS SYSTEM.

3-37. Figure 3-11 outlines the routes by which data travels internally from one register to another. Although the buses are represented by a single line in this figure, assume each line to be composed of 16 individual lines, one for each register bit. The computer uses an "R-S-T" bus configuration. This is a conventional notation designating a three-bus system which applies two input buses (R and S) to an arithmetic unit with output on the third bus (T). The use of two input buses permits arithmetic operations to combine the contents of two registers. A common example would be the execution of the "ADA 100" instruction previously used. In this example, the contents of location 100_8 is the number 22_8. During execution of the instruction, this number (22_8) would be read into the T-register. The other number (1000_8) is in the A-register. Simultaneously (by a method described under the next paragraph heading, instruction logic) both the T-register and the A-register are read onto their respective buses (S and R). The two numbers are added in the arithmetic logic circuits, and the result (1022_8) is stored via the T-bus back into the A-register as the accumulated sum.

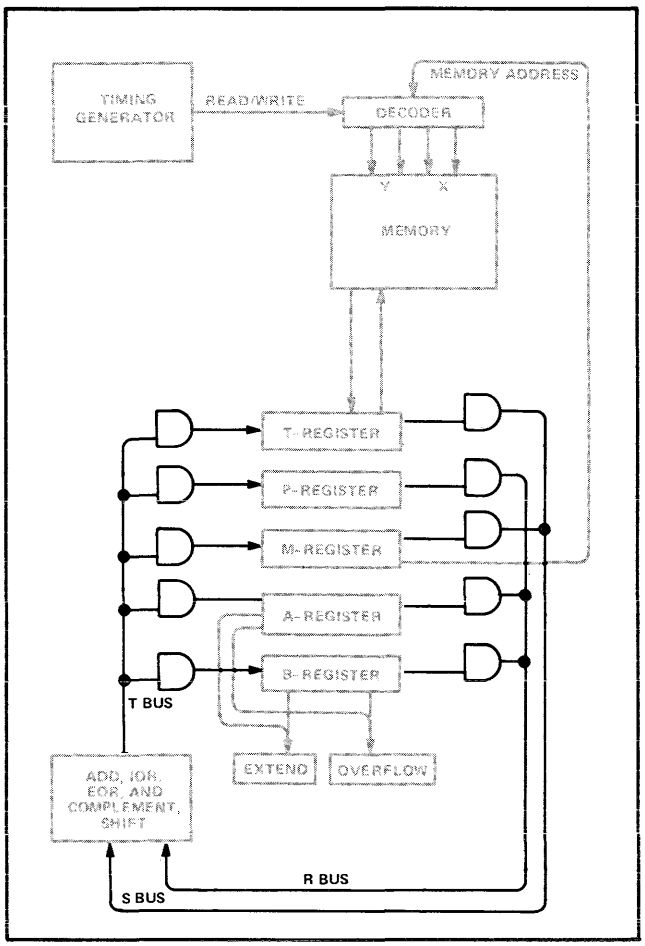


Figure 3-11. Bus System Block Diagram

3-38. Note that several register combinations are possible as inputs to the arithmetic logic. One point worth noting is that since the A- and B-registers are addressable as memory locations, the contents of these registers can be transferred via the R- and T-buses into the T-register. From this point, the contents can be combined in the manner described above with either accumulator (including combining the number with itself; e.g., "add A to A"). This is all accomplished in one instruction.

3-39. THE INSTRUCTION LOGIC.

3-40. Figure 3-12 shows the elements of the instruction logic in the computer. As indicated in the figure, timing is essential to the operation of the instruction logic. The following descriptions do not detail all timing relationships, since these vary with instructions, but it should be understood that timing pulses are gated with each operation to make it occur in proper sequence. A general introduction to machine timing is given in paragraph 2-7 of Section II.

3-41. As shown in figure 3-12, the six most significant bits read out of memory during each memory cycle are applied to the 6-bit Instruction Register (I-reg), which decodes the instruction. (Actually, the Instruction Register receives its information via the T-register.) Only during the

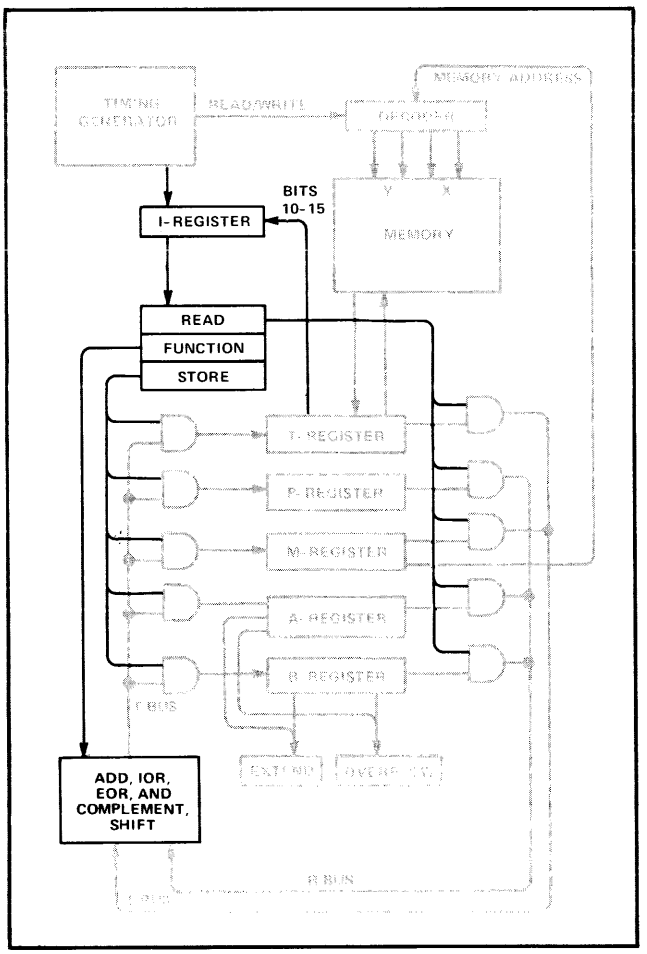


Figure 3-12. Instruction Logic Block Diagram

fetch phase, however, are these bits recognized as an instruction code (as determined by a fetch phase, PH1, signal from the timing generator). At this time, the decoded instruction enables three functional operations, which in turn will become active at specific times, depending on the instruction. These operations are described individually in the next three paragraphs.

3-42. READ. The read signal, shown connected to the output gate of all five working registers, strobes the data of one or two registers onto their corresponding buses (R and S). This places the data at the inputs of the arithmetic logic circuits.

3-43. FUNCTION. The function signal activates one of the six listed arithmetic functions. The selected function alters or combines the data on the R- and/or S-buses, and routes the resulting data out on the T-bus.

3-44. STORE. The store signal, shown connected to the input gate of all five working registers, effectively opens the input of one or more of these registers to accept the data which appears on the T-bus (preceding paragraph). In many cases, depending on the instruction, only part of the information on the T-bus is stored into a register.

3-45. THE INPUT/OUTPUT SYSTEM.

3-46. Figure 3-13 shows the means by which data is transferred into and out of the computer. This is the input/output system; all elements shown are contained within the main frame. Interface arrangements are shown for only two external devices, one input and one output. Actually the system has capability for 16 interfaces in the main frame, plus 32 or 16 additional interfaces in an HP 2150B or HP 2151A Input/Output Extender. The Switch Register is shown as part of the input/output system, and is considered to be an input device.

3-47. As indicated by figure 3-13, the input/output control logic is used to process all input/output operations. Input/output control operates in two ways:

a. Processes input/output instructions.
b. Processes service requests by peripheral devices.

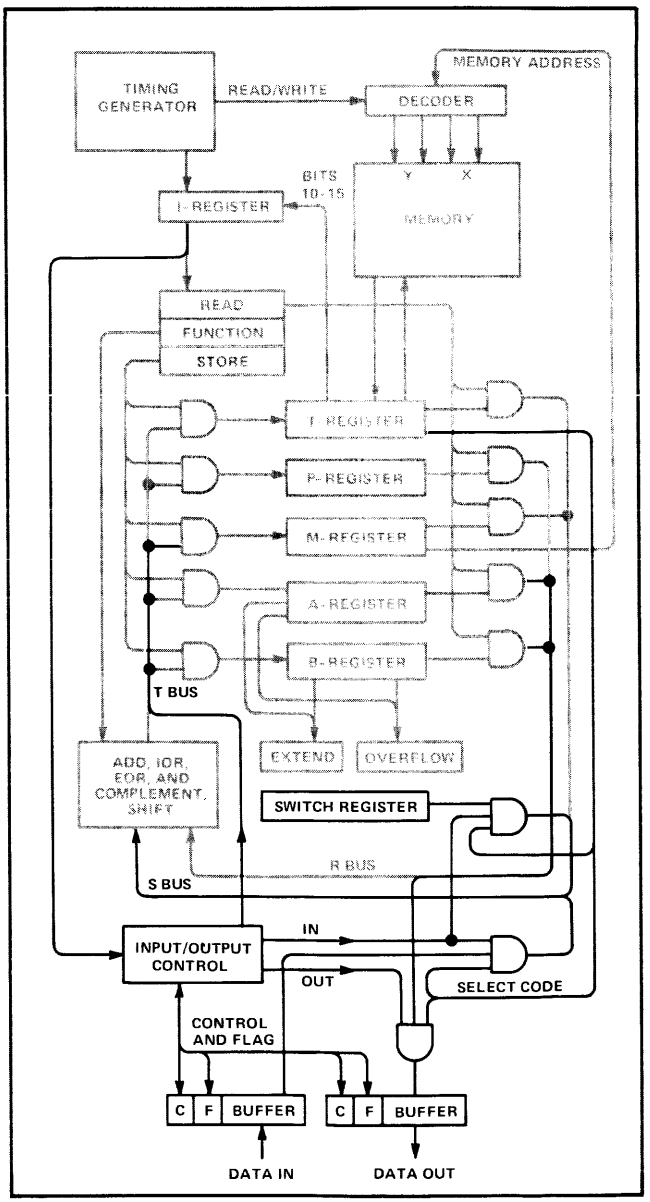


Figure 3-13. Input/Output System Block Diagram

3-48. These two types of operations are separately discussed in the following paragraphs.

3-49. PROCESSING INSTRUCTIONS. Input/output instructions from the Instruction Register are routed to input/output control, which translates the instruction into appropriate driving signals. One such signal is an "In" signal, which strobes all interface positions for input (represented by two "and" gates in figure 3-13, one accepting data from a buffer register and one accepting data from the Switch Register). Only one of these interface positions can be enabled, according to the select code (bits 0 through 5 from the T-register), and the corresponding data is strobed by the "In" pulse onto the S-bus. From there it is transferred via the T-bus into the A- or B-register (as enabled by a Store signal at the A or B input gate).

3-50. Another driving signal is the "Out" signal. This signal strobes all interface positions for output (one shown in figure 3-13). The select code from the T-register enables one interface position, and permits the "Out" signal to strobe the data on the R-bus into the corresponding output buffer. (The data on the R-bus was read out of the A- or B-registers by a read signal.)

3-51. In addition to transferring data, as in the preceding two paragraphs, input/output control can (according to instruction) send out signals to test the state of Control and Flag bits (C and F), or to set or reset these bits. The select code determines which interface will receive the signal from input/output control. The Control and Flag bits are command signals for transferring data between the buffer and the peripheral device (peripheral not shown).

3-52. PROCESSING SERVICE REQUESTS. If a specific instruction has at some previous time enabled the interrupt system (considered to be in the input/output control block in figure 3-13), a peripheral device may request new data from the computer (if output) or request to feed new data to the computer (if input). This request for service is done by setting the interface Flag bit. The Flag signal, via input/output control, interrupts computer operation by forcing the M-register to be set (via the T-bus) to a memory address specified by the Flag. At the same time, the fetch phase is set so that the computer must execute the instruction contained in the specified memory cell. Generally this instruction will be a jump to a service subroutine. This subroutine consists of instructions that will prepare or accept the new data. On completion of service, it is the subroutine's responsibility to return the P- and M-register to the values they contained before being interrupted.

SECTION IV

OPERATING THE COMPUTER

4-1. INTRODUCTION.

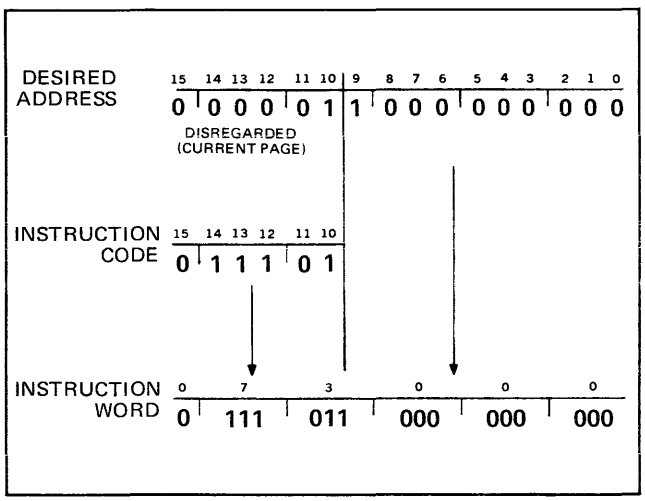
4-2. The purpose of this section is to relate "theoretical" operations described in the preceding section to visible actions. Specific information is given to familiarize the user with panel controls, and to enable the user to perform basic operations on the computer, when necessary, without input/output devices or software aids. These purely manual operations are most commonly encountered in computer maintenance, and in loading, examining, and changing small sections of memory (e.g., loading the Basic Binary Loader).

4-3. Obviously, manual usage of the computer is not the intended mode of operation for practical applications. Therefore this section does not attempt to teach programming to the extent of practical problem solving. This aspect is the subject of training materials supplied with user training courses, which are provided by the Cupertino Division of Hewlett-Packard. User training concentrates on the efficient use of software to solve problems. Instructions for usage of the computer via input/output devices are given in Volumes Three and Four of the computer manual (Input/Output System Operation Manual, and Programmer's Reference Manuals).

4-4. CODING.

4-5. This section assumes familiarity with binary and octal numbering systems. Table A-4 (Consolidated Coding Table) in the appendix of this volume is used as a reference for instruction codes; if more detail is required, refer to the information given in paragraph 2-45 (Instructions). As a reminder: a "one" is coded by a switch of the Switch Register being in the up position, and is indicated by a register light being on. A "zero" is coded by a switch in the down position, and is indicated by a register light being off.

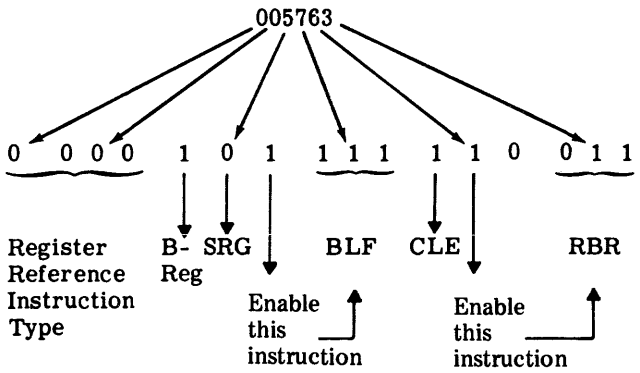
4-6. All numbers used for addresses or contents in this section are octal numbers unless otherwise specified. Notation of instruction codes in octal numbers is an operator's convenience for loading and reading binary information. The meaning of the octal code can be understood only when it is broken down into its binary elements. For example, note the first instruction code to appear in this text, (paragraph 4-19). The instruction is STA 3000 (Store A-register into memory location 003000; initial zeros of address assumed). The coded instruction word is 073000. Refer now to the Consolidated Coding Table (table A-4 in the appendix). Note that the code for STA consists of ones in bit positions 14, 13, and 12, and a zero in bit position 11. Since indirect addressing is not being used at this time, bit 15 is a zero. Bit 10 must be a one, since the program and all references will be on the same (current) memory page. This accounts for bits 10 through 15. The remaining bits (0 through 9), which comprise the memory address, are simply the corresponding bits of the desired address. The desired address in this case is 003000. This breaks down in binary form as shown in the top row of figure 4-1. Note that all bits higher than bit 9 of the desired address are disregarded by the programmer when composing the instruction word. This is because these bits fall outside of the page-size limits. The M-register, which contains the page-designating bits, will hold the bits constant at execute time, as commanded by bit 10 of the instruction code.



02116-A-29

Figure 4-1. Coding a Memory Reference Instruction Word

4-7. It is evident that the octal digit "3" in the resulting instruction word, 073000, is the result of three individual factors: bit 11 (a zero) specifies the A-register, bit 10 (a one) specifies current page, and bit 9 (a one) is an address bit. This requirement of using bits having separate, individual meanings to compose an octal digit is frequently encountered. For example, suppose that it is desired to rotate the B-register left three places and clear the Extend bit, all in one instruction. From the Shift-Rotate group definitions (paragraph 2-74), it is determined that a suitable method for a three-place rotation is to rotate the B-register left four places (BLF), then right one place (RBR). The resulting octal code for the instruction which will accomplish these actions (including the clearing of the Extend bit) is 005763. The way this number was composed can be shown by breaking it down into its binary components, as follows:



Note

The ability to code instructions in octal form is essential to the procedures given in the remainder of this section. It is therefore strongly recommended that the reader take the time at this point to study the composition of the above instruction code, with reference to the Consolidated Coding Table in the appendix. As an exercise, do a similar breakdown of the following example: 003145, which is a single-word instruction to skip if Extend is set, clear Extend, and complement and increment the A-register. Five microinstructions are involved. Determine which ones, from the code.

4-8. COMPUTER TURN-ON.

4-9. Assuming that installation of the computer has been completed, turn on computer power using the POWER switch. Initially, the HALT pushbutton and the FETCH indicator should light. The register lights will come on in a random pattern. Should one or more of these indications fail when turning on the computer, refer to Volume Two, Installation and Maintenance Manual.

4-10. It is good practice when turning on the computer, or when beginning any new operation, to press the PRESET pushbutton and to ensure that the LOADER switch is in the PROTECTED position.

CAUTION

The following procedures, to the end of this section, are designed to be performed on the computer while reading the text. Considerable loading effort can be saved if the entire set of procedures is performed in the sequence given, without any interruptions which might disturb procedures in progress. Since these procedures alter memory, the operator should also be certain that he is not destroying valuable information which may have been stored previously in the computer. Memory locations used in these procedures are:

1001 through 1010
1020 through 1036
2166 through 2207
2766 through 3036
3777 through 4003

4-11. PRELIMINARY OPERATIONS.

4-12. The first and most basic operation is to put some information into the computer memory. The following paragraphs, through 4-21, outline in detail two methods of doing this. One method is to manually store the setting of the Switch Register directly into a specified memory cell by using the front-panel operating controls. The other method is to let the computer itself do the storing operation. The purpose in showing these two methods is to demonstrate that computer "instructions" are equivalent to operating controls.

4-13. Figure 4-2 illustrates the two memory storing methods. Note that in the first case the information is transferred from the Switch Register to a location in memory. In the second case (Programmed Storing), the transfer is from the A-register. For simplicity, information will be put into the A-register manually from the Switch Register (broken LOAD A line). However, as will be seen later, this information could come from anywhere in memory or from the B-register (broken LDA lines). Note also that, for simplicity, figure 4-2 omits detailed routing via the bus system and T-register as described in the preceding section.

4-14. MANUAL STORING.

4-15. First it is necessary to decide where in memory the information is to be stored. For illustrative purposes, location 003000 has been selected. To direct the computer to this address, set the number into the Switch Register, as shown in step 1 of figure 4-3. Then press the LOAD ADDRESS pushbutton (step 2). This immediately transfers the setting of the Switch Register into the P- and M-registers, as can be read from the indicator lights. The computer is now "at" location 003000 (the addressed location).

4-16. Now the operator can store any desired information into the addressed location. An easy to recognize pattern of zeros and ones in alternating groups of three is suggested in figure 4-3 (in octal: 070707). Complete steps 3 and 4 of figure 4-3. Note that the P- and M-registers have incremented to the next location (which will not be used at this point). The T-register indicates the information (070707) which went into memory.

4-17. To verify that location 003000 does indeed contain the information 070707, complete steps 5 through 8. Again, note that the P- and M-registers, at the conclusion of this procedure, are one step "ahead" of the information displayed in the T-register. This is because the P- and M-registers must direct the computer to the next location, whereas the T-register always indicates information resulting from previous action.

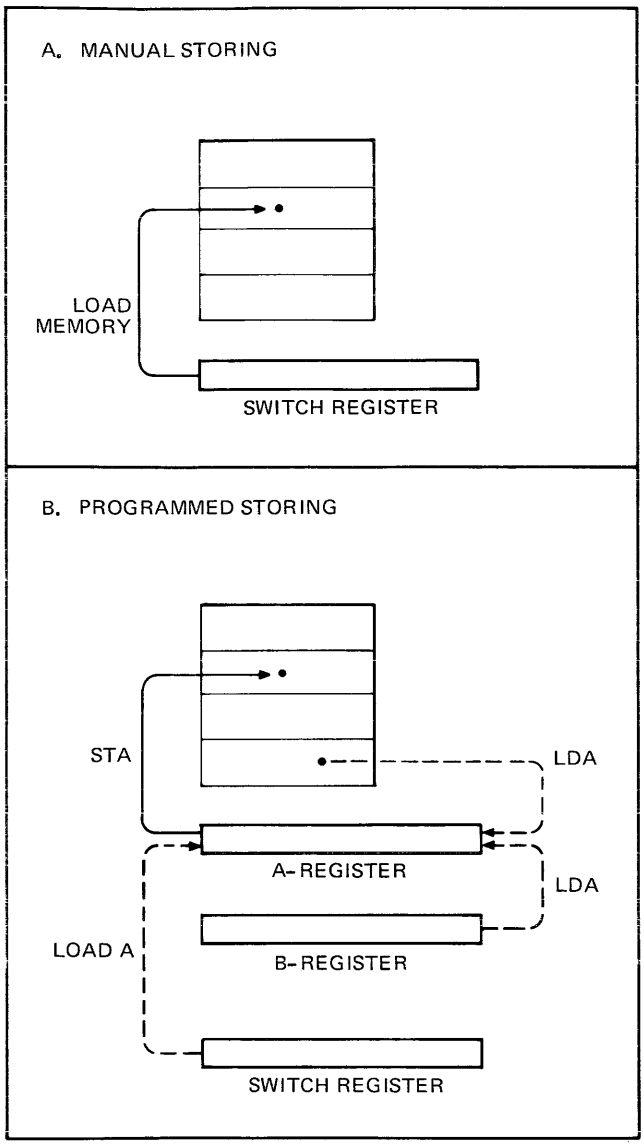


Figure 4-2. Two Methods of Storing Information in Memory

4-18. PROGRAMMED STORING.

4-19. For the computer to perform its own storing operation, it is first necessary to put into memory the instruction (STA, Store contents of A-register) which will accomplish this. Then the computer can be directed to the place in memory where this instruction is located; pressing the RUN pushbutton will then let the computer go ahead and execute the instruction. After doing so, the computer will look for its next instruction in the following location, and will attempt to continue running. Since it is unknown what other information may be in memory, it is necessary to stop the computer as soon as the desired action is completed, simply by putting a halt (HLT) instruction in the immediately succeeding location. The required program therefore consists of two instructions: STA, HLT.

4-20. The manual storage procedure of paragraphs 4-14 through 4-17 puts an easy to recognize pattern (070707) into location 003000. It is the objective of the next paragraph (procedure detailed in figure 4-4) to let the computer put a different pattern (all ones) into the same location, replacing the previous pattern. This new pattern is loaded into the A-register before the program is run.

4-21. Steps 1 through 6 of figure 4-4 store the two-word program into memory, using the two locations immediately preceding the location to be altered (003000). Steps 7 and 8 load the new pattern into the A-register. Steps 9 and 10 verify that the old pattern is still in location 003000. Steps 11, 12, and 13 cause the program to be run. The computer executes this program in 4.8 microseconds; therefore the computer will be back in the halt condition (HALT light on) faster than can be visually detected. Steps 14 and 15 verify that the new pattern (177777) is now in location 003000.

4-22. THE STORED PROGRAM.

4-23. The preceding descriptions have demonstrated that internal, presettable commands can control operation of the computer in the same manner as front-panel controls. If the computer were constructed like a mechanical calculator, there might be panel controls to "add" or "subtract", but this would be defeating the design principles of a computer. The intent is to provide flexibility through use of internal commands which can be arranged to occur in a specific sequence, and to limit panel controls to the minimum required to initiate operation. This, in essence, is the concept of the stored-program computer. The following paragraphs discuss the elements of the stored program.

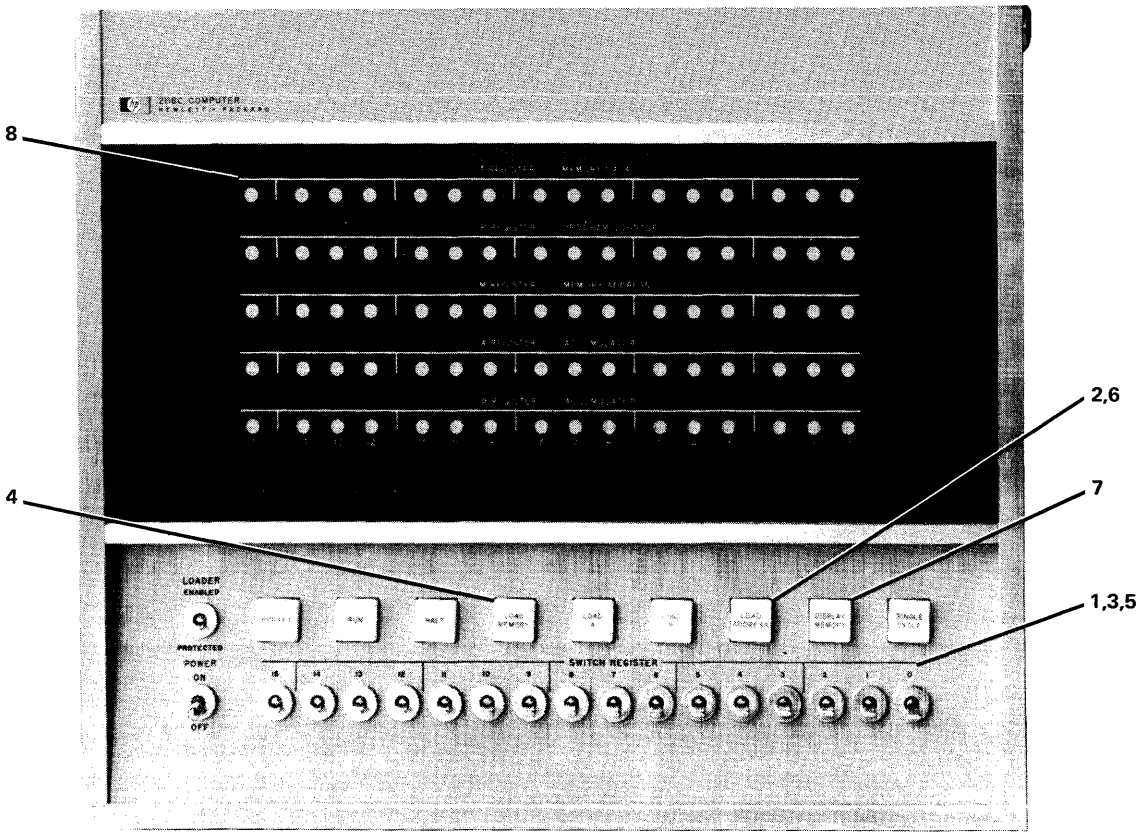
4-24. A program consists of a sequence of computer words, stored in memory, which control operation of the computer. The general term "computer words" is used rather than the restrictive term "instructions" since the stored information generally includes three types of words:

a. The instruction word.
b. The data word.
c. The address word.

4-25. Although these terms are to some extent self-explanatory, the distinction and usage requires illustration. For purposes of illustration, the simple program example used in the preceding descriptions will be expanded and examined in more detail, beginning at paragraph 4-31. Before proceeding, however, the method of writing programs in a concise, meaningful form will be presented. Notation of this kind becomes increasingly necessary as programs grow larger.

4-26. PROGRAM TABLE.

4-27. Table 4-1 puts into tabular form the two-word program previously used as an example in paragraphs 4-18 through 4-21. The information in this table corresponds to steps 1 through 6 of figure 4-4. The format of the table is used for explanatory purposes within this volume only, but



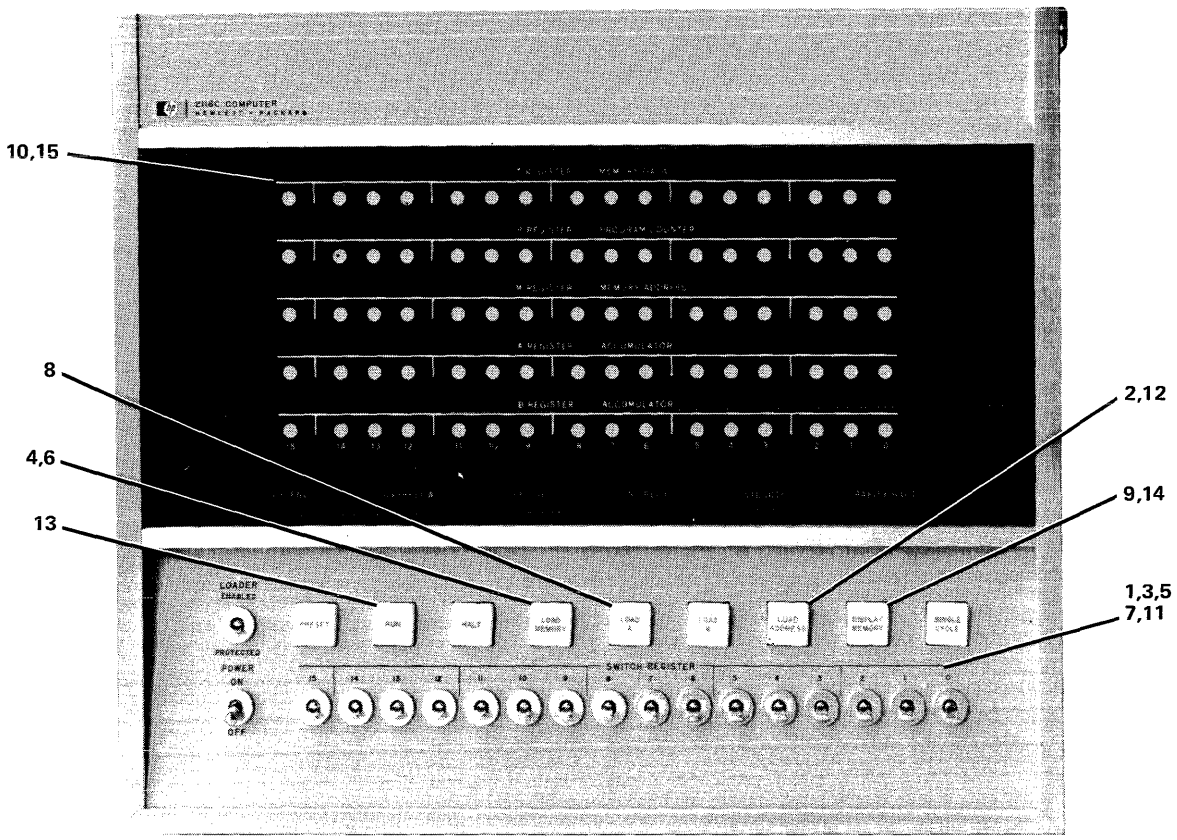
STORE

1. Set to 003000 (0 000 011 000 000 000).
2. Press LOAD ADDRESS.
3. Set to 070707 (0 111 000 111 000 111).
4. Press LOAD MEMORY.

CHECK

5. Set to 003000.
6. Press LOAD ADDRESS.
7. Press DISPLAY MEMORY.
8. T-register indicates contents of memory location 003000: 070707 (no change).

Figure 4-3. Storing Information Manually



LOAD PROGRAM	1.	Set Switch Register to 002776.
	2.	Press LOAD ADDRESS.
	3.	Set to 073000 (STA 3000).
	4.	Press LOAD MEMORY.
	5.	Set to 102000 (HLT).
	6.	Press LOAD MEMORY.
LOAD NEW INFORMATION	7.	Set to 177777.
	8.	Press LOAD A.
CHECK OLD INFORMATION	9.	Press DISPLAY MEMORY.
	10.	T-register indicates contents of memory location 003000: 070707 (from manual storing procedure).
RUN PROGRAM	11.	Set to 002776.
	12.	Press LOAD ADDRESS.
	13.	Press RUN.
CHECK NEW INFORMATION	14.	Press DISPLAY MEMORY.
	15.	T-register indicates new contents of memory location 003000: 177777.

Figure 4-4. Storing Information by Program

resembles in general arrangement the format required for using the assembler coding forms. Sample programs in this section are organized to expand on each preceding program, step by step. Shaded portions of the program tables correspond exactly to previously discussed material, and are therefore not described in detail. This permits the discussions to concentrate on the new (unshaded) portions of the sample program.

4-28. ADDRESS. The Address column of the program table states where in memory the program words (contents) are to be stored. The first listed address states where the program is to begin; this is termed the "starting address". The starting address of the program shown in table 4-1 is 002776; the program stops at the location immediately following (002777). Although the program never advances to location 003000 (the location immediately following 002777), this address must be listed in the program table as a reminder that this memory location will be used by the program.

4-29. CONTENTS. As previously explained, the stored program can consist of three types of words: instructions, data, or even the address of another location. Therefore the contents of a location specified by an address may take various forms in the Contents column. Most memory locations of a program will be instructions; the instruction mnemonic is listed under "Instruction (or Data)" in the table. If the content is not an instruction (usually a pure number representing data or an address), it will also appear under this heading, as shown in table 4-2. In the case of memory reference instructions, the address of the location affected by the instruction is listed under the Memory Reference heading. For example, the first instruction listed in table 4-1 is a command to store the A-register contents in location 003000. Location 003000 is the affected location (i.e., the Memory Reference). The D/I, A/B, and Z/C headings are also used only in the case of memory reference instructions. As a reminder to code a one-bit for I (indirect addressing), B (B-register), and C (current page), only these three indicators will be given in the tables; D (direct addressing), A (A-register), and Z (page zero), all coded by zero bits, are otherwise assumed. The Octal Code column is used for the coded version of the desired contents. This column comprises the "machine-language program", since this is the information which is loaded into the computer. As far as the computer is concerned, these numbers are the program. Note that no specific contents need be loaded for address 003000, since the STA 3000 instruction will destroy any information previously contained here.

4-30. REMARKS. A short explanation accompanying each assigned address of the program is helpful in communicating the intent of program details to other persons, and also can serve as a reminder to the original programmer when re-examining the program at a later time. Words used for the Remarks column should be carefully chosen to be as concise and meaningful as possible. Understanding a given program can be difficult enough without adding confusion through vague documentation. For example, it would not be incorrect to say for the first instruction of table 4-1: "Store contents of A in location 3000." However, this does not say any more than the instruction word itself says (STA 3000). The remark suggested in table 4-1 states what is expected to be in the A-register (a "pattern"), and raises the questions of what the pattern is, and how it happened to get into the A-register. This leads the operator to look for further documentation (in this case the text of this manual), which tells him how to preset the A-register. Additional words to indicate the need for presetting the A-register could be added, improving the message still further. Conversely, the simple remark "Halt" in the next line requires no additional comment.

4-31. PROGRAM EXECUTION.

4-32. Table 4-2 lists the program used as an example in this discussion. The main purpose of the program is to show where and when the three types of program words (instruction, data, and address) occur. In the process of so doing, detailed actions for simple addition and indirect addressing will also be illustrated. The program adds 5 to 5 and puts the result (10 decimal, or 12 octal) into location 003000. Note that the middle three lines of the program are the same as the example given in table 4-1. The first two lines expand the program to accomplish the addition, and the last two lines are data and address words used by the program.

4-33. LOADING THE PROGRAM. The program is loaded into the computer manually, using the sample procedure given in steps 1 thru 4 of figure 4-3. Steps 1 and 2 need

Table 4-1. Program Table

ADDRESS	CONTENTS						REMARKS
	INSTRUCTION (OR DATA)	MEMORY REFERENCE	D/I	A/B	Z/C	OCTAL CODE	
002776	STA	3000			C	073000	Get pattern from A, put in 3000.
002777	HLT					102000	Halt.
003000							Reserved for answer.

Table 4-2. Program to Show Instruction, Data, and Address Words

ADDRESS	CONTENTS						REMARKS
	INSTRUCTION (OR DATA)	MEMORY REFERENCE	D/I	A/B	Z/C	OCTAL CODE	
002774	LDA	3001			C	063001	Put augend in A.
002775	ADA	3777	I		C	143777	Add the addend specified by 3777.
002776	STA	3000			C	073000	Put answer in 3000.
002777	HLT					102000	Halt.
003000						-	Reserved for answer.
003001	5					000005	Data.
003777	3001					003001	Address of addend is 3001.

be done only once for most of the program, since each LOAD MEMORY operation automatically increments the address in the P- and M-registers. Specifically, the procedure is:

a. Set the Switch Register to the starting address (002774), and press LOAD ADDRESS.

b. Set the Switch Register to the first word of the program (063001), and press LOAD MEMORY.

c. Set the Switch Register to the next word of the program, and press LOAD MEMORY. Repeat this step until the first six words have been loaded. For the fifth word (which requires no contents), it is convenient to simply press LOAD MEMORY with the HLT code still in the Switch Register. A halt instruction in this location does no harm.

d. For the seventh word, which is not in sequence with the other six, it is necessary to set the address (003777) into the Switch Register and press LOAD ADDRESS. Then set the Switch Register to the contents (003001), and press LOAD MEMORY.

4-34. RUNNING THE PROGRAM. Again set the Switch Register to the starting address (002774) and press LOAD ADDRESS. Now press RUN. Immediately the computer switches to the halt condition, having executed the problem and stored the answer in location 003000 in 12.8 microseconds. To verify that the computer has arrived at the right answer (000012), press the DISPLAY MEMORY pushbutton. The answer is in the T-register. This demonstrates how fast the computer operates, but does not show what operations it went through to arrive at its answer. Therefore, the following paragraphs will re-run the program step by step in order to show these operations.

4-35. SINGLE CYCLE OPERATION. Table 4-3 shows the contents of each register following each operation of the SINGLE CYCLE pushbutton. The program will be executed in eight steps (i.e., eight machine phases). The following eight paragraphs describe each of these steps. The program is initially set up by setting the Switch Register to the starting address (002774) and pressing the LOAD ADDRESS pushbutton. The conditions now existing are shown in the top line of table 4-3: the P- and M-registers hold the starting address, and the remaining registers can be in any state. The FETCH phase indicator light on the panel is on, indicating that the first machine phase will be a fetch phase; this is an effect of the LOAD ADDRESS switch.

4-36. Press the SINGLE CYCLE pushbutton (first step). The conditions of the registers after the computer has completed this first phase are shown in the step 1 line of table 4-3. The computer interprets any word read out of memory during a fetch phase as an instruction word. It is the programmer's responsibility to ensure that the computer does find an instruction in every location to which the P-register goes. This is ensured by properly filling out the program table; e.g., in table 4-2, the program (P-register) starts at 002774, and stops at 002777. Every one of these locations must have an instruction word as its contents. Later in the fetch phase (T6 and T7), the memory reference bits (0 through 9) of the T-register are transferred into bits 0 through 9 of the M-register. The remaining bits of the M-register are left unchanged (since there is no reference to page zero), thus completing the memory reference address in the M-register. In comparing the contents of the T- and M-registers in step 1 of table 4-3, be careful not to assume that the complete octal digits "3001" are transferred; the digit "3" is a composite of three binary bits with different code meanings. Also occurring at the end of the fetch phase is the setting of the EXECUTE (phase 3) condition. The P- and M-registers are not yet affected.

4-37. Press the SINGLE CYCLE pushbutton again (step 2) to complete execution of the LDA 3001 instruction. Step 2 of table 4-3 shows register conditions existing after completion of the execute phase. This is the phase in which the computer gets the data requested by the memory reference, and does with it whatever is commanded by the instruction code. The read portion of the memory cycle reads the contents of the location addressed by the M-register (now at 003001) into the T-register. This information, read out of memory by the execute phase, is a data word. It is the programmer's responsibility to ensure that a data word

Table 4-3. Single Cycle Execution of a Program

STEP	INSTRUCTION	T-REGISTER	P-REGISTER	M-REGISTER	A-REGISTER	B-REGISTER	PHASE
		Any	002774	002774	Any	(Not used)	FETCH
1	LDA	063001	002774	003001	Any		EXECUTE
2		000005	002775	002775	000005		FETCH
3	ADA,	143777	002775	003777	000005		INDIRECT
4	I	003001	002775	003001	000005		EXECUTE
5		000005	002776	002776	000012		FETCH
6	STA	073000	002776	003000	000012		EXECUTE
7		000012	002777	002777	000012		FETCH
8	HLT	102000	003000	003000	000012		FETCH

(or an indirect address) is contained in all locations to which there is a memory reference (unless the location is to be used by the program for storage). As seen in table 4-2, there are three memory references; therefore the table accounts for three addresses in addition to the four addresses assigned to the program instructions. One of these three is a storage location, one is data, and one is an indirect address. In this step, the information read out is the data "5". The data is transferred from the T-register to the A-register during the execute phase. Therefore the number 5 appears in both registers. At the end of this phase, the P- and M-registers are set to the address of the next instruction (002775), and the fetch condition is set (FETCH light on), for reading of the next instruction.

4-38. Press SINGLE CYCLE (step 3). This fetches the next instruction (143777) out of location 002775. The code 143777 means: add, to whatever is in the A-register, the contents of a memory location which can be found by going first to location 3777 for more information. This is what is implied by the symbolic form: ADA 3777,I. The indirect bit (bit 15 of the word now in the T-register) caused the setting of the indirect phase (INDIRECT light on), and the memory reference bits (0 through 9) have been transferred into the M-register. The P- and A-registers remain as they were. The indirect phase is ready to begin.

4-39. Press SINGLE CYCLE (step 4). The computer always interprets information read out of memory during an indirect phase as an address word. This word (003001) is transferred to the M-register as the new memory reference for the current ADA instruction. Both T- and M-registers now contain 003001. Since bit 15 of this word is a zero (direct address), the execute condition is set (EXECUTE light on). If this bit had been a one (indirect), the indirect condition would remain set, and a further memory reference would be obtained in the next step. However, with this example, the computer now knows that the addend data is located in 003001. It happens, in this example, that this is the same location from which the augend was taken; however, the address word could just as well refer to any location in memory.

4-40. Press SINGLE CYCLE (step 5). In the execute phase of the ADA instruction, the data in location 003001 is read out (the number 5), and is added to the existing contents of the A-register (which up until now also contained the number 5). The T-register therefore contains 5, and the A-register contains 12. As usual, the last operation for any instruction is to advance the P- and M-registers to the location of the next instruction (002776) and to set the fetch phase condition.

4-41. Press SINGLE CYCLE (step 6). The fetch phase of the STA 3000 instruction reads the instruction word (073000) out of location 002776, transfers the memory reference bits to the M-register, and sets the execute phase condition.

4-42. Press SINGLE CYCLE (step 7). The execute phase puts the A-register contents (000012) into the memory via the T-register. Therefore both registers indicate this value. As usual, the P- and M-registers are advanced to the address of the next instruction (002777), and the fetch phase condition is set.

4-43. Press SINGLE CYCLE (step 8). The halt instruction is read out of memory, and the computer halts. The DISPLAY MEMORY pushbutton can now be pressed to verify that location 003000 has received the correct answer, 000012.

4-44. REFERENCING OTHER PAGES.

4-45. The procedures given in the preceding paragraphs used three memory reference instructions: LDA 3001; ADA 3777,I; and STA 3000. All of these instructions were stored in the second page of memory; i.e., they were stored in locations 2774, 2775, and 2776. In addition, the addresses to which these instructions referred (3001, 3777, 3000) were also located in the second page of memory. Thus each memory reference is a "current page" reference; i.e., no reference is made to an address which is outside the page in which the program itself is operating.

4-46. One program reference (ADA 3777,I) went to the page limit. This instruction could not have been ADA 4000,I, which refers to a location just one address higher. Location 4000 is not on the current page. On the other hand, ADA 1777 (with or without I) is possible, even though location 1777 also is not on the current page. The following paragraphs, through 4-62, deal with the special considerations for referencing memory pages other than the current page. The first step is to know what constitutes a page of memory.

4-47. CONCEPT OF THE MEMORY PAGE.

4-48. The necessity for dividing memory into pages arises from the fundamental design concept of combining the instruction code and the memory reference into one computer word. This contributes to speed and efficiency in the computer, but also limits the number of bits available for the memory reference. Bits 0 through 9 of the memory reference instructions are available for the memory address. Refer now to table 4-4 and note under the "Memory Reference Bits" column that the possible range of numbers using these bits is (in octal) 0000 through 1777. To form addresses any higher than 1777 requires the addition of bits listed under the "Page Bits" column.

4-49. In the computer, a reference to memory is implemented by transferring bits 0 through 9 of the instruction word from the T-register to the M-register during the fetch phase. The remaining bits, during the fetch phase, stay at the value present before the fetch phase began. (Optionally, these bits can be reset to zero for a reference to page 0; this is relatively simple to accomplish internally.) Thus the programmer must know if these bits currently agree with the corresponding bits of the address he wishes to reference. To assist the programmer in this task, the convention is established of dividing memory into blocks called "pages". Each block contains 2000 (octal) memory locations (or 1024 decimal). This block size is determined by the range of direct addressing capability (0000 through 1777), and each such block is assigned a "page number".

Table 4-4. Memory Pages

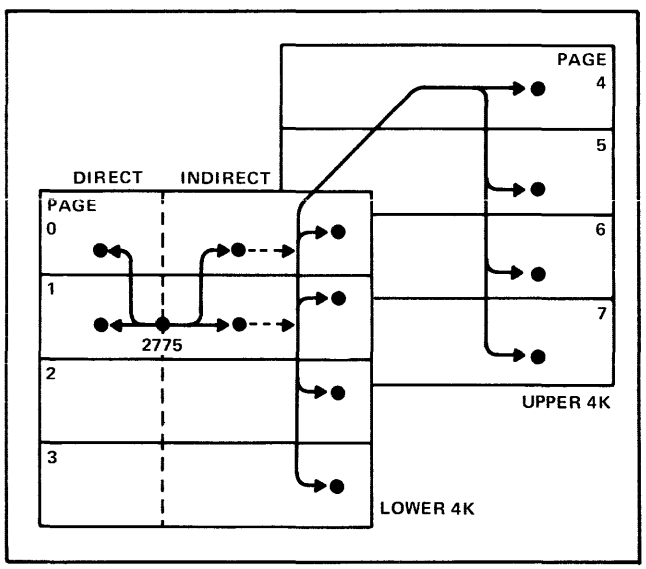
PAGE NO.	OCTAL ADDRESSES	COMPLETE BINARY ADDRESSES (M-REGISTER)										
		PAGE BITS						MEMORY REFERENCE BITS				
0	00000	(*)	0	0	0	0	0	0	0	0	0	0
	01777		0	0	0	0	1	1	1	1	1	1
1	02000		0	0	0	1	0	0	0	0	0	0
	03777		0	0	0	1	1	1	1	1	1	1
2	04000		0	0	0	1	0	0	0	0	0	0
	05777		0	0	0	1	1	1	1	1	1	1
3	06000		0	0	0	1	1	0	0	0	0	0
	07777		0	0	0	1	1	1	1	1	1	1
4	10000		0	0	1	0	0	0	0	0	0	0
	11777		0	0	1	0	0	1	1	1	1	1
5	12000		0	0	1	0	1	0	0	0	0	0
	13777		0	0	1	0	1	1	1	1	1	1
6	14000		0	0	1	1	0	0	0	0	0	0
	15777		0	0	1	1	0	1	1	1	1	1
7	16000		0	0	1	1	1	0	0	0	0	0
	17777		0	0	1	1	1	1	1	1	1	1

*Direct/Indirect bit does not form part of an address.

4-50. Identification of page numbers is simplified by considering the 5 page bits (see table 4-4) as a separate binary word. Thus 00000_2 is page 0; 00001_2 is page 1; 00010_2 is page 2; etc. Going back to the problem example in paragraph 4-46 (where it was stated that the ADA instruction in location 2775 could not reference location 4000), the situation can be analyzed as follows:

a. Current address is 000 010 111 111 101 (02775).
b. Page number (first five bits) is 00001_2 (page 1).
c. Desired reference is 000 100 000 000 000 (04000).
d. Page number (first five bits) is 00010_2 (page 2).

4-51. The desired reference requires a page change, or, in other words, bits 10 through 15 of the M-register must be altered, in addition to the usual alteration of bits 0 through 9. To do so requires use of a programming technique described under paragraph 4-55 (indirect references). A simpler technique of addressing another page (limited to page 0 only) is discussed first in the following paragraph. Figure 4-5 illustrates both methods by showing individual memory cells which are addressable from a location on page 1. This source location may be thought of as location 2775, the same example as in the preceding discussions. Page 1 is the current page.



2106-16

Figure 4-5. Direct and Indirect References to Other Pages

4-52. DIRECT REFERENCES.

4-53. The arrows going left from "location 2775" in figure 4-5 show that, without using an indirect address, an instruction at this point can reference a location on either the current page or page 0. This doubles the range of possible references for instructions which are located on any page other than page 0. Bit 10 of the instruction word is reserved for distinguishing which page is referenced (zero for page 0, or one for current page). This distinction must always be considered when coding any memory reference instruction, or an erroneous reference may be made. The memory reference bits alone are not sufficient to identify a location. For example, ADA 5777 and ADA 1777 (assuming that the program is operating in page 2) have identical memory reference bits:

ADA 5777: 0 100 01(1 111 111 111)
ADA 1777: 0 100 00(1 111 111 111)

4-54. Only bit 10, the zero/current indicator, can make the distinction. The "C" in the Consolidated Coding Table is a reminder that bit 10 must be coded a one when referencing current page. Otherwise it must be a zero for all memory reference instructions. Remember that bit 10 of the instruction word is not an address bit. Its function is to control bits 10 through 15 of the M-register: to either reset these bits to zero, or leave them alone. This provides an easy, direct access to information on page 0 from any other page, thus making page 0 useful for storage of data. Programs are generally stored on other pages (as the examples in this section are) in order to reserve page 0 for information which may be referenced frequently.

4-55. INDIRECT REFERENCES.

4-56. The arrows going right from "location 2775" in figure 4-5 show that, by using an indirect address in the first referenced location, any location in memory can then be accessed. As in the preceding paragraph, the initial reference (contained in the instruction word), can refer to a location on either the current page or page 0. Broken lines in figure 4-5 indicate this optional choice. Either way, the initial reference is simply an intermediate step to the final desired reference. Obviously an added machine operation (indirect phase) is required, as well as the added memory location. The means of telling the computer that this additional step is desired is to code a one in bit 15 of the instruction word. An "I" in the Consolidated Coding Table is a reminder to do this.

4-57. PROGRAM EXAMPLE.

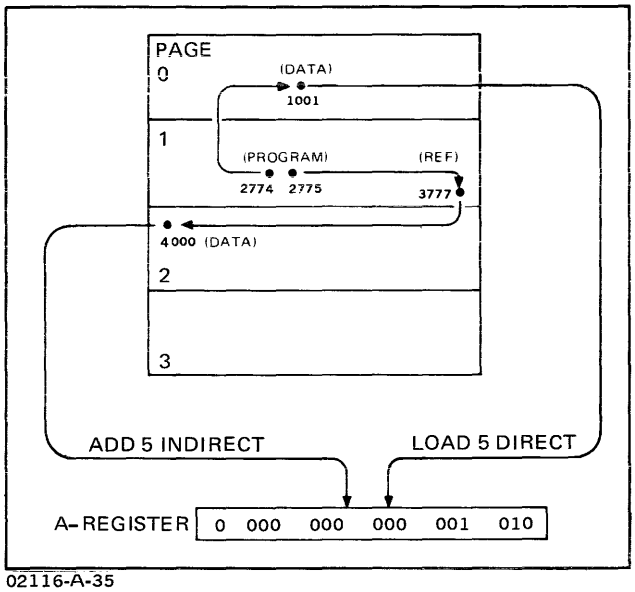
4-58. Table 4-5 lists a program illustrating both a direct reference to page 0 and an indirect reference to page 2. As before, the program itself operates approximately in the middle of page 1. This program differs from that of table 4-2 in that the data, instead of being stored on the current page (location 3001), now appears in two different locations: location 1001 on page 0, and location 4000 on page 2. Figure 4-6 shows in simplified form the referencing accomplished by this program.

4-59. LOADING THE PROGRAM. Unless memory has been disturbed, the program of table 4-5 can be loaded by making a few changes to the existing conditions of the computer on completion of the preceding procedures. (The reader, at this point in the text, should be able to load a complete program, given octal addresses and octal-coded contents; refer back to paragraph 4-33 if necessary.) Changes required are:

a. Load location 002774 with contents 061001.
b. Load location 003777 with contents 004000.

c. Load location 004000 with contents 000005.
d. Load location 001001 with contents 000005.

4-60. DIRECT REFERENCE. Set the Switch Register to the starting address (002774), and press LOAD ADDRESS. Remembering that only bits 0 through 9 of the word about to be read out of memory are transferred to the M-register, watch bit 10 of the M-register and press SINGLE CYCLE once. Bit 10, a page bit, has changed from a one to a zero, thus changing pages. This situation is shown in figure 4-6, where the instruction word in location 2774 is causing location 1001 to be addressed. The contents of location 1001 is known to be "5"; this will be loaded into the A-register in the next (execute) phase. Again watch bit 10 of the M-register and press SINGLE CYCLE. The page indication returns to page 1 to address the next instruction (in 2775), and the data (octal 5) is in the A-register. Referring to figure 4-6, an instruction on page 1 has commanded data from page 0 (by direct reference) to be put into the A-register.



02116-A-35
Figure 4-6. Examples of Interpage Referencing

4-61. INDIRECT REFERENCE. It was previously pointed out that a direct reference from location 2775 to 4000 is not possible. The method for making an indirect reference is to make an initial reference to a location on the current page, pick up a 15-bit address there, and use that address to reference location 4000 (refer to figure 4-6). Although the initial reference could be anywhere on the current page or page 0, location 3777 (which is immediately adjacent to location 4000) has been chosen to emphasize the concept of page boundaries.

4-62. Watching bits 11 and 10 of the M-register, press SINGLE CYCLE. These bits remain 01_2 (page 1) for the initial reference to location 3777 on the current page. Note that the computer has acknowledged the fact that indirect addressing is desired, since the INDIRECT light is on; this condition was specified by a one in bit 15 of the instruction word (now visible in the T-register). Again watching bits 11 and 10 of the M-register, press SINGLE CYCLE. These bits change to 10_2 (page 2) for the indirect reference to location 4000. Since bit 15 of the T-register is now a zero (not "indirect"), the EXECUTE indicator lights. This means that the next phase will execute the instruction, and the M-register will return to page 1 for the next instruction. Watching bits 11 and 10 of the M-register, press SINGLE CYCLE. These bits return to 01_2 to address location 2776. The remaining actions are the same as in table 4-3, steps 6, 7, and 8. Press SINGLE CYCLE three more times to complete the program.

4-63. JUMPS.

4-64. In all previous examples, although random references to various points in memory were made, the program itself (i.e., the list of instruction words) was located in a few consecutive locations in page 1. This strict sequential operation would be severely limiting for practical applications. Therefore provision must be made for the program to move freely throughout available memory. The jump instructions (JMP and JSB) provide this capability.

Table 4-5. Program for Interpage Referencing

ADDRESS	CONTENTS						REMARKS
	INSTRUCTION (OR DATA)	MEMORY REFERENCE	D/I	A/B	Z/C	OCTAL CODE	
002774	LDA	1001				061001	Get augend from page Zero, put in A.
002775	ADA	3777	I		C	143777	Add the addend specified by 3777.
002776	STA	3000			C	073000	Put answer in 3000.
002777	HLT					102000	Halt.
003000						-	Reserved for answer.
003777	4000					004000	Address of addend is 4000.
004000	5					000005	Data (on Page 2).
001001	5					000005	Data (on Page 0).

4-65. The essential difference between these two instructions is that the JMP (Jump) instruction unconditionally suspends operation at the currently used area of memory and continues operation in a new area, whereas JSB (Jump to Subroutine) provides a means of "remembering" the location where the jump command was given, thus enabling a return to that point at some later time. Table 4-6 illustrates both kinds of jumps by treating the program previously developed (add 5 + 5) as a subroutine, and adding a few preliminary instructions. These preliminary instructions represent the "main program"; for simplicity of instruction, several NOP (No Operation) instructions are inserted to represent a more lengthy sequence of working instructions.

4-66. The special considerations for referencing other pages, as covered in the preceding discussion, apply to the jump instructions. This means that the program can jump directly to any location on either current page or page 0, or indirectly to any location in memory. The program example in table 4-6 illustrates both a direct JMP and an indirect JMP, but only a direct JSB. An indirect JSB occurs in the same way as does the indirect JMP.

4-67. LOADING THE PROGRAM. If memory remains undisturbed from preceding procedures, the new program can be loaded simply by loading the "Octal Code" contents into the corresponding "Address" for those items not shaded in table 4-6. Otherwise it is necessary to load all 15 addresses listed in the table. Note that LOAD ADDRESS must be used three times, since three separate areas of memory are being loaded.

4-68. THE JMP INSTRUCTION. Set the Switch Register to the starting address (002100), and press LOAD ADDRESS. Assume that a working program has been running sequentially up to this point (i.e., the P-register increments by one on completion of each instruction). For example, watch the P-register and press SINGLE CYCLE. This causes execution of the first NOP instruction, and advances the P-register from 002100 to 002101. In location 2101 is the instruction to jump to location 2200. Since a direct jump is a one-phase instruction, the jump will be completed in the next operation. Watch the P-register and press SINGLE CYCLE, noting that the indication does not increment by one, but rather jumps from 002101 to 002200. If the intervening locations had contained instructions, those instructions would be omitted from the sequence of this program. Press SINGLE CYCLE two more times, and note that the P-register increments normally from the new operating point of 002200.

4-69. THE JSB INSTRUCTION. The P-register is now at the location (2202) which contains the instruction to jump to the subroutine which begins at location 2773. This subroutine, as the Remarks column states, is a procedure to add 5 plus 5. It is desired, upon completion of the subroutine, to return to the main program at the succeeding location (2203). It happens that the HLT instruction is located in 2203, but a program-continuing instruction could as well be stored there, and the program (P-register) would advance as usual to 2204, 2205, etc.

4-70. The JSB instruction, unlike JMP, requires two phases. The first phase (fetch) only references the location being jumped to; i.e., the P-register does not change in this phase. Watch the M-register and press SINGLE CYCLE noting that location 2773 is referenced, but the P-register still "remembers" the location (2202) where the jump command was given. The next phase will store the return address into the referenced location, and will complete the jump. Watch the P- and M-registers and press SINGLE CYCLE. Both registers now address the first instruction of

Table 4-6. Examples of Program Jumps

ADDRESS	CONTENTS						REMARKS
	INSTRUCTION (OR DATA)	MEMORY REFERENCE	D/I	A/B	Z/C	OCTAL CODE	
002100	NOP					000000	Program starts here (no operation).
002101	JMP	2200			C	026200	Jump to 2200.
002200	NOP					000000	No operation.
002201	NOP					000000	No operation.
002202	JSB	2773			C	016773	Jump to 5+5 subroutine at 2773.
002203	HLT					102000	Halt.
002773						-	Reserved for return address.
002774	LDA	1001				061001	Get augend from page Zero, put in A.
002775	ADA	3777	I		C	143777	Add the addend specified by 3777.
002776	STA	3000			C	073000	Put answer in 3000.
002777	JMP	2773	I		C	126773	Return to main program via 2773.
003000						-	Reserved for answer.
003777	4000					004000	Address of addend is 4000.
004000	5					000005	Data (on Page 2).
001001	5					000005	Data (on Page 0).

the subroutine location 2774. Note also that the T-register indicates the number 2203, the return address, which was stored into location 2773 in the phase just completed. This value is one higher than the location jumped from, since obviously a return to location 2202 would send the program right back into the subroutine, and it would loop continuously without ever reaching 2203.

4-71. Now press the SINGLE CYCLE pushbutton seven more times. This executes the three instructions of the subroutine, which are identical to the instructions of the previous program (table 4-5). The content of location 2777, however, is now an indirect jump via location 2773. Location 2773, remember, contains the return address. Watch the M-register and press SINGLE CYCLE; this references location 2773. Since the next phase will be an indirect phase (INDIRECT light is on), the content of the referenced location will be interpreted as an address. The indirect phase will complete the jump to that address. Watch the P- and M-registers and again press SINGLE CYCLE. These registers now address location 2203 of the main program, completing the jump out of the subroutine. Press SINGLE CYCLE to execute the HLT instruction contained in location 2203.

4-72. The preceding three paragraphs show how subroutines are accessed. By definition, a subroutine is a sequence of instructions designed to perform a single task, with provisions included to allow entry from any point in a program and return to the same point. The contents of locations 2773 through 2777 comprise a typical subroutine. The single task is an addition, and the entry and return requirement is guaranteed by storing the return address in location 2773 (a function of the JSB instruction) and by including an indirect jump via this location at the end of the subroutine (programmer's responsibility).

4-73. SUMMARY.

4-74. This volume has presented a basic instruction to how the computer operates, with equal emphasis on both hardware and programming. The succeeding three volumes present specialized descriptions on each of these two aspects. Volume Two describes the processor hardware in detail, and Volume Three deals with the input/output system hardware. Volume Four provides detailed information for programming of the computer with the aid of Hewlett-Packard software.

APPENDIX A

REFERENCE TABLES

Table A-1. Glossary of Terms Used in This Volume

absolute — Pertaining to an address fully defined by a memory address number, or to a program which contains such addresses (as opposed to one containing symbolic addresses).

accumulator — A register in which numbers are totaled or manipulated, or temporarily stored for transfers to and from memory or external devices.

add — Restrictive: "two's complement" addition of binary numbers. General: any arithmetic addition.

address — A number (noun) which identifies one location in memory. Also (verb), the process of directing the computer to read a specified memory location (synonymous with "reference").

address modification — A programming technique of changing the address referred to by a memory reference instruction, so that each time that particular instruction is executed, it will affect a different memory location.

address word — A computer word which contains only the address of a memory location.

ALGOL — A programming language that uses familiar arithmetic conventions instead of detailed instruction-by-instruction mnemonic coding as used with the Assembler.

alter — A modification of the contents of an accumulator or extend bit; e.g., clear, complement, or increment.

analog — Pertaining to information which can have continuously variable values, as opposed to digital information, which can be varied in degrees no smaller than the value of the least significant digit.

'and' — A logical operation in which the resultant quantity (or signal) is true if all of the input values are true, and is false if at least one of the input values is false.

A-register — One of the computer's two accumulator registers. These registers are used for arithmetic operations and for information transfers to and from device interfaces.

arithmetic logic — The circuitry involved in manipulating the information contained in a computer's accumulators.

arithmetic operation — Restrictive: a mathematical operation involving fundamental arithmetic (addition, subtraction, multiplication, division), specifically excluding logical and shifting operations. General: any manipulation of numbers.

Assembler — A program for HP computers which converts a program prepared in symbolic form (i.e., using defined symbols and mnemonics to represent instructions, addresses, etc.) to binary machine language.

base — The quantity of different digits used in a particular numbering system. The base in the binary numbering system is two; thus there are two digits (0 and 1). In the decimal system (base 10), there are ten digits (0 through 9).

base page — The lowest numbered page of a computer's memory. It can be directly addressed from any other page.

BASIC — An easily learned, conversational programming language using elementary mathematics.

Basic Binary Loader — A series of instructions for HP computers which will load, into memory, programs prepared with absolute addresses, using defined input devices.

Basic Control System — A collection of programs for HP computers which direct the loading, combining, library searching, debugging, and input/output procedures for programs generated by the user.

binary — Denoting the numbering system based on the radix two. Binary digits are restricted to the values 0 and 1.

binary-coded decimal — A coding method for representing each decimal digit (0-9) by specific combinations of four binary bits. For example, the 8-4-2-1 "bcd" code commonly used with HP computers represents "1" as 0001, and "9" as 1001.

binary point — The fractional dividing point of a binary numeral; equivalent to decimal point in the decimal numbering system.

binary program — A program (or its recorded form) in which all information is in binary machine language.

bit — A single digit in a binary number, or in the recorded representation of such a number (by hole punches, magnetic states, etc.). The digit can have one of only two values, 0 or 1.

bit density — A physical specification referring to the number of bits which can be recorded per unit of length or area.

Table A-1. Glossary (Continued)

bit-serial — One bit at a time, as opposed to bit-parallel in which all bits of a character can be handled simultaneously.

bistable — Pertaining to an electronic circuit having two stable states, controllable by external switching signals; analogous to an on-off switch.

B-register — One of the computer's two accumulator registers. These registers are used for arithmetic operations and for information transfers to and from device interfaces.

buffer — A register used for intermediate storage of information in the transfer sequence between the computer's accumulators and a peripheral device.

bus — A major electrical path connecting two or more electrical circuits.

carry — A digit, or equivalent signal, resulting from an arithmetic operation which causes a positional digit to equal or exceed the base of the effective numbering system.

character — The general term to include all symbols such as alphabetic letters, numerals, punctuation marks, mathematical operators, etc. Also, the coded representation of such symbols.

checkerboard — An alternating pattern of zeros and ones stored in a computer for testing purposes.

clear — Reset; the binary "zero" condition.

code — A system of symbols which can be used by machines such as a computer, and which in specific arrangements have a special external meaning.

communication system — A computer system having facilities for long-distance transfers of information between remote and central stations.

comparator — An instrument for comparing digitized measurements against presettable upper and lower limits, and giving an indication of the comparison result.

compiler — A language translation program, used to transform symbols meaningful to a human operator to codes meaningful to a computer. More restrictively, a program which translates a machine-independent source language into the machine language of a specific computer, thus excluding assemblers.

computation — The processing of information within the computer.

computer (digital) — An electronic instrument capable of accepting, storing, and arithmetically manipulating information, which includes both data and the controlling program. The information is handled in the form of coded binary digits (0 and 1), represented by dual voltage levels, magnetic states, punched holes, etc.

computer word — See "word".

configuration — The arrangement of either hardware instruments or software routines when combined to operate as a system.

contents — The information stored in a register or a memory location.

control bit — A signal, or the stored indication of this signal, which controls the transfer of information to and from peripheral devices associated with the computer.

core — The smallest element of a core storage memory module. It is a ring of ferrite material that can be magnetized in clockwise or counterclockwise directions to represent the two binary digits, 0 and 1.

current page — The memory page comprising all those locations which are on the same page as a given instruction.

data acquisition — The gathering, measuring, digitizing, and recording of continuous form (analog) information.

data reduction — The transformation of raw information gathered by measuring or recording equipment into a more condensed, organized, or useful form.

data word — A computer word consisting of a number, a fact, or other information which is to be processed by the computer.

debug — Check for and correct errors in a program.

decimal — Denoting the numbering system based on the radix ten.

decrement — To change the value of a number in the negative direction. If not otherwise stated, a decrement by one is usually assumed.

device — An electronic or electromechanical instrument. Most commonly implies measuring, reading, or recording equipment.

diagnostic — (adj) Relating to test programs for detection of errors in the functioning of hardware or software, or the messages resulting from such tests. Also (noun), the test program or message itself.

digital voltmeter — An electronic voltage measuring device which provides a readout in digital form on the instrument panel, and commonly (essential for computer purposes) also codes the measurement result in binary-coded decimal form as an electrical output.

direct memory access — A means of transferring a block of information words directly between an external device

Table A-1. Glossary (Continued)

and the computer's memory, bypassing the need for repeating a service routine for each word. This method greatly speeds the transfer process.

disable — A signal condition which prohibits some specific event from proceeding.

disc storage — A means of storing binary digits in the form of magnetized spots on a rotating circular plate coated with a magnetic material. The information is stored and retrieved by read-write heads positioned over the surface of the disc.

documentation — Manuals and other printed materials (tables, listings, diagrams, etc.) which provide instructive information for usage and maintenance of a manufactured product, including both hardware and software.

double-length word — A word, due to its length, which requires two computer words to represent it. Double-length words are normally stored in two adjacent memory locations.

driver — An input/output routine to provide automatic operation of a specific device with the computer.

dump — To record memory contents on an external medium (e.g., tape).

effective address — The address of a memory location ultimately affected by a memory reference instruction. It is possible for one instruction to go through several indirect addresses to reach the effective address.

electronic counter — An electronic instrument used to measure physical quantities by specially controlled counting of electrical pulses.

enable — A signal condition which permits some specific event to proceed, whenever it is ready to do so.

'exclusive or' — A logical operation in which the resultant quantity (or signal) is true if one of the two input values is true, and is false if the input values are both true or both false.

execute — To fully perform a specific operation, such as would be accomplished by an instruction or a program.

execute phase — A predetermined state of the internal computer logic which causes the computer to interpret as data the information read out of memory during a memory cycle.

exit sequence — A series of instructions to conclude operation in one area of a program and to move to another area.

Extend — A one-bit register in the computer, which extends the effective length of the A- or B-registers to 17 bits for certain additions and rotations.

fetch phase — A predetermined state of the internal computer logic which causes the computer to interpret as an instruction the information read out of memory during a memory cycle.

fixed point — A numerical notation in which the fractional point (whether decimal, octal, or binary) appears at a constant, predetermined position. Compare with floating point.

flag bit — A signal, or the stored indication of this signal, which indicates the readiness of a peripheral device of the computer to transfer information.

flip-flop — An electronic circuit having two stable states, and thus capable of storing a binary digit. Its states are controlled by signal levels at the circuit input, and are sensed by signal levels at the circuit output.

floating point — A numerical notation in which the integer and the exponent of a number are separately represented (frequently by two computer words), so that the implied position of the fractional point (decimal, octal, or binary) can be freely varied with respect to the integer digits. Compare with fixed point.

flowchart — A diagram representing the operation of a computer program.

format — A predetermined arrangement of bits or characters.

Formatter — A program which provides the linkage between FORTRAN read/write statements and the Basic Control System's Input/Output Control program, with any appropriate conversions.

FORTRAN — A programming language (or the compiler which translates this language) which permits programs to be written in a form resembling algebra, rather than in detailed instruction-by-instruction form (as for assemblers.

gate — An electronic circuit capable of performing logical functions such as "and", "or", "nor", etc.

hardware — Electronic or electromechanical components, instruments, or systems.

hardware diagnostics — A collection of programs for the computer designed to assist in the identification of hardware malfunctions.

high core — Core memory locations having high-numbered addresses.

Table A-1. Glossary (Continued)

'inclusive or' — A logical operation in which the resultant quantity (or signal) is true if at least one of the input values is true, and is false if the input values are all false.

increment — To change the value of a number in the positive direction. If not otherwise stated, an increment by one is usually assumed.

incremental magnetic tape — A form of magnetic tape recording in which the recording transport advances by small increments (e.g. 0.005"), stopping the tape advancement long enough to record one character at the spot located under the recording head.

indirect address — The address initially specified by an instruction when it is desired to use that location to re-direct the computer to some other location to find the "effective address" for the instruction.

indirect phase — A predetermined state of the internal computer logic which causes the computer to interpret as an address the information read out of memory during a memory cycle.

information — A unit or set of knowledge represented in the form of discrete "words", consisting of an arrangement of symbols or (so far as the digital computer is concerned) binary digits.

inhibit — To prevent a specific event from occurring.

initialize — The procedure of setting various parts of a stored program to starting values, so that the program will behave the same way each time it is repeated. The procedures are included as part of the program itself.

input — Information transferred from a peripheral device into the computer. Also can apply to the transfer process itself.

input/output — Relating to the equipment or method used for transmitting information into or out of the computer.

input/output channel — The complete input or output facility for one individual device or function, including its assigned position in the computer, the interface circuitry, and the external device.

Input/Output Control — A program of the Basic Control System which provides linkage between the input/output requests of a user program and the appropriate drivers.

input/output system — The circuitry involved in transferring information between the computer's accumulators and its peripheral devices.

instruction — A written statement, or the equivalent computer-acceptable code, which tells the computer to execute a specified single operation.

instruction code — The arrangement of binary digits which tell the computer to execute a particular instruction.

instruction logic — The circuitry involved in moving binary information between registers, memory, and buffers in prescribed manners, according to instruction codes.

Instruction Register — An internal 6-bit register of the computer, which forms part of its instruction logic. The Instruction Register receives the 6 most significant bits of the T-register when each new instruction is read out of memory, and retains these bits for instruction identification. It is not usually considered to be a "working register".

instruction word — A computer word containing an instruction code. The code bits may occupy all or (as in the case of memory reference instruction words) only part of the word.

interface — The connecting circuitry which links the central processor of a computer system to its peripheral devices.

interrupt — The process, initiated by an external device, which causes the computer to interrupt a program in progress, generally for the purpose of transferring information between that device and the computer.

interrupt location — A memory location whose contents (always an instruction) are executed upon interrupt by a specific device.

interrupt phase — A predetermined state of the internal computer logic which causes the computer to suspend operation of a program in progress, and branch to a specific service routine.

jump — An instruction which breaks the strict sequential location-by-location operation of a program, and directs the computer to continue at another specified location anywhere in memory.

label — Any arrangement of symbols, usually alphanumeric, used in place of an absolute memory address in computer programming.

language — The set of symbols, rules, and conventions used to convey information, either at the human level or at the computer level.

library routine — A routine designed to accomplish some commonly used mathematical function, and kept permanently available on a library program tape (e.g., HP Relocatable Library).

load — Put information into (memory, a register, etc.). Also (e.g., loading tape), to put the information medium into the appropriate device.

loader — A program designed to assist in transferring information from an external device into a computer's memory.

Table A-1. Glossary (Continued)

location — A group of storage elements in the computer's memory (e.g., 17 cores in the memory module), which can store one computer word. Each such location is identified by a number (address) to facilitate storage and retrieval of information in selectable locations.

logical operation — A mathematical process based on the principles of truth tables; e.g., "and", "inclusive or" and "exclusive or" operations.

logic diagram — A diagram that represents the detailed internal functioning of electronic hardware, using binary logic symbols rather than electronic component symbols (see "schematic diagram").

logic equation — A written mathematical statement, using symbols and rules derived from Boolean algebra. Specifically (hardware design), a means of stating the conditions required to obtain a given signal.

loop — A sequence of instructions in which the last instruction is a jump back to the first instruction.

low core — Core memory locations having low-numbered addresses.

machine — Pertaining to the computer hardware (e.g., machine timing, machine language).

machine language — The form of coded information (consisting of binary digits) which can be directly accepted and used by the computer. Other languages require translation to this form, generally with the aid of translation programs (assemblers and compilers).

machine timing — The regular cycle of events in the operation of internal computer circuitry. The actual events will differ for various processes, but the timing is constant through each recurring cycle.

macroinstruction — An instruction, similar in binary coding to the computer's basic machine language instructions, which is capable of producing a variable number of machine language instructions.

magnetic tape recording — A means of recording information on a strip of magnetic coated material, such that binary bits can be represented by reversals of the direction of magnetization.

magnitude — That portion of a computer word which indicates the absolute value of a number, thus excluding the sign bit.

math routine — A program designed to accomplish a single mathematical function.

media conversion — The transferral of recorded information from one recording medium (e.g., punched paper tape, magnetic tape, etc.) to another recording medium.

memory — An organized collection of storage elements (e.g., ferrite cores), into which a unit of information consisting of a binary digit can be stored, and from which it can later be retrieved. Also, a device not necessarily having individual storage elements, but which has the same storage and retrieval capabilities (e.g., magnetic discs).

memory cycle — That portion of the computer's internal timing during which the contents of one location of memory are read out (into the Transfer Register) and written back into that location.

memory module — A complete segment of core storage, capable of storing a definable number of computer words (e.g., 8192 words in the computer memory module). Computer storage capacity is incremental by modules, and is frequently rounded off and abbreviated as "8K" (i.e., 8192 or approximately 8000 words), "16K" (16,384 or 16,000), "24K", etc.

memory protect — A means of preventing inadvertent alteration of a selectable segment of memory.

memory reference — The address of the memory location specified by a memory reference instruction; i.e., the location affected by the instruction.

merge — "inclusive or".

microinstruction — An instruction which forms part of a larger, composite instruction.

mnemonic — An abbreviation or arrangement of symbols used to assist human memory. For example, "STB" calls to mind the term "Store B-register" much more readily than would, say, "instruction 74".

M-register — The memory address register of the computer; i.e., the register which controls the access to each memory location.

multi-level indirect — Indirect addressing using two or more indirect addresses in sequence to find the effective address for the current instruction.

multiple-precision — Referring to arithmetic in which the computer, for greater accuracy, uses two or more words to represent one number.

Mylar — A DuPont trademark for a polyester film used as a more durable medium (in place of paper tape) for punched tape records, and as a base for magnetic tape.

nine's complement — A number so modified that the addition of the modified number and its original value, plus one, will equal an even power of ten. A nine's complement number is obtained mathematically by subtracting the original value from a string of 9's.

Table A-1. Glossary (Continued)

non-return to zero — A technique of magnetic tape recording in which the recording device does not turn off the magnetizing flux between recording of individual characters. The flux is always at saturation level during recording, and bits are indicated by reversals of flux polarity.

nuclear scaler — An electronic instrument used to detect and count nuclear events, such as gamma ray measurements.

octal — Denoting a numbering system based on the radix eight. Octal digits are restricted to the values 0 through 7.

octal code — A notation for writing machine language programs with the use of octal numbers instead of binary numbers.

octal point — The fractional dividing point of an octal numeral; equivalent to decimal point in the decimal numbering system.

off line — Pertaining to the operation of peripheral equipment not under control of the computer.

one's complement — A number so modified that the addition of the modified number and its original value, plus one, will equal an even power of two. A one's complement number is obtained mathematically by subtracting the original value from a strong of 1's, and electronically by inverting the states of all binary bits in the number.

on line — Pertaining to the operation of peripheral equipment under computer control.

output — Information transferred from the computer to a peripheral device. Also can apply to the transfer process itself.

output coupler — An instrument which provides the interconnecting circuitry between a measuring instrument and a recording instrument.

Overflow — A one-bit register in the computer, which indicates that the result of an addition in the A- or B-register has exceeded the maximum possible signed value (+32767 or -32768, decimal). The addition result will therefore be missing one or more significant bits

packed word — A computer word containing two or more independent units of information. This is done to conserve storage when information requires relatively few bits of the computer word.

page — An artificial division of memory consisting of a fixed number of locations, dictated by the direct addressing range of memory reference instructions.

page zero — The memory page which includes the lowest numbered memory addresses.

parity bit — A supplementary bit added to an information word to make the total of one-bits be always either odd or even. This permits checking the accuracy of information transfers.

pass — The complete process of reading a set of recorded information (one tape, one set of cards, etc.) through an input device, from beginning to end.

peripheral device — An instrument or machine electrically connected to the computer, but which is not part of the computer itself.

phase — One of several specific states of the internal computer logic, usually set up by instructions being executed, to determine how the computer should interpret information read out of memory.

photoelectric reader — An input device which senses characters (on punched tape, cards, pages, etc.) by optical light strobe and detection circuits.

plane (bit) — An arrangement of ferrite cores on a matrix of control and sensing wires.

power failure control — A means of sensing primary power failure so that a special routine may be executed in the finite period of time available before the regulated dc supplies discharge to unusable levels. The special routine may be used to preserve the state of a program in progress, or to shut down external processes.

P-register — The program counter register of the computer; i.e., the register which keeps track of (or "counts") the stored locations of the instructions in a program being executed.

Prepare Control System — A program designed to assist in the preparation of a Basic Control System program, to a specified arrangement of input/output devices.

priority — The automatic regulation of events so that chosen actions will take precedence over others in cases of timing conflict.

program — A plan for the solution of a problem by a computer, and consisting of a sequence of computer instructions.

process control — Automatic control of manufacturing processes by use of a computer.

processor — The central unit of a computer system (i.e., the device which accomplishes the arithmetic manipulations), exclusive of peripheral devices. Frequently (when used as adjective) also excludes interface components, even though normally contained within the processor unit; thus processor options exclude interface (input/output) options.

Table A-1. Glossary (Continued)

program listing — A printed record (or equivalent binary-output program) of the instructions in a program.

programmer — A person who writes computer programs. Also hardware, an interface card or instrument which sets up (or programs) the various functions of one measuring instrument.

programming — The process of creating a program.

pseudo-instruction — A symbolic statement, similar to assembly language instructions in general form, but meaningful only to the program containing it, rather than to the computer as a machine instruction.

punched tape — A strip of tape, usually paper, on which information is represented by coded patterns of holes punched in columns across the width of the tape. Commonly, there are 8 hole positions (channels) across the tape.

read — The process of transferring information from an input device into the computer. Also, the process of taking information out of the computer's memory (see "memory cycle").

real time — Time elapsed between events occurring externally to the computer. A computer which accepts and processes information from one such event and is ready for new information before the next event occurs is said to operate in a "real-time environment".

reference — Shortened form of "memory reference".

register — An array of hardware binary circuits (flip-flops, switches, etc.) for temporary storage of information. Unlike mass storage devices such as memory cores, registers can be wired to permit flexible control of the contained information, for arithmetic operations, shifts, transfers, etc.

relocatable — Pertaining to programs whose instructions can be loaded into any stated area of memory.

Relocatable Library — A collection of programs for HP computers to provide the user with commonly used mathematical and formatting routines.

Relocating Loader — A computer program capable of loading and combining relocatable programs (i.e., programs having symbolic rather than absolute addresses).

reset — A signal condition representing a binary zero.

rotate — A positional shift of all bits in an accumulator (and possibly an extend bit as well), with those bits lost off one end of the accumulator "rotated" around to enter vacated positions at the other end.

routine — A program or program segment designed to accomplish a single function.

sampling — The process of taking a measurement of a signal existing at a measuring instrument's input during a short (sample) period. The length of the sample period is a predetermined function of the measuring instrument.

scanner — A device for sequentially switching multiple signal sources to one measuring or recording instrument.

schematic diagram — A diagram that represents the detailed internal electrical circuit arrangement of electronic hardware, using conventional electronic component symbols.

select code — A number assigned to input/output channels for purposes of identification in information transfers between the computer and external devices.

service routine — A sequence of instructions designed to accomplish the transfer of information between a particular device and the computer.

set — A signal condition representing a binary one.

seven's complement — A number so modified that the addition of the modified number and its original value, plus one, will equal an even power of eight. A seven's complement number is obtained mathematically by subtracting the original value from a string of 7's.

shift — Restrictive (arithmetic shift): to multiply or divide the magnitude portion of a word (bits 0 through 14) by a power of two, using a positional shift of these bits. General: any positional shift of bits.

sign — The algebraic plus or minus indicator for a mathematical quantity. Also, the binary digit or electrical polarity representing same.

significant digit — A digit so positioned in a numeral as to contribute a definable degree of precision to the numeral. In conventional written form, the most significant digit in a numeral is the leftmost digit, and the least significant digit is the rightmost digit.

skip — An instruction which causes the computer to omit the instruction in the immediately following location. A skip is usually arranged to occur only if certain specified conditions are sensed and found to be true, thus allowing various decisions to be made.

Table A-1. Glossary (Continued)

software — Computer programs. Also, the tapes or cards on which the programs are recorded.

software package — A complete collection of related programs, not necessarily combined as a single entity.

source program — A program (or its recorded form) written in some programming language other than machine language and thus requiring translation. The translated form is the "object program".

starting address — The address of a memory location in which is stored the first instruction of a given program.

statement — An instruction in any computer-related language other than machine language.

store — To put information into a memory location, register, or device capable of retaining the information for later access.

subroutine — A sequence of instructions designed to perform a single task, with provisions included to allow some other program to cause execution of the task sequence as if it were part of its own program.

symbolic address — A label assigned in place of absolute numeric addresses, usually for purposes of relocation (see "relocatable").

Symbolic Editor — A program for HP computers which is used to add, delete, or correct selectable portions of any symbolic program.

symbolic file — A recorded collection of computer words, with a symbolic address assigned to each word.

system — An assembly of units (e.g., hardware instruments or software routines), combined to work as a larger integrated unit having the capabilities of all the separate units.

System Input/Output (software) — A collection of input/output programs to add input/output capability to HP Fortran, Assembler, and Symbolic Editor, and to some user programs.

time period — The smallest division of time in the computer's internal timing cycle (see "machine timing").

T-register — The Transfer Register of the computer; i.e., the register which directly receives words from memory, and directly applies words to memory.

truth table — A table listing all possible configurations and resultant values for any given Boolean algebra function.

two's complement — A number so modified that the addition of the modified number and its original value will equal an even power of two. A two's complement number is obtained mathematically by subtracting the original value from an appropriate power of the base two (i.e., from 1_1, 10_2, 100_2, etc.), and electronically by inverting the states of all binary bits in the number and adding one (complement and increment).

updated program — A program to which additions, deletions, or corrections have been made.

user — The person or persons who program and operate a particular computer.

utility routine — A standard routine to assist in the operation of the computer (e.g., device drivers, sorting routines, etc.) as opposed to mathematical (library) routines.

waiting loop — A sequence of instructions (frequently only two) which are repeated indefinitely until a desired external event occurs, such as the receipt of a Flag signal.

word — A set of binary digits handled by the computer as a unit of information. Its length is determined by hardware design; e.g., the number of cores per location, and the number of flip-flops per register.

working register — A register whose contents can be modified under control of a program. Thus a register consisting of manually-operated switches is not considered a working register.

write — The process of transferring information from the computer to an output device. Also, the process of storing (or restoring) information into the computer's memory (see "memory cycle").

Table A-2. Mnemonics and Abbreviations Used in This Volume

A	Ampere
	A-register (A accumulator)
ac	Alternating current
ADA	Add to A
ADB	Add to B
ADF	Add function
ALF	Rotate A left four places
ALR	A left shift, clear sign
ALS	A left shift
AND	"And" instruction
ANF	"And" function
ARS	A right shift
ASA	American Standards Association
ASG	Alter-skip group
ASR	Automatic send-receive
B	B-register (B accumulator)
bcd (BCD)	Binary-coded decimal
BCS	Basic control system
BLF	Rotate B left four places
BLR	B left shift, clear sign
BLS	B left shift
bpi	Bits per inch
BRS	B right shift
BTU/hr	British thermal units, per hour
C	Centigrade
	Clear (flag or overflow)
	Control (bit or signal)
	Current page (page addressing)
CCA	Clear and complement A
CCB	Clear and complement B
CCE	Clear and complement extend
CLA	Clear A
CLB	Clear B
CLC	Clear control
CLE	Clear extend
CLF	Clear flag
CLO	Clear overflow
CMA	Complement A
CMB	Complement B
CME	Complement extend
CMF	Complement function
CPA	Compare to A
CPB	Compare to B
C16	Bit 16 carry
D	Direct (addressing)
	Disable (microinstruction group)
dc	Direct current
DMA	Direct memory access
E	Enable (microinstruction group)
	Extend
e.g.	For example (exempli gratia)
ELA	Rotate extend left with A
ELB	Rotate extend left with B
EOF	"Exclusive or" function
ERA	Rotate extend right with A
ERB	Rotate extend right with B
F	Fahrenheit
	Flag (bit or signal)
H	Hold (flag or overflow)
HLT	Halt
HP	Hewlett-Packard
Hz	Hertz
I	Indirect (addressing)
	I-register (instruction register)
i.e.	That is (id est)
in.	Inch
INA	Increment A
INB	Increment B
I/O	Input/output
IOF	"Inclusive or" function
IOG	Input/output group
IOR	"Inclusive or" instruction
ips	Inches per second
IR	Instruction register
ISZ	Increment, skip if zero
JMP	Jump
JSB	Jump to subroutine
K	Kilo (thousand)
kg	Kilograms
lb	Pound
LDA	Load (memory) into A
LDB	Load (memory) into B
LIA	Load input into A
LIB	Load input into B
M	M-register (memory address)
mA	Milliamperes
MAC	Macroinstruction
MHz	Megahertz
MIA	Merge into A
MIB	Merge into B
ms	Milliseconds
mV	millivolts
NOP	No operation
oct	Octal
OTA	Output from A
OTB	Output from B
OVF	Overflow flip-flop
P	P-register (program counter)
PH	Phase
RAL	Rotate A left
RAR	Rotate A right
RB	R-bus
RBL	Rotate B left
RBR	Rotate B right
RL	Rotate left
RLL	Rotate left to least significant bit

Table A-2. Mnemonics and Abbreviations Used in This Volume (Continued)

RRS	Rotate right to sign bit	STA	Store A
RSS	Reverse skip sense	STB	Store B
		STC	Set control
sec	Seconds	STF	Set flag
SEZ	Skip if extend is zero	STO	Set overflow
SFC	Skip if flag is clear	SZA	Skip if A is zero
SFS	Skip if flag is set	SZB	Skip if B is zero
SIO	System input/output		
SKF	Skip on flag (signal)	T	T-register (transfer register)
SL	Shift left		Time periods
SLA	Skip if least significant bit of A is zero	TB	T-bus
SLB	Skip if least significant bit of B is zero	TR	T-register
SLM	Shift left magnitude		
SOC	Skip if overflow clear	V	Volts
SOS	Skip if overflow set	Vac	Volts (alternating current)
SRG	Shift-rotate group		
SRM	Shift right magnitude	XOR	"Exclusive or" instruction
SSA	Skip if sign of A is zero		
SSB	Skip if sign of B is zero	Z	Page zero

Table A-3. Powers of Two

1	0	1 0
2	1	0 5
4	2	0 25
8	3	0 125
16	4	0 062 5
32	5	0 031 25
64	6	0 015 625
128	7	0 007 812 5
256	8	0 003 906 25
512	9	0 001 953 125
1 024	10	0 000 976 562 5
2 048	11	0 000 488 281 25
4 096	12	0 000 244 140 625
8 192	13	0 000 122 070 312 5
16 384	14	0 000 061 035 156 25
32 768	15	0 000 030 517 578 125
65 536	16	0 000 015 258 789 062 5
131 072	17	0 000 007 629 394 531 25
262 144	18	0 000 003 814 697 265 625
524 288	19	0 000 001 907 348 632 812 5
1 048 576	20	0 000 000 953 674 316 406 25
2 097 152	21	0 000 000 476 837 158 203 125
4 194 304	22	0 000 000 238 418 579 101 562 5
8 388 608	23	0 000 000 119 209 289 550 781 25
16 777 216	24	0 000 000 059 604 644 775 390 625
33 554 432	25	0 000 000 029 802 322 387 695 312 5
67 108 864	26	0 000 000 014 901 161 193 847 656 25
134 217 728	27	0 000 000 007 450 580 596 923 828 125
268 435 456	28	0 000 000 003 725 290 298 461 914 062 5
536 870 912	29	0 000 000 001 862 645 149 230 957 031 25
1 073 741 824	30	0 000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0 000 000 000 465 661 287 307 739 257 812 5
4 294 967 296	32	0 000 000 000 232 830 643 653 869 628 906 25
8 589 934 592	33	0 000 000 000 116 415 321 826 934 814 453 125
17 179 869 184	34	0 000 000 000 058 207 660 913 467 407 226 562 5
34 359 738 368	35	0 000 000 000 029 103 830 456 733 703 613 281 25
68 719 476 736	36	0 000 000 000 014 551 915 228 366 851 806 640 625
137 438 953 472	37	0 000 000 000 007 275 957 614 183 425 903 320 312 5
274 877 906 944	38	0 000 000 000 003 637 978 807 091 712 951 660 156 25
549 755 813 888	39	0 000 000 000 001 818 989 403 545 856 475 830 078 125
1 099 511 627 776	40	0 000 000 000 000 909 494 701 772 928 237 915 039 062 5
2 199 023 255 552	41	0 000 000 000 000 454 747 350 886 464 118 957 519 531 25
4 398 046 511 104	42	0 000 000 000 000 227 373 675 443 232 059 478 759 765 625
8 796 093 022 208	43	0 000 000 000 000 113 686 837 721 616 029 739 379 882 812 5
17 592 186 044 416	44	0 000 000 000 000 056 843 418 860 808 014 869 689 941 406 25
35 184 372 088 832	45	0 000 000 000 000 028 421 709 430 404 007 434 844 970 703 125
70 368 744 177 664	46	0 000 000 000 000 014 210 854 715 202 003 717 422 485 351 562 5
140 737 488 355 328	47	0 000 000 000 000 007 105 427 357 601 001 858 711 242 675 781 25
281 474 976 710 656	48	0 000 000 000 000 003 552 713 678 800 500 929 355 621 337 890 625
562 949 953 421 312	49	0 000 000 000 000 001 776 356 839 400 250 464 677 810 668 945 312 5
1 125 899 906 842 624	50	0 000 000 000 000 000 888 178 419 700 125 232 338 905 334 472 656 25
2 251 799 813 685 248	51	0 000 000 000 000 000 444 089 209 850 062 616 169 452 667 236 328 125
4 503 599 627 370 496	52	0 000 000 000 000 000 222 044 604 925 031 308 084 726 333 618 164 062 5
9 007 199 254 740 992	53	0 000 000 000 000 000 111 022 302 462 515 654 042 363 166 809 082 031 25
18 014 398 509 481 984	54	0 000 000 000 000 000 055 511 151 231 257 827 021 181 583 404 541 015 625
36 028 797 018 963 968	55	0 000 000 000 000 000 027 755 575 615 628 913 510 590 791 702 270 507 812 5

Table A-4. Consolidated Coding Table

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEMORY REFERENCE INSTRUCTIONS															
D/I	AND		001	0	Z/C	← Memory Address →									
D/I	XOR		010	0	Z/C										
D/I	IOR		011	0	Z/C										
D/I	JSB		001	1	Z/C										
D/I	JMP		010	1	Z/C										
D/I	ISZ		011	1	Z/C										
D/I	AD*		100	A/B	Z/C										
D/I	CP*		101	A/B	Z/C										
D/I	LD*		110	A/B	Z/C										
D/I	ST*		111	A/B	Z/C										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SHIFT-ROTATE GROUP INSTRUCTIONS															
0	SRG		000	A/B	0	D/E	*LS		000	†CLE	D/E	‡SL*	*LS		000
				A/B	0	D/E	*RS		001		D/E		*RS		001
				A/B	0	D/E	R*L		010		D/E		R*L		010
				A/B	0	D/E	R*R		011		D/E		R*R		011
				A/B	0	D/E	*LR		100		D/E		*LR		100
				A/B	0	D/E	ER*		101		D/E		ER*		101
				A/B	0	D/E	EL*		110		D/E		EL*		110
				A/B	0	D/E	*LF		111		D/E		*LF		111
				NOP	000				000		000				000
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALTER-SKIP GROUP INSTRUCTIONS															
0	ASG		000	A/B	1	CL* 01	CLE 01			SEZ	SS*	SL*	IN*	SZ*	RSS
				A/B	1	CM* 10	CME 10								
				A/B	1	CC* 11	CCE 11								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAC AND INPUT/OUTPUT INSTRUCTIONS															
1	MAC		000	A/B	0										
1	IOG		000	A/B	1	H/C	HLT		000	← Select Code →					
					1	0	STF		001						
					1	1	CLF		001						
					1	0	SFC		010						
					1	0	SFS		011						
				A/B	1	H/C	MI*		100						
				A/B	1	H/C	LI*		101						
				A/B	1	H/C	OT*		110						
				0	1	H/C	STC		111						
				1	1	H/C	CLC		111						
					1	0	STO		001		000			001	
					1	1	CLO		001		000			001	
					1	H/C	SOC		010		000			001	
					1	H/C	SOS		011		000			001	

Notes:
1) * = A or B. Use with bit 11 as 0 (A-Register) or 1 (B-Register).
2) D/I, A/B, Z/C, D/E, H/C coded: 0/1.
3) †CLE: Only this bit is required.
4) ‡SL*: Only this bit and bit 11 (A/B as applicable) are required.

EUROPE

AUSTRIA
Unilabor GmbH
Wissenschaftliche Instrumente
Rummelhardtgasse 6
P.O. Box 33
A-1095 **Vienna**
Tel: (222) 42 61 81, 43 13 94
Cable: LABORINSTRUMENT Vienna
Telex: 75 762

BELGIUM
Hewlett-Packard S.A. Benelux
348 Boulevard du Souverain
B-1160 **Brussels**
Tel: (02) 722240
Cable: PALOBEN Brussels
Telex: 23 494

DENMARK
Hewlett-Packard A/S
Datavej 38
DK-3460 **Birkerod**
Tel: (01) 81 66 40
Cable: HEWPACK AS
Telex: 66 40

Hewlett-Packard A/S
Torvet 9
DK-8600 **Silkeborg**
Tel: (06)-82-71-66

FINLAND
Hewlett-Packard Oy
Bulevardi 26
P.O. Box 12185
Helsinki 12
Tel: 13-730
Cable: HEWPACKOY-Helsinki
Telex: 12-1563

FRANCE
Hewlett-Packard France
Quartier de Courtaboeuf
Boite Postale No. 6
F-91 **Orsay**
Tel: 1-920 88 01
Cable: HEWPACK Orsay
Telex: 60048

Hewlett-Packard France
4 Quai des Etroits
F-69 **Lyon** 5ème
Tel: 78-42 63 45
Cable: HEWPACK Lyon
Telex: 31617

Hewlett-Packard France
29 rue de la Gara
F-31 **Blagnac**
Tel: (61) 85 82 29
Telex: 51957

GERMANY
Hewlett-Packard Vertriebs-GmbH
Berliner Strasse 117
Postfach 560/40
D6 Nieder-Eschbach/Ffm 56
Tel: (0611) 50 10 64
Cable: HEWPACKSA Frankfurt
Telex: 41 32 49 FRA

Hewlett-Packard Vertriebs-GmbH
Wilmersdorfer Strasse 113/114
D-1000 **Berlin W.** 12
Tel: (0311) 3137046
Telex: 18 34 05

Hewlett-Packard Vertriebs-GmbH
Herrenbergerstrasse 110
D7030 **Böblingen,** Württemberg
Tel: (07031) 66 72 86
Cable: HEPAG Böblingen
Telex: 72 65 739

Hewlett-Packard Vertriebs-GmbH
Vogelsanger Weg 38
D4 **Düsseldorf**
Tel: (0211 63 80 31/35
Telex: 85/86 533

Hewlett-Packard Vertriebs-GmbH
Wendenstr. 23
D2 **Hamburg** 1
Tel: (0411) 24 05 51/52
Cable: HEWPACKSA Hamburg
Telex: 21 53 32

Hewlett-Packard Vertriebs-GmbH
Reginfriedstrasse 13
D8 **München** 9
Tel: (0811) 69 59 71/75
Cable: HEWPACKSA München
Telex: 52 49 85

GREECE
Kostas Karayannis
18, Ermou Street
Athens 126
Tel: 230301,3,5
Cable: RAKAR Athens
Telex: 21 59 62 RKAR GR

IRELAND
Hewlett-Packard Ltd.
224 Bath Road
Slough, SLI 4 DS, Bucks
Tel: Slough 753-33341
Cable: HEWPIE Slough
Telex: 84413

ITALY
Hewlett-Packard Italiana S.p.A.
Via Amerigo Vespucci 2
1-20124 **Milan**
Tel: (2) 6251 (10 lines)
Cable: HEWPACKIT Milan
Telex: 32046

Hewlett-Packard Italiana S.p.A.
Palazzo Italia
Piazza Marconi 25
1-00144 **Rome** - Eur
Tel: 6-591 2544
Cable: HEWPACKIT Rome
Telex: 61514

NETHERLANDS
Hewlett-Packard Benelux, N.V.
Weerdestein 117
P.O. Box 7825
Amsterdam, Z 11
Tel: 020-42 77 77
Cable: PALOBEN Amsterdam
Telex: 13 216

NORWAY
Hewlett-Packard Norge A/S
Box 149
Nesveien 13
N-1344 **Haslum**
Tel: (02)-53 83 60
Cable: HEWPACK Oslo
Telex: 16621

PORTUGAL
Telectra
Empresa Tecnica de Equipamentos
Electricos, S.a.r.l.
Rua Rodrigo da Fonseca 103
P.O. Box 2531
Lisbon 1
Tel: 68 60 72
Cable: TELECTRA Lisbon
Telex: 1598

SPAIN
Ataio Ingenieros SA
Enrique Larreta 12
Madrid, 16
Tel: 215 35 43
Cable: TELEATAIO Madrid
Telex: 27249E

Ataio Ingenieros SA
Ganduxer 76
Barcelona 6
Tel: 211-44-66
Cable: TELEATAIO Barcelona

SWEDEN
Hewlett-Packard Sverige AB
Enighetsvägen 1-3
Fack
S-161 20 **Bromma** 20
Tel: (08) 98 12 50
Cable: MEASUREMENTS Stockholm
Telex: 10721

Hewlett-Packard Sverige AB
Hagakersgatan 9C
S-431 41 **Mölndal**
Tel: 031 - 27 68 00
Telex: 21 312 hpmindl

SWITZERLAND
Hewlett Packard Schweiz AG
Zurcherstrasse 20
CH-8952 **Schlieren** Zurich
Tel: (051) 98 18 21/24
Cable: HPAG CH
Telex: 53933

Hewlett Packard Schweiz A.G.
Rue du Bois-du-Lan 7
1217 Meyrin 2 Geneva
Tel: (022) 41 54 00
Cable: HEWPACKSA Geneva
Telex: 27333 HPSA CH

TURKEY
Telekom Engineering Bureau
P.O. Box 376
Karakoy
Istanbul
Tel: 49 40 40
Cable: TELEMATION Istanbul

UNITED KINGDOM
Hewlett-Packard Ltd.
224 Bath Road
Slough, SLI 4 DS, Bucks
Tel: Slough (0753) 33341
Cable: HEWPIE Slough
Telex: 84413

Hewlett-Packard Ltd.
The Graftons
Stamford New Road
Altrincham, Cheshire
Tel: 061 928-8626
Telex: 668068

YUGOSLAVIA
Belram S.A.
83 avenue des Mimosas
Brussels 1150, Belgium
Tel: 34 33 32, 34 26 19
Cable: BELRAMEL Brussels
Telex: 21790

SOCIALIST COUNTRIES PLEASE CONTACT:
Hewlett-Packard Ges.m.b.H
Innstrasse 23/2
Postfach
A1204 **Vienna,** Austria
Tel: (222) 3366 06/09
Cable: HEWPACK Vienna
Telex: 75923

ALL OTHER EUROPEAN COUNTRIES CONTACT:
Hewlett-Packard S.A.
Rue du Bois-du-Lan 7
1217 Meyrin 2 **Geneva**
Switzerland
Tel: (022) 41 54 00
Cable: HEWPACKSA Geneva
Telex: 2.24.86

AFRICA, ASIA, AUSTRALIA

ANGOLA
Telectra Empresa Técnia de Equipamentos Eléctricos SAR
Rua de Barbosa Rodrigues 42-1°
Box 6487
Luanda
Cable: TELECTRA Luanda

AUSTRALIA
Hewlett-Packard Australia Pty. Ltd.
22-26 Weir Street
Glen Iris, 3146
Victoria
Tel: 20.1371 (6 lines)
Cable: HEWPARD Melbourne
Telex: 31024

Hewlett-Packard Australia Pty. Ltd.
61 Alexander Street
Crows Nest 2065
New South Wales
Tel: 43.7866
Cable: HEWPARD Sydney
Telex: 21561

Hewlett-Packard Australia Pty. Ltd.
97 Churchill Road
Prospect 5082
South Australia
Tel: 65.2366
Cable: HEWPARD Adelaide

Hewlett Packard Australia Pty. Ltd.
2nd Floor, Suite 13
Casablanca Buildings
196 Adelaide Terrace
Perth, W.A. 6000
Tel: 21-3330
Cable: HEWPARD Perth

Hewlett-Packard Australia Pty. Ltd.
10 Woolley Street
P.O. Box 191
Dickson A.C.T. 2602
Tel: 49-8194
Cable: HEWPARD Canberra ACT

Hewlett-Packard Australia Pty. Ltd.
75 Simpsons Road
Bardon
Queensland, 4068
Tel: 36-5411

CEYLON
United Electricals Ltd.
P.O. Box 681
Yahala Building
Staples Street
Colombo 2
Tel: 5496
Cable: HOTPOINT Colombo

CYPRUS
Kypronics
19 Gregorios & Xenopoulos Road
P.O. Box 1152
Nicosia
Tel: 6282-75628
Cable: HE-I-NAMI

ETHIOPIA
African Salespower & Agency Private Ltd., Co.
P. O. Box 718
58/59 Cunningham St.
Addis Ababa
Tel: 12285
Cable: ASACO Addisababa

HONG KONG
Schmidt & Co. (Hong Kong) Ltd.
P.O. Box 297
1511, Prince's Building 15th Floor
10, Chater Road
Hong Kong
Tel: 240168, 232735
Cable: SCHMIDTCO Hong Kong

INDIA
Blue Star Ltd.
Kasturi Buildings
Jamshedji Tata Rd.
Bombay 20BR, India
Tel: 29 50 21
Telex: 2156
Cable: BLUEFROST

Blue Star Ltd.
Band Box House
Prabhadevi
Bombay 25DD, India
Tel: 45 73 01
Telex: 2156
Cable: BLUESTAR

Blue Star Ltd.
14/40 Civil Lines
Kanpur, India
Tel: 6 88 82
Cable: BLUESTAR

Blue Star, Ltd.
7 Hare Street
P.O. Box 506
Calcutta 1, India
Tel: 23-0131
Telex: 655
Cable: BLUESTAR

Blue Star Ltd.
Blue Star House,
34 Ring Road
Lajpat Nagar
New Delhi 24, India
Tel: 62 32 76
Telex: 463
Cable: BLUESTAR

Blue Star Ltd.
17-C Ulsoor Road
Bangalore-8

Blue Star, Ltd.
96 Park Lane
Secunderabad 3, India
Tel: 7 63 91
Cable: BLUEFROST

Blue Star, Ltd.
23/24 Second Line Beach
Madras 1, India
Tel: 2 39 55
Telex: 379
Cable: BLUESTAR

Blue Star, Ltd.
1B Kaiser Bungalow
Dindli Road
Jamshedpur, India
Tel: 38 04
Cable: BLUESTAR

INDONESIA
Bah Bolon Trading Coy. N.V.
Djalah Merdeka 29
Bandung
Tel: 4915; 51560
Cable: ILMU
Telex: 08-809

IRAN
Telecom, Ltd.
P. O. Box 1812
240 Kh. Saba Shomali
Teheran
Tel: 43850, 48111
Cable: BASCOM Teheran

ISRAEL
Electronics & Engineering Div. of Motorola Israel Ltd.
17 Aminadav Street
Tel-Aviv
Tel: 36941 (3 lines)
Cable: BASTEL Tel-Aviv
Telex: Bastel Tv 033-569

JAPAN
Yokogawa-Hewlett-Packard Ltd.
Ohashi Building
1-59-1 Yoyogi
Shibuya-ku, Tokyo
Tel: 03-370-2281/7
Telex: 232-2024YHP
Cable: YHPMARKET TOK 23-724

Yokogawa-Hewlett-Packard Ltd.
Nisei Ibaragi Bldg.
2-2-8 Kasuga
Ibaragi-Shi
Osaka
Tel: (0726) 23-1641
Telex: 385-5332 YHPOSAKA

Yokogawa-Hewlett-Packard Ltd.
Ito Building
No. 59, Kotori-cho
Nakamura-ku, **Nagoya City**
Tel: (052) 551-0215

Yokogawa-Hewlett-Packard Ltd.
Nitto Bldg.
2300 Shinohara-cho,
Kohoku-ku
Yokohama 222
Tel: (405) 432-1504/5

KENYA
Kenya Kinetics
P.O. Box 18311
Nairobi, Kenya
Tel: 57726
Cable: PROTON

KOREA
American Trading Co.,
Korea, Ltd.
Seoul P.O. Box 1103
7th & 8th floors, DaeKyung Bldg.
107 Sejong Ro
Chongro-Ku, Seoul
Tel: 75-5841 (4 lines)
Cable: AMTRACO Seoul

LEBANON
Constantin E. Macridis
Clemenceau Street
P.O. Box 7213
Beirut
Tel: 220846
Cable: ELECTRONUCLEAR Beirut

MALAYSIA
MECOMB Malaysia Ltd.
2 Lorong 13/6A
Section 13
Petaling Jaya, **Selangor**
Cable: MECOMB Kuala Lumpur

MOZAMBIQUE
A. N. Goncalves, LDA.
4.1 Apt. 14 Av. D. Luis
P.O. Box 107
Lourenco Marques
Cable: NEGON

NEW ZEALAND
Hewlett-Packard (N.Z.) Ltd.
94-96 Dixson St.
P.O. Box 9443
Wellington, N.Z.
Tel: 56-559
Cable: HEWPACK Wellington

Hewlett Packard (N.Z.) Ltd.
Box 51092
Pukuranga

PAKISTAN (EAST)
Mushko & Company, Ltd.
1, Jinnah Avenue
Dacca 2
Tel: 280058
Cable: NEWDEAL Dacca

PAKISTAN (WEST)
Mushko & Company, Ltd.
Oosman Chambers
Abdullah Haroon Road
Karachi 3
Tel: 511027, 512927
Cable: COOPERATOR Karachi

PHILIPPINES
Electromex Inc.
Makati Commercial Center
2129 Pasong Tamo
Makati, Rizal D 708
P.O. Box 1028
Manila
Tel: 89-85-01; 88-91-71
Cable: ELEMEX Manila

SINGAPORE
Mechanical and Combustion Engineering Company Ltd.
9, Jalan Kilang
Red Hill Industrial Estate
Singapore, 3
Tel: 642361-3; 632611
Cable: MECOMB Singapore

Hewlett-Packard Far East Area Office
P.O. Box 87
Alexandra Post Office
Singapore 3
Tel: 633022
Cable: HEWPACK SINGAPORE

SOUTH AFRICA
Hewlett Packard South Africa (Pty.), Ltd.
P.O. Box 31716
Braamfontein Transvaal
Milnerton
30 De Beer Street
Johannesburg
Tel: 725-2080, 725-2030
Telex: 0226 JH
Cable: HEWPACK Johannesburg

Hewlett Packard South Africa (Pty.), Ltd.
Breecastle House
Bree Street
Cape Town
Tel: 3-6019, 3-6545
Cable: HEWPACK Cape Town
Telex: 5-0006

Hewlett Packard South Africa (Pty.), Ltd.
30B Glenwood Centre
Corner Hunt & Moore Roads
P.O. Box 99

TAIWAN
Hewlett Packard Taiwan
39 Chung Shiao West Road
Sec. 1
Overseas Insurance
Corp. Bldg. 7th Floor
Taipei
Tel: 579-605, 579-610, 579-613
Telex: TP824 HEWPACK
Cable: HEWPACK Taipei

THAILAND
The International Engineering Co., Ltd.
P. O. Box 39
614 Sukhumvit Road
Bangkok
Tel: 910722 (7 lines)
Cable: GYSOM
TLX INTENCO BK-226 Bangkok

UGANDA
Uganda Tele-Electric Co., Ltd.
P.O. Box 4449
Kampala
Tel: 57279
Cable: COMCO Kampala

VIETNAM
Peninsular Trading Inc.
P.O. Box H-3
216 Hien-Vuong
Saigon
Tel: 20805, 93398
Cable: PENTRA, SAIGON 242

ZAMBIA
R. J. Tilbury (Zambia) Ltd.
P.O. Box 2792
Lusaka
Zambia, Central Africa
Tel: 73793
Cable: ARJAYTEE, Lusaka

MEDITERRANEAN AND MIDDLE EAST COUNTRIES NOT SHOWN PLEASE CONTACT:
Hewlett-Packard Correspondence Office
Piazza Marconi 25
I-00144 Rome-Eur, Italy
Tel: (6) 59 40 29
Cable: HEWPACKIT Rome
Telex: 61514

OTHER AREAS NOT LISTED, CONTACT:
Hewlett-Packard INTERCONTINENTAL
3200 Hillview Ave.
Palo Alto, California 94304
Tel: (415) 326-7000
(Feb. 71 493-1501)
TWX: 910-373-1267
Cable: HEWPACK Palo Alto
Telex: 034-8461

HEWLETT • PACKARD SALES AND SERVICE

UNITED STATES

ALABAMA
P.O. Box 4207
2003 Byrd Spring Road S.W.
Huntsville 35802
Tel: (205) 881-4591
TWX: 810-726-2204

ARIZONA
2336 E. Magnolia St.
Phoenix 85034
Tel: (602) 252-5061
TWX: 910-951-1330

5737 East Broadway
Tucson 85716
Tel: (602) 298-2313
TWX: 910-952-1162

CALIFORNIA
1430 East Orangethorpe Ave.
Fullerton 92631
Tel: (714) 870-1000

3939 Lankershim Boulevard
North Hollywood 91604
Tel: (213) 877-1282
TWX: 910-499-2170

1101 Embarcadero Road
Palo Alto 94303
Tel: (415) 327-6500
TWX: 910-373-1280

2220 Watt Ave.
Sacramento 95825
Tel: (916) 482-1463
TWX: 910-367-2092

9606 Aero Drive
San Diego 92123
Tel: (714) 279-3200
TWX: 910-335-2000

COLORADO
7965 East Prentice
Englewood 80110
Tel: (303) 771-3455
TWX: 910-935-0705

CONNECTICUT
508 Tolland Street
East Hartford 06108
Tel: (203) 289-9394
TWX: 710-425-3416

111 East Avenue
Norwalk 06851
Tel: (203) 853-1251
TWX: 710-468-3750

FLORIDA
P.O. Box 24210
2806 W. Oakland Park Blvd.
Ft. Lauderdale 33307
Tel: (305) 731-2020
TWX: 510-955-4099

P.O. Box 20007
Herndon Station 32814
621 Commonwealth Avenue
Orlando
Tel: (305) 841-3970
TWX: 810-850-0113

GEORGIA
P.O. Box 28234
450 Interstate North
Atlanta 30328
Tel: (404) 436-6181
TWX: 810-766-4890

ILLINOIS
5500 Howard Street
Skokie 60076
Tel: (312) 677-0400
TWX: 910-223-3613

INDIANA
3839 Meadows Drive
Indianapolis 46205
Tel: (317) 546-4891
TWX: 810-341-3263

LOUISIANA
P.O. Box 856
1942 Williams Boulevard
Kenner 70062
Tel: (504) 721-6201
TWX: 810-955-5524

MARYLAND
6707 Whitestone Road
Baltimore 21207
Tel: (301) 944-5400
TWX: 710-862-9157

P.O. Box 1648
2 Choke Cherry Road
Rockville 20850
Tel: (301) 948-6370
TWX: 710-828-9684

MASSACHUSETTS
32 Hartwell Ave.
Lexington 02173
Tel: (617) 861-8960
TWX: 710-326-6904

MICHIGAN
21840 West Nine Mile Road
Southfield 48075
Tel: (313) 353-9100
TWX: 810-224-4882

MINNESOTA
2459 University Avenue
St. Paul 55114
Tel: (612) 645-9461
TWX: 910-563-3734

MISSOURI
11131 Colorado Ave.
Kansas City 64137
Tel: (816) 763-8000
TWX: 910-771-2087

2812 South Brentwood Blvd.
St. Louis 63144
Tel: (314) 962-5000
TWX: 910-760-1670

NEW JERSEY
W. 120 Century Road
Paramus 07652
Tel: (201) 265-5000
TWX: 710-990-4951

1060 N. Kings Highway
Cherry Hill 08034
Tel: (609) 667-4000
TWX: 710-892-4945

NEW MEXICO
P.O. Box 8366
Station C
6501 Lomas Boulevard N.E.
Albuquerque 87108
Tel: (505) 265-3713
TWX: 910-989-1665

156 Wyatt Drive
Las Cruces 88001
Tel: (505) 526-2485
TWX: 910-983-0550

NEW YORK
1702 Central Avenue
Albany 12205
Tel: (518) 869-8462
TWX: 710-441-8270

1219 Campville Road
Endicott 13760
Tel: (607) 754-0050
TWX: 510-252-0890

82 Washington Street
Poughkeepsie 12601
Tel: (914) 454-7330
TWX: 510-248-0012

39 Saginaw Drive
Rochester 14623
Tel: (716) 473-9500
TWX: 510-253-5981

5858 East Molloy Road
Syracuse 13211
Tel: (315) 454-2486
TWX: 710-541-0482

1 Crossways Park West
Woodbury 11797
Tel: (516) 921-0300
TWX: 510-223-0811

NORTH CAROLINA
P.O. Box 5188
1923 North Main Street
High Point 27262
Tel: (919) 885-8101
TWX: 510-926-1516

OHIO
25575 Center Ridge Road
Cleveland 44145
Tel: (216) 835-0300
TWX: 810-427-9129

3460 South Dixie Drive
Dayton 45439
Tel: (513) 298-0351
TWX: 810-459-1925

1120 Morse Road
Columbus 43229
Tel: (614) 846-1300

OKLAHOMA
2919 United Founders Boulevard
Oklahoma City 73112
Tel: (405) 848-2801
TWX: 910-830-6862

OREGON
Westhills Mall, Suite 158
4475 S.W. Scholls Ferry Road
Portland 97225
Tel: (503) 292-9171
TWX: 910-464-6103

PENNSYLVANIA
2500 Moss Side Boulevard
Monroeville 15146
Tel: (412) 271-0724
TWX: 710-797-3650

1021 8th Avenue
King of Prussia Industrial Park
King of Prussia 19406
Tel: (215) 265-7000
TWX: 510-660-2670

RHODE ISLAND
873 Waterman Ave.
East Providence 02914
Tel: (401) 434-5535
TWX: 710-381-7573

TEXAS
P.O. Box 1270
201 E. Arapaho Rd.
Richardson 75080
Tel: (214) 231-6101
TWX: 910-867-4723

P.O. Box 22813
6300 Westpark Drive
Suite 100
Houston 77027
Tel: (713) 781-6000
TWX: 910-881-2645

231 Billy Mitchell Road
San Antonio 78226
Tel: (512) 434-4171
TWX: 910-871-1170

UTAH
2890 South Main Street
Salt Lake City 84115
Tel: (801) 487-0715
TWX: 910-925-5681

VERMONT
P.O. Box 2287
Kennedy Drive
South Burlington 05401
Tel: (802) 658-4455
TWX: 510-299-0025

VIRGINIA
P.O. Box 6514
2111 Spencer Road
Richmond 23230
Tel: (703) 285-3431
TWX: 710-956-0157

WASHINGTON
433-108th N.E.
Bellevue 98004
Tel: (206) 454-3971
TWX: 910-443-2303

***WEST VIRGINIA**
Charleston
Tel: (304) 768-1232

FOR U.S. AREAS NOT LISTED:
Contact the regional office nearest you: **Atlanta, Georgia** . . . North Hollywood, **California** . . . Paramus, **New Jersey** . . . **Skokie, Illinois.** Their complete addresses are listed above.

*Service Only

CANADA

ALBERTA
Hewlett-Packard (Canada) Ltd.
11745 Jasper Ave.
Edmonton
Tel: (403) 482-5561
TWX: 610-831-2431

BRITISH COLUMBIA
Hewlett-Packard (Canada) Ltd.
4519 Canada Way
North Burnaby 2
Tel: (604) 433-8213
TWX: 610-922-5059

MANITOBA
Hewlett-Packard (Canada) Ltd.
511 Bradford Ct.
Winnipeg
Tel: (204) 786-7581
TWX: 610-671-3531

NOVA SCOTIA
Hewlett-Packard (Canada) Ltd.
2745 Dutch Village Rd.
Suite 206
Halifax
Tel: (902) 455-0511
TWX: 610-271-4482

ONTARIO
Hewlett-Packard (Canada) Ltd.
880 Lady Ellen Place
Ottawa 3
Tel: (613) 722-4223
TWX: 610-562-1952

Hewlett-Packard (Canada) Ltd.
50 Galaxy Blvd.
Rexdale
Tel: (416) 677-9611
TWX: 610-492-4246

QUEBEC
Hewlett-Packard (Canada) Ltd.
275 Hymus Boulevard
Pointe Claire
Tel: (514) 697-4232
TWX: 610-422-3022
Telex: 01-20607

FOR CANADIAN AREAS NOT LISTED:
Contact Hewlett-Packard (Canada) Ltd. in Pointe Claire, at the complete address listed above.

CENTRAL AND SOUTH AMERICA

ARGENTINA
Hewlett-Packard Argentina S.A.C.e.I
Lavalle 1171 - 3°
Buenos Aires
Tel: 35-0436, 35-0627, 35-0431
Telex: 012-1009
Cable: HEWPACKARG

BRAZIL
Hewlett-Packard Do Brasil I.e.C Ltda.
Rua Frei Caneca 1119
Sao Paulo - 3, SP
Tel: 288-7111, 287-5858
Cable: HEWPACK Sao Paulo

Hewlett-Packard Do Brasil
Praca Dom Feliciano 78
Salas 806/808
Porto Alegre
Rio Grande do Sul (RS)-Brasil
Tel: 25-8470
Cable: HEWPACK Porto Alegre

Hewlett-Packard Do Brasil I.e.C. Ltda.
Rua da Matriz 29
Botafogo ZC-02
Rio de Janeiro, GB
Tel: 246-4417
Cable: HEWPACK Rio de Janeiro

Hewlett-Packard Do Brasil Indústria e Comercia Ltda.
Praca Dom Feliciano 78
Salas 806-8
Porto Alegre RGS

CHILE
Héctor Calcagni y Cia, Ltda.
Bustos, 1932-3er Piso
Casilla 13942
Santiago
Tel: 423 96
Cable: CALCAGNI Santiago

COLOMBIA
Instrumentacion
Henrik A. Langebaek & Kier Ltda.
Carrera 7 No. 48-59
Apartado Aereo 6287
Bogota, 1 D.E.
Tel: 45-78-06, 45-55-46
Cable: AARIS Bogota
Telex: 44400 INSTCO

COSTA RICA
Lic. Alfredo Gallegos Gurdián
Apartado 10159
San José
Tel: 21-86-13
Cable: GALGUR San José

ECUADOR
Laboratorios de Radio-Ingenieria
Calle Guayaquil 1246
Post Office Box 3199
Quito
Tel: 212-496; 219-185
Cable: HORVATH Quito

EL SALVADOR
Electrónica
Apartado Postal 1589
Blvd. Venezuela 1231
San Salvador
Tel: 217527; 214895
Cable: ELECTRONICA
San Salvador

MEXICO
Hewlett-Packard Mexicana, S.A. de C.V.
622 Adolfo Prieto
Col. del Valle
Mexico 12, D.F.
Tel: 543-4232; 523-1874
Telex: 0017-74507

NICARAGUA
Roberto Terán G.
Apartado Postal 689
Edificio Terán
Managua
Tel: 3451, 3452
Cable: ROTERAN Managua

PANAMA
Electrónico Balboa, S.A.
P.O. Box 4929
Ave. Manuel Espinosa No. 13-50
Bldg. Alina
Panama City
Tel: 230833
Telex: 3481003, Curundu, Canal Zone
Cable: ELECTRON Panama City

PERU
Compañia Electro Medica S.A.
Ave. Enrique Canaual 312
San Isidro
Casilla 1030
Lima
Tel: 22-3900
Cable: ELMED Lima

PUERTO RICO
San Juan Electronics, Inc.
P.O. Box 5167
Ponce de Leon 154
Pda. 3-PTA de Tierra
San Juan 00906
Tel: (809) 725-3342, 722-3342
Cable: SATRONICS San Juan
Telex: SATRON 3450 332

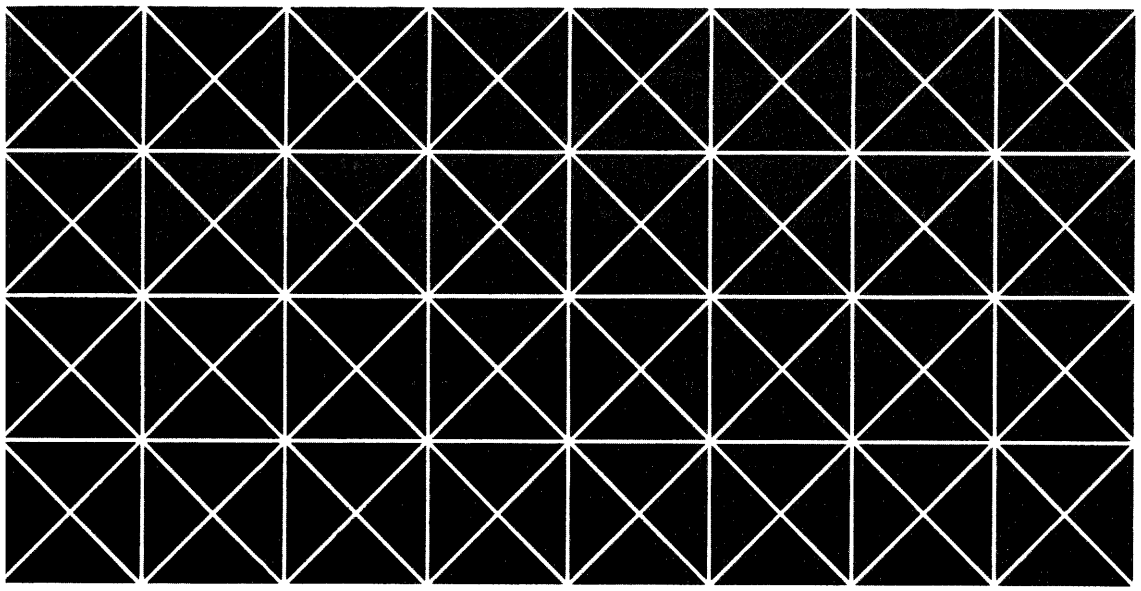
SURINAME
Surtel-Radio Holland N.V.
P.O. Box 155
Paramaribo
Tel: 72118
Cable: Treurniet Paramaribo

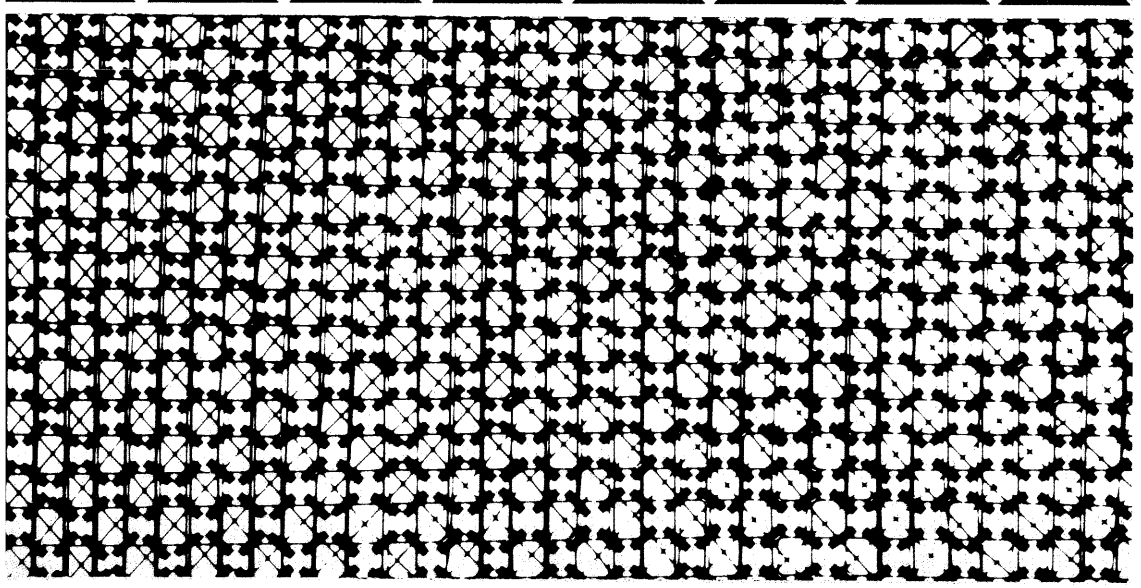
URUGUAY
Pablo Ferrando S.A.
Comercial e Industrial
Avenida Italia 2877
Casilla de Correo 370
Montevideo
Tel: 40-3102
Cable: RADIUM Montevideo

VENEZUELA
Hewlett-Packard De Venezuela C.A.
Apartado 50933
Caracas
Tel: 71.88.05, 71.88.69, 71.99.30
Cable: HEWPACK Caracas
Telex: 39521146

FOR AREAS NOT LISTED,

CONTACT:
Hewlett-Packard INTERCONTINENTAL
3200 Hillview Ave.
Palo Alto, California 94304
Tel: (415) 493-1501
TWX: 910-373-1267
Cable: HEWPACK Palo Alto
Telex: 034-8461





HEWLETT hp PACKARD

02116-91755

PRINTED IN U.S.A.