



21MX COMPUTER SERIES

reference manual



HEWLETT-PACKARD COMPANY
11000 WOLFE ROAD, CUPERTINO, CALIFORNIA, 95014

MANUAL PART NO. 02108-90002
MICROFICHE PART NO. 02108-90003

Printed: MAY 1974
Printed in U.S.A.

CONTENTS

Section I	Page
SYSTEM FEATURES	
Microprogramming Capabilities	1-2
System Speed	1-2
Memory Space	1-3
Special Functions and Security	1-3
Additional Hardware Facilities	1-3
Hewlett-Packard Software	1-3
Input/Output	1-4
Specifications	1-4
System Expansion and Enhancement	1-4

Section II	Page
OPERATING FEATURES	
Hardware Registers	2-1
A-Register	2-1
B-Register	2-1
M-Register	2-1
T-Register	2-1
P-Register	2-1
S-Register	2-1
Extend Register	2-1
Overflow Register	2-1
Display Register	2-2
X- and Y-Registers	2-2
Control and Indicators	2-2
Operator Panel	2-2
Rear Panel	2-2
Internal Switches	2-2
Basic Operating Examples	2-3
Cold Start Procedure	2-3
Manual Loading	2-3
Running Programs	2-3

Section III	Page
PROGRAMMING INFORMATION	
Data Formats	3-1
Memory Addressing	3-1
Paging	3-1
Direct and Indirect Addressing	3-3
Reserved Memory Locations	3-3
Nonexistent Memory	3-4
Base Set Instruction Formats	3-4
Memory Reference Instructions	3-4
Register Reference Instructions	3-4
Input/Output Instructions	3-4
Extended Arithmetic Memory	
Reference Instructions	3-5
Extended Arithmetic Register	
Reference Instructions	3-5
Base Set Instruction Coding	3-5
Memory Reference Instructions	3-5
Register Reference Instructions	3-7
Shift/Rotate Group	3-7
Alter/Skip Group	3-10

Input/Output Instructions	3-11
Extended Arithmetic Memory	
Reference Instructions	3-13
Extended Arithmetic Register	
Reference Instructions	3-14
Extended Instruction Group Coding	3-16
Index Register Instructions	3-16
Jump Instructions	3-19
Byte Manipulation Instructions	3-20
Bit Manipulation Instructions	3-21
Word Manipulation Instructions	3-22
Floating Point Instruction Coding	3-23
Instruction Execution Times	3-24
Interrupt System	3-24
Power Fail Interrupt	3-26
Parity Error Interrupt	3-26
Memory Protect Interrupt	3-28
Dual-Channel Port Controller Interrupt	3-30
Input/Output Interrupt	3-30
Central Interrupt Register	3-30
Interrupt System Control	3-30

Section IV	Page
INPUT/OUTPUT SYSTEM	
Input/Output Addressing	4-1
Input/Output Priority	4-2
Interface Elements	4-3
Control Bit	4-3
Flag Bit	4-3
Buffer	4-4
Input/Output Data Transfer	4-4
Input Data Transfer	
(Interrupt Method)	4-4
Output Data Transfer	
(Interrupt Method)	4-5
Noninterrupt Data Transfer	4-5
Input	4-5
Output	4-6
Dual-Channel Port Controller	4-6
DCPC Operation	4-6
DCPC Initialization	4-7

Appendix	Page
Computer Physical Layout	A-2
Character Codes	A-3
Octal Arithmetic	A-4
Octal/Decimal Conversions	A-5
Mathematical Equivalents	A-6
Octal Combining Tables	A-8
Instruction Codes in Octal	A-9
Base Set Instruction Codes	
in Binary	A-10
Extended Instruction Group Codes	
in Binary	A-11
Interrupt and I/O Control Summary	A-12
Extend and Overflow Examples	A-14

HP Computer Museum

www.hpmuseum.net

For research and education purposes only.

ILLUSTRATIONS

Title	Page	Title	Page
HP 2105A and HP 2108A Microprogrammable Processors	1-1	Input/Output System	4-1
Operator Panel Controls and Indicators	2-2	I/O Address Assignments	4-2
Data Formats and Octal Notation	3-2	Priority Linkage	4-2
Base Set Instruction Formats	3-4	Interrupt Sequences	4-3
Shift and Rotate Functions	3-7	Input Data Transfer (Interrupt Method)	4-4
Examples of Double-Word Shifts and Rotates	3-15	Output Data Transfer (Interrupt Method)	4-5
		DCPC Input Data Transfer	4-7
		DCPC Control Word Formats	4-7

TABLES

Title	Page	Title	Page
Specifications	1-4	HP 2105A Interrupt Assignments	3-24
Options and Accessories	1-8	HP 2108A Interrupt Assignments	3-24
Front Panel Control and Indicator Functions	2-4	Instruction Execution Times	3-25
Memory Paging	3-3	Sample Power Fail Subroutine	3-27
Reserved Memory Locations	3-3	Sample Memory Protect/Parity Error Subroutine	3-29
Shift/Rotate Group Combining Guide	3-7	Noninterrupt Transfer Routines	4-6
Alter/Skip Group Combining Guide	3-10	DCPC Initialization Program	4-8

ALPHABETICAL INDEX OF INSTRUCTIONS

Instruction	Page
ADA Add to A	3-5
ADB Add to B	3-5
ADX Add Memory to X	3-16
ADY Add Memory to Y	3-16
ALF Rotate A Left Four	3-8
ALR A Left Shift, Clear Sign	3-8
ALS A Left Shift	3-8
AND "And" to A	3-5
ARS A Right Shift	3-8
ASL Arithmetic Shift Left (32)	3-14
ASR Arithmetic Shift Right (32)	3-14
BLF Rotate B Left Four	3-8
BLR B Left Shift, Clear Sign	3-8
BLS B Left Shift	3-8
BRS B Right Shift	3-8
CAX Copy A to X	3-17
CAY Copy A to Y	3-17
OBS Clear Bits	3-21
CBT Compare Bytes	3-20
CBX Copy B to X	3-17
CBY Copy B to Y	3-17
CCA Clear and Complement A	3-10
CCB Clear and Complement B	3-10
CCE Clear and Complement E	3-10
CLA Clear A	3-10
CLB Clear B	3-10
CLC Clear Control	3-12
CLE Clear E	3-9, 3-10
CLF Clear Flag	3-12
CLO Clear Overflow	3-12
CMA Complement A	3-10
CMB Complement B	3-10
CME Complement E	3-11
CMW Compare Words	3-22
CPA Compare to A	3-6
CPB Compare to B	3-6
CXA Copy X to A	3-17
CXB Copy X to B	3-17
CYA Copy Y to A	3-17
CYB Copy Y to B	3-17
DIV Divide	3-14
DLD Double Load	3-14
DST Double Store	3-14
DSX Decrement X and Skip if Zero	3-17
DSY Decrement Y and Skip if Zero	3-17
ELA Rotate E Left with A	3-9
ELB Rotate E Left with B	3-9
ERA Rotate E Right with A	3-9
ERB Rotate E Right with B	3-9
FAD Floating Point Add	3-23
FDV Floating Point Divide	3-23
FIX Floating Point to Integer	3-23
FLT Integer to Floating Point	3-24
FMP Floating Point Multiply	3-24
FSB Floating Point Subtract	3-24
HLT Halt	3-12
INA Increment A	3-11
INB Increment B	3-11
IOR "Inclusive Or" to A	3-6
ISX Increment X and Skip if Zero	3-17
ISY Increment Y and Skip if Zero	3-17
ISZ Increment and Skip if Zero	3-6

Instruction	Page
JLY Jump and Load Y	3-19
JMP Jump	3-6
JPY Jump Indexed by Y	3-20
JSB Jump to Subroutine	3-6
LAX Load A Indexed by X	3-18
LAY Load A Indexed by Y	3-18
LBT Load Byte	3-20
LBX Load B Indexed by X	3-18
LBY Load B Indexed by Y	3-18
LDA Load A	3-6
LDB Load B	3-6
LDX Load X from Memory	3-18
LDY Load Y from Memory	3-18
LIA Load Input to A	3-12
LIB Load Input to B	3-12
LSL Logical Shift Left (32)	3-16
LSR Logical Shift Right (32)	3-16
MBT Move Bytes	3-21
MIA Merge Into A	3-12
MIB Merge Into B	3-12
MPY Multiply	3-14
MVW Move Words	3-23
NOP No Operation	3-9
OTA Output A	3-12
OTB Output B	3-12
RAL Rotate A Left	3-9
RAR Rotate A Right	3-9
RBL Rotate B Left	3-9
RBR Rotate B Right	3-9
RRL Rotate Left (32)	3-16
RRR Rotate Right (32)	3-16
RSS Reverse Skip Sense	3-11
SAX Store A Indexed by X	3-18
SAY Store A Indexed by Y	3-19
SBS Set Bits	3-22
SBT Store Byte	3-21
SBX Store B Indexed by X	3-19
SBY Store B Indexed by Y	3-19
SEZ Skip if E is Zero	3-11
SFB Scan For Byte	3-21
SFC Skip if Flag Clear	3-12
SFS Skip if Flag Set	3-12
SLA Skip if LSB of A is Zero	3-10, 3-11
SLB Skip if LSB of B is Zero	3-10, 3-11
SOC Skip if Overflow Clear	3-12
SOS Skip if Overflow Set	3-12
SSA Skip if Sign of A is Zero	3-11
SSB Skip if Sign of B is Zero	3-11
STA Store A	3-6
STB Store B	3-7
STC Set Control	3-12
STF Set Flag	3-12
STO Set Overflow	3-12
STX Store X to Memory	3-19
STY Store Y to Memory	3-19
SZA Skip if A is Zero	3-11
SZB Skip if B is Zero	3-11
TBS Test Bits	3-22
XAX Exchange A and X	3-19
XAY Exchange A and Y	3-19
XBX Exchange B and X	3-19
XBY Exchange B and Y	3-19
XOR "Exclusive Or" to A	3-7

SYSTEM FEATURES

SECTION

I

The HP 2105A and HP 2108A Microprogrammable Processors (figure 1-1) accommodate a variety of memory configurations to form the new and powerful HP 21MX Computer Series. Salient features of these computers, which utilize the latest developments in semiconductor technology, are as follows.

PROCESSOR

- Powerful user-microprogrammable processor with 178 microinstructions and 4K of control store space.
- 128 standard instructions including 80 instructions which emulate the HP 2100 Series Computer; 42 new instructions for indexing, byte and bit manipulation, byte and word moves, and byte string scanning; and 6 single-precision floating point instructions.
- 4 general-purpose registers, two of which may be used as index registers.
- Fully microprogrammed processor, including all arithmetic functions, input/output, and operator panel control.

- Initial binary loader is ROM resident and callable by a pushbutton switch on the operator panel. A paper tape loader ROM is standard; provision is made for up to three additional loader ROM's, which are available as options or may be user-generated.
- Operator panel is standard.
- Writable control store is optional.
- Programmable ROM (pROM) writer available as a supporting product.

POWER SUPPLY

- Power module supplies power for the processor, 32K words of memory, all memory system options, all standard and optional microcode packages, and a wide range of I/O controller configurations.
- Power module operates over a wide range of line voltage and line frequency variations; operates through a line loss of 2-1/2 hertz.

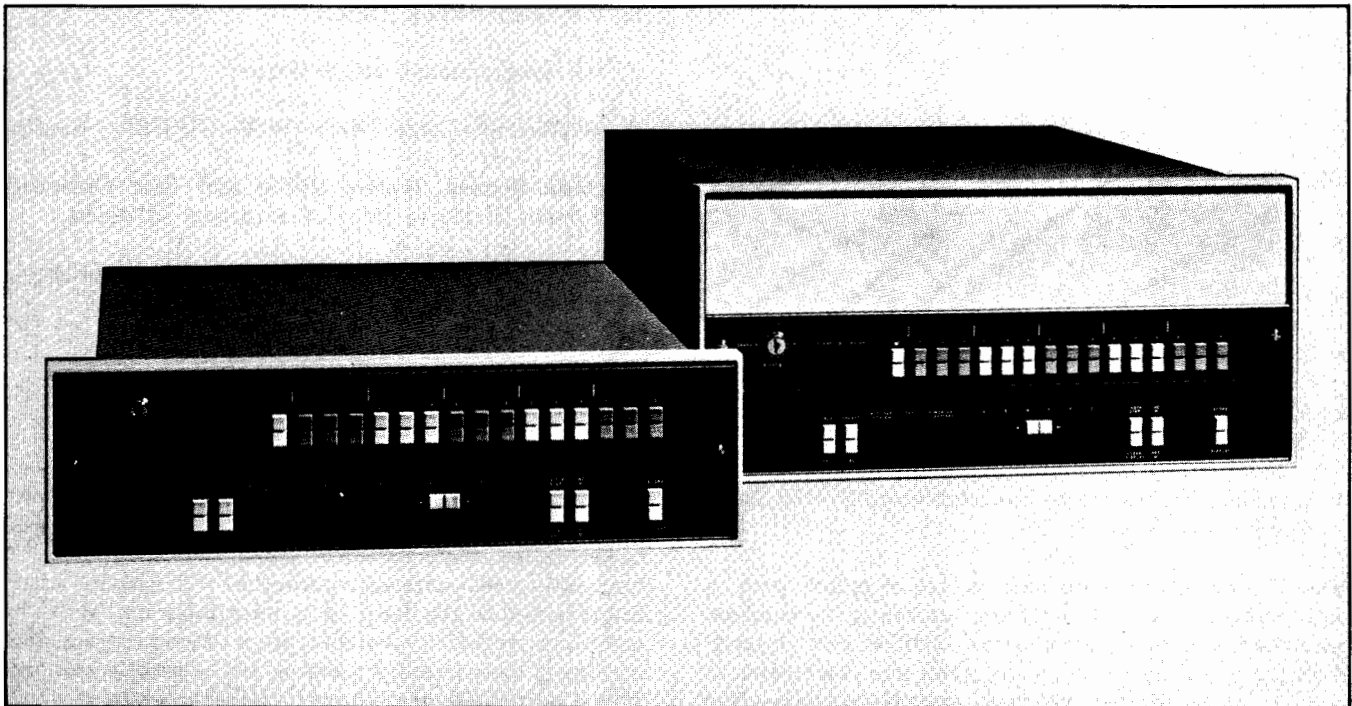


Figure 1-1. HP 2105A and HP 2108A Microprogrammable Processors

MEMORY

- N-channel MOS semiconductor memory with 650-nS cycle time; expandable as follows:

HP 2101A Memory — expandable to 32K in mainframe (both HP 2105A and HP 2108A).

HP 2102A Memory — expandable to 16K in HP 2105A and to 32K in HP 2108A.

- Memory parity generation and checking is standard.
- 2 hours of memory sustaining power with power fail recovery system option; this option provides an automatic restart capability.
- Optional dual-channel port controller with bidirectional input/output transfer; provides for direct memory access, program assignable to any I/O channel.
- Memory, I/O, and infinite indirect addressing protection provided by the memory protect option.
- Memory integrity is maintained through a line loss of 10 hertz.

INPUT/OUTPUT

- Fully microprogrammed I/O instructions.
- HP 2105A has four I/O channels (standard).
- HP 2108A has nine I/O channels (standard).
- I/O channels can be optionally increased with one or two I/O extenders (17 additional channels each).
- Compatibility maintained with existing HP 2100 Series Computer I/O interfaces.

These compact data processors are supplied with a comprehensive set of software, including assemblers, compilers, and operating systems. A full line of Hewlett-Packard peripherals and I/O interface kits is available to provide a flexible and efficient systems package. The programming languages are those commonly encountered in data processing and control.

Each consists of 1,024 directly addressable locations configured into four modules of 256 locations each. Each control store location accommodates one microinstruction, which in turn consists of a 24-bit word encompassing six micro-orders. The address space of either processor is 4,096 words.

Microprograms in control store for executing the various machine functions are as follows:

- Base instruction set (two modules)
- Extended instruction group (one module)
- Floating point instructions (one module)

Unused modules of control store are available for user-supplied microprograms. Microprogramming capabilities from a hardware standpoint are provided in *Writable Control Store for User Microprogramming Operating and Reference Manual*, part no. 12978-90001.

This manual covers in detail the following:

a. HP Microassembler

b. HP Micro Debug Editor

c. HP Programmable ROM Writer

d. HP Writable Control Store (WCS) I/O Utility Routine

e. Basic Control System (BCS) and Disc Operating System (DOS) versions of a through d above.

Some of the more important benefits of microprogramming are presented in following paragraphs.

1-2. SYSTEM SPEED

Microprogramming can increase the system speed in many ways. Since microinstructions are executed from 5 to 10 times faster than machine language instructions, a frequently used software subroutine will execute much faster when in the form of a microprogram. With 14 additional registers available to a microprogram, the number of main memory accesses can be greatly reduced. This is particularly significant in real-time systems which are compute-bound (i.e., systems in which the I/O is performed faster than the computation).

1-3. MEMORY SPACE

By converting software routines into microprograms, main memory space is freed for other purposes. The routines remain instantly callable, as opposed to routines which are relegated to disc or drum storage.

1-4. SPECIAL FUNCTIONS AND SECURITY

The computer instruction set can be expanded to perform functions that are oriented to specific applications. Thus, the general-purpose computer can become a special-purpose machine uniquely adapted to a particular environment. Because of the relative inaccessibility of microprograms as compared to conventional software in main memory, proprietary packages which are coded as microprograms inherently have a high degree of security.

1-5. ADDITIONAL HARDWARE FACILITIES

Through microprogramming, 14 additional registers are accessible. Software instructions may be invented to reference and use these registers. In addition, due to the three-operand format of the computer microinstruction, instructions can be created which perform some function with the contents of two or more registers and store the result in yet another register. A flag bit, also not otherwise accessible, may also be used.

The Writable Control Store (WCS) option provides a read-write control store module which can be used for the development and execution of user-supplied microprograms. Microprograms in WCS are executed at the same speed as those in the read-only control store. Two WCS cards may be used in an HP 2108A and one WCS card may be used in an HP 2105A. The module containing the basic instruction set must always be in the form of Read-Only-Memory (ROM) integrated circuit chips. Each WCS module consists of a single card which plugs into a computer I/O slot, thus eliminating the need for extensive cabling or an additional power supply. A WCS card contains 256 24-bit locations of Random-Access-Memory (RAM), including all necessary address and read/write circuits. WCS can be written into or read under computer control using standard input/output instructions. An I/O utility routine makes it possible for FORTRAN and ALGOL programs to write into or read from a WCS module using a conventional subroutine call. A WCS module is read at full speed by way of a flat cable connecting it to the control section of the computer. Software supplied with the WCS card includes a micro-assembler, a micro debug editor, a WCS I/O driver, a WCS I/O utility routine, and a WCS diagnostic.

The Programmable ROM Writer option makes it possible for the user to permanently transfer microprograms to programmable Read-Only-Memory chips which can then be physically added to the control section of the computer. The Programmable ROM Writer consists of a single card which plugs into a computer I/O slot, thus eliminating the

need for extensive cabling or an additional power supply. A small box is connected to the Programmable ROM Writer card by way of a cable; the programmable ROM chip to be burned is mounted on the box by the computer operator. A stand-alone computer program, supplied with the Programmable ROM Writer, burns and verifies the chip using punched tape input.

1-6. HEWLETT-PACKARD SOFTWARE

Software for the HP 21MX Computer Series includes four high-level programming languages: HP FORTRAN, HP FORTRAN IV, HP ALGOL, and HP BASIC, plus an efficient, extended assembler which is callable by FORTRAN and ALGOL. Utility software includes a debugging routine, a symbolic editor, and a library of commonly used computational procedures such as Boolean, trigonometric, and plotting functions, real/integer conversions, natural log, square root, etc.

Hewlett-Packard provides several systems built around BASIC interpreters. The single-terminal BASIC system allows the user to prepare and run BASIC language programs conversationally through a teleprinter. Programs can also be entered through a tape reader and punched out on tape punches. A similar system, Educational BASIC, allows BASIC programs to be translated from marked cards. The time-shared BASIC systems provide an extended version of the BASIC language to 16 or 32 users simultaneously. The extensions to BASIC allow the user to store and access large amounts of data in an external mass memory, to manipulate strings of characters, and to store and retrieve programs in mass memory.

Several operating systems are available, covering a wide range of applications. The Basic Control System, which simplifies the control of input/output operations, also provides relocatable loading and linking of user programs. The time-shared systems, using conversational BASIC language, permit up to 32 terminals to be connected to the system, either directly or by telephone lines via Dataphones. The Hewlett-Packard Real-Time Executive (RTE) system permits several programs to run in real-time concurrently with general-purpose background programs. This allows multiple data-processing capabilities where separate computers are not economically feasible. The user can write programs in HP Assembly, FORTRAN, or ALGOL languages. A Magnetic Tape System and a Disc Operating System are also available. These systems greatly increase the speed and simplicity of assembling, compiling, loading, and executing user programs.

The Hewlett-Packard User Library includes over 800 tested and documented BASIC programs contributed for users. The library is segmented into the following five categories:

- Data Handling and Programming Utilities
- Scientific and Numerical Analyses

- Operations Research and Business Applications
- Education
- Demonstration Routines

occur on a cycle-stealing basis, not subject to the I/O priority structure. The total bandwidth through both DCPC channels is 616,666 words per second. The CPU computation continues at a reduced rate even when full channel transfer bandwidth is being utilized.

1-7. INPUT/OUTPUT

Interfacing of peripheral devices is accomplished by plug-in interface printed-circuit assemblies (PCA's). The HP 2105A mainframe can accommodate up to four interface PCA's, expandable to 36 with two optional HP 12979A I/O Extenders. The HP 2108A mainframe can accommodate up to nine interface PCA's, expandable to 41 with two optional HP 12979A I/O Extenders. Interface PCA's are available for a wide variety of peripheral devices, and virtually all interfaces developed for use with HP 2100 Series Computers may be used with the HP 21MX Computer Series.

All I/O channels are buffered and bidirectional, and are serviced through a multilevel vectored priority interrupt structure. The two dual-channel port controller (DCPC) channels are program-assignable to any two of the I/O channels in the mainframe, expandable to all I/O channels if a DCPC is installed in the I/O extender. DCPC transfers

1-8. SPECIFICATIONS

Table 1-1 lists the specifications of the HP 2105A and HP 2108A Microprogrammable Processors and the HP 2101A and HP 2102A Memory Systems. Both processors have been approved by the Underwriters' Laboratories (UL) and the Canadian Standards Association (CSA).

1-9. SYSTEM EXPANSION AND ENHANCEMENT

Table 1-2 lists the options and accessories available to expand or enhance the computer system. On an original order, specify the desired system configuration by option number. For a field update of the existing system, specify the system addition by accessory number.

Table 1-1. Specifications

PROCESSOR	
CONTROL STORE	
Type:	Bipolar LSI ROM semiconductor.
Size:	Up to sixteen 256-word modules.
CONTROL PROCESSOR	
Address Space:	4,096 words.
Word Size:	24 bits.
Word Formats:	Four.
Word Fields:	Five.
ROM Cycle:	325 nanoseconds.
REGISTERS	
Accumulators:	Two (A and B), 16 bits each. Implicitly addressable; also explicitly addressable as memory.
Index:	Two (X and Y), 16 bits each.
Memory Control:	Two (T and P), 16 bits each; one (M) 15 bits.
Supplementary:	Two (overflow and extend), one bit each.
Manual Data:	One (display), 16 bits.
Scratch Pads:	Twelve, 16 bits each; accessible to microprogrammer.
MEMORY PARITY CHECK	
HP 2105A Processor:	Monitors all words read from memory. Switch selectable to either halt or ignore parity when detected. A parity indication is displayed on operator panel.
HP 2108A Processor:	Same as for HP 2105A. With memory protect option, interrupt on parity error occurs.

Table 1-1. Specifications (Continued)

PROCESSOR (Continued)**POWER FAIL INTERRUPT**

Priority:

Highest priority interrupt.

Power Failure:

Detects power failure and generates an interrupt to trap cell for user-written power-failure routine, terminates activities, and halts processor. A minimum of 500 μ S is available for the routine. Automatic restart is provided as a memory system option.

PROTECTION

Loaders:

All loaders reside in special ROM's separate from control ROM and are loaded into last 64 words of main memory by activating operator panel switches. Paper tape loader is standard; three additional switch-selectable loader spaces are provided to accommodate other modes of operation as a user option. User-generated loaders may be written in Assembly Language.

Volatility:

Mains ac standby mode and sustaining power for line loss of 2.5 Hz before entering power fail routine. Power fail recovery is a memory system option.

INPUT/OUTPUT

Priority Interrupt:

Multilevel vectored priority interrupt determined by interface channel assignment.

I/O Channels:

HP 2105A: four internal I/O channels; expandable to 36 channels with two I/O extenders.

HP 2108A: nine internal I/O channels; expandable to 41 channels with two I/O extenders.

Current Available to I/O:

SUPPLY	HP 2105A	HP 2108A
+5V	6.0A	13.0A
-2V	2.0A	4.0A
+12V	1.0A	1.5A
-12V	1.0A	1.5A

Note: Current availability to I/O assumes 32K memory, dual-channel port controller, and maximum available control store installed in main-frame.

PHYSICAL CHARACTERISTICS

Width:

16-3/4 inches (42.55 cm) behind rack mount; 19 inches (48.26 cm) operator panel width on sides.

Depth:

23-1/2 inches (59.69 cm); 23 inches (58.42 cm) behind operator panel.

Height:

HP 2105A: 5-1/4 inches (13.31 cm) in rack mount.

HP 2108A: 8-3/4 inches (22.23 cm) in rack mount.

Weight:

HP 2105A: 39 pounds (17.69 kg).

HP 2108A: 45 pounds (20.41 kg).

Table 1-1. Specifications (Continued)

PROCESSOR (Continued)**ELECTRICAL CHARACTERISTICS**

Input Line Voltage:	110V or 220V ac ($\pm 20\%$), single phase.
Line Frequency:	47 to 66 Hz.
Power:	HP 2105A: 400W maximum. HP 2108A: 525W maximum.
Line Overvoltage Protect:	Input crowbar in series with line fuse.
Output Protect:	All voltages protected against overvoltage and overcurrent.
Output Voltage Regulation:	$\pm 5\%$.
Thermal Sensing:	Monitors internal temperature and automatically shuts down if temperature exceeds specified level.

ENVIRONMENTAL LIMITATIONS

Ambient Temperature:	Operating: 32° to 131°F (0° to 55°C). Nonoperating: -40° to 167°F (-40° to 75°C).
Altitude:	Operating: 15,000 feet (4,573 meters). Nonoperating: 25,000 feet (7,622 meters).
Relative Humidity:	50 to 95% at 77° to 104°F (25° to 40°C).
Shock:	Tested for 30g shock for 11 milliseconds over a 1/2 sine wave shape.
Vibration:	Can withstand vibration of 1g at 44 cycles per second.

VENTILATION

Air Flow:	Intake on left-hand side; exhaust on right-hand side.
Heat Dissipation:	HP 2105A: 1365 BTU's (344 kilocalories)/hour, max. HP 2108A: 1795 BTU's (452 kilocalories)/hour, max.

MEMORY SYSTEMS**HP 2101A MEMORY**

Density:	High density; 16K words per module.
Configuration:	Available in 8K or 16K configuration; only one 8K module allowed per system.

HP 2102A MEMORY

Density:	Medium density; 8K words per module.
Configuration:	Available in 4K or 8K configuration; only one 4K module allowed per system.

MEMORY ORGANIZATION

Type:	4K chip N-channel MOS/RAM semiconductor.
Word Size:	16 bits plus parity bit.
Configuration:	Controller and multiple plug-in memory modules.
Page Size:	1,024 words.
Direct Addressing:	2 pages.
Indirect Addressing:	32K.
System Cycle Time:	650 nanoseconds.
Volatility Protection:	Mains ac standby mode and sustaining power for line loss of 10 Hz is standard. Power fail recovery system is optional.

Table 1-1. Specifications (Continued)

MEMORY SYSTEMS (Continued)**MEMORY PROTECT (HP 2108A only)**

Installation:	Plugs into slot 111 of memory PCA cage.
Priority:	Second highest priority interrupt (shared with memory parity).
Operation:	Initiated under programmed control; protects any amount of memory, I/O, or privileged instruction when implemented in the HP 2108A Processor.
Fence Register:	Set under program control; memory below fence is protected.
Interrupt:	Interrupts to trap cell for subroutine when user program (1) attempts to alter a protected location, (2) attempts to jump into the protected area, (3) or attempts to execute an I/O instruction.
Violation Register:	Contains memory address of violating instruction.
Parity Error Interrupt:	Provides interrupt signal when parity error is detected; saves address of error in violation register.
Infinite Indirect Protect:	Interrupts are enabled after three levels of indirect addressing.

DUAL-CHANNEL PORT CONTROLLER

Installation:	Plugs into slot 110 of memory PCA cage.
Number of Channels:	Two.
Number of Memory Ports:	One.
Registers/Channel:	Two (word count and address).
Word Size:	16 bits.
Maximum Block Size:	32,768 words.
I/O Assignable:	Assignable to any two I/O channels; all logic necessary to facilitate bidirectional direct memory to and from I/O is contained on this controller.
Transfer Rate:	616,666 words per second maximum.
Priority:	Highest: DCPC Channel 1. Middle: DCPC Channel 2. Lowest: Processor.

POWER FAIL RECOVERY SYSTEM

Power Restart:	Detects resumption of power and generates an interrupt to trap cell for user-written restart program which has been protected in memory by the sustaining battery.
Power Control and Charge Unit:	Monitors battery charge status and provides trickle charge.
Sustaining Battery:	Type: 12V nickel cadmium. Charging rate: 350 milliamperes. Capacity: 4 ampere-hours; will sustain 32K main memory for 2 hours.

DESCRIPTION	OPTION NO.	ACCESSORY NO.
2105A OR 2108A PROCESSOR		
Scientific Instruction Set	-003	12977A
Writable Control Store	-005	12978A
Disc Loader Rom	-014	12992A
230V, 50-Hz Operation	-015	--
User Control Store Board	--	12945A
Programmable ROM Writer	--	12909B
 2101A MOS MEMORY SYSTEM		
Dual-Channel Port Controller	-001	12897A
Memory Protect*	-003	12892A
8K Memory Module	-008	12999A
16K Memory Module	-016	12997A
Memory System Cable	--	12993A
Power Fail Recovery System	--	12944A
 2102A MOS MEMORY SYSTEM		
Dual-Channel Port Controller	-001	12897A
Memory Protect*	-003	12892A
4K Memory Module	-004	12994A
8K Memory Module	-008	12998A
Memory System Cable	--	12993A
Power Fail Recovery System	--	12944A
 12979A I/O EXTENDER		
Dual-Channel Port Controller	-001	12898A
2nd I/O Extender	-010	--
230V, 50-Hz Operation	-015	--
*For HP 2108A Processor only.		

OPERATING FEATURES

SECTION

II

This section describes the hardware registers accessible to the programmer and the functions of the various operating controls and indicators. Also included are basic operating examples such as a cold start procedure to load a program via a punched-tape reader, manually loading a short program via the operator panel, and running a program after it has been loaded into memory. These examples are included only to illustrate the simplicity of operation. For detailed operating procedures, refer to the *HP 21MX Computer Series Operator's Manual*, part no. 02108-90004.

2-1. HARDWARE REGISTERS

The computer has eight 16-bit working registers, six of which can be selected for display and modification by operator panel controls; two 1-bit registers; and one 16-bit display register. Two of the eight 16-bit working registers are made accessible to software through the extended instruction group. The function of these registers are described in following paragraphs.

2-2. A-REGISTER

The A-register is a 16-bit accumulator that holds the results of arithmetic and logical operations performed by programmed instructions. This register can be addressed directly by any memory reference instruction as location 000000 (octal), thus permitting interrelated operations with the B-register (e.g., "add B to A," "compare B with A," etc.) using a single-word instruction.

2-3. B-REGISTER

The B-register is a second 16-bit accumulator, which can hold the results of arithmetic and logic operations completely independent of the A-register. The B-register can be addressed directly by any memory reference instruction as location 000001 (octal) for interrelated operations with the A-register.

2-4. M-REGISTER

The M-register holds the address of the memory cell currently being read from or written into by the CPU.

2-5. T-REGISTER

All data transferred into or out of memory is routed through the T-register. When displayed, the T-register indicates the contents of the memory location currently pointed to by the M-register. The A- or B-register contents are displayed if the M-register contents are 000000 or 000001, respectively.

2-6. P-REGISTER

The P-register holds the address of the next instruction to be fetched from memory. Since this is a "lookahead" register, the P-register contents will frequently differ from the M-register contents.

2-7. S-REGISTER

The S-register is a 16-bit utility register. In the halt or run mode, the S-register can be loaded via the display register. In the run mode, the S-register can be addressed as an input/output device (select code 01) and can input and output data to and from the A- and B-registers.

2-8. EXTEND REGISTER

The one-bit extend register is used to link the A- and B-registers by rotate instructions or to indicate a carry from the most-significant bit (bit 15) of the A- or B-register by an add instruction (ADA, ADB) or an increment instruction (INA, INB, but not ISZ). This is of significance primarily for multiple-precision arithmetic operations. If already set (logic 1), the extend bit cannot be cleared by a carry. However, the extend bit can be selectively set, cleared, complemented, or tested by programmed instructions. When the operator panel EXTEND indicator is lighted, the extend bit is set.

2-9. OVERFLOW REGISTER

The one-bit overflow register is used to indicate that an add instruction (ADA, ADB), divide instruction (DIV), or an increment instruction (INA, INB, but not ISZ) referencing the A- or B-register has caused (or will cause) the accumulators to exceed the maximum positive or negative number that can be contained in these registers. The overflow bit can be selectively set, cleared, or tested

Left (ASL).

2-10. DISPLAY REGISTER

The display register, which is included on the operator panel, provides a means of displaying and modifying the contents of the six 16-bit working registers when the computer is in the halt mode. An illuminating indicator is located directly above each of the 16 bit switches; a lighted indicator denotes a logic 1 and an unlighted indicator denotes a logic 0. When the computer is in the run mode, the contents of the S-register are displayed automatically.

2-11. X- AND Y-REGISTERS

These two 16-bit registers, designated X and Y, are accessed through the use of 30 index register instructions and 2 jump instructions described under paragraphs 3-22 and 3-23, respectively.

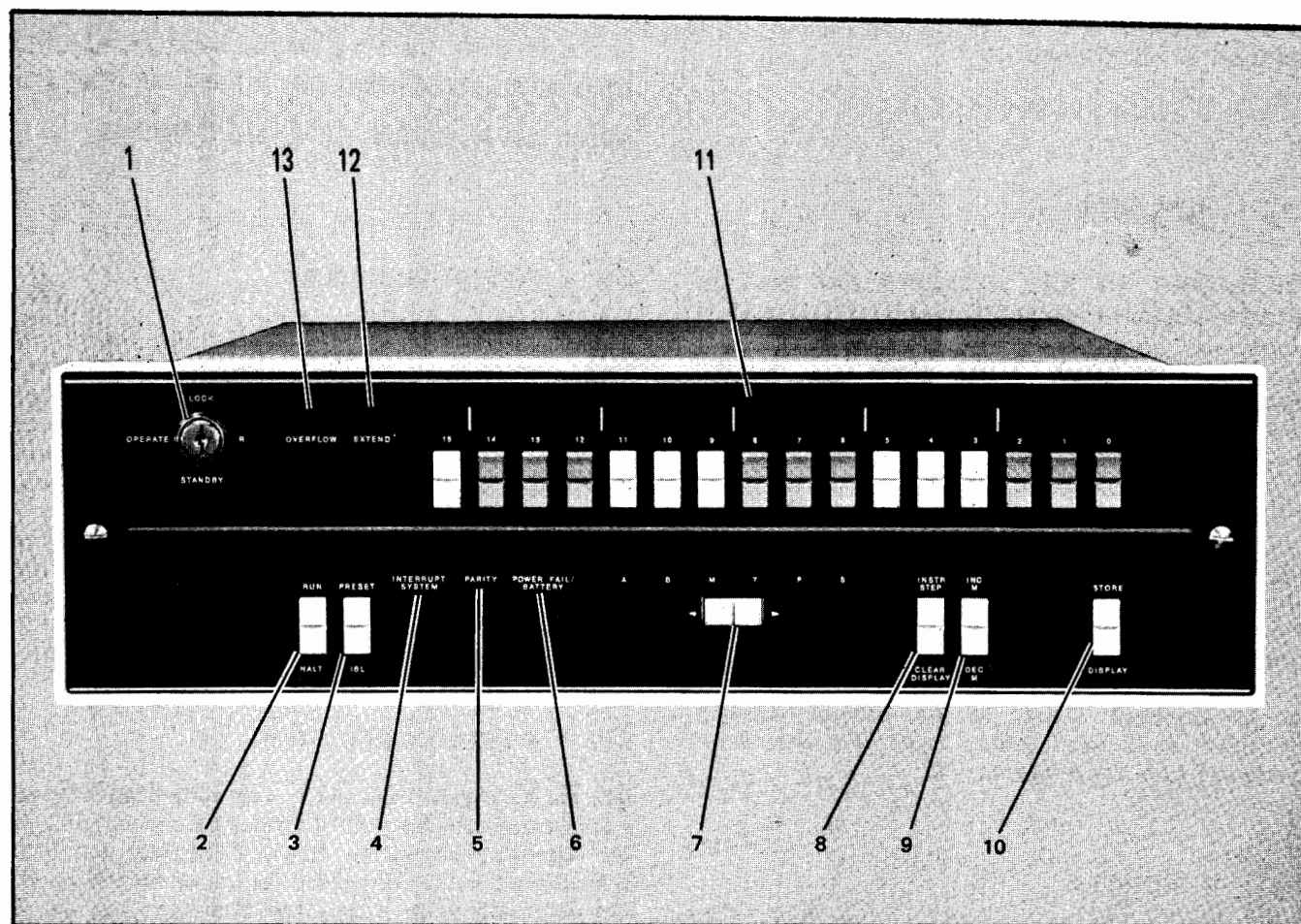
The location and function of the various controls and indicators mounted on the operator panel are illustrated in figure 2-1 and described in table 2-1. All operator panel controls (except the key-operated switch) are two-position, momentary-contact rocker switches; the status of the computer is displayed by light-emitting diodes.

2-14. REAR PANEL

Two switches are located on the rear panel. The \sim LINE switch controls the application of ac power to the computer power supply and the ventilating fans. The BATTERY switch allows the optional battery to be disconnected to prevent discharge during extended periods of nonoperation.

2-15. INTERNAL SWITCHES

Two toggle switches are mounted on the rear of the central processor unit printed-circuit assembly (PCA). The setting of the $\overline{\text{ARS/ARS}}$ switch determines the action that the computer will take in the event of a primary power failure



2270-1

Figure 2-1. Operator Panel Controls and Indicators

and the setting of the HLT PE/INT-IGNORE switch determines the action to be taken in the event of a parity error or memory protect violation. Programming considerations concerning these switches are given in section III.

2-16. BASIC OPERATING EXAMPLES

2-17. COLD START PROCEDURE

This procedure describes a cold start using the standard paper tape loader ROM, which will allow a program to be loaded via a punched-tape reader. At the rear of the computer, set the \sim LINE and BATTERY switches to ON and proceed as follows:

- a. On the operator panel, set key-operated switch to OPERATE.
- b. Press left half or right half of Register Select switch to select S-register.
- c. Press CLEAR DISPLAY and set bits 6 through 11 to display octal select code of tape reader.
- d. Set bits 15 and 14 to zeros to select standard paper tape loader ROM.
- e. Press STORE and then press IBL. The paper tape loader is now loaded into the uppermost 64 locations of memory and the select code of the tape reader is patched according to the contents of the S-register. The P-register is now pointing to the first instruction of the loader.
- f. Turn on tape reader and prepare it for reading. Press PRESET and then press RUN. The program will now be read into memory and the computer will halt with the T-register selected automatically. A successful load is indicated if the Display Register contents are 102077 (octal).

2-18. MANUAL LOADING

Short programs can be loaded manually from the operator panel as follows:

- a. Press left half or right half of Register Select switch to select M-register.
- b. Press CLEAR DISPLAY and set Display Register to starting address of program.
- c. Press STORE. Select T-register and change contents of Display Register to binary code of first instruction to be loaded; press STORE.
- d. Enter next instruction in Display Register and press STORE. (Pressing STORE with T-register selected automatically increments M-register.)
- e. Repeat step d until entire program has been loaded.

2-19. RUNNING PROGRAMS

To run a program after it has been loaded, proceed as follows:

- a. Press left or right half of Register Select switch to select P-register.
- b. Press CLEAR DISPLAY and set Display Register to starting address of program.
- c. Press STORE, PRESET, and RUN.

The RUN indicator will remain lighted as long as the program is running. If the key-operated switch is set to OPERATE, all operator panel controls except the Display Register, CLEAR DISPLAY, and HALT switches are disabled. The S-register is displayed automatically and can be changed manually via the Display Register bit switches. If the key-operated switch is set to LOCK, the RUN and HALT switches are disabled and all other switches are enabled within the constraints of the run and halt modes.

Table 2-1. Operator Panel Control and Indicator Functions

FIG. 2-1, INDEX NO.	NAME	FUNCTION																	
1	STANDBY/OPERATE/ LOCK/R	<p>Four-position, key-operated switch; does <i>not</i> control the application of ac power to the mainframe. (The \simLINE switch on the rear panel controls the ac power input to the processor power supply and ventilating fans.)</p> <p>STANDBY. Memory contents are sustained and the battery is on charge. CPU and I/O power are off; I/O interfaces may be installed or removed without damage. Key is removable.</p> <p>OPERATE. Power is supplied to the entire mainframe. Key is not removable.</p> <p>LOCK. The RUN and HALT switches are disabled; all other functions are enabled (within the constraints of the run/halt modes). Key is removable.</p> <p>R (reset). If memory data becomes invalid due to a prolonged power failure, the automatic restoration of power will preclude the CPU from running without operator intervention because the Power On (PON) signal is held off. (This condition will cause the POWER FAIL/BATTERY indicator to be lighted steadily; i.e., not blinking.) Rotating this switch momentarily to R allows the PON signal to be generated and reset this condition. Key is not removable.</p>																	
2	RUN/HALT	<p>RUN. Starts CPU and lights the RUN indicator. All operator panel functions are disabled except Display Register, CLEAR DISPLAY, and HALT. Pressing RUN automatically causes the S-register contents to be displayed, and no other register can be selected during the run mode; thus, the Display Register effectively becomes the S-register, which may be addressed as select code 01 by the program.</p> <p>HALT. Halts the computer at the end of the current instruction and turns off the RUN indicator. All other operator panel controls become enabled. The T-register is selected automatically for display.</p>																	
3	PRESET/IBL	<p>PRESET. Disables the interrupt system and clears the parity indicator and overflow bit (if set). From I/O channel 06 up, clears control flip-flops and sets flags. Pressing and holding PRESET upon the restoration of power will force a ARS condition (see paragraph 3-30).</p> <p>IBL (initial binary loader). Causes the contents of the standard paper tape loader ROM or the optional loader ROM's to be written into the uppermost 64 memory locations. Bits 15 and 14 of the S-register select the desired loader ROM as follows:</p> <table border="1"> <thead> <tr> <th colspan="2">BITS</th><th rowspan="2">LOADER SELECTED</th></tr> <tr> <th>15</th><th>14</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>Standard paper tape loader ROM</td></tr> <tr> <td>0</td><td>1</td><td>Option loader 1 ROM</td></tr> <tr> <td>1</td><td>0</td><td>Option loader 2 ROM</td></tr> <tr> <td>1</td><td>1</td><td>Option loader 3 ROM</td></tr> </tbody> </table> <p>Bits 6 through 11 of the S-register must be set to the octal select code of the loading device.</p>	BITS		LOADER SELECTED	15	14	0	0	Standard paper tape loader ROM	0	1	Option loader 1 ROM	1	0	Option loader 2 ROM	1	1	Option loader 3 ROM
BITS		LOADER SELECTED																	
15	14																		
0	0	Standard paper tape loader ROM																	
0	1	Option loader 1 ROM																	
1	0	Option loader 2 ROM																	
1	1	Option loader 3 ROM																	

Table 2-1. Operator Panel Control and Indicator Functions (Continued)

FIG. 2-1, INDEX NO.	NAME	FUNCTION
4	INTERRUPT SYSTEM	Indicates the status of the interrupt system. When lighted, the interrupt system is enabled (Flag set); when turned off, the interrupt system is disabled (Flag clear).
5	PARITY	Lights when a parity error occurs as a result of reading from memory. In the halt mode, the light can be turned off by pressing the PRESET switch. With the memory protect option installed (HP 2108A only) and the parity error interrupt enabled, the indicator is turned off automatically by a parity error interrupt and is therefore not ordinarily lighted long enough to be visible.
6	POWER FAIL/BATTERY	<p>If the power fail/automatic restart feature is enabled (i.e., internal ARS/ARS switch is set to ARS position as described in section III), the indicator will light when power is restored. This light can be turned off by pressing the PRESET switch in the halt mode.</p> <p>This light will flash on and off upon the application of mains power until the battery is known to contain enough power to sustain memory.</p>
7	◀ Register Select ▶	<p>In the halt mode, this switch allows any one of the six working registers (A, B, M, T, P, or S) to be selected for display and modification. Pressing the left half of the switch moves the "dot" indicator left; pressing the right half of the switch moves the "dot" indicator right. The register currently selected is indicated by the appropriate indicator light.</p> <p>After a programmed or manual halt, the T-register is selected automatically for display. In this case, the T-register holds the contents of the last accessed memory cell. In the case of a programmed halt, the halt instruction will be displayed.</p>
8	INSTR STEP/CLEAR DISPLAY	<p>INSTR STEP. Pressing and releasing this switch while in the halt mode advances the program to the next instruction. If the T-register indicator lights when the switch is released, infinite indirect addressing is indicated. Actuating this switch does not actually place the computer in the run mode.</p> <p>CLEAR DISPLAY. In the run or halt mode, clears the Display Register; i.e., contents become 000000.</p>
9	INC M/DEC M	<p>INC M. In the halt mode, increments the M-register contents.</p> <p>DEC M. In the halt mode, decrements the M-register contents.</p> <p>Note: Incrementing and decrementing occur even when the M-register is not displayed.</p>

Table 2-1. Operator Panel Control and Indicator Functions (Continued)

FIG. 2-1, INDEX NO.	NAME	FUNCTION
10	STORE/DISPLAY	<p>STORE. In the halt mode, stores the contents of the Display Register into the selected working register (A, B, M, T, P, or S). If the Register Select "dot" is pointing to T and STORE is pressed, the contents of the Display Register will be loaded into memory cell m, the M-register will be incremented automatically to $m + 1$, and the Display Register will <i>not</i> be updated. This latter feature allows the same data to be stored in consecutive memory locations (e.g., halts in the trap cells, same word into a buffer, etc.). If the Register Select "dot" is pointing to any register other than T, only that one register will be updated when STORE is pressed.</p> <p>DISPLAY. Places the present contents of the selected register into the Display Register. Used to recall a register after the Display Register contents have been changed or to display the next contents of the T-register after STORE is pressed.</p>
11	Display Register	In the halt mode, displays the contents of the register currently pointed to by the Register Select "dot;" only the S-register is displayed during the run mode. A logic 1 is signified when the displayed bit indicator is lighted; a logic 0 is signified when the displayed bit indicator is not lighted. Pressing the upper half of the switch sets that bit to a logic 1; pressing the lower half of the switch sets that bit to a logic 0. The Display Register is cleared to all zeros when the CLEAR DISPLAY switch is pressed.
12	EXTEND	In both the run and halt modes, continuously displays the contents of the extend register. When lighted, the extend bit is set (logic 1).
13	OVERFLOW	In both the run and halt modes, continuously displays the content of the overflow register. When lighted, the overflow bit is set (logic 1).

This section describes the software data formats and machine-language instruction coding required to operate the computer and its associated input/output system. A description of the vectored priority interrupt system is also included.

3-1. DATA FORMATS

As shown in figure 3-1, the basic data format is a 16-bit word in which bit positions are numbered from 0 through 15 in order of increasing significance. Bit position 15 of the data format is used for the sign bit; a logic 0 in this position indicates a positive number and a logic 1 in this position indicates a negative number. The data is assumed to be a whole number and the binary point is therefore assumed to be to the right of the number.

The basic word can also be divided into two 8-bit bytes or combined to form a 32-bit double word. The byte format is used for character-oriented input/output devices; packing two bytes of data into one 16-bit word is accomplished by software drivers. In I/O operations, the higher-order byte (byte 1) is the first to be transferred.

The integer double-word format is used for extended arithmetic in conjunction with the extended arithmetic instructions described under paragraphs 3-19 and 3-20. Bit position 15 of the most-significant word is the sign bit and the binary point is assumed to be to the right of the least-significant word. The integer value is expressed by the remaining 31 bits. When addressing a double word in memory, the address refers to the least-significant word location; the next higher memory address contains the most-significant word. When loaded into the accumulators, the B-register contains the most-significant word and the A-register contains the least-significant word.

The floating-point double-word format is used with floating-point software. Bit position 15 of the most-significant word is the mantissa sign and bit position 0 of the least-significant word is the exponent sign. Bits 1 through 7 of the least-significant word express the exponent and the remaining bits (bits 8 through 15 of the least-significant word and bits 0 through 14 of the most-significant word) express the mantissa. Since the mantissa is assumed to be a fractional value, the binary point appears to the left of the mantissa. Software drivers convert decimal numbers to this binary form and normalize the quantity expressed (sign and leading mantissa differ). If either the mantissa or the exponent is negative, that part is stored in two's complement form.

The number must be in the approximate range of 10^{-38} to 10^{+38} . When loaded into the accumulators, the B-register contains the most-significant word and the A-register contains the least-significant word.

Figure 3-1 also illustrates the octal notation for both single-length (16-bit) and double-length (32-bit) words. Each group of three bits, beginning at the right, is combined to form an octal digit. A single-length (16-bit) word can therefore be fully expressed by six octal digits and a double-length (32-bit) word can be fully expressed by 11 octal digits. Octal notation is not shown for byte or floating-point formats, since bytes normally represent characters and floating-point numbers are given in decimal form.

The range of representable numbers for single-word data is +32,767 to -32,768 (decimal) or +77,777 to -100,000 (octal). The range of representable numbers for double-word integer data is +2,147,483,647 to -2,147,483,648 (decimal) or +17,777,777,777 to -20,000,000,000 (octal).

3-2. MEMORY ADDRESSING

3-3. PAGING

The computer memory is logically divided into pages of 1,024 words each. A page is defined as the largest block of memory that can be directly addressed by the address bits of a single-length memory reference instruction. (Refer to paragraph 3-8.) These memory reference instructions use 10 bits (bits 0 through 9) to specify a memory address; thus, the page size is 1,024 locations (2000 octal). Octal addresses for each page, up to a maximum memory size of 32K, are listed in table 3-1.

Provision is made to directly address one of two pages: page zero (the base page consisting of locations 00000 through 01777) and the current page (the page in which the instruction itself is located). Memory reference instructions reserve bit 10 to specify one or the other of these two pages. To address locations on any other page, indirect addressing is used as described in following paragraphs. Page references are specified by bit 10 as follows:

- a. Logic 0 = Page Zero (Z).
- b. Logic 1 = Current Page (C).

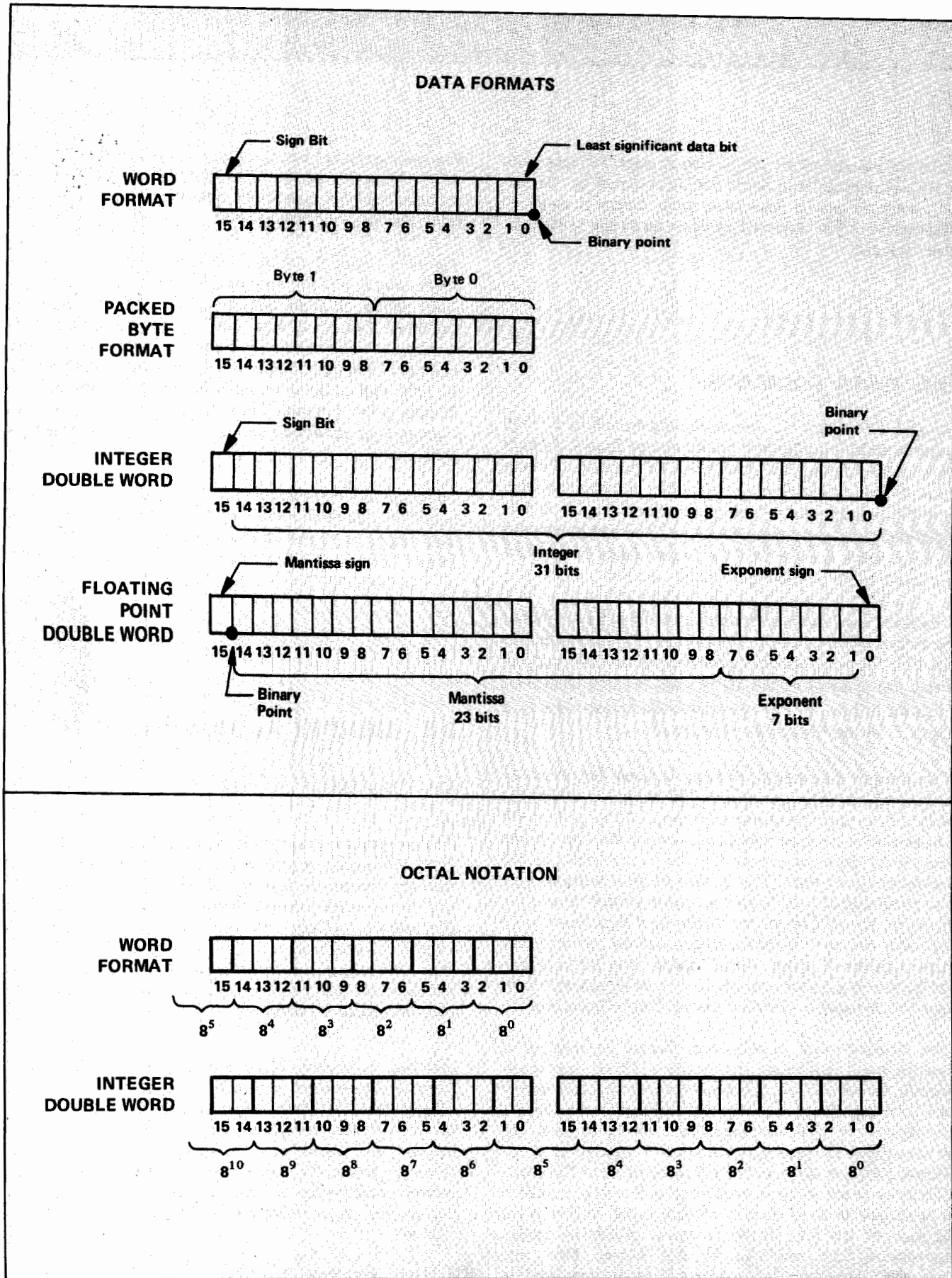


Figure 3-1. Data Formats and Octal Notation

Table 3-1. Memory Paging

MEMORY SIZE	PAGE	OCTAL ADDRESSES
4K ↓	0	00000 to 01777
	1	02000 to 03777
	2	04000 to 05777
	3	06000 to 07777
8K ↓	4	10000 to 11777
	5	12000 to 13777
	6	14000 to 15777
	7	16000 to 17777
12K ↓	8	20000 to 21777
	9	22000 to 23777
	10	24000 to 25777
	11	26000 to 27777
16K ↓	12	30000 to 31777
	13	32000 to 33777
	14	34000 to 35777
	15	36000 to 37777
24K ↓	16	40000 to 41777
	17	42000 to 43777
	18	44000 to 45777
	19	46000 to 47777
	20	50000 to 51777
	21	52000 to 53777
	22	54000 to 55777
	23	56000 to 57777
32K ↓	24	60000 to 61777
	25	62000 to 63777
	26	64000 to 65777
	27	66000 to 67777
	28	70000 to 71777
	29	72000 to 73777
	30	74000 to 75777
	31	76000 to 77777

3-4. DIRECT AND INDIRECT ADDRESSING

All memory reference instructions reserve bit 15 to specify either direct or indirect addressing. For single-length memory reference instructions, bit 15 of the instruction word is used; for extended arithmetic memory reference instructions, bit 15 of the address word is used. Indirect addressing uses the address part of the instruction to access another word in memory, which is taken as the new memory reference for the same instruction. This new address word is a full 16 bits long: 15 address bits plus another direct/indirect bit. The 15-bit length of the address permits access to any location in memory. If bit 15 again specifies indirect addressing, still another address is obtained; thus, multistep 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. Direct or indirect addressing is specified by bit 15 as follows:

- Logic 0 = Direct (D).
- Logic 1 = Indirect (I).



3-5. RESERVED MEMORY LOCATIONS

The first 64 memory locations of the base page (octal addresses 00000 through 00077) are reserved as listed in table 3-2. The first two locations are reserved as addresses for the two 16-bit accumulators (the A- and B-registers). Locations 00004 through 00077 are reserved for priority interrupts; as long as locations 00006 through 00077 do not have actual priority interrupt assignments, as determined by the options and input/output devices included in the system configuration, these locations can be used for programming purposes.

The uppermost 64 locations of memory for any given configuration are reserved for the initial binary loader. The initial binary loader is permanently resident in a read-only memory (ROM) and loaded into the uppermost 64 memory locations by a pushbutton switch on the operator panel. These 64 locations are not protected and can therefore be used for temporary storage of data, trap cells, buffers, etc.

Table 3-2. Reserved Memory Locations

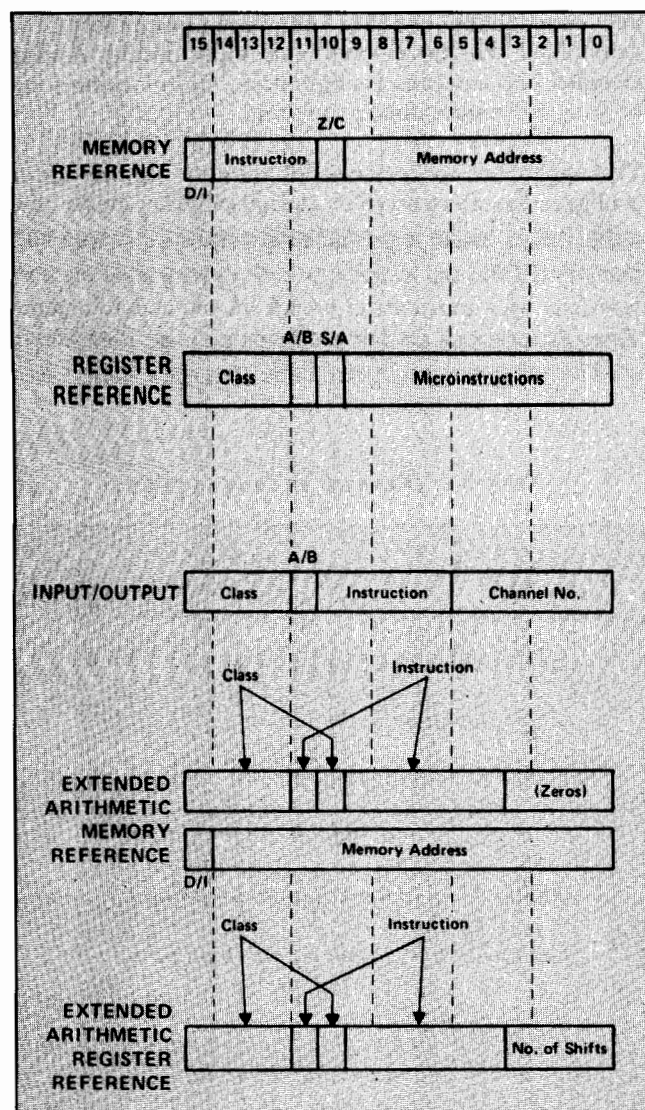
MEMORY LOCATION	PURPOSE
00000	A-register address.
00001	B-register address.
00002-00003	Exit sequence if contents of A-register and B-register are used as executable words.
00004	Power-fail interrupt (highest priority).
00005	Memory parity and memory protect interrupt.
00006	Reserved for dual-channel port controller (DCPC) channel 1.
00007	Reserved for dual-channel port controller (DCPC) channel 2.
00010-00077	Interrupt locations in decreasing order of priority; e.g., location 00010 has priority over 00011.

3-6. NONEXISTENT MEMORY

Nonexistent memory is defined as those locations not physically implemented in the machine. Any attempt to write into a nonexistent memory location will be ignored (no operation). Any attempt to read from a nonexistent memory location will return an all-zeros word (000000 octal); no parity error occurs.

3-7. BASE SET INSTRUCTION FORMATS

The base set of instructions are classified according to format. The five formats used are illustrated in figure 3-2 and described in following paragraphs. In all cases where a single bit is used to select one of two cases (e.g., D/I), the choice is made by coding a logic 0 or logic 1, respectively.



2270-3

Figure 3-2. Base Set Instruction Formats

3-8. MEMORY REFERENCE INSTRUCTIONS

This class of instructions, which combines an instruction code and a memory address into one 16-bit word, is used to

execute some function involving data in a specific memory location. Examples are storing, retrieving, and combining memory data to and from the accumulators (A- and B-registers) or causing the program to jump to a specified location in memory.

The memory cell referenced (i.e., the absolute address) is determined by a combination of 10 memory address bits (0 through 9) in the instruction word and 5 bits (10 through 14) assumed from the current contents of the M-register. This means that memory reference instructions can directly address any word in the current page; additionally, if the instruction is given in some location other than the base page (page zero), bit 10 (Z/C) of the instruction doubles the addressing range to 2,048 locations by allowing the selection of either page zero or the current page. (This causes bits 10 through 14 of the address contained in the M-register to be set to zero instead of assuming the current contents of the M-register.) This feature provides a convenient linkage between all pages of memory, since page zero can be reached directly from any other page.

As discussed under paragraph 3-4, bit 15 is used to specify direct or indirect memory addressing. Note also that since the A- and B-registers are addressable, any single-word memory reference instruction can apply to either of these registers as well as to memory cells. For example, an ADA 0001 instruction adds the contents of the B-register (address 0001) to the contents currently held in the A-register; specify page zero for these operations since the addresses of the A- and B-registers are on page zero.

3-9. REGISTER REFERENCE INSTRUCTIONS

In general, the register reference instructions manipulate bits in the A-register, B-register, and E-register; there is no reference to memory. This group includes 39 basic microinstructions which may be combined to form a one-word multiple instruction that can operate in various ways on the contents of the A-, B-, and E-registers. These 39 instructions are divided into two subgroups: the shift/rotate group (SRG) and the alter/skip group (ASG). The appropriate subgroup is specified by bit 10 (S/A). Typical operations are clear and/or complement a register, conditional skips, and register increment.

3-10. INPUT/OUTPUT INSTRUCTIONS

The input/output instructions use bits 6 through 11 for a variety of I/O instructions and bits 0 through 5 to apply the instructions to a specific I/O channel. This provides the means of controlling all peripherals connected to the I/O channels and for transferring data to and from these peripherals. Included also in this group are instructions to control the interrupt system, overflow bit, and computer halt.

3-11. EXTENDED ARITHMETIC MEMORY REFERENCE INSTRUCTIONS

As the single-word memory reference instruction described previously, the extended arithmetic memory reference instructions include an instruction code and a memory address. In this case, however, two words are required. The first word specifies the extended arithmetic class (bits 12 through 15 and 10) and the instruction code (bits 4 through 9 and 11); bits 0 through 3 are not needed and are coded with zeros. The second word specifies the memory address of the operand. Since the full 15 bits are used for the address, this type of instruction may directly address any location in memory. As with all memory reference instructions, bit 15 is used to specify direct or indirect addressing. Operations performed by this class of instructions are integer multiply and divide (using double-length product and dividend) and double load and double store.

3-12. EXTENDED ARITHMETIC REGISTER REFERENCE INSTRUCTIONS

This class of instructions provides long shifts and rotates on the combined contents of the A- and B-registers. Bits 12 through 15 and 10 identify the instruction class; bits 4 through 9 and 11 specify the direction and type of shift; and bits 0 through 3 control the number of shifts, which can range from 1 to 16 places.

3-13. BASE SET INSTRUCTION CODING

Machine language coding for the base set of instructions are provided in following paragraphs. Definitions for these instructions are grouped according to the instruction type: memory reference, register reference, input/output, extended arithmetic memory reference, and extended arithmetic register reference.

Directly above each definition is a diagram showing the machine language coding for that instruction. The gray shaded bits code the instruction type and the blue shaded bits code the specific instruction. Unshaded bits are further defined in the introduction to each instruction type. The mnemonic code and instruction name are included above each diagram.

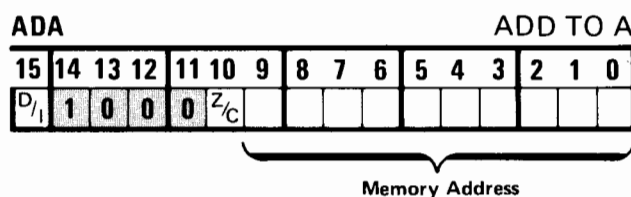
In all cases where an additional bit is used to specify a secondary function (D/I, Z/C, or H/C), the choice is made by coding a logic 0 or logic 1, respectively. In other words, a logic 0 codes D (direct addressing), Z (zero page), or H (hold flag); a logic 1 codes I (indirect addressing), C (current page), or C (clear flag).

3-14. MEMORY REFERENCE INSTRUCTIONS

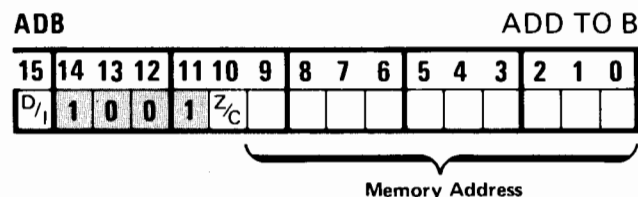
The following 14 memory reference instructions execute a function involving data in memory. Bits 0 through 9

specify the affected memory location on a given memory page or, if indirect addressing is specified, the next address to be referenced. Indirect addressing may be continued to any number of levels; when bit 15 (D/I) is a logic 0 (specifying direct addressing), that location will be taken as the effective address. The A- and B-registers may be addressed as locations 00000 and 00001 (octal), respectively.

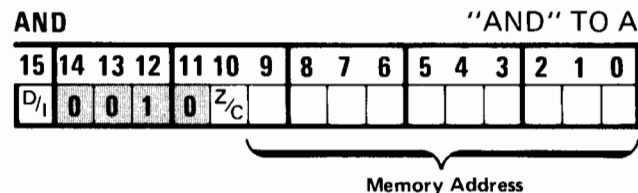
If bit 10 (Z/C) is a logic 0, the memory address is on page zero; if bit 10 is a logic 1, the memory address is on the current page. If the A- or B-register is addressed, bit 10 must be a logic 0 to specify page zero, unless the current page is page zero.



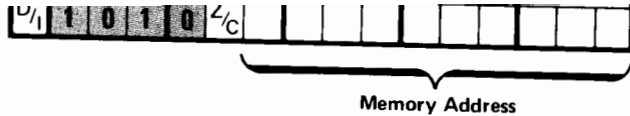
Adds the contents of the addressed memory location to the contents of the A-register. The sum remains in the A-register and the contents of the memory cell are unaltered. The result of this addition may set the extend bit or the overflow bit.



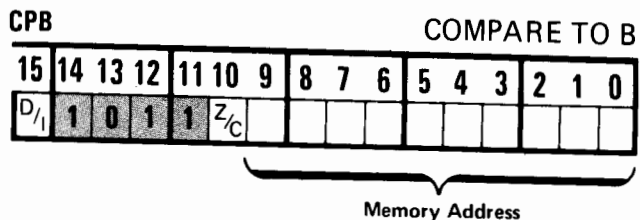
Adds the contents of the addressed memory location to the contents of the B-register. The sum remains in the B-register and the contents of the memory cell are unaltered. The result of this addition may set the extend bit or the overflow bit.



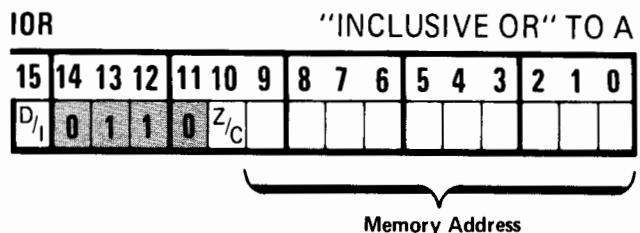
Combines the contents of the addressed memory location and the contents of the A-register by performing a logical "and" operation. The contents of the memory cell are unaltered.



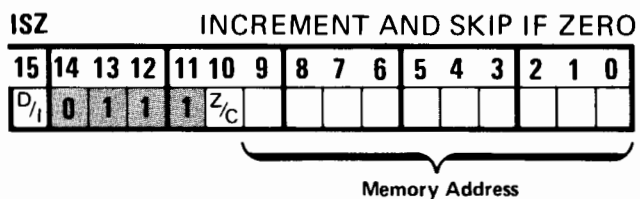
Compares the contents of the addressed memory location with the contents of the A-register. If the two 16-bit words are not identical, the next instruction is skipped; i.e., the P-register advances two counts instead of one count. If the two words are identical, the next sequential instruction is executed. Neither the A-register contents nor memory cell contents are altered.



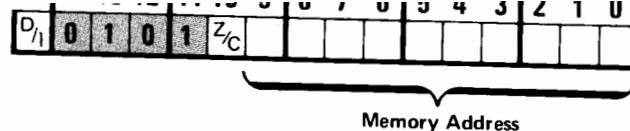
Compares the contents of the addressed memory location with the contents of the B-register. If the two 16-bit words are not identical, the next instruction is skipped; i.e., the P-register advances two counts instead of one count. If the two words are identical, the next sequential instruction is executed. Neither the B-register contents nor memory cell contents are altered.



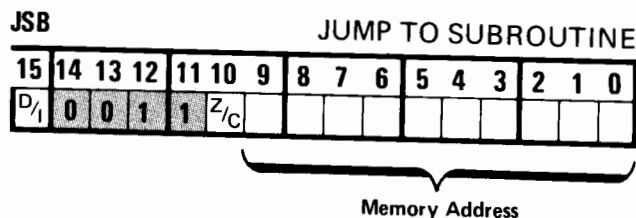
Combines the contents of the addressed memory location and the contents of the A-register by performing a logical "inclusive or" operation. The contents of the memory cell are unaltered.



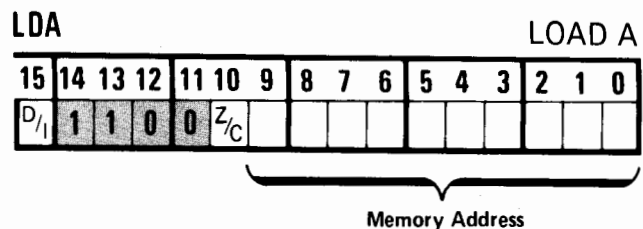
Adds one to the contents of the addressed memory location. If the result of this operation is zero (memory contents incremented from 177777 to 000000), the next instruction is skipped; i.e., the P-register is advanced two counts instead of one count. If the result of this operation is not zero, the next sequential instruction is executed. In either case, the incremented value is written back into the memory cell. An ISZ instruction referencing locations 0000 or 0001 (A- or B-register) cannot set the extend bit or the overflow bit.



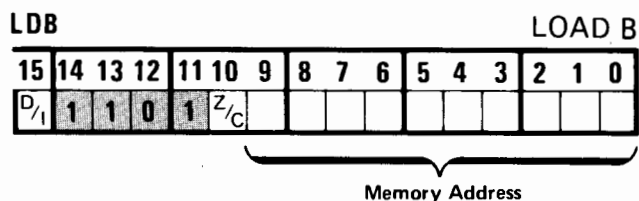
Transfers control to the addressed memory location. That is, a JMP causes the P-register count to set according to the memory address portion of the JMP instruction so that the next instruction will be read from that location.



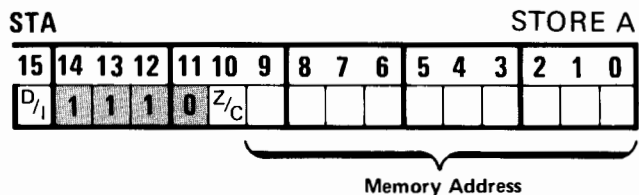
This instruction, executed in location P (P-register count), causes the computer control to jump unconditionally to the memory location (m) specified by the memory address portion of the JSB instruction. The contents of the P-register plus one (return address) is stored in memory location m, and the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



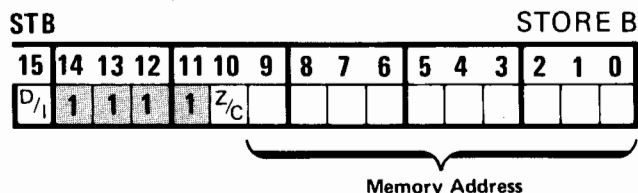
Loads the contents of the addressed memory location into the A-register. The contents of the memory cell are unaltered.



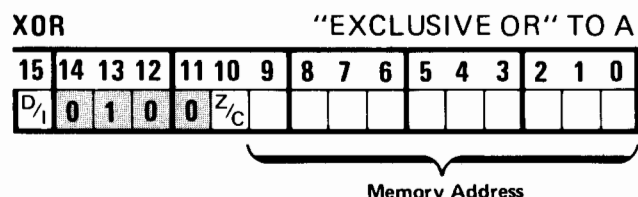
Loads the contents of the addressed memory location into the B-register. The contents of the memory cell are unaltered.



Stores the contents of the A-register in the addressed memory location. The previous contents of the memory cell are lost; the A-register contents are unaltered.



Stores the contents of the B-register in the addressed memory location. The previous contents of the memory cell are lost; the B-register contents are unaltered.



Combines the contents of the addressed memory location and the contents of the A-register by performing a logical "exclusive or" operation. The contents of the memory cell are unaltered.

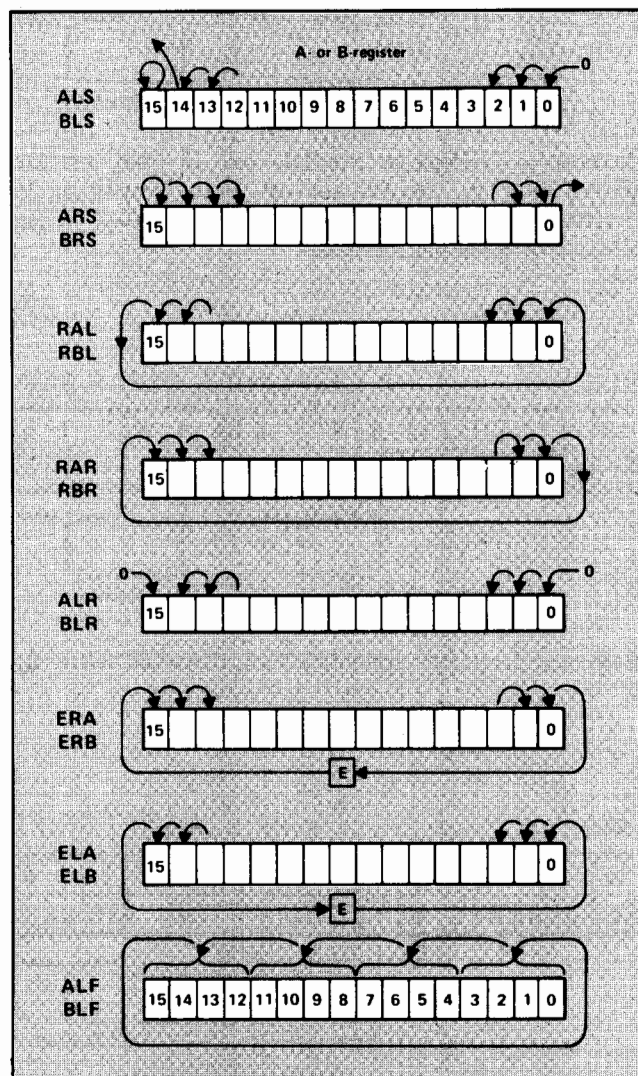
3-15. REGISTER REFERENCE INSTRUCTIONS

The 39 register reference instructions execute functions on data contained in the A-register, B-register, and E-register. These instructions are divided into two groups: the shift/rotate group (SRG) and the alter/skip group (ASG). In each group, several instructions may be combined into one word and are thus individually termed "microinstructions." Since the two groups perform separate and distinct functions, microinstructions from the two groups cannot be mixed. Unshaded bits in the coding diagrams are available for combining other microinstructions.

3-16. SHIFT/ROTATE GROUP. The 20 instructions in the shift/rotate group (SRG) are defined first; this group is specified by setting bit 10 to a logic 0. A comparison of the various shift/rotate functions are illustrated in figure 3-3. Rules for combining microinstructions in this group are as follows (refer to table 3-3):

- Only one microinstruction can be chosen from each of the two multiple-choice columns.
- References can be made to either the A-register or B-register, but not both.
- Sequence of execution is from left to right.
- In machine code, use zeros to exclude unwanted microinstructions.
- Code a logic 1 in bit position 9 to enable shifts or rotates in the first position; code a logic 1 in bit position 4 to enable shifts or rotates in the second position.
- The extend bit is not affected unless specifically stated. However, if a "rotate-with-E" instruction (ELA, ELB, ERA, or ERB) is coded but disabled by a logic 0 in bit position 9 and/or position 4, the E-register will be

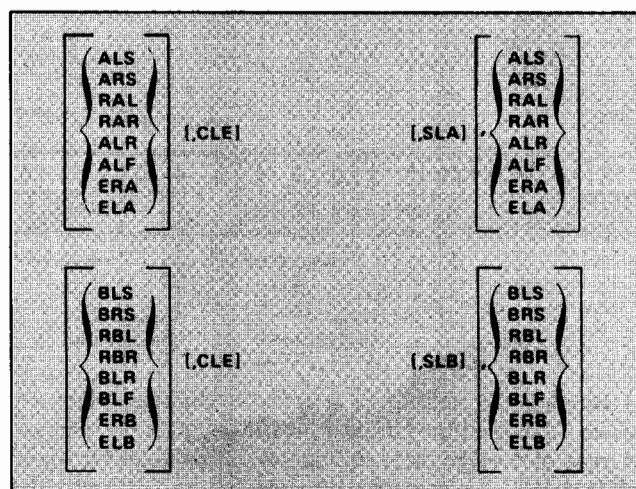
updated even though the A- or B-register contents are not affected; to avoid this situation, code a "no operation" (three zeros) in the first and/or second positions.

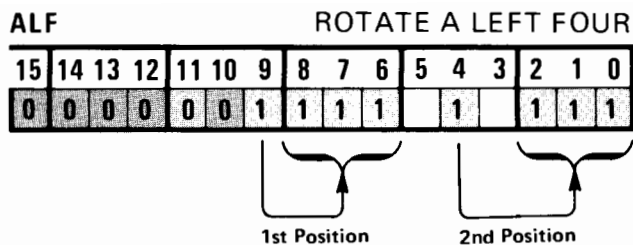


2270-4

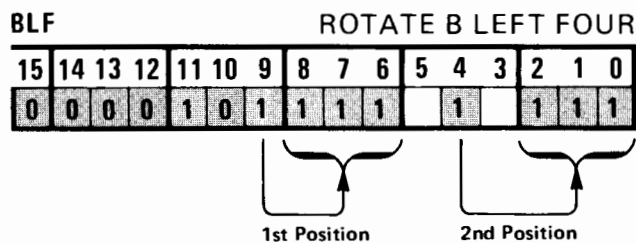
Figure 3-3. Shift and Rotate Functions

Table 3-3. Shift/Rotate Group Combining Guide

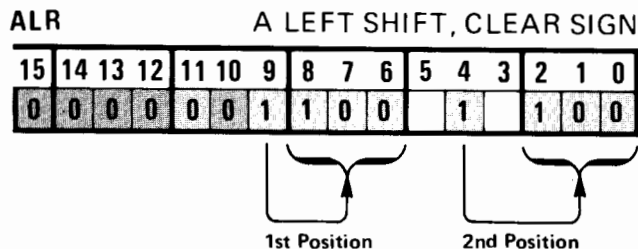




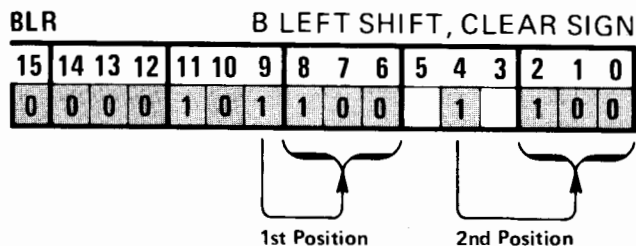
Rotates the A-register contents (all 16 bits) left four places. Bits 15, 14, 13, and 12 rotate around to bit positions 3, 2, 1, and 0, respectively. Equivalent to four successive RAL instructions.



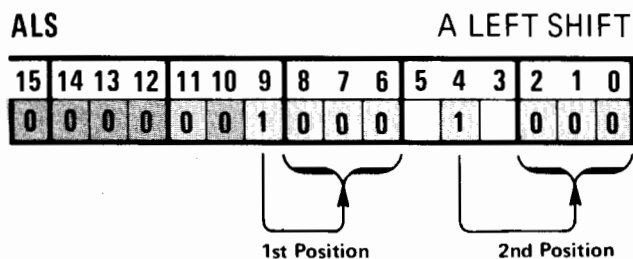
Rotates the B-register contents (all 16 bits) left four places. Bits 15, 14, 13, and 12 rotate around to bit positions 3, 2, 1, and 0, respectively. Equivalent to four successive RBL instructions.



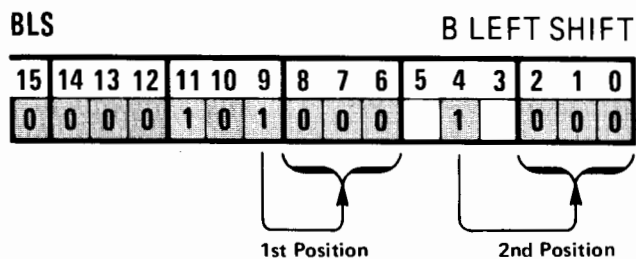
Shifts the A-register contents left one place and clears sign bit 15.



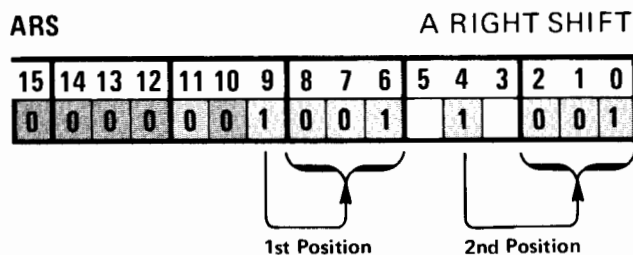
Shifts the B-register contents left one place and clears sign bit 15.



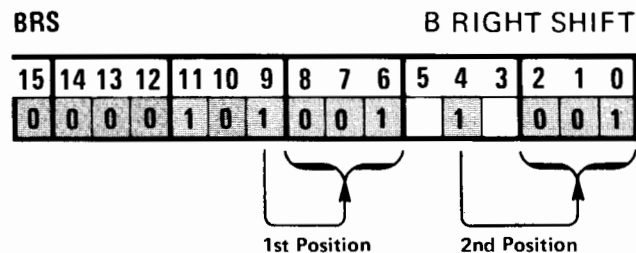
Arithmetically shifts the A-register contents left one place, 15 magnitude bits only; bit 15 (sign) is not affected. The bit shifted out of bit position 14 is lost; a logic 0 replaces vacated bit position 0.



Arithmetically shifts the B-register contents left one place, 15 magnitude bits only; bit 15 (sign) is not affected. The bit shifted out of bit position 14 is lost; a logic 0 replaces vacated bit position 0.



Arithmetically shifts the A-register contents right one place, 15 magnitude bits only; bit 15 (sign) is not affected. A copy of the sign bit is shifted into bit position 14; the bit shifted out of bit position 0 is lost.



Arithmetically shifts the B-register contents right one place, 15 magnitude bits only; bit 15 (sign) is not affected. A copy of the sign bit is shifted into bit position 14; the bit shifted out of bit position 0 is lost.

CLE**CLEAR E**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Clears the E-register; i.e., the extend bit becomes a logic 0.

NOP**NO OPERATION**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

This all-zeros instruction causes a no-operation cycle.

ELA**ROTATE E LEFT WITH A**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	1	1	0	1	0	0	1	1	0

1st Position 2nd Position

Rotates the E-register content left with the A-register contents (one place). The E-register content rotates into bit position 0; bit 15 rotates into the E-register.

RAL**ROTATE A LEFT**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	0	1	0	1	0	0	1	0	0

1st Position 2nd Position

Rotates the A-register contents left one place (all 16 bits). Bit 15 rotates into bit position 0.

ELB**ROTATE E LEFT WITH B**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	1	1	1	0	1	0	0	1	1	0

1st Position 2nd Position

Rotates the E-register content left with the B-register contents (one place). The E-register content rotates into bit position 0; bit 15 rotates into the E-register.

RAR**ROTATE A RIGHT**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	0	1	1	1	0	0	1	1	0

1st Position 2nd Position

Rotates the A-register contents right one place (all 16 bits). Bit 0 rotates into bit position 15.

ERA**ROTATE E RIGHT WITH A**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	1	0	1	1	0	0	1	0	1

1st Position 2nd Position

Rotates the E-register content right with the A-register contents (one place). The E-register content rotates into bit position 15; bit 0 rotates into the E-register.

RBL**ROTATE B LEFT**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	1	0	1	0	1	0	0	1	0	0

1st Position 2nd Position

Rotates the B-register contents left one place (all 16 bits). Bit 15 rotates into bit position 0.

ERB**ROTATE E RIGHT WITH B**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	1	1	0	1	1	0	0	1	0	1

1st Position 2nd Position

Rotates the E-register content right with the B-register contents (one place). The E-register content rotates into bit position 15; bit 0 rotates into the E-register.

RBR**ROTATE B RIGHT**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	1	0	1	1	1	0	0	1	1	0

1st Position 2nd Position

Rotates the B-register contents right one place (all 16 bits). Bit 0 rotates into bit position 15.

SLA SKIP IF LSB OF A IS ZERO

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0							1			

Skips the next instruction if the least-significant bit (bit 0) of the A-register is a logic 0.

SLB SKIP IF LSB OF B IS ZERO

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0							1			

Skips the next instruction if the least-significant bit (bit 0) of the B-register is a logic 0.

3-17. ALTER/SKIP GROUP. The 19 instructions comprising the alter/skip group (ASG) are defined next. This group is specified by setting bit 10 to a logic 1. Rules for combining microinstructions are as follows (refer to table 3-4):

- Only one microinstruction can be chosen from each of the two multiple-choice columns.
- References can be made to either the A-register or B-register, but not both.
- Sequence of execution is from left to right.
- If two or more skip functions are combined, the skip function will occur if either or both conditions are met. One exception exists: refer to the RSS instruction.
- In machine code, use zeros to exclude unwanted microinstructions.

Table 3-4. Alter/Skip Group Combining Guide

<div> <div>CLA</div> <div>CMA</div> <div>CCA</div> </div>	[SEZ]	<div> <div>CLE</div> <div>CME</div> <div>CCE</div> </div>	[SSA] [SLA] [INA] [SZA] [RSS]
<div> <div>CLB</div> <div>CMB</div> <div>CCB</div> </div>	[SEZ]	<div> <div>CLE</div> <div>CME</div> <div>CCE</div> </div>	[SSB] [SLB] [INB] [SZB] [RSS]

CCA CLEAR AND COMPLEMENT A

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	1								

Clears and complements the A-register contents; i.e., the contents of the A-register become 177777 (octal). This is the two's complement form of -1.

CCB CLEAR AND COMPLEMENT B

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	1	1	1								

Clears and complements the B-register contents; i.e., the contents of the B-register become 177777 (octal). This is the two's complement form of -1.

CCE CLEAR AND COMPLEMENT E

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0		1			1	1						

Clears and complements the E-register content (extend bit); i.e., the extend bit becomes a logic 1.

CLA CLEAR A

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	1								

Clears the A-register; i.e., the contents of the A-register become 000000 (octal).

CLB CLEAR B

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	1	0	1								

Clears the B-register; i.e., the contents of the B-register become 000000 (octal).

CLE CLEAR E

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0		1			0	1						

Clears the E-register; i.e., the extend bit becomes a logic 0.

CMA COMPLEMENT A

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	0								

Complements the A-register contents (one's complement).

CMB COMPLEMENT B

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	1	1	0								

Complements the B-register contents (one's complement).

CME**COMPLEMENT E**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0		1			1	0						

Complements the E-register content (extend bit).

INA**INCREMENT A**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1								1		

Increments the A-register by one. Can result in setting the extend bit or the overflow bit.

INB**INCREMENT B**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	1								1		

Increments the B-register by one. Can result in setting the extend bit or the overflow bit.

RSS**REVERSE SKIP SENSE**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0		1										1

Skip occurs for any of the following skip instructions, if present, when the non-zero condition is met. An RSS without a skip instruction in the word causes an unconditional skip. If a word with RSS also includes both SSA and SLA (or SSB and SLB), bits 15 and 0 must both be logic 1's for a skip to occur; in all other cases, a skip occurs if one or more skip conditions are met.

SEZ**SKIP IF E IS ZERO**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0		1					1					

Skips the next instruction if the E-register content (extend bit) is a logic 0.

SLA**SKIP IF LSB OF A IS ZERO**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1								1		

Skips the next instruction if the least-significant bit (bit 0) of the A-register is a logic 0; i.e., skips if an even number is in the A-register.

SLB**SKIP IF LSB OF B IS ZERO**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	1							1			

Skips the next instruction if the least-significant bit (bit 0) of the B-register is a logic 0; i.e., skips if an even number is in the B-register.

SSA**SKIP IF SIGN OF A IS ZERO**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1					1					

Skips the next instruction if the sign bit (bit 15) of the A-register is a logic 0; i.e., skips if a positive number is in the A-register.

SSB**SKIP IF SIGN OF B IS ZERO**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	1					1					

Skips the next instruction if the sign bit (bit 15) of the B-register is a logic 0; i.e., skips if a positive number is in the B-register.

SZA**SKIP IF A IS ZERO**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1									1	

Skips the next instruction if the A-register contents are zero (16 zeros).

SZB**SKIP IF B IS ZERO**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	1									1	

Skips the next instruction if the B-register contents are zero (16 zeros).

3-18. INPUT/OUTPUT INSTRUCTIONS

The following input/output instructions provide the capability of setting or clearing the I/O flag and control bits, testing the state of the overflow and the I/O flag bits, and transferring data between specific I/O devices and the A- and B-registers. In addition, specific instructions in this group control the vectored priority interrupt system and can cause a programmed halt.

distinguishes between set control and clear control; otherwise, bit 11 may be a logic 0 or a logic 1 without affecting the instruction (although the assembler will assign zeros in this case). In those instructions where bit position 9 includes the letters H/C, the programmer has the choice of holding (logic 0) or clearing (logic 1) the device flag after executing the instruction. (Exception: the H/C bit associated with instructions SOC and SOS holds or clears the overflow bit instead of the device flag.) Bits 8, 7, and 6 specify the appropriate I/O instruction and bits 5 through 0 form a two-digit octal select code (address) to apply the instruction to one of up to 64 input/output devices or functions.

CLC

CLEAR CONTROL

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	1	H/C	1	1	1						

Select Code

Clears the control bit of the selected I/O channel or function. This turns off the specific device channel and prevents it from interrupting. A CLC 00 instruction clears all control bits from select code 06 upward, effectively turning off all I/O devices.

CLF

CLEAR FLAG

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0		1	1	0	0	1						

Select Code

Clears the flag of the selected I/O channel or function. A CLF 00 instruction disables the interrupt system for all select codes except power fail (select code 04) and parity error (select code 05), which are always enabled; this does not affect the status of the individual channel flags.

CLO

CLEAR OVERFLOW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	1	1	0	0	1	0	0	0	0	0	1

Clears the overflow bit.

HLT

HALT

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0		1	H/C	0	0	0						

Select Code

Halts the computer and holds or clears the flag of the selected I/O channel. The HLT instruction has the same effect as pressing the operator panel HALT pushbutton; i.e., the RUN indicator turns off and all operator panel control switches are enabled. The HLT instruction will be contained in the T-register, which is selected and displayed automatically when the computer halts. The P-register contents will normally contain the HLT location plus one ($P + 1$).

LIA

LOAD INTO A

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	1	H/C	1	0	1						

Select Code

Loads the contents of the I/O buffer associated with the selected device into the A-register.

LIB

LOAD INTO B

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	1	H/C	1	0	1						

Select Code

Loads the contents of the I/O buffer associated with the selected device into the B-register.

MIA

MERGE INTO A

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	1	H/C	1	0	0						

Select Code

By executing a logical "inclusive or" function, merges the contents of the I/O buffer associated with the selected device into the A-register.

MIB

MERGE INTO B

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	1	H/C	1	0	0						

Select Code

By executing a logical "inclusive or" function, merges the contents of the I/O buffer associated with the selected device into the B-register.

OTA**OUTPUT A**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	1	H/C	1	1	0						

Select Code

Outputs the contents of the A-register to the I/O buffer associated with the selected device. If the I/O buffer is less than 16 bits in length, the least-significant bits of the A-register are normally loaded. (Some exceptions to this exist, depending on the type of output device.) The contents of the A-register are not altered.

OTB**OUTPUT B**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	1	H/C	1	1	0						

Select Code

Outputs the contents of the B-register to the I/O buffer associated with the selected device. If the I/O buffer is less than 16 bits in length, the least-significant bits of the B-register are normally loaded. (Some exceptions to this exist, depending on the type of output device.) The contents of the B-register are not altered.

SFC**SKIP IF FLAG CLEAR**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0		1	0	0	1	0						

Select Code

Skips the next programmed instruction if the flag of the selected channel is clear (device busy).

SFS**SKIP IF FLAG SET**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0		1	0	0	1	1						

Select Code

Skips the next programmed instruction if the flag of the selected channel is set (device ready).

SOC**SKIP IF OVERFLOW CLEAR**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	1	H/C	0	1	0	0	0	0	0	0	1

Skips the next programmed instruction if the overflow bit is clear. Use the H/C bit (bit 9) to either hold or clear the overflow bit following the completion of this instruction (whether the skip is taken or not).

SOS**SKIP IF OVERFLOW SET**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	1	H/C	0	1	1	0	0	0	0	0	1

Skips the next programmed instruction if the overflow bit is set. Use the H/C bit (bit 9) to either hold or clear the overflow bit following the completion of this instruction (whether the skip is taken or not).

STC**SET CONTROL**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	1	H/C	1	1	1						

Select Code

Sets the control bit of the selected I/O channel or function.

STF**SET FLAG**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0		1	0	0	0	1						

Select Code

Sets the flag of the selected I/O channel or function. An STF 00 instruction enables the interrupt system for all select codes except power fail (select code 04) and parity error (select code 05), which are always enabled.

STO**SET OVERFLOW**

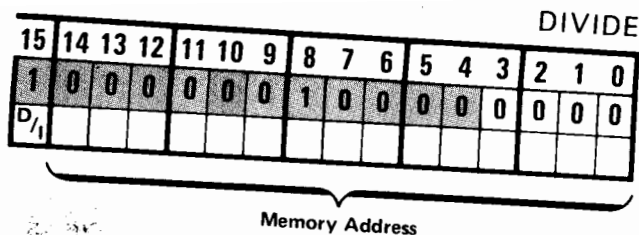
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1

Sets the overflow bit.

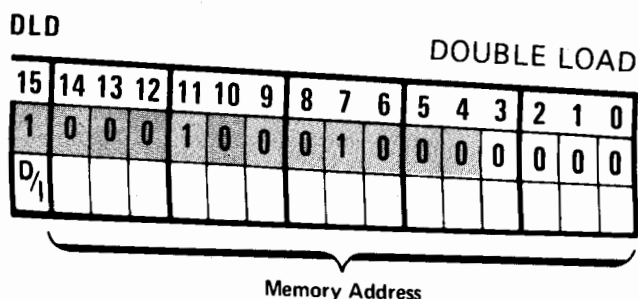
3-19. EXTENDED ARITHMETIC MEMORY REFERENCE INSTRUCTIONS

The four extended arithmetic memory reference instructions provide for integer multiply and divide and for loading and storing double-length words to and from the A- and B-registers. The complete instruction requires two words: one for the instruction code and one for the address. When stored in memory, the instruction word is the first to be fetched; the address word is in the next sequential location.

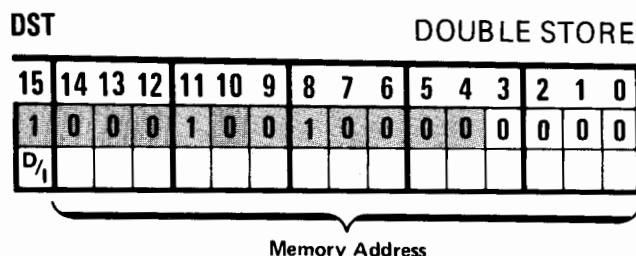
Since 15 bits are available for the address, these instructions can directly address any location in memory. As for all memory reference instructions, indirect addressing to any number of levels may also be used. A logic 0 in bit position 15 specifies direct addressing; a logic 1 specifies indirect addressing.



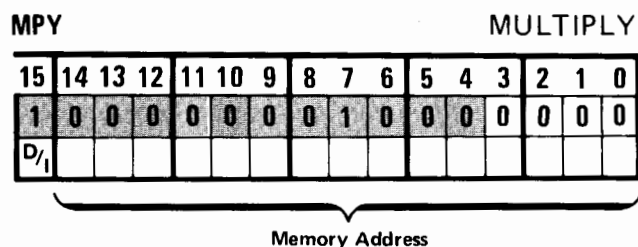
Divides a double-word integer in the combined B- and A-registers by a 16-bit integer in the addressed memory location. The result is a 16-bit integer quotient in the A-register and a 16-bit integer remainder in the B-register. Overflow can result from an attempt to divide by zero, or from an attempt to divide by a number too small for the dividend. In the former case (divide by zero), execution will be attempted with unpredictable results left in the B- and A-registers. In the latter case (divisor too small), the division will not be attempted and the B- and A-register contents will be unchanged except that a negative quantity will be made positive. If there is no divide error, the overflow bit is cleared.



Loads the contents of addressed memory location *m* (and *m* + 1) into the A- and B-registers, respectively.



Stores the double-word quantity in the A- and B-registers into addressed memory locations *m* (and *m* + 1), respectively.



Multiplies a 16-bit integer in the A-register by a 16-bit integer in the addressed memory location. The resulting double-length integer product resides in the B- and

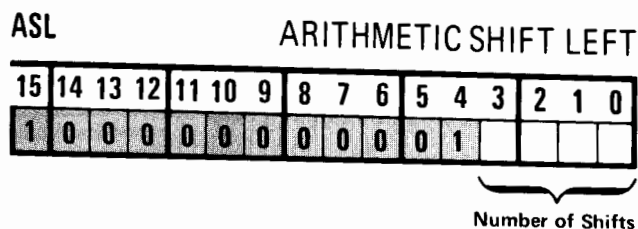
A-registers, with the B-register containing the sign bit and the most-significant 15 bits of the quantity. The A-register may be used as an operand (i.e., memory address 0), resulting in an arithmetic square. Overflow cannot occur because the instruction clears the overflow bit.

3-20. EXTENDED ARITHMETIC REGISTER REFERENCE INSTRUCTIONS

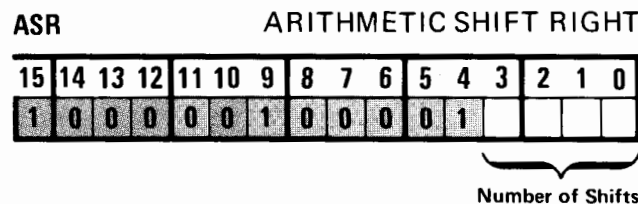
The six extended arithmetic register reference instructions provide various types of shifting operations on the combined contents of the B- and A-registers. The B-register is considered to be to the left (most-significant word) and the A-register is considered to be to the right (least-significant word). An example of each type of shift operation is illustrated in figure 3-4.

The complete instruction is given in one word and includes four bits (unshaded) to specify the number of shifts (1 to 16). By viewing these four bits as a binary-coded number, the number of shifts is easily expressed; i.e., binary-coded 1 = 1 shift, binary-coded 2 = 2 shifts . . . binary-coded 15 = 15 shifts. The maximum number of 16 shifts is coded with four zeros, which essentially exchanges the contents of the B- and A-registers.

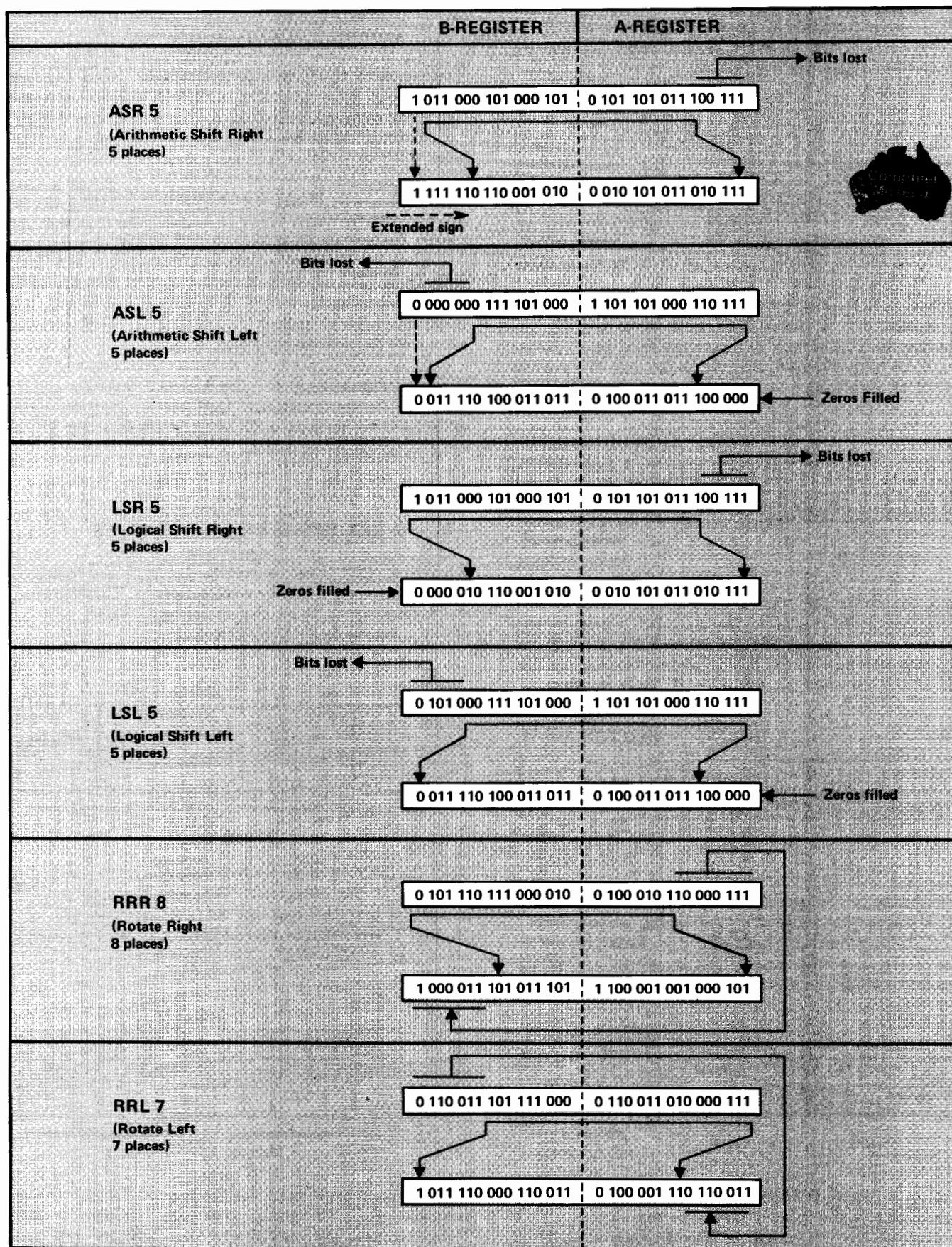
The extend bit is not affected by any of the following instructions. Except for the arithmetic shifts, overflow also is not affected.



Arithmetically shifts the combined contents of the B- and A-registers left *n* places. The value of *n* may be any number from 1 through 16. Zeros are filled into vacated low-order positions of the A-register. The sign bit is not affected, and data bits are lost out of bit position 14 of the B-register. If any one of the lost bits is a significant data bit ("1" for positive numbers, "0" for negative numbers), the overflow bit will be set; otherwise, overflow will be cleared during execution. See ASL example in figure 3-4. Note that two additional shifts in this example would cause an error by losing a significant '1'.



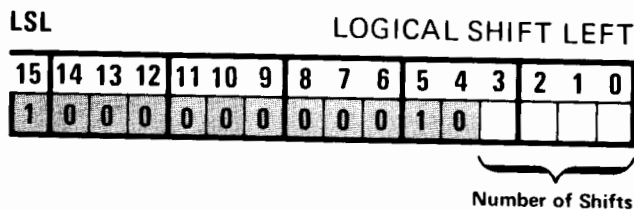
Arithmetically shifts the combined contents of the B- and A-registers right *n* places. The value of *n* may be any number from 1 through 16. The sign bit is unchanged and



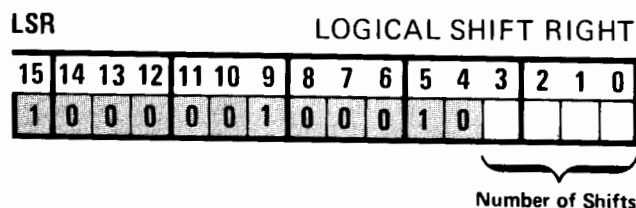
2270-5

Figure 3-4. Examples of Double-Word Shifts and Rotates

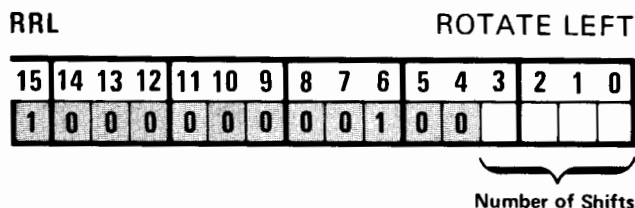
is extended into bit positions vacated by the right shift. Data bits shifted out of the least-significant end of the A-register are lost. Overflow cannot occur because the instruction clears the overflow bit.



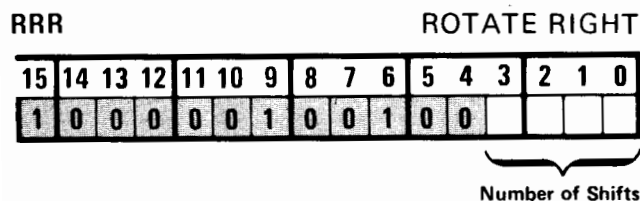
Logically shifts the combined contents of the B- and A-registers left n places. The value of n may be any number from 1 through 16. Zeros are filled into vacated low-order bit positions of the A-register; data bits are lost out of the high-order bit positions of the B-register.



Logically shifts the combined contents of the B- and A-registers right n places. The value of n may be any number from 1 through 16. Zeros are filled into vacated high-order bit positions of the B-register; data bits are lost out of the low-order bit positions of the A-register.



Rotates the combined contents of the B- and A-registers left n places. The value of n may be any number from 1 through 16. No bits are lost or filled in. Data bits shifted out of the high-order end of the B-register are rotated around to enter the low-order end of the A-register.



Rotates the combined contents of the B- and A-registers right n places. The value of n may be any number from 1 through 16. No bits are lost or filled in. Data bits shifted out of the low-order end of the A-register are rotated around to enter the high-order end of the B-register.

3-21. EXTENDED INSTRUCTION GROUP CODING

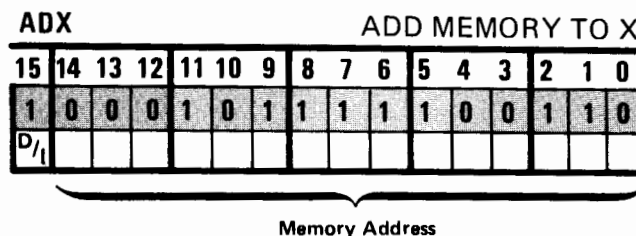
The extended instruction group includes index register instructions, bit and byte manipulation instructions, and move and compare instructions. The index registers are two hardware registers which are not accessible by the base set instructions described previously.

Instructions comprising the extended instruction group are one, two, or three words in length. The first word is always the instruction code; operand addresses are given in the words following the instruction code or in the A- and B-registers. The operand addresses are 15 bits long, with bit 15 (most-significant bit) indicating direct or indirect addressing. Bit 15 must be a zero for the JPY address since indirect addressing is not allowed.

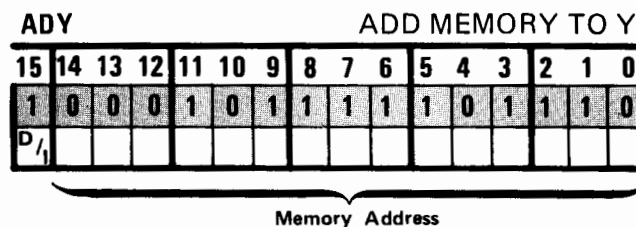
Operand addresses given in the A- and B-registers are 15 bits long for word addresses and 16 bits long for byte addresses. No indirect addressing is allowed. Bit 15 is ignored for word addresses.

3-22. INDEX REGISTER INSTRUCTIONS

The index registers are two of the 14 scratch pad registers normally accessible only by microprograms. The following instructions make these two 16-bit registers (X and Y) directly accessible by the software.



Adds the contents of the addressed memory location to the contents of the X-register. The sum remains in the X-register and the contents of the memory cell are unaltered. The result of this addition may set the extend bit or the overflow bit.



Adds the contents of the addressed memory location to the contents of the Y-register. The sum remains in the Y-register and the contents of the memory cell are unaltered. The result of this addition may set the extend bit or the overflow bit.

CAX								COPY A TO X							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	0	1	1	1	1	1	0	0	0	0	1

Copies the contents of the A-register into the X-register. The contents of the A-register are unaltered.

CAY								COPY A TO Y							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	0	1	1	1	1	1	0	1	0	0	1

Copies the contents of the A-register into the Y-register. The contents of the A-register are unaltered.

CBX								COPY B TO X							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	0	0	0	0	1

Copies the contents of the B-register into the X-register. The contents of the B-register are unaltered.

CBY								COPY B TO Y							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	0	1	0	0	1

Copies the contents of the B-register into the Y-register. The contents of the B-register are unaltered.

CXA								COPY X TO A							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	0	1	1	1	1	1	0	0	1	0	0

Copies the contents of the X-register into the A-register. The contents of the X-register are unaltered.

CXB								COPY X TO B							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	0	0	1	0	0

Copies the contents of the X-register into the B-register. The contents of the X-register are unaltered.

CYA								COPY Y TO A							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	0	1	1	1	1	1	0	1	1	0	0

Copies the contents of the Y-register into the A-register. The contents of the Y-register are unaltered.

CYB								COPY Y TO B							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	0	1	1	0	0

Copies the contents of the Y-register into the B-register. The contents of the Y-register are unaltered.

DSX								DECREMENT X AND SKIP IF ZERO							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	1	0	0	0	1

Subtracts one from the contents of the X-register. If the result of this operation is zero (X-register decremented from 000001 to 000000), the next instruction is skipped; i.e., the P-register count is advanced two counts instead of one count. If the result is not zero, the next sequential instruction is executed.

DSY								DECREMENT Y AND SKIP IF ZERO							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	1	1	0	0	1

Subtracts one from the contents of the Y-register. If the result of this operation is zero (Y-register decremented from 000001 to 000000), the next instruction is skipped; i.e., the P-register count is advanced two counts instead of one count. If the result is not zero, the next sequential instruction is executed.

ISX								INCREMENT X AND SKIP IF ZERO							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	1	1	0	0	0

Adds one to the contents of the X-register. If the result of this operation is zero (X-register rolls over to 000000 from 177777), the next instruction is skipped; i.e., the P-register count is advanced two counts instead of one count. If the result is not zero, the next sequential instruction is executed.

ISY								INCREMENT Y AND SKIP IF ZERO							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	1	1	0	0	0

Adds one to the contents of the Y-register. If the result of this operation is zero (Y-register rolls over to 000000 from 177777), the next instruction is skipped; i.e., the P-register count is advanced two counts instead of one count. If the result is not zero, the next sequential instruction is executed.

LAX LOAD A INDEXED BY X															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	0	1	1	1	1	1	0	0	0	1	0
D ₁₅															

Operand Address

Loads the A-register with the contents indicated by the effective address, which is computed by adding the contents of the X-register to the operand address. The effective address is loaded into the M-register; the X-register and memory contents are not altered. Indirect addressing is resolved before indexing; bit 15 of the effective address is ignored.

LAY LOAD A INDEXED BY Y															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	0	1	1	1	1	1	0	1	0	1	0
D ₁₅															

Operand Address

Loads the A-register with the contents indicated by the effective address, which is computed by adding the contents of the Y-register to the operand address. The effective address is loaded into the M-register; the Y-register and memory contents are not altered. Indirect addressing is resolved before indexing; bit 15 of the effective address is ignored.

LBX LOAD B INDEXED BY X															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	0	0	0	1	0
D ₁₅															

Operand Address

Loads the B-register with the contents indicated by the effective address, which is computed by adding the contents of the X-register to the operand address. The effective address is loaded into the M-register; the X-register and memory contents are not altered. Indirect addressing is resolved before indexing; bit 15 of the effective address is ignored.

LBY LOAD B INDEXED BY Y															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	0	1	0	1	0
D ₁₅															

Operand Address

Loads the B-register with the contents indicated by the effective address, which is computed by adding the contents of the Y-register to the operand address. The effective address is loaded into the M-register; the X-register and memory contents are not altered. Indirect addressing is resolved before indexing; bit 15 of the effective address is ignored.

LDX LOAD X FROM MEMORY															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	0	0	1	0	1
D ₁₅															

Memory Address

Loads the contents of the addressed memory location into the X-register. The A- and B-registers may be addressed as locations 00000 and 00001, respectively; however, if it is desired to load from the A- or B-register, copy instructions CAX or CBX should be used since they are more efficient.

LDY LOAD Y FROM MEMORY															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	0	1	1	0	1
D ₁₅															

Memory Address

Loads the contents of the addressed memory location into the Y-register. The A- and B-registers may be addressed as locations 00000 and 00001, respectively; however, if it is desired to load from the A- or B-register, copy instructions CAY or CBY should be used since they are more efficient.

SAX STORE A INDEXED BY X															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
D ₁₅															

Operand Address

Stores the contents of the A-register into the location indicated by the effective address, which is computed by adding the contents of the X-register to the operand address. The effective address is loaded into the M-register; the A- and X-register contents are not altered. Indirect addressing is resolved before indexing; bit 15 of the effective address is ignored.

SAY STORE A INDEXED BY Y															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	0	1	1	1	1	1	0	1	0	0	0
D ₁₅															

Operand Address

Stores the contents of the A-register into the location indicated by the effective address, which is computed by adding the contents of the Y-register to the operand address. The effective address is loaded into the M-register; the A- and Y-register contents are not altered. Indirect addressing is resolved before indexing; bit 15 of the effective address is ignored.

SBX STORE B INDEXED BY X															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	0	0	0	0	0
D ₁₅															

Operand Address

Stores the contents of the B-register into the location indicated by the effective address, which is computed by adding the contents of the X-register to the operand address. The effective address is loaded into the M-register; the B- and X-register contents are not altered. Indirect addressing is resolved before indexing; bit 15 of the effective address is ignored.

SBY STORE B INDEXED BY Y															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	0	1	0	0	0
D ₁₅															

Operand Address

Stores the contents of the B-register into the location indicated by the effective address, which is computed by adding the contents of the Y-register to the operand address. The effective address is loaded into the M-register; the B- and Y-register contents are not altered. Indirect addressing is resolved before indexing; bit 15 of the effective address is ignored.

STX STORE X TO MEMORY															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	0	0	0	1	1
D ₁₅															

Memory Address

Stores the contents of the X-register into the addressed memory location. The A- and B-registers may be addressed as locations 00000 and 00001, respectively. The X-register contents are not altered.

STY STORE Y TO MEMORY															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	0	1	0	1	1
D ₁₅															

Memory Address

Stores the contents of the Y-register into the addressed memory location. The A- and B-registers may be addressed as locations 00000 and 00001, respectively. The Y-register contents are not altered.

XAX EXCHANGE A AND X															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	0	1	1	1	1	1	0	0	1	1	1

Exchanges the contents of the A- and X-registers.

XAY EXCHANGE A AND Y															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	0	1	1	1	1	1	0	1	1	1	1

Exchanges the contents of the A- and Y-registers.

XBX EXCHANGE B AND X															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	0	0	1	1	1

Exchanges the contents of the B- and X-registers.

XBY EXCHANGE B AND Y															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	0	1	1	1	1

Exchanges the contents of the B- and Y-registers.

3-23. JUMP INSTRUCTIONS

The following two jump instructions involving the Y-register allow a program to either jump to or exit from a subroutine.

JLY JUMP AND LOAD Y															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	1	0	0	1	0
D ₁₅															

Memory Address

This instruction is designed for entering a subroutine. The instruction, executed in location P, causes computer

control to jump unconditionally to the memory location specified in the memory address. Indirect addressing may be specified. The contents of the P-register plus two (return address) is loaded into the Y-register. A return to the main program sequence at $P + 2$ may be effected by a JPY instruction (described next). A memory protect check is performed by this instruction. The effective address may not be below the fence, including the addressable A- and B-registers.

JPY JUMP INDEXED BY Y															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	1	1	0	1	0
0															

Operand Address

Transfers control to the effective address, which is computed by adding the contents of the Y-register to the operand address. Indirect addressing is not allowed. The effective address is loaded into the P-register; the Y-register contents are not altered. A memory protect check is performed by this instruction. The effective address may not be below the fence, including the addressable A- and B-registers.

3-24. BYTE MANIPULATION INSTRUCTIONS

A byte address is defined as two times the word address plus zero or one, depending on whether the byte is in the high-order position (bits 8 through 15) or low-order position (bits 0 through 7) of the word containing it. If the byte of interest is in bit positions 8 through 15 of memory location 100, for example, then the address of that byte is $2 * 100 + 0$, or 200; the address of the low-order byte in the same location is 201 ($2 * 100 + 1$). Because of the way byte addresses are defined, 16 bits are required to cover all possible byte addresses in a 32K-word memory configuration. Hence, for byte addressing, bit 15 does not indicate indirect addressing.

Byte addresses 000 through 003 reference bytes in the A- and B-registers. These addresses will not cause memory violations. The user should, however, be careful in referencing these byte addresses; for example, storing into byte address 002 or 003 would destroy the byte address originally contained in the B-register.

CBT COMPARE BYTES															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	1	0	1	1	0
D ₁₅															

Memory Address

Compares the bytes in string 1 with those in string 2. This is a three-word instruction where

Word 1 = Instruction code,

Word 2 = Address of word containing the string count, and

Word 3 = All-zeros word reserved for use by microcode.

The operand addresses are in the A- and B-registers. The A-register contains the first byte address of string 1 and the B-register contains the first byte address of string 2.

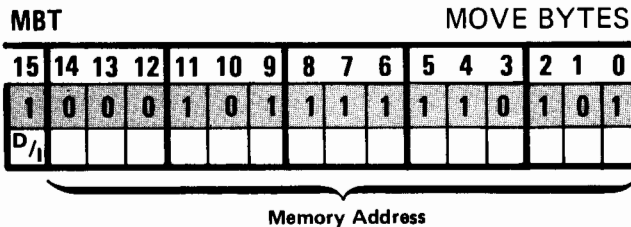
The number of bytes to be compared is given in the memory location addressed by Word 2 of the instruction. The strings are compared one byte at a time; the *i*th byte in string 1 is compared with the *i*th byte in string 2. The comparison is performed arithmetically; i.e., each byte is treated as a positive number. If all bytes in string 1 are identical with all bytes in string 2, the "equal" exit is taken. As soon as two bytes are compared and found to be different, the "less than" or "greater than" exit is taken, depending on whether the byte in string 1 is less than or greater than the byte in string 2. The three ways this instruction exits are as follows:

- No skip if string 1 is equal to string 2; the P-register advances one count from Word 3 of the instruction. The A-register contains its original value incremented by the count stored in the address specified in Word 2.
- Skips one word if string 1 is less than string 2; the P-register advances two counts from Word 3 of the instruction. The A-register contains the address of the byte in string 1 where the comparison stopped.
- Skips two words if string 1 is greater than string 2; the P-register advances three counts from Word 3 of the instruction. The A-register contains the address of the byte in string 1 where the comparison stopped.

For all three exits, the B-register will contain its original value incremented by the count stored in the address specified in Word 2. This instruction is interruptible. The interrupt routine is expected to save and restore the contents of the A- and B-registers. During the interrupt, the remaining count is stored in Word 3 of the instruction.

LBT LOAD BYTE															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	1	0	0	1	1

This one word instruction loads into the A-register the byte whose address is contained in the B-register. The byte is right-justified with leading zeros in the left byte. The B-register is incremented by one.



Moves bytes in a left-to-right manner; i.e., the byte having the lowest address from the source is moved first. This is a three word instruction where

Word 1 = Instruction code,

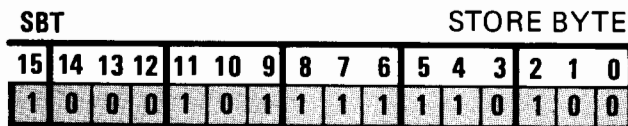
Word 2 = Address of word containing the byte count, and

Word 3 = All-zeros word reserved for use by microcode.

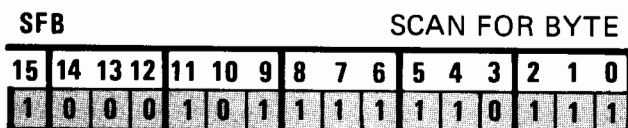
The operand addresses are in the A- and B-registers. The A-register contains the first byte address source and the B-register contains the first byte address destination.

The number of bytes to be moved is given by a 16-bit positive integer addressed by Word 2 of the instruction. The byte address in the A- and B-registers are incremented as each byte is being moved. Thus, at the end of the operation, the A- and B-registers are incremented by the number of bytes moved. Wraparound of the byte address would result from a carry out of bit position 15; therefore, if the destination became 000, 001, 002, or 003, the next byte would be moved into the A- or B-register and destroy the proper byte addresses for the move operation. For each byte move, a memory protect check is performed.

This instruction is interruptible. The interrupt routine is expected to save and restore the contents of the A- and B-registers. During the interrupt, the remaining count is stored in Word 3 of the instruction.



Stores the A-register low-order (right) byte in the byte address contained in the B-register. The B-register is incremented by one. A memory protect check is performed before the byte is stored. The left byte in the A-register does not have to be zeros. The other byte in the same word of the stored byte is not altered.



This is a one word instruction with the operands in the A- and B-registers. The A-register contains a termination

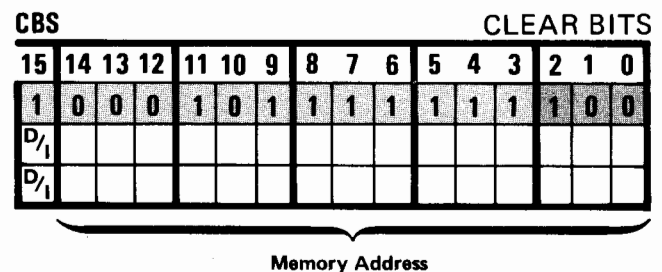
byte (high-order byte) and a test byte (low-order byte). The B-register contains the first byte address of the string to be scanned.

A string of bytes is scanned starting at the byte address given in the B-register. Scanning terminates when a byte in the string matches either the test byte or the termination byte in the A-register. The manner in which the instruction exits depends on which byte is matched first. If a byte in the string matches the test byte, the instruction will not skip upon exit; the B-register will contain the address of the byte matching the test byte. If a byte in the string matches the termination byte, the instruction will skip one word upon exit; the B-register will contain the address of the byte matching the termination byte *plus one*.

The scanning operation will not continue indefinitely even if neither the termination byte nor test byte exists in memory. These bytes are in the A-register with byte addresses 000 and 001, respectively. Thus, if no match is made by the time the B-register points to the last byte in memory, the B-register will roll over to zero and the next test will match the termination byte in the A-register with itself.

3-25. BIT MANIPULATION INSTRUCTIONS

The following three instructions allow any number of bits in a specified memory location to be cleared, set, or tested.



Clears bits in the addressed location. This is a three-word instruction where

Word 1 = Instruction code,

Word 2 = Address of a 16-bit mask, and

Word 3 = Address of word where bits are to be cleared.

The bits to be cleared correspond to logic 1's in the mask. The bits corresponding to logic 0's in the mask are not affected. A memory protect check is performed prior to modifying the word in memory.

Memory Address

Word 1 = Instruction code,

Word 2 = Address of a 16-bit mask, and

Word 3 = Address of word where bits are to be set.

TEST BITS															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	1	1	1	1	1	1	1	0	1
D ₁₅															
D ₁₄															

Memory Address

Word 1 = Instruction code,

Word 2 = Address of a 16-bit mask, and

Word 3 = Address of word in which bits are to be tested.

3-26. WORD MANIPULATION INSTRUCTIONS

3-22

Memory Address

Word 1 = Instruction code,

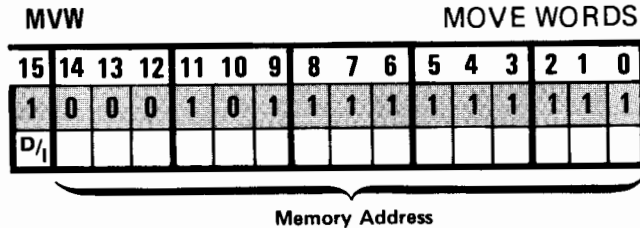
Word 2 = Address of word containing the word
count, and

Word 3 = All-zeros word reserved for use by microcode.

The number of words to be compared is given in the memory location addressed by Word 2 of the instruction. The arrays are compared one word at a time; the i th word in array 1 is compared with the i th word in array 2. This comparison is performed arithmetically; i.e., each word is considered a two's complement number. If all words in array 1 are equal to all words in array 2, the "equal" exit is taken. As soon as two words are compared and found to be different, the "less than" or "greater than" exit is taken, depending on whether the word in array 1 is less than or greater than the word in array 2. The three ways this instruction exits are as follows:

- a. No skip if array 1 is equal to array 2; the P-register advances one count from Word 3 of the instruction. The A-register contains its original value incremented by the word count stored in the address specified in Word 2.
- b. Skips one word if array 1 is less than array 2; the P-register advances two counts from Word 3 of the instruction. The A-register contains the address of the word in array 1 where the comparison stopped.
- c. Skips two words if array 1 is greater than array 2; the P-register advances three counts from Word 3 of the instruction. The A-register contains the address of the word in array 1 where the comparison stopped.

For all three exits, the B-register will contain its original value incremented by the word count stored in the address specified in Word 2. This instruction is interruptible. The interrupt routine is expected to save and restore the contents of the A- and B-registers. During the interrupt, the remaining count is stored in the address specified in Word 3 of the instruction.



Moves words in a left-to-right manner; i.e., the word having the lowest address in the source is moved first. This is a three-word instruction where

Word 1 = Instruction code,

Word 2 = Address of word containing the count, and

Word 3 = All-zeros word reserved for use by microcode.

The operand addresses are in the A- and B-registers. The A-register contains the first word address source and the B-register contains the first word address destination. The number of words to be moved is a 16-bit positive integer addressed by Word 2 of the instruction. The word addresses in the A- and B-registers are incremented as each word is being moved. Thus, at the end of the operation, the A- and B-registers are incremented by the number of words moved.

Wraparound of the word address would result from a carry into bit position 15 (i.e., at 32767). If the destination address became 000 or 001, the next word would be moved into the A- or B-register and destroy the proper word addresses for the move operation. For each word move, a memory protect check is performed.

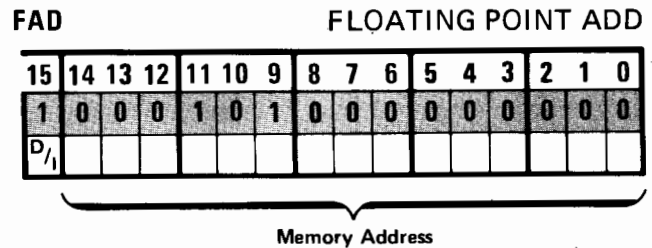
This instruction is interruptible. The interrupt routine is expected to save and restore the contents of the A- and B-registers. During the interrupt, the remaining count is stored in Word 3 of the instruction.

3-27. FLOATING POINT INSTRUCTION CODING

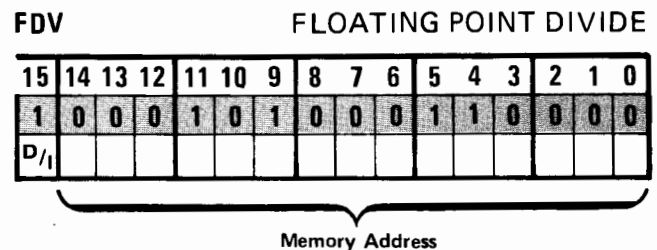
The following six floating point instructions make it possible to add, subtract, multiply, and divide floating point numbers and to convert quantities from floating point format to integer format or vice versa.

Each of the four arithmetic instructions requires two words of memory: one for the instruction code and one for the operand address. Since a full 15 bits are available for the operand address, these instructions can directly address any location in memory. As with all memory reference instructions, indirect addressing to any number of levels is permitted. A logic 0 in bit position 15 specifies direct addressing; a logic 1 specifies indirect addressing.

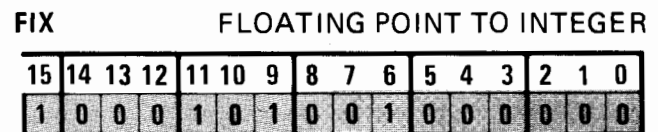
The execution times of the floating point instructions are specified under paragraph 3-28. These instructions are noninterruptible; any attempted interrupt is held off for the full execution time of the currently active floating point instruction. However, data transfer via the dual-channel port controller (DCPC) is not held off.



Adds the floating point quantity in the A- and B-registers to the floating point quantity in the specified memory locations. The floating point result is returned in the A- and B-registers. Overflow occurs if the result lies outside the range $(1-2^{-23}) * -2^{127}$ through $(1-2^{-23}) * 2^{127}$. In such a case, the overflow bit is set and the result $(1-2^{-23}) * 2^{-129}$ is returned to the A- and B-registers. Underflow occurs if the result lies within the range $(1+2^{-22}) * -2^{129}$ through $(1+2^{-22}) * 2^{-129}$. In such a case, the overflow bit is set and the result 0 is returned to the A- and B-registers.



Divides the floating point quantity in the A- and B-registers by the floating point quantity in the specified memory locations. The floating point is returned to the A- and B-registers. Overflow and underflow are as described for the FAD instruction.



Converts the floating point quantity in the A- and B-registers to integer format. The integer result is returned to the A-register. If the magnitude of the floating point number is <1 , the integer 0 is returned. If the magnitude of the exponent of the floating point number is $\geq 2^{16}$, the integer 32767 (077777 octal) is returned and the overflow bit is set.

FLT INTEGER TO FLOATING POINT

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	0	0	1	0	1	0	0	0	0

Converts the integer quantity in the A-register to floating point format. The floating point result is returned to the A- and B-registers.

FMP FLOATING POINT MULTIPLY

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0
D ₁															

Memory Address

Multiplies the floating point quantity in the A- and B-registers by the floating point quantity in the specified memory locations. The floating point result is returned to the A- and B-registers. Overflow and underflow are as described for the FAD instruction.

FSB FLOATING POINT SUBTRACT

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0
D ₁															

Memory Address

Subtracts the floating point quantity in the specified memory locations from the floating point quantity in the A- and B-registers. The floating point result is returned to the A- and B-registers. Overflow and underflow are as described for the FAD instruction.

3-28. INSTRUCTION EXECUTION TIMES

Table 3-5 lists the execution times required for the various instructions. Since the timing requirements of the input/output group instructions depend on the time period in which the instruction begins, programs should not rely on the execution times for accurate, real-time measurements.

There are no phases in the HP 2105A and HP 2108A Processors in the true sense of the word. Phases such as execute, indirect, interrupt, and fetch are inherent in the state of the processor, which can be defined as the contents of the read-only memory (ROM) address register.

3-29. INTERRUPT SYSTEM

The vectored priority interrupt system has up to 60 distinct interrupt levels, each of which has a unique priority assignment. Each interrupt level is associated with a numerically corresponding interrupt location in memory.

Of the 60 interrupt levels, the two highest priority levels are reserved for hardware faults (power fail and parity error), the next two are reserved for dual-channel port controller completion interrupts, and the remaining levels are available for I/O device channels. Tables 3-6 and 3-7 list the interrupt levels in priority order for the HP 2105A and HP 2108A Processors, respectively.

Table 3-6. HP 2105A Interrupt Assignments

CHANNEL (Octal)	INTERRUPT LOCATION	ASSIGNMENT
04	00004	Power Fail Interrupt
05	00005	Memory Parity/Protect Interrupt
06	00006	DCPC Channel 1 Completion Interrupt
07	00007	DCPC Channel 2 Completion Interrupt
10	00010	I/O Device (highest priority)
11 - 13	00011-00013	I/O Device (Mainframe)
14 - 35	00014-00035	I/O Device (Extender No. 1)
36 - 57	00036-00057	I/O Device (Extender No. 2)

Table 3-7. HP 2108A Interrupt Assignments

CHANNEL (Octal)	INTERRUPT LOCATION	ASSIGNMENT
04	00004	Power Fail Interrupt
05	00005	Memory Parity/Protect Interrupt
06	00006	DCPC Channel 1 Completion Interrupt
07	00007	DCPC Channel 2 Completion Interrupt
10	00010	I/O Device (highest priority)
11 - 20	00011-00020	I/O Device (Mainframe)
21 - 42	00021-00042	I/O Device (Extender No. 1)
43 - 64	00043-00064	I/O Device (Extender No. 2)

Table 3-5. Instruction Execution Times

INSTRUCTION	EXECUTION TIME (μ S)	INSTRUCTION	EXECUTION TIME (μ S)
Memory Reference Group^{1,2}		Extended Instruction Group	
ADA/B, AND, IOR, LDA/B, STA/B, XOR	1.94	CAX, CBX, CAY, CBY CXA, CXB, CYA, CYB	2.275
CPA/B (no skip) (skip)	2.27 2.59	XAX, XBX, XAY, XBY ISX, ISY, DSX, DSY	3.250
ISZ (no skip) (skip)	2.59 2.92	LDX, LDY (direct address) (indirect address)	4.875 4.875 ^a
JMP	1.94	STX, STY (direct address) (indirect address)	5.20 5.20 ^a
JSB	2.27	LAX, LBX, LAY, LBY (direct address) (indirect address)	4.875 5.525 ^a
Shift/Rotate Group³	2.6 - 2.92	SAX, SBX, SAY, SBY (direct address) (indirect address)	5.20 5.85 ^a
Alter/Skip Group³		ADX, ADY (direct address) (indirect address)	4.875 4.875 ^a
No skip, no increment	2.59	JLY (direct address) (indirect address)	5.525 5.525 ^a
No skip, increment A/B	2.92	JPY	4.55
Skip, no increment	2.59	LBT	4.875 avg
Skip, increment A/B	2.92	SBT	6.01 avg
Input/Output Group⁴	2.59 - 3.89	MBT	8.775 ^{a,b,g}
Extended Arithmetic Group⁵		MVW	7.8 ^{a,c,g}
ASL, ASR, LSL, LSR, RRL, RRR	3.57 - 8.43	CBT	8.775 ^{a,d,g}
DLD	4.54	CMW	7.8 ^{a,e,g}
DST	4.86	SFB (for test byte match) (for term. byte match)	3.575 ^{f,g} 2.275 ^{f,g}
MPY	12.32 - 13.30	CBS, SBS	7.8 ^a
DIV	15.92 - 18.20	TBS	8.125 ^a
Floating Point Group			
FAD	21.78 - 53.95		
FDV	41.2 - 75.72		
FIX	6.50 - 12.02		
FLT	10.72 - 34.42		
FMP	48.10 - 56.88		
FSB	22.75 - 57.20		
¹ Memory refresh consumes 0.65 μ S maximum no more often than every 32.5 μ S. ² Add 1.3 μ S for each indirect address level. ³ NOP or RSS requires 2.92 μ S whereas a JMP *+1 or JMP *+2 requires only 1.94 μ S. ⁴ Depends on which I/O time period (T2, 3, 4, 5, 6) the instruction begins. ⁵ Depends on number of shifts specified (1 to 16).		^a Add 1.3 μ S for each level of indirect addressing. ^b Add 7.31 μ S for each byte moved or compared. ^c Add 3.25 μ S for each word moved or compared. ^d Add 8.125 μ S for each byte moved or compared. ^e Add 3.575 μ S for each word moved or compared. ^f Add 4.875 μ S for each byte moved or compared. ^g Add 7.15 μ S for each interrupt of the instruction.	

As an example of the simplicity of the interrupt system, an interrupt request from I/O channel 12 will cause an interrupt to memory location 00012. This request for service will be granted on a priority basis higher than afforded to channel 13 but lower than that afforded to channel 11. Thus, a transfer in progress via channel 13 would be suspended to allow channel 12 to proceed. On the other hand, a transfer in progress via channel 11 cannot be interrupted by channel 12.

Any device can be selectively enabled or disabled under program control, thus switching the device into or out of the interrupt structure. In addition, the entire interrupt system, except power fail and parity error interrupts, can be enabled or disabled under program control using a single instruction.

Interrupt requests received while the computer is in the halt mode will be processed, in order of priority, when the computer is placed in the run mode. Input/output priority is covered in more detail in section IV.

3-30. POWER FAIL INTERRUPT

The computer is equipped with power-sensing circuits. When primary (mains) power fails or drops below a predetermined operating level while the computer is running, an interrupt to memory location 00004 is automatically generated. This interrupt is given the highest priority in the system and cannot be turned off or otherwise disabled. Memory location 00004 is intended to contain a jump-to-subroutine (JSB) instruction referencing the entry point of a power fail subroutine; however, location 00004 may alternatively contain a halt (HLT) instruction. The interrupt capability of lower-priority operations is automatically inhibited while a power fail subroutine is in process.

A minimum of 500 microseconds is available between the detection of a power failure and the loss of usable power supply power to execute a power fail subroutine; the purpose of such a subroutine is to transfer the current state of the computer system into memory and then halt the computer. A sample power fail subroutine is given in table 3-8. The optional battery will supply enough power to preserve the contents of memory for a sustained power mains outage of up to 2 hours.

Since the computer might be unattended by an operator, the user has a switch-selectable option of what action the computer will take upon the restoration of primary power. When the switch (A1S2) is set to the ARS position, the computer will halt when power is restored regardless of whether the computer was running or halted when the failure occurred. (No operator panel indication is given.)

Note: Switch A1S2 is mounted on the CPU and is not considered an operator control. The setting of this switch is normally determined prior to or during system installation.

When A1S2 is in the ARS position, the automatic restart feature is enabled. After a built-in delay of about half a second following the return to normal power levels, another interrupt to location 00004 occurs. This time the power-down portion of the subroutine is skipped and the power-up portion begins. (Refer to table 3-8.) If the computer was not running when the power failure occurred, the computer is halted immediately. If the computer was running, those conditions existing at the time of the power fail interrupt are restored and the computer continues the program from the point of the interruption. Alternatively, if location 00004 contains a HLT instruction instead of a JSB instruction, the computer will halt and light the POWER FAIL/BATTERY indicator.

To allow for the possibility of a second power failure occurring while the power-up portion of the subroutine is in process, the user should limit the combined power-down and power-up instructions to less than 100. If the computer memory does not contain a subroutine to service the interrupt, location 00004 should contain a HLT 04 instruction (102004 octal).

A Set Control instruction (STC 04) must be given at the end of any restart routine. This instruction re-initializes the power-fail logic and restores the interrupt capability to the lower priority functions. Pressing the PRESET switch on the operator panel performs the same function as the STC 04 instruction.

The optional battery sustains the contents of memory when mains power is off. If the battery becomes discharged when mains power is off, the operator must turn the operator panel key switch to the reset (R) position before the computer will operate with mains power restored. This will also clear the entire memory to zeros in order to restore correct parity (only if memory is lost).

3-31. PARITY ERROR INTERRUPT

Parity checking of memory is a standard feature in the computer. The parity logic continuously generates correct parity for all words written into memory and monitors the parity of all words read out of memory. Correct parity is defined as having the total number of "1" bits in a 17-bit memory word (16 data bits plus the parity bit) equal to an odd value. If a "1" bit (or any odd number of "1" bits) is either dropped or added in the transfer process, a Parity Error signal is generated when that word is read out of memory.

The Parity Error signal may either halt the computer or cause the computer to take some other action as determined by an internal switch (A1S1) mounted on the CPU. When the switch is in the HALT PE position and a parity error occurs, the computer will halt and light the PARITY indicator. The PARITY indicator will remain lighted until the PRESET switch is pressed.

Table 3-8. Sample Power Fail Subroutine

LABEL	OPCODE	OPERAND	COMMENTS
PFAR	NOP		Power Fail/Auto Restart Subroutine
	SFC	4B	Skip if interrupt was caused by a power failure
	JMP	UP	Power is being restored, reset state of computer system
DOWN	STA	SAVA	Save A-register contents
	CCA		Set switch indicating that the computer was running
	STA	SAVR	when power failed
	STB	SAVB	Save B-register contents
	ERA,ALS		Transfer E-register content to A-register bit 15
	SOC		Increment A-register if Overflow
	INA		is set
	STA	SAVEO	Save E- and O-register contents
	LDA	PFAR	Save contents of P-register at time of
	STA	SAVP	power failure
	LIA	1B	Save contents of
	STA	SAVS	S-register
	STX	SAVX	Save contents of X-register
	STY	SAVY	Save contents of Y-register
	:		Insert user-written routine to save I/O
			device states
	CLC	4B	Turn on restart logic so computer will restart when power is restored
			after momentary power failure
	HLT		Shutdown
UP	LDA	SAVR	Was computer running
	SZA,RSS		when power failed?
	JMP	HALT	No
	CLA		Yes, reset computer Run switch to
	STA	SAVR	initial state
	LDA	FENCE	Restore the memory protect
	OTA	5B	fence register contents
	:		Insert user-written routine to restore
			I/O device states
	LDA	SAVEO	Restore the contents
	CLO		of the
	SLA,ELA		E-register and
	STF	1B	O-register
	LDA	SAVS	Restore the contents of the
	OTA	1B	S-register
	LDA	SAVA	Restore A-register contents
	LDB	SAVB	Restore B-register contents
	LDX	SAVX	Restore X-register contents
	LDY	SAVY	Restore Y-register contents
	STC	4B	Reset power fail logic for next power failure
	STC	5B	Turn on memory protect
	JMP	SAVP,I	Transfer control to program in execution at time of power failure
HALT	HLT		Return computer to halt mode
FENCE	OCT	2000B	Fence address storage (must be updated each time fence is
			changed)
SAVEO	OCT	0	Storage for E and O
SAVA	OCT	0	Storage for A
SAVB	OCT	0	Storage for B
SAVS	OCT	0	Storage for S
SAVX	OCT	0	Storage for X
SAVY	OCT	0	Storage for Y
SAVP	OCT	0	Storage for P
SAVR	OCT	0	Storage for Run switch

Note: Switch A1S1 is mounted on the CPU and is not considered an operator control. The setting of this switch is normally determined prior to or during installation or when the memory protect PCA is installed at the user's site.

If switch A1S1 is in the INT/IGNORE position, the action that the computer will take when a parity error occurs is as follows:

- a. If the memory protect PCA is installed and the parity error logic has not been disabled by a CLF 05 instruction, an interrupt to memory location 00005 is generated. This location may alternatively contain a JSB instruction referencing the entry point of a user-written memory protect subroutine, or contain a HLT instruction.
- b. If the memory protect PCA is not installed, or if the memory protect option is installed but the parity error logic has been disabled by a CLF 05 instruction, the parity error will be ignored and the PARITY indicator will light.

Note: Memory protect is an option available only with the HP 2108A Processor. The memory protect PCA is dedicated to memory slot 111.

In conjunction with memory protect, it is possible to determine the memory address containing the parity error. The error address will be loaded automatically into the violation register of the memory protect logic and from there it is accessible to the user by programming an LIA 05 or LIB 05 instruction.

When a parity error occurs, it is recommended that the entire program or set of data containing the error location be reloaded. However, by knowing the address and the contents of the error location, the user may be able to determine what operations have taken place as a result of reading the erroneous word. For example, if the erroneous word was an instruction, several other locations may be affected. By individually checking and correcting the contents of all affected memory locations, the user may resume running the program without the necessity of a complete reload. If software is being generated, this may also need correcting.

3-32. MEMORY PROTECT INTERRUPT

The memory protect option (HP 2108A only) provides the capability of protecting a selected block of memory of any size, from a settable fence address downward, against

alteration by programmed instructions except those directly involving the A- and B-registers. Any programmed instruction except JMP may freely address the A- and B-registers as locations 00000 and 00001 (octal), respectively.

The memory protect logic, when enabled by an STC 05 instruction, also prohibits the execution of all I/O instructions (including HLT 01) except those referencing I/O select code 01, the S-register and the overflow register. This feature limits the control of I/O operations to interrupt control only. Thus, by programming the system to direct all I/O interrupts to an executive program residing in protected memory, the executive program can have exclusive control of the I/O system.

The memory protect logic is disabled automatically by any interrupt (except when the interrupt location contains an I/O instruction) and must be re-enabled by an STC 05 instruction at the end of each interrupt subroutine.

Programming rules pertaining to the use of memory protect are as follows (assuming that an STC 05 instruction has been given):

- a. Location 00002 is the lower boundary of protected memory. (Locations 00000 and 00001 are the A- and B-register addresses.)
- b. JMP instructions may not reference the A- or B-register; however, a JSB instruction may do so.
- c. The upper boundary (memory address) is loaded into the fence register from the A- or B-register by an OTA 05 or OTB 05 instruction, respectively. Memory locations below but not including this address are protected.
- d. Execution will be inhibited and an interrupt to location 00005 will occur if a JMP, JSB, ISZ, STA, STB, or DST instruction* either directly or indirectly addresses a location in protected memory, or if any I/O instruction is attempted (including HLT but excluding those addressing select code 01, the S-register and the overflow register). After three successive levels of indirect addressing, the memory Protect logic will allow a pending I/O interrupt.
- e. Any instruction not mentioned in step d of this paragraph is legal even if the instruction directly references a protected memory address. In addition, indirect addressing through protected memory by those instructions listed in step d is legal provided that the ultimate effective address is outside the protected memory area.

*Also CBT, JLY, JPY, MVB, MVW, SAX, SAY, SBX, SBY, STX, and STY of the extended instruction group.

Following a memory protect interrupt, the address of the illegal instruction will be present in the violation register. This address is made accessible to the programmer by an LIA 05 or LIB 05 instruction, which loads the address into the A- or B-register.

Since parity error and memory protect share the same interrupt location, it is necessary to distinguish which type of error is responsible for the interrupt. A parity error is indicated if, after the LIA (or LIB) 05 instruction is executed, bit 15 of the selected register is a logic 1; a

memory protect violation is indicated if bit 15 is a logic 0. In either case, the remaining 15 bits of the selected register contains the octal address of the error location.

Table 3-9 illustrates a sample memory protect and parity error subroutine. An assumption made for this example is that the location immediately following the error location is an appropriate return point. This may not always be the case, however, because it may be deemed advisable to abort the program in process and return to a supervisory program.

Table 3-9. Sample Memory Protect/Parity Error Subroutine

LABEL	OPCODE	OPERAND	COMMENTS
MPPE	NOP		Memory Protect/Parity Error Subroutine
	CLF	0B	Turn off interrupt system to inhibit I/O devices
	CLF	5B	Turn off PE interrupt during subroutine
	STA	SVA	Save A-register contents
	STB	SVB	Save B-register contents
	LIA	5B	Get contents of violation register in MP logic
	SSA		Check bit 15 to determine kind of error
	JMP	PERR	If a 1, go to parity error routine
	JMP	MPTR	If a 0, go to memory protect routine
			User's routine in case of memory protect violation
MPTR	—		
	—		
	—		
	etc.		
	—		
	—		
	—		
	LDA	SVA	Restore A-register
	LDB	SVB	Restore B-register
	STF	0B	Enable interrupt system
PERR	STF	5B	Turn on parity error interrupt
	STC	5B	Turn on memory protect interrupt
	JMP	MPPE,I	Exit the subroutine
	—		User's routine in case of parity error
	—		
	—		
	etc.		
	—		
	—		
	JMP	PERR-6B	Restore accumulators, turn on interrupts, exit

3-33. DUAL-CHANNEL PORT CONTROLLER INTERRUPT

The optional dual-channel port controller (DCPC) allows high-speed block transfer of data between input/output devices and memory. For the most part, the DCPC operates independently of the interrupt system in that the only time that a DCPC interrupt occurs is when the specified block of data has been transferred. Since there are two DCPC channels, two interrupt locations are reserved for this purpose; location 00006 is reserved for channel 1 and location 00007 is reserved for channel 2. Channel 1 interrupt has priority over the channel 2 interrupt. Because DCPC interrupts are primarily completion signals to the programmer, and are therefore application dependent, no interrupt subroutine example is considered necessary.

3-34. INPUT/OUTPUT INTERRUPT

The remaining interrupt locations (00010 through 00077 octal) are reserved for I/O devices; this represents a total of 56 (decimal) locations, one for each I/O channel. In a typical I/O operation, the computer issues a programmed command such as Set Control/Clear Flag (STC,C) to one or more external devices to initiate an input (read) or an output (write) operation. Each device will then either put data into or accept data from an input/output buffer on its associated interface PCA. During this time, the computer may continue running a program or may be programmed into a waiting loop to wait for a specific device to complete a read or write operation. Upon the completion of a read or write operation, each device returns a Flag signal to the computer. These Flag signals are passed through a priority network which allows only one device to be serviced regardless of the number of Flag signals present at that time. The Flag signal with the highest priority generates an Interrupt signal at the end of the current machine cycle except under the following circumstances:

- a. Interrupt system disabled or interface PCA interrupt disabled.
- b. JMP indirect or JSB indirect instruction not sufficiently executed. These instructions inhibit all interrupts except power fail or memory protect until the succeeding instruction is executed.
- c. Instruction in an interrupt location not sufficiently executed, even if that interrupt is of lower priority. Any interrupt inhibits the entire interrupt system until the succeeding instruction is executed.

- d. Optional dual-channel port controller in the process of transferring data.
- e. Current instruction is one that may affect the priorities of I/O devices; e.g., Set Control (STC), Clear Control (CLC), Set Flag (STF), and Clear Flag (CLF). The interrupt in this case must wait until the succeeding instruction is executed.

After an interface PCA has been issued a Set Control command and its Flag flip-flop becomes set, all interrupt requests from lower-priority devices are inhibited until this Flag flip-flop is cleared by a Clear Flag (CLF) instruction. A service subroutine in process for any device can be interrupted only by a higher-priority device; then, after the higher-priority device is serviced, the interrupted service subroutine may continue. In this way it is possible for several service subroutines to be in the interrupt state at one time; each of these service subroutines will be allowed to continue after the higher-priority device is serviced. All such service subroutines normally end with a JMP indirect instruction to return the computer to the point of the interrupt.

3-35. CENTRAL INTERRUPT REGISTER

Each time an interrupt occurs, the address of the interrupt location is stored in the central interrupt register. The contents of this register are accessible at any time by executing an LIA 04 or LIB 04 instruction. This loads the address of the most recent interrupt into the A- or B-register.

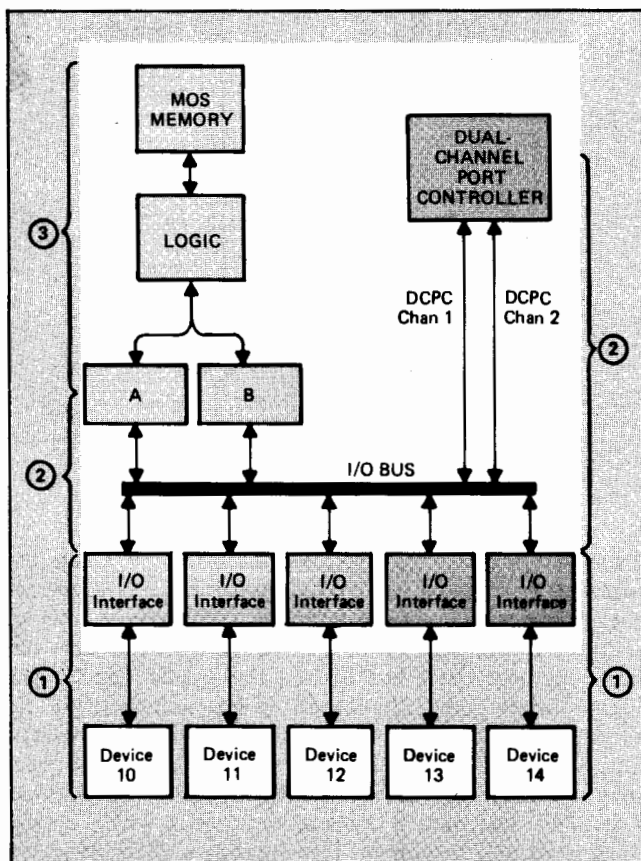
3-36. INTERRUPT SYSTEM CONTROL

I/O address 00 is the master control address for the interrupt system. An STF 00 instruction enables the entire interrupt system and a CLF 00 disables the interrupt system. The two exceptions to this are the power fail interrupt, which cannot be disabled, and parity error interrupt, which can only be selective enabled or disabled by an STF 05 or CLF 05, respectively.

Whenever power is initially applied, a clear signal to I/O address 00 automatically disables the interrupt system. Programs dependent on the interrupt mode of operation must include an STF 00 instruction to ensure that the interrupt system is enabled in the run mode.

The purpose of the input/output system is to transfer data between the computer and external devices. As shown in figure 4-1, data is normally transferred through the A- or B-register. An input transfer of this type occurs in three distinct steps: (1) between the external device and its interface PCA in the computer, (2) between the interface PCA and the A- or B-register, and (3) between the A- or B-register and memory. This three-step process also applies to an output transfer except in reverse order. This type of transfer, which is executed under program control, allows the computer logic to manipulate the data during the transfer process.

Also shown in figure 4-1, data may be transferred automatically under control of the dual-channel port controller (DCPC) option. Once the DCPC has been initialized, no programming is involved and the transfer is reduced to a two-step process: (1) the transfer between the device and its interface PCA and (2) the transfer between the interface PCA and memory. The two DCPC channels are assignable to operate with any two device interface PCA's.



2270-30

Figure 4-1. Input/Output System

Since a DCPC transfer eliminates programmed loading and storing via the accumulators, the time involved is very short. Thus, the DCPC is used with high-speed devices capable of transferring data at rates up to 616,666 sixteen-bit words per second. Further information on the DCPC option is given under paragraph 4-13.



4-1. INPUT/OUTPUT ADDRESSING

As shown in figure 4-2, an external device is connected by cable directly to an interface PCA located inside the computer mainframe. The interface PCA, in turn, plugs into one of the input/output slots, each of which is assigned a fixed address commonly referred to as the device select code. The computer can then communicate with a specific device on the basis of its select code.

Figure 4-2 shows an interface PCA inserted in the I/O slot having the highest priority; this channel is assigned select code 10 (octal). If it is decided that the associated device should have lower priority, its interface PCA and cable may simply be exchanged with those occupying some other I/O slot. This will change both the priority and the I/O address; however, due to priority chaining (refer to paragraph 4-2), there can be no vacant slots from select code 10 to the highest used select code (if the interrupt mode is to be used).

Only select codes 10 through 77 (octal) are available for input/output devices; the lower select codes (00 through 07) are reserved for other features. Figure 4-2 illustrates the I/O select codes available in the HP 2105A and HP 2108A Processor mainframes.

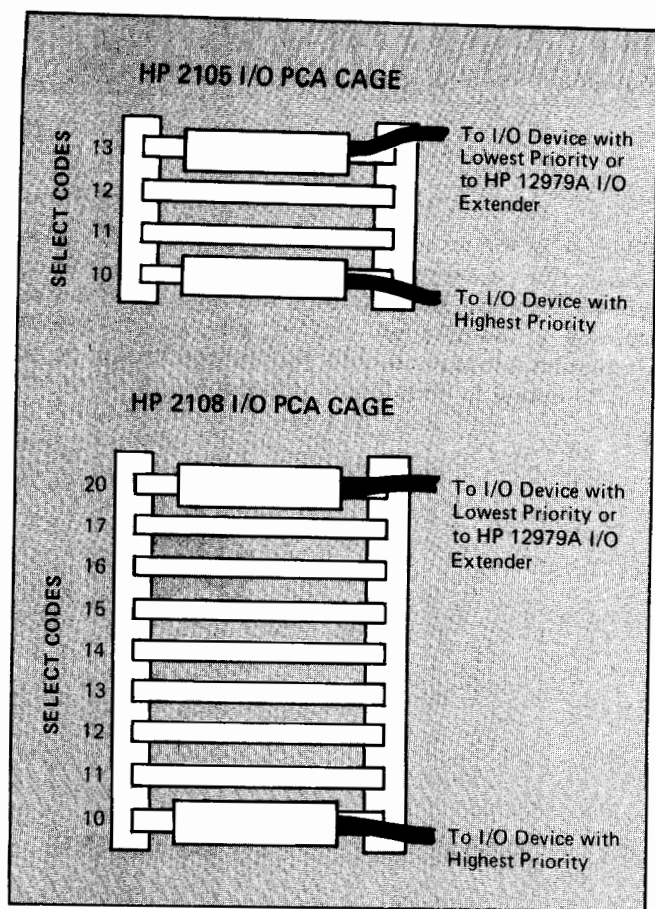


Figure 4-2. I/O Address Assignments

If an I/O extender is used with the HP 2105A, channel 13 is used for interconnection of the extender and select codes 14 through 35 will be available in the extender; a second extender may be used and have the availability of select codes 36 through 57.

If an I/O extender is used with the HP 2108A, channel 20 is used for interconnection of the extender and select codes 21 through 42 will be available in the extender; a second extender may be used and have the availability of select codes 43 through 64.

4-2. INPUT/OUTPUT PRIORITY

When a device is ready to be serviced, it causes its interface PCA to request an interrupt so that the computer will interrupt the current program and service the device. Since many device interface PCA's will be requesting service at random times, it is necessary to establish an orderly sequence for granting interrupts. Secondly, it is desirable that high-speed devices should not have to wait for low-speed device transfers. Both of these requirements are met by a series-linked priority structure illustrated by figure 4-3. The bold line, representing a priority enabling signal, is routed in series through each PCA capable of causing an interrupt. The PCA cannot interrupt unless this enabling signal is present at its input.

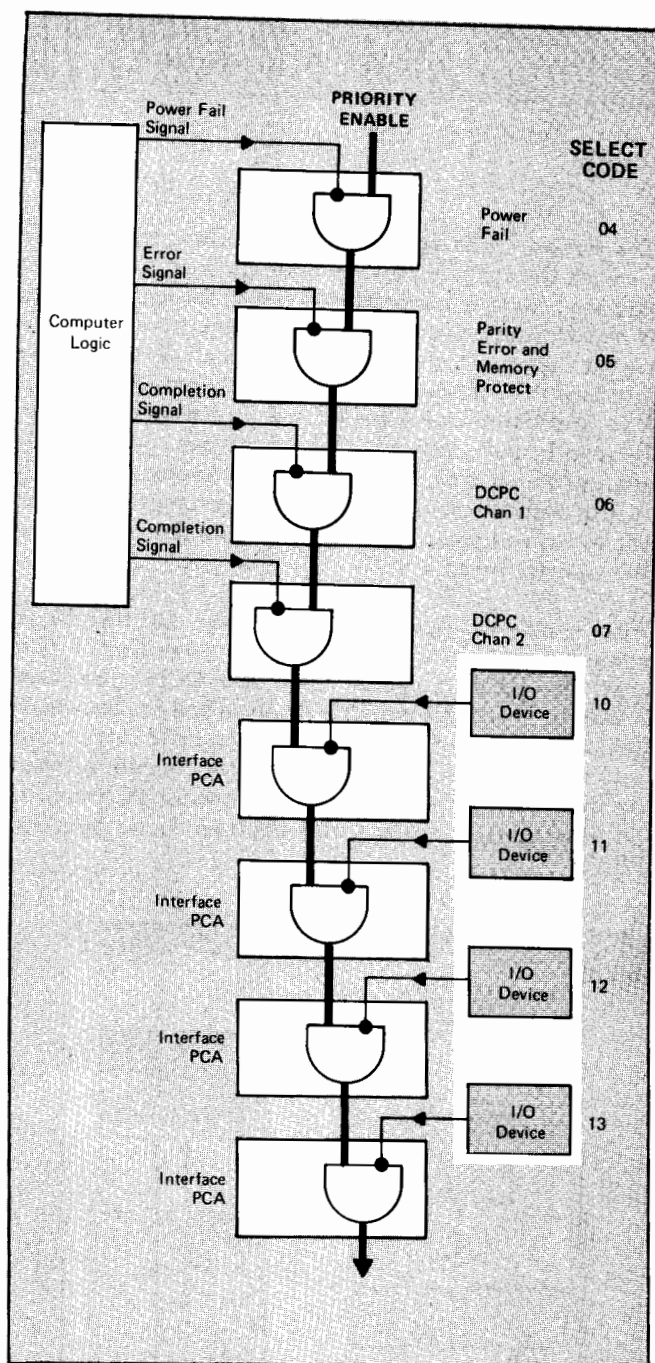
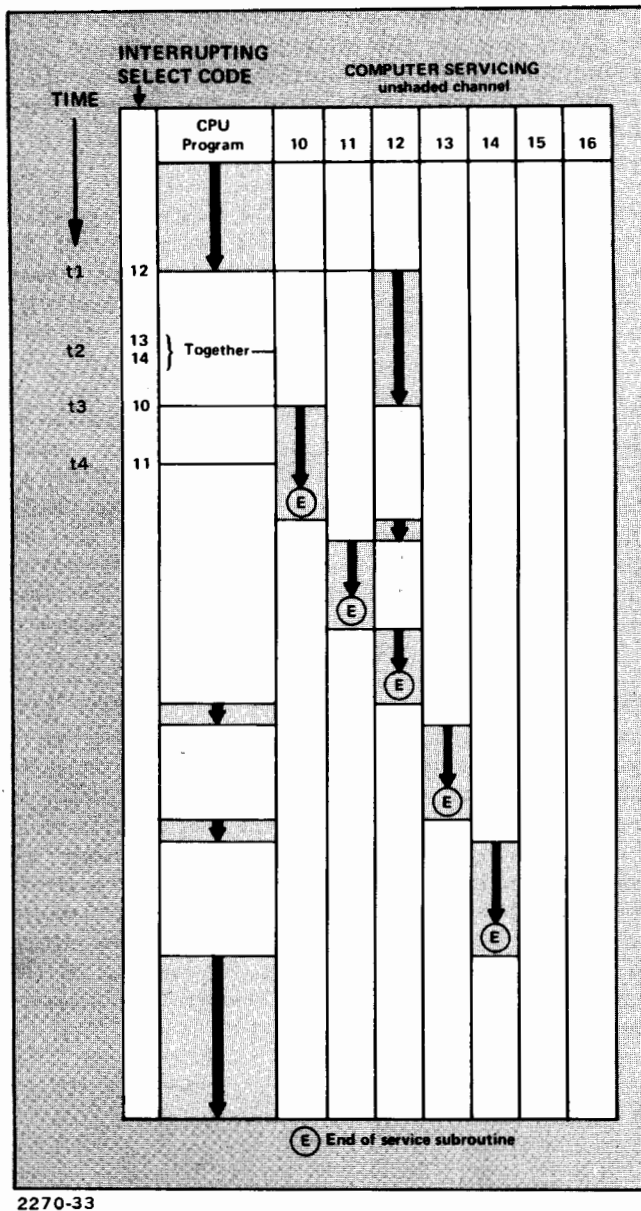


Figure 4-3. Priority Linkage

Each device (or other interrupt function) can break the enabling line when it requests an interrupt. If two devices simultaneously request an interrupt, obviously the device with the lowest select code will be the first one that can interrupt because it has broken the enable line for the higher select code. The other device cannot begin its service routine until the first device is finished; however, a still higher priority device (one with a lower select code) may interrupt the service routine of the first device. Figure 4-4 illustrates a hypothetical case in which several devices require service by interrupting a CPU program. Both simultaneous and time-separated interrupt requests are considered.



2270-33

Figure 4-4. Interrupt Sequences

Assume that the computer is running a CPU program when an interrupt from I/O channel 12 occurs (at reference time t1). A JSB instruction in the interrupt location for select code 12 causes a program jump to the service routine for the channel 12 device. The JSB instruction automatically saves the return address (in a location which the programmer must reserve in his routine) for a later return to the CPU program.

The routine for channel 12 is still in progress when several other devices request service (set flag). First, channels 13 and 14 request simultaneously at t2; however, since neither one has priority over channel 12, their flags are ignored and channel 12 continues its transfer. But at t3, a higher priority device on channel 10 requests service. This request interrupts the channel 12 transfer and causes the channel 10 transfer to begin. The JSB instruction saves the return address for return to the channel 12 routine.

During the channel 10 transfer, device 11 sets the channel 11 flag (t4). Since it has lower priority than channel 10, device 11 must wait until the end of the channel 10 routine. And since the channel 10 routine, when it ends, contains a return address to the channel 12 routine, program control temporarily returns to channel 12 (even though the waiting channel 11 has higher priority). The JMP,I instruction used for the return inhibits all interrupts until fully executed. At the end of this short interval, the channel 11 interrupt request is granted.

When channel 11 has finished its routine, control is returned to channel 12, which at last has sufficient priority to complete its routine. Since channel 12 has been saving a return address in the main CPU program, it returns control to this point.

The two waiting interrupt requests from channels 13 and 14 are now enabled. Channel 13 has the higher priority and goes first. At the end of the channel 13 routine, control is temporarily returned to the CPU program. Then, the lowest priority channel (channel 14) interrupts and completes its transfer. Finally, control is returned to the CPU program, which resumes processing.

4.3. INTERFACE ELEMENTS

The interface PCA provides the communication link between the computer and an external device. The interface PCA includes three basic elements which either the computer or the device can control in order to effect the necessary communication. These three elements are the control bit, flag bit, and buffer.

4.4. CONTROL BIT

This is a one-bit register used by the computer to turn on the device channel. When set, the control bit generates a start command to the device, allowing it to perform one operation cycle (e.g., read or write one character or word). The interface PCA cannot interrupt unless the control bit is set. The control bit is set by an STC (set control) instruction and cleared by a CLC (clear control) instruction, both of which must be accompanied by a specific select code (e.g., STC 12 or CLC 12). The device cannot affect the control bit.

4.5. FLAG BIT

This is a one-bit register primarily used by the device to indicate (when set) that a transmission between the device and the interface PCA buffer has been completed. Computer instructions can also set the flag (STF), clear the flag (CLF), test if it is set (SFS), and test if it is clear (SFC). The device cannot clear the flag bit. If the corresponding control bit is set, priority is high, and the interrupt system is enabled, setting the flag bit will cause an interrupt to the location corresponding to the device select code.

4-6. BUFFER

The buffer register is used for intermediate storage of data. Typically, the data capacity is 8 or 16 bits, but this is entirely dependent on the type of device.

4-7. INPUT/OUTPUT DATA TRANSFER

The following paragraphs describe how data is transferred between memory and input/output devices. A summary of I/O group instructions pertinent to the computer interrupt and control functions is provided in the appendix. The sequences presented for interrupt and noninterrupt methods of data transfer are highly simplified in order to present an overall view without the involvement of software operating systems and device drivers. For more detailed information, refer to the documentation supplied with the appropriate software system or I/O subsystem.

4-8. INPUT DATA TRANSFER (INTERRUPT METHOD)

Figure 4-5 illustrates the sequence of events required to input data using the interrupt method. Note that some operations are under control of the computer program (programmer's responsibility) and some of the operations are automatic. Note also that the interface PCA (device controller) is installed in the slot assigned to select code 12.

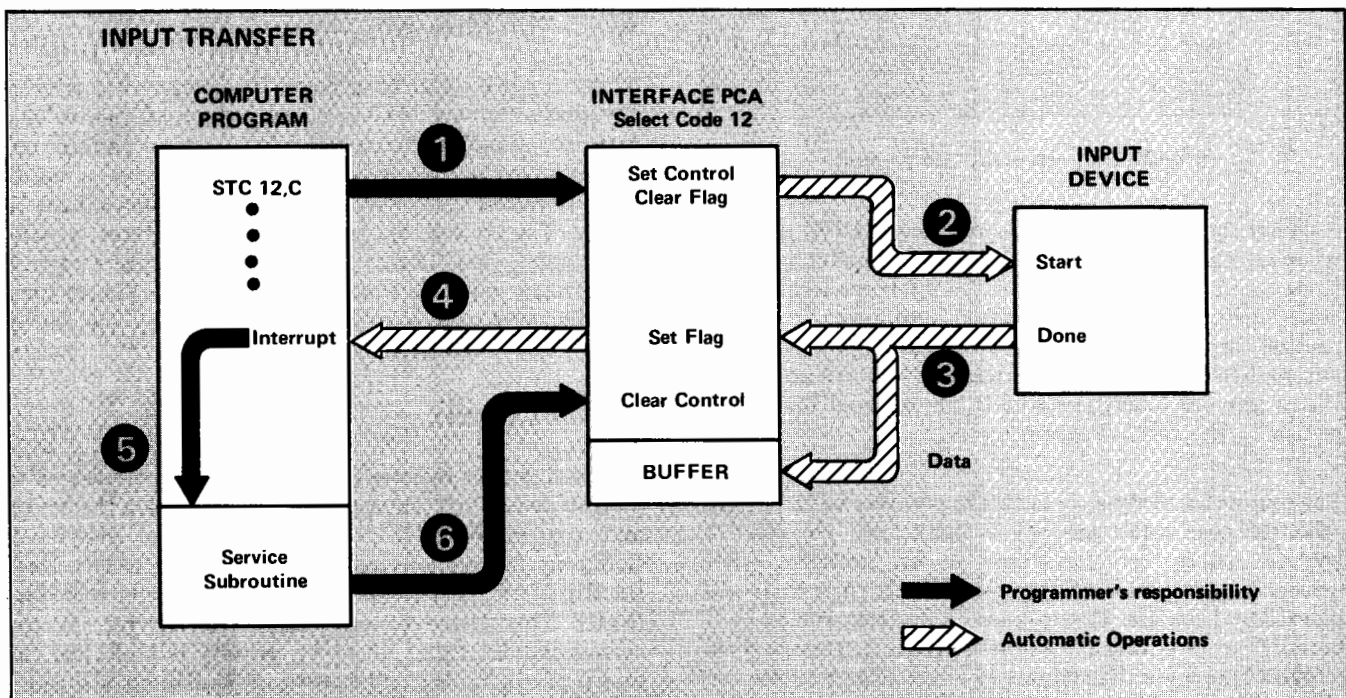
The operations begins (1) with the programmed instruction STC 12,C which sets the Control flip-flop and clears the Flag flip-flop on the interface PCA. Since the

next few operations are under control of the hardware, the computer program may continue the execution of other instructions. Setting the Control flip-flop causes the PCA to output a Start signal (2) to the device, which reads out a data character and asserts the Done signal (3).

The device Done signal sets the PCA Flag flip-flop, which in turn generates an interrupt (4) assuming that the interrupt conditions are met; i.e., the interrupt system must be on (STF 00 previously given), no higher priority interrupt is pending, and the Control flip-flop is set (done in step 1).

The interrupt causes the current computer program to be suspended and control is transferred to a service subroutine (5). It is the programmer's responsibility to provide the linkage between the interrupt location (00012 in this case) and the service subroutine. It is also the programmer's responsibility to include in his service subroutine the instructions for processing the data (loading into an accumulator, manipulating if necessary, and storing into memory).

The subroutine may then issue further STC 12,C commands to transfer additional data characters. One of the final instructions in the service subroutine must be CLC 12. This step (6) restores the interrupt capability to lower priority devices and returns the interface PCA to its static "ready" condition (Control clear and Flag set). This condition is initially established by the computer at power turn-on and it is the programmer's responsibility to return the interface PCA to the same condition on the completion of each data transfer operation. At the end of the subroutine, control is returned to the interrupted program via previously established linkages.



2270-6

Figure 4-5. Input Data Transfer (Interrupt Method)

4-9. OUTPUT DATA TRANSFER (INTERRUPT METHOD)

Figure 4-6 illustrates the sequence of events required to output data using the interrupt method. Again note the distinction between programmed and automatic instructions. It is assumed that the data to be transferred has been loaded into the A-register and is in a form suitable for output. The interface PCA in this example is assumed to be in the slot assigned to select code 13.

The output operation begins with a programmed instruction (OTA 13) to transfer the contents of the A-register to the interface PCA buffer (1). This is followed (2) by the instruction STC 13,C which sets the Control flip-flop and clears the Flag flip-flop on the interface PCA. Since the next few operations are under control of the hardware, the computer program may continue the execution of other instructions. Setting the Control flip-flop causes the PCA to output the buffered data and a Start signal (3) to the device, which writes (e.g., punches, stores, etc.) the data character and asserts the Done signal (4).

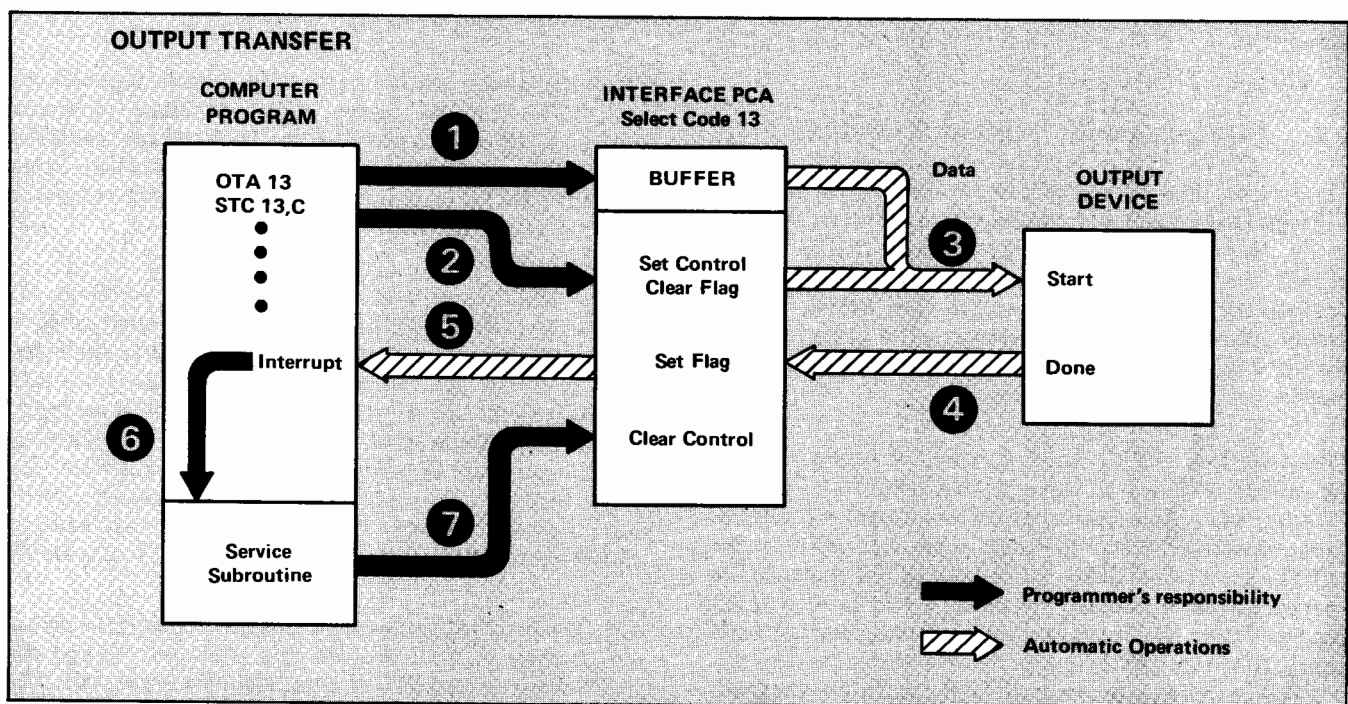
The device Done signal sets the PCA Flag flip-flop, which in turn generates an interrupt (5) provided that the interrupt system is on, priority is high, and the Control flip-flop is set (done in step 2). The interrupt causes the current computer program to be suspended, and control is transferred to a service subroutine (6). It is the programmer's responsibility to provide the linkage between the interrupt location (00013 in this case) and the service subroutine. The detailed contents of the subroutine are also the programmer's responsibility, and the contents will vary with the type of device.

The subroutine may then output further data to the interface PCA and reissue the STC 13,C command for additional data character transfers. One of the final instructions in the service subroutine must be a clear control (CLC 13). This step (7) allows lower priority devices to interrupt, and restores the channel to its static "ready" condition (Control clear and Flag set). At the end of the subroutine, control is returned to the interrupted program via previously established linkages.

4-10. NONINTERRUPT DATA TRANSFER

It is also possible to transfer data without using the interrupt system. This involves a "wait-for-flag" method in which the computer commands the device to operate and then waits for the completion response. In using this method to transfer data, it is assumed that the computer time is relatively unimportant. The programming is very simple, consisting of only four words of in-line coding as shown in table 4-1. Each of these routines will transfer one word or character of data. It is also assumed that the interrupt system is turned off (STF 00 not previously given).

4-11. INPUT. As described under paragraph 4-8, an STC 12,C instruction begins the operation by commanding the device to read one word or character. The computer then goes into a waiting loop, repeatedly checking the status of the flag bit. If the Flag flip-flop is not set, the JMP *-1 instruction causes a jump back to the SFS instruction. (The *-1 operand is assembler notation for "this location minus one.") When the Flag flip-flop is set, the skip condition for SFS is met and the JMP instruction is skipped. The computer thus exits from the waiting loop



2270-7

Figure 4-6. Output Data Transfer (Interrupt Method)

Table 4-1. Noninterrupt Transfer Routines

INPUT	
INSTRUCTIONS	COMMENTS
STC 12,C	Start device
SFS 12	Is input ready?
JMP *-1	No, repeat previous instruction
LIA 12	Yes, load input into A-register

OUTPUT	
INSTRUCTIONS	COMMENTS
OTA 13	Output A-register to buffer
STC 13,C	Start device
SFS 13	Has device accepted the data?
JMP *-1	No, repeat previous instruction
NOP	Yes, proceed

4-12. OUTPUT. The first step, which is to transfer the data to the interface PCA buffer, is the OTA 13 instruction. Then STC 13,C commands the device to operate and accept the data. The computer then goes into a waiting loop as described in the preceding paragraph. When the Flag flip-flop becomes set, indicating that the device has accepted the output data, the computer exits from the loop. (The final NOP is for illustration purposes only.)

4-13. DUAL-CHANNEL PORT CONTROLLER

The optional dual-channel port controller (DCPC) provides a direct data path, software assignable, between memory and a high-speed peripheral device; the DCPC accomplishes this by stealing an I/O cycle instead of interrupting to a service subroutine. The DCPC logic is capable of stealing every consecutive I/O cycle and can therefore transfer data at rates up to 616,666 words per second.

There are two DCPC channels, each of which may be separately assigned to operate with any I/O interface PCA, including those installed in the optional HP 12979A Input/Output Extender (assuming that the I/O extender DCPC option is installed). When both DCPC channels are operating simultaneously, channel 1 has priority over channel 2. The combined maximum transfer rate for both channels operating together is 616,666 words per second; the rate available to channel 2 is therefore the rate difference between 616,666 and the actual operating rate of channel 1.

CPU when the DCPC is operating is the difference between 616,666 words per second and the actual transfer rates of DCPC channels 1 and 2 combined.

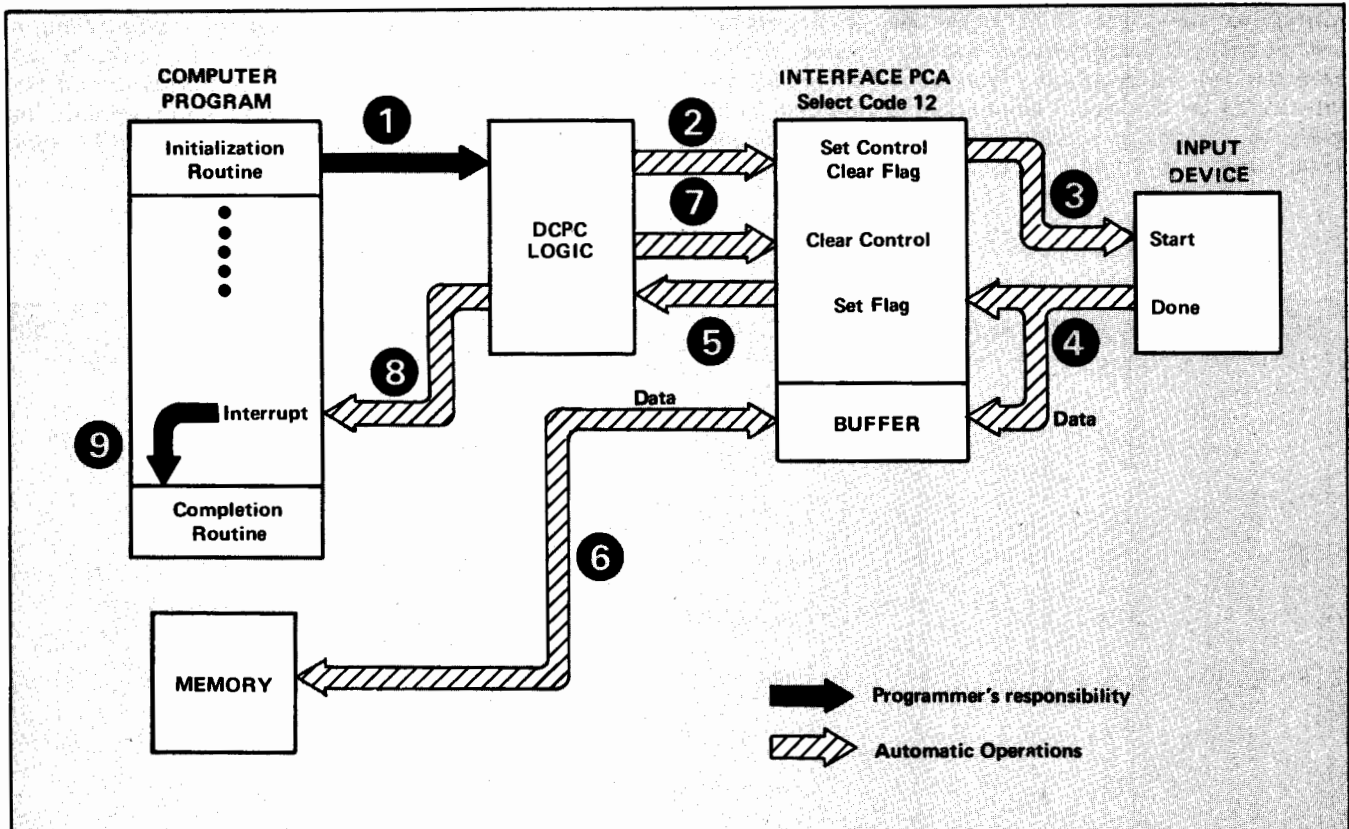
Transfers via the DCPC are on a full-word basis; hardware packing and unpacking of bytes are not provided. The word count register is a full 16 bits in length, and data transfers are accomplished in blocks. The transfer is initiated by an initialization routine, and from then on the operation is under automatic control of the hardware. The initialization routine specifies the direction of the data transfer (in or out), where in memory to read or write, which I/O channel to use, and how much data to transfer. Completion of the block transfer is signalled by an interrupt to location 00006 (for channel 1) or to location 00007 (for channel 2) if the interrupt system is enabled. It is also possible to check for completion by testing the status of the flag for select code 06 or 07, or by interrogating the word count register with an LIA/B to select code 02 (for channel 1) or to select code 03 (for channel 2). A block transfer in process can be aborted with an STF 06 or 07 instruction.

4-14. DCPC OPERATION. Figure 4-7 illustrates the sequence of operations for a DCPC input data transfer. A comparison with the conventional interrupt method (figure 4-5) shows that much more of the DCPC operation is automatic. Remember that the procedure in figure 4-5 must be repeated for each word or character. In figure 4-7, the automatic DCPC operation will transfer a block of data of any size limited only by the available memory space. The sequence of events is as follows. (An input data transfer is illustrated; the minor differences for an output transfer are explained in text.)

The initialization routine sets up the control registers on the DCPC (1) and issues the first start command (STC 12,C) directly to the interface PCA. (If the operation is an output, the interface PCA buffer is also loaded at this time.) The DCPC logic is now turned on and the computer program continues with other instructions.

Setting the Control and clearing the Flag flip-flops (2) causes the interface PCA to send a Start signal (with a data word if it is an output transfer) to the external device (3). The device goes through a read or write cycle and returns a Done signal (with a data word if it is an input transfer). The Done signal (4) sets the PCA Flag flip-flop which, regardless of priority, immediately requests the DCPC logic to steal an I/O cycle (5) and transfer a word into (or out of) memory. The process now repeats back to the beginning of this paragraph to transfer the next word.

After the specified number of words have been transferred, the interface PCA Control flip-flop is cleared (7) and the DCPC logic generates a completion interrupt (8). The program control is now forced to a completion routine (9), the contents of which is the programmer's responsibility.



2270-8

Figure 4-7. DCPC Input Data Transfer

4-15. DCPC INITIALIZATION. The information required to initialize the DCPC (direction, memory allocation, I/O channel assignment, and block length) are given by three control words. These three words must be addressed specifically to the DCPC. Figure 4-8 illustrates the format of the three control words. Control Word 1 (CW1) identifies the I/O channel to be used and provides two options selectable by the programmer:

Bit 15

1 = give STC (in addition to CLF) to I/O channel at end of each DCPC cycle (except on last cycle, if input)

0 = no STC

Bit 13

1 = give CLC to I/O channel at end of block transfer

0 = no CLC

Control Word 2 (CW2) gives the starting memory address for the block transfer and bit 15 determines whether data is to go into memory (logic 0) or out of memory (logic 1). Control Word 3 (CW3) is the two's complement of the number of words to be transferred into or out of memory (i.e., the block length). This number can be from -1 to 32,768, although it is limited in the practical case by available memory.

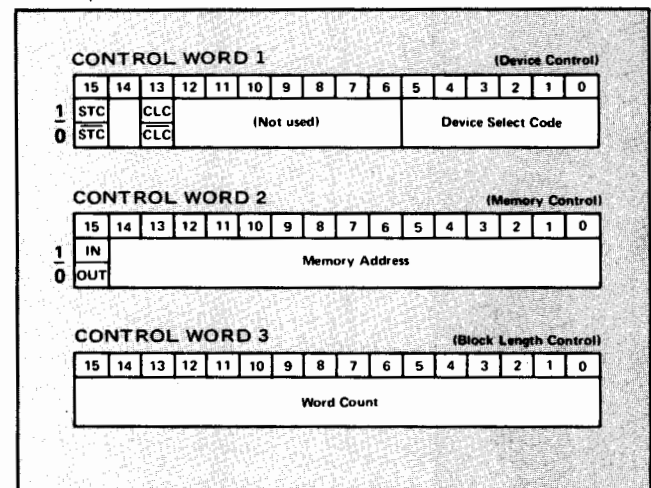


Figure 4-8. DCPC Control Word Formats

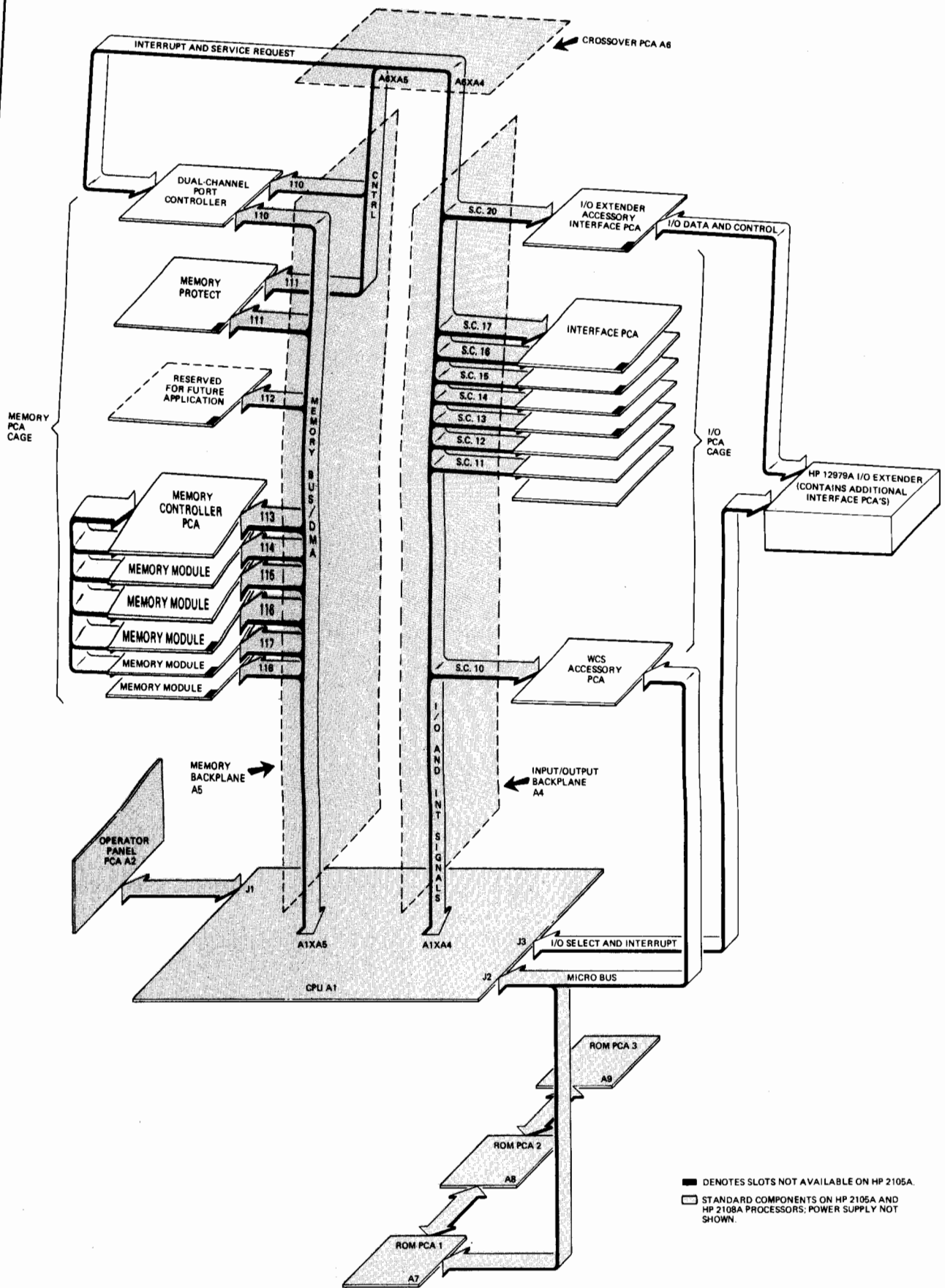
Table 4-2 gives the basic program sequence for outputting the control words to the DCPC. As shown in this table, CLC 2 and STC 2 perform switching functions to prepare the logic for either CW2 or CW3. The device is assumed to be in I/O slot channel 10, and it is also assumed that its start command is STC 10B, C. The sample values of CW1, CW2, and CW3 will read a block of 50 words and store these in locations 200 through 261 (octal). The STC 06B,C

instruction starts the DCPC operation. A flag-status method of detecting the end-of-transfer is used in this example; an interrupt to location 00006 could be substituted for this test. The program in table 4-2 could easily be changed to operate on channel 2 by changing select codes 2 to 3 and 6 to 7.

One important difference should be noted when doing a DCPC input operation from a disc or a drum. Due to the asynchronous nature of disc or drum memories and the design of the interface PCA, the order of starting must be reversed from the order given; i.e., start the DCPC first and then start the disc (or drum).

Table 4-2. DCPC Initialization Program

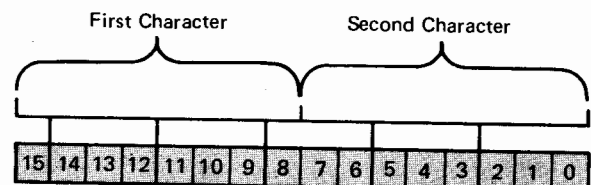
LABEL	OPCODE	OPERAND	COMMENTS
ASGN1	LDA	CW1	Fetches control word 1 (CW1) from memory and loads it in A-register.
	OTA	6B	Outputs CW1 to DCPC Channel 1.
MAR1	CLC	2B	Prepares Memory Address Register to receive control word 2 (CW2).
	LDA	CW2	Fetches CW2 from memory and loads it in A-register.
	OTA	2B	Outputs CW2 to DCPC Channel 1.
WCR1	STC	2B	Prepares Word Count Register to receive control word 3 (CW3).
	LDA	CW3	Fetches CW3 from memory and loads it in A-register.
	OTA	2B	Outputs CW3 to DCPC Channel 1.
STRT1	STC	10B,C	Start input device.
	STC	6B,C	Activate DCPC Channel 1.
	SFS	6B	Wait while data transfer takes place or, if interrupt processing is used,
	JMP	*-1	continue program.
⋮	⋮	⋮	
	HLT		Halt
CW1	OCT	120010	Assignment for DCPC Channel 1 (ASGN1); specifies I/O channel select code address (10 ₈), STC after each word is transferred, and CLC after final word is transferred.
CW2	OCT	100200	Memory Address Register control. DCPC Channel 1 (MAR1); specifies memory input operation and starting memory address (200 ₈).
CW3	DEC	-50	Word Count Register control. DCPC Channel 1 (WCR1); specifies the 2's complement of the number of character words in the block of data to be transferred (50 ₁₀).



CHARACTER CODES

ASCII Character	First Character Octal Equivalent	Second Character Octal Equivalent
A	040400	000101
B	041000	000102
C	041400	000103
D	042000	000104
E	042400	000105
F	043000	000106
G	043400	000107
H	044000	000110
I	044400	000111
J	045000	000112
K	045400	000113
L	046000	000114
M	046400	000115
N	047000	000116
O	047400	000117
P	050000	000120
Q	050400	000121
R	051000	000122
S	051400	000123
T	052000	000124
U	052400	000125
V	053000	000126
W	053400	000127
X	054000	000130
Y	054400	000131
Z	055000	000132
a	060400	000141
b	061000	000142
c	061400	000143
d	062000	000144
e	062400	000145
f	063000	000146
g	063400	000147
h	064000	000150
i	064400	000151
j	065000	000152
k	065400	000153
l	066000	000154
m	066400	000155
n	067000	000156
o	067400	000157
p	070000	000160
q	070400	000161
r	071000	000162
s	071400	000163
t	072000	000164
u	072400	000165
v	073000	000166
w	073400	000167
x	074000	000170
y	074400	000171
z	075000	000172
0	030000	000060
1	030400	000061
2	031000	000062
3	031400	000063
4	032000	000064
5	032400	000065
6	033000	000066
7	033400	000067
8	034000	000070
9	034400	000071
NUL	000000	000000
SOH	000400	000001
STX	001000	000002
ETX	001400	000003
EOT	002000	000004
ENQ	002400	000005

ASCII Character	First Character Octal Equivalent	Second Character Octal Equivalent
ACK	003000	000006
BEL	003400	000007
BS	004000	000010
HT	004400	000011
LF	005000	000012
VT	005400	000013
FF	006000	000014
CR	006400	000015
SO	007000	000016
SI	007400	000017
DLE	010000	000020
DC1	010400	000021
DC2	011000	000022
DC3	011400	000023
DC4	012000	000024
NAK	012400	000025
SYN	013000	000026
ETB	013400	000027
CAN	014000	000030
EM	014400	000031
SUB	015000	000032
ESC	015400	000033
FS	016000	000034
GS	016400	000035
RS	017000	000036
US	017400	000037
SPACE	020000	000040
!	020400	000041
"	021000	000042
#	021400	000043
\$	022000	000044
%	022400	000045
&	023000	000046
'	023400	000047
(024000	000050
)	024400	000051
*	025000	000052
+	025400	000053
,	026000	000054
-	026400	000055
.	027000	000056
/	027400	000057
:	035000	000072
;	035400	000073
<	036000	000074
=	036400	000075
>	037000	000076
?	037400	000077
@	040000	000100
[055400	000133
\	056000	000134
]	056400	000135
^	057000	000136
_	057400	000137
`	060000	000140
{	075400	000173
	076000	000174
}	076400	000175
~	077000	000176
DEL	077400	000177



OCTAL ARITHMETIC

ADDITION

TABLE

0	01	02	03	04	05	06	07
1	02	03	04	05	06	07	10
2	03	04	05	06	07	10	11
3	04	05	06	07	10	11	12
4	05	06	07	10	11	12	13
5	06	07	10	11	12	13	14
6	07	10	11	12	13	14	15
7	10	11	12	13	14	15	16

EXAMPLE

Add: 3677 octal
 + 1331 octal
(111-) carries
 5230 octal

MULTIPLICATION

TABLE

1	02	03	04	05	06	07
2	04	06	10	12	14	16
3	06	11	14	17	22	25
4	10	14	20	24	30	34
5	12	17	24	31	36	43
6	14	22	30	36	44	52
7	16	25	34	43	52	61

EXAMPLE

Multiply: 657 octal
 X 54 octal

3274
 4153
 45024 octal

(Reminder: add in octal)

COMPLEMENT

To find the two's complement form of an octal number. (Same procedure whether converting from positive to negative or negative to positive.)

RULE

1. Subtract from the maximum representable octal value.
2. Add one.

EXAMPLE

Two's complement of 556_8 :

17777
 - 000556
 177221
 + 1
 177222₈

OCTAL/DECIMAL CONVERSIONS

OCTAL TO DECIMAL

TABLE

OCTAL	DECIMAL
0-7	0-7
10-17	8-15
20-27	16-23
30-37	24-31
40-47	32-39
50-57	40-47
60-67	48-55
70-77	56-63
100	64
200	128
400	256
1000	512
2000	1024
4000	2048
10000	4096
20000	8192
40000	16384
77777	32767

EXAMPLE

Convert 463_8 to a decimal integer.

$$400_8 = 256_{10}$$

$$60_8 = 48_{10}$$

$$3_8 = \underline{3_{10}}$$

307 decimal

DECIMAL TO OCTAL

TABLE

DECIMAL	OCTAL
1	1
10	12
20	24
40	50
100	144
200	310
500	764
1000	1750
2000	3720
5000	11610
10000	23420
20000	47040
32767	77777

EXAMPLE

Convert 5229_{10} to an octal integer.

$$5000_{10} = 11610_8$$

$$200_{10} = 310_8$$

$$20_{10} = 24_8$$

$$9_{10} = \underline{11_8}$$

$$12155_8$$

(Reminder: add in octal)

NEGATIVE DECIMAL TO TWO'S COMPLEMENT OCTAL

TABLE

DECIMAL	2's COMP
-1	177777
-10	177766
-20	177754
-40	177730
-100	77634
-200	177470
-500	177014
-1000	176030
-2000	174040
-5000	168170
-10000	154360
-20000	130740
-32768	100000

EXAMPLE

Convert -629_{10} to two's complement octal.

$$-500_{10} = 177014_8$$

$$-100_{10} = 177634_8$$

$$-20_{10} = 177754_8 \quad (\text{Add in octal})$$

$$-9_{10} = \underline{177767_8}$$

$$176613_8$$

For reverse conversion (two's complement octal to negative decimal):

1. Complement, using procedure on facing page.
2. Convert to decimal, using OCTAL TO DECIMAL table.

MATHEMATICAL EQUIVALENTS

$2 \pm n$ IN DECIMAL

2^n	n	2^{-n}							
1	0	1.0	65 536	16	0.00001	52587	89062	5	
2	1	0.5	131 072	17	0.00000	76293	94531	25	
4	2	0.25							
8	3	0.125	262 144	18	0.00000	38146	97265	625	
16	4	0.0625	524 288	19	0.00000	19073	48632	8125	
32	5	0.03125	1 048 576	20	0.00000	09536	74316	40625	
64	6	0.01562 5							
128	7	0.00781 25	2 097 152	21	0.00000	04768	37158	20312 5	
256	8	0.00390 625	4 194 304	22	0.00000	02384	18579	10156 25	
512	9	0.00195 3125	8 388 608	23	0.00000	01192	09289	55078 125	
1 024	10	0.00097 65625	16 777 216	24	0.00000	00596	04644	77539 0625	
2 048	11	0.00048 82812 5	33 554 432	25	0.00000	00298	02322	38769 53125	
4 096	12	0.00024 41406 25	67 108 864	26	0.00000	00149	01161	19384 76562 5	
8 192	13	0.00012 20703 125	134 217 728	27	0.00000	00074	50580	59692 38281 25	
16 384	14	0.00006 10351 5625	268 435 456	28	0.00000	00037	25290	29846 19140 625	
32 768	15	0.00003 05175 78125	536 870 912	29	0.00000	00018	62645	14923 09570 3125	
			1 073 741 824	30	0.00000	00009	31322	57461 54785 15625	
			2 147 483 648	31	0.00000	00004	65661	28730 77392 57812 5	
			4 294 967 296	32	0.00000	00002	32830	64365 38696 28906 25	

$10 \pm n$ IN OCTAL

10^n	n	10^{-n}							
1	0	1.000 000 000 000 000 00	112 402 762 000	10	0.000 000 000 006 676 337 66				
12	1	0.063 146 314 631 463 146 31	1 351 035 564 000	11	0.000 000 000 000 537 657 77				
144	2	0.005 075 341 217 270 243 66	16 432 451 210 000	12	0.000 000 000 000 043 136 32				
1 750	3	0.000 406 111 564 570 651 77	221 411 634 520 000	13	0.000 000 000 000 003 411 35				
23 420	4	0.000 032 155 613 530 704 15	2 657 142 036 440 000	14	0.000 000 000 000 000 264 11				
303 240	5	0.000 002 476 132 610 706 64	34 327 724 461 500 000	15	0.000 000 000 000 000 022 01				
3 641 100	6	0.000 000 206 157 364 055 37	434 157 115 760 200 000	16	0.000 000 000 000 000 001 63				
46 113 200	7	0.000 000 015 327 745 152 75	5 432 127 413 542 400 000	17	0.000 000 000 000 000 000 14				
575 360 400	8	0.000 000 001 257 143 561 06	67 405 553 164 731 000 000	18	0.000 000 000 000 000 000 01				
7 346 545 000	9	0.000 000 000 104 560 276 41							

MATHEMATICAL EQUIVALENTS

 2^x IN DECIMAL

x	2^x
0.001	1.00069 33874 62581
0.002	1.00138 72557 11335
0.003	1.00208 16050 79633
0.004	1.00277 64359 01078
0.005	1.00347 17485 09503
0.006	1.00416 75432 38973
0.007	1.00486 38204 23785
0.008	1.00556 05803 98468
0.009	1.00625 78234 97782

x	2^x
0.01	1.00695 55500 56719
0.02	1.01395 94797 90029
0.03	1.02101 21257 07193
0.04	1.02811 38266 56067
0.05	1.03526 49238 41377
0.06	1.04246 57608 41121
0.07	1.04971 66836 23067
0.08	1.05701 80405 61380
0.09	1.06437 01824 53360

x	2^x
0.1	1.07177 34625 36293
0.2	1.14869 83549 97035
0.3	1.23114 44133 44916
0.4	1.31950 79107 72894
0.5	1.41421 35623 73095
0.6	1.51571 65665 10398
0.7	1.62450 47927 12471
0.8	1.74110 11265 92248
0.9	1.86606 59830 73615

 $n \log_{10} 2, n \log_2 10$ IN DECIMAL

n	$n \log_{10} 2$
1	0.30102 99957
2	0.60205 99913
3	0.90308 99870
4	1.20411 99827
5	1.50514 99783

$n \log_2 10$
3.32192 80949
6.64385 61898
9.96578 42847
13.28771 23795
16.60964 04744

n	$n \log_{10} 2$
6	1.80617 99740
7	2.10720 99696
8	2.40823 99653
9	2.70926 99610
10	3.01029 99566

$n \log_2 10$
19.93156 85693
23.25349 66642
26.57542 47591
29.89735 28540
33.21928 09489

MATHEMATICAL CONSTANTS IN OCTAL SCALE

$\pi =$	(3.11037 552421) ₍₈₎
$\pi^{-1} =$	(0.24276 301556) ₍₈₎
$\sqrt{\pi} =$	(1.61337 611067) ₍₈₎
$\ln \pi =$	(1.11206 404435) ₍₈₎
$\log_2 \pi =$	(1.51544 163223) ₍₈₎
$\sqrt{10} =$	(3.12305 407267) ₍₈₎

$e =$	(2.55760 521305) ₍₈₎
$e^{-1} =$	(0.27426 530661) ₍₈₎
$\sqrt{e} =$	(1.51411 230704) ₍₈₎
$\log_{10} e =$	(0.33626 754251) ₍₈₎
$\log_2 e =$	(1.34252 166245) ₍₈₎
$\log_2 10 =$	(3.24464 741136) ₍₈₎

$\gamma =$	(0.44742 147707) ₍₈₎
$\ln \gamma =$	-(0.43127 233602) ₍₈₎
$\log_2 \gamma =$	-(0.62573 030645) ₍₈₎
$\sqrt{2} =$	(1.32404 746320) ₍₈₎
$\ln 2 =$	(0.54271 027760) ₍₈₎
$\ln 10 =$	(2.23273 067355) ₍₈₎

Refer to octal instruction codes given on the following page.
To combine code for indirect addressing, merge "100000" with octal instruction code.

REGISTER REFERENCE INSTRUCTIONS

Shift-Rotate Group (SRG)

1. select to operate on A or B
2. select 1 to 4 micros, not more than one from each column.
3. combine octal codes (leading zeros omitted) by inclusive or.
4. order of execution is from column 1 to column 4.

A Operations

1	2	3	4
ALS (1000)	CLE (40)	SLA (10)	ALS (20)
ARS (1100)			ARS (21)
RAL (1200)			RAL (22)
RAR (1300)			RAR (23)
ALR (1400)			ALR (24)
ERA (1500)			ERA (25)
ELA (1600)			ELA (26)
ALF (1700)			ALF (27)

B Operations

1	2	3	4
BLS (5000)	CLE (4040)	SLB (4010)	BLS (4020)
BRS (5100)			BRS (4021)
RBL (5200)			RBL (4022)
RBR (5300)			RBR (4023)
BLR (5400)			BLR (4024)
ERB (5500)			ERB (4025)
ELB (5600)			ELB (4026)
BLF (5700)			BLF (4027)

Alter-Skip Group (ASG)

1. select to operate on A or B.
2. select 1 to 8 micros, not more than one from each column.
3. combine octal codes (leading zeros omitted) by inclusive or.
4. order of execution is from column 1 to column 8.

A Operations

1	2	3	4
CLA (2400)	SEZ (2040)	CLE (2100)	SSA (2020)
CMA (3000)		CME (2200)	
CCA (3400)		CCE (2300)	

5	6	7	8
SLA (2010)	INA (6040)	CLE (6100)	RSS (2001)

B Operations

1	2	3	4
CLB (6400)	SEZ (6040)	CLE (6100)	SSB (6020)
CMB (7000)		CME (6200)	
CCB (7400)		CCE (6300)	

5	6	7	8
SLB (6010)	INB (6004)	SZB (6002)	RSS (6001)

INPUT/OUTPUT INSTRUCTIONS

Clear Flag

Refer to octal instruction codes given on the following page.
To clear flag after execution (instead of holding flag), merge "001000" with octal instruction code.





INSTRUCTION CODES IN OCTAL

[illegible]

BASE SET INSTRUCTION CODES IN BINARY

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
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
0	SRG	000		A/B	0	D/E	*LS		000	†CLE	D/E	‡SL*	*LS		000
				A/B	0	D/E	*RS		001				*RS		001
				A/B	0	D/E	R*L		010				R*L		010
				A/B	0	D/E	R*R		011				R*R		011
				A/B	0	D/E	*LR		100				*LR		100
				A/B	0	D/E	ER*		101				ER*		101
				A/B	0	D/E	EL*		110				EL*		110
				A/B	0	D/E	*LF		111				*LF		111
				NOP		000			000						000
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	ASG	000		A/B	1		CL*	01		CLE		01	SEZ	SS*	SL*
				A/B			CM*	10		CME		10		IN*	SZ*
				A/B			CC*	11		CCE		11			RSS
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	IOG	000			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	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	EAG	000		MPY**		000			010		000			000	
				DIV**		000			100		000			000	
				DLD**		100			010		000			000	
				DST**		100			100		000			000	
				ASR		001			000		0	1			
				ASL		000			000		0	1			
				LSR		001			000		1	0			
				LSL		000			000		1	0			
				RRR		001			001		0	0			
				RRL		000			001		0	0			

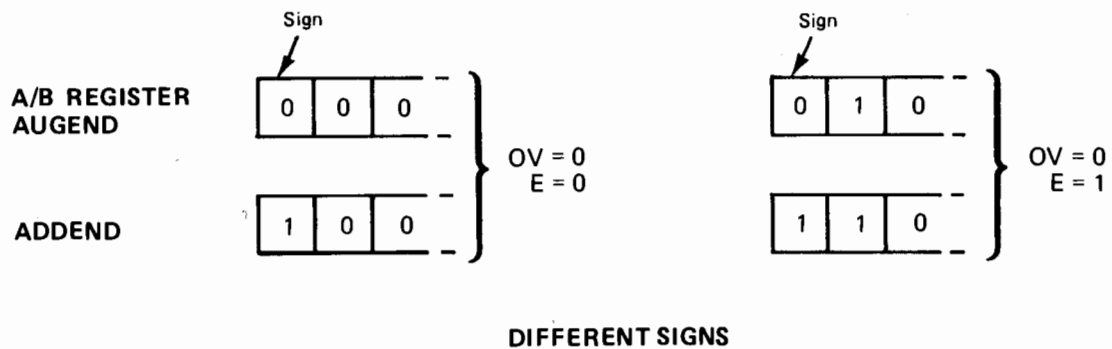
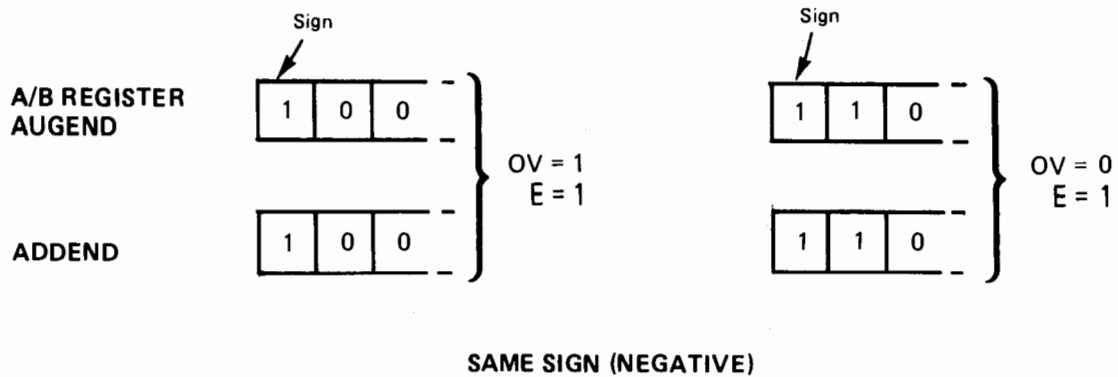
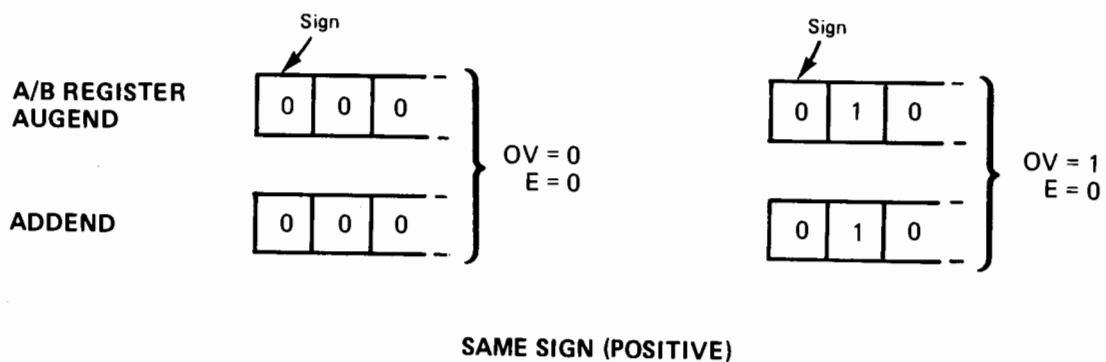
EXTENDED INSTRUCTION GROUP CODES IN BINARY

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAX/SAY/SBX/SBY	1	0	0	0	A/B	0	1	1	1	1	1	0	X/Y	0	0	0
CAX/CAY/CBX/CBY	1	0	0	0	A/B	0	1	1	1	1	1	0	X/Y	0	0	1
LAX/LAY/LBX/LBY	1	0	0	0	A/B	0	1	1	1	1	1	0	X/Y	0	1	0
STX/STY	1	0	0	0	1	0	1	1	1	1	1	0	X/Y	0	1	1
CXA/CYA/CXB/CYB	1	0	0	0	A/B	0	1	1	1	1	1	0	X/Y	1	0	0
LDX/LDY	1	0	0	0	1	0	1	1	1	1	1	0	X/Y	1	0	1
ADX/ADY	1	0	0	0	1	0	1	1	1	1	1	0	X/Y	1	1	0
XAX/XAY/XBX/XBY	1	0	0	0	A/B	0	1	1	1	1	1	0	X/Y	1	1	1
ISX/ISY/DSX/DSY	1	0	0	0	1	0	1	1	1	1	1	1	X/Y	0	0	I/D
JUMP INSTRUCTIONS	1	0	0	0	1	0	1	1	1	1	1	1		0	1	0
	<div>JLY = 0</div> <div>JPY = 1</div>															
BYTE INSTRUCTIONS	1	0	0	0	1	0	1	1	1	1	1	0				
	<div>LBT = 0 1 1</div> <div>SBT = 1 0 0</div> <div>MBT = 1 0 1</div> <div>CBT = 1 1 0</div> <div>SFB = 1 1 1</div>															
BIT INSTRUCTIONS	1	0	0	0	1	0	1	1	1	1	1	1				
	<div>SBS = 0 1 1</div> <div>CBS = 1 0 0</div> <div>TBS = 1 0 1</div>															
WORD INSTRUCTIONS	1	0	0	0	1	0	1	1	1	1	1	1	1	1		
	<div>CMW = 0</div> <div>MVW = 1</div>															

	Clears all Control FF's from S.C. 06 and up; effectively turns off all I/O devices.	NOP	Prepares DCPC channel 1 to receive and store the direction of data flow and the starting memory address.	Prepares DCPC channel 2 to receive and store the direction of data flow and the starting memory address.
STF	Turns on interrupt system.	STO sets overflow bit.	NOP	NOP
CLF	Turns off interrupt system except power fail (S.C. 04) and parity error (S.C. 05).	CLO clears overflow bit.	NOP	NOP
SFS	Skip if interrupt system is on.	SOS	NOP	NOP
SFC	Skip if interrupt system is off.	SOC	NOP	NOP
LIA/B	NOP	Loads S-register contents into A/B register.	Loads present contents of DCPC channel 1 word count register into A/B register.	Loads present contents of DCPC channel 2 word count register into A/B register.
MIA/B	NOP	Merges S-register contents into A/B register.	Merges present contents of DCPC channel 1 word count register into A/B register.	Merges present contents of DCPC channel 2 word count register into A/B register.
OTA/B	NOP	Outputs A/B register contents into display register.	<ol style="list-style-type: none"> 1. Outputs to DCPC channel 1 the block length in 2's complement form (previously prepared by an STC 02 instruction). 2. Outputs to DCPC channel 1 the direction of data flow and the starting memory address (previously prepared by a CLC 02 instruction). 	<ol style="list-style-type: none"> 1. Outputs to DCPC channel 2 the block length in 2's complement form (previously prepared by an STC 03 instruction). 2. Outputs to DCPC channel 2 the direction of data flow and the starting memory address (previously prepared by a CLC 03 instruction).

S.C. 04	S.C. 05	S.C. 06	S.C. 07	S.C. 10-77
Re-initializes power-fail logic and restores interrupt capability to lower priority functions.	Turns on memory protect.	Sets Control FF on DCPC channel 1 (activates DMA).	Sets Control FF on DCPC channel 2 (activates DMA).	Sets PCA Control FF and turns on device on channel specified by S.C.
Re-initialize power-fail logic.	NOP	Clears Control FF on DCPC channel 1 (reestablishes priority with STF; does not turn off DCPC).	Clears Control FF on DCPC channel 2 (reestablishes priority with STF; does not turn off DCPC).	Clears PCA Control FF and turns off device (aborts data transfer if programmed).
Flag FF sets automatically when power comes up. (No program control possible.)	Turns on parity error interrupt.	1. Hardware Controlled: Sets Flag FF when word count is reached and turns off DCPC channel 1. 2. Program Controlled: Aborts data transfer.	1. Hardware Controlled: Sets Flag FF when word count is reached and turns off DCPC channel 2. 2. Program Controlled: Aborts data transfer.	PCA Flag FF sets when data transfer is complete.
Flag FF clears automatically when power fail occurs. (No program control possible.)	Turns off parity error interrupt.	Clears Flag FF on DCPC channel 1.	Clears Flag FF on DCPC channel 2.	Clears PCA Flag FF.
NOP	Skip if parity error interrupt is on.	Tests if DCPC channel 1 data transfer is complete.	Tests if DCPC channel 2 data transfer is complete.	Tests if I/O channel data transfer is complete.
Skip if power fail has occurred.	Skip if parity error interrupt is off.	Tests if DCPC channel 1 data transfer is still in progress.	Tests if DCPC channel 2 data transfer is still in progress.	Tests if I/O channel data transfer is still in progress.
Loads contents of central interrupt register (S.C. of last interrupting device) into least-significant bits of A/B register.	Loads contents of violation register into A/B register: Bit 15 = 1 = PE Bit 15 = 0 = MPV	NOP	NOP	Loads contents of PCA buffer into A/B register.
Merges contents of central interrupt register into least-significant bits of A/B register.	Merges contents of violation register into A/B register.	NOP	NOP	Merges contents of PCA buffer into A/B register.
NOP	Outputs first addressable memory location to fence register.	Outputs to DCPC channel 1 the S.C. of I/O channel. Specify STC after each word; CLC after block.	Outputs to DCPC channel 2 the S.C. of I/O channel. Specify STC after each word; CLC after block.	Outputs data from A/B register into PCA buffer.

EXTEND AND OVERFLOW EXAMPLES



ELECTRONIC

SALES & SERVICE OFFICES

UNITED STATES

ALABAMA

8290 Whitesburg Dr., S.E.
P.O. Box 4207
Huntsville 35892
Tel: (205) 881-4591
TWX: 810-726-2204

ARIZONA

2336 E. Magnolia St.
Phoenix 85034
Tel: (602) 244-1361
TWX: 910-951-1330

5737 East Broadway
Tucson 85711
Tel: (602) 298-2313
TWX: 910-952-1162
(Effective Dec. 15, 1973)
2424 East Aragon Rd.
Tucson 85706
Tel: (602) 889-4661

CALIFORNIA

1430 East Orangethorpe Ave.
Fullerton 92631
Tel: (714) 870-1000
TWX: 910-592-1288

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

6305 Arizona Place
Los Angeles 90045
Tel: (213) 649-2511
TWX: 910-328-6148

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
P.O. Box 23333
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

12 Lunar Drive
New Haven 06525
Tel: (203) 389-6551
TWX: 710-465-2029

FLORIDA

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

P.O. Box 13910
6177 Lake Ellenor Dr.
Orlando 32809
Tel: (305) 859-2900
TWX: 810-850-0113

GEORGIA

P.O. Box 28234
450 Interstate North
Atlanta 30328
Tel: (404) 436-6181
TWX: 910-766-4890

HAWAII

2875 So. King Street
Honolulu 96814
Tel: (808) 955-4455

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 840
3239 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

20010 Century Blvd.
Germantown 20767
Tel: (31) 428-0700
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

23855 Research Drive
Farmington 48024
Tel: (313) 476-6400
TWX: 810-242-2900

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

148 Weldon Parkway
Maryland Heights 63043
Tel: (314) 567-1455
TWX: 910-764-0830

*NEVADA

Las Vegas
Tel: (702) 382-5777

NEW JERSEY

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

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

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

6 Automation Lane
Computer Park
Albany 12205
Tel: (518) 458-1550
TWX: 710-441-8270

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

New York City

Manhattan, Bronx
Contact Paramus, NJ Office
Tel: (201) 265-5000
Brooklyn, Queens, Richmond
Contact Woodbury, NY Office
Tel: (516) 921-0300

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: 510-221-2168

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
Dayton 45449
Tel: (513) 859-8202
TWX: 810-427-9129

330 Progress Rd.
Dayton 45449
Tel: (513) 859-8202
TWX: 810-459-1925

6665 Busch Blvd.
Columbus 43229
Tel: (614) 846-1300

OKLAHOMA

P.O. Box 32008
Oklahoma City 73132
Tel: (405) 721-0200
TWX: 910-830-6862

OREGON

17890 SW Boones Ferry Road
Tualatin 97062
Tel: (503) 620-3350
TWX: 910-467-8714

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

*TENNESSEE

Memphis
Tel: (901) 274-7472

TEXAS

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

P.O. Box 27409
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

VIRGINIA

P.O. Box 6514
2111 Spencer Road
Richmond 23230
Tel: (804) 285-3431
TWX: 910-956-0157

WASHINGTON

Bellevue Office Pk.
1203 - 114th SE
Bellevue 98004
Tel: (206) 454-3971
TWX: 910-443-2303

*WEST VIRGINIA

Charleston
Tel: (304) 345-1640

WISCONSIN

9431 W. Beloit Road
Suite 117
Milwaukee 53227
Tel: (414) 541-0550

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.
11748 Kingsway Ave.
Edmonton T5C 0X5
Tel: (403) 452-3670
TWX: 610-831-2431

Hewlett-Packard (Canada) Ltd.
825 - 8th Ave., S.W.
Suite 804
Calgary
Tel: (403) 262-4279

BRITISH COLUMBIA

Hewlett-Packard (Canada) Ltd.
837 E. Cordova St.
Vancouver 6
Tel: (604) 254-0531

MANITOBA

Hewlett-Packard (Canada) Ltd.
513 Century St.
Winnipeg
Tel: (204) 786-7581
TWX: 610-671-3531

NOVA SCOTIA

Hewlett-Packard (Canada) Ltd.
513 Century Village Rd.
Suite 210
Halifax
Tel: (902) 455-0511
TWX: 610-271-4482

ONTARIO

Hewlett-Packard (Canada) Ltd.
275 Woodward Dr.
Ottawa K2C 0P9
Tel: (613) 255-6180, 255-6530
TWX: 610-562-8968

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

QUEBEC

Hewlett-Packard (Canada) Ltd.
275 Hermès Boulevard
Pointe Claire H9R 1G7
Tel: (514) 697-4232
Telex: 01-20607

Hewlett-Packard (Canada) Ltd.
2376 Galvani Street
Stefey G1N 4G4
Tel: (418) 688-8710

FOR CANADIAN AREAS NOT LISTED:

Contact Hewlett-Packard (Canada) Ltd. in Pointe Claire.

CENTRAL AND SOUTH AMERICA

ARGENTINA

Hewlett-Packard Argentina
S.A.C.e.I.
Lavalle 1171 - 3°
Buenos Aires
Tel: 35-0436, 35-0627, 35-0341
Telex: 012-1009
Cable: HEWPACK ARG

BOLIVIA

Stambuk & Mark (Bolivia) LTDA.
Av. Mariscal, Santa Cruz 1342
La Paz
Tel: 40626, 53163, 52421
Telex: 3560014
Cable: BUKMAR

BRAZIL

Hewlett-Packard Do Brasil
I.E.C. Ltda.
Rua Frei Caneca 1119
01307-Sao Paulo-SP
Tel: 288-7111, 287-5858
Telex: 309151/2/3
Cable: HEWPACK Sao Paulo

Hewlett-Packard Do Brasil
I.E.C. Ltda.
Praça Dom Feliciano, 78
90000-Porto Alegre-RS
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
20000-Rio de Janeiro-GB
Santiago
Tel: 265-2643
Telex: 210079 HEWPACK
Cable: HEWPACK Rio de Janeiro

CHILE
Héctor Calcañi y Cia, Ltda.
Casilla 15 475
Santiago
Tel: 423 96
Cable: CALCAGNI Santiago

COLOMBIA
Instrumentación
Henrik A. Langebaek & Kier S.A.
Carrera 7 No. 48-59
Apartado Aéreo 6287
Bogotá, D.E.
Tel: 45-78-06, 45-55-46
Cable: AARIS Bogotá
Telex: 44400INSTCO

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

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

EL SALVADOR
Electrónica Asociados
Apartado Postal 1582
Centro Comercial Gigante
San Salvador, El Salvador C.A.
Paseo Escalón 4649-4° Piso
Tel: 23-44-60, 23-32-37
Cable: ELECAS

GUATEMALA
IPESA
Avenida La Reforma 3-48,
Zona 9
Guatemala
Tel: 63627, 64736
Telex: 4192 TELTRO GU

MEXICO
Hewlett-Packard Mexicana,
S.A. de C.V.
Torres Adalid No. 21, 11 Piso
Col. del Valle
Mexico 12, D.F.
Tel: 543-42-32
Telex: 017-74-507

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

PANAMA
Electrónico Baiboa, S.A.
P.O. Box 4929
Ave. Manuel Espinosa No. 13-50
Bldg. Alina
Panama City
Tel: 230833
Telex: 3481103, Curunda,
Canal Zone
Cable: ELECTRON Panama City

PARAGUAY
Z. J. Melamed S.R.L.
División: Aparatos y Equipos
Médicos
División: Aparatos y Equipos
Científicos y de Investigación
P.O. Box 676
Chile, 482, Edificio Victoria
Asunción
Tel: 4-5069, 4-6272
Cable: RAMEL

PERU
Compañía Electro Médica 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
Cable: SATRON 3450 332

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
Edificio Segre
Tercera Transversal
Los Ruices Norte
Caracas 107
Tel: 35-00-11
Telex: 21146 HEWPACK
Cable: HEWPACK Caracas

FOR AREAS NOT LISTED,

CONTACT:
Hewlett-Packard
Inter-Americas
3200 Hillview Ave.
Palo Alto, California 94304
Tel: (415) 493-1501
TWX: 910-373-1267
Cable: HEWPACK Palo Alto
Telex: 034-8300, 034-8493

Hewlett-Packard Ges.m.b.H.
Handelska 52/3
P.O. Box 7
A-1205 Vienna
Tel: (0222) 33 66 06 to 09
Cable: HEWPAK Vienna
Telex: 75923 hewpak a

BELGIUM
Hewlett-Packard Benelux
S.A./N.V.
Avenue de Col-Vert, 1,
(Groenkruglaan)
B-1170 Brussels
Tel: (02) 72 22 40
Cable: PALOBEN Brussels
Telex: 23 494 paloben bru

DENMARK
Hewlett-Packard A/S
Datavej 38
DK-3460 Birkerød
Tel: (01) 81 66 40
Cable: HEWPAK AS
Telex: 166 40 hp as

Hewlett-Packard A/S
Torvet 9
DK-8600 Silkeborg
Tel: (06) 82 71 66
Telex: 166 40 hp as
Cable: HEWPAK AS

FINLAND
Hewlett-Packard Oy
Bulevardi 26
P.O. Box 12185
SF-00120 Helsinki 12
Tel: (90) 13730
Cable: HEWPAK OY Helsinki
Telex: 12-15363 hel

FRANCE
Hewlett-Packard France
Quartier de Courtaubouff
Boite Postale No. 6
F-91401 Orsay
Tel: (1) 907 78 25
Cable: HEWPAK Orsay
Telex: 60048

Hewlett-Packard France
Agence Regionale
4 Quai des Etoiles
F-69321 Lyon Cedex 1
Tel: (78) 42 63 49
Cable: HEWPAK Lyon
Telex: 31617

Zone Aéronautique
Avenue Clement Ader
F-31770 Colomiers
Tel: (61) 86 81 55
Telex: 51957

Hewlett-Packard France
Agence Régionale
Boulevard Ferialo-Gamarra
Boite Postale No. 11
F-13100 Luyens
Tel: (47) 24 00 66
Telex: 41770

Hewlett-Packard France
Agence Régionale
63, Avenue de Rochester
F-35000 Rennes
Tel: (99) 36 33 21
Telex: 74912 F

Hewlett-Packard France
Agence Régionale
74, Allée de la Robertsau
F-57000 Strasbourg
Tel: (88) 35 23 20/21
Telex: 89141
Cable: HEWPAK STRBG

GERMAN FEDERAL
REPUBLIC
Hewlett-Packard GmbH
Vertriebszentrale Frankfurt
Bernerstrasse 117
Postfach 560 140
D-6000 Frankfurt 56
Tel: (0611) 50 04-1
Cable: HEWPAKSA Frankfurt
Telex: 41 32 49 fra

Hewlett-Packard GmbH
Vertriebsbüro Böblingen
Herrnbergerstrasse 110
D-7030 Böblingen, Württemberg
Tel: (07031) 66 72 87
Cable: HEPAK Böblingen
Telex: 72 65 739 bbn

Hewlett-Packard GmbH
Vertriebsbüro Düsseldorf
Vogelsanger Weg 38
D-4000 Düsseldorf
Tel: (0211) 63 80 31/38
Telex: 85/86 533 hpdd d

Vertriebsbüro Hamburg
Wendenstr. 23
D-2000 Hamburg 1
Tel: (040) 24 13 93
Cable: HEWPAKSA Hamburg
Telex: 21 63 032 hphh d

Hewlett-Packard GmbH
Vertriebsbüro Hannover
Meiendorfer Strasse 3
D-3000 Hannover-Kleefeld
Tel: (0511) 55 06 26

Hewlett-Packard GmbH
Vertriebsbüro Nürnberg
Hersbruckerstrasse 42
D-8500 Nürnberg
Tel: (0911) 57 10 66
Telex: 623 860

Hewlett-Packard GmbH
Vertriebsbüro München
Unterhachinger Strasse 28
ISAR Center
D-8012 Ottobrunn
Tel: (089) 601 30 61/7
Telex: 52 49 85
Cable: HEWPAKSA München
(West Berlin)

Hewlett-Packard GmbH
Vertriebsbüro Berlin
Wilmerdorfer Strasse 113/114
D-1000 Berlin W. 12
Tel: (030) 3137046
Telex: 18 34 05 hpbln d

GREECE
Kostas Karayannis
18, Erμου Street
GR-Athens 126
Tel: 3230-303, 3230-305
Cable: RAKAR Athens
Telex: 21 59 62 rak ar

IRELAND
Hewlett-Packard Ltd.
224 Bath Road
GB-Slough, SL1 4 DS, Bucks
Tel: Slough (0753) 33341
Cable: HEWPIE Slough
Telex: 848413

Hewlett-Packard Ltd.
The Graftons
Stamford New Road
Altrincham, Cheshire
Tel: (061) 928-9021
Telex: 668068

Hewlett-Packard Italiana S.p.A.
Via Amerigo Vesputci 2
I-20124 Milan
Tel: (2) 6251 (10 lines)
Cable: HEWPAKIT Milan
Telex: 32046

Hewlett-Packard Italiana S.p.A.
Piazza Marconi, 25
I-00144 Rome - Eur
Tel: (6) 5912544-5, 5915947
Cable: HEWPAKIT Rome
Telex: 61514

Hewlett-Packard Italiana S.p.A.
Vicolo Pastori, 3
I-35100 Padova
Tel: (49) 66 40 62
Telex: 32046 via Milan
Tel: (11) 53 82 64
Telex: 32046 via Milan

LUXEMBURG
Hewlett-Packard Benelux
S.A./N.V.
Avenue de Col-Vert, 1,
(Groenkruglaan)
B-1170 Brussels
Tel: (03 02) 72 22 40
Cable: PALOBEN Brussels
Telex: 23 494

NETHERLANDS
Hewlett-Packard Benelux/N.V.
Weerdestein 117
P.O. Box 7825
NL-Amsterdam, 1011
Tel: 020-42 77 77, 44 29 66
Cable: PALOBEN Amsterdam
Telex: 13 216 hepa nl

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

PORTUGAL
Teletra-Empresa Tecnica de Equipamentos Electronicos S.a.r.l.
Rua Rodrigo da Fonseca 103
P.O. Box 2531
P-Lisbon 1
Tel: (19) 68 60 72

Cable: TELETRA Lisbon
Telex: 1598

SPAIN
Hewlett-Packard Española, S.A.
Jerar No 8
E-Madrid 16
Tel: 458 26 00
Telex: 23515 hpe

Hewlett-Packard Española, S.A.
Milanesado 21-23
E-Barcelona 17
Tel: (3) 203 62 00
Telex: 52603 hpbe e

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

Hewlett-Packard Sverige AB
Hagakersgatan 9C
S-431 41 Mölndal
Tel: (031) 27 68 00/01
Telex: Via Bromma

SWITZERLAND
Hewlett Packard (Schweiz) AG
Zürcherstrasse 20
P.O. Box 64
CH-8952 Schlieren Zurich
Tel: (01) 98 18 21/24
Cable: HPAG CH
Telex: 53933 hpag ch
Hewlett-Packard (Schweiz) AG
9, Chemin Louis-Pictet
CH-1214 Vernier—Geneva
Tel: (022) 41 4950
Cable: HEWPAKSA Geneva
Telex: 27 333 hpssa ch

TURKEY
Telekom Engineering Bureau
Saglik Sok No: 15/1
Ayaspaza-Beyoglu
P.O. Box 437 Beyoglu
TR-Istanbul
Tel: 49 40 40

Cable: TELEATION Istanbul
Telex: 1598

UNITED KINGDOM
Hewlett-Packard Ltd.
224 Bath Road
GB-Slough, SL1 4 DS, Bucks
Tel: Slough (0753) 33341
Cable: HEWPIE Slough
Telex: 848413

Hewlett-Packard Ltd.
"The Graftons"
Stamford New Road
GB-Altrincham, Cheshire
Tel: (061) 928-9021
Telex: 668068

Hewlett-Packard Ltd.'s registered
address for V.A.T. purposes
only:
70, Finchbury Pavement
London, EC2A1SX
Registered No: 690597

SOCIALIST COUNTRIES
PLEASE CONTACT:
Hewlett-Packard Ges.m.b.H.
Handelskai 52/3
P.O. Box 7
A-1205 Vienna
Ph: (0222) 33 66 06 to 09
Cable: HEWPAKSA Vienna
Telex: 75923 hewpak a

ALL OTHER EUROPEAN
COUNTRIES CONTACT:
Hewlett-Packard S.A.
Rue du Bois-du-Lan 7
P.O. Box 85
CH-1217 Meyrin 2 Geneva
Switzerland
Tel: (022) 41 54 00
Cable: HEWPAKSA Geneva
Telex: 2 24 86

AFRICA, ASIA, AUSTRALIA

ANGOLA
Teletra-Empresa Tecnica
de Equipamentos Electronicos
SARL
Rua de Barbosa, Rodrigues,
42-1, D1
P.O. Box 6487
Luanda
Cable: TELETRA Luanda

AUSTRALIA
Hewlett-Packard Australia
Pty. Ltd.
22-26 Weir Street
Glen Iris, 3146
Victoria
Tel: 20-1371 (6 lines)
Cable: HEWPAK Melbourne
Telex: 31 024

Hewlett-Packard Australia
Pty. Ltd.
31 Bridge Street
Pymble,
New South Wales, 2073
Tel: 449 6566
Telex: 21561
Cable: HEWPAK Sydney

Hewlett-Packard Australia
Pty. Ltd.
97 Churchill Road
Prospect 5082
South Australia
Tel: 44 8151
Cable: HEWPAK Adelaide

Hewlett-Packard Australia
Pty. Ltd.
Casablanca Buildings
196 Adelaide Terrace
Perth, W.A. 6000
Tel: 25-6800
Cable: HEWPAK Perth

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

Hewlett-Packard Australia
Pty. Ltd.
2nd Floor, 49 Gregory Terrace
Brisbane, Queensland, 4000
Tel: 29 1544

CEYLON
United Electricals Ltd.
P.O. Box 681
60, Park St.
Colombo 2
Tel: 26696
Cable: HOTPOINT Colombo

CYPRUS
Kypronics
19 Gregorios & Xenopoulos Road
P.O. Box 1152
CY-Nicosia
Tel: 45628/29
Cable: KYPRONICS PANDEHIS

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
Connaught Centre
39th Floor
Connaught Road, Central
Hong Kong
Tel: 240168, 232735
Telex: HKX4766
Cable: SCHMIDTCO Hong Kong

INDIA
Blue Star Ltd.
Kasturi Buildings
Jamshedji Tata Rd.
Bombay 400 020
Tel: 29 50 21
Telex: 3751
Cable: BLUEFROST
Blue Star Ltd.
Sahas
414/2 Vir Savarkar Marg
Prabhadevi
Bombay 400 025
Tel: 45 78 87
Telex: 4093
Cable: FROSTBLUE

Blue Star Ltd.
Band Box House
Prabhadevi
Bombay 400 025
Tel: 45 73 01
Telex: 3751
Cable: BLUESTAR

Blue Star Ltd.
14/40 Civil Lines
Kampur 208 001
Tel: 6 88 82
Cable: BLUESTAR

Blue Star, Ltd.
7 Hare Street
Tel: 03-370-2281/92
Calcutta 700 001
Tel: 23-0131
Telex: 655
Cable: BLUESTAR
Blue Star Ltd.
Blue Star House,
34 Ring Road
Lajpat Nagar
New Delhi 110 024
Tel: 62 32 76
Telex: 2463
Cable: BLUESTAR

Blue Star, Ltd.
Blue Star House
11/11A Magarath Road
Bangalore 560 025
Tel: 55668
Telex: 430
Cable: BLUESTAR

Blue Star, Ltd.
1-1-117/1
Sarojini Devi Road
Secunderabad 500 003
Tel: 7 63 91, 7 73 93
Cable: BLUEFROST
Telex: 459

Blue Star, Ltd.
23/24 Second Line Beach
Madras 600 001
Tel: 23954
Telex: 379
Cable: BLUESTAR

Blue Star, Ltd.
Nathraj Mansions
2nd Floor Bistupur
Jamshedpur 831 001
Tel: 38 04
Cable: BLUESTAR
Telex: 240

INDONESIA
Bah Belon Trading Coy. N.V.
Dialah Merdeka 29
Bandung
Tel: 29 50 21
Telex: 4915; 51560
Cable: ILMU
Telex: 08-809

IRAN
Multi Corp International Ltd.
Avenue Soraya 130
P.O. Box 1212
IR-Teheran
Tel: 83 10 35-39
Cable: MULTICORP Tehran
Telex: 2893 MCI TN

ISRAEL
Electronics & Engineering
Div. of Motorola Israel Ltd.
17 Aminadav Street
Tel-Aviv
Tel: 36941 (3 lines)
Cable: BASTEL Tel-Aviv
Telex: 33569

JAPAN
Yokogawa-Hewlett-Packard Ltd.
Ohashi Building
1-59-1 Yoyogi
Shibuya-ku, Tokyo
Tel: 03-370-2281/92
Tel: 232-2024/YHP
Cable: YHPMARKET TOK 23-724
Yokogawa-Hewlett-Packard Ltd.
Nisei Ibaragi Bldg.
2-2-8 Kasuga
Ibaragi-Shi
Osaka
Tel: (0726) 23-1641
Telex: 5332-385 YHP OSAKA
Yokogawa-Hewlett-Packard Ltd.
Nakamo Building
No. 24 Kamisasaizima-cho
Nakamura-ku, Nagoya City
Tel: (052) 571-5171
Yokogawa-Hewlett-Packard Ltd.
Nitto Bldg.
2-4-2 Shinohara-Kita
Kohoku-ku
Yokohama 222
Tel: 045-432-1504
Telex: 382-3204 YHP YOK

Yokogawa-Hewlett-Packard Ltd.
Chuo Bldg.
Rm. 603 3.
2-Chome
IZUMI-CHO,
Mito, 310
Tel: 0292-25-7470

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

KOREA
American Trading Company
Korea,
I.P.O. Box 1103
Dae Kyung Bldg., 8th Floor
107 Sejong-Ro,
Chongro-Ku, Seoul
Tel: (4 lines) 73-8924-7
Cable: AMTRACO Seoul

LEBANON
Constantin E. Macridis
P.O. Box 7213
RL-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. Gonçalves, Ltd.
162, Av. D. Luis
P.O. Box 107
Lourenço Marques
Tel: 27091, 27114
Telex: 6-208 Negon Mo
Cable: NEGON

NEW ZEALAND
Hewlett-Packard (N.Z.) Ltd.
98-98 Dixon Street
P.O. Box 9443
Courtenay Place,
Wellington
Tel: 59-559
Telex: 3898
Cable: HEWPAK Wellington

Hewlett-Packard (N.Z.) Ltd.
Pakuranga Professional Centre
267 Pakuranga Highway
Box 51092
Pakuranga
Tel: 569-651
Cable: HEWPAK, Auckland

NIGERIA
The Electronics Instrumenta-
tions Ltd. (TEIL)
144 Agege Motor Rd., Mushin
P.O. Box 6645
Lagos
Cable: THEITEIL Lagos

The Electronics Instrumenta-
tions Ltd. (TEIL)
15th Floor Cocoa House
P.M.B. 5402
Ibadan
Tel: 22325
Cable: THEITEIL Ibadan

PAKISTAN
Mushko & Company, Ltd.
Osman Chambers
Abdullah Haroon Road
Karachi 3
Tel: 511027, 512927
Cable: COOPERATOR Karachi
Mushko & Company, Ltd.
38B, Satellite Town
Rawalpindi
Tel: 41924
Cable: FEMUS Rawalpindi

PHILIPPINES
Electromex, Inc.
6th Floor, Amalgamated
Development Corp. Bldg.
Ayala Avenue, Makati, Rizal
C.C.P.O. Box 1028
Makati, Rizal
Tel: 86-18-87, 87-76-77,
87-86-88, 87-18-45, 88-91-71,
83-81-12, 83-82-12
Cable: ELEMEX Manila

SINGAPORE
Mechanical & Combustion
Engineering Company Pte.,
Ltd.
10-12, Jalan Kilang
Red Hill Industrial Estate
Singapore, 3
Tel: 647151 (7 lines)
Cable: MECOMB Singapore

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

SOUTH AFRICA
Hewlett Packard South Africa
(Pty.), Ltd.
Hewlett-Packard House
Daphne Street, Wendywood,
Sandton, Transvaal 2001
Tel: 407641 (five lines)

Hewlett Packard South Africa
(Pty.), Ltd.
Breechle House
Bree Street
Cape Town
Tel: 2-6941/2/3
Cable: HEWPAK Cape Town
Telex: 0006 CT

Hewlett Packard South Africa
(Pty.), Ltd.
641 Ridge Road, Durban
P.O. Box 99
Overport, Natal
Tel: 88-6102
Telex: 567954
Cable: HEWPAK

TAIWAN
Hewlett Packard Taiwan
39 Chung Shiao West Road
Sec. 1
Overseas Insurance
Corp. Bldg. 7th Floor
Taipei
Tel: 389160, 1, 2, 375121,
Ext. 240-249
Telex: TP824 HEWPAK
Cable: HEWPAK Taipei

THAILAND
UNIMESA Co., Ltd.
Chongkinee Building
56 Suriwongse Road
Bangkok
Tel: 37956, 31300, 31307,
37540
Cable: UNIMESA Bangkok

UGANDA
Uganda Tele-Electric Co., Ltd.
P.O. Box 4449
Kampala
Tel: 52729
Cable: COMCO Kampala

VIETNAM
Peninsular Trading Inc.
P.O. Box H-3
216 Hien-Vuong
Saigon
Tel: 20-805, 93398
Cable: PENTRA, SAIGON 242

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

MEDITERRANEAN AND
MIDDLE EAST COUNTRIES
NOT SHOWN PLEASE
CONTACT:
Hewlett-Packard
Co-ordination Office for
Mediterranean and Middle
East Operations
Piazza Marconi 25
I-00144 Rome-Eur, Italy
Tel: (6) 59 40 29
Cable: HEWPAKIT Rome
Telex: 61514

OTHER AREAS NOT
LISTED, CONTACT:
Hewlett-Packard
Export Trade Company
3200 Hillview Ave.,
Palo Alto, California 94304
Tel: (415) 326-7000
(Feb. 71 493-1501)
TWK: 910-373-1267
Cable: HEWPAK Palo Alto
Telex: 034-8300, 034-8493

