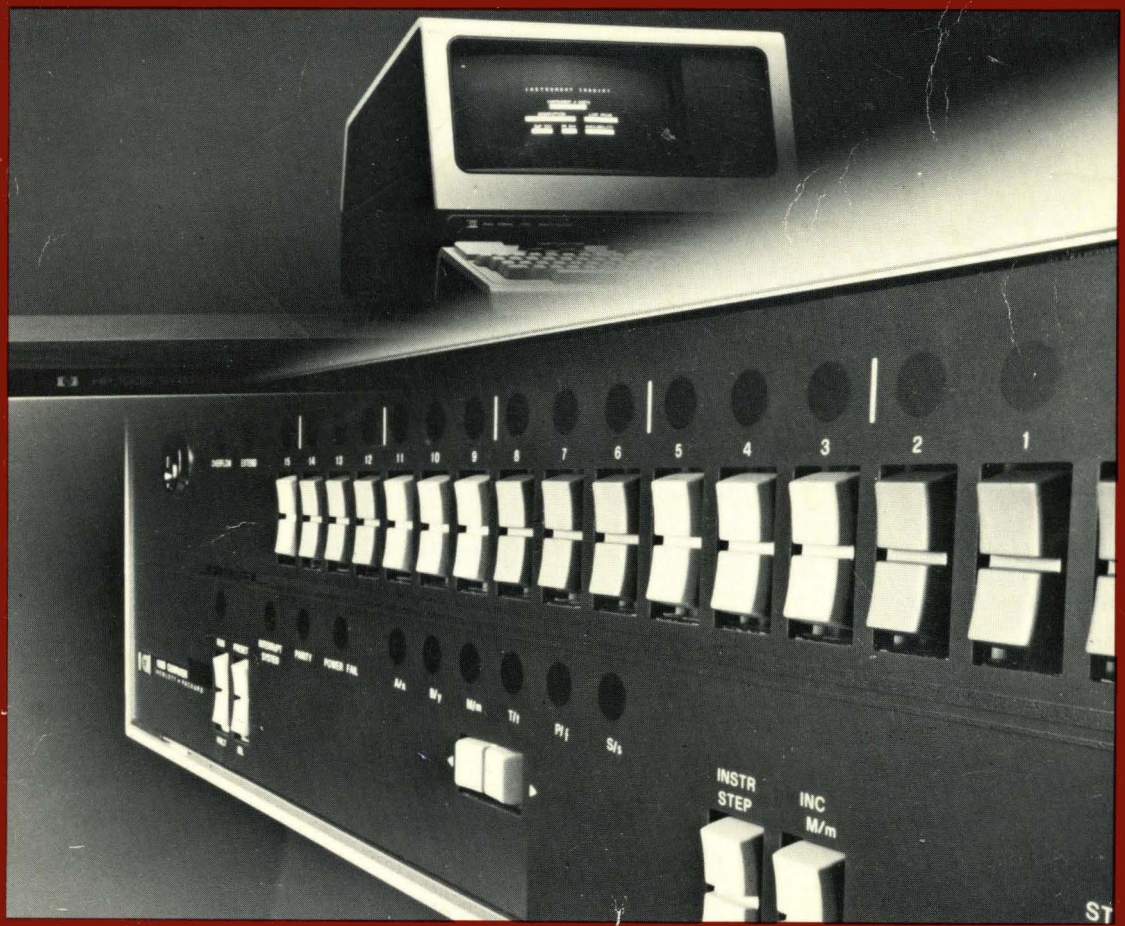
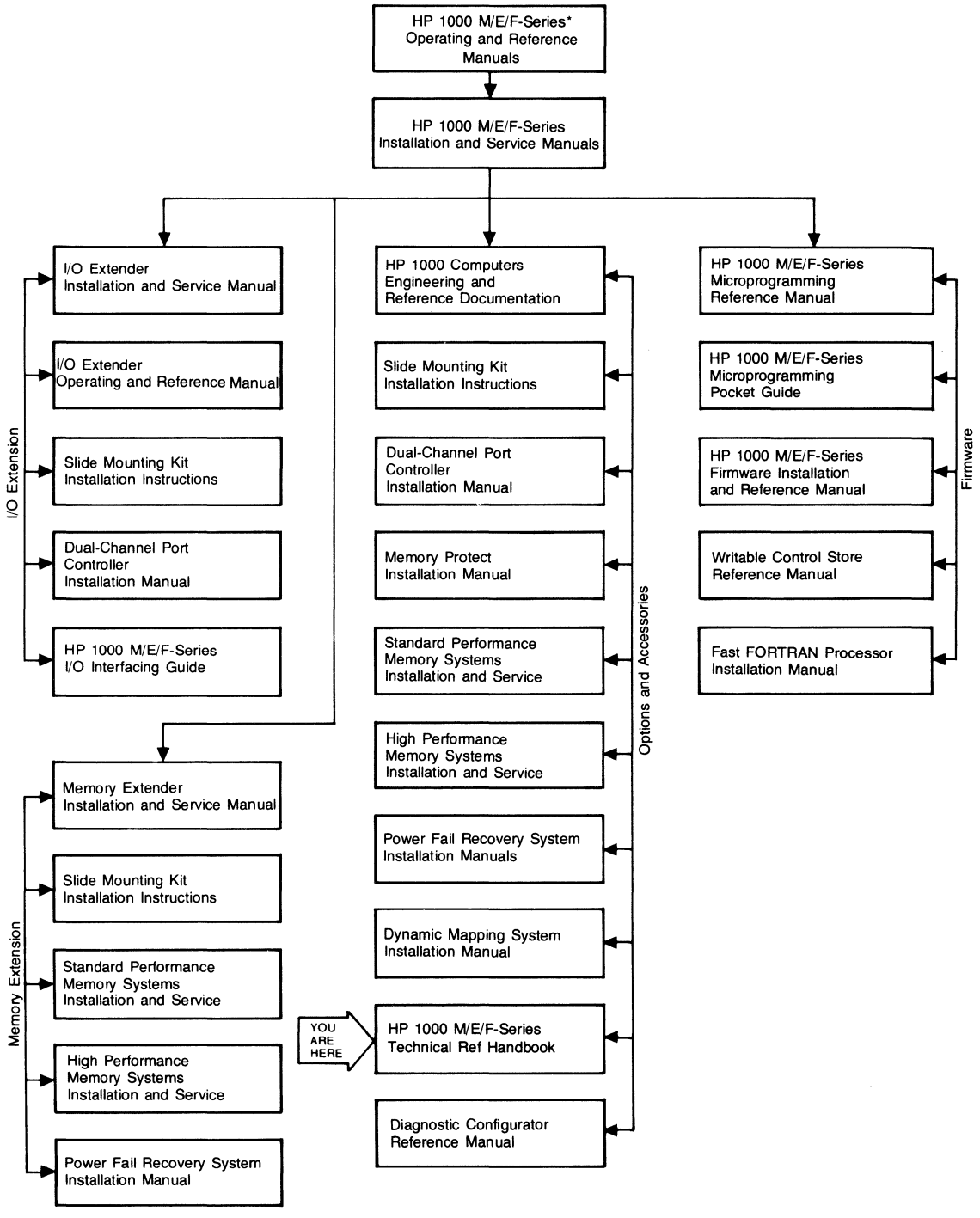
HEWLETT PACKARD

HP 1000 M/E/F-Series Computers Technical Reference Handbook



DOCUMENTATION MAP



*M/E/F refers to your particular series of HP 1000 Computer.

PREFACE

This manual combines technical references for the HP 1000 family of computers in one convenient publication. The technical descriptions will be helpful in evaluating the capability of Hewlett-Packard computers, as they include programming information and detailed descriptions of the interface to the computers' Input/Output structure.

The information in this manual is intended to be used strictly as a guide and does not necessarily represent current policies and products supported by Hewlett-Packard.

Previous manuals refer to the HP 1000 family of computers as HP 21MX. The terms HP 1000 and HP 21MX are synonymous.

Further information on Hewlett-Packard computer products is available from your local Hewlett-Packard field office, one of more than 172 Sales and Service Offices throughout the world.

Part I Reference Guide

Part II I/O Interfacing Guide

Part I

HP 1000 M/E/F-SERIES COMPUTERS
reference guide

CONTENTS

CONTENTS (Continued)

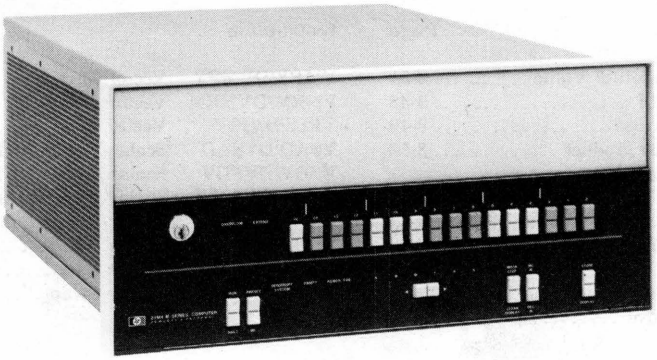
ILLUSTRATIONS

TABLES

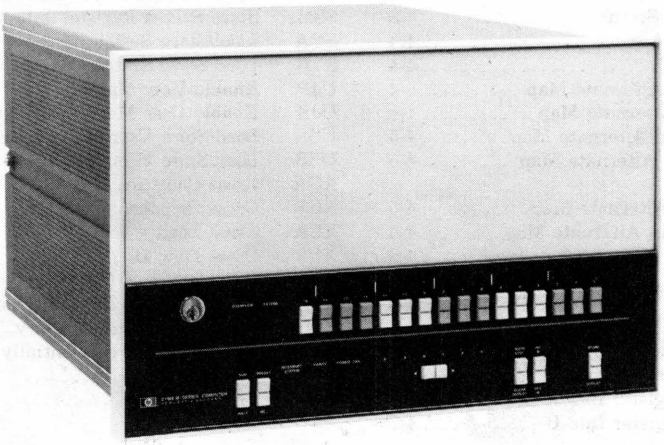
ALPHABETICAL INDEX OF STANDARD INSTRUCTIONS

ALPHABETICAL INDEX OF VECTOR INSTRUCTIONS

ALPHABETICAL INDEX OF DYNAMIC MAPPING SYSTEM INSTRUCTIONS



HP 2108B



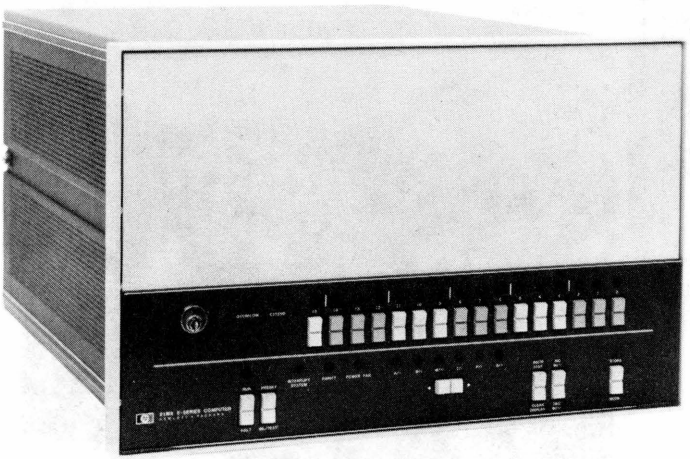
HP 2112B

7700-116A,B

Figure 1-1A. HP 1000 M-Series Microprogrammable Computers

HP 2109B

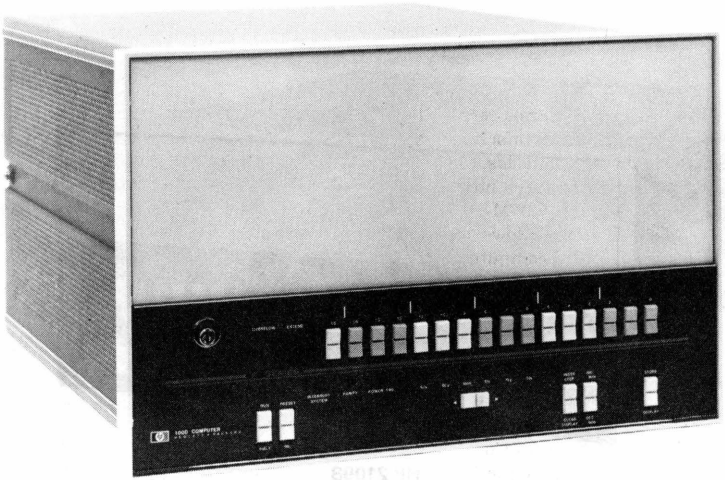
7700-82



HP 2113B

7700-83

Figure 1-1B. HP 1000 E-Series Microprogrammable Computers



7700-209

HP 2111F

7700-210

HP 2117F

Figure 1-1C. HP 1000 F-Series Microprogrammable Computers

SYSTEM FEATURES

SECTION I

The HP 1000 M-Series, E-Series and F-Series computers shown in Figure 1-1 are high-performance machines designed to satisfy a wide range of computing needs. Because of a unique design philosophy, many features have been incorporated as standard in the HP 1000 M-, E- and F-Series Computers; this same philosophy allows optional features to be added at low cost. HP 1000 computers have traditional HP quality and reliability built in from the ground up and compatibility with previous Hewlett-Packard computers is maintained. HP 1000 Computers provide very cost-effective solutions to a variety of systems applications.

Throughout this manual, the term HP 1000 is used to refer to all three models (M-, E- and F-Series Computers) when the information presented pertains equally to all three. A particular series of HP 1000 Computer is referred to only if the information presented is unique to that series.

1-1. ARCHITECTURE

HP 1000 Computers have a proven architecture that features a fully microprogrammed control processor, including all arithmetic functions, input/output and operator panel control. Four general purpose registers are available, two of which may be used as index registers. There are 128 (2108M, 2112M, 2109E and 2113E), 192 (2111F) or 230 (2117F) standard instructions including index instructions, integer and floating point arithmetic instructions, input/output instructions and a full complement of instructions for logical operations as well as bit and byte manipulation. In addition, the optional HP 12824A Vector Instruction Set provides the F-Series Computer with 38 additional instructions (19 single and 19 double precision formats).

At the heart of the HP 1000 Computer is a microprogrammable control section that directs the operations of the other functional units of the computer. Microprogramming can increase system speed in several 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 microprogrammed. With 12 scratch pad registers available to a microprogrammer, 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).

The HP 1000 E-Series and F-Series Computers have been optimized for performance. The control section has been speeded up over the M-Series Computer for certain operations due to a sophisticated technique of varying the microinstruction cycle time. In addition, the efficiency of the microprogrammed routines that determine the machine language operation has been increased. The result is a high-performance computer that retains compatibility with earlier Hewlett-Packard computers and the flexibility of user microprogramming. The central processor unit (CPU)-to-memory interface is totally asynchronous in E/F-Series Computers, thus assuring that faster memory speeds can be used to good advantage. The E/F-Series Computers are approximately twice as fast as the M-Series Computers.

A hardware-implemented Floating Point Processor (FPP) in the F-Series Computers provides high-speed dedicated logic for performing floating point arithmetic operations. The FPP provides exceptionally fast execution of single precision (32-bit), extended precision (48-bit) and double precision (64-bit) floating point operations. Single precision operations are accurate to at least 6 decimal places, extended precision operations are accurate to at least 11 decimal places and double precision operations are accurate to at least 16 decimal places. (For the HP 2111F, the FPP is housed within the computer mainframe; for the HP 2117F, the FPP is housed in a separate assembly cabled to the computer mainframe.)

The F-Series Computers include a powerful Scientific Instruction Set (SIS) that performs thirteen trigonometric, logarithmic, exponentiation and polynominal evaluation functions. These instructions utilize the Floating Point Processor as a powerful computing resource to achieve extremely high performance in conjunction with high accuracy.

For applications in which more performance is required, the HP 1000 standard instruction sets can be further expanded. Off-the-shelf enhancements include the Dynamic Mapping System (DMS, standard on HP 2112M, except option 013, HP 2113E, except options 012 and 013, and HP 2117F; optional on all other HP 1000 Computers). For expanded memory management, and the Fast FORTRAN Processor (standard with the HP 2111F and HP 2117F) that accelerates the execution of frequently-used FORTRAN operations. Optional enhancements for the M/E/F-Series include the DS/1000 Firmware for DS/1000 Network Communications and RTE-IV Extended Memory Area. The Vector Instruction Set firmware is an option available for the F-Series only.

1-2. USER MICROPROGRAMMING

The power and flexibility of microprogramming is made readily available to the HP 1000 Computer user through the microinstruction set of 180 (M-Series) or 211 (E-Series and F-Series) micro-orders. In addition to the 12 scratch pad registers and the other internal registers, the microprogrammer may address up to 4k (M-Series) or 16k (E-Series and F-Series), 24-bit words of control store. The E-Series and F-Series Computers also support up to three levels of nested subroutines in microprograms. Closely resembling assembly language programming in simplicity, microprogramming offers the advantages of speed and security as well as the ability to expand the instruction set to meet any computing need. Microprogramming is supported by Hewlett-Packard through software assembly and debug packages and customer training courses. User-developed microprograms may be permanently fused in programmable read-only memory (PROM) chips for mounting on the Firmware Extension Module or may be loaded into Writable Control Store (WCS) modules where they can be dynamically altered.

1-3. SELF-TEST ROUTINES

A comprehensive set of self-test routines permanently stored in Read-Only Memory (ROM) are standard in E-F-Series computers. Two of these routines, executed each time the IBL/TEST function is performed, tests the CPU and the enabled memory (up to 32k bytes for E-Series and up to 64k bytes for the F-Series) for quick verification of operating condition. A third test, executed every time the machine is powered up, tests the CPU and all installed memory. Three more tests, executed via the operator panel, test the Floating Point Processor, the Scientific Instruction Set, and the Vector Instruction Set firmware on the F-Series computers only.

For data integrity, memory parity checking is provided as a standard feature with all memory systems.

The fault control memory system provides fault-secure memory operation to the HP 1000 family of computers. The system consists of an HP 2102C (M-Series and E-Series) or an HP 2102H (E-Series or F-Series) Memory Controller and one or more check bit array boards, along with the appropriate number of memory modules. The system is capable of correcting all single-bit errors and of detecting all double-bit errors and most multiple-bit errors. The fault control system is particularly valuable in computer systems with large amounts of memory or where fault-secure operation is essential.

1-4. LOCAL AND REMOTE BOOTSTRAP LOADERS

The initial binary loading (IBL) function is easily performed on HP 1000 computers. For local bootstrap loading, a 64-word ROM-resident IBL program is called by pushbutton switch on the operator panel. A paper tape loader ROM and a disc loader ROM are standard. One or two additional loader ROM's may be included; these programs may be purchased as accessories or may be user-generated.

E-Series and F-Series computers at remote sites can be force-loaded from a central location through the use of the Remote Program Load (RPL) capability. All of the information normally keyed into the operator panel is set in a special set of switches on the CPU board so that the bootstrapping sequence may be initiated from a remote site.

1-5. POWER SYSTEM

HP 1000 computers are equipped with power systems designed to continue normal operations in environments where power may fluctuate widely. Input line voltages and frequencies may vary widely without affecting the operation of the computer. The optional Power Fail Recovery System provides automatic restart capability and, depending on the memory size, also provides between 1.75 and 4.25 hours of memory sustaining power in the event of complete power failure. (See Power Fail Recovery System specifications in Table 1-1).

1-6. MEMORY SYSTEMS

HP 1000 Computers are available with either of two semiconductor memory systems based on 16k-bit MOS/RAM chips that offer field-proven reliability and economy. The memory systems broken down by series, consist of the following:

M-Series — An HP 2102B Memory Controller and one or more memory modules, ranging in capacity from 64k to 128k bytes. The memory system has a system cycle time of 640 nanoseconds.

E-Series — The Standard Performance Memory System consists of an HP 2102B Memory Controller, one or more memory modules, ranging in capacity from 64k to 128k bytes and a system cycle time of 595 nanoseconds. The High Performance Memory System consists of an HP 2102E Memory Controller, one or more memory modules, ranging in capacity from 64k to 128k bytes and a memory cycle time of 350 nanoseconds.

F-Series — An HP 2102E High Performance Memory Controller and either 64k bytes (2111F) or 128k bytes (2117F) of memory.

Addressing physical memory configurations larger than 64k byte configurations is possible only through the use of the HP 12976B (M-Series) or HP 13305A (E-Series and F-Series) Dynamic Mapping System (DMS). The DMS, which is a combination of hardware and firmware, is a powerful memory management scheme that allows HP 1000 Computer users to address up to two million bytes of memory and provides read and/or write protection of each individual 2048 byte page. Four independent memory maps are provided, one for the system, one for the user and two Port Controller maps for direct memory access operations. Control of the DMS is implemented through the use of 38 instructions.

For HP 1000 Computers (except the 2105A) to which the Dynamic Mapping System has been added, memory can be expanded further with the HP 12990B Memory Extender. The extender has space for nine additional semiconductor memory modules. As much as 1152k bytes of memory can be added, depending on the memory modules used.

1-7. INPUT/OUTPUT

The input/output system for the HP 1000 computers features a multilevel vectored priority interrupt structure. There are 60 distinct interrupt levels, each of which has a unique priority assignment. Any I/O device can be selectively enabled or disabled, or the entire interrupt system (except power fail and parity error interrupts) can be enabled or disabled under program control.

Data transfer between the computer and I/O devices may take place under program control, Dual Channel Port Controller (DCPC) control, or under microprogram control. The DCPC provides two direct links between memory and I/O devices and is program assignable to any two devices. DCPC transfers occur on an I/O cycle-stealing basis not subject to the I/O priority interrupt structure. Refer to the Direct Memory Access specifications in Table 1-1 for the output transfer rates and the DCPC latency times. For applications where higher transfer rates are desirable, E-Series computers have a special microprogrammed I/O capability that will allow transfer rates of up to 3.18 million bytes per second. This capability can be attained by making simple modifications to the interface hardware and by writing a block I/O control microprogram.

The HP 2108M, HP 2109E and HP 2111F Computers have nine I/O channels in their mainframes; the HP 2112M, HP 2113E and HP 2117F computers have fourteen. The number of available channels may be increased by adding one or two HP 12979B I/O entenders, providing sixteen channels each. All I/O channels are fully buffered and bidirectional. Because of the modular design of the HP 1000 Computers, mainframe memory capacity is completely independent of I/O capacity so that either memory or I/O modules may be added without taking valuable mainframe space from the other. A full line of I/O interface controllers is available with HP 1000 Computers for interfacing to any of the broad line of HP manufactured peripherals or to specialized devices.

1-8. SOFTWARE

The HP 1000 Computers maintain extensive program compatibility with earlier HP 1000 and HP 2100 series computers so that users can take advantage of many man-years of software development.

A wide range of operating system software is available. Real-time executive (RTE) systems, available in disc and main memory-resident versions, are multiprogramming systems that permit priority scheduling of several real-time programs while concurrent background processing takes place.

The memory-based RTE-M and disc-based RTE-IV operating systems can support up to 2.048 megabytes of memory, managed by DMS. Comprehensive software is also available for computer networking.

Languages supported by Hewlett-Packard operating systems include two high-level compilers: HP FORTRAN IV; and HP BASIC; plus an extended, efficient assembler that is callable by FORTRAN. Utility software includes a debugging routine, editor, and an extensive library of commonly used computational routines.

HP 1000 Computer users may also take advantage of a wide variety of thoroughly tested and documented programs that have been contributed to the Hewlett-Packard User Library.

1-9. SPECIFICATIONS

Tables 1-1A, 1-1B and 1-1C list the specifications for the HP 1000 M-Series, E-Series and F-Series computers, respectively. All HP 1000 computers have been product accepted by the Underwriters' Laboratories (UL) and the Canadian Standards Association (CSA).

1-10. SYSTEM EXPANSION AND ENHANCEMENT

Tables 1-2A, 1-2B and 1-2C list the options and accessories available to expand or enhance the HP 1000 M-Series, E-Series and F-Series computer systems, respectively.

Table 1-1A. M-Series Specifications

CENTRAL PROCESSOR

Address Space:	4,096 bytes (direct addressing) 65,536 bytes (indirect addressing) 2,097,152 bytes with Dynamic Mapping System (optional)
Word Size:	16 bits
Instruction Set:	128 standard instructions
Memory Reference:	14
Register Reference:	43
Input/Output:	13
Extended Arithmetic:	10
Index:	32
Bit Byte, Word Manipulation:	10
Floating Point:	6
Registers:	10
Accumulators:	Two (A and B), 16 bits each. Explicitly addressable; also addressable as memory locations.
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
Display:	One, 16 bits

CONTROL PROCESSOR

Address Space:	4,096 words (16 modules of 256 words each)
Word Size:	24 bits
Word Formats:	Four
Word Fields:	Five
Instruction Execution Time:	325 ns
Micro-Orders	180
Operations:	15
Special:	32
ALU and Conditional:	68
Store:	32
S-Bus:	32
Reverse Skip Sense:	1

INITIAL BINARY LOADERS ROM resident; capacity of four 64-word programs callable from operator panel.

INPUT/OUTPUT

Interrupt Structure: Multilevel vectored priority interrupt; priority determined by interrupt location.

I/O SYSTEM SIZE	HP 2108B	HP 2112B
Standard I/O Channels	9	14
With One Extender	25	30
With Two Extenders	41	46

Compatibility: Instruction set and program compatible with HP 1000 E-Series computers (time loop programs excepted).

Table 1-1A. M-Series Specifications (Continued)

Current Available for: I/O, Memory and Accessories

MODEL	+5V	+12V	−12V	−2V
2108B	38.8A	2.5A	2.0A	4.0A
2112B	38.8A	2.5A	2.0A	4.0A

DC Required:

MODEL	+5V	−2V
12892B Memory protect	1.25A	.05A
12897B DCPC	2.4A	.05A
12731A MEM	3.9A	—
12976B DMS	6.29A	—
12977B FFP	1.66A	—
12778B DMI	1.66A	—
13197A 1k WCS	2.2A	.01A
12990B Memory extender	0	—
12992 Loader ROMS (each)	.13A	—
12979B I/O Ext. buffer card	2.0A	1.35A
2102B Memory controller	1.2A	0.01A
2102C Fault Memory controller	3.29A	—
12779A Check Bit Array	0.52A	—
12780A Check Bit Array	0.73A	—
12747A 128k Byte Memory Module	0.85A	—
13047A 2k UCS	7.39A[1]	—
12945A .5k UCS	2.2A[2]	—
12791A Firmware Expansion Module	5.4A	—

[1] *1.15A + 0.78A for each 256 instructions; 7.39A when fully loaded.*
[2] *1.2A* + 0.525A for each of 8 blocks of instruction words; 5.4A when fully loaded.

DIRECT MEMORY ACCESS Available only with DCPC accessory.

Number of Channels: Two

Word Size: 16 bits

Maximum Transfer Block Size: 32,768 words

I/O Assignable: Assignable to any two I/O channels; all logic necessary to facilitate bidirectional direct memory transfer to and from I/O is contained on DCPC (controller).

Transfer Rate:
(Any Memory) 1.23 Mbytes/s

DCPC Latency (Channel 1): Latency is defined as the time interval between the generation of a Service Request (SRQ) signal by an I/O device through the initiation of a DCPC channel 1 cycle to the completion of the I/O data transfer to or from the I/O interface PCA. Subsequent consecutive cycles execute at a specified DCPC rate.

Input and Output Latency Times:
(Times in us)

	TYPICAL	MAXIMUM
Input	2.22	2.93
Output	2.54	3.25

Table 1-1A. M-Series Specifications (Continued)

PHYSICAL CHARACTERISTICS

Width: 42.6cm (16 3/4 in) behind rack mount; 48.3cm (19 in) front panel width on sides

Depth: 59.7cm (23 1/2 in); 58.4cm (23 in) behind rack mounting ears

MODEL	HP 2108B	HP 2112B
Height	22.2cm (8-3/4 in)	31.1cm (12-1/4 in)
Weight	20.4kg (45 lbs)	29.5kg (65 lbs)

ELECTRICAL CHARACTERISTICS

Line Voltage: 88 to 132Vac; 176 to 264 Vac

Line Frequency: 47.5 to 66 Hz

Power Dissipation: 770 watts (maximum).

Power Supply: Sustains computer over a line loss of no greater than 8 ms at the nominal line (mains) voltage.

Input Line Transients: Sustains ±500V, 50μs pulse on power lines; sustains ±1kV, 100ns pulses on power lines.

Output Protection: All regulated voltages protected from overvoltage and overcurrent conditions.

Output Voltage Regulation: ±5% (except −2V is ±10%, and +30V is unregulated).

Thermal Sensing: Monitors internal temperature and automatically shuts down when computer temperature exceeds specified maximum operating temperature. Resets automatically when temperature returns to below specified maximum operating temperature.

ENVIRONMENTAL LIMITATIONS

Operating Temperature: 0° to 55°C (+32° to 131°F)

Storage Temperature: −40° to 75°C (−40° to 167°F)

Relative Humidity: 20% to 95% at 40°C (104°F), non-condensating

Ventilation and Heat Dissipation: Intake: left-hand side; Exhaust: right-hand side.

	MODEL	2108B	2112B
Heat dissipation	KCal/hr. max. BTU/hr. max.	538 2138	538 2138
Air flow	cubic meters /min. cubic feet /min.	5.7 200	7.9 280

Altitude: Transportable to 15 300m (50 000 ft) in non-operating condition and 4500m (15,000 ft) for operation

Vibration and Shock: **Vibration:** 0.30mm (0.012 in) p-p, 10-55 Hz, 3 axis

Shock: 30g, 11 Ms, 1/2 sine, 3 axis

Contact factory for review of any application requiring operation under continuous vibration.

Table 1-1A. M-Series Specifications (Continued)

MEMORY SYSTEMS

Type:	16k N-channel MOS semiconductor RAM.
Word Size:	16 bits plus parity bit
Configuration:	Controller plus multiple plug-in memory modules. Available in 32k and 128k byte modules.
Page Size:	2,048 bytes
Address Space:	65,536 bytes without DMS; 2,097,152 bytes with DMS (2108B and 2112B)
System Cycle Time:	650ns
Volatility Protection:	Sustaining power for line loss of no greater than 8ms at the minimum line (mains) voltage. Power fail recovery system is optional.
Parity Error Detection:	Monitors all words read from memory. Switch selectable for either halt or ignore interrupt error when detected. With memory protect or DMS accessory, interrupt on parity error occurs.

POWER FAIL

Interrupt Priority:	Highest priority interrupt.
Power Failure:	Detects power failure and generates an interrupt to user-written power-failure routine. A minimum of 500 μs is available for the routine.
Power Fail Recovery System:	Available as an accessory. (HP 12944B or HP 12991B).
Power Restart:	Detects resumption of power and generates an interrupt to user-written automatic restart program which has been protected in memory by the sustaining battery.
Power Control and Charge Unit:	Monitors battery charge status and provides 2A maximum charging current.
Sustaining Battery:	Type: 14 volt, 5 ampere-hours, sealed lead acid Charging rate: 2A, maximum Capacity:HP 12944B and HP 12991B will sustain memory for the period of time shown in the graph below.

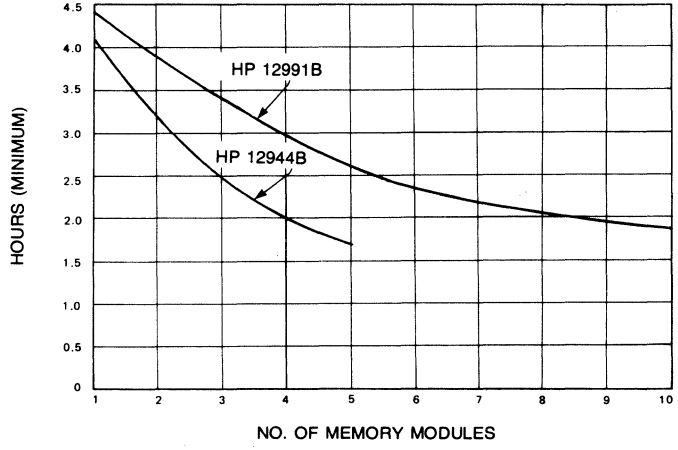


Table 1-1B. E-Series Specifications

CENTRAL PROCESSOR	
Address Space:	4,096 bytes (direct addressing) 65,536 bytes (indirect addressing) 2,097,152 bytes with Dynamic Mapping System (optional)
Word Size:	16 bits
Instruction Set:	128 standard instructions
Memory Reference:	14
Register Reference:	43
Input/Output:	13
Extended Arithmetic:	10
Index:	32
Bit Byte, Word Manipulation:	10
Floating Point:	6
Registers:	10
Accumulators:	Two (A and B), 16 bits each. Explicitly addressable; also addressable as memory locations.
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
Display:	One, 16 bits
CONTROL PROCESSOR	
Address Space:	16,384 words (64 modules of 256 words each)
Word Size:	24 bits
Word Formats:	Four
Word Fields:	Five
Instruction Execution Time:	Variable; 175 nS or 280 nS
Micro-Orders (Microinstructions):	211
Operations:	15
Special:	32
ALU and Conditional:	68
Store:	32
S-Bus:	32
Reverse Skip Sense:	32
INITIAL BINARY LOADERS	ROM resident; capacity of four 64-word programs callable from operator panel. Computer can be configured for forced cold loading from a remote site.
SELF-TEST	Automatic tests of CPU and memory operating condition. Executed on cold power-up and whenever operator panel IBL/TEST switch is pressed.

Table 1-1B. E-Series Specifications (Continued)

INPUT/OUTPUT

Interrupt Structure: Multilevel vectored priority interrupt; priority determined by interrupt location.

I/O SYSTEM SIZE	HP 2109B	HP 2113B
Standard I/O Channels	9	14
With One Extender	25	30
With Two Extenders	41	46

Compatibility: Instruction set and program compatible with HP 1000 M-Series computers (time loop programs excepted).

Current Available for: I/O, Memory and Accessories

MODEL	+5V	+12V	−12V	−2V
2109B	38.8A	2.5A	2.0A	4.0A
2113B	38.8A	2.5A	2.0A	4.0A

DC Required:

MODEL	+5V	−2V
12892B Memory protect	1.25A	.05A
12897B DCPC	2.4A	.05A
12731A MEM	3.9A	—
13307A DMS	.78A	—
13306A FFP	1.8A	—
13197A 1k WCS	2.2A	.01A
12990B Memory extender	0	—
12992 Loader ROMS (each)	.13A	—
12979B I/O Ext. buffer card	2.0A	1.35A
2102B Memory Controller	1.2A	—
2102C Fault Memory controller	3.29A	—
12779A Check Bit Array	0.52A	—
12780A Check Bit Array	0.73A	—
12747A 128k Byte Memory Module	0.85A	—
2102E Memory Controller	2.56A	—
13047A 2k UCS	7.39A[1]	—
13304A FAB	1.8A[2]	—
12791A Firmware Expansion Module	5.4[3]	—

[1]1.15A + 0.78A for each 256 instructions; 7.39A when fully loaded.
[2]Due to power saver circuit, this is the maximum current regardless of amount of microcode loaded on the board.
[3]1.2A + 0.525A for each 8 blocks of instruction words; 5.4A when fully loaded.

DIRECT MEMORY ACCESS Available only with DCPC accessory.

Number of Channels: Two

Word Size: 16 bits

Maximum Transfer Block Size: 32,768 words

I/O Assignable: Assignable to any two I/O channels; all logic necessary to facilitate bidirectional direct memory transfer to and from I/O is contained on DCPC (controller).

Table 1-1B. E-Series Specifications (Continued)

DIRECT MEMORY ACCESS (Continued)

Transfer Rate:
(All rates in Mbytes/s)

(Standard Performance Memory)

HP 2102B	Minimum	Typical	Maximum
Input w/DMS	1.884	1.950	2.098
w/o DMS			
Output w/DMS	1.626	1.676	1.782
w/o DMS	1.672	1.778	1.938

(Fault Control Memory)

2102C			
Input w/DMS	1.946	1.018	2.096
w/o DMS			
Output w/DMS	1.586	1.626	1.726
w/o DMS	1.670	1.724	1.780

(High Performance Memory)

HP 2102E			
Input w/DMS	2.282	2.284	2.284
w/o DMS			
Output w/DMS	2.038	2.114	2.196
w/o DMS	2.196	2.284	2.284

DCPC Latency (Channel 1): Latency is defined as the time interval between the generation of a Service Request (SRQ) signal by an I/O device through the initiation of a DCPC channel 1 cycle to the completion of the I/O data transfer to or from the I/O interface PCA. Subsequent consecutive cycles execute at a specified DCPC rate.

Input and Output Latency Times:
(Times in us)

(Standard Performance Memory)

HP 2102B	Minimum	Typical	Maximum
Input w/o DMS	2.73	2.91	3.15
w/DMS	2.84	3.12	3.26
Output w/o DMS	3.43	3.64	3.95
w/DMS	3.57	3.92	4.10

(Fault Control Memory)

HP 2102C			
Input w/o DMS	2.98	3.12	3.26
w/DMS	3.09	3.33	3.46
Output w/o DMS	3.75	3.93	4.11
w/DMS	3.89	4.21	4.36

(High Performance Memory)

HP 2102E			
Input w/o DMS	2.21	2.21	2.28
w/DMS	2.24	2.28	2.35
Output w/o DMS	2.75	2.75	2.84
w/DMS	2.80	2.87	2.98

Table 1-1B. E-Series Specifications (Continued)

DIRECT MEMORY ACCESS (Continued)

A USER MICROCODE sequence of seven consecutive reads may provide conditions to produce the absolute worst case latency time. The absolute worst case latency times are as follows:

Memory System	Input	Output
HP 2102B	4.095 us	4.935 us
HP 2102C	5.125 us	6.031 us
HP 2102E	3.050 us	3.681 us

PHYSICAL CHARACTERISTICS

Width: 42.6cm (16 3/4 in) behind rack mount; 48.3cm (19 in) front panel width on sides

Depth: 59.7cm (23 1/2 in); 58.4cm (23 in) behind rack mounting ears

MODEL	HP 2109B	HP 2113B
Height	22.2cm (8-3/4 in)	31.1cm (17-1/4 in)
Weight	20.4kg (45 lbs)	29.5kg (65 lbs)

ELECTRICAL CHARACTERISTICS

Line Voltage: 88 to 132V; 176 to 264V

Line Frequency: 47.5 to 66 Hz

Power Dissipation: 770 watts (maximum).

Power Supply: Sustains computer over a line loss of no greater than 8 ms at the nominal line (mains) voltage.

Input Line Transients: Sustains ±500V, 50μs pulse on power lines; sustains ±1kV, 100ns pulses on power lines.

Output Protection: All regulated voltages protected from overvoltage and overcurrent conditions.

Output Voltage Regulation: ±5% (except −2V is ±10%, and +30V is unregulated).

Thermal Sensing: Monitors internal temperature and automatically shuts down when computer temperature exceeds operating temperature. Resets automatically when temperature returns to operating temperature.

ENVIRONMENTAL LIMITATIONS

Operating Temperature: 0° to 55°C (+32° to 131°F)

Storage Temperature: −40° to 75°C (−40° to 167°F)

Relative Humidity: 20% to 95% at 40°C (104°F), non-condensating

Table 1-1B. E-Series Specifications (Continued)

ENVIRONMENTAL LIMITATIONS (Continued)

Ventilation and Heat Dissipation: Intake: left-hand side; Exhaust: right-hand side.

	MODEL	2109B	2113B
Heat dissipation	KCal/hr. max.	538	538
	BTU/hr. max.	2138	2138
Air flow	cubic meters /min.	6.63	9.23
	cubic feet /min.	200	280

Altitude: Transportable to 15,300m (50,000 ft) in non-operating condition and 4570m (15,000 ft) for operation

Vibration and Shock: **Vibration:** .30mm (0.12 in) p-p, 10-55 Hz, 3 axis
Shock: 30g, 11 Ms, 1/2 sine, 3 axis
Contact factory for review of any application requiring operation under continuous vibration.

MEMORY SYSTEMS

Type: 16k N-channel MOS semiconductor RAM.

Word Size: 16 bits plus parity bit

Configuration: Controller plus multiple plug-in memory modules. Available in 32k and 128k byte modules.

Page Size: 2,048 bytes

Address Space: 65,536 bytes without DMS; 1,835,008 bytes with DMS (2109B and 2113B)

System Cycle Times:
(All cycle times in ns)

Cycle	Minimum	Typical	Maximum
(Standard Performance Memory)			
READ w/o DMS	560	595	665
READ w/DMS	595	665	700
WRITE	595	665	700
REFRESH	595	665	700
(Fault Control Memory)			
READ w/o DMS	595	630	665
READ w/DMS	630	700	730
WRITE	595	630	665
REFRESH	595	630	665
(High Performance Memory)			
READ w/o DMS	350	350	385
READ w/DMS	385	420	455
WRITE	350	350	385
REFRESH	350	350	385

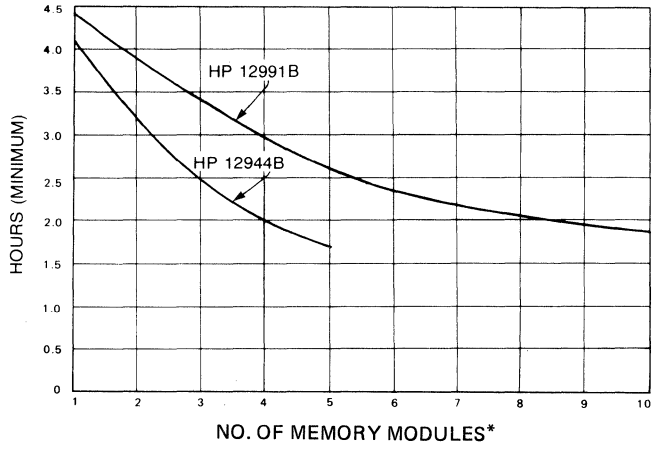
Table 1-1B. E-Series Specifications (Continued)

MEMORY SYSTEMS (Continued)

Volatility Protection:	Sustaining power for line loss of no greater than 8ms at the minimum line (mains) voltage. Power fail recovery system is optional.
Parity Error Detection:	Monitors all words read from memory. Switch selectable for either halt or ignore interrupt error when detected. With memory protect or DMS accessory, interrupt on parity error occurs.

POWER FAIL

Interrupt Priority:	Highest priority interrupt.
Power Failure:	Detects power failure and generates an interrupt to user-written power-failure routine. A minimum of 500 μs is available for the routine.
Power Fail Recovery System:	Available as an accessory. (HP 12944B or HP 12991B).
Power Restart:	Detects resumption of power and generates an interrupt to user-written automatic restart program which has been protected in memory by the sustaining battery.
Power Control and Charge Unit:	Monitors battery charge status and provides 2A maximum charging current.
Sustaining Battery:	Type: 14 volt, 5 ampere-hours, sealed lead acid Charging rate: 2A, maximum Capacity: HP 12944B and HP 12991B will sustain memory for the period of time shown in the graph below.



*INCLUDES MEMORY BOARDS AND FAULT CONTROL CHECK BIT ARRAY BOARDS.

Table 1-1C. F-Series Specifications

HARDWARE SUPPLIED	
Computer Mainframe:	HP 2111F Computer mainframe HP 2117F Computer mainframe and Floating Point Processor assembly
Memory Controller:	HP 2102E
Memory Module:	HP 2111F: Two HP 12741A modules HP 2117F: HP 12747H module
Dynamic Mapping System:	HP 13305A (HP 2117F)
CENTRAL PROCESSOR	
Address Space:	4,096 bytes (direct addressing) 65,536 bytes (indirect addressing) 2,097,152 bytes with Dynamic Mapping System
Word Size:	16 bits
Instruction Set:	192 standard instructions (HP 2111F); 230 standard instructions (HP 2117F)
Memory Reference:	14
Register Reference:	43
Input/Output:	13
Extended Arithmetic:	10
Index:	32
Bit Byte, Word Manipulation:	10
Floating Point:	24
Scientific:	13
Fast FORTRAN:	21
Double Integer:	12
Dynamic Mapping:	38 (HP 2117F)
Vector (Optional):	38 (19 single and 19 double precision formats)
Registers:	10
Accumulators:	Two (A and B), 16 bits each. Explicitly addressable; also addressable as memory locations.
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
Display:	One, 16 bits
CONTROL PROCESSOR	
Address Space:	16,384 words (64 modules of 256 words each)
Word Size:	24 bits
Word Formats:	Four
Word Fields:	Five
Instruction Execution Time:	Variable; 175 nS or 280 nS
Micro-Orders	211
Operations:	15
Special:	32
ALU and Conditional:	68
Store:	32
S-Bus:	32
Reverse Skip Sense:	32

Table 1-1C. F-Series Specifications

SCIENTIFIC INSTRUCTION SET

Data Format: Single precision (32-bit) floating point

Accuracy: RMS relative error for the various Scientific Instruction Set functions is as follows:

Function	RMS Rel. Error	Function	RMS Rel. Error
SIN	8.80E-8	SQRT	6.74E-8
COS	8.82E-8	EXP	1.38E-7
TAN	1.99E-7	ALOG	1.28E-7
ATAN	1.34E-7	ALOGT	1.39E-7
TANH	1.33E-7		

INITIAL BINARY LOADERS ROM resident; capacity of four 64-word programs callable from operator panel. Computer can be configured for forced cold loading from a remote site.

SELF-TEST Automatic tests of CPU and memory operating condition. Executed on cold power-up and whenever operator panel IBL/TEST switch is pressed.

INPUT/OUTPUT

Interrupt Structure: Multilevel vectored priority interrupt; priority determined by interrupt location.

I/O SYSTEM SIZE	HP 2111F	HP 2117F
Standard I/O Channels	9	14
With One Extender	25	30
With Two Extenders	41	46

Compatibility: Instruction set: The HP 1000 F-Series instruction set is backwards compatible with all previous 2100 Series computers.

Program: All programs written for 2100 Series computers are compatible with the F-Series, except those with timing loop dependence.

Current Available for: I/O, Memory and Accessories

MODEL	+5V	+12V	−12V	−2V
2111F	21.9A	2.5A	2.0A	6.0A
2117F	28.8A	2.5A	2.0A	6.0A

Table 1-1C. F-Series Specifications

INPUT/OUTPUT (Continued)

DC Required:

MODEL	+5V	−2V
12892B Memory protect	1.25A	.05A
12897B DCPC	2.4A	.05A
12731A MEM	3.9A	—
13307A DMI	.78A	—
13197A 1k WCS	2.2A	.01A
12990B Memory extender	0	—
12992 Loader ROMS (each)	.13A	—
12979B I/O Ext. buffer card	2.0A	1.35A
2102E Memory Controller	2.56A	—
12747H 128k Byte Memory Module	.48A	—
2102H Fault Memory Controller	3.3A	—
12779H Check Bit Array	.25A	—
12780H Check Bit Array	.3A	—
13047A 2k UCS	7.39A[1]	—
13304A FAB	1.8A[2]	—
12791A FEM	5.2[3]	—

[1]1.15A + 0.78A for each 256 instructions; 7.39A when fully loaded.
[2]Due to power saver circuit, this is the maximum current regardless of amount of microcode loaded on the board.
[3]5.2 amperes fully loaded. 1.2 amperes unloaded.

DIRECT MEMORY ACCESS Available only with DCPC accessory.

Number of Channels: Two

Word Size: 16 bits

Maximum Transfer Block Size: 32,768 words

I/O Assignable: Assignable to any two I/O channels; all logic necessary to facilitate bidirectional direct memory transfer to and from I/O is contained on DCPC (controller).

Transfer Rate:
(All rates in Mbytes/s)

(High Performance Memory)

HP 2102E	Minimum	Typical	Maximum
Input w/DMS	2.282	2.284	2.284
w/o DMS			
Output w/DMS	2.038	2.114	2.196
w/o DMS	2.196	2.284	2.284

(High Performance Fault Control Memory)

HP 2102H			
Input w/DMS	2.28	2.28	2.28
w/o DMS			
Output w/DMS	1.902	1.968	2.038
w/o DMS	2.038	2.114	2.196

DCPC Latency (Channel 1): Latency is defined as the time interval between the generation of a Service Request (SRQ) signal by an I/O device through the initiation of a DCPC channel 1 cycle to the completion of the I/O data transfer to or from the I/O interface PCA. Subsequent consecutive cycles execute at a specified DCPC rate.

Table 1-1C. F-Series Specifications

DIRECT MEMORY ACCESS (Continued)

Input and Output Latency Times:
(Times in us)

(High Performance Memory)

HP 2102E	Minimum	Typical	Maximum
Input w/o DMS	2.21	2.21	2.28
w/DMS	2.21	2.28	2.35
Output w/o DMS	2.75	2.75	2.84
w/DMS	2.80	2.87	2.98
HP 2102H			
Input w/o DMS	2.31	2.42	2.52
w/DMS	2.38	2.56	2.74
Output w/o DMS	2.84	2.94	3.05
w/DMS	2.98	3.16	3.33

(High Performance Fault Control Memory)

A USER MICROCODE sequence of seven consecutive reads may provide conditions to produce the absolute worst case latency time. The absolute worst case latency times are as follows:

Memory System	Input	Output
HP 2102E	3.050 us	3.681 us
HP 2102H	3.510 us	4.100 us

PHYSICAL CHARACTERISTICS

Width: 42.6cm (16 3/4 in) behind rack mount; 48.3cm (19 in) front panel width on sides

Depth: 62.2cm (24 1/2 in); 58.4cm (23 in) behind rack mounting ears

MODEL	HP 2111F	HP 2117F
Height	31.1cm (12-1/4 in)	44.5cm (17-1/2 in)
Weight	30kg (66 lbs)	50kg (110 lbs)

ELECTRICAL CHARACTERISTICS

Line Voltage: 2111F: 88-132V; 176-264V with option 015.
2117F: Computer mainframe same as 2111F; Floating Point Processor voltage selector offers choice of 90-110V, 108-132V, 198-242V, and 216-264V ranges.

Line Frequency: 47.5 to 66 Hz

Power Dissipation: 2111F: 770 watts (maximum). 2117F: 970 watts (maximum).

Table 1-1C. F-Series Specifications

ELECTRICAL CHARACTERISTICS (Continued)

Power Supply:	Sustains computer over a line loss of no greater than 8 ms at the nominal line (mains) voltage.
Input Line Transients:	Sustains ±500V, 50μs pulse on power lines; sustains ±1kV, 100ns pulses on power lines.
Output Protection:	All regulated voltages protected from overvoltage and overcurrent conditions.
Output Voltage Regulation:	±5% (except −2V is ±10%, and +30V is unregulated).
Thermal Sensing:	Monitors internal temperature and automatically shuts down when computer temperature exceeds specified maximum operating temperature. Resets automatically when temperature returns to below specified maximum operating temperature.

ENVIRONMENTAL LIMITATIONS

Operating Temperature:	0° to 55°C (+32° to 131°F)
Storage Temperature:	−40° to 75°C (−40° to 167°F)
Relative Humidity:	20% to 95% at 40°C (104°F), non-condensating
Ventilation and Heat Dissipation:	Intake: left-hand side; Exhaust: right-hand side.

	MODEL	2111F	2117F*
Heat dissipation	KCal/hr. max.	580	752
	BTU/hr. max.	2303	2986
Air flow	cubic meters /min.	7.9	11
	cubic feet /min.	280	390

*Includes both computer and floating point processor mainframes.

Altitude:	Transportable to 15 240m (50 000 ft) in non-operating condition and 4570m (15 000 ft) for operation
Vibration and Shock:	Vibration: 0.30mm (0.012 in) p-p, 10-55 Hz, 3 axis Shock: 30g, 11 Ms, 1/2 sine, 3 axis Contact factory for review of any application requiring operation under continuous vibration.

MEMORY SYSTEMS

Type:	16k N-channel MOS semiconductor RAM.
Word Size:	16 bits plus parity bit
Configuration:	Controller plus multiple plug-in memory modules. Available in 32k and 128k byte modules.
Page Size:	2,048 bytes
Address Space:	65,536 bytes without DMS; 2,097,152 bytes with DMS (2111F and 2117F)

Table 1-1C. F-Series Specifications

MEMORY SYSTEMS (Continued)

System Cycle Times:
(All cycle times in ns)

(High Performance Memory)

Cycle	Minimum	Typical	Maximum
READ w/o DMS	350	350	385
READ w/DMS	385	420	455
WRITE	350	350	385
REFRESH	350	350	385
READ w/o DMS	386	421	456
READ w/DMS	456	491	526
WRITE	386	421	456
REFRESH	386	421	456

(High Performance Fault Control Memory)

Volatility Protection: Sustaining power for line loss of no greater than 8ms at the minimum line (mains) voltage. Power fail recovery system is optional.

Parity Error Detection: Monitors all words read from memory. Switch selectable for either halt or ignore interrupt error when detected. With memory protect or DMS, interrupt on parity error occurs.

POWER FAIL

Interrupt Priority: Highest priority interrupt.

Power Failure: Detects power failure and generates an interrupt to user-written power-failure routine. A minimum of 500 μs is available for the routine.

Power Fail Recovery System: Available as an accessory. (HP 12991B).

Power Restart: Detects resumption of power and generates an interrupt to user-written automatic restart program which has been protected in memory by the sustaining battery.

Power Control and Charge Unit: Monitors battery charge status and provides 2A maximum charging current.

Sustaining Battery: Type: 14 volt, 5 ampere-hours, sealed lead acid
Charging rate: 2A, maximum
Capacity:HP 12991B will sustain memory for the period of time shown in the graph below.

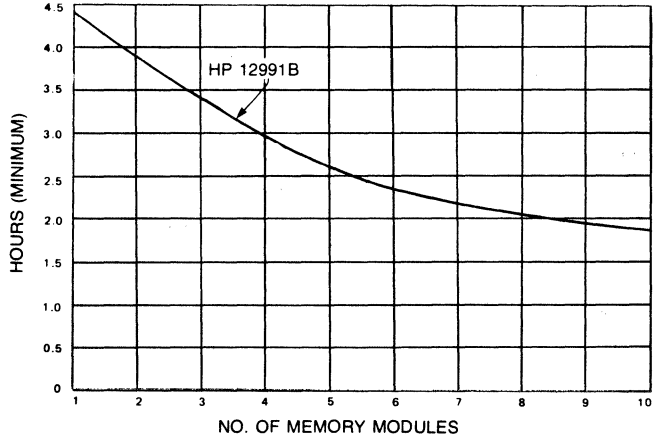


Table 1-2A. M-Series Options and Accessories

DESCRIPTION	OPTION NO.	ACCESSORY NO.
HP 2108M and HP 2112M Computers		
230V Operation	015	
Power Fail Recovery System (HP 2108B)		12944B
Power Fail Recovery System (HP 2112B or 12990B)		12991B
Memory Protect		12892B
Dual-Channel Port Controller (DCPC)		12897B
User ROM Control Store Board		12945A
Dynamic Mapping System		12976B
Fast FORTRAN Processor		12977B
1K Writable Control Store		13197A
Slide Mounting Kit (HP 2108M, HP 12979B, HP 12990B)		12903B
Slide Mounting Kit (HP 2112M)		12903C
Disc Loader ROM for HP 7900/01 or HP 2883		12992A
Disc Loader ROM for HP 7906/20/25		12992B
Loader ROM for 2644/45/48 Mini-Cartridge Tape Loader		12992C
Loader ROM for 7970 Mag Tape		12992D
Loader ROM for 9885 Flexible Disc		12992E
Disc Loader ROM for 7900/01		12992F
Terminal Loader ROM for HP 2644A/2645A/2648A		12992C
Magnetic Tape Loader ROM for HP 7970B/E		12992D
2k User Control Store Board		13047A
Firmware Expansion Module		12791A
HP 2102 MOS Memory System		
Memory Controller	015	2102B
64k Byte Memory Module		12746A
128k Byte Memory Module		12747A
Fault Control Memory Controller		2102C
256k Byte Check Bit Array Board		12779A
512k Byte Check Bit Array Board		12780A
128k Byte Standard Performance Memory Package		12784A
256k Byte Standard Performance Memory Package		12784B
512k Byte Standard Performance Memory Package		12784C
1024k Byte Standard Performance Memory Package		12784D
128k Byte Std. Perf. Fault Control Memory Package		12785A
256k Byte Std. Perf. Fault Control Memory Package		12785B
512k Byte Std. Perf. Fault Control Memory Package		12785C
1024k Byte Std. Perf. Fault Control Memory Package		12785D
Input/Output Extender		12979B
230V Operation	015	
Dual CPU Kit		12781A
Dual Channel Port Controller		12898A
Memory Extender		12990B
230V Operation	015	

Table 1-2B. E-Series Options and Accessories

DESCRIPTION	OPTION NO.	ACCESSORY NO.
HP 2109B and HP 2113B Computers		
230V Operation	015	—
Power Fail Recovery System (HP 2109E)		12944B
Power Fail Recovery System (HP 2113E or 12990B)		12991B
Memory Protect		12892B
Dual-Channel Port Controller (DCPC)		12897B
E-Series Firmware Accessory Board		13304A
E-Series Dynamic Mapping System [1,2]		13305A
E-Series Fast FORTRAN Processor [2]		13306A
1k Writable Control Store		13197A
Slide Mounting Kit (HP 2109E, HP 12979B, HP 12990B)		12903B
Slide Mounting Kit (HP 2113E)		12903C
Dynamic Mapping Instructions		13307A
Memory Expansion Module		12731A
Loader ROM for 2644A and 2645A [3]		12992C
Loader ROM for 7970B/E [3]		12992D
2k User Control Store		13047A
HP 2102 MOS Memory Systems		
Standard Performance Memory Systems		
Memory Controller		2102B
64k Byte Memory Module		12746A
128k Byte Memory Module		12747A
128k Byte Memory Package		12786A
256k Byte Memory Package		12786B
512k Byte Memory Package		12786C
1024k Byte Memory Package		12786D
Standard Performance Fault Control Memory Systems		
Fault Control Memory Controller		2102C
Check Bit Array (lower half)		12779A
Check Bit Array (full)		12780A
128k Byte Memory Package		12787A
256k Byte Memory Package		12787B
512k Byte Memory Package		12787C
1024k Byte Memory Package		12787D
High Performance Memory Systems		
Memory Controller		2102E
64k Byte Memory Module		12746H
128k Byte Memory Module		12747H
128k Byte Memory Package		12788A
256k Byte Memory Package		12788B
512k Byte Memory Package		12788C
1024k Byte Memory Package		12788D
High Performance Fault Control Memory Systems		
Memory Controller		2102H
Check Bit Array (lower half)		12779H
Check Bit Array (full)		12780H
64k Byte Memory Module		12746H
128k Byte Memory Module		12747H
128k Byte Memory Package		12789A
256k Byte Memory Package		12789B
512k Byte Memory Package		12789C
1024 Byte Memory Package		12789D
Input/Output Extender	015	12979B
230V Operation		—
DCPC for I/O Extender		12898A
Memory Extender	015	12990B
230V Operation		—

[1] *Includes Memory Protect.*
[2] *Requires E-Series Firmware Accessory Board.*
[3] *See latest Loader ROM Installation Manual for a complete listing of Loader ROMs.*

Table 1-2C. F-Series Options and Accessories

DESCRIPTION	OPTION NO.	ACCESSORY NO.
HP 2111F and HP 2117F Computers		
Delete standard 128k Byte memory system (2117F) and replace with 64k Byte memory system	−013	—
Delete standard memory when ordering alternate memory	−014	—
230V Operation	−015	—
Power Fail Recovery System (HP 2111F, HP 2117F HP 12990B)		12991B
Memory Protect		12892B
Dual-Channel Port Controller (DCPC)		12897B
Dynamic Mapping System[1]		13305A
1K Writable Control Store		13197A
Slide Mounting Kit (HP 12979B, HP 12990B)		12903B
Slide Mounting Kit (HP 2111F)		12903C
Dynamic Mapping Instructions		13307A
Memory Expansion Module		12731A
Firmware Expansion Module		12791A
Vector Instruction Set		12824A
Loader ROM for 2644A and 2645[2]		12992C
Loader ROM for 7970B/E[2]		12992D
2k User Control Store		13047A
HP 2102 MOS Memory Systems		
High Performance Memory System		—
Memory Controller		2102E
64k Byte Memory Module		12746H
128k Byte Memory Module		12747H
128k Byte Memory Package		12788A
256k Byte Memory Package		12788B
512k Byte Memory Package		12788C
1024k Byte Memory Package		12788D
High Performance Fault Control Memory System		—
Memory Controller		2102H
Check Bit Array (lower half)		12779H
Check Bit Array (full)		12780H
64k Byte Memory Module		12746H
128 Byte Memory Module		12747H
128k Byte Memory Package		12789A
256k Byte Memory Package		12789B
512k Byte Memory Package		12789C
1024k Byte Memory Package		12789D
Input/Output Extender		12979B
230V Operation	−015	—
DCPC for I/O Extender		12898A
Memory Extender		12990B
230V Operation	−015	—

[1]Includes Memory Protect.
[2]See latest Loader ROM Installation Manual for a complete listing of Loader ROMs.

OPERATING PROCEDURES

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 an input device, manually loading a short program via the operator panel, and running a program after it has been loaded into memory.

2-1. HARDWARE REGISTERS

The computer has eight 16-bit working registers which can be selected for display and modification by operator panel controls; two 1-bit registers; and one 16-bit display register. The functions 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.

2-7. S-REGISTER

The S-register is a 16-bit utility register. The S-register can be addressed as an input/output device (select code 01) and, in the run mode, it is displayed in the operator panel display register. Thus, the S-register may serve as a communication link between the computer and operator.

2-8. EXTEND REGISTER

The one-bit extend register is used by rotate instructions to link the A- and B-registers or to indicate a carry from the most-significant bit (bit 15) of the A- or B-register by an add instruction or an increment instruction. 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. This register can also be accessed from the operator panel by entering the special register display mode described under paragraph 2-23.

2-9. OVERFLOW REGISTER

The one-bit overflow register is used to indicate that an add instruction, divide instruction, or an increment instruction 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 by programmed instructions. The operator panel OVERFLOW indicator will remain lighted until the overflow is cleared. This register can also be accessed from the operator panel by entering the special register display mode described under paragraph 2-23.

2-10. DISPLAY REGISTER

The display register, which is included on the operator panel, provides a means of displaying and/or modifying the contents of the eight 16-bit working registers and the special 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 3-24 and 3-25, respectively. These registers can also be accessed from the operator panel by entering the special register display mode described under paragraph 2-23.

2-12. OPERATOR PANEL AND POWER SUPPLY OPERATING CONTROLS

The location and function of the various controls and indicators mounted on the operator panel and the power supply are illustrated in Figure 2-1 and described in Table 2-1. All operator panel controls are two-position, momentary-contact rocker switches; the status of the computer is displayed by light-emitting diodes.

2-13. REAR PANEL

The rear panels and the I/O PCA cages for the HP 2108M and HP 2109E computers are shown in Figure 2-2 and described in Table 2-2. The rear panel and I/O PCA cage for the HP 2111F are shown in Figure 2-3 and described in Table 2-3. The rear panel and I/O PCA cage for the HP 2112M, HP 2113E and HP 2117F are shown in Figure 2-4 and described in Table 2-4. The rear panel of the Floating Point Processor (FPP) is also shown in Figure 2-4 and described in Table 2-4.

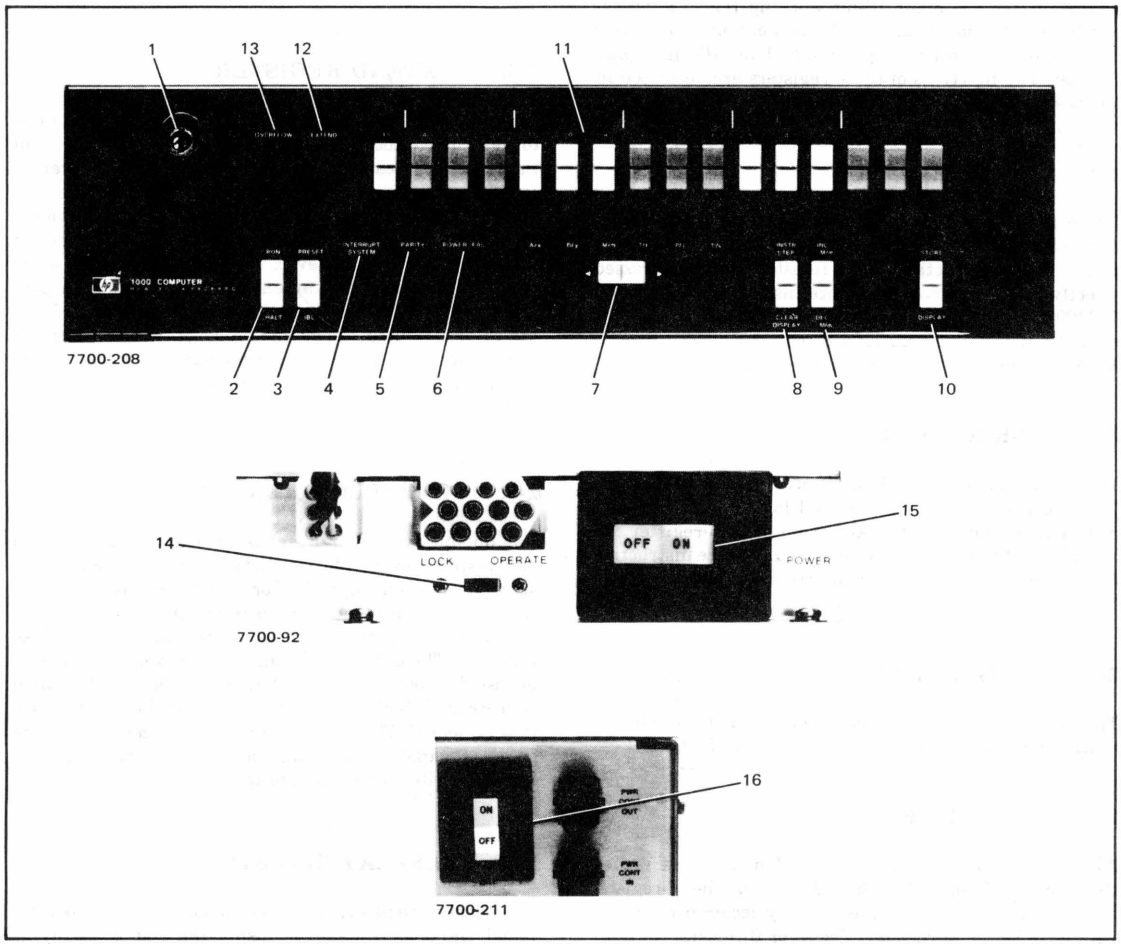


Figure 2-1. Operator Panel and Power Supply Controls and Indicators

Table 2-1. Operator Panel and Power Supply Control and Indicator Functions

<table>
<tr><th>FIG. 2-1, INDEX NO.</th><th>NAME</th><th>FUNCTION</th></tr>
<tr><td>1</td><td>Key</td><td>Secures the operator panel when access to the ~POWER OFF/ON and LOCK/OPERATE switches is not desired.</td></tr>
<tr><td>2</td><td>RUN/HALT</td><td>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.

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.</td></tr>
<tr><td>3</td><td>PRESET-IBL/TEST</td><td>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 an ARS condition (see paragraph 6-1).

IBL (initial binary loader)/TEST. Causes the contents of the selected loader ROM to be written into the uppermost 64 memory locations and the automatic execution of firmware self-tests 1 and 2. (Refer to paragraph 2-31.) Bits 15 and 14 of the S-register select the desired loader ROM as follows:
<table>
<tr><th colspan="2">BITS</th><th rowspan="2">LOADER SELECTED</th></tr>
<tr><th>15</th><th>14</th></tr>
<tr><td>0</td><td>0</td><td>Loader ROM 00_2</td></tr>
<tr><td>1</td><td>0</td><td>Loader ROM 10_2</td></tr>
<tr><td>0</td><td>1</td><td>Loader ROM 01_2</td></tr>
<tr><td>1</td><td>1</td><td>Loader ROM 11_2</td></tr>
</table>
Bits 6 through 11 of the S-register must be set to the octal select code of the loading device.</td></tr>
<tr><td>4</td><td>INTERRUPT SYSTEM</td><td>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).</td></tr>
<tr><td>5</td><td>PARITY</td><td>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 or DMS installed 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.</td></tr>
<tr><td>6</td><td>POWER FAIL</td><td>If the power fail/automatic restart feature is enabled (i.e., internal ARS/ARS switch is set to ARS position as described in Section VI) the indicator will light when power is restored. This light can be turned off by pressing the PRESET switch in the halt mode.</td></tr>
</table>

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

FIG. 2-1, INDEX NO.	NAME	FUNCTION
7	◄ Register Select ►	In the halt mode, this switch allows any one of the working registers or special registers (A, B, M, T, P, S, x, y, m, t, f, or s) to be selected for display and/or 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. Note: During standard register display mode, the "dot" indicator is the lighted indicator; i.e., the selected register (A, B, M, T, P, or S) is indicated by the lighted indicator and the remaining indicators are not lighted. During special register display mode, the "dot" indicator is the unlighted indicator; i.e., the selected register (x, y, m, t, f, or s) is indicated by the indicator not lighted and the remaining indicators are lighted. 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.
8	INSTR STEP/CLEAR DISPLAY	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. (See note for additional information.) CLEAR DISPLAY. In the run or halt mode, clears the Display Register; i.e., contents become 000000.
9	INC M/m-DEC M/m	INC M. In the halt mode, increments the M-register contents during standard display mode and the m-register contents during special display mode. (Refer to paragraph 2-25.) DEC M. In the halt mode, decrements the M-register contents during standard display mode and the m-register contents during the special display mode. (Refer to paragraph 2-25.) Note: Incrementing and decrementing occur even when the M- or m-register is not displayed.
10	STORE/MODE	STORE. In the halt and standard display modes, 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 *not* 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. (Refer to paragraph 2-23 STORE functions during special register display mode.)

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

FIG. 2-1, INDEX NO.	NAME	FUNCTION
10	STORE/MODE	MODE. Selects either standard or special register display mode. If pressed when operating in standard register display mode, switches operator panel to special register display mode. If pressed when operating in special register display mode, switches operator panel back to standard register display mode.
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 content 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 overflow register. When lighted, the overflow bit is set (logic 1).
Power Supply Front Panel Controls		
14	LOCK/OPERATE	LOCK. The RUN and HALT switches are disabled; all other functions are enabled (within the constraints of the run/halt modes). OPERATE. All switches are enabled.
15	~POWER ON/OFF	Two position circuit breaker. Controls application of ac line power to computer power supply and ventilating fans; provides protection against ac line power overload.
FPP (2117F only) Rear Panel Control		
16	~LINE ON/OFF	Two position circuit breaker. Controls application of ac line power to Floating Point Processor power supply and ventilating fans; provides protection against ac line power overload.

NOTE

When pressing the INSTR STEP switch and performing a jump instruction while monitoring the T-, P-, or M-register, the following will be noted:

a. The P-register will go to the operand target address.
b. The M-register will go to the operand target minus one (M=P−1).
c. The T-register will display the memory location prior to the current instruction to be executed.

Pressing the INSTR STEP switch will not cause an instruction step if there is a pending interrupt and the interrupt system is on; pressing PRESET or turning the interrupt system off will re-enable the INSTR STEP function.

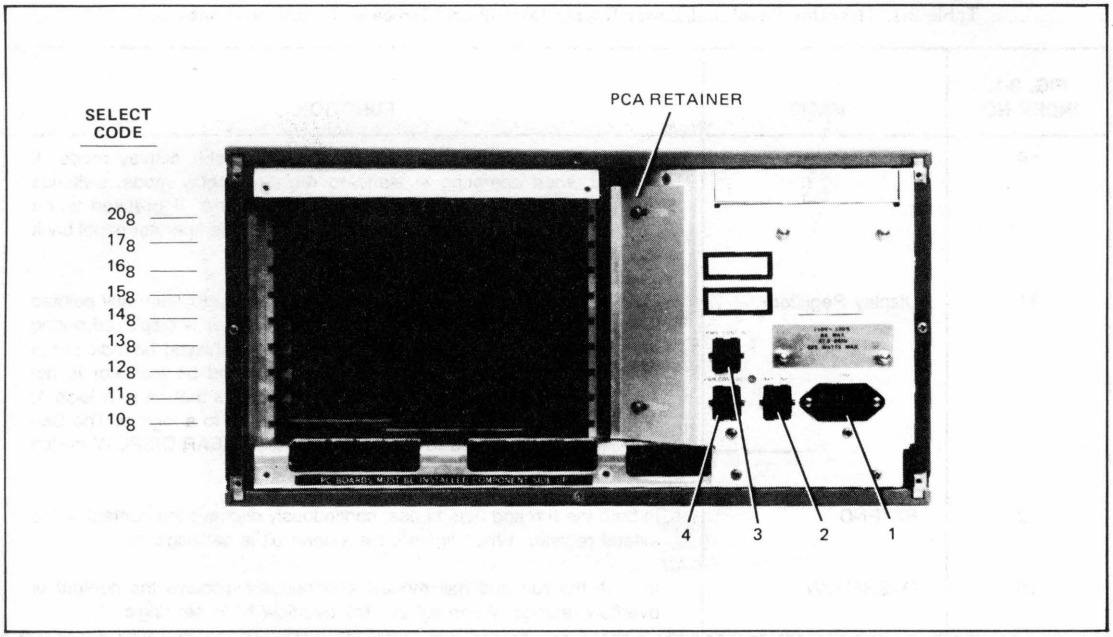
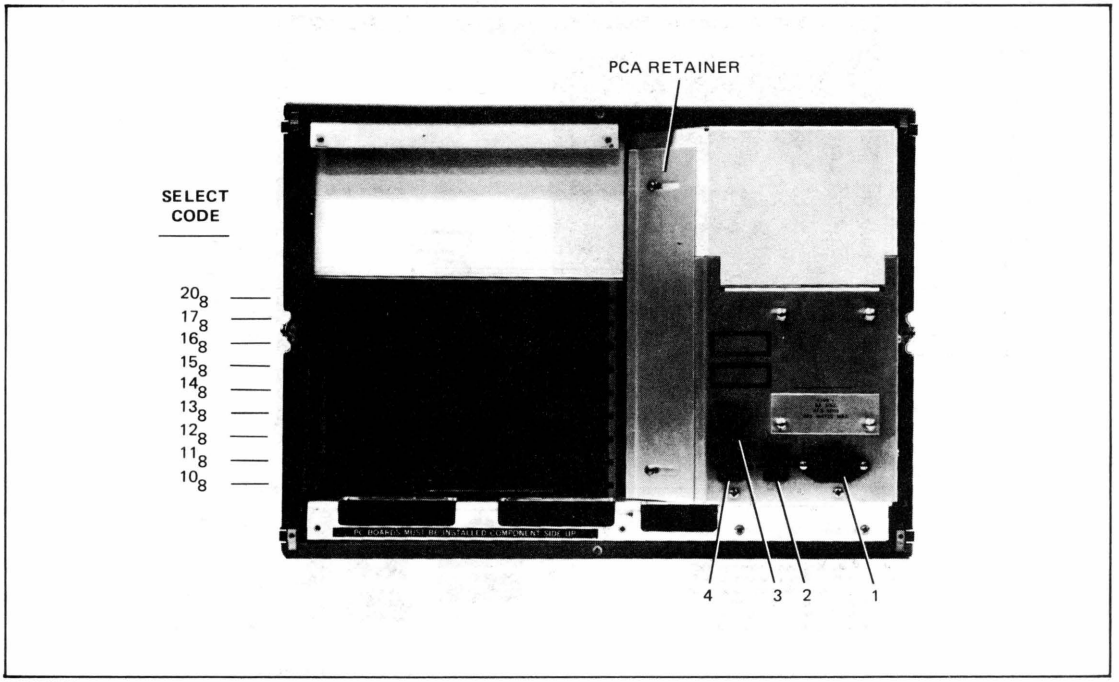


Figure 2-2. HP 2108M and HP 2109E Rear Panel and I/O PCA Cage

Table 2-2. HP 2108M and HP 2109E Rear Panel Features

FIG. 2-2, INDEX NO.	NAME	FUNCTION
1	~LINE connector	Three-input power connector; provides means of connecting ac line power to computer.
2	BAT INPUT connector	Nine-pin connector; provides means of connecting optional battery to memory sustaining circuits.
3 4	PWR CONT IN and PWR CONT OUT connectors	Two nine-pin connectors; provides means of connecting an external memory extender, I/O extender, or satellite computer (in any combination of two units) to main computer. The power supplies in all the interconnected units must be turned on before the CPU will start. When connected, the CPU monitors the ac and dc power in these units. An ac power failure in any one of the interconnected units will cause a power fail interrupt to be generated by the CPU and the CPU to halt. A dc power failure will cause the computer to stop. The interconnected units will remain inoperative until the failure has been corrected.



7700-207

Figure 2-3. HP 2111F Rear Panel and I/O PCA Cage

Table 2-3. HP 2111F Rear Panel Features

FIG. 2-3, INDEX NO.	NAME	FUNCTION
1	~LINE connector	Three-input power connector; provides means of connecting ac line power to computer.
2	BAT INPUT connector	Nine-pin connector; provides means of connecting optional battery to memory sustaining circuits.
3 4	PWR CONT IN and PWR CONT OUT connectors	Two nine-pin connectors; provides means of connecting an external memory extender, I/O extender, or satellite computer (in any combination of two units) to main computer. The power supplies in all the interconnected units must be turned on before the CPU will start. When connected, the CPU monitors the ac and dc power in these units. An ac power failure in any one of the interconnected units will cause a power fail interrupt to be generated by the CPU and the CPU to halt. A dc power failure will cause the computer to stop. The interconnected units will remain inoperative until the failure has been corrected.

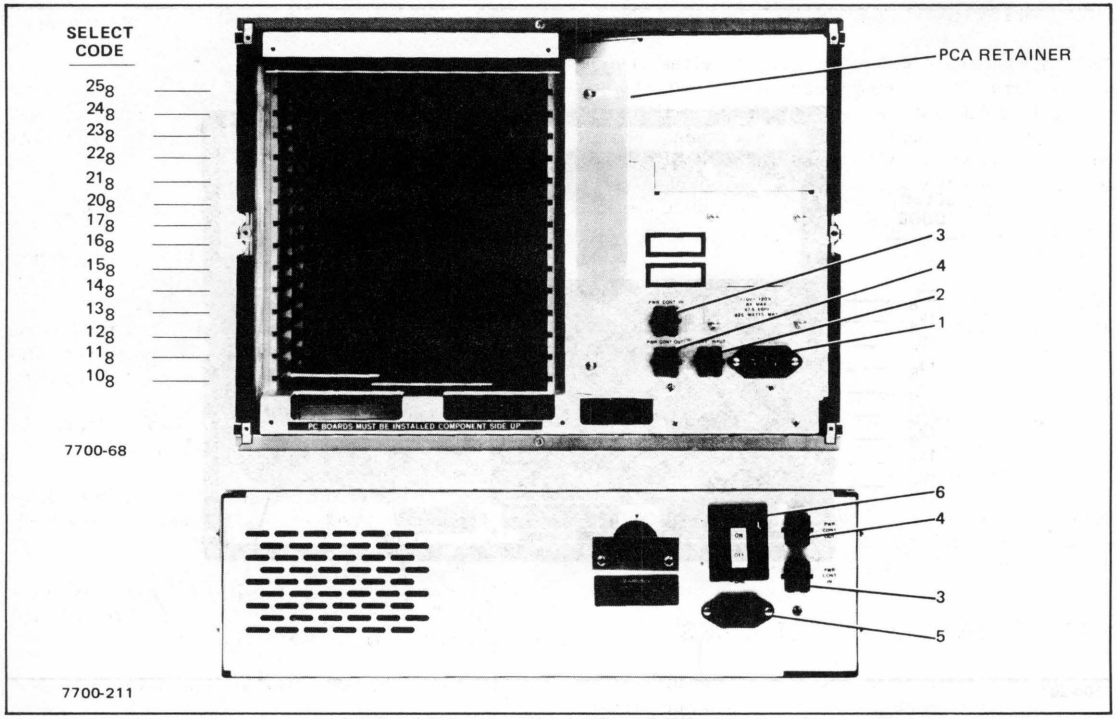


Figure 2-4. HP 2112M, HP 2113E and HP 2117F Rear Panel and I/O PCA Cage

Table 2-4. HP 2112M, HP 2113E and HP 2117F Rear Panel Features

FIG. 2-4, INDEX NO.	NAME	FUNCTION
1	~LINE connector	Three-input power connector; provides means of connecting ac line power to computer.
2	BAT INPUT connector	Nine-pin connector; provides means of connecting optional battery to memory sustaining circuits.
3 4	PWR CONT IN and PWR CONT OUT connectors	Two nine-pin connectors; provides means of connecting an external memory extender, I/O extender, or satellite computer (in any combination of two units) to main computer. The power supplies in all the interconnected units must be turned on before the CPU will start. When connected, the CPU monitors the ac and dc power in these units. An ac power failure in any one of the interconnected units will cause a power fail interrupt to be generated by the CPU and the CPU to halt. A dc power failure will cause the computer to stop. The interconnected units will remain inoperative until the failure has been corrected.
5	~LINE connector	Three-input power connector; provides means of connecting ac line power to Floating Point Processor.
6	~LINE ON/OFF	Two position circuit breaker. Controls application of ac line power to Floating Point Processor power supply and ventilating fans; provides protection against ac line power overload.

2-14. INTERNAL SWITCHES

Two toggle switches are mounted on the rear of the central processor unit (CPU) printed-circuit assembly (PCA). The setting of the ARS/ARS switch determines the action that the computer will take in the event of a primary power failure and the setting of the HLT /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 VI.

Also mounted on the central processor unit PCA are the RPL configuration switches. These switches as used as follows:

a. To enable RPL operation.
b. Select one of two loader ROM's (10_2 and 11_2).
c. Patch in the select code of loading device.
d. Select subchannel of loading device.

Details concerning the configuration of these switches are given in the Installation and Service Manual for your particular model of HP 1000 computer.

2-15. OPERATING PROCEDURES

The following procedures describe a cold power-up; how to load programs manually; how to load programs using punched tape, disc, magnetic tape, or other such media; how to verify and run programs; and how to enter the special register display mode.

2-16. COLD POWER-UP

Perform the cold power-up as follows:

a. Lower operator panel. Set the ~POWER OFF/ON switch to OFF. If computer is equipped with an optional power fail recovery system, set BATTERY switch to OFF. On the Floating Point Processor (if present), set the ~LINE switch to ON.

b. Set LOCK/OPERATE switch to OPERATE. Wait approximately three seconds and set ~POWER OFF/ON switch to ON.

c. Set BATTERY switch to ON. Raise and close the operator panel. Turn key fully counterclockwise.

d. The diagnostic will begin execution in the E-Series and F-Series and the Display Register can be observed incrementing as each 32k (E-Series) or 64k (F-Series) byte block of memory is tested. When a cold power-up is performed, these computers automatically execute a firmware self-test that checks most of the computer registers and functions and all physical memory. (Refer to paragraphs 2-31 through 2-40 for additional firmware self-test information.) This test executes in approximately 10 seconds or less depending on memory size and clears memory upon completion.

e. Upon successful completion, the T-register will automatically be selected for display.

If a computer or memory failure is detected, the display register and all six working register indicators are lighted. In this case, refer to the pertinent HP 1000 Computer Installation and Service Manual.

2-17. LOADING PROGRAMS MANUALLY

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

a. Press MODE switch if required to obtain standard register display mode and 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. Press STORE.

c. 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.

2-18. LOADING PROGRAMS FROM PAPER TAPE READER

Use the following steps to first load the contents of the standard paper tape loader ROM into memory and then load your program by means of a tape reader. Proceed as follows:

a. Press MODE switch if required to obtain standard register display mode and press left half (◀) or right half (▶) of Register Select switch to select S-register.

c. Press STORE, PRESET, and then press IBL/TEST. 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 contains the address of the first instruction of the loader. (Starting addresses versus memory size are listed in Table 2-5.)

d. A successful load is indicated if the OVERFLOW light remains off. An unsuccessful load is indicated if the OVERFLOW light is on; this will occur if the select code programmed in step b was less than 10 (octal) or if a memory hardware fault is detected.

e. 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).

If the halt code displayed is not 102077 (octal), one of two possible error condition halt codes will be displayed. If the halt code displayed is 102055 (octal), an address error is indicated; check to ensure that the proper tape was used or that the tape was not installed backwards. If the halt code displayed is 102011 (octal), a checksum error is indicated; check for a possible defective or dirty tape or tape reader.

Table 2-5. Starting Address Vs Memory Size

MEMORY SIZE (in bytes)	STARTING ADDRESS (in octal) OF THE PAPER TAPE LOADER
16k	017700
32k	037700
48k	057700
64k and up	077700

2-19. LOADING PROGRAMS FROM DISC DRIVE

Use the following steps to first load the contents of the standard disc loader ROM into memory and then load your program by means of an HP 7900A, HP 7901A, HP 7905A, HP 7906, HP 7920, or HP 7925 Disc Drive. Proceed as follows:

a. Press MODE switch if required to obtain standard register display mode and press left half (◀) or right half (▶) of Register Select switch to select S-register.

b. Press CLEAR DISPLAY and set bit 15 to logic 1 to select standard disc loader ROM.

c. Set bits 13 and 12 as shown below to select appropriate disc drive.

BITS 13	BITS 12	DISC SELECTED
0	0	HP 7900A or HP 7901A
0	1	HP 7906, or HP 7920/7925

d. Set bits 11 through 6 to octal select code of disc drive interface PCA.

e. Set bits 2 through 0 as shown below to select corresponding disc subchannel.

BITS 1	BITS 0	DISC LOADING DEVICE
0	0	HP 7900A (fixed disc)
0	1	HP 7900A or HP 7901A (removable disc)
0	0	HP 7906 (Head #0, top of removable disc) HP 7920/7925 (Head #0)
0	1	HP 7906 (Head #1, bottom of removable disc) HP 7920/7925 (Head #1)
1	0	HP 7906 (fixed disc) HP 7920/7925 (Head #2)
1	1	HP 7920/7925 (Head #3)

f. Press STORE, PRESET, and then IBL/TEST. The disc loader is now loaded into the uppermost 64 locations of memory and the select code of the disc drive is patched according to the contents of the S-register. The P-register contains the address of the first instruction of the loader. (Starting addresses versus memory size are listed in Table 2-4.)

g. A successful load is indicated if the OVERFLOW light remains off. An unsuccessful load is indicated if the OVERFLOW light is on; this will occur if the select code programmed in step d was less than 10 (octal) or if a memory hardware fault is detected.

h. Turn on and prepare disc drive for operation 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).

2-20. LOADING PROGRAMS FROM OTHER LOADING DEVICES

The following procedure is used when loading programs from a disc, magnetic tape, or other such media. The contents of the optional loader ROM, associated with the loading device, must be loaded before the program can be loaded. Locations have been provided within the computer to accommodate two optional loaders; i.e., optional loader ROM 01_2 and 11_2. Each of these loaders is used to control the loading of programs from a particular type of loading device. It is assumed that the optional loader ROM, associated with the loading device to be used, is installed in the computer and that its location is known.

Use the following steps to first load the contents of one of the optional loader ROM's into memory and then load your program by means of a disc, magnetic tape, or other such media. The program must be in binary form and must contain absolute addresses. Assuming that the loading device has been prepared for reading, proceed as follows:

a. Press MODE switch if required to obtain standard register display mode and press left half (◀) or right half (▶) of Register Select switch to select S-register.

b. Press CLEAR DISPLAY and set bit 6 through 11 to display octal select code of loading device.

c. Set bits 15 and 14 as listed in Table 2-6 to select the optional loader corresponding to your loading device.

d. Set bits 0 through 5, 12, and 13 as outlined in the instructions provided with optional loader.

e. Press STORE and then press IBL/TEST. The optional loader is now loaded into the uppermost 64 locations of memory and the select code of the loading device is patched according to the contents of the S-register. The P-register is now pointing to the address of the first instruction of the optional loader. (Starting addresses versus memory size are listed in Table 2-4.)

f. A successful load is indicated if the OVERFLOW light remains off. An unsuccessful load is indicated if the OVERFLOW light is on; this will occur if the select code programmed in step b was less than 10 (octal) or if a memory hardware fault is detected.

g. Verify that the loading device is prepared 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 program load is typically indicated if the contents of the Display Register are 102077 (octal). Refer to the instructions included with each optional loader for the specific halt code used.

Table 2-6. Optional Loader Selection

BIT		LOADER SELECTED
15	14	
0	1	Loader ROM 01_2
1	1	Loader ROM 11_2

2-21. VERIFYING PROGRAMS

If desired, programs may be verified after loading by the following procedure:

a. Press MODE switch if required to obtain standard register display mode and press left half (◀) or right half (▶) of Register Select switch to select M-register.

b. Press CLEAR DISPLAY and set Display Register to the binary starting address of the program.

c. Press STORE. Select T-register and verify that the binary instruction code is displayed as desired for the first program instruction.

d. Press INC M to increment the contents of the M-register by one and verify that the binary instruction code displayed is as desired for next programmed instruction.

e. Repeat step d until all programmed instructions have been verified. Pressing DEC M permits the previous programmed instruction to be verified.

2-22. RUNNING PROGRAMS

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

a. Press MODE switch if required to obtain standard register display mode and press left half (◀) 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 LOCK/OPERATE switch is set to OPERATE, all operator panel controls except the Display Register, CLEAR DISPLAY, and HALT switches are disabled.

The RUN indicator will remain lighted as long as the program is running. If the LOCK/OPERATE switch is set to OPERATE, all operator panel controls except the Display Register, CLEAR DISPLAY, and HALT switches are disabled.

During the run mode, the contents of the S-register are automatically selected for display in the Display Register and none of the other registers can be selected. Therefore, the Display Register effectively becomes the S-register and it can be directly addressed as I/O select code 01 (octal) by the program.

If the LOCK/OPERATE switch is set to LOCK, the functions of the RUN/HALT switch are disabled. All other operator panel controls are enabled within the constraints of the run or halt mode of operation.

NOTE

If the computer has the Power Fail Recovery System installed, the BAT TEST switch *must not* be pressed unless the battery selector is set to INT.

2-23. SPECIAL REGISTER DISPLAY MODE

The special register display mode provides the capability of displaying and/or modifying the contents of the X- and Y-registers (paragraph 2-11), the Dynamic Mapping System (DMS) registers, the extend and overflow registers (paragraphs 2-8 and 2-9), the central interrupt register (CIR), and the interrupt system. To enter the special regis-

ter display mode, use the MODE switch as discussed in Table 2-1 and press the left half (◀) or right half (▶) of the Register Select switch as required to move the unlighted "dot" indicator above the register (x, y, m, t, f, or s) to be displayed. It should be noted that the operator panel is switched back to the standard register display mode whenever the RUN switch is pressed. Each of the special register displays is discussed in the following paragraphs.

2-24. X- AND Y- (x and y) REGISTERS. When either of these registers is selected, the current contents of the register is indicated by the Display Register as discussed in paragraph 2-10 and Table 2-1. If the STORE switch is pressed while either x or y is selected, the contents of the display are loaded into the selected register. The display is not altered.

2-25. DMS MAP (m, t, and f) REGISTERS. The m-register is a 7-bit register that holds the current memory map register number (0 - 177_8) being read from or written into by the CPU. When selected for display, bits 6 - 0 indicate the memory map register number on the Display Register. The memory map register number can be incremented or decremented with the INC M/m - DEC M/m switch as discussed in Table 2-1. Any memory map can be quickly accessed when the m-register is selected for display by entering the memory map register number on the Display Register and pressing the STORE switch. To read the contents of the selected register number, select the t-register for display. The t-register is a 16-bit register that holds the selected register number contents and the read/write protect bits. When selected for display, bits 9 - 0 on the Display Register indicate the contents of the memory map register addressed by the M-register. If bit 15 is set (logic 1), the memory page is read-protected; if bit 14 is set, the memory page is write-protected. Bits 13 10 are always zero. If DMS is not installed and this register is selected for display, the display will be all 1's. To change the contents of the addressed memory map, enter the new contents on the Display Register with bit switches 9 - 0 and press the STORE switch. To read- or write-protect the memory map register, set the Display Register bit switches 15 or 14 to 1, respectively and press the STORE switch. If the INC M/m - DEC M/m switch is pressed when this register is selected for display, the contents of the next or previous memory map register address will be displayed.

The f-register is the 16-bit Memory Expansion Module status register and, when selected for display, indicates the address of the base page fence and DMS status as discussed in paragraph 4-2. If DMS is not installed and this register is selected for display, the display will be all 1's. To alter the address of the base page fence, enter the new address on the Display Register bit switches 10 - 0 and press the STORE switch. Status bits 15 - 11 cannot be altered. If Display Register bit switches 15 - 11 are changed from the original display and the STORE switch is pressed, no action takes place and the original contents of the f-register is again displayed.

2-26. STATUS (s) REGISTER. The s-register is a 16-bit status register. When selected for display, bit 15 indicates the status of the overflow register. (Refer to paragraph 2-9.) Bit 14 indicates the status of the extend register. (Refer to paragraph 2-8.) Bit 13 indicates the status of the interrupt system; a logic 1 indicates that the system is enabled. Bits 12 - 6 are always zero. Bits 5 - 0 indicate the current contents of the Central Interrupt Register (CIR) which is the octal select code of the device that last interrupted the computer. When the s-register is displayed, the overflow register bit, extend register bit, and interrupt system can be set as desired with Display Register bit switches 15, 14, and 13 respectively, and then pressing the STORE switch. The contents of CIR cannot be altered.

Bit 13 of the s-register will not display the status of the interrupt system correctly and cannot be changed if memory protect is enabled.

2-27. SHUTDOWN PROCEDURES

One of the following procedures should be used when the computer is to be shut down during periods of nonoperation. The first procedure should be used when it is necessary to sustain memory contents for a brief period (the battery is discharging whenever the computer ~ POWER switch is in the OFF position with the battery switch set to INT). The second procedure should be used when it is not necessary to sustain the contents of memory.

2-28. SHUTDOWN PROCEDURE (MEMORY SUSTAINED). Use the following procedure to shut down the computer when it is necessary to sustain memory during brief periods of nonoperation:

a. The computer MUST BE equipped with the optional Power Fail Recovery system, or the contents of memory will be lost when power is removed from the computer. You must know the number of Memory Array PCA's and if your computer is equipped with a Fault Control Memory System the number of Check Bit Array PCA's to determine the amount of time that the contents of memory can be sustained. Determine the number of Memory and Check Bit Array modules installed in your computer, then refer to the sustaining battery graph in Table 1-1 to calculate the amount of time that memory can be sustained in your computer.

CAUTION

Depending on the amount of discharge the battery can take up to 16 hours to fully recharge.

b. Ensure that the BATTERY switch is set to INT. Then set the computer ~ POWER ON/OFF switch to OFF, record the time and calculate the time that power must be restored to the computer to ensure that the

integrity of the memory is preserved. Restore power at or before the time previously calculated.

NOTE

If the computer is housed in a system cabinet and you want power to the other system components, ensure that the cabinet power switch is ON.

2-29. SHUTDOWN (MEMORY NOT SUSTAINED). Use the following procedure to shut down the computer when it is not necessary to sustain memory during periods of nonoperation:

a. Set ~ POWER OFF/ON switch to OFF. If computer is equipped with optional power fail recovery system, set BATTERY switch to OFF to prevent the battery from discharging.

b. Set computer ~ POWER OFF/ON switch to OFF, or, if the computer is housed in a system cabinet, set the system power switch to remove ac power.

All contents of memory and internal registers will be lost. When operation is to be resumed, the cold power-up procedure and program loading must be repeated.

2-30. REMOTE PROGRAM LOADING (E-SERIES AND F-SERIES ONLY)

The optional Remote Program Load (RPL) capability is a combination of hardware and firmware that allows loading and initiating execution of program to be controlled from a remote site. The hardware consists of a set of eight switches called the configuration switches mounted on the CPU board and typically, an appropriate communications interface board. (The HP 12966A or HP 12968A must be used if an HP standard interface is utilized.) The configuration switches are used to program the information normally entered into the S-register from the operator panel for Initial Binary Loader (IBL) operations. (Refer to the HP 1000 E/F-Series Computer Installation and Service manual for switch programming details.) The communications interface board provides the link through which remote devices initiate RPL operations. The firmware is a micro-routine that automatically reads the information from the configuration switches, calls the IBL routine specified by the configuration switches, and issues the RUN command from a remote computer site, a console device, or automatically when line power is applied to the computer.

The following conditions will cause execution of the RPL micro-routine if RPL is enabled and the LOCK/OPERATE switch is in the LOCK position:

a. Cold Power-up;

b. Programmed halt instructions (106XXX, 107XXX); or

c. Forced halts initiated via the I/O system. (This allows system consoles via their interface or remote systems via data communication links to initiate the RPL sequence.)

The following conditions will not cause execution of the RPL micro-routine, even though RPL is enabled:

a. LOCK/OPERATE switch is in the OPERATE position;

b. Auto-restart is enabled (A1S2 set to ARS), power coming up, and PRESET has not been pressed;

c. Programmed halt instructions (102XXX, 103XXX); or

d. Parity halt (A1S1 set to HALT).

A flow chart of the complete RPL sequence is shown in Figure 2-5.

2-31. SELF-TEST FIRMWARE

The standard microprogrammed base set includes three tests that are designed to conveniently and quickly test the computer and memory without supplementary diagnostics. These firmware self-tests are not designed as a substitute for more complex software diagnostics and it may frequently be the case that you require a more thorough and detailed testing than provided by these standard self-test routines. Firmware test for the scientific instruction set (SIS) and the floating point processor (FFP) can be executed via the operator panel.

2-32. TEST DESCRIPTIONS

2-33. TEST 1. Test 1 tests most of the computer registers and functions. This test will not alter or destroy the contents of any working register or memory. An error condition will set all display registers, indicator bits, and the overflow register. The execution time is negligible.

2-34. TEST 2. Test 2 is a fast microprogrammed memory test that checks the presently enabled memory space (up to 64k bytes). The microprogram reads each memory location, complements the data and writes it back, reads it, compares it to expected data, then complements it and writes it back into memory. The execution time is negligible and is non-destructive to memory data. An error condition is usually accompanied by a parity error indication and will set all display registers and indicator bits and clear the overflow register. The A-register will contain the expected (good) data, the B-register will contain the actual (bad) data, and the M-register will contain the logical memory location of the failure. If all display indicators and display registers do not light, but the PARITY indicator on the operator panel is lit, a check bit array or a parity bit on a memory module has failed.

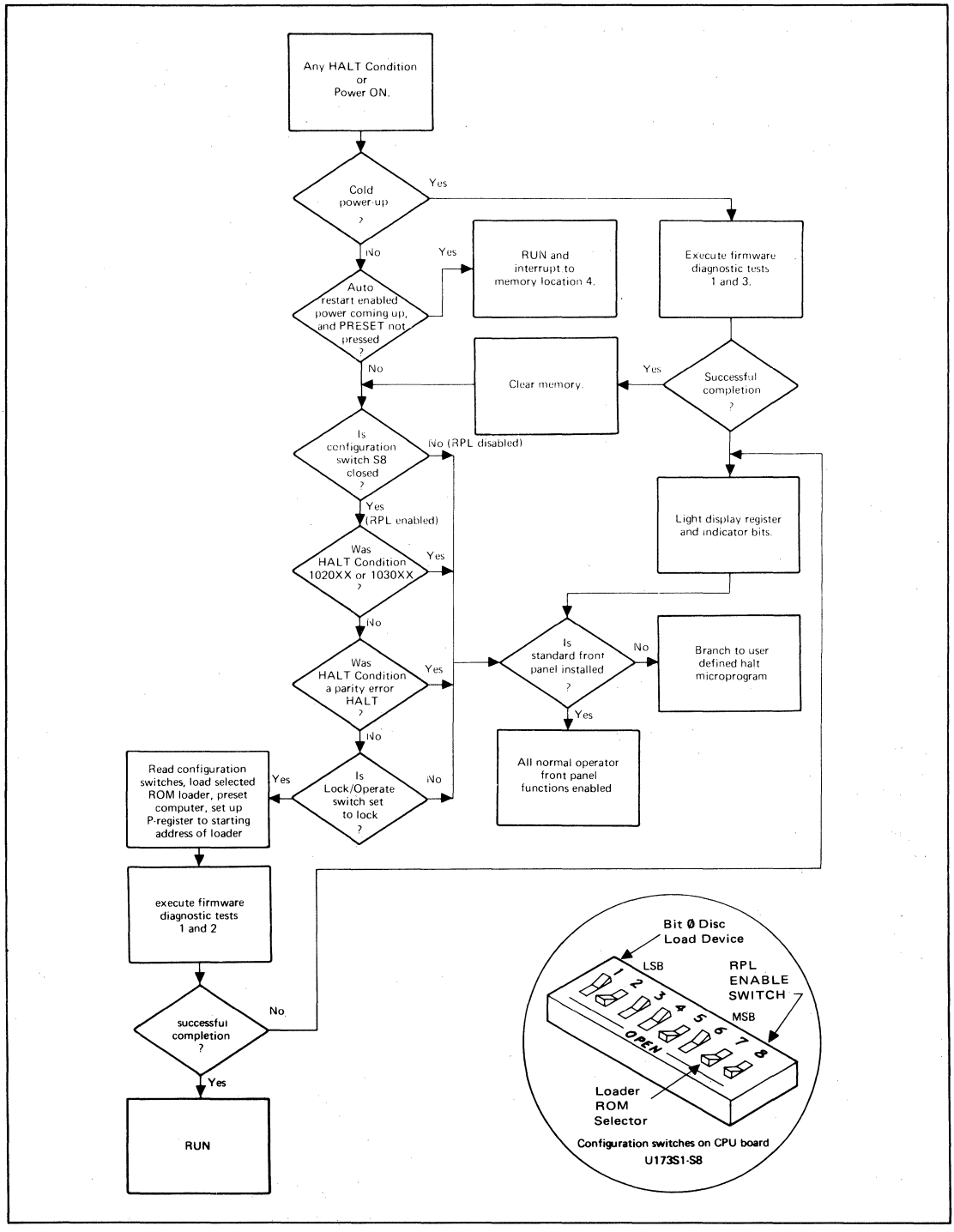


Figure 2-5. Remote Program Load Routine

2-35. TEST 3. Test 3 is a significantly more sophisticated microprogrammed memory test. All memory installed in the computer will be tested. Execution time is dependent on the amount of memory installed; approximately one second per 64k bytes. The display register will increment as memory in each 64k byte space is tested. Error reporting is the same as in Test 2 except the S-register will contain the number of the 64k byte space where the memory failure occurred. (Refer to the HP 1000 E/F-Series Computer Installation and Service Manual for additional information.)

2-36. TEST EXECUTION

On a cold power-up (paragraph 2-16), Tests 1 and 3 will each be executed once.

Pressing the IBL-TEST switch on the operator panel will not only perform the loader function as described in paragraphs 2-18 and 2-19, it will also cause the execution of Tests 1 and 2.

Tests 1 and 3 may also be executed via the operator panel as follows:

a. Set P = 0.

b. Set A = 100000 (octal).

c. At the end of the diagnostic (when executed once), memory will be filled with the value in the S-register. Set the S-register if desired.

d. Press PRESET.

e. To loop the diagnostic, set the LOCK/OPERATE switch to LOCK.

f. Press INSTR STEP, the diagnostic will now be executed.

g. While the diagnostic is being executed, if S-register bit 5 is set, the S-register will display the number of executions (bit 5 = LSB).

If the LOCK/OPERATE switch is set to LOCK position, the microprogrammed diagnostic will continuously loop.

2-37. FPP TEST. (F-Series only) To execute the Floating Point Processor (FPP) test proceed as follows:

a. Store 105004 (octal) in the A-register.

b. Store 0 in the P-register and press PRESET. If the OVFL light remains on, refer to the F-Series computer installation and service manual for troubleshooting information.

c. Press INSTR STEP.

d. A 102077 (octal) in the display register (S) indicates successful completion; otherwise, refer to the F-Series computer installation and service manual.

2-38. FFP TEST. (F-Series only) To execute FFP firmware test, proceed as follows:

a. Store 105200 (octal) in the A-register.

b. Store 0 in the P-register.

c. Press PRESET; then press INSTR STEP.

A 102077 (octal) in the display register (S) indicates successful completion; otherwise, refer to the F-Series computer installation and service manual.

2-39. SIS TEST. (F-Series only) To execute the Scientific Instruction Set (SIS) firmware test, proceed as follows:

a. Store 105337 (octal) in the A-register.

b. Store 0 in the P-register.

c. Press PRESET; then press INSTR STEP.

A 102077 (octal) in the display register (S) indicates successful completion; otherwise, refer to the F-Series computer installation and service manual.

2-40. VIS TEST. (F-Series only) To execute the Vector Instruction Set (VIS) firmware test, proceed as follows:

a. Store 105477 (octal) in the A-register.

b. Store 0 in the P-register.

c. Press PRESET; then press INSTR STEP.

A 102077 (octal) in the display register (S) indicates successful completion. Any other display in the S-register indicates that the VIS firmware test failed and the VIS On-Line Diagnostic, part no. 12824-16001, should be run. The diagnostic operating procedures are contained in the VIS Users Manual, part no. 12824-90001.

2-41. EXCHANGING I/O INTERFACES

Figures 2-2, 2-3 and 2-4 show the I/O PCA cages and the select codes associated with each slot. Select code 10 (octal) has the highest priority in the interrupt structure and the highest numbered select code has the lowest priority. When it becomes necessary to install a new I/O interface PCA or change the location of an existing one, proceed as follows:

CAUTION

If the computer is not equipped with an optional power fail recovery system, the contents of memory will be lost when the line (mains) voltages are off. Therefore, store any contents of memory to be saved in another medium for later retrieval; then perform steps c and e through j below.

a. Check that the ~POWER switch is set to ON.

NOTE

If the computer is equipped with a power fail recovery system, ensure that there is a place to support the battery box with it connected to the computer.

b. Set the BATTERY switch to INT.

c. Loosen the four captive screws securing the I/O cage cover to the computer rear frame.

d. Place the I/O cage cover with the battery box on a support so that the battery cable remains connected.

e. Set the ~POWER switch to OFF.

f. Loosen the two screws holding the I/O PCA retainer and slide the retainer to permit the removal or installation of the I/O PCA's.

NOTE

If the I/O PCA associated with RPL is relocated, then the internal RPL configuration switches must be reset. Details on how to configure these switches are given in the HP 1000 E-Series or F-Series Computer Installation and Service Manual.

g. Install the new I/O interface PCA or exchange I/O interface PCA's as required. If an HP 12979B I/O Extender is to be used, install its interface PCA in the first available lowest priority I/O slot.

h. Slide the I/O PCA retainer to the left and tighten the two screws.

i. Apply power to the computer by setting the ~POWER switch to ON.

j. Secure the I/O cage cover with the battery box to the rear frame of the computer by fastening the four captive screws.

2-42. HALT CODES

Table 2-7 provides a quick reference to those halt codes associated with the input device loader. These halt codes are displayed in the Display Register, when the computer is in the halt mode.

2-43. ABNORMAL INDICATIONS

Table 2-8 provides a quick reference to the operator panel indications that occur when an abnormal condition exists during operation in the normal register display mode. Abnormal indications encountered during the execution of the firmware tests are discussed in paragraphs 2-31 through 2-38.

Table 2-7. Halt Codes

HALT CODE (in octal)	COMMENTS
102077	Indicates a successful program load from paper tape and typically indicates a successful program load from disc, magnetic tape, or other such media.
102055	Indicates that an address error occurred while loading from input device.
102011	Indicates that a checksum error occurred while loading from input device.

Table 2-8. Abnormal Indications

INDICATION	ABNORMAL CONDITION	REMEDY
POWER FAIL light remains on.	Indicates that power has been restored after a power failure.	Press HALT; then PRESET or execute an STC 04 or CLC 04 instruction.
PARITY light is on.	Indicates that a parity error occurred while reading from memory.	Refer to *HP 1000 E-Series or F-Series Installation and Service Manual.*
OVERFLOW light is on after IBL/ TEST is pressed.	Indicates that:	
	a. The presence of memory was not detected.	a. Check that memory modules are installed and programmed correctly.
	b. The programmed select code was less than 10 (octal).	b. Check that the programmed select code is within range.
	c. The memory was defective.	c. Refer to *HP 1000 E-Series or F-Series Installation and Service Manual.*
Operator panel indicators are irregular after cold power-up; Register Select switch cannot select registers.	Indicates that:	
	a. Power supply or CPU PCA is defective.	a. Refer to *HP 1000 E-Series or F-Series Computer Installation and Service Manual.*
	b. If Power Fail Recovery System is installed, battery cable is not connected or connector 12991-60002 is not connected to BAT INPUT connector.	b. Connect battery cable from battery box (or connect connector 12991-60002) to BAT INPUT connector.

PROGRAMMING INFORMATION

SECTION III

This section describes the software data formats and the base set machine-language instruction coding required to operate the computer and its associated input/output system. This section also describes the Scientific Instruction Set of the F-Series Computers as well as the Fast FORTRAN Processor instructions which are optional with the E-Series computers and standard with the F-Series Computers. Machine-language instruction coding for the Dynamic Mapping System (standard with the HP 2117F and optional with all other HP 1000 Computers) is presented in Section VI.

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 integer. 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 double integer format is used for extended arithmetic in conjunction with the extended arithmetic instructions described under paragraphs 3-21 and 3-22. 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.

The two floating point formats in Figure 3-1 are 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 express the mantissa. A single precision floating point number is made up of a 23-bit mantissa (fraction) and sign and a 7-bit exponent and sign, thus providing six significant decimal digits in the mantissa. An extended precision floating point number is made up of a 39-bit mantissa and a 7-bit exponent and sign, thus providing 11 significant decimal digits in the mantissa. A double precision floating point number is made up of a 55-bit mantissa and a 7-bit exponent and sign, thus providing 16 significant decimal digits in the mantissa. If either the mantissa or the exponent is negative, that part must be 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 A-register contains the most-significant word and the B-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 integer data is +32,767 to −32,768 (decimal) or +77,777 to −100,000 (octal). The range of representable numbers for double 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. ADDRESSING

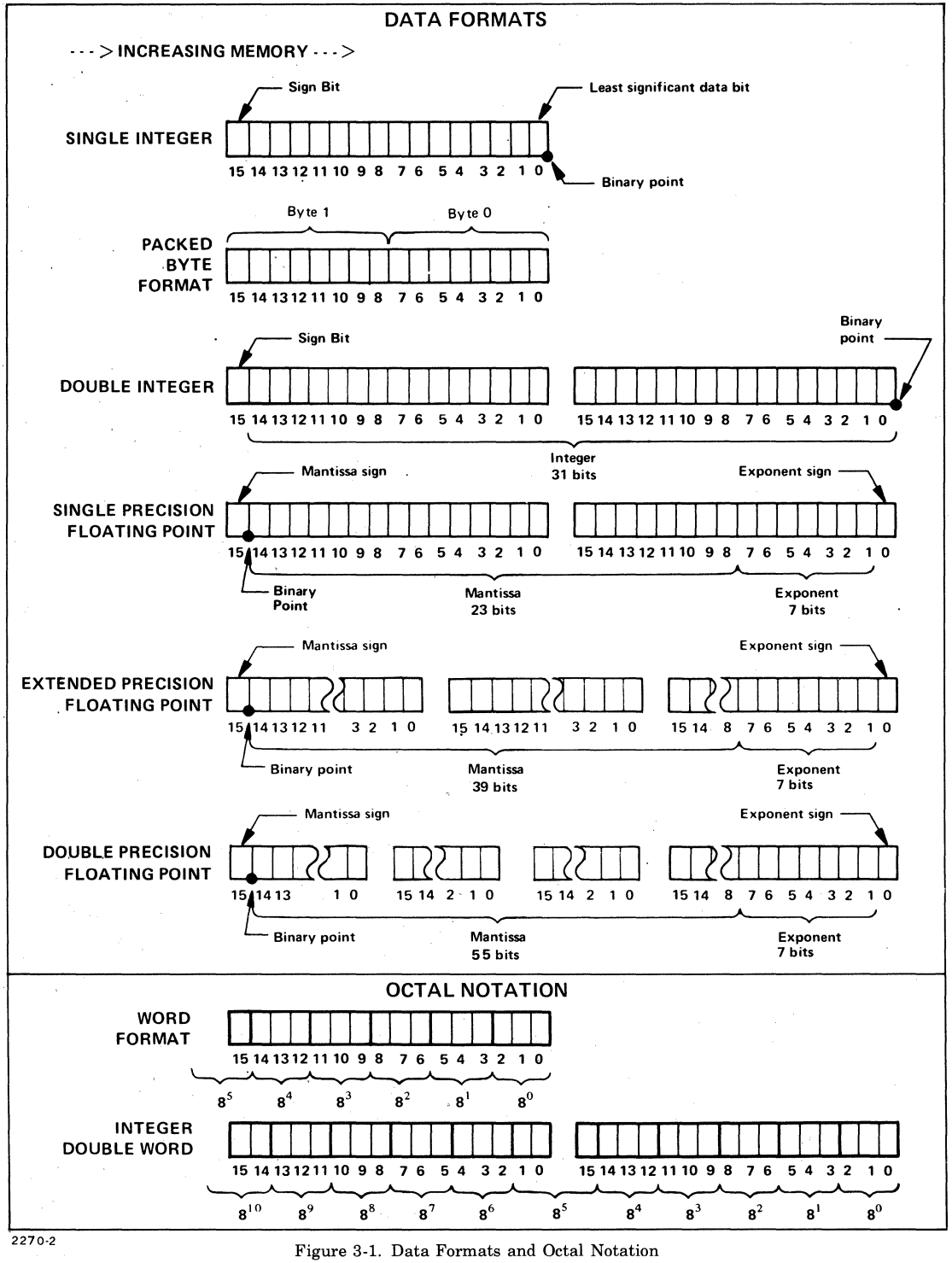
3-3. PAGING

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 words, 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).



2270-2

Figure 3-1. Data Formats and Octal Notation

Table 3-1. Memory Paging

MEMORY SIZE	PAGE	OCTAL ADDRESSES
	0	00000 to 01777
	1	02000 to 03777
	2	04000 to 05777
	3	06000 to 07777
	4	10000 to 11777
	5	12000 to 13777
	6	14000 to 15777
	7	16000 to 17777
	8	20000 to 21777
	9	22000 to 23777
	10	24000 to 25777
	11	26000 to 27777
	12	30000 to 31777
	13	32000 to 33777
	14	34000 to 35777
16K ↓	15	36000 to 37777
	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
	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 of the instruction. Direct or indirect addressing is specified by bit 15 as follows:

a. Logic 0 = Direct (D).

b. 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). If options or input/output devices corresponding to locations 00005 through 00077 are not 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, memory protect, and DMS 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.

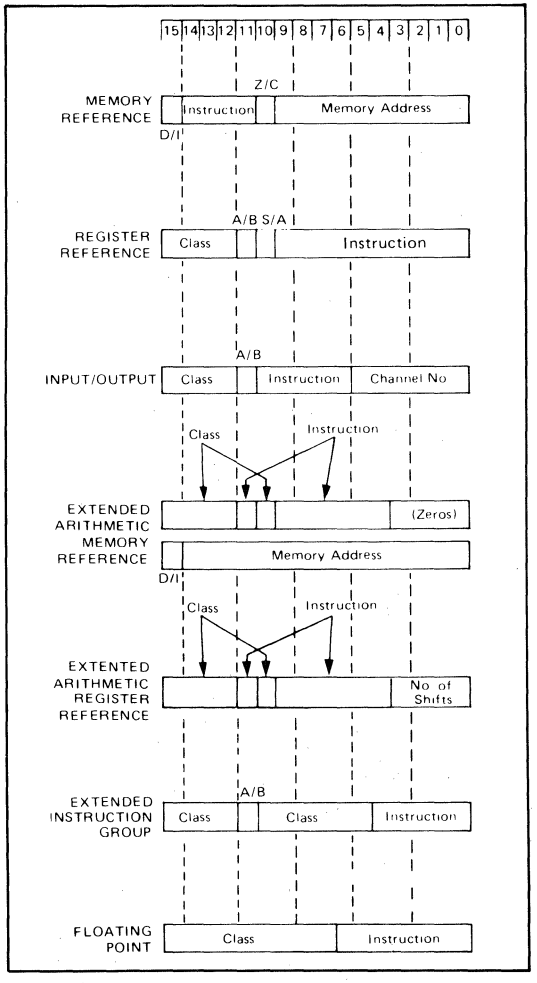


Figure 3-2. Base Set Instruction Formats

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 instructions 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. EXTENDED INSTRUCTIONS

The extended instructions include index register instructions, bit and byte manipulation instructions, and move and compare instructions. 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) generally indicating direct or indirect addressing.

3-14. FLOATING POINT INSTRUCTIONS

The floating point instructions allow addition, subtraction, multiplication, and division of floating point quantities. Conversion routines are provided for transforming numerical integer representations to/from floating point representations.

3-15. 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 light gray shaded bits code the instruction type and the dark gray 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.

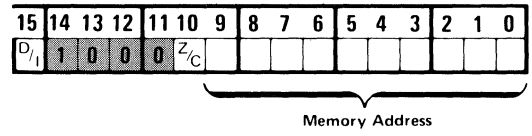
In all cases where an additional bit is used to specify a secondary function (D/I, Z/C, or H/C), the choice is made by coding a logic 0 or logic 1, respectively. In other words, a logic 0 codes D (direct addressing), Z (zero page), or H (hold flag); a logic 1 codes I (indirect addressing), C (current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

3-16. 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.

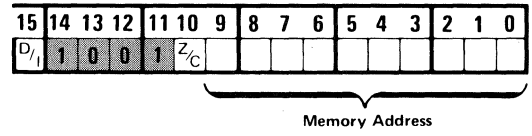
If bit 10 (Z/C) is a logic 0, the memory address is on page zero; if bit 10 is a logic 1, the memory address is on the current page. If the A- or B-register is addressed, bit 10 must be a logic 0 to specify page zero, unless the current page is page zero.

ADA ADD TO A



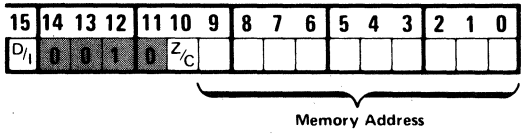
Adds the contents of the addressed memory location to the contents of the A-register. The sum remains in the A-register and the contents of the memory cell are unaltered. The result of this addition may set the extend bit or the overflow bit. (Extend and overflow examples are illustrated on page A-13.)

ADB ADD TO B



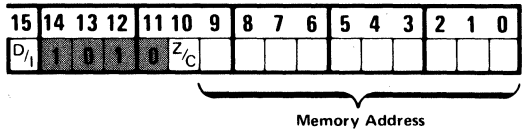
Adds the contents of the addressed memory location to the contents of the B-register. The sum remains in the B-register and the contents of the memory cell are unaltered. The result of this addition may set the extend bit or the overflow bit. (Extend and overflow examples are illustrated on page A-13.)

AND "AND" TO A



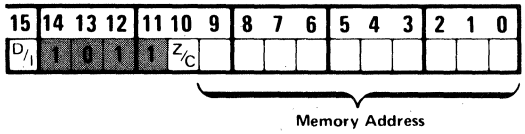
Combines the contents of the addressed memory location and the contents of the A-register by performing a logical "and" operation. The contents of the memory cell are unaltered.

CPA COMPARE TO A



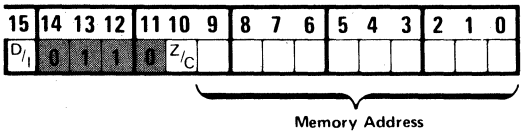
Compares the contents of the addressed memory location with the contents of the A-register. If the two 16-bit words are not identical, the next instruction is skipped; i.e., the P-register advances two counts instead of one count. If the two words are identical, the next sequential instruction is executed.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

CPB COMPARE TO B



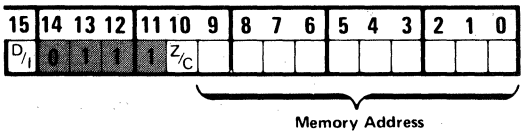
Compares the contents of the addressed memory location with the contents of the B-register. If the two 16-bit words are not identical, the next instruction is skipped; i.e., the P-register advances two counts instead of one count. If the two words are identical, the next sequential instruction is executed.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

IOR "INCLUSIVE OR" TO A



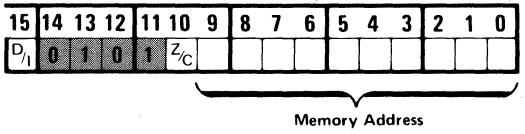
Combines the contents of the addressed memory location and the contents of the A-register by performing a logical "inclusive or" operation. The contents of the memory cell are unaltered.

ISZ INCREMENT AND SKIP IF ZERO



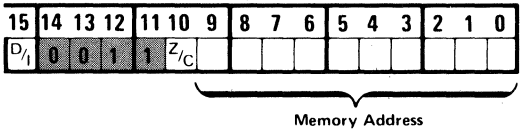
Adds one to the contents of the addressed memory location. If the result of this operation is zero (memory contents incremented from 177777 to 000000), the next instruction is skipped; i.e., the P-register is advanced two counts instead of one count. If the result of this operation is not zero, the next sequential instruction is executed. In either case, the incremented value is written back into the memory cell.

JMP JUMP



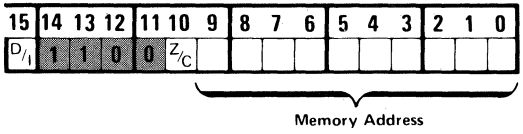
Transfers control to the addressed memory location. That is, a JMP causes the P-register count to set according to the memory address portion of the JMP instruction so that the next instruction will be read from that location.

JSB JUMP TO SUBROUTINE



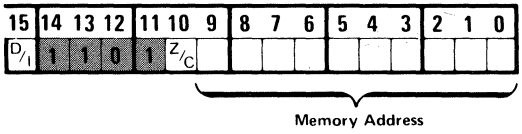
This instruction, executed in location P (P-register count), causes the computer control to jump unconditionally to the memory location (m) specified by the memory address portion of the JSB instruction. The contents of the P-register plus one (return address) is stored in memory location m, and the 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.

LDA LOAD A



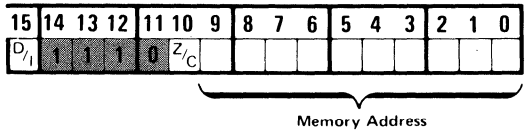
Loads the contents of the addressed memory location into the A-register. The contents of the memory cell are unaltered.

LDB LOAD B



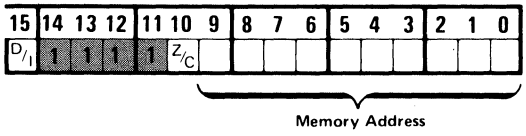
Loads the contents of the addressed memory location into the B-register. The contents of the memory cell are unaltered.

STA STORE A



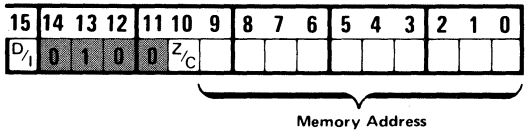
Stores the contents of the A-register in the addressed memory location. The previous contents of the memory cell are lost; the A-register contents are unaltered.

STB STORE B



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.

XOR "EXCLUSIVE OR" TO A



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.

3-17. 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. Since the two groups perform separate and distinct functions, instructions from the two groups cannot be mixed. Unshaded bits in the coding diagrams are available for combining other instructions.

3-18. 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 Figure 3-3. Rules for combining instructions in this group are as follows (refer to Table 3-3):

a. Only one instruction can be chosen from each of the two multiple-choice columns.

b. References can be made to either the A-register or B-register, but not both.

c. Sequence of execution is from left to right.

d. In machine code, use zeros to exclude unwanted microinstructions.

e. 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.

f. 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.

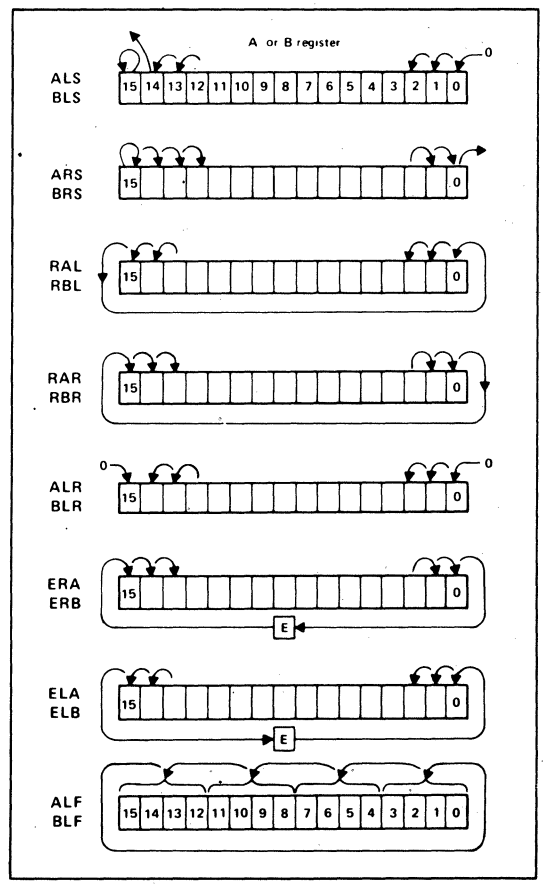
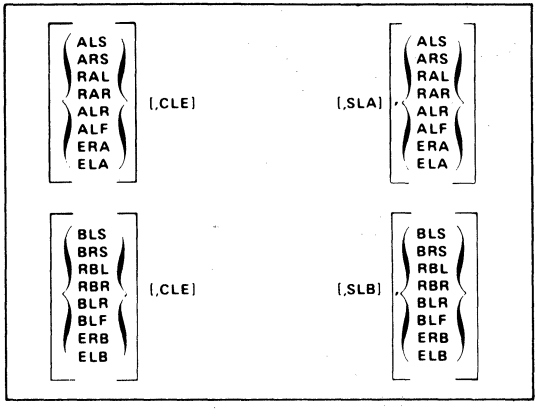
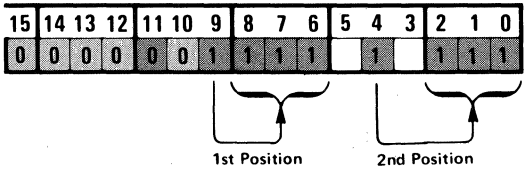


Figure 3-3. Shift and Rotate Functions

Table 3-3. Shift/Rotate Group Combining Guide

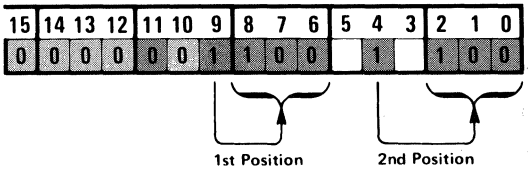


ALF ROTATE A LEFT FOUR



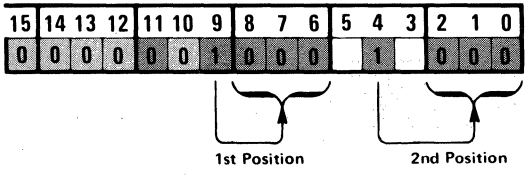
Rotates the A-register contents (all 16 bits) left four places. Bits 15, 14, 13, and 12 rotate around to bit positions 3, 2, 1, and 0, respectively. Equivalent to four successive RAL instructions.

ALR A LEFT SHIFT, CLEAR SIGN



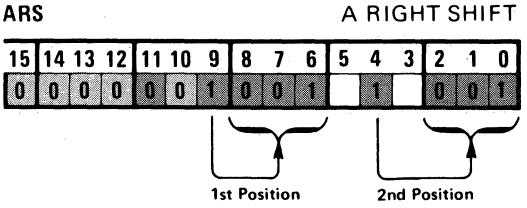
Shifts the A-register contents left one place and clears sign bit 15.

ALS A LEFT SHIFT



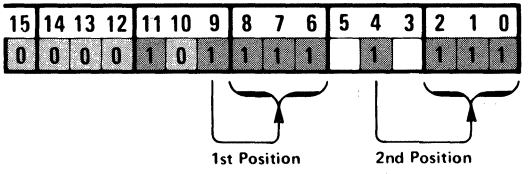
Arithmetically shifts the A-register contents left one place, 15 magnitude bits only; bit 15 (sign) is not affected. The bit shifted out of bit position 14 is lost; a logic 0 replaces vacated bit position 0.

ARS A RIGHT SHIFT



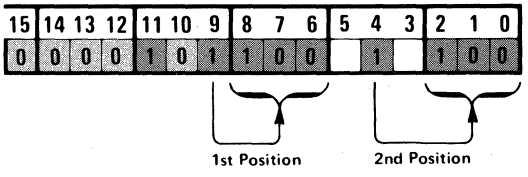
Arithmetically shifts the A-register contents right one place, 15 magnitude bits only; bit 15 (sign) is not affected. A copy of the sign bit is shifted into bit position 14; the bit shifted out of bit position 0 is lost.

BLF ROTATE B LEFT FOUR



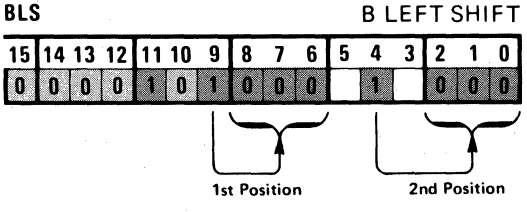
Rotates the B-register contents (all 16 bits) left four places. Bits 15, 14, 13, and 12 rotate around to bit positions 3, 2, 1, and 0, respectively. Equivalent to four successive RBL instructions.

BLR B LEFT SHIFT, CLEAR SIGN



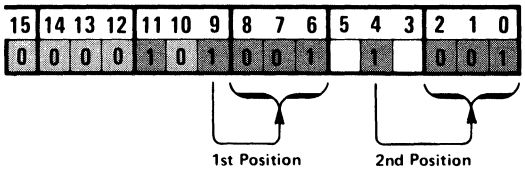
Shifts the B-register contents left one place and clears sign bit 15.

BLS B LEFT SHIFT



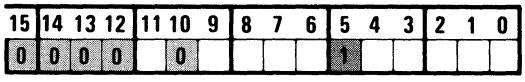
Arithmetically shifts the B-register contents left one place, 15 magnitude bits only; bit 15 (sign) is not affected. The bit shifted out of bit position 14 is lost; a logic 0 replaces vacated bit position 0.

BRS B RIGHT SHIFT



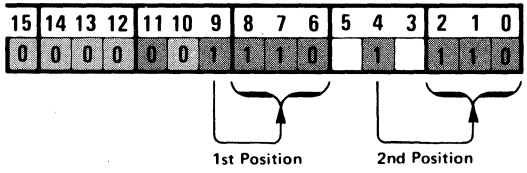
Arithmetically shifts the B-register contents right one place, 15 magnitude bits only; bit 15 (sign) is not affected. A copy of the sign bit is shifted into bit position 14; the bit shifted out of bit position 0 is lost.

CLE CLEAR E



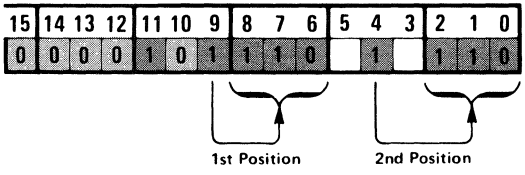
Clears the E-register; i.e., the extend bit becomes a logic 0.

ELA ROTATE E LEFT WITH A



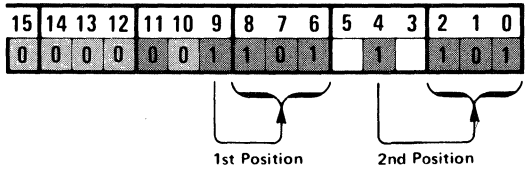
Rotates the E-register content left with the A-register contents (one place). The E-register content rotates into bit position 0; bit 15 rotates into the E-register.

ELB ROTATE E LEFT WITH B



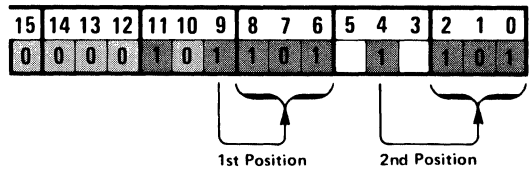
Rotates the E-register content left with the B-register contents (one place). The E-register content rotates into bit position 0; bit 15 rotates into the E-register.

ERA ROTATE E RIGHT WITH A



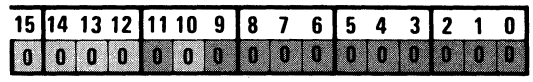
Rotates the E-register content right with the A-register contents (one place). The E-register content rotates into bit position 15; bit 0 rotates into the E-register.

ERB ROTATE E RIGHT WITH B



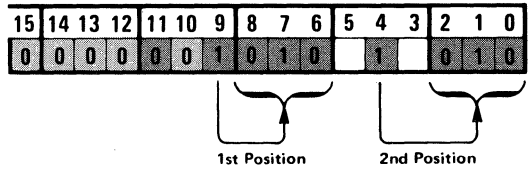
Rotates the E-register content right with the B-register contents (one place). The E-register content rotates into bit position 15; bit 0 rotates into the E-register.

NOP NO OPERATION



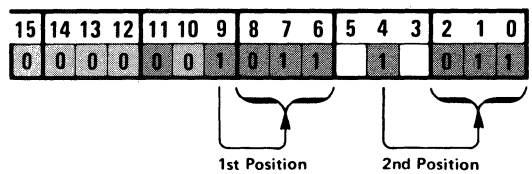
This all-zeros instruction causes a no-operation cycle.

RAL ROTATE A LEFT



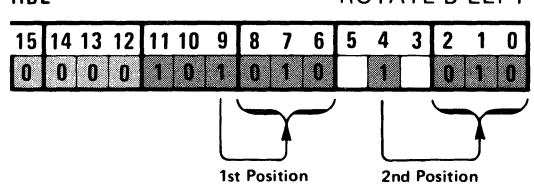
Rotates the A-register contents left one place (all 16 bits). Bit 15 rotates into bit position 0.

RAR ROTATE A RIGHT



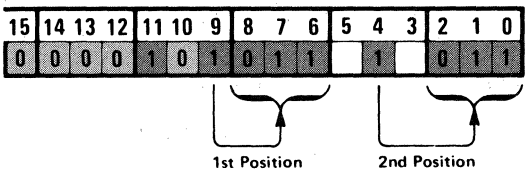
Rotates the A-register contents right one place (all 16 bits). Bit 0 rotates into bit position 15.

RBL ROTATE B LEFT



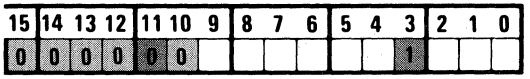
Rotates the B-register contents left one place (all 16 bits). Bit 15 rotates into bit position 0.

RBR ROTATE B RIGHT



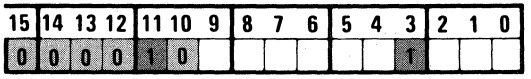
Rotates the B-register contents right one place (all 16 bits). Bit 0 rotates into bit position 15.

SLA SKIP IF LSB OF A IS ZERO



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



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

3-19. 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 instructions are as follows (refer to Table 3-4):

a. Only one instruction can be chosen from each of the two multiple-choice columns.

b. References can be made to either the A-register or B-register, but not both.

c. Sequence of execution is from left to right.

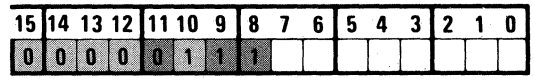
d. If two or more skip functions are combined, the skip function will occur if either or both conditions are met. One exception exists: refer to the RSS instruction.

e. In machine code, use zeros to exclude unwanted instructions.

Table 3-4. Alter/Skip Group Combining Guide

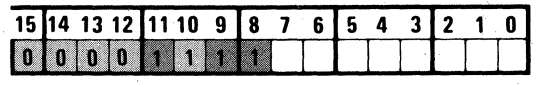
[CLA / CMA / CCA]	[,SEZ]	[,CLE / ,CME / ,CCE]	[,SSA] [,SLA] [,INA] [,SZA] [,RSS]
[CLB / CMB / CCB]	[,SEZ]	[,CLE / ,CME / ,CCE]	[,SSB] [,SLB] [,INB] [,SZB] [,RSS]

CCA CLEAR AND COMPLEMENT A



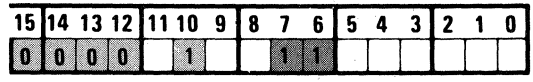
Clears and complements the A-register contents; i.e., the contents of the A-register become 177777 (octal). This is the two's complement form of -1.

CCB CLEAR AND COMPLEMENT B



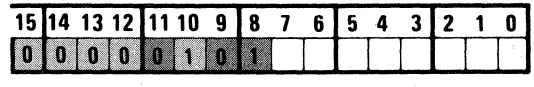
Clears and complements the B-register contents; i.e., the contents of the B-register become 177777 (octal). This is the two's complement form of -1.

CCE CLEAR AND COMPLEMENT E



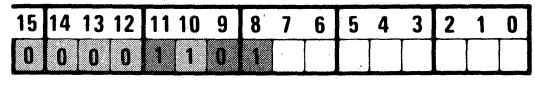
Clears and complements the E-register content (extend bit); i.e., the extend bit becomes a logic 1.

CLA CLEAR A



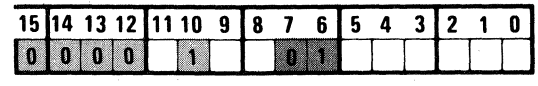
Clears the A-register; i.e., the contents of the A-register become 000000 (octal).

CLB CLEAR B



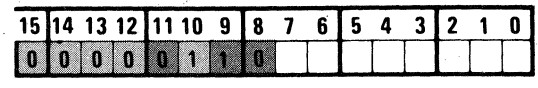
Clears the B-register; i.e., the contents of the B-register become 000000 (octal).

CLE CLEAR E



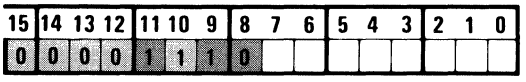
Clears the E-register; i.e., the extend bit becomes a logic 0.

CMA COMPLEMENT A



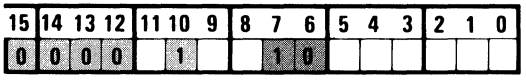
Complements the A-register contents (one's complement).

CMB COMPLEMENT B



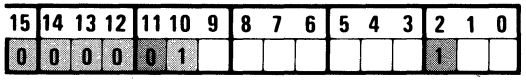
Complements the B-register contents (one's complement).

CME COMPLEMENT E



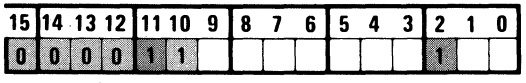
Complements the E-register content (extend bit).

INA INCREMENT A



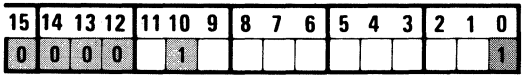
Increments the A-register by one. The overflow bit will be set if an increment of the largest positive number (077777 octal) is made. The extend bit will be set if an all-ones word (177777 octal) is incremented.

INB INCREMENT B



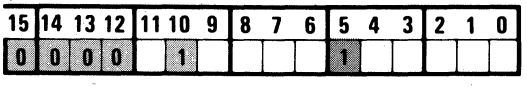
Increments the B-register by one. The overflow bit will be set if an increment of the largest positive number (077777 octal) is made. The extend bit will be set if an all-ones word (177777 octal) is incremented.

RSS REVERSE SKIP SENSE



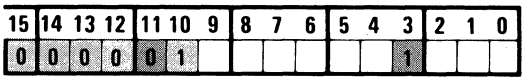
Skip occurs for any of the following skip instructions, if present, when the non-zero condition is met. An RSS without a skip instruction in the word causes an unconditional skip. If a word with RSS also includes both SSA and SLA (or SSB and SLB), bits 15 and 0 must both be logic 1's for a skip to occur;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

SEZ SKIP IF E IS ZERO



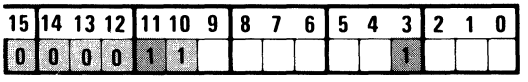
Skips the next instruction if the E-register content (extend bit) is a logic 0.

SLA SKIP IF LSB OF A IS ZERO



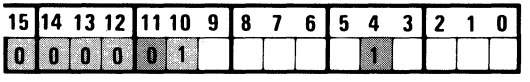
Skips the next instruction if the least-significant bit (bit 0) of the A-register is a logic 0; i.e., skips if an even number is in the A-register.

SLB SKIP IF LSB OF B IS ZERO



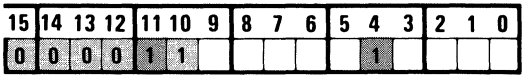
Skips the next instruction if the least-significant bit (bit 0) of the B-register is a logic 0; i.e., skips if an even number is in the B-register.

SSA SKIP IF SIGN OF A IS ZERO



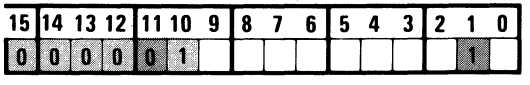
Skips the next instruction if the sign bit (bit 15) of the A-register is a logic 0; i.e., skips if a positive number is in the A-register.

SSB SKIP IF SIGN OF B IS ZERO



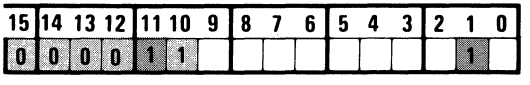
Skips the next instruction if the sign bit (bit 15) of the B-register is a logic 0; i.e., skips if a positive number is in the B-register.

SZA SKIP IF A IS ZERO



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

SZB SKIP IF B IS ZERO



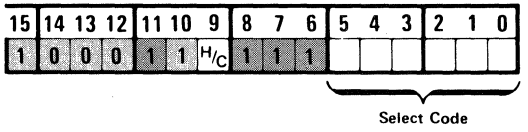
Skips the next instruction if the B-register contents are zero (16 zeros).

3-20. 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.

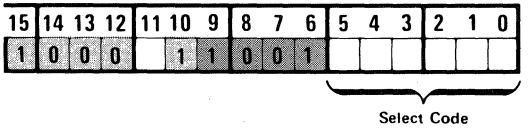
Bit 11, where relevant, specifies the A- or B-register or distinguishes between set control and clear control; otherwise, bit 11 may be a logic 0 or a logic 1 without affecting the instruction (although the assembler will assign zeros in this case). In those instructions where bit position 9 includes 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



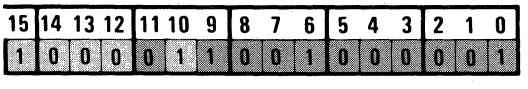
Clears the control bit of the selected I/O channel or function. This turns off the specific device channel and prevents it from interrupting. CLC 00 instruction clears all control bits from select code 06 upward, effectively turning off all I/O devices.

CLF CLEAR FLAG



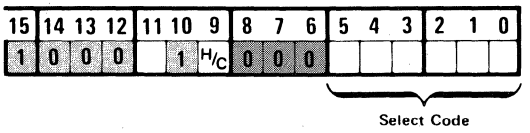
Clears the flag of the selected I/O channel or function. A CLF 00 instruction disables the interrupt system for all select codes except power fail (select code 04) and parity error (select code 05), which are always enables; this does not affect the status of the individual channel flags.

CLO CLEAR OVERFLOW



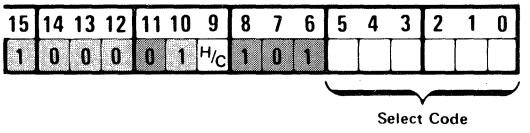
Clears the overflow bit.

HLT HALT



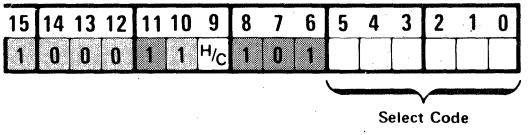
Halts the computer and holds or clears the flag of the selected I/O channel. The HLT instruction has the same effect as pressing the operator panel HALT pushbutton. The HLT instruction will be contained in the T-register, which is selected and displayed automatically when the computer halts. The P-register will contain the HLT location plus one.

LIA LOAD INTO A



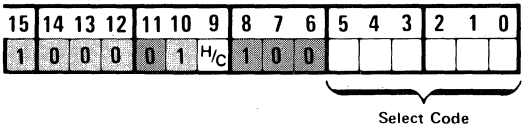
Loads the contents of the I/O buffer associated with the selected device into the A-register.

LIB LOAD INTO B



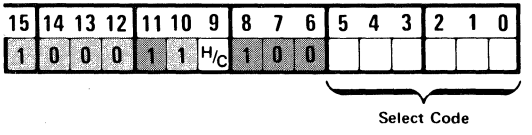
Loads the contents of the I/O buffer associated with the selected device into the B-register.

MIA MERGE INTO A



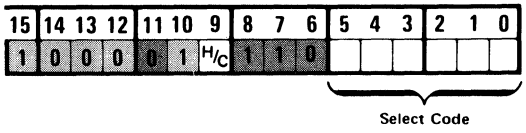
By executing a logical "inclusive or" function, merges the contents of the I/O buffer associated with the selected device into the A-register.

MIB MERGE INTO B



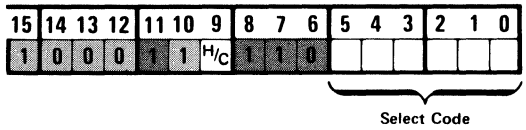
By executing a logical "inclusive or" function, merges the contents of the I/O buffer associated with the selected device into the B-register.

OTA OUTPUT A



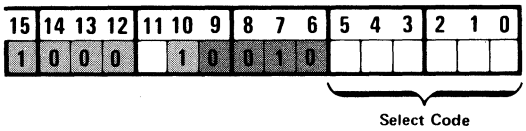
Outputs the contents of the A-register to the I/O buffer associated with the selected device. If the I/O buffer is less than 16 bits in length, the least-significant bits of the A-register are normally loaded. (Some exceptions to this exist, depending on the type of output device.) The contents of the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

OTB OUTPUT B



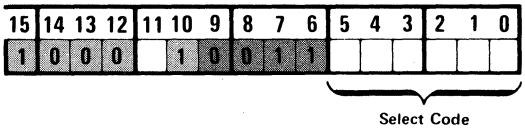
Outputs the contents of the B-register to the I/O buffer associated with the selected device. If the I/O buffer is less than 16 bits in length, the least-significant bits of the B-register are normally loaded. (Some exceptions to this exist, depending on the type of output device.) The contents of the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

SFC SKIP IF FLAG CLEAR



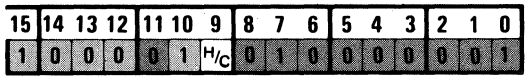
Skips the next programmed instruction if the flag of the selected channel is clear (device busy).

SFS SKIP IF FLAG SET



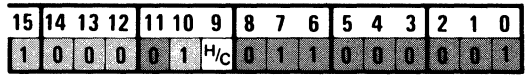
Skips the next programmed instruction if the flag of the selected channel is set (device ready). Used with "wait-for-flag" I/O programming, usually the interrupt system is off. If used with the interrupt system on, use a CLF instruction to clear the flag and eliminate the interrupt request.

SOC SKIP IF OVERFLOW CLEAR



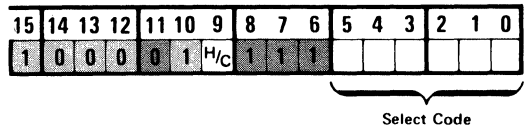
Skips the next programmed instruction if the overflow bit is clear. Use the H/C bit (bit 9) to either hold or clear the overflow bit following the completion of this instruction (whether the skip is taken or not).

SOS SKIP IF OVERFLOW SET



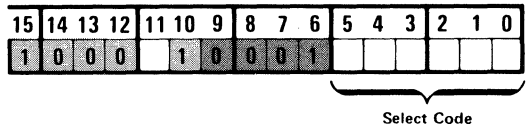
Skips the next programmed instruction if the overflow bit is set. Use the H/C bit (bit 9) to either hold or clear the overflow bit following the completion of this instruction (whether the skip is taken or not).

STC SET CONTROL



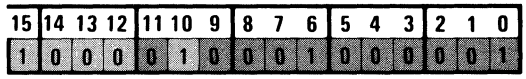
Sets the control bit of the selected I/O channel or function.

STF SET FLAG



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 04) which is always enabled and parity error (select code 05), which is selectively controllable.

STO SET OVERFLOW



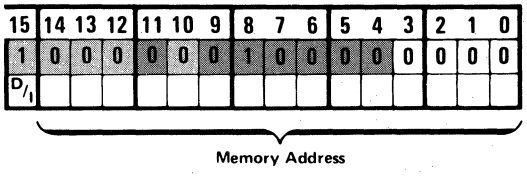
Sets the overflow bit.

3-21. 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.

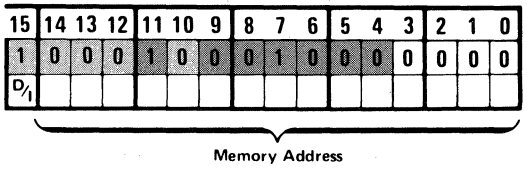
Since 15 bits are available for the address, these instructions can directly address any location in memory. As for all memory reference instructions, indirect addressing to any number of levels may also be used. A logic 0 in bit position 15 specifies direct addressing; a logic 1 specifies indirect addressing.

DIV DIVIDE



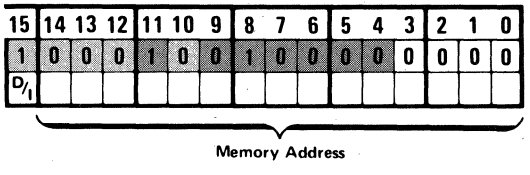
Divides a double-word integer in the combined B- and A-registers by a 16-bit integer in the addressed memory location. The result is a 16-bit integer quotient in the A-register and a 16-bit integer remainder in the B-register. Overflow can result from an attempt to divide by zero, or from an attempt to divide by a number too small for the dividend. In the former case (divide by zero), 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. In the latter case (divisor too small), the execution will be attempted with unpredictable results left in the B- and A-registers. If there is no divide error, the overflow bit is cleared.

DLD DOUBLE LOAD



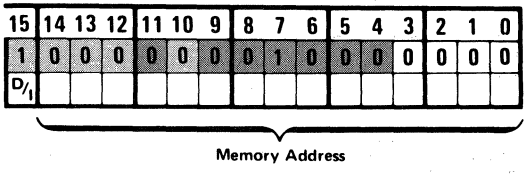
Loads the contents of addressed memory location m (and m + 1) into the A- and B-registers, respectively.

DST DOUBLE STORE



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

MPY MULTIPLY



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. The instruction clears the overflow bit.

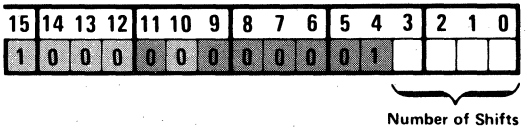
3-22. EXTENDED ARITHMETIC REGISTER REFERENCE INSTRUCTIONS

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 shirts is coded with four zeros, which essentially exchanges the contents of the B- and A-registers.

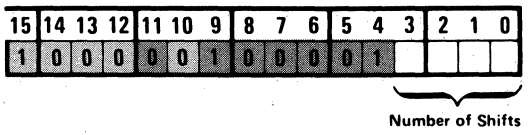
The extend bit is not affected by any of the following instructions. Except for the arithmetic shifts, overflow also is not affected.

ASL ARITHMETIC SHIFT LEFT



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".

ASR ARITHMETIC SHIFT RIGHT



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

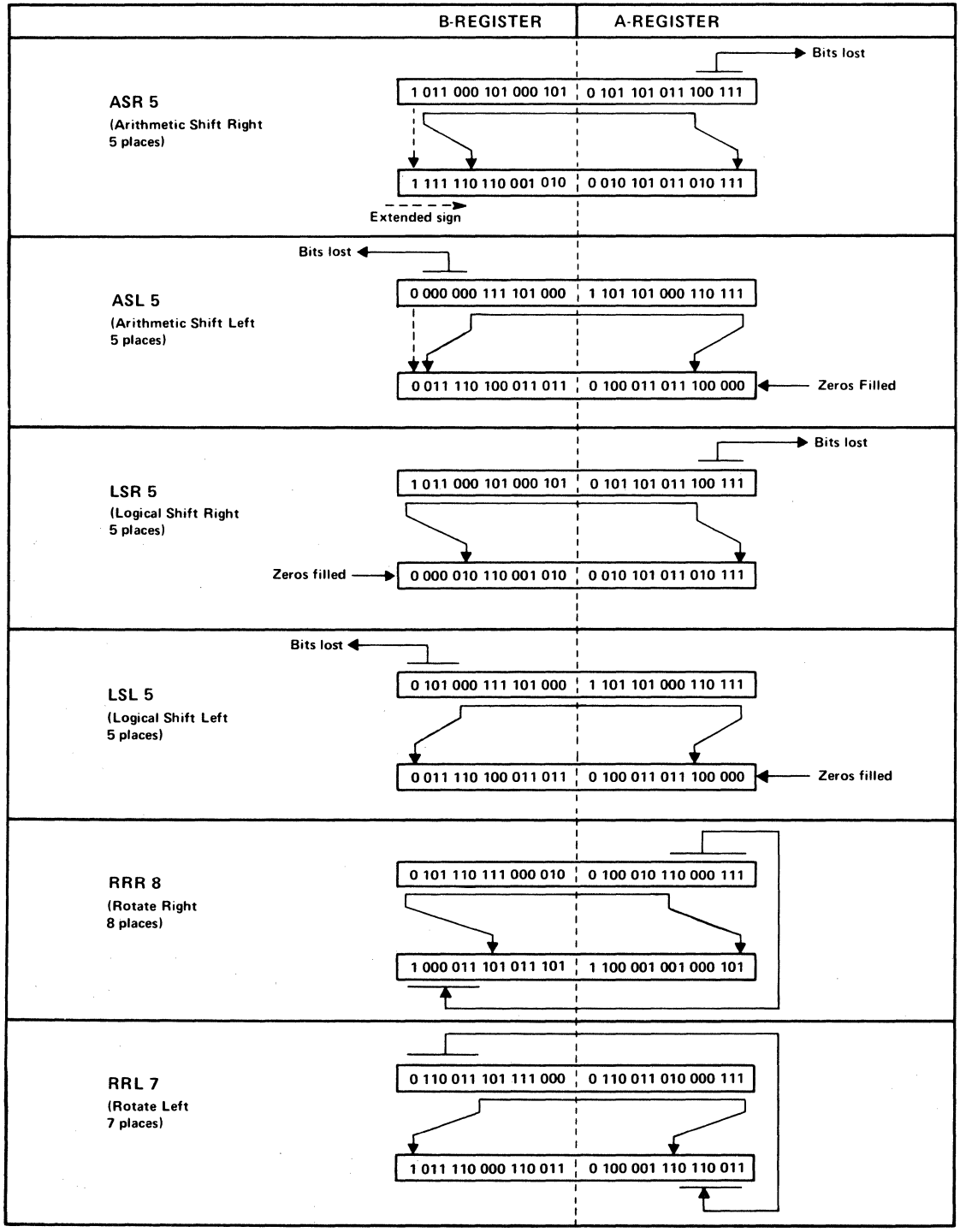
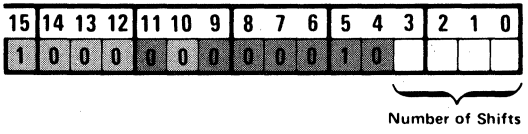


Figure 3-4. Examples of Double-Word Shifts and Rotates

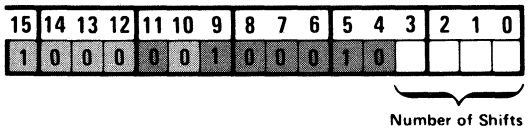
is extended into bit positions vacated by the right shift. Data bits shifted out of the least-significant end of the A-register are lost. Overflow cannot occur because the instruction clears the overflow bit.

LSL LOGICAL SHIFT LEFT



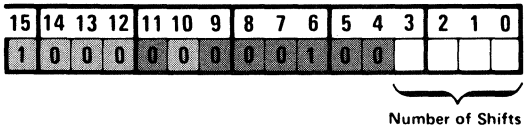
Logically shifts the combined contents of the B- and A-registers left n places. The value of n may be any number from 1 through 16. Zeros are filled into vacated low-order bit positions of the A-register; data bits are lost out of the high-order bit positions of the B-register.

LSR LOGICAL SHIFT RIGHT



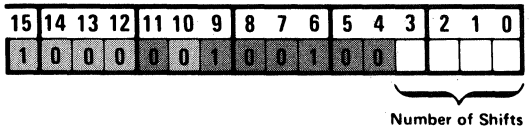
Logically shifts the combined contents of the B- and A-registers right n places. The value of n may be any number from 1 through 16. Zeros are filled into vacated high-order bit positions of the B-register; data bits are lost out of the low-order bit positions of the A-register.

RRL ROTATE LEFT



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.

RRR ROTATE RIGHT

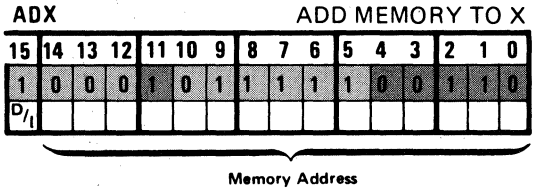


Rotates the combined contents of the B- and A-registers right n places. The value of n may be any number from 1 through 16. No bits are lost or filled in. Data bits shifted out of the low-order end of the A-register are rotated around to enter the high-order end of the B-register.

3-23. EXTENDED INSTRUCTION GROUP

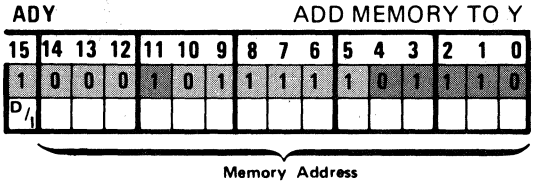
3-24. INDEX REGISTER INSTRUCTIONS. The index registers (X and Y) are two 16-bit registers accessible by the following instructions.

ADX ADD MEMORY TO X



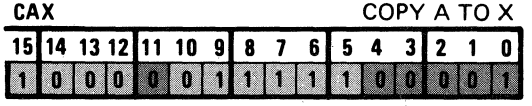
Adds the contents of the addressed memory location to the contents of the X-register. The sum remains in the X-register and the contents of the memory cell are unaltered. The result of this addition may set the extend bit or the overflow bit.

ADY ADD MEMORY TO Y



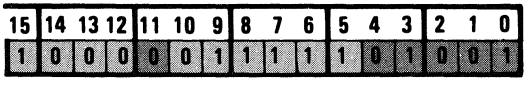
Adds the contents of the addressed memory location to the contents of the Y-register. The sum remains in the Y-register and the contents of the memory cell are unaltered. The result of this addition may set the extend bit or the overflow bit.

CAX COPY A TO X



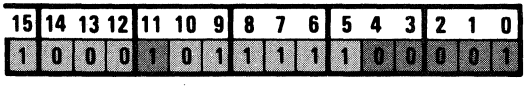
Copies the contents of the A-register into the X-register. The contents of the A-register are unaltered.

CAY COPY A TO Y



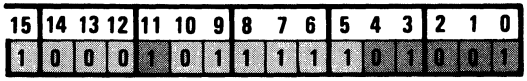
Copies the contents of the A-register into the Y-register. The contents of the A-register are unaltered.

CBX COPY B TO X



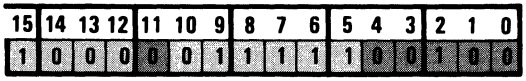
Copies the contents of the B-register into the X-register. The contents of the B-register are unaltered.

CBY COPY B TO Y



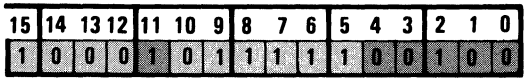
Copies the contents of the B-register into the Y-register. The contents of the B-register are unaltered.

CXA COPY X TO A



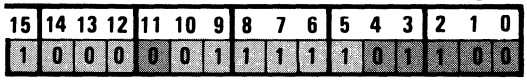
Copies the contents of the X-register into the A-register. The contents of the X-register are unaltered.

CXB COPY X TO B



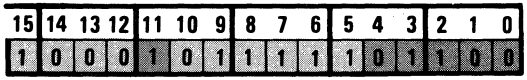
Copies the contents of the X-register into the B-register. The contents of the X-register are unaltered.

CYA COPY Y TO A



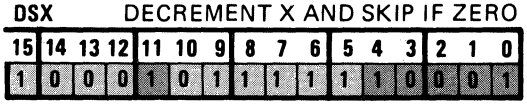
Copies the contents of the Y-register into the A-register. The contents of the Y-register are unaltered.

CYB COPY Y TO B



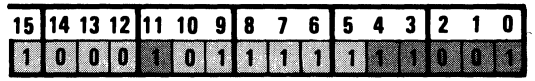
Copies the contents of the Y-register into the B-register. The contents of the Y-register are unaltered.

DSX DECREMENT X AND SKIP IF ZERO



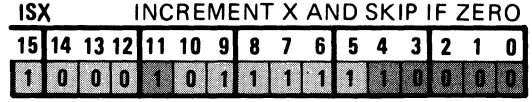
Subtracts one from the contents of the X-register. If the result of this operation is zero (X-register decremented from 000001 to 00000), the next instruction is skipped; i.e., the P-register count is advanced two counts instead of one count. If the result is not zero, the next sequential instruction is executed.

DSY DECREMENT Y AND SKIP IF ZERO



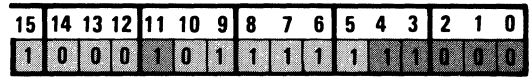
Subtracts one from the contents of the Y-register. If the result of this operation is zero (Y-register decremented from 000001 to 000000), the next instruction is skipped; i.e., the P-register count is advanced two counts instead of one count. If the result is not zero, the next sequential instruction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

ISX INCREMENT X AND SKIP IF ZERO



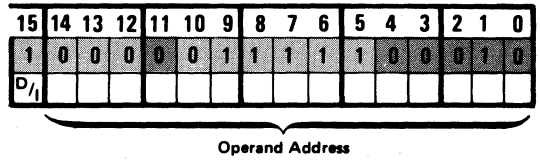
Adds one to the contents of the X-register. If the result of this operation is zero (X-register rolls over to 000000 from 177777), the next instruction is skipped; i.e., the P-register count is advanced two counts instead of one count. If the result is not zero, the next sequential instruction is executed.

ISY INCREMENT Y AND SKIP IF ZERO



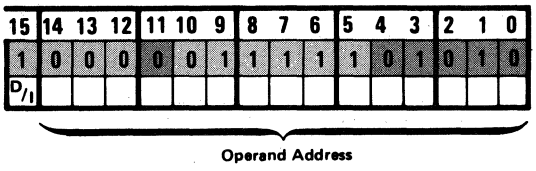
Adds one to the contents of the Y-register. If the result of this operation is zero (Y-register rolls over to 000000 from 177777), the next instruction is skipped; i.e., the P-register count is advanced two counts instead of one count. If the result is not zero, the next sequential instruction is executed.

LAX LOAD A INDEXED BY X



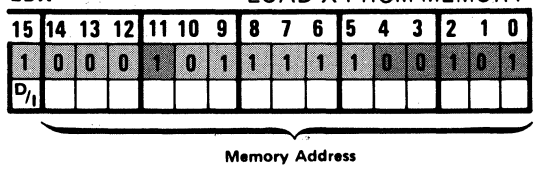
Loads the A-register with the contents indicated by the effective address, which is computed by adding the contents of the X-register to the operand address. The effective address is loaded into the M-register; the X-register and memory contents are not altered. Indirect addressing is resolved before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

LAY LOAD A INDEXED BY Y



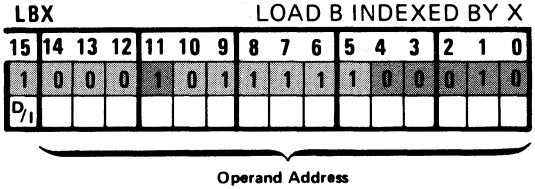
Loads the A-register with the contents indicated by the effective address, which is computed by adding the contents of the Y-register to the operand address. The effective address is loaded into the M-register; the Y-register and memory contents are not altered. Indirect addressing is resolved before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

LDX LOAD X FROM MEMORY



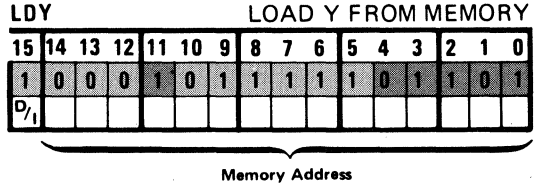
Loads the contents of the addressed memory location into the X-register. The A- and B-registers may be addressed as locations 000000 and 000001, respectively; however, if it is desired to load from the A- or B-register, copy instructions CAX or CBX should be used since they are more efficient.

LBX LOAD B INDEXED BY X



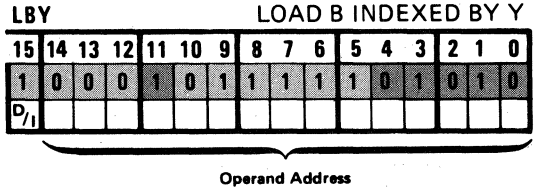
Loads the B-register with the contents indicated by the effective address, which is computed by adding the contents of the X-register to the operand address. The effective address is loaded into the M-register; the X-register and memory contents are not altered. Indirect addressing is resolved before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

LDY LOAD Y FROM MEMORY



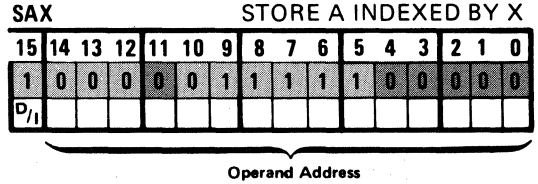
Loads the contents of the addressed memory location into the Y-register. The A- and B-registers may be addressed as locations 00000 and 00001, respectively; however, if it is desired to load from the A- or B-register, copy instructions CAY or CBY should be used since they are more efficient.

LBY LOAD B INDEXED BY Y



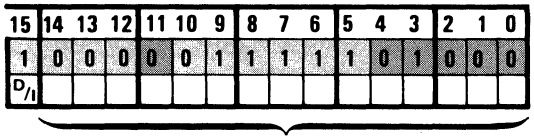
Loads the B-register with the contents indicated by the effective address, which is computed by adding the contents of the Y-register to the operand address. The effective address is loaded into the M-register, the X-register and memory contents are not altered. Indirect addressing is resolved before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

SAX STORE A INDEXED BY X



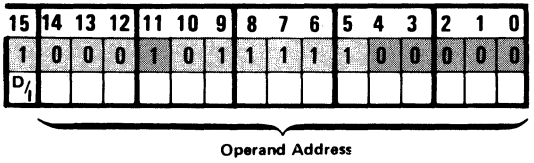
Stores the contents of the A-register into the location indicated by the effective address, which is computed by adding the contents of the X-register to the operand address. The effective address is loaded into the M-register; the A- and X-register contents are not altered. Indirect addressing is resolved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

SAY STORE A INDEXED BY Y



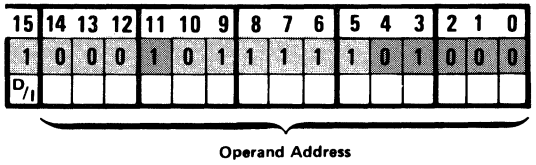
Stores the contents of the A-register into the location indicated by the effective address, which is computed by adding the contents of the Y-register to the operand address. The effective address is loaded into the M-register; the A- and Y-register contents are not altered. Indirect addressing is resolved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

SBX STORE B INDEXED BY X



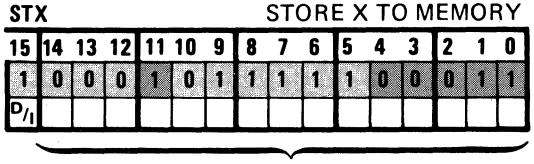
Stores the contents of the B-register into the location indicated by the effective address, which is computed by adding the contents of the X-register to the operand address. The effective address is loaded into the M-register; the B- and X-register contents are not altered. Indirect addressing is resolved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

SBY STORE B INDEXED BY Y



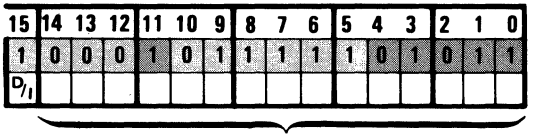
Stores the contents of the B-register into the location indicated by the effective address, which is computed by adding the contents of the Y-register to the operand address. The effective address is loaded into the M-register; the B- and Y-register contents are not altered. Indirect addressing is resolved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

STX STORE X TO MEMORY



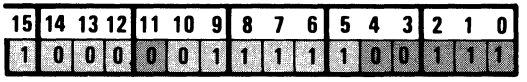
Stores the contents of the X-register into the addressed memory location. The A- and B-registers may be addressed as locations 00000 and 00001, respectively. The X-register contents are not altered.

STY STORE Y TO MEMORY



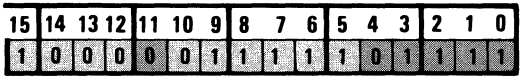
Stores the contents of the Y-register into the addressed memory location. The A- and B-registers may be addressed as locations 00000 and 00001, respectively. The Y-register contents are not altered.

XAX EXCHANGE A AND X



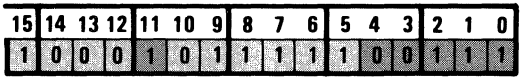
Exchanges the contents of the A- and X-registers.

XAY EXCHANGE A AND Y



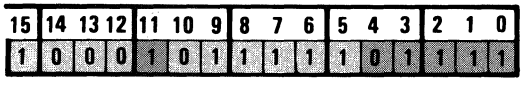
Exchanges the contents of the A- and Y-registers.

XBX EXCHANGE B AND X



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

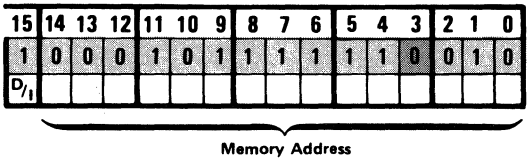
XBY EXCHANGE B AND Y



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

3-25. JUMP INSTRUCTIONS. The following two jump instructions involving the Y-register allow a program to either jump to or exit from a subroutine.

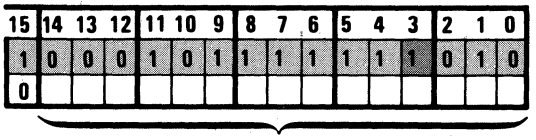
JLY JUMP AND LOAD Y



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

trol 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



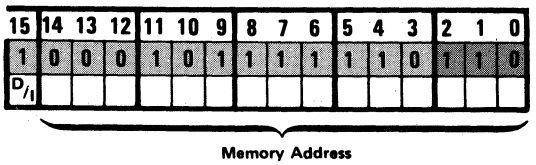
Transfers control to the effective address, which is computed by adding the contents of the Y-register to the operand address. Indirect addressing is not allowed. The effective address is loaded into the P-register; the Y-register contents are not altered. A memory protect check is performed by this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

3-26. 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



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 number of bytes to be compared is restricted to a positive integer greater than zero. The strings are compared one byte at a time; the ith byte in string 1 is compared with the ith 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:

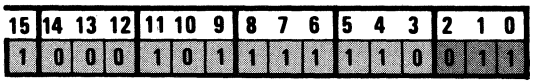
a. No skip if string 1 is equal to string 2; the P-register advances one count from Word 3 of the instruction. The A-register contains its original value incremented by the count stored in the address specified in Word 2.

b. 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.

c. 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.

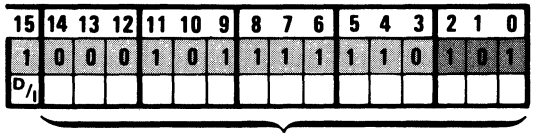
For all three exits, the B-register will contain its original value incremented by the count stored in the address specified in Word 2. This instruction is interruptible. The interrupt routine is expected to save and restore the contents of the A- and B-registers. During the interrupt, the remaining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

LBT LOAD BYTE



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.

MBT MOVE BYTES



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

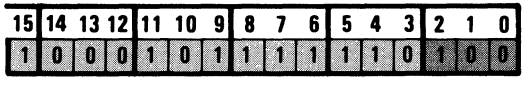
Word 3 = All-zeros word reserved for use by microcode.

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 greater than zero 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.

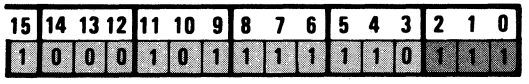
This instruction is interruptible. The interrupt routine is expected to save and restore the contents of the A- and B-registers. During the interrupt, the remaining count is stored in Word 3 of the instruction.

SBT STORE BYTE



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.

SFB SCAN FOR BYTE



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.

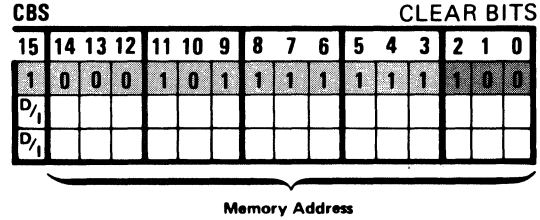
A string of bytes is scanned starting at the byte address given in the B-register. Scanning terminates when a byte in the string matches either the test byte or the termination byte in the A-register. The manner in which the instruction exits depends on which byte is matched first. If a byte in the string 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.

This instruction is interruptible. The interrupt routine is expected to save and restore the contents of the A- and B-registers.

3-27. BIT MANIPULATION INSTRUCTIONS. The following three instructions allow any number of bits in a specified memory location to be cleared, set, or tested.

CBS CLEAR BITS



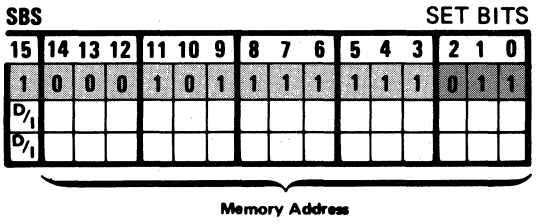
Clears bits in the addressed location. This is a three-word instruction where

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.



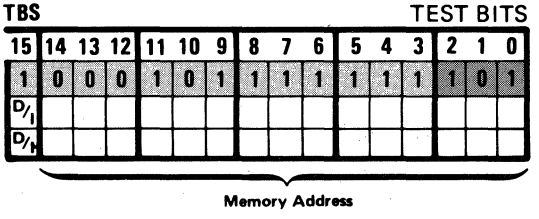
Sets bits in the addressed location. This is a three-word instruction where

Word 1 = Instruction code,

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

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

The bits to be set 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.



Tests (compares) bits in the addressed location. This is a three-word instruction where

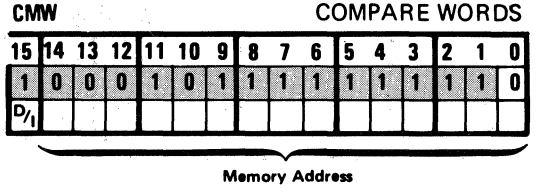
Word 1 = Instruction code,

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

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

The bits in the addressed memory word corresponding to logic 1's in the mask are tested. If all the bits tested are 1's, the instruction will not skip; otherwise the instruction will skip one word (i.e., the P-register will advance two counts from Word 3 of the instruction).

3-28. WORD MANIPULATION INSTRUCTIONS. The following instructions facilitate the comparing and moving of word arrays.



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

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 operand addresses are in the A- and B-registers. The A-register contains the first word address of array 1 and the B-register contains the first word address of array 2. Bit 15 of the addresses in the A- and B-registers are ignored; i.e., no indirect addressing allowed.

The number of words to be compared is given in the memory location addressed by Word 2 of the instruction; the number of words to be compared is restricted to a positive integer greater than zero. The arrays are compared one word at a time; the ith word in array 1 is compared with the ith 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:

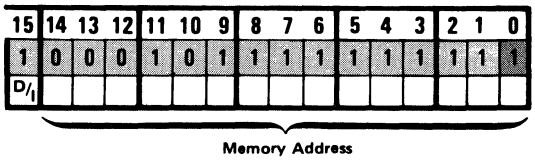
a. No skip if array 1 is equal to array 2; the P-register advances one count from Word 3 of the instruction. The A-register contains its original value incremented by the word count stored in the address specified in Word 2.

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 Word 3 of the instruction.

MVW MOVE WORDS



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 greater than zero 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.

3-29. FLOATING POINT INSTRUCTIONS

The floating point instructions allow addition, subtraction, multiplication, and division of both single precision (32-bit) and extended precision (48-bit) floating point quantities, and conversion of quantities from floating point format to integer format or vice versa. Data formats are shown in Figure 3-1. Except for zero, all floating point operands must be normalized (i.e., sign of mantissa differs from most significant bit of mantissa).

For multiple-word instructions, indirect addressing to any number of levels is permitted for the word(s) indicated as memory address. A logic 0 in bit position 15 specifies direct addressing; a logic 1 specifies indirect addressing.

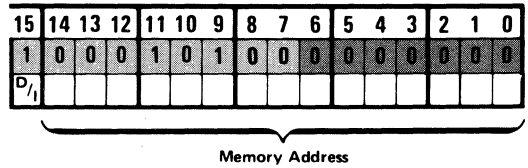
After initiating an operation in the Floating Point Processor (FPP), the floating point firmware waits for FPP completion. If completion is not signalled by the FPP within 25 microseconds, an FPP malfunction exists. The firmware indicates this condition by creating a memory protect violation (memory protect installed), setting the overflow flag, and setting the A-register to all ones (an unnormalized number).

The execution times of the floating point instructions are specified under paragraph 3-32. 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.

Information required for direct user-microprogramming utilizing the Floating Point Processor is provided in the *HP 1000 E/F-Series Microprogramming Reference Manual*, part no. 02109-90004.

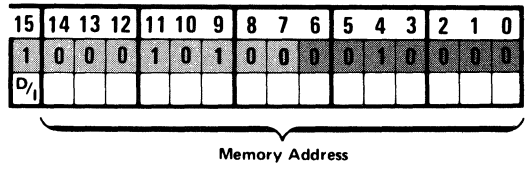
3-30. SINGLE PRECISION OPERATIONS. Overflow for single precision operations occurs if the result lies outside the range of representable single precision floating point numbers $[-2^{127},(1-2^{-23})2^{127}]$. In such a case, the overflow flag is set and the result $(1-2^{-23})\,2^{127}$ is returned to the A- and B-registers. Underflow occurs if the result lies inside the range $[-2^{-129}(1+2^{-22}),2^{-129}]$. In such a case, the overflow flag is set and the result 0 is returned to the A- and B-registers.

FAD FLOATING POINT ADD



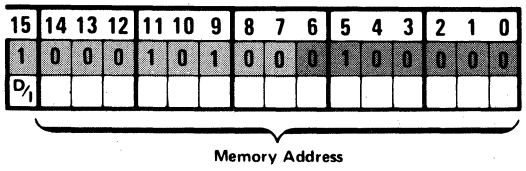
Adds the floating point quantity in the A- and B-registers to the floating point quantity in the specified memory locations. The floating point result is returned to the A- and B-registers.

FSB FLOATING POINT SUBTRACT



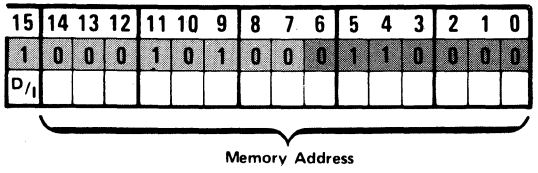
Subtracts the floating point quantity in the specified memory locations from the floating point quantity in the A- and B-registers. The floating point result is returned to the A- and B-registers.

FMP FLOATING POINT MULTIPLY



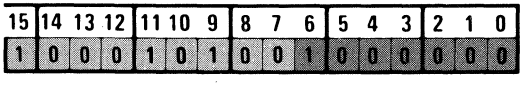
Multiplies the floating point quantity in the A- and B-registers by the floating point quantity in the specified memory locations. The floating point result is returned to the A- and B-registers.

FDV FLOATING POINT DIVIDE



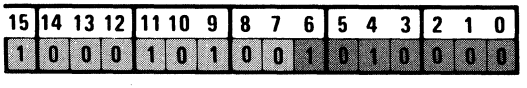
Divides the floating point quantity in the A- and B-registers by the floating point quantity in the specified memory locations. The floating point result is returned to the A- and B-registers.

FIX FLOATING POINT TO SINGLE INTEGER



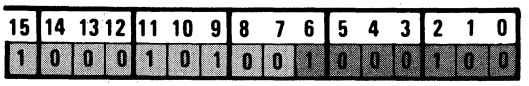
Converts the floating point quantity in the A- and B-registers to single integer format. The integer result is returned to the A-register. If the magnitude of the floating point number is <1, regardless of sign, the integer 0 is returned. If the magnitude of the exponent of the floating point number is ⩾16, regardless of sign, the integer 32767 (077777 octal) is returned as the result and the overflow flag is set.

FLT SINGLE INTEGER TO FLOATING POINT



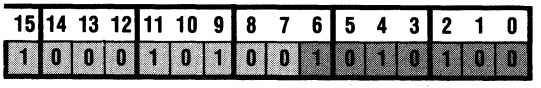
Converts the single integer quantity in the A-register to single precision floating point format. The floating point result is returned to the A- and B-registers.

.FIXD (F-SERIES ONLY)* FLOATING POINT TO DOUBLE INTEGER



Converts the floating point quantity in the A- and B-registers to double integer format. The integer result is returned to the A- and B-registers. (The A-register contains the most-significant word and the B-register contains the least-significant word.) If the magnitude of the floating point number is <1, regardless of sign, the integer 0 is returned. If the magnitude of the floating point number is ⩾32, regardless of sign, the integer $2^{31}-1$ is returned as the result and the overflow flag is set.

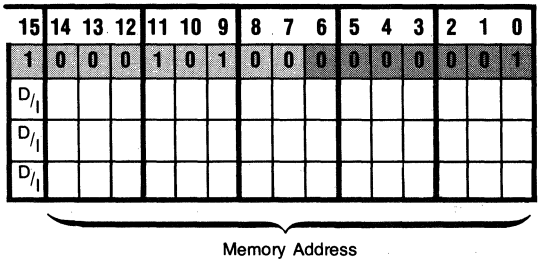
.FLTD (F-SERIES ONLY)* DOUBLE INTEGER TO FLOATING POINT



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

3-31. EXTENDED PRECISION OPERATIONS. (F-SERIES ONLY) Overflow for extended precision operations occurs if the result lies outside the range of representable extended precision floating point numbers [-2^{127}, $(1-2^{-39})\,2^{+127}$]. In such a case, the overflow flag is set and $(1-2^{-39})\,2^{127}$ is returned as the result. Underflow occurs if the result lies inside the range [$-2^{-129}\,(1+2^{-38})$, 2^{-129}]. In such a case, the overflow flag is set and 0 is returned as the result.

.XADD* EXTENDED FLOATING POINT ADD

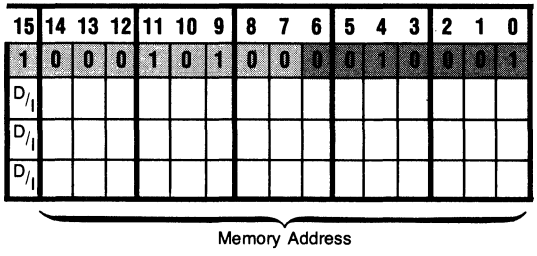


Adds two extended precision floating point quantities (augend plus addend). This is a four-word instruction where

Word 1 = Instruction code.
Word 2 = Address of result.
Word 3 = Address of augend.
Word 4 = Address of addend.

*For HP Assembly Language usage, refer to paragraph 3-43.

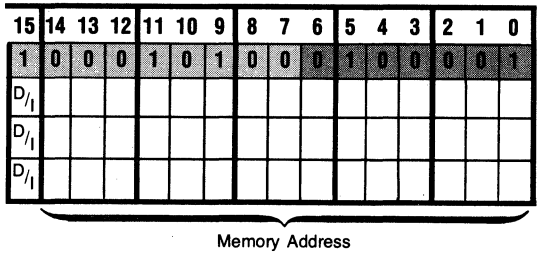
.XSUB* EXTENDED FLOATING POINT SUBTRACT



Subtracts one extended precision floating point quantity from another (minuend minus subtrahend). This is a four-word instruction where

Word 1 = Instruction code.
Word 2 = Address of result.
Word 3 = Address of minuend.
Word 4 = Address of subtrahend.

.XMPY* EXTENDED FLOATING POINT MULTIPLY

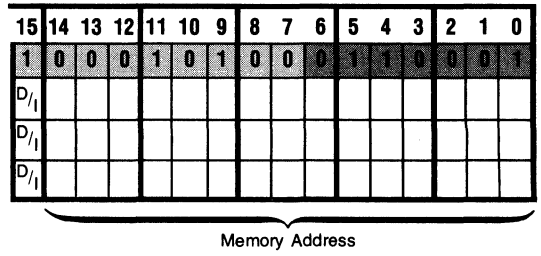


Multiplies one extended precision floating point quantity by another (multiplicand by multiplier). This is a four-word instruction where

Word 1 = Instruction code.
Word 2 = Address of result.
Word 3 = Address of multiplicand.
Word 4 = Address of multiplier.

*For HP Assembly Language usage, refer to paragraph 3-43.

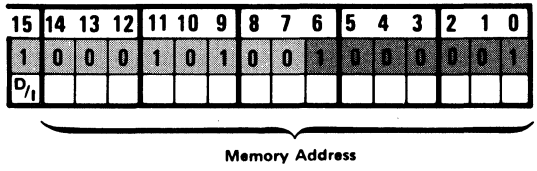
.XDIV* EXTENDED FLOATING POINT DIVIDE



Divides one floating point quantity by another (dividend by divisor). This is a four-word instruction where

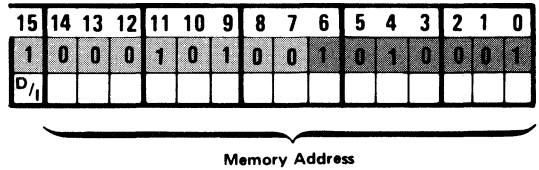
Word 1 = Instruction code.
Word 2 = Address of result.
Word 3 = Address of divident.
Word 4 = Address of divisor.

.XFXS* EXTENDED FLOATING POINT TO SINGLE INTEGER



Converts the extended precision floating point quantity in the specified memory locations to single integer format. The integer result is returned to the A-register. If the magnitude of the floating point number is <1, regardless of sign, 0 is returned as the result. If the magnitude of the exponent of the floating point number is ≥ 16, regardless of sign, the integer $2^{15}-1$ is returned as the result and the overflow flag is set.

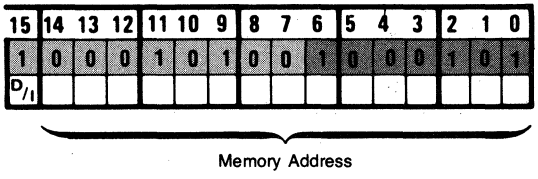
.XFTS* SINGLE INTEGER TO EXTENDED FLOATING POINT



Converts the single integer quantity in the A-register to extended precision floating point format. The floating point result is returned to the specified memory locations.

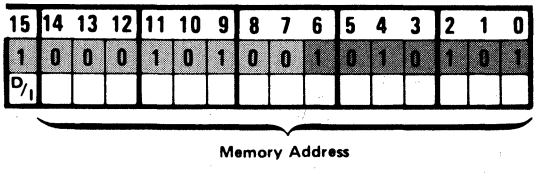
*For HP Assembly Language usage, refer to paragraph 3-43.

.XFXD* EXTENDED FLOATING POINT TO DOUBLE INTEGER



Converts the extended precision floating point quantity in the specified memory locations to double integer format. The integer result is returned to the A- and B-registers. (The A-register contains the most-significant word and the B-register contains the least-significant word.) If the magnitude of the floating point number is <1, regardless of sign, 0 is returned as the result. If the magnitude of the exponent of the floating point number is ≥ 32, regardless of sign, the integer $2^{31}-1$ is returned as the result and the overflow flag is set.

.XFTD* DOUBLE INTEGER TO EXTENDED FLOATING POINT

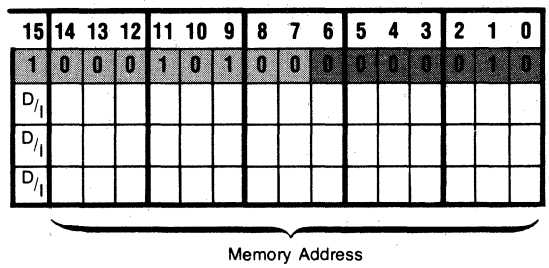


Converts the double integer quantity in the A- and B-registers to extended precision floating point format. The floating point result is returned to the specified memory locations.

3-32. DOUBLE PRECISION OPERATIONS. (F-SERIES ONLY) Overflow for double precision operations occurs if the result lies outside the range of representable double precision floating point numbers $[-2^{127}, (1-2^{-55})\ 2^{-127}]$. In such a case, the overflow flag is set and $(1-2^{-55})\ 2^{127}$ is returned as the result. Underflow occurs if the result lies inside the range $[-2^{-129}\ (1+2^{-54}), 2^{-129}]$. In such a case, the overflow flag is set and 0 is returned as the result.

*For HP Assembly Language usage, refer to paragraph 3-43.

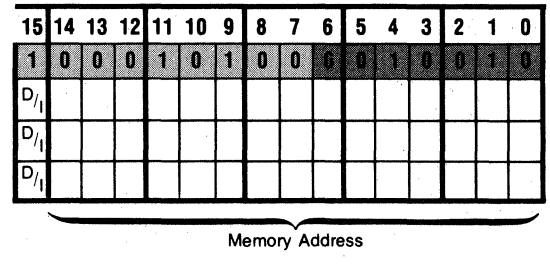
.TADD* DOUBLE FLOATING POINT ADD



Adds two double precision floating point quantities (augend plus addend). This is a four-word instruction where

Word 1 = Instruction code.
Word 2 = Address of result.
Word 3 = Address of augend.
Word 4 = Address of addend.

.TSUB* DOUBLE FLOATING POINT SUBTRACT

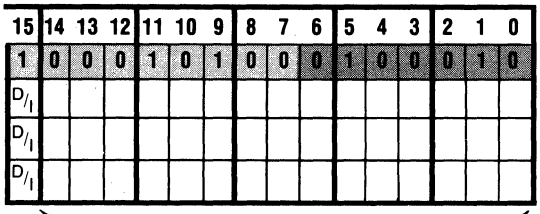


Subtracts one double precision floating point quantity from another (minuend minus subtrahend). This is a four-word instruction where

Word 1 = Instruction code.
Word 2 = Address of result.
Word 3 = Address of minuend.
Word 4 = Address of subtrahend.

*For HP Assembly Language usage, refer to paragraph 3-43.

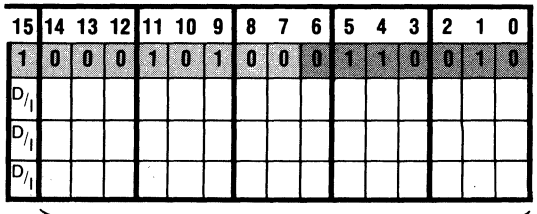
.TMPY* DOUBLE FLOATING POINT MULTIPLY



Multiplies one double precision floating point quantity by another (multiplicand by multiplier). This is a four-word instruction where

Word 1 = Instruction code.
Word 2 = Address of result.
Word 3 = Address of multiplicand.
Word 4 = Address of multiplier.

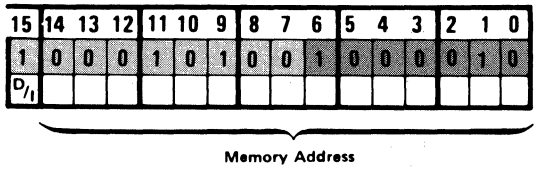
.TDIV* DOUBLE FLOATING POINT DIVIDE



Divides one double precision floating point quantity by another (dividend by divisor). This is a four-word instruction where

Word 1 = Instruction code.
Word 2 = Address of result.
Word 3 = Address of divident.
Word 4 = Address of divisor.

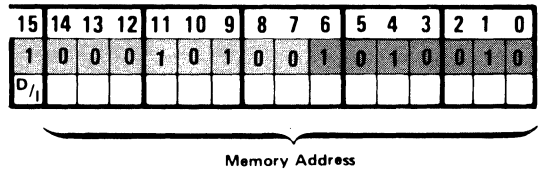
.TFXS* DOUBLE FLOATING POINT TO SINGLE INTEGER



Converts the double precision floating point quantity in the specified memory locations to single integer format. The integer result is returned to the A-register. If the magnitude of the floating point number is <1, regardless of sign, 0 is returned as the result. If the magnitude of the exponent of the floating point number is ≥16, regardless of sign, the integer $2^{15}-1$ is returned as the result and the overflow flag is set.

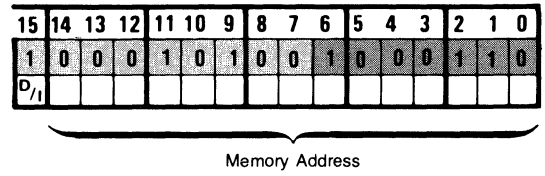
*For HP Assembly Language usage, refer to paragraph 3-43.

.TFTS* SINGLE INTEGER TO DOUBLE FLOATING POINT



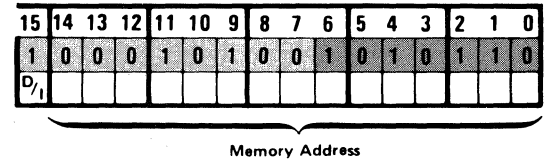
Converts the single integer quantity in the A-register to double precision floating point format. The floating point result is returned to the specified memory locations.

.TFXD* DOUBLE FLOATING POINT TO DOUBLE INTEGER



Converts the double precision floating point quantity in the specified memory locations to double integer format. The integer result is returned to the A- and B-registers. (The A-register contains the most-significant word and the B-register contains the least-significant word.) If the magnitude of the floating point number is <1, regardless of sign, 0 is returned as the result. If the magnitude of the exponent of the floating point number is ≥32, regardless of sign, the integer $2^{31}-1$ is returned as the result and the overflow flag is set.

.TFTD* DOUBLE INTEGER TO DOUBLE FLOATING POINT



Converts the double integer quantity in the A- and B-registers to double precision floating point format. The floating point result is returned to the specified memory locations.

3-33. INSTRUCTION EXECUTION TIMES

Tables 3-5A, 3-5B and 3-5C list the execution times required for the various base set instructions on the M-, E- and F-Series Computers, respectively.

*For HP Assembly Language usage, refer to paragraph 3-43.

Table 3-5A. Typical Base Set Instruction Execution Times (M-Series)

INSTRUCTION	EXECUTION TIME (μS)
Memory Reference Group[1,2]	
ADA/B, AND, IOR, LDA/B, XOR	1.94
STA/B	2.27
CPA/B (no skip)	2.27
(skip)	2.59
ISZ (no skip)	2.59
(skip)	2.92
JMP	1.94
JSB	2.27
Shift/Rotate Group[3]	2.59 - 2.92
Alter/Skip Group[3]	
No skip, no increment	2.59
No skip, increment A/B	2.92
Skip, no increment	2.59
Skip, increment A/B	2.92
Input/Output Group[4]	2.59 - 3.89
Extended Arithmetic Group[5]	
ASL, ASR, LSL, LSR, RRL, RRR	3.57 - 8.43
DLD	4.54
DST	4.86
MPY	12.32 - 13.30
DIV	15.92 - 18.20
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

INSTRUCTION	EXECUTION TIME (μS)
Extended Instruction Group	
CAX, CBX, CAY, CBY CXA, CXB, CYA, CYB	2.275
XAX, XBX, XAY, XBY ISX, ISY, DSX, DSY	3.250
LDX, LDY	
(direct address)	4.875
(indirect address)	4.875[a]
STX, STY	
(direct address)	5.20
(indirect address)	5.20[a]
LAX, LBX, LAY, LBY	
(direct address)	4.875
(indirect address)	5.525[a]
SAX, SBX, SAY, SBY	
(direct address)	5.20
(indirect address)	5.85[a]
ADX, ADY	
(direct address)	4.875
(indirect address)	4.875[a]
JLY (direct address)	5.525
(indirect address)	5.525[a]
JPY	4.55
LBT	4.875 avg
SBT	6.01 avg
MBT	8.775[a,b,g]
MVW	7.8[a,c,g]
CBT	8.775[a,d,g]
CMW	7.8[a,e,g]
SFB (for test byte match)	3.575[f,g]
(for term. byte match)	2.275[f,g]
CBS, SBS	7.8[a]
TBS	8.125[a]

[1] Memory refresh consumes 0.65 μS maximum no more often than every 30 μS.

[2] Add 1.3 μS for each indirect address level.

[3] NOP or RSS requires 2.92 μS whereas a JMP *+1 or JMP *+2 requires only 1.94 μS.

[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 μS for each indirect address level.

b. Add 7.31 μS for each byte moved or compared.

c. Add 3.25 μS for each word moved or compared.

d. Add 8.125 μS for each byte moved or compared.

e. Add 3.575 μS for each word moved or compared.

f. Add 4.875 μS for each byte moved or compared.

g. Add 7.15 μS for each interrupt of the instruction.

Table 3-5B. Typical Base Set Instruction Execution Times[1] (E-Series)

INSTRUCTION	EXECUTION TIME (us)				
	Standard Performance Memory		High Performance Memory		Fault Control Memory
	Non-DMS	DMS	Non-DMS	DMS	DMS
Memory Reference Group					
ADA/B,AND,IOR,LDA/B,XOR	1.190	1.330	0.910	0.910	1.400
1st Indirect Level	0.595	0.665	0.455	0.455	0.700
Each Additional Level	0.630	0.665	0.630	0.630	0.700
STA/B	1.855	1.995	1.260	1.330	2.030
1st Indirect Level	0.595	0.665	0.455	0.455	0.700
Each Additional Level	0.630	0.665	0.630	0.630	0.700
CPA/B (no skip)	1.225	1.330	1.085	1.085	1.400
(skip)	1.715	1.855	1.435	1.435	1.925
1st Indirect Level	0.595	0.665	0.455	0.455	0.700
Each Additional Level	0.630	0.665	0.630	0.630	0.700
ISZ (no skip)	2.030	2.170	1.540	1.540	2.205
(skip)	2.030	2.170	1.610	1.610	2.205
1st Indirect Level	0.595	0.665	0.455	0.455	0.700
Each Additional Level	0.630	0.665	0.630	0.630	0.700
JMP	0.735	0.735	0.735	0.735	0.735
1st Indirect Level	0.595	0.665	0.350	0.420	0.700
2nd Indirect Level	0.595	0.665	0.560	0.490	0.700
Each Additional Level	1.190	1.330	0.805	0.875	1.400
JSB	1.855	1.925	1.610	1.680	2.030
1st Indirect Level	0.595	0.665	0.350	0.420	0.700
2nd Indirect Level	0.595	0.665	0.560	0.490	0.700
Each Additional Level	1.190	1.330	0.805	0.875	1.400
Shift/Rotate Group					
(no skip)	1.190	1.330	0.910	0.910	1.400
(skip)	1.785	1.995	1.260	1.330	2.100
Alter/Skip Group					
(no skip)	1.190	1.330	0.910	0.910	1.400
(skip)	1.785	1.995	1.260	1.330	2.100
Input/Output Group[2]					
SFS, SFC, SOS, SOC					
(no skip)	1.575-2.275	1.575 to 2.275	1.575 to 2.275	1.575 to 2.275	1.575 to 2.275
(skip)	2.030-2.730	2.170 to 2.870	1.960 to 2.660	1.960 to 2.660	2.240 to 2.940
Extended Arithmetic Group					
ASL	2.065 +0.175/ shift	2.135 +0.175/ shift	1.820 +0.175/ shift	1.890 +0.175/ shift	2.170 +0.175/ shift
ASR	1.610 +0.175/ shift	1.680 +0.175/ shift	1.470 +0.175/ shift	1.470 +0.175/ shift	1.715 +0.175/ shift

Table 3-5B. Typical Base Set Instruction Execution Times[1] (E-Series) (Continued)

INSTRUCTION	EXECUTION TIME (us)				
	Standard Performance Memory		High Performance Memory		Fault Control Memory
	Non-DMS	DMS	Non-DMS	DMS	DMS
Extended Arithmetic Group (Cont.)					
LSL,LSR,RRL,RRR	1.715 +0.175/ shift	1.785 +0.175/ shift	1.470 +0.175/ shift	1.540 +0.175/ shift	1.820 +0.175/ shift
DLD	3.185	3.500	2.065	2.135	3.675
DST	3.710	3.990	2.695	2.765	4.060
MPY	5.740 to 6.720	5.985 to 7.035	5.320 to 6.055	5.320 to 6.125	6.125 to 7.210
DIV	8.085 to 9.625	8.295 to 9.940	7.665 to 9.065	7.665 to 9.065	8.400 to 10.115
Floating Point Group					
FAD	13.125 to 27.650	13.300 to 27.825	12.195 to 27.440	12.985 to 27.510	13.405 to 27.930
FDV	34.020 to 47.320	34.195 to 47.495	33.810 to 47.110	33.880 to 47.180	34.300 to 47.600
FIX	4.302 to 7.595	4.375 to 7.665	4.060 to 7.350	4.130 to 7.420	4.410 to 7.700
FLT	13.825 to 29.435	14.000 to 29.610	13.615 to 29.225	13.685 to 29.295	14.105 to 29.715
FMP	25.480 to 35.105	25.655 to 35.280	25.270 to 34.895	25.340 to 34.965	25.760 to 35.385
FSB	13.825 to 29.435	14.000 to 29.610	13.615 to 29.225	13.685 to 29.295	14.105 to 29.715
Extended Instruction Group (Index Register Instructions)					
CAX,CBX,CAY,CBY CXA,CXB,CYA,CYB	1.435	1.505	1.295	1.295	1.540
XAX,XBX,XAY,XBY	2.065	2.170	1.925	1.925	2.240
DSX,DSY (no skip)	2.030	2.170	1.750	1.750	2.240
(skip)	2.520	2.730	1.995	2.065	2.835
ISZ,ISY (no skip)	2.030	2.170	1.750	1.750	2.240
(skip)	2.520	2.730	1.750	1.750	2.835
LDX,LDY	3.045	3.185	2.660	2.730	3.255
Each Indirect Level	1.190	1.330	0.805	0.875	1.400
STX,STY	3.430	3.570	2.940	2.940	3.605
Each Indirect Level	1.190	1.330	0.805	0.875	1.400
LAX,LBX,LAY,LBY	3.745	3.885	3.185	3.255	3.955
Each Indirect Level	1.190	1.330	0.805	0.875	1.400
SAX,SAY,SBX,SBY	3.815	3.885	3.465	3.465	3.885
Each Indirect Level	1.190	1.330	0.805	0.875	1.400
ADX,ADY	2.975	3.045	2.730	2.800	3.080
Each Indirect Level	1.190	1.330	0.805	0.875	1.400
JLY	2.800	2.905	2.660	2.660	2.975
Each Indirect Level	1.190	1.330	0.805	0.875	1.400
JPY	2.625	2.835	2.275	2.275	2.940

Table 3-5B. Typical Base Set Instruction Execution Times[1] (E-Series) (Continued)

INSTRUCTION	EXECUTION TIME (us)				
	Standard Performance Memory		High Performance Memory		Fault Control Memory
	Non-DMS	DMS	Non-DMS	DMS	DMS
Extended Instruction Group (Cont.)					
(Byte Manipulation Instructions)					
LBT (from high byte)	3.780	3.885	3.640	3.640	3.955
(from low byte)	3.500	3.605	3.360	3.360	3.675
SBT (to high byte)	4.830	4.970	4.340	4.410	5.005
(to low byte)	4.445	4.620	3.885	4.025	4.655
MBT	4.270 +4.235/ byte	4.515 +4.340/ byte	3.745 +4.045/ byte	3.815 +4.080/ byte	4.655 +4.410/ byte
Each Indirect Level	1.190	1.330	0.805	0.875	1.400
CBT	4.270 +4.445/ byte	4.515 +4.550/ byte	3.745 +4.480/ byte	3.815 +4.480/ byte	4.655 +4.620/ byte
Each Indirect Level	1.190	1.330	0.805	0.875	1.400
SFB (if compare exit)	2.170 +2.735/ byte	2.240 +2.770/ byte	1.925 +2.735/ byte	1.995 +2.735/ byte	2.275 +2.805/ byte
(if terminal exit)	2.590 +2.735/ byte	2.660 +2.770/ byte	2.450 +2.735/ byte	2.450 +2.735/ byte	2.695 +2.805/ byte
(Word Manipulation Instructions)					
CMW	4.270 +2.870/ word	4.515 +3.010/ word	3.745 +2.380/ word	3.815 +2.520/ word	4.655 +3.080/ word
Each Indirect Level	1.190	1.330	0.805	0.875	1.400
MVW	4.270 +1.750/ word	4.515 +1.820/ word	3.745 +1.680 word	3.815 +1.680 word	4.655 +1.855/ word
Each Indirect Level	1.190	1.330	0.805	0.875	1.400
(Bit Manipulation Instructions)					
CBS,SBS	5.215	5.425	4.480	4.550	5.495
Each Indirect Level	1.190	1.330	0.805	0.875	1.400
TBS (no skip)	5.355	5.565	4.935	4.935	5.670
(skip)	5.670	5.950	4.725	4.795	6.090
Each Indirect Level	1.190	1.330	0.805	0.875	1.400

[1] Semiconductor memory refresh may increase program execution times by up to 3%.

[1] Depends on which I/O time period (T2,3,4,5,6) the instruction begins.

Table 3-5C. Typical Base Set Instruction Execution Times[1]

INSTRUCTION	EXECUTION TIME (us)			
	High Performance Memory		High Performance Fault Control Memory	
	Non-DMS	DMS	Non-DMS	DMS
Memory Reference Group				
ADA/B,AND,IOR,LDA/B,XOR	0.910	0.910	0.910	0.980
1st Indirect Level	0.455	0.455	0.455	0.490
Each Additional Level	0.630	0.630	0.630	0.630
STA/B	1.260	1.260	1.330	1.400
1st Indirect Level	0.455	0.455	0.455	0.490
Each Additional Level	0.630	0.630	0.630	0.630
CPA/B (no skip)	1.085	1.085	1.085	1.120
(skip)	1.435	1.435	1.435	1.505
1st Indirect Level	0.455	0.455	0.455	0.490
Each Additional Level	0.630	0.630	0.630	0.630
ISZ (no skip)	1.540	1.540	1.540	1.610
(skip)	1.610	1.610	1.610	1.680
1st Indirect Level	0.455	0.455	0.455	0.490
Each Additional Level	0.630	0.630	0.630	0.630
JMP	0.735	0.735	0.735	0.735
1st Indirect Level	0.350	0.350	0.350	0.385
2nd Indirect Level	0.560	0.560	0.560	0.595
Each Additional Level	0.805	0.875	0.875	0.980
JSB	1.260	1.260	1.330	1.400
1st Indirect Level	0.350	0.420	0.420	0.490
2nd Indirect Level	0.560	0.490	0.490	0 490
Each Additional Level	0.805	0.875	0.875	0.980
Shift/Rotate Group				
(no skip)	0.910	0.910	0.910	0.980
(skip)	1.260	1.330	1.330	1.470
Alter/Skip Group				
(no skip)	0.910	0.910	0.910	0.980
(skip)	1.260	1.330	1.330	1.470
Input/Output Group[2]				
SFS, SFC, SOS, SOC				
(no skip)	1.575	1.575	1.575	1.575
(skip)	1.855	1.855	1.855	1.890
Extended Arithmetic Group				
ASL	1.820	1.890	1.890	1.960
	+0.175/shift	+0.175/shift	+0.175/shift	+0.175/shift
ASR	1.470	1.470	1.470	1.505
	+0.175/shift	+0.175/shift	+0.175/shift	+0.175/shift
LSL,LSR,RRL,RRR	1.470	1.540	1.540	1.610
	+0.175/shift	+0.175/shift	+0.175/shift	+0.175/shift
DLD	2.520	2.590	2.590	2.765
DST	2.695	2.695	2.765	2.905
MPY	5.320 to 6.055	5.320 to 6.125	5.320 to 6.125	5.425 to 6.300
DIV	7.665 to 9.065	7.665 to 9.065	7.665 to 9.065	7.770 to 9.205

Table 3-5C. Typical Base Set Instruction Execution Times[1] (Continued)

INSTRUCTION	EXECUTION TIME (us)			
	High Performance Memory		High Performance Fault Control Memory	
	Non-DMS	DMS	Non-DMS	DMS
Extended Instruction Group				
(Index Register Instructions)				
CAX.CBX,CAY,CBY,CXA,CXB,CYA,CYB	1.295	1.295	1.295	1.330
XAX,XBX,XAY,XBY	1.925	1.925	1.925	1.960
DSX,DSY (no skip)	1.750	1.750	1.750	1.820
(skip)	1.995	2.065	2.065	2.205
ISX,ISY (no skip)	1.750	1.750	1.750	1.820
(skip)	1.995	2.065	2.065	2.205
LDX,LDY	2.660	2.730	2.730	2.835
Each Indirect Level	0.805	0.875	0.875	0.980
STX,STY	2.940	2.940	2.940	3.010
Each Indirect Level	0.805	0.875	0.875	0.980
LAX,LBX,LAY,LBY	3.360	3.430	3.430	3.535
Each Indirect Level	0.805	0.875	0.875	0.980
SAX,SAY,SBX,SBY	3.465	3.465	3.465	3.500
Each Indirect Level	0.805	0.875	0.875	0.980
ADX,ADY	2.730	2.800	2.800	2.870
Each Indirect Level	0.805	0.875	0.875	0.980
JLY	2.660	2.660	2.660	2.695
Each Indirect Level	0.805	0.875	0.875	0.980
JPY	2.275	2.275	2.275	2.345
(Byte Manipulation Instructions)				
LBT (from high byte)	3.640	3.640	3.640	3.675
(from low byte)	3.360	3.360	3.360	3.395
SBT (to high byte)	4.340	4.410	4.410	4.515
(to low byte)	3.885	4.025	4.025	4.165
MBT	3.745	3.815	3.815	3.955
	+4.042/byte	+4.078/byte	+4.078/byte	+4.130/byte
Each Indirect Level	0.805	0.875	0.875	0.980
CBT	3.745	3.815	3.815	3.955
	+4.480/byte	+4.480/byte	+4.480/byte	+4.480/byte
Each Indirect Level	0.805	0.875	0.875	0.980
SFB (if compare exit)	1.925	1.995	1.995	2.065
	+2.730/byte	+2.730/byte	+2.730/byte	+2.730/byte
(if terminal exit)	2.450	2.450	2.450	2.485
	+2.730/byte	+2.730/byte	+2.730/byte	+2.730/byte
(Word Manipulation Instructions)				
CMW	3.75	3.815	3.815	3.955
	+2.380/word	+2.520/word	+2.520/word	+2.660/word
Each Indirect Level	0.805	0.875	0.875	0.980
MVW	3.745	3.815	3.815	3.955
	+1.680/word	+1.680/word	+1.680/word	+1.680/word
Each Indirect Level	0.805	0.875	0.875	0.980
(Bit Manipulation Instructions)				
CBS,SBS	4.480	4.550	4.550	4.690
Each Indirect Level	0.805	0.875	0.875	0.980
TBS (no skip)	4.935	4.935	4.935	5.040
(skip)	5.005	5.075	5.075	5.250
Each Indirect Level	0.805	0.875	0.875	0.984

Table 3-5C. Typical Base Set Instruction Execution Times[1] (Continued)

INSTRUCTION	EXECUTION TIME (us)			
	High Performance Memory		High Performance Fault Control Memory	
	Non-DMS	DMS	Non-DMS	DMS
Floating Point Group				
FAD	4.760 to 7.630	4.830 to 7.700	4.830 to 7.700	4.970 to 7.840
FSB	4.760 to 7.630	4.830 to 7.700	4.830 to 7.700	4.970 to 7.840
FMP	6.020 to 6.370	6.090 to 6.440	6.090 to 6.440	6.230 to 6.580
FDV	6.020 to 9.240	6.090 to 9.310	6.090 to 9.310	6.230 to 9.450
FIX	3.640 to 5.250	3.640 to 5.250	3.640 to 5.250	3.675 to 5.285
FLT	3.395 to 4.550	3.395 to 4.550	3.395 to 4.550	3.430 to 4.585
FIXD	3.465 to 6.230	3.465 to 6.230	3.465 to 6.230	3.500 to 6.265
FLTD	3.290 to 5.705	3.290 to 5.705	3.290 to 5.705	3.325 to 5.740
.XADD	10.220 to 13.720	10.360 to 13.860	10.430 to 13.930	10.710 to 14.210
.XSUB	10.220 to 13.720	10.360 to 13.860	10.430 to 13.930	10.710 to 14.210
.XMPY	12.110 to 12.915	12.250 to 13.055	12.320 to 13.125	12.600 to 13.405
.XDIV	12.110 to 17.395	12.250 to 17.535	12.320 to 17.605	12.600 to 17.885
.XFXS	5.600 to 6.755	5.670 to 6.825	5.670 to 6.825	5.810 to 6.965
.XFTS	5.775 to 6.510	5.775 to 6.510	5.854 to 6.580	5.915 to 6.650
.XFXD	6.230 to 8.645	6.300 to 8.715	6.300 to 8.715	6.440 to 8.855
.XFTD	6.475 to 8.015	6.475 to 8.015	6.545 to 8.085	6.580 to 8.120
.TADD	9.730 to 14.840	10.360 to 15.120	10.430 to 15.190	11.095 to 15.855
.TSUB	9.730 to 14.840	10.360 to 15.120	10.430 to 15.190	11.095 to 15.855
.TMPY	12.880 to 13.685	13.510 to 14.315	13.580 to 14.395	13.790 to 15.050
.TDIV	12.880 to 19.670	13.510 to 20.300	13.580 to 20.370	14.245 to 21.035
.TFXS	6.125 to 7.280	6.265 to 7.420	6.265 to 7.420	6.475 to 7.630
.TFTS	6.195 to 6.930	6.195 to 6.930	6.265 to 7.000	6.335 to 7.070
.TFXD	6.755 to 9.170	6.895 to 9.310	6.895 to 9.310	7.105 to 9.520
.TFTD	6.895 to 8.435	6.895 to 8.435	6.965 to 8.505	7.000 to 8.540

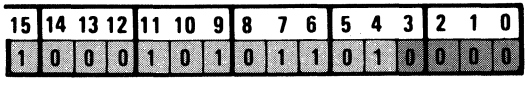
[1]Semiconductor memory refresh may increase program execution times by up to 3%.

3-34. SCIENTIFIC INSTRUCTION SET (F-SERIES ONLY)

The Scientific Instruction Set (SIS) utilizes the Floating Point Processor (FPP) to perform nine trigonometric and logarithmic functions. After initiating an operation in the FPP, the SIS firmware waits for FPP completion. If completion is not signalled by the FPP within 25 microseconds, an FPP malfunction exists. The firmware indicates this condition by creating a memory protect violation (memory protect installed), setting the overflow flag, and setting the A-register to all ones (an unnormalized number).

The following paragraphs provide machine language coding and definitions for the SIS instructions. Error conditions and codes are given in Table 3-6. Note that except for zero, all floating point operands must be normalized (i.e., sign of mantissa differs from most significant bit of mantissa).

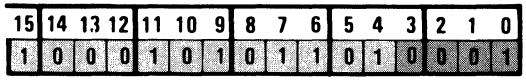
TAN* TANGENT



Calculates the tangent of the single precision floating point quantity (in radians) contained in the A- and B-registers. The result is returned to the A- and B-registers. A normal return will skip the next instruction. An error return will execute the next instruction, set the overflow bit, and return an ASCII error code in the A- and B-registers.

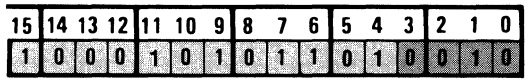
*For HP Assembly Language usage, refer to paragraph 3-43.

SQRT* SQUARE ROOT



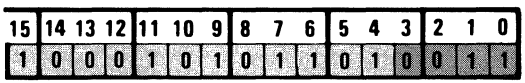
Calculates the square root of the single precision floating point quantity contained in the A- and B-registers. The result is returned to the A- and B-registers. A normal return will skip the next instruction. An error return will execute the next instruction, set the overflow bit, and return an ASCII error code in the A- and B-registers.

ALOG* NATURAL LOGARITHM



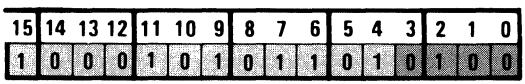
Calculates the natural logarithm of the single precision floating point quantity contained in the A- and B-registers. The result is returned to the A- and B-registers. A normal return will skip the next instruction. An error return will execute the next instruction, set the overflow bit, and return an ASCII error code in the A- and B-registers.

ATAN* ARCTANGENT



Calculates the arctangent of the single precision floating point quantity contained in the A- and B-registers. The result (in radians) is returned to the A- and B-registers.

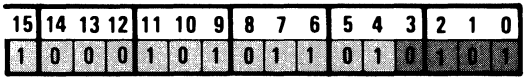
COS* COSINE



Calculates the cosine of the single precision floating point quantity (in radians) contained in the A- and B-registers. The result is returned to the A- and B-registers. A normal return will skip the next instruction. An error return will execute the next instruction, set the overflow bit, and return an ASCII error code in the A- and B-registers.

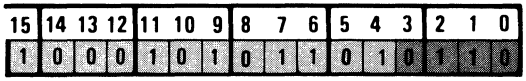
*For HP Assembly Language usage, refer to paragraph 3-43.

SIN* SINE



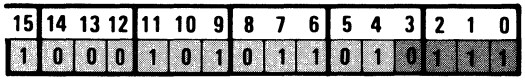
Calculates the sine of the single precision floating point quantity (in radians) contained in the A- and B-registers. The result is returned to the A- and B-registers. A normal return will skip the next instruction. An error return will execute the next instruction, set the overflow bit, and return an ASCII error code in the A- and B-registers.

EXP* E TO THE POWER X



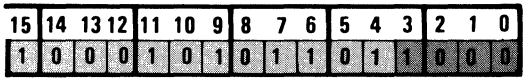
Calculates e to the power x of the single precision floating point quantity contained in the A- and B-registers. The result is returned to the A- and B-registers. A normal return will skip the next instruction. An error condition will execute the next instruction, set the overflow bit, and return an ASCII error code in the A- and B-registers.

ALOGT* COMMON LOGARITHM



Calculates the common logarithm of the single precision floating point quantity contained in the A- and B-registers. The result is returned to the A- and B-registers. A normal return will skip the next instruction. An error condition will execute the next instruction, set the overflow bit, and return an ASCII error code in the A- and B-registers.

TANH* HYPERBOLIC TANGENT



Calculates the hyperbolic tangent of the single precision floating point quantity contained in the A- and B-registers. The result is returned to the A- and B-registers.

*For HP Assembly Language usage, refer to paragraph 3-43.

DPOLY* POLYNOMIAL EVALUATION

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	1	0	1	1	0	1	1	0	0	1
F	S	X	X	X	X	X	X	X	X	X	X	X	X	X	T
D/I															
D/I															
D/I															
D/I															
D/I															

Memory Address

Evaluates a polynomial or quotient of polynomials using 64-bit floating-point. This is a seven-word instruction where:

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of result Y (64-bit floating).
Word 4 = Address of argument X (64-bit floating).
Word 5 = Address of coefficient P_M (64-bit floating).
Word 6 = Address of numerator order M (integer).
Word 7 = Address of denominator order N (integer).

Define $P(Z) = P_M Z^M + P_{M-1} Z^{M-1} + \ldots + P_1 Z + P_0$
$Q(Z) = Z^N + Q_{N-1} + \ldots + Q_1 Z + Q_0$

The computation performed depends on the values of bits F, S, T of the sub-opcode:

F=0: $Y = P(X)/Q(X)$
F=1, S=0, T=1: $Y = P(X^2)/Q(X^2)$
F=1, S=1, T=0: $Y = X*P(X^2)/Q(X^2)$
F=1, S=1, T=1: $Y = P(X^2)/(Q(X^2)-P(X^2))$ (N>0)
F=1, S=1, T=0: $Y = X*P(X^2)/(Q(X^2)-P(X^2))$ (N>0)

Horner's Rule is used to evaluate the polynomial(s). The coefficients must be stored sequentially in memory, starting with P_M, in the order:

$$P_M, P_{M-1}, \ldots, P_1, P_0, Q_{N-1}, \ldots, Q_1, Q_0$$

where $Q_N = 1.0$ is implied but not stored. If N=0, no coefficients are provided for Q and only P is evaluated. The case N=0 and S=1 is not allowed.

Any underflow or overflow which occurs invalidates the final result. M must be at least one. The A,B,X,Y and E registers are undefined after this instruction. The O register is set if any underflow or overflow occurs, else cleared.

This instruction is interruptible. Since it restarts after an interrupt, it is not recomended for very large values of (M+N).

Timing: Approximately 15+9M if N=0
20+9(M+N) if N>0

*For HP Assembly Language usage, refer to paragraph 3-43.

/ATLG* (1−X)/(1+X)

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

Memory Address

Performs the computation X = (1−X)/(1+X).

Word 1 = Instruction code.
Word 2 = Direct address of X (64-bit floating).

The A,B,X,Y,E and O registers are undefined after this instruction.

.FPWR* EXPONENTIATION

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

Memory Address

Raises a 32-bit floating-point number to an integer power. This is a two-word instruction, where:

Word 1 = Instruction code.
Word 2 = Address of base X (32-bit floating).

The power I is supplied in the A-register. It is unsigned and must be in the range [2,32768]. The left-to-right binary method is used to compute X^I, e.g. if $I=83_{10} = 123_8 = 1010011_2$ then

$X^2 = X*X$	10
$X^4 = X^2*X^2$	100
$X^5 = X^4*X$	101
$X^{10} = X^5*X^5$	1010
$X^{20} = X^{10}*X^{10}$	10100
$X^{40} = X^{20}*X^{20}$	101000
$X^{41} = X^{40}*X$	101001
$X^{82} = X^{41}*X^{41}$	1010010
$X^{83} = X^{82}*X$	1010011

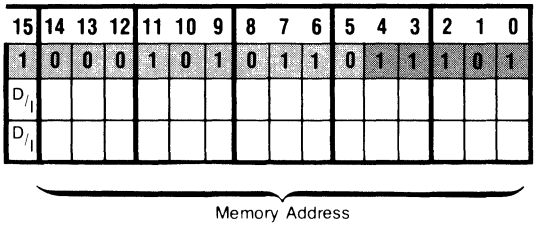
The X,Y and E registers are undefined. The O register is set if underflow or overflow occur else cleared. The A- and B-registers contain the result.

Timing: Approximately 20 + 2.7M + 3.5N microseconds
where M = (# bits in I)
N = (# bits set in I)

Maximum time non-interruptible: Approximately 32 μs.

*For HP Assembly Language usage, refer to paragraph 3-43.

.TPWR* EXPONENTIATION



Raises a 64-bit floating-point number to an integer power. This is a three-word instruction, where:

Word 1 = Instruction code.
Word 2 = Address of result (64-bit floating).
Word 3 = Address of base X (64-bit floating).

The power I is supplied in the A-register. It is unsigned and must be in the range [2,32768]. The left-to-right binary method is used.

The A,B,X,Y and E registers are undefined. The O register is set if underflow or overflow occur else cleared.

Timing: Appriximately 22 + 4.5M + 6N microseconds

where M = (# bits in I)
N = (# bits set in I)

Maximum time non-interruptible: Approximately 42 μs.

Table 3-6. SIS Instruction Error Codes

INSTRUCTION	ERROR CODE
TAN	09OR if |x| > 32768*π/4
SQRT	03UN if x < 0
ALOG	02UN if x ≤ 0
ATAN	None
COS	05OR if |x| > 32768*π/4
SIN	05OR if |x| > 32768*π/4
EXP	07OF if x > 88.029678
ALOGT	02UN if x ≤ 0
TANH	None

Where:
OF = Integer or floating point overflow.
OR = Out of range.
UN = Undefined.

*For HP Assembly Language usage, refer to paragraph 3-43.

3-35. EXECUTION TIMES AND INTERRUPTS

Table 3-7 lists the typical execution times required for the SIS instructions. Also listed is the maximum period of non-interruptible instruction execution. If an instruction is interrupted, its execution restarts from the beginning.

Table 3-7. Typical Scientific Instruction Set Execution Times

INSTRUCTION	EXECUTION TIME (us)	MAXIMUM NON-INTERRUPTIBLE (us)
TAN	48.4	8
SQRT	30.9	11
ALOG	43.3	8
ATAN	42.4	12
COS	47.9	9
SIN	47.6	8
EXP	44.7	8
ALOGT	49.4	8
TANH	57.2	9
DPOLY	See Text	
/ATLG	25	28
.FPWR	See Text	32
.TPWR	See Text	42

3-36. FAST FORTRAN INSTRUCTIONS (Standard with F-Series, Optional with M- and E-Series)

The Fast FORTRAN Processor instructions perform several frequently-used FORTRAN operations including parameter passing, array address calculations, and floating point conversion, packing, rounding and normalization operations.

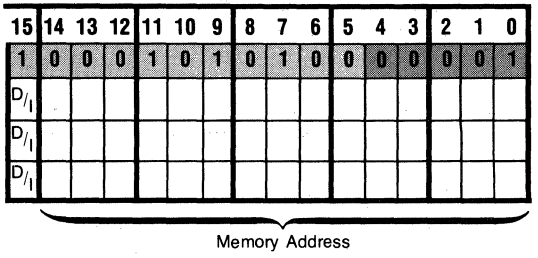
The Fast FORTRAN Processor instructions as well as the extended precision add, subtract, multiply and divide instructions are standard with the HP 1000 F-Series Computers. The extended precision add, subtract, multiply and divide instructions are included as part of the optional Fast FORTRAN Processor instructions on the M/E-Series Computers and possess different instruction codes as those in the F-Series.

For multiple-word instructions, indirect addressing to any number of levels is permitted for the word(s) indicated as a memory address. A logic 0 in bit position 15 specifies direct addressing; a logic 1 specifies indirect addressing.

The following paragraphs provide machine language coding and definitions for the fast FORTRAN instructions. Data formats are shown in Figure 3-1.

For a more detailed description of the fast FORTRAN instructions see the DOS/RTE Relocatable Library Reference Manual, HP Part No. 24998-90001.

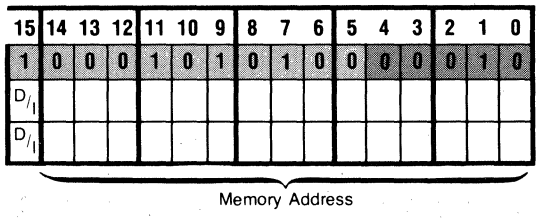
DBLE* SINGLE FLOATING POINT TO EXTENDED FLOATING POINT



Converts the single precision floating point quantity in specified memory locations to an extended precision floating point quantity. The result is returned to other specified memory locations. This is a four-word instruction where

Word 1 = Instruction code.
Word 2 = Return address.
Word 3 = Address of result.
Word 4 = Address of operand.

SNGL* EXTENDED FLOATING POINT TO SINGLE FLOATING POINT

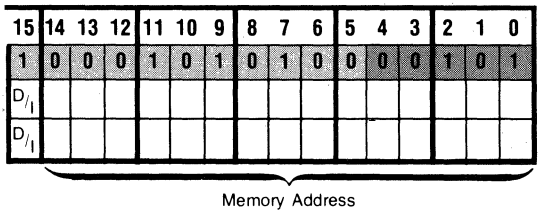


Converts the extended precision floating point quantity in the specified memory locations to a single precision floating point quantity. The result is placed in the A- and B-registers. This is a three word instruction where

Word 1 = Instruction code.
Word 2 = Return address.
Word 3 = Address of operand.

*For HP Assembly Language usage, refer to paragraph 3-43.

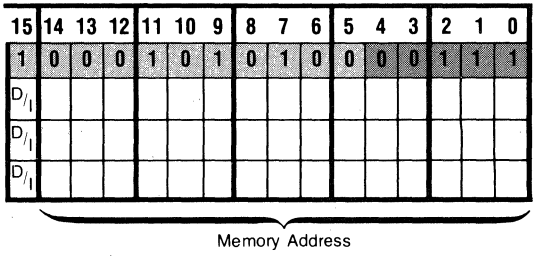
.DFER* TRANSFER EXTENDED FLOATING POINT



Transfers an extended precision floating point quantity (three consecutive words) from one memory location to another. The source address +3 is returned to the A-register; the destination address +3 is returned to the B-register. This is a three word instruction where

Word 1 = Instruction code.
Word 2 = Destination address.
Word 3 = Source address.

.BLE (F-Series Only)* SINGLE FLOATING POINT TO DOUBLE FLOATING POINT

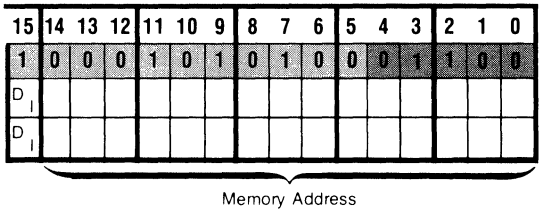


Converts the single precision floating point quantity in specified memory locations to a double precision floating point quantity. The result is returned to other specified memory locations. This is a four-word instruction where

Word 1 = Instruction code.
Word 2 = Return address.
Word 3 = Address of result.
Word 4 = Address of operand.

*For HP Assembly Language usage, refer to paragraph 3-43.

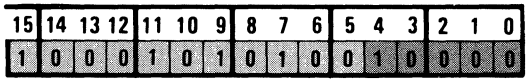
.NGL (F-Series Only) DOUBLE FLOATING POINT TO SINGLE FLOATING POINT



Converts the double precision floating point quantity in the specified memory locations to a single precision floating point quantity. The result is placed in the A- and B-registers. This is a three word instruction where

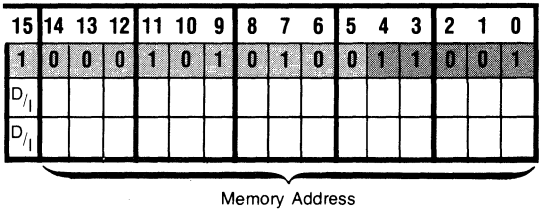
Word 1 = Instruction code.
Word 2 = Return address.
Word 3 = Address of operand.

.XFER* TRANSFER EXTENDED FLOATING POINT



Transfers an extended precision floating point quantity (three consecutive words) from one memory location to another. The A-register must contain the source address and the B-register must contain the destination address. The source address +3 is returned to the A-register; the destination address +3 is returned to the B-register.

.CFER* (E/F-Series Only) TRANSFER COMPLEX OR DOUBLE FLOATING POINT

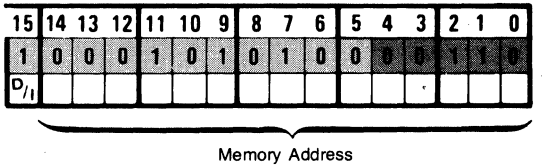


Transfers a double precision floating point quantity (four consecutive words) from one memory location to another. The source address +4 is returned to the A-register; the destination address +4 is returned to the B-register. This is a three word instruction where

Word 1 = Instruction code.
Word 2 = Destination address.
Word 3 = Source address.

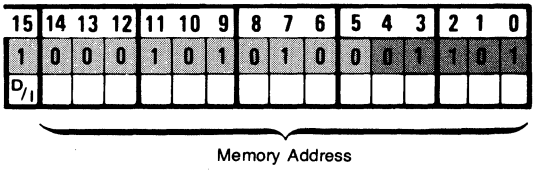
*For HP Assembly Language usage, refer to paragraph 3-43.

.XPAK* NORMALIZE AND PACK EXTENDED FLOATING POINT



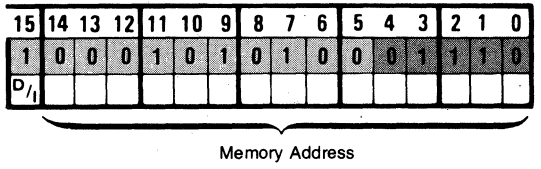
Normalizes, rounds, and packs with exponent the mantissa of an extended precision floating point quantity located in the specified memory locations. The value of the exponent must be in the A-register. The result is returned to the same specified memory locations.

.XCOM* COMPLEMENT EXTENDED FLOATING POINT ADD



Complements in place an unpacked extended precision floating point quantity located in the specified memory locations. The result is returned to the same specified memory locations. If the exponent is adjusted, the A-register equals 1; if not, the A-register equals 0.

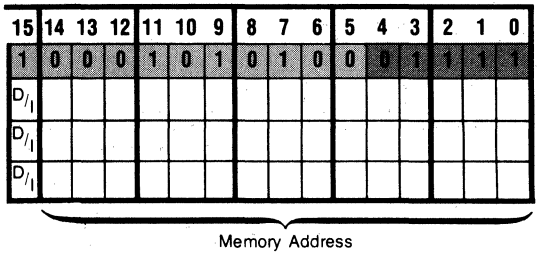
..DCM* COMPLEMENT AND NORMALIZE EXTENDED FLOATING POINT



Complements and normalizes in place an unpacked extended precision floating point quantity located in the specified memory locations. The result is returned to the same specified memory locations.

*For HP Assembly Language usage, refer to paragraph 3-43.

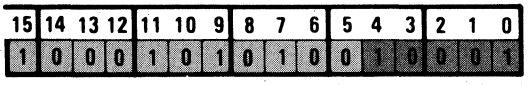
DDINT* EXTENDED FLOATING POINT TRUNCATION



Truncates the extended precision floating point quantity in specified memory locations to an integer value. The extended precision floating point result is returned to other specified memory locations. This is a four-word instruction where

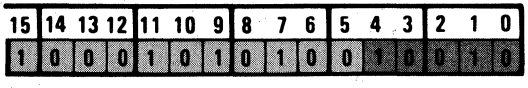
Word 1 = Instruction code.
Word 2 = Return address.
Word 3 = Address of result.
Word 4 = Address of operand.

.GOTO* TRANSFER CONTROL



Transfers control to the location indicated by a FORTRAN computed GO TO statement: GO TO (K1,K2, . . . ,Kn),J. If J<1, then result address is K1; if J>n, address is Kn; otherwise, address is KJ. The succeeding memory locations must define the return address, the value of J, and the parameters.

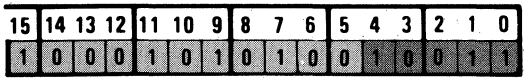
..MAP* COMPUTE ARRAY ELEMENT ADDRESS



Computes the address of a specified element of a 2 or 3 dimensional array; returns the address to the A-register. The A-register must equal 0 for a 2 dimensional array or 1 for a 3 dimensional array. The B-register must contain the number of words per variable. Succeeding memory locations must define base address, element subscripts, and length of first (or first and second) dimension.

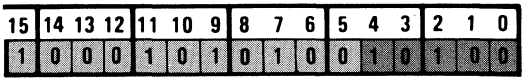
*For HP Assembly Language usage, refer to paragraph 3-43.

.ENTR* TRANSFER PARAMETER ADDRESSES



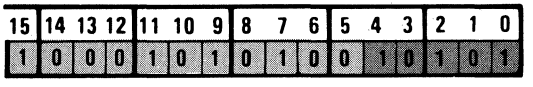
Transfers the true addresses of parameters from a calling sequence into a subroutine; adjusts return address to the true return point. A true address is determined by eliminating all indirect references.

.ENTP* TRANSFER PARAMETER ADDRESSES



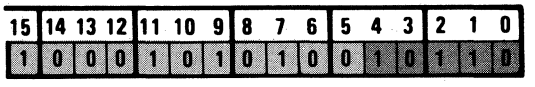
Transfers the true addresses of parameters from a calling sequence into a subroutine; adjusts return address to the true return point. A true address is determined by eliminating all indirect references, used for privileged or re-entrant subroutines.

.PWR2* X TIMES 2 TO THE POWER N



Calculates for floating point x and integer n: $y = x*2^n$. The floating point quantity must be in the A- and B-registers; the succeeding memory location must define integer n. The first word of the two word floating point result is returned to the A-register; the second word, to the B-register.

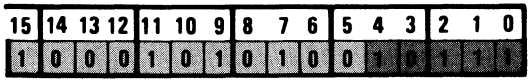
.FLUN* UNPACK FLOATING POINT QUANTITY



Unpacks a floating point quantity. The lower part of the floating point quantity must be in the B-register. The exponent is returned to the A-register; the lower part of the mantissa is returned to the B-register, left justified.

*For HP Assembly Language usage, refer to paragraph 3-43.

$SETP SET A TABLE

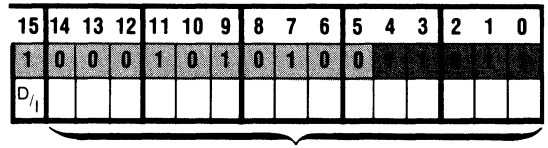


Sets a table of increasing numbers in consecutive memory locations. The A-register must contain the initial number and the B-register must contain the initial memory address (direct only); the contents of the succeeding memory location must define the number of memory locations (count ≥ 0). Entries in the table are established by incrementing the initial address and number by one (1) for each successive entry until the last number, initial number +COUNT−1, is reached. On return, the B-register equals the initial address +COUNT.

NOTE

If the initial address +COUNT −1 results in an address which is beyond the end of logical memory, addresses within the base page are destroyed.

..TCM* COMPLEMENT AND NORMALIZE DOUBLE FLOATING POINT (F-Series Only)

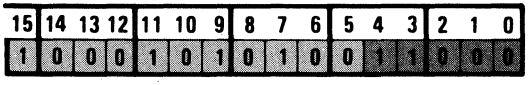


Complements and normalizes in place a packed double precision floating point quantity located in the specified memory locations. The result is returned to the same specified memory locations.

3-37. EXTENDED PRECISION OPERATIONS.

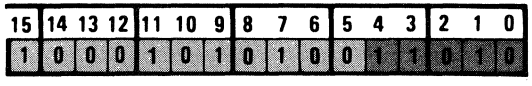
Overflow for extended precision operations occurs if the result lies outside the range of representable extended precision floating point numbers $[-2^{127},(1-2^{-39})\ 2^{-127}]$. In such a case, the overflow flag is set and $(1-2^{-39})\ 2^{127}$ is returned as the result. Underflow occurs if the result lies inside the range $[-2^{-129}(1+2^{-38}),2^{-129}]$. In such a case, the overflow flag is set and 0 is returned as the result.

.PACK* NORMALIZE FLOATING POINT QUANTITY



Converts the signed mantissa of a floating point quantity into a normalized format. The floating point mantissa must be in the A- and B-registers. The succeeding memory location must contain the exponent. The first word of the two word floating point result is returned to the A-register; the second word, to the B-register.

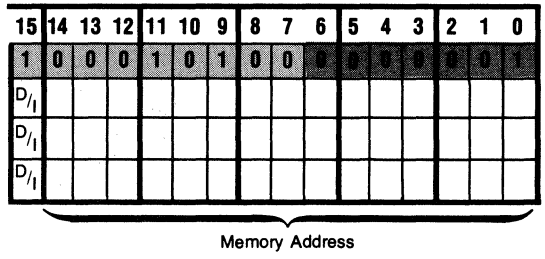
..FCM* COMPLEMENT AND NORMALIZE SINGLE FLOATING POINT (F-Series Only)



Complements and normalizes in place a packed single precision floating point quantity located in the A- and B-registers. The result is returned to the A- and B-registers.

*For HP Assembly Language usage, refer to paragraph 3-43.

.XADD* EXTENDED FLOATING POINT ADD

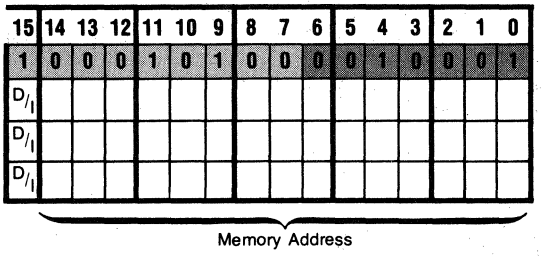


Adds two extended precision floating point quantities (augend plus addend). This is a four-word instruction where

Word 1 = Instruction code.
Word 2 = Address of result.
Word 3 = Address of augend.
Word 4 = Address of addend.

*For HP Assembly Language usage, refer to paragraph 3-43.

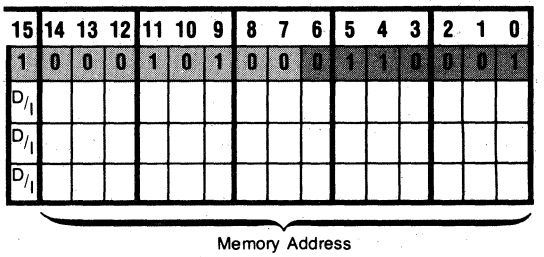
.XSUB* EXTENDED FLOATING POINT SUBTRACT



Subtracts one extended precision floating point quantity from another (minuend minus subtrahend). This is a four-word instruction where

Word 1 = Instruction code.
Word 2 = Address of result.
Word 3 = Address of minuend.
Word 4 = Address of subtrahend.

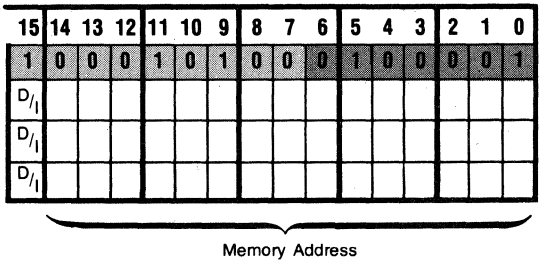
.XDIV* EXTENDED FLOATING POINT DIVIDE



Divides one floating point quantity by another (dividend by divisor). This is a four-word instruction where

Word 1 = Instruction code.
Word 2 = Address of result.
Word 3 = Address of dividend.
Word 4 = Address of divisor.

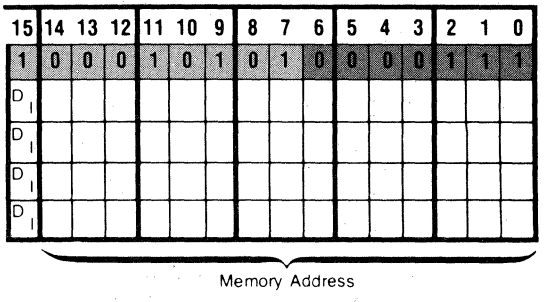
.XMPY* EXTENDED FLOATING POINT MULTIPLY



Multiplies one extended precision floating point quantity by another (multiplicand by multiplier). This is a four-word instruction where

Word 1 = Instruction code.
Word 2 = Address of result.
Word 3 = Address of multiplicand.
Word 4 = Address of multiplier.

XADD* EXTENDED FLOATING POINT ADD



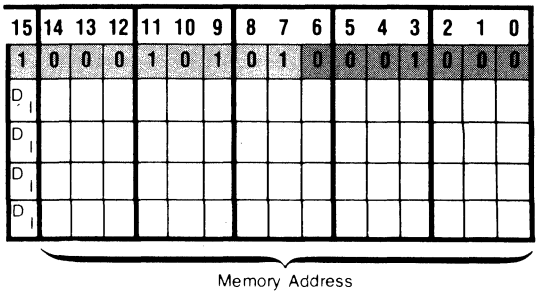
Adds two extended precision floating point quantities (augend plus addend). This is a five-word instruction where

Word 1 = Instruction code.
Word 2 = Return address.
Word 3 = Address of result.
Word 4 = Address of augend.
Word 5 = Address of addend.

*For HP Assembly Language usage, refer to paragraph 3-43.

*For HP Assembly Language usage, refer to paragraph 3-43.

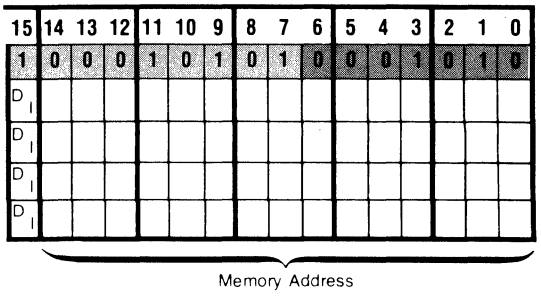
XSUB* EXTENDED FLOATING POINT SUBTRACT



Subtracts one extended precision floating point quantity from another (minuend minus subtrahend). This is a five-word instruction where

Word 1 = Instruction code.
Word 2 = Return address.
Word 3 = Address of result.
Word 4 = Address of minuend.
Word 5 = Address of subtrahend.

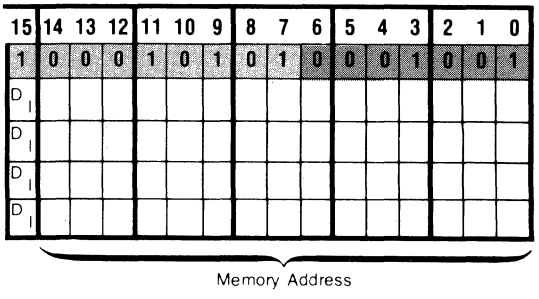
XDIV* EXTENDED FLOATING POINT DIVIDE



Divides one extended precision floating point quantity by another (dividend by divisor). This is a five-word instruction where

Word 1 = Instruction code.
Word 2 = Return address.
Word 3 = Address of result.
Word 4 = Address of dividend.
Word 5 = Address of divisor.

XMPY* EXTENDED FLOATING POINT MULTIPLE



Multiplies one extended precision floating point quantity by another (multiplicand by multiplier). This is a five-word instruction where

Word 1 = Instruction code.
Word 2 = Return address.
Word 3 = Address of result.
Word 4 = Address of multiplicand.
Word 5 = Address of multiplier.

3-38. EXECUTION TIMES AND INTERRUPTS

Table 3-8 lists the typical execution times required for the Fast FORTRAN Processor instructions. Also listed is the maximum period of non-interruptible instruction execution time where applicable. If an instruction is interrupted, it restarts execution from the beginning.

*For HP Assembly Language usage, refer to paragraph 3-43.

*For HP Assembly Language usage, refer to paragraph 3-43.

Table 3-8A. Typical Fast FORTRAN Instruction Execution Times (M-Series)

INSTRUCTION	EXECUTION* TIME (us)	MAXIMUM NON-INTERRUPTIBLE (us)
.XADD	42.25 to 130.98	75.08
.XSUB	42.25 to 130.98	75.08
.XMPY	80.28 to 105.3	75.08
.XDIV	107.9 to 163.15	75.08
XADD	42.25 to 130.98	75.08
XSUB	42.25 to 130.98	75.08
XMPY	80.28 to 105.3	75.08
XDIV	107.9 to 163.15	75.08
DBLE	15.28	
SNGL	20.15 to 65.0	
.DFER	9.75 to 13.33	
.XFER	9.75 to 13.33	
.SETP	7.8 + (1.95 × Count)	Interruptable For Count >30
.PACK	13.33 to 52.33	
.XPACK	19.5 to 99.13	75.08
.XCOM	12.68 to 15.28	
..DCM	25.35 to 104.33	75.08
DDINT	31.53 to 183.3	75.08
.GOTO	13.0	
..MAP	26.98 to 43.88	
.ENTR	16.25 + (3.9 × NP)	
.ENTP	15.98 + (3.9 × NP)	
.PWR2	12.35	
.FLUN	4.23	

*Using standard performance memory.

Table 3-8B. Typical Fast FORTRAN Instruction Execution Times (E-Series)

INSTRUCTION	EXECUTION* TIME (us)	MAXIMUM NON-INTERRUPTIBLE (us;
.XADD	37.5 to 50.2	25.7
.XSUB	37.5 to 50.2	25.7
.XMPY	56.0 to 64.8	36.4
.XDIV	80.0 to 92.4	37.8
XADD	38.0 to 50.7	
XSUB	37.5 to 50.2	
XMPY	56.7 to 65.5	
XDIV	80.7 to 93.1	
DBLE	13.02	
SNGL	18.2	
.DFER	12.8	
.XFER	8.96 to 12.7	
.CFER	14.9	
$SETP	6.4 + (1.2 × Count)	Interruptable For Count >30
.PACK	19.2 to 27.2	
.XPACK	18.9 to 29.5	11.6
.XCOM	11.7 to 12.1	
..DCM	21.1 to 33.4	12.2
DDINT	23.9 to 58.6	30.6
.GOTO	10.6	
..MAP	17.7 to 27.2	
.ENTR	13.9 + (3.7 × NP)	
.ENTP	13.6 + (3.7 × NP)	
.PWR2	8.4	
.FLUN	3.1	

*Using standard performance memory.

Table 3-8C. Typical Fast FORTRAN Instruction Execution Times (F-Series)

INSTRUCTION	EXECUTION TIME (us)	MAXIMUM NON-INTERRUPTIBLE (us)
DBLE	9.3	
SNGL	14.3	
.DFER	9.0	
.BLE	9.5	
.NGL	16.0	
.XFER	8.96 to 12.7	
.XPAK	18.9 to 29.5	11.6
.XCOM	11.7 to 12.1	
..DCM	22.1 to 33.4	12.2
DDINT	23.9 to 58.6	30.6
.GOTO	10.6	
..MAP	17.7 to 27.2	
.ENTR	10.1 + 2.5*NP	
.ENTP	13.6 + 3.7*NP	
.PWR2	8.4	
.FLUN	3.1	
$SETP	6.4 + 1.2*COUNT	42.4
.PACK	19.2 to 27.2	
.CFER	10.3	
..FCM	5.1	
..TCM	9.8	

NP = number of parameters

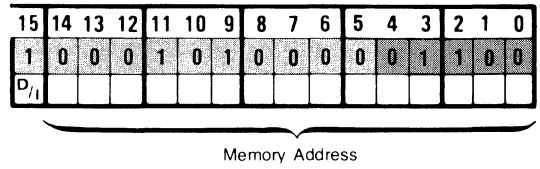
3-39. DOUBLE INTEGER INSTRUCTIONS (F-SERIES ONLY)

The double integer instructions allow arithmetic and test operations on 32-bit integer quantities. The data format for double integer values is shown in Figure 3-1. Double integer values contained in the (A,B) registers have the most significant bits in the A-register. Values stored in memory require two locations. The operand address in a double integer instruction points to the first memory location, which contains the most significant bits.

Instructions which do not return information in the extend or overflow bits will not alter the state of these flags. Operations which may return an overflow condition will clear overflow at entry.

The instructions .DMP, .DDI, and .DDIR utilize the floating point processor. If FPP completion is not signalled within 25 microseconds, an FPP malfunction exists. The firmware indicates this condition by creating a memory protect violation (memory protect installed) and setting the overflow flag.

.DAD* DOUBLE INTEGER ADD

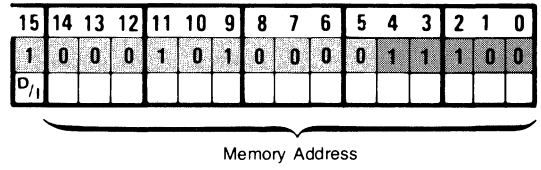


Performs the double integer operation:

(A,B) = (A,B) + <OPND>

The contents of <OPND> are unaltered. In the event of overflow, the overflow bit is set and the returned result contains the lower 32-bits of the actual sum, in unsigned form. The extend bit will be set if an unsigned carry out of the A-register occurs.

.DSB* DOUBLE INTEGER SUBTRACT

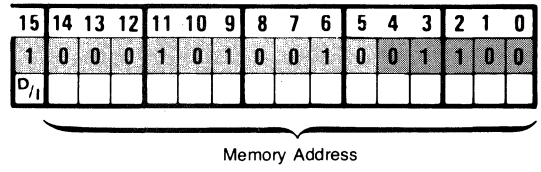


Performs the double integer operation:

(A,B) = (A,B) − <OPND>

The contents of <OPND> are unaltered. In the event of overflow, the overflow bit is set and the returned result contains the lower 32-bits of the actual difference, in unsigned form. The extend bit will be set if an unsigned borrow out of the A-register occurs.

.DSBR* DOUBLE INTEGER SUBTRACT REVERSE



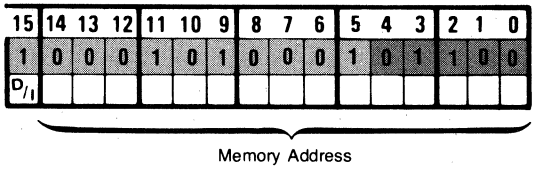
Performs the double integer operation:

(A,B) = <OPND> − (A,B)

The contents of <OPND> are unaltered. In the event of overflow, the overflow bit is set and the returned result contains the lower 32-bits of the actual difference, in unsigned form. The extend bit will be set if an unsigned out of operand.

*For HP Assembly Language usage, refer to paragraph 3-43.

.DMP * DOUBLE INTEGER MULTIPLY

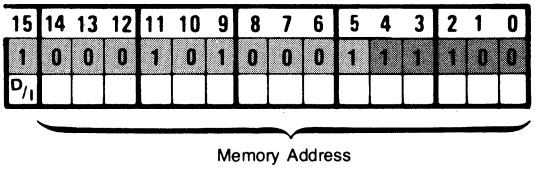


Performs the double integer operation:

(A,B) = (A,B) × <OPND>

The contents of <OPND> are unaltered. If overflow occurs, the result (077777, 177777) is returned and overflow is set.

.DDI * DOUBLE INTEGER DIVIDE

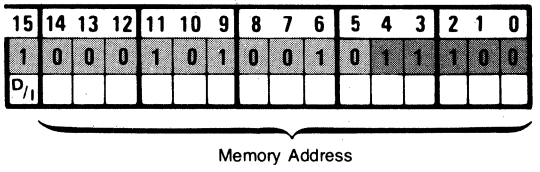


Performs the double integer operation:

(A,B) = (A,B) ÷ <OPND>

The contents of <OPND> are unaltered. If overflow or divide by zero occurs, the result (077777, 177777) is returned and overflow is set.

.DDIR * DOUBLE INTEGER DIVIDE REVERSE



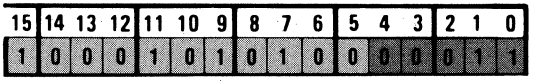
Performs the double integer operation:

(A,B) = <OPND> ÷ (A,B)

The contents of <OPND> are unaltered. If overflow or divide by zero occurs, the result (077777, 177777) is returned and overflow is set.

*For HP Assembly Language usage, refer to paragraph 3-43.

.DNG * DOUBLE INTEGER NEGATE

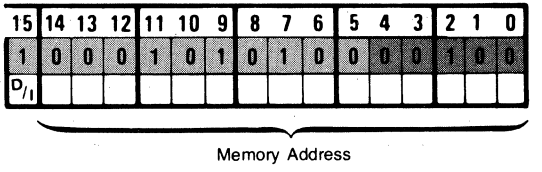


Performs the double integer operation:

(A,B) = − (A,B)

An input value of (100000, 000000) is left unchanged and overflow is set. An input value of zero will cause the extend bit to be set.

.DCO * DOUBLE INTEGER COMPARE

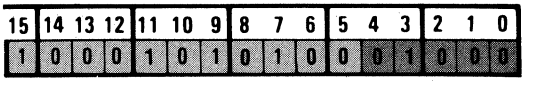


Compares the double integers (A,B) and <OPND>

If (A,B) = <OPND> Return to P+2
If (A,B) < <OPND> Return to P+3
If (A,B) > <OPND> Return to P+4

where P is the address of the .DCO instruction. The value of both double integers and the overflow bit are unaltered.

.DIN * DOUBLE INTEGER INCREMENT



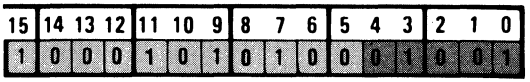
Performs the double integer operation:

(A,B) = (A,B) + 1

An input value of 077777, 177777) will return a result of 100000, 000000) and set overflow. An input value of (177777, 177777) will return a result of zero and cause the extend bit to be set.

*For HP Assembly Language usage, refer to paragraph 3-43.

.DDE * DOUBLE INTEGER DECREMENT

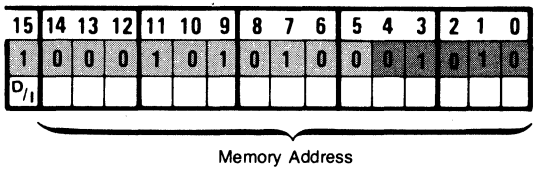


Performs the double integer operation:

(A,B) = (A,B) − 1

An input value of (100000, 000000) will return the result (077777, 177777) and set overflow. An input value of zero will return the result (177777, 177777) and cause the extend bit to be set.

.DIS * DOUBLE INTEGER INCREMENT AND SKIP IF ZERO

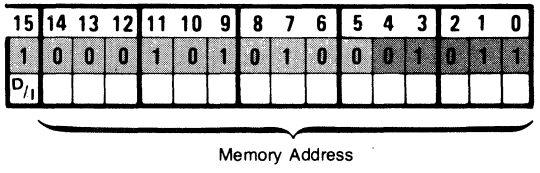


Performs the double integer operation:

<OPND> = <OPND> + 1

If the new value of <OPND> equals zero, the next instruction will be skipped. The value in <OPND> is treated as an unsigned number, and a carry out of the <OPND> is ignored.

.DDS * DOUBLE INTEGER DECREMENT AND SKIP IF ZERO



Performs the double integer operation:

<OPND> = <OPND> − 1

If the new value of <OPND> equals zero, the next instruction will be skipped. The value in <OPND> is treated as an unsigned number, and a borrow out of the <OPND> is <OPND> is ignored.

*For HP Assembly Language usage, refer to paragraph 3-43.

3-40. EXECUTION TIMES

Table 3-9 lists the typical execution times for the double integer instructions.

Table 3-9. Typical Double Integer Instructions Execution Times (Microseconds)

INSTRUCTION	EXECUTION TIME
.DAD	4.6
.DSB	5.4
.DMP	15.9
.DDI	19.9
.DSBR	6.2
.DDIR	19.3
.DNG	2.2
.DIN	1.7
.DDE	2.1
.DIS	4.7
.DDS	5.0
.DCO	4.8

3-41. VECTOR INSTRUCTION SET (OPTIONAL WITH F-SERIES ONLY)

The Vector Instruction Set (VIS) performs arithmetic operations on arrays of floating point numbers. VIS utilizes the Floating Point Processor (FPP) as a computing resource and provides nineteen operations in both single and double precision formats, for a total of 38 instructions. After initiating an operation in the FPP, the VIS firmware waits for FPP completion. If completion is not signalled within 25 microseconds, an FPP malfunction exists. The firmware indicates this condition by creating a memory protect violation (memory protect installed) and setting the overflow flag. Overflow will always be cleared on a normal VIS termination.

Vector instructions require six to ten memory locations to specify parameters of the following type:

Opcode – Specifies the microcode entry point. Bit 11, or P-bit, indicates the precision of the operation. (=0 for single precision, P=1 for double precision.)

Sub-opcode – Specifies the arithmetic function within a class of operations. Bit 15 of the sub-opcode must always be zero. P-bits indicate the precision as in the opcode. Don't-care bits are indicated by "X".

The remaining parameters are addresses which may be direct or indirect, as indicated by bit 15. These include:

Vector – Specifies the address of the first vector element to be processed. Vector elements require two (single precision) or four (double precision) memory locations. All vectors in a given instruction must be of the same precision. Note that for instructions that contain two vector operands, these operands may both specify the same vector. Similarly, the result vector may replace one of the operands.

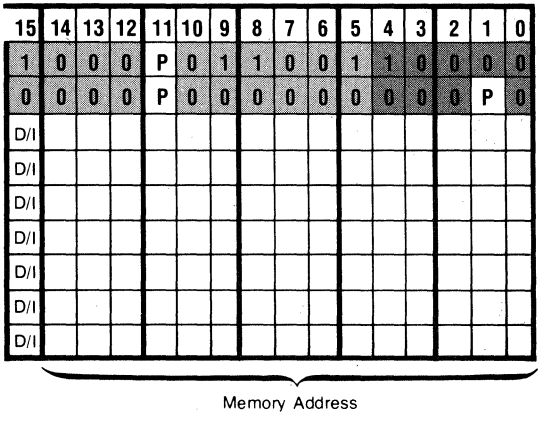
Scalar – Specifies the address of a single floating point quantity. Scalars are used for both operands or results. The precision of the scalar must match that of the associated vectors.

Integer – Specifies the address of an integer quantity in which a result is returned.

Increment – Specifies the address of an integer quantity associated with each vector. The increment indicates the spacing between vector elements to be processed. (An increment of 1 indicates that each element will be processed, an increment of 2 indicates every other element, etc.) An increment of zero will cause the first element of the vector to be used in all operations. Negative increments will step through the vector in reverse order, i.e., decreasing memory locations. Vector elements skipped over by the increment will not be modified.

#Elements – Specifies the address of an integer quantity indicating the number of vector elements to be processed. A value less than or equal to zero will result in a NOP operation.

VADD/DVADD * VECTOR ADD



*For HP Assembly Language usage, refer to paragraph 3-43.

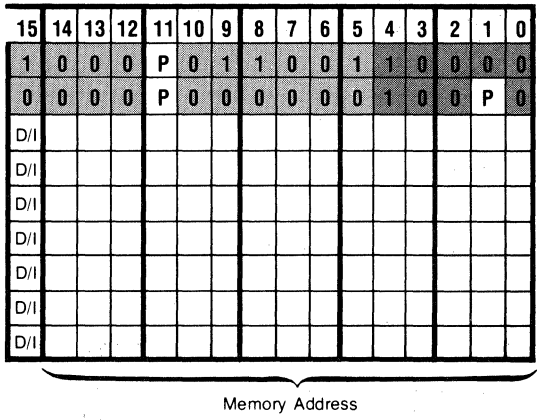
Performs the vector operation:

V3 = V1 + V2

This is a nine-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of vector V1.
Word 4 = Address of increment INCR1.
Word 5 = Address of vector V2.
Word 6 = Address of increment INCR2.
Word 7 = Address of vector V3.
Word 8 = Address of increment INCR3.
Word 9 = Address of # elements N.

VSUB/DVSUB * VECTOR SUBTRACT



Performs the vector operation:

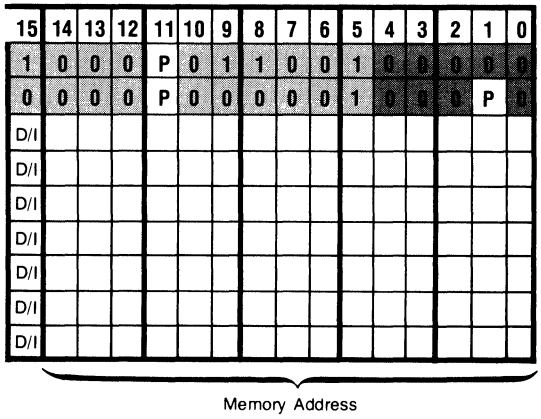
V3 = V1 – V2

This is a nine-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of vector V1.
Word 4 = Address of increment INCR1.
Word 5 = Address of vector V2.
Word 6 = Address of increment INCR2.
Word 7 = Address of vector V3.
Word 8 = Address of increment INCR3.
Word 9 = Address of # elements N.

*For HP Assembly Language usage, refer to paragraph 3-43.

VMPY/DVMPY * VECTOR MULTIPLY



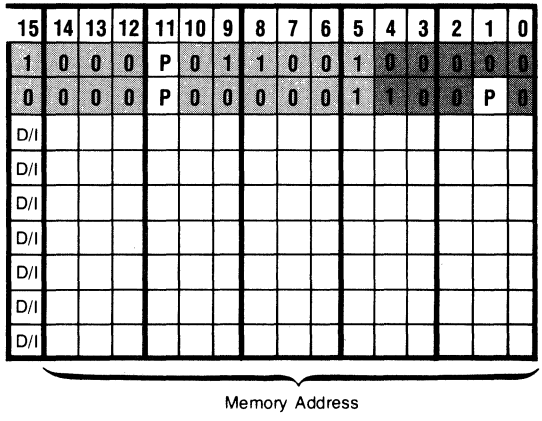
Performs the vector operation:

V3 =V1 * V2

This is a nine-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of vector V1.
Word 4 = Address of increment INCR1.
Word 5 = Address of vector V2.
Word 6 = Address of increment INCR2.
Word 7 = Address of vector V3.
Word 8 = Address of increment INCR3.
Word 9 = Address of # elements N.

VDIV/DVDIV * VECTOR DIVIDE



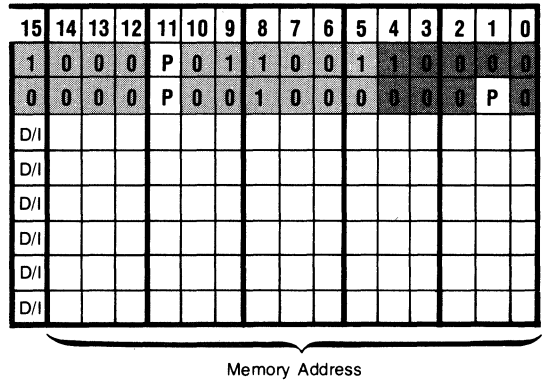
Performs the vector operation:

V3 = V1 / V2

This is a nine-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of vector V1.
Word 4 = Address of increment INCR1.
Word 5 = Address of vector V2.
Word 6 = Address of increment INCR2.
Word 7 = Address of vector V3.
Word 8 = Address of increment INCR3.
Word 9 = Address of # elements N.

VSAD/DVSAD * SCALAR-VECTOR ADD



Performs the vector operation:

V2 = S + V1

This is a eight-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of scalar S.
Word 4 = Address of vector V1.
Word 5 = Address of increment INCR1.
Word 6 = Address of vector V2.
Word 7 = Address of increment INCR2.
Word 8 = Address of # elements N.

*For HP Assembly Language usage, refer to paragraph 3-43.

*For HP Assembly Language usage, refer to paragraph 3-43.

VSSB/DVSSB * SCALAR-VECTOR SUBTRACT

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

Memory Address

Performs the vector operation:

V2 = S − V1

This is a eight-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of scalar S.
Word 4 = Address of vector V1.
Word 5 = Address of increment INCR1.
Word 6 = Address of vector V2.
Word 7 = Address of increment INCR2.
Word 8 = Address of # elements N.

VSMY/DVSMY * SCALAR-VECTOR MULTIPLY

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

Memory Address

Performs the vector operation:

V2 = S * V1

This is a eight-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of scalar S.
Word 4 = Address of vector V1.
Word 5 = Address of increment INCR1.
Word 6 = Address of vector V2.
Word 7 = Address of increment INCR2.
Word 8 = Address of # elements N.

VSDV/DVSDV * SCALAR-VECTOR DIVIDE

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

Memory Address

Performs the vector operation:

V2 = S / V1

This is a eight-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of scalar S.
Word 4 = Address of vector V1.
Word 5 = Address of increment INCR1.
Word 6 = Address of vector V2.
Word 7 = Address of increment INCR2.
Word 8 = Address of # elements N.

*For HP Assembly Language usage, refer to paragraph 3-43.

*For HP Assembly Language usage, refer to paragraph 3-43.

VPIV/DVPIV * VECTOR PIVOT

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

Memory Address

Performs the vector operation:

V3 = S * V1 + V2

This is a ten-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of scalar S.
Word 4 = Address of vector V1.
Word 5 = Address of increment INCR1.
Word 6 = Address of vector V2.
Word 7 = Address of increment INCR2.
Word 8 = Address of vector V3.
Word 9 = Address of increment INCR3.
Word 10 = Address of # elements N.

VABS/DVABS * VECTOR ABSOLUTE VALUE

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

Memory Address

Performs the vector operation:

V2 = ABS (V1)

This is a seven-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of vector V1.
Word 4 = Address of increment INCR1.
Word 5 = Address of vector V2.
Word 6 = Address of increment INCR2.
Word 7 = Address of # elements N.

VSUM/DVSUM * VECTOR SUM

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

Memory Address

Performs the vector operation:

SUM = Σ V1

Note that for VSUM the sum is internally accumulated in double precision; the answer is then truncated to single precision.

This is a six-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of scalar SUM.
Word 4 = Address of vector V1.
Word 5 = Address of increment INCR1.
Word 6 = Address of # elements N.

*For HP Assembly Language usage, refer to paragraph 3-43.

*For HP Assembly Language usage, refer to paragraph 3-43.

VNRM/DVNRM * VECTOR NORM

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

Memory Address

Performs the vector operation:

SUM = Σ ABS (V1)

Note that for VNRM the sum is internally accumulated in double precision; the answer is then truncated to single precision.

This is a six-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of scalar SUM.
Word 4 = Address of vector V1.
Word 5 = Address of increment INCR1.
Word 6 = Address of # elements N.

VDOT/DVDOT * VECTOR DOT PRODUCT

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

Memory Address

Performs the vector operation:

DOT = Σ V1 * V2

Note that for VDOT the product and sum is internally accumulated in double precision; the answer is then truncated to single precision.

This is a eight-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of scalar DOT.
Word 4 = Address of vector V1.
Word 5 = Address of increment INCR1.
Word 6 = Address of vector V2.
Word 7 = Address of increment INCR2.
Word 8 = Address of # elements N.

VMAX/DVMAX * VECTOR MAXIMUM VALUE

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

Memory Address

Performs the vector operation:

IMAX = Position (MAX(V1))

Note that IMAX is the position of the maximum of those elements that were tested, as requested by INCR1 and N. If INCR1 # 1, the position is given by:

IPOS = 1 + INCR1 * (IMAX − 1)

This is a six-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of integer IMAX.
Word 4 = Address of vector V1.
Word 5 = Address of increment INCR1.
Word 6 = Address of # elements N.

*For HP Assembly Language usage, refer to paragraph 3-43.

*For HP Assembly Language usage, refer to paragraph 3-43.

VMAB/DVMAB *

VECTOR MAXIMUM ABSOLUTE VALUE

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

Memory Address

Performs the vector operation:

IMAB = Position (MAX(ABS(V1)))

Note that IMAB is the position of the maximum absolute value of those elements that were tested, as requested by INCR1 and N. If INCR1 # 1, the position is given by:

IPOS = 1 + INCR1 * (IMAB − 1)

This is a six-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of integer IMAB.
Word 4 = Address of vector V1.
Word 5 = Address of increment INCR1.
Word 6 = Address of # elements N.

VMIN/DVMIN *

VECTOR MINIMUM VALUE

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

Memory Address

Performs the vector operation:

IMIN = Position (MIN(V1))

Note that IMIN is the position of the minimum absolute value of those elements that were tested, as requested by INCR1 and N. If INCR1 # 1, the position is given by:

IPOS = 1 + INCR1 * (IMIN − 1)

This is a six-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of integer IMIN.
Word 4 = Address of vector V1.
Word 5 = Address of increment INCR1.
Word 6 = Address of # elements N.

VMIB/DVMIB *

VECTOR MINIMUM ABSOLUTE VALUE

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

Memory Address

Performs the vector operation:

IMIB = Position (MIN(ABS(V1)))

Note that IMIB is the position of the minimum absolute value of those elements that were tested, as requested by INCR1 and N. If INCR1 # 1, the position is given by:

IPOS = 1 + INCR1 * (IMIB − 1)

This is a six-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of integer IMIB.
Word 4 = Address of vector V1
Word 5 = Address of increment INCR1.
Word 6 = Address of # elements N.

*For HP Assembly Language usage, refer to paragraph 3-43.

*For HP Assembly Language usage, refer to paragraph 3-43.

VMOV/DVMOV * VECTOR MOVE

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

Memory Address

Performs the vector operation:

V2 = V1

This is a seven-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of vector V1.
Word 4 = Address of increment INCR1.
Word 5 = Address of vector V2.
Word 6 = Address of increment INCR2.
Word 7 = Address of # elements N.

VSWP/DVSWP * VECTOR SWAP

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

Memory Address

Performs the vector operation:

V1 < = = > V2

This is a seven-word instruction, where

Word 1 = Instruction code.
Word 2 = Sub-opcode.
Word 3 = Address of vector V1.
Word 4 = Address of increment INCR1.
Word 5 = Address of vector V2.
Word 6 = Address of increment INCR2.
Word 7 = Address of # elements N.

*For HP Assembly Language usage, refer to paragraph 3-43.

3-42. EXECUTION TIMES AND INTERRUPTS

Table 3-10 lists the typical execution times for the VIS instructions. VIS times are composed of two parts: a fixed time per instruction execution, and a per-element loop time. Thus the time to process 100 elements equals the fixed time plus 100 multiplied by loop time.

Also listed in Table 3-10 are the maximum periods of non-interruptable instruction execution. When a VIS instruction is interrupted, the current state of execution is stored in the sub-opcode memory location, as well as the A,B,X, and Y registers. When re-entered following the interrupt processing, the VIS instruction will resume execution from the point of suspension.

Table 3-10. Typical Vector Instruction Set Execution Times (All times in microseconds)

INSTRUCTION	FIXED TIME	LOOP TIME	MAX. NON-INT
VADD	14.6	7.7	21
VSUB	14.6	7.7	21
VMPY	14.8	7.7	21
VDIV	16.0	7.7	21
VSAD	13.6	7.7	21
VSSB	13.6	7.7	21
VSMY	13.9	7.7	21
VSDV	14.0	7.7	21
VPIV	16.6	10.2	26
VABS	12.4	6.9	19
VSUM	19.7	5.9	23
VNRM	20.0	5.9	23
VDOT	19.7	10.5	29
VMAX	8.5	7.9	17
VMAB	7.6	8.4	17
VMIN	8.5	7.9	17
VMIB	7.6	8.4	17
VMOV	11.7	4.4	17
VSWP	11.8	7.5	20
DVADD	15.8	11.1	24
DVSUB	15.8	11.1	24
DVMPY	18.0	11.1	24
DVDIV	20.5	12.7	24
DVSAD	15.6	12.2	24
DVSSB	15.6	12.2	24
DVSMY	17.6	12.6	24
DVSDV	18.8	11.1	24
DVPIV	18.5	14.6	31
DVABS	13.3	9.4	23
DVSUM	21.1	6.1	23
DVNRM	21.2	6.1	23
DVDOT	21.2	12.5	31
DVMAX	8.4	9.8	18
DVMAB	7.4	10.6	18
DVMIN	8.4	9.9	18
DVMIB	7.4	10.5	18
DVMOV	12.5	6.4	19
DVSWP	13.0	11.3	25

3-43. ASSEMBLY LANGUAGE

New instructions not recognized by the HP Assembler require different handling in HP Assembly Language programming. These instructions are asterisked in the preceding paragraphs and must be used in the form: JSB x, where x is the instruction. (The instruction, x, must be declared as an external at the beginning of the assembly language program.) Most of these instructions correspond to library subroutines* and must be implemented into HP RTE systems (as described in the following paragraph) to enable their execution in hardware-firmware instead of in software.

3-44. RTE IMPLEMENTATION

New instructions are implemented in an RTE-M, RTE-II, or RTE-IV system simply by changing library entry points during the parameter input phase of system generation. (Refer to the appropriate RTE manual for the system generation procedure.) Using the list of entry point opcodes given in Table 3-11, make the entry changes as indicated below:

```
.FAD,RP,105000
.FSB,RP,105020
 .
 .
 .
.DVMOV,RP,105472
.DVSWP,RP,105473
```

Alternatively, entry points may be changed by loading (via LOADR) a "replacement" program when user programs are loaded. An example replacement program is shown below.

```
ASMB,L,R
        NAM REPLA
        ENT .XADD, .XSUB, . . . , . PACK
.FAD RPL 105000B

.FSB RPL 105020B
 .
 .
 .
.DVMOV RPL 105472
.DVSWP RPL 105473
       END
```

*Refer to the Relocatable Library Reference Manual, part no. 24998-90001.

Table 3-11. Instructions and Opcodes for RTE Implementation

INSTRUCTION MNEMONIC	OCTAL OPCODE	INSTRUCTION MNEMONIC	OCTAL OPCODE
.FAD	105000	DDINT	105217
.FSB	105020	.GOTO	105221
.FMP	105040	..MAP	105222
.FDV	105060	.ENTR	105223
IFIX*	105100	.ENTP	105224
.FIXD	105104	.PWR2	105225
FLOAT*	105120	.FLUN	105226
.FLTD	105124	$SETP	105227
.XADD	105001	.PACK	105230
XSUB	105021	.CFER	105231
.XMPY	105041	..FCM	105232
.XDIV	105061	..TCM	105233
.XFXS	105101	.DAD	105014
.DINT*	105101	.DSB	105034
.FIXD	105104	.DMP	105054
.XFXD	105105	.DDI	105074
.XFTS	105121	.DSBR	105114
.IDBL*	105121	.DDIR	105134
.FLTD	105124	.DNG	105203
.XFTD	105125	.DCO	105204
.TADD	105002	.DIN	105210
.TSUB	105022	.DDE	105211
.TMPY	105042	.DIS	105212
.TDIV	105062	.DDS	105213
.TFXS	105102	.VECT†	101460
.TINT*	105102	VPIV	101461
.TFXD	105106	VABS	101462
.TFTS	105122	VSUM	101463
.ITBL*	105122	VNRM	101464
.TFTD	105126	VDOT	101465
TAN	105320	VMAX	101466
SQRT	105321	VMAB	101467
ALOG	105322	VMIN	101470
ATAN	105323	VMIB	101471
COS	105324	VMOV	101472
SIN	105325	VSWP	101473
EXP	105326	.ERES†	101474
ALOGT	105327	.ESEG†	101475
TANH	105330	.VSET†	101476
DPOLY	105331	.DVCT†	105460
/CMRT†	105332	DVPIV	105461
/ATLG	105333	DVABS	105462
.FPWR	105334	DVSUM	105463
.TPWR	105335	DVNRM	105464
DBLE	105201	DVDOT	105465
SNGL	105202	DVMAX	105466
.DFER	105205	DVMAB	105467
.BLE	105207	DVMIN	105470
.NGL	105214	DVMIB	105471
.XFER	105220	DVMOV	105472
.XPAK	105206	DVSWP	105473
.XCOM	105215		
..DCM	105216		

*These mnemonics are FORTRAN intrinsics that are equivalenced to the appropriate opcodes.

†Not directly user callable. Used by HP 1000 software

DYNAMIC MAPPING SYSTEM

SECTION IV

The basic addressing space of the HP 1000 Computers is 32,768 words, which is referred to as *logical* memory. The amount of MOS memory actually installed in the computer system is referred to as *physical* memory. An HP 1000 Computer with the Dynamic Mapping System (DMS) has an addressing capability for one million words of physical memory. The DMS allows logical memory to be mapped into physical memory through the use of four dynamically alterable memory maps. (The DMS is standard for the HP 2117F and optional for all other HP 1000 Computers.)

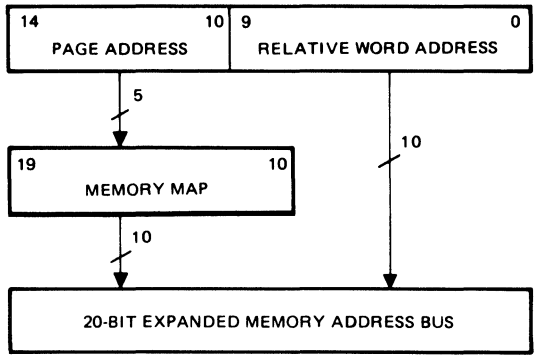


Figure 4-2. Expanded Memory Addressing Scheme

4-1. MEMORY ADDRESSING

The basic memory addressing scheme provides for addressing 32 pages of logical memory, each of which consists of 1,024 words. This memory is addressed through a 15-bit memory address bus shown in Figure 4-1. The upper 5 bits of this bus provide the page address and the lower 10 bits provide the relative word address within the page.

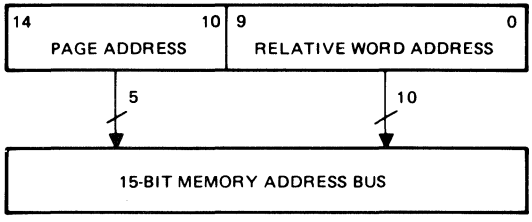


Figure 4-1. Basic Memory Addressing Scheme

The Memory Expansion Module (MEM), which is part of the DMS option, converts the 5-bit page address into a 10-bit page address and thereby allows 1,024 (2^{10}) pages to be addressed. This conversion is accomplished by allowing the original 5-bit address to identify one of the 32 12-bit registers within a "memory map." Each of these map registers contains the new user-specified 10-bit page address. This new page address is combined with the original 10-bit relative address to form a 20-bit memory address bus as shown in Figure 4-2.

4-2. MAP REGISTER LOADING

Conversion of the basic 16-bit word data format to and from the map register 12-bit word data format is shown in Figure 4-3. Bits 13 through 10 of the basic data format are not used by the memory map registers. Read and write memory protect violations are discussed in paragraph 4-3.

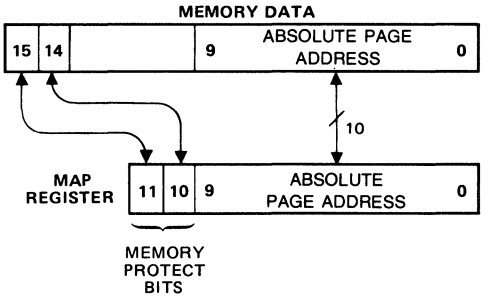


Figure 4-3. Basic Word Format Vs Map Register Format

4-3. STATUS AND VIOLATION REGISTERS

The MEM also includes a status register and a violation register. As shown in Table 4-1, the MEM status register

contents enable the programmer to determine whether the MEM was enabled or disabled at the time of the last interrupt and the address of the base page fence. The MEM violation register contents enable the programmer to determine whether a fault occurred in the hardware or the software so that the proper corrective steps may be taken. Refer to Table 4-2.

Table 4-1. MEM Status Register Format

BIT	SIGNIFICANCE
15	0 = MEM disabled at last interrupt 1 = MEM enabled at last interrupt
14	0 = System map selected at last interrupt 1 = User map selected at last interrupt
13	0 = MEM disabled currently 1 = MEM enabled currently
12	0 = System map selected currently 1 = User map selected currently
11	0 = Protected mode disabled currently 1 = Protected mode enabled currently
10	Portion mapped*
9	Base page fence bit 9
8	Base page fence bit 8
7	Base page fence bit 7
6	Base page fence bit 6
5	Base page fence bit 5
4	Base page fence bit 4
3	Base page fence bit 3
2	Base page fence bit 2
1	Base page fence bit 1
0	Base page fence bit 0

*Bit 10	Mapped Address (M)
0	Fence $\leq$ M $<$ 2000_8
1	1 $<$ M $<$ Fence

Note: The base page fence separates the reserved (mapped) memory from the shared (unmapped) memory. Bit 10 specifies which area is reserved (mapped). (Refer to LFA and LFB instructions contained in paragraph 4-6.)

Table 4-2. MEM Violation Register Format

BIT	SIGNIFICANCE
15	Read violation*
14	Write violation*
13	Base page violation*
12	Privileged instruction violation*
11	Reserved
10	Reserved
9	Reserved
8	Reserved
7	0 = ME bus disabled at violation 1 = ME bus enabled at violation
6	0 = MEM disabled at violation 1 = MEM enabled at violation
5	0 = System map enabled at violation 1 = User map enabled at violation
4	Map address bit 4
3	Map address bit 3
2	Map address bit 2
1	Map address bit 1
0	Map address bit 0

*Significant when associated bit is set.

Any attempt to read from a read-protected page will result in a read violation and the memory read will not occur. Any attempt to write into a write-protected page will result in a write violation and the memory will not be altered. In addition, if a page is write protected, a jump or jump indirect instruction to that page will cause a write violation and the jump will not occur. It should be noted that all violation rules are ignored for DCPC signals.

If a read or write violation occurs, the MEM signals the memory protect logic that a violation has occurred which causes the memory protect logic to generate an interrupt. As discussed in paragraph 6-3, memory violations are interrupted to select code 05 and a DMS violation can be distinguished from a memory protect violation by executing an SFS 05 instruction. If the skip occurs, DMS is in violation; if no skip occurs, memory protect is in violation.

4-4. MAP SEGMENTATION

All registers within the memory map are dynamically alterable. To maximize the system performance capability, the MEM includes four separate memory maps: the User Map, System Map, and two Dual-Channel Port Controller (DCPC) Maps. (See Figure 4-4.) These maps, which are manipulated through the use of 38 machine-language instructions, are addressed as a contiguous register block. It should be noted that the base page fence applies to both the System Map and the User Map.

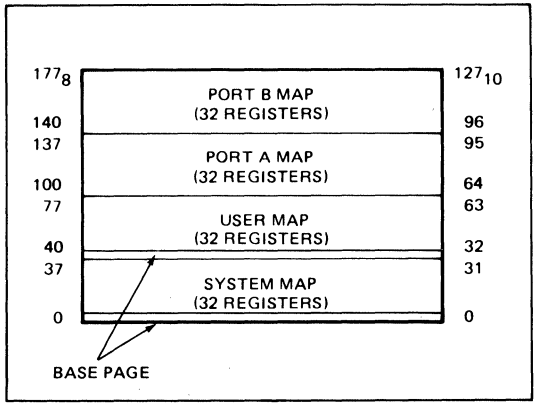


Figure 4-4. Map Segmentation

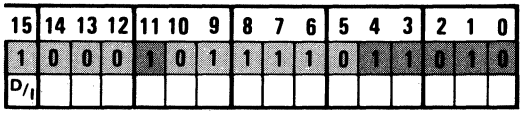
4-5. POWER FAIL CHARACTERISTICS

A power failure automatically enables the System Map, and a minimum of 500 microseconds is assured the programmer for executing a power fail routine. Since all maps are disabled and none are considered valid upon the restoration of power, the power fail routine should include instructions to save as many maps as desired.

4-6. DMS INSTRUCTION CODING

Machine language coding and definitions of the 38 Dynamic Mapping System instructions are provided on this and following pages. A sample map load and enable routine is given in paragraph 4-8.

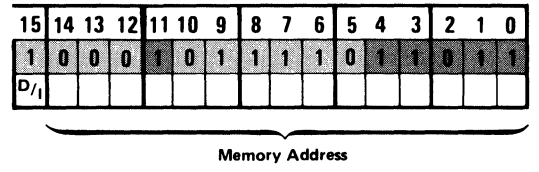
DJP DISABLE MEM AND JUMP



Disables the translation and protection features of the MEM hardware. Prior to disabling, the P-register is set to the effective memory address. As a result of executing this instruction, normal I/O interrupts are held off until the first opportunity following the fetch of the next instruction, unless three or more levels of indirect addressing are used.

This instruction will normally generate an MEM violation when executed in the protected mode. In this case, the status of the MEM is not affected and the jump will not occur; however, if the System map is enabled, the instruction is allowed.

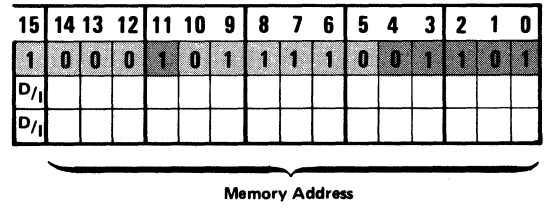
DJS DISABLE MEM AND JUMP TO SUBROUTINE



Disables the translation and protection features of the MEM hardware. Prior to disabling, the P-register is set one count past the effective memory address (m + 1) and the return address is stored in location m. As a result of executing this instruction, normal I/O interrupts are held off until the first opportunity following the fetch of the next instruction, unless three or more levels of indirect addressing are used.

This instruction will normally generate an MEM violation when executed in the protected mode. In this case, the status of the MEM is not affected and the jump will not occur; however, if the System map is enabled, the instruction is allowed.

JRS JUMP AND RESTORE STATUS



Causes the status of the MEM to be restored. This is a three-word instruction where

Word 1 = Instruction code,
Word 2 = Status word address, and
Word 3 = Jump address.

Only bits 15 and 14 of the status word are used; the remaining bits (13-0) of the status word are ignored. Bits 15 and 14 restore the MEM status as follows:

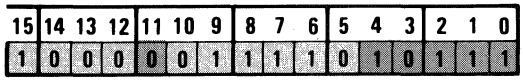
Bit 15 = 0 = MEM will be disabled
= 1 = MEM will be enabled

Bit 14 = 0 = System map will be selected
= 1 = User map will be selected

As a result of executing this instruction, normal I/O interrupts are held off until after the fetch of the next instruction, unless a total of three or more levels of indirect addressing are used in Word 2 (status word address) and Word 3 (jump address). For example, if Word 2 contains one level of indirect addressing and Word 3 contains two levels of indirect addressing, interrupts will not be held off past the fetch of the next instruction.

This instruction will normally generate an MEM violation when executed in the protected mode. In this case, the status of the MEM is not affected and the jump will not occur; however, if the System map is enabled, the instruction is allowed.

LFA LOAD FENCE FROM A

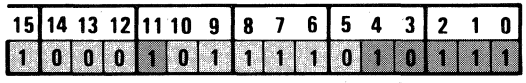


Loads the contents of the A-register into the base page fence register. Bits 9-0 of the A-register specify the address in page zero where shared (unmapped) memory is separated from reserved (mapped) memory. Bit 10 is used as follows to specify which portion is mapped:

Bit 10	Mapped Address (M)
0	Fence $\leq$ M $<$ 2000_8
1	1 $<$ M $<$ Fence

This instruction will normally generate an MEM violation when executed in the protected mode; however, it is allowed if the System map is enabled. When an MEM violation does occur, the fence is not altered.

LFB LOAD FENCE FROM B

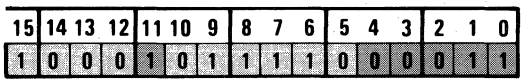


Loads the contents of the B-register into the base page fence register. Bits 9-0 of the B-register specify the address in page zero where shared (unmapped) memory is separated from reserved (mapped) memory. Bit 10 is used as follows to specify which portion is mapped:

Bit 10	Mapped Address (M)
0	Fence $\leq$ M $<$ 2000_8
1	1 $<$ M $<$ Fence

This instruction will normally generate an MEM violation when executed in the protected mode; however, it is allowed if the System map is enabled. When an MEM violation does occur, the fence is not altered.

MBF MOVE BYTES FROM ALTERNATE MAP

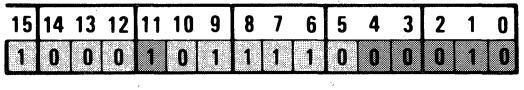


Moves a string of bytes using the alternate program map for source reads and the currently enabled map for destination writes. The A-register contains the source byte address and the B-register contains the destination byte address. The initial byte addresses in the A- and B-registers must be even byte addresses. The byte in bits 15 through 8 of a word is the even byte. The X-register contains the octal number of bytes to be moved. The number of bytes to be moved is restricted to a positive integer greater than zero. If the contents of the X-register is zero, the instruction will be a NOP. If the contents of the X-register is a negative integer, a large indeterminate block of memory will be transferred. Both the source and destination must begin on word boundaries.

The instruction is interruptible on an even number of byte transfers, thus maintaining the even word boundaries in the A- and B-registers. The interrupt routine is expected to save and restore the current contents of the A-, and B-, and X-registers to allow continuation of the instruction at the next entry. When the byte string move is completed, the X-register will always be zero and the A- and B-registers will contain their original value incremented by the number of bytes moved.

This instruction can cause an MEM violation only if read or write protection rules are violated.

MBI MOVE BYTES INTO ALTERNATE MAP

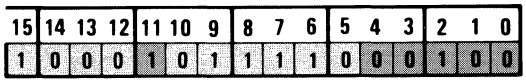


Moves a string of bytes using the currently enabled map for source reads and the alternate program map for destination writes. The A-register contains the source byte address and the B-register contains the destination byte address. The initial byte addresses in the A- and B-registers must be even byte addresses. The byte in bits 15 through 8 of a word is the even byte. The X-register contains the octal number of bytes to be moved. The number of bytes to be moved is restricted to a positive integer greater than zero. If the contents of the X-register is zero, the instruction will be a NOP. If the contents of the X-register is a negative integer, a large indeterminate block of memory will be transferred. Both the source and destination must begin on word boundaries.

The instruction is interruptible on an even number of byte transfers, thus maintaining the even word boundaries in the A- and B-registers. The interrupt routine is expected to save and restore the current contents of the A-, B-, and X-registers to allow continuation of the instruction at the next entry. When the byte string move is completed, the X-register will always be zero and the A- and B-registers will contain their original value incremented by the number of bytes moved.

This instruction will always cause an MEM violation when executed in the protected mode and no bytes will be transferred.

MBW MOVE BYTES WITHIN ALTERNATE MAP

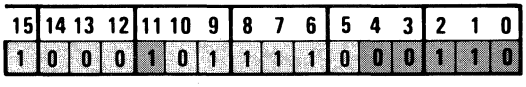


Moves a string of bytes with both the source and destination addresses established through the alternate program map. The A-register contains the source byte address and the B-register contains the destination byte address. The initial byte addresses in the A- and B-registers must be even byte addresses. The byte in bits 15 through 8 of a word is the even byte. The X-register contains the octal number of bytes to be moved. The number of bytes to be moved is restricted to a positive integer greater than zero. If the contents of the X-register is zero, the instruction will be a NOP. If the contents of the X-register is a negative integer, a large indeterminate block of memory will be transferred. Both the source and destination must begin on word boundaries.

The instruction is interruptible on an even number of byte transfers, thus maintaining the even word boundaries in the A- and B-registers. The interrupt routine is expected to save and restore the current contents of the A-, B-, and X-registers to allow continuation of the instruction at the next entry. When the byte string move is completed, the X-register will always be zero and the A- and B-registers will contain their original value incremented by the number of bytes moved.

This instruction will always cause an MEM violation when executed in the protected mode and no bytes will be transferred.

MWF MOVE WORDS FROM ALTERNATE MAP

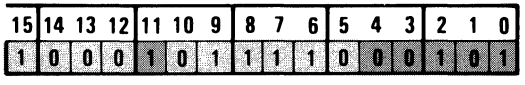


Moves a string of words using the alternate program map for source reads and the currently enabled map for destination writes. The A-register contains the source address and the B-register contains the destination address. The X-register contains the octal number of words to be moved. The number of words to be moved is restricted to a positive integer greater than zero. If the contents of the X-register is zero, the instruction will be a NOP. If the contents of the X-register is a negative integer, a large indeterminate block of memory will be transferred.

The instruction is interruptible. The interrupt routine is expected to save and restore the current contents of the A-, B-, and X-registers to allow continuation of the instruction at the next entry. When the word string move is completed, the X-register will always be zero and the A- and B-registers will contain their original value incremented by the number of words moved.

This instruction can cause an MEM violation only if read and write protection rules are violated.

MWI MOVE WORDS INTO ALTERNATE MAP

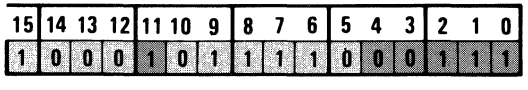


Moves a string of words using the currently enabled map for source reads and the alternate program map for destination writes. The A-register contains the source address and the B-register contains the destination address. The X-register contains the octal number of words to be moved. The number of words to be moved is restricted to a positive integer greater than zero. If the contents of the X-register is zero, the instruction will be a NOP. If the contents of the X-register is a negative integer, a large indeterminate block of memory will be transferred.

The instruction is interruptible. The interrupt routine is expected to save and restore the current contents of the A-, B-, and X-registers to allow continuation of the instruction at the next entry. When the word string move is conpleted, the X-register will always be zero and the A- and B-registers will contain their original value incremented by the number of words moved.

This instruction will always cause an MEM violation when executed in the protected mode and no words will be transferred.

MWW MOVE WORDS WITHIN ALTERNATE MAP



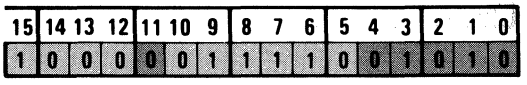
Moves a string of words with both the source and destination addresses established through the alternate program map. The A-register contains the source address and the B-register contains the destination address. The X-register contains the octal number of words to be moved.

The number of words to be moved is restricted to a positive integer greater than zero. If the contents of the X-register is zero, the instruction will be a NOP. If the contents of the X-register is a negative integer, a large indeterminate block of memory will be transferred.

The instruction is interruptible. The interrupt routine is expected to save and restore the current contents of the A-, B-, and X-registers to allow continuation of the instruction at the next entry. When the word string move is completed, the X-register will always be zero and the A- and B-registers will contain their original value incremented by the number of words moved.

The instruction will always cause an MEM violation when executed in the protected mode and no words will be transferred.

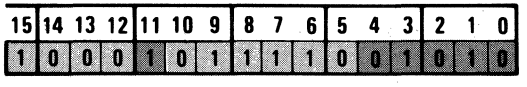
PAA LOAD/STORE PORT A MAP PER A



Transfers the 32 Port A map registers to or from memory. If bit 15 of the A-register is clear, the Port A map is *loaded* from memory starting from the address specified in bits 10-4 of the A-register. If bit 15 of the A-register is set, the Port A map is *stored* into memory starting at the address specified in bits 14-0 of the A-register. When the load/store operation is complete, the A-register will be incremented by 32 to allow multiple map instructions.

An attempt to load any map register when in the protected mode will cause an MEM violation. An attempt to store the Port A map is allowed within the constraints of write protected memory.

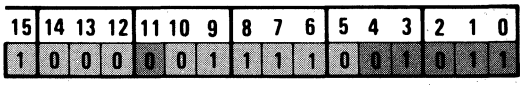
PAB LOAD/STORE PORT A MAP PER B



Transfers the 32 Port A map registers to or from memory. If bit 15 of the B-register is clear, the Port A map is *loaded* from memory starting from the address specified in bits 14-0 of the B-register. If bit 15 of the B-register is set, the Port A map is *stored* into memory starting at the address specified in bits 14-0 of the B-register. When the load/store operation is complete, the B-register will be incremented by 32 to allow multiple map instructions.

An attempt to load any map register when in the protected mode will cause an MEM violation. An attempt to store the Port A map is allowed within the constraints of write protected memory.

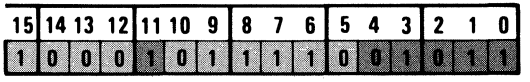
PBA LOAD/STORE PORT B MAP PER A



Transfers the 32 Port B map registers to or from memory. If bit 15 of the A-register is clear, the Port B map is *loaded* from memory starting from the address specified in bits 14-0 of the A-register. If bit 15 of the A-register is set, the Port B map is *stored* into memory starting at the address specified in bits 14-0 of the A-register. When the load/store operation is complete, the A-register will be incremented by 32 to allow multiple map instructions.

An attempt to load any map register when in the protected mode will cause an MEM violation. An attempt to store the Port B map is allowed within the constraints of write protected memory.

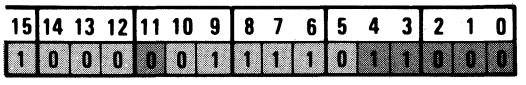
PBB LOAD/STORE PORT B MAP PER B



Transfers the 32 Port B map registers to or from memory. If bit 15 of the B-register is clear, the Port B map is *loaded* from memory starting from the address specified in bits 14-0 of the B-register. If bit 15 of the B-register is set, the Port B map is *stored* into memory starting at the address specified in bit 14-0 of the B-register. When the load/store operation is complete, the B-register will be incremented by 32 to allow multiple map instructions.

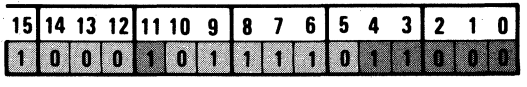
An attempt to load any map register when in the protected mode will cause an MEM violation. An attempt to store the Port B map is allowed within the constraints of the write protected memory.

RSA READ STATUS REGISTER INTO A



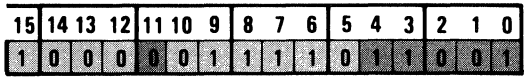
Reads the contents of the MEM status register into the A-register. This instruction can be executed at any time. The format of the MEM status register is given in Table 4-1.

RSB READ STATUS REGISTER INTO B



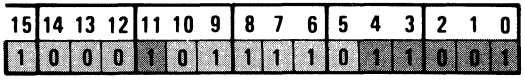
Reads the contents of the MEM status register into the B-register. This instruction can be executed at any time. The format of the MEM status register is given in Table 4-1.

RVA READ VIOLATION REGISTER INTO A



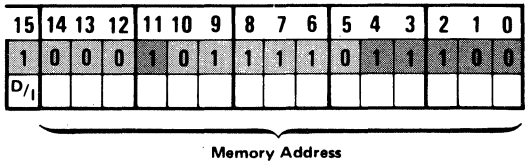
Reads the contents of the MEM violation register into the A-register. This instruction can be executed at any time. The format of the MEM violation register is given in Table 4-2.

RVB READ VIOLATION REGISTER INTO B



Reads the contents of the MEM violation register into the B-register. This instruction can be executed at any time. The format of the MEM violation register is given in Table 4-2.

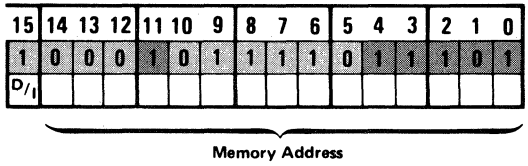
SJP ENABLE SYSTEM MAP AND JUMP



Causes the MEM hardware to use the set of 32 map registers, referred to as the System map, for translating all programmed memory references. Prior to enabling the System map, the P-register is set to the effective memory address. As a result of executing this instruction, normal I/O interrupts are held off until the first opportunity following the fetch of the next instruction, unless three or more levels of indirect addressing are used.

This instruction will normally generate an MEM violation when executed in the protected mode. In this case, the status of the MEM is not affected and the jump will not occur; however, if the System map is enabled, the instruction is allowed and effectively executes a JMP *+1,I.

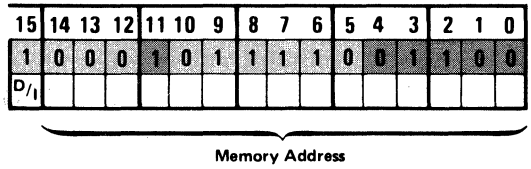
SJS ENABLE SYSTEM MAP AND JUMP TO SUBROUTINE



Causes the MEM hardware to use the set of 32 map registers, referred to as the System map, for translating all programmed memory references. Prior to enabling the System map, the P-register is set one count past the effective memory address (m + 1). After enabling the System map, the return address is stored in m. As a result of executing this instruction, normal I/O interrupts are held off until the first opportunity following the fetch of the next instruction, unless three or more levels of indirect addressing are used.

This instruction will normally generate an MEM violation when executed in the protected mode. In this case, the status of the MEM is not affected and the jump will not occur; however, if the system map is enabled, the instruction is allowed and effectively executes a JSB *+1,I.

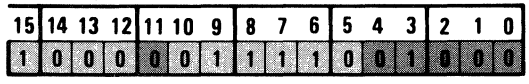
SSM STORE STATUS REGISTER INTO MEMORY



Stores the 16-bit contents of the MEM status register into the address memory location. The status register contents are not altered. This instruction is used in conjunction with the JRS instruction to allow easy processing of interrupts, which always select the System map (if the MEM is enabled). The format of the MEM status register is listed in Table 4-1.

This instruction can cause an MEM violation only if write protection rules are violated.

SYA LOAD/STORE SYSTEM MAP PER A



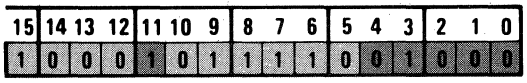
Transfers the 32 System map registers to or from memory. If bit 15 of the A-register is clear, the System map is *loaded* from memory starting from the address specified in bits 14-0 of the A-register. If bit 15 of the A-register is set, the System map is *stored* into memory starting at the address specified in bits 14-0 of the A-register. When the load/store operation is complete, the A-register will be incremented by 32 to allow multiple map instructions.

NOTE

If not in the protected mode, the MEM provides no protection against altering the contents of maps while they are currently enabled.

An attempt to load any map in the protected mode will cause an MEM violation. An attempt to store the System map is allowed within the constraints of write protected memory.

SYB LOAD/STORE SYSTEM MAP PER B



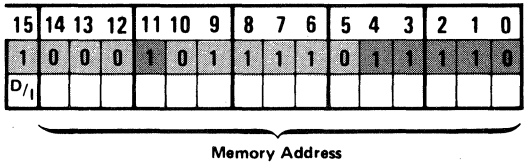
Transfers the 32 System map registers to or from memory. If bit 15 of the B-register is clear, the System map is *loaded* from memory starting from the address specified in bits 14-0 of the B-register. If bit 15 of the B-register is set, the System map is *stored* into memory starting at the address specified in bits 14-0 of the B-register. When the load/store operation is complete, the B-register will be incremented by 32 to allow multiple map instructions.

NOTE

If not in the protected mode, the MEM provides no protection against altering the contents of maps while they are currently enabled.

An attempt to load any map in the protected mode will cause an MEM violation. An attempt to store the System map is allowed within the constraints of write protected memory.

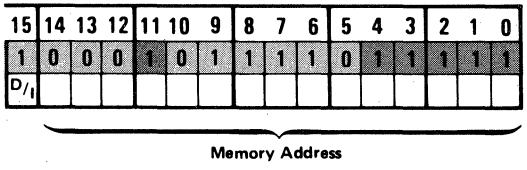
UJP ENABLE USER MAP AND JUMP



Causes the MEM hardware to use the set of 32 map registers, referred to as the User map, for translating all programmed memory references. Prior to enabling the User map, the P-register is set to the effective memory address. As a result of executing this instruction, normal I/O interrupts are held off until the first opportunity following the fetch of the next instruction, unless three or more levels of indirect addressing are used.

This instruction will normally generate an MEM violation when executed in the protected mode. In this case, the status of the MEM is not affected and the jump will not occur; however, if the System map is enabled, the instruction is allowed.

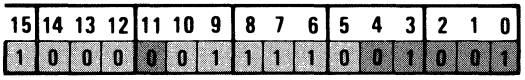
UJS ENABLE USER MAP AND JUMP TO SUBROUTINE



Causes the MEM hardware to use the set of 32 map registers, referred to as the User map, for translating all programmed memory references. Prior to enabling the User map, the P-register is set one count past the effective memory address (m + 1). After enabling the System map, the return address is stored in m. As a result of executing this instruction, normal I/O interrupts are held off until the first opportunity following the fetch of the next instruction, unless three or more levels of indirect addressing are used.

This instruction will normally generate an MEM violation when executed in the protected mode. In this case, the status of the MEM is not affected and the jump will not occur; however, if the System map is enabled, the instruction is allowed.

USA LOAD/STORE USER MAP PER A



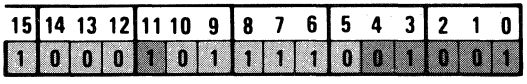
Transfers the 32 User map registers to or from memory. If bit 15 of the A-register is clear, the User map is *loaded* from memory starting from the address specified in bits 14-0 of the A-register. If bit 15 of the A-register is set, the User map is *stored* into memory starting at the address specified in bits 14-0 of the A-register. When the load/store operation is complete, the A-register will be incremented by 32 to allow multiple map instructions.

NOTE

If not in the protected mode, the MEM provides no protection against altering the contents of maps while they are currently enabled.

An attempt to load any map in the protected mode will cause an MEM violation. An attempt to store the User map is allowed within the constraints of write protected memory.

USB LOAD/STORE USER MAP PER B



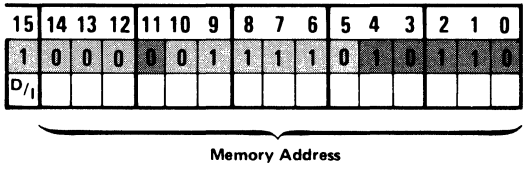
Transfer the 32 User map registers to or from memory. If bit 15 of the B-register is clear, the User map is *loaded* from memory starting from the address specified in bits 14-0 of the B-register. If bit 15 of the B-register is set, the User map is *stored* into memory starting at the address specified in bits 14-0 of the B-register. When the load/store operation is complete, the B-register will be incremented by 32 to allow multiple map instructions.

NOTE

If not in the protected mode, the MEM provides no protection against altering the contents of maps while they are currently enabled.

Any attempt to load any map in the protected mode will cause an MEM violation. An attempt to store the User map is allowed within the constraints of write protected memory.

XCA CROSS COMPARE A

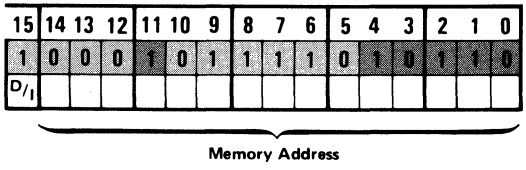


Compares the contents of the A-register with the contents of the addressed memory location. If the two 16-bit words are not identical, the next instruction is skipped; i.e., the P-register advances three counts instead of two counts. If the two words are identical, the next instruction is executed. Neither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

This instruction uses the alternate program map to determine the addressed memory location. If the MEM is currently disabled, then a compare directly with physical memory occurs.

This instruction will cause an MEM violation only if read protection rules are violated.

XCB CROSS COMPARE B

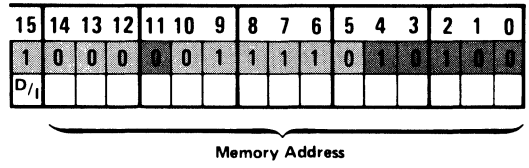


Compares the contents of the B-register with the contents of the addressed memory location. If the two 16-bit words are not identical, the next instruction is skipped; i.e., the P-register advances three counts instead of two counts. If the two words are identical, the next instruction is executed. Neither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

This instruction uses the alternate program map to determine the addressed memory location. If the MEM is currently disabled, then a compare directly with physical memory occurs.

This instruction will cause an MEM violation only if read protection rules are violated.

XLA CROSS LOAD A

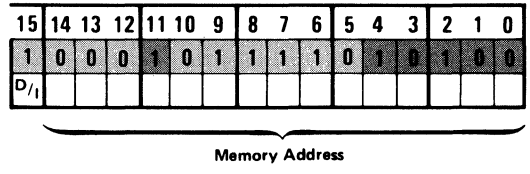


Loads the contents of the specified memory address into the A-register. The contents of the memory cell are not altered.

This instruction uses the alternate program map to fetch the operand. If the MEM is currently disabled, then a load directly from physical memory occurs.

This instruction will cause an MEM violation only if read protection rules are violated.

XLB CROSS LOAD B

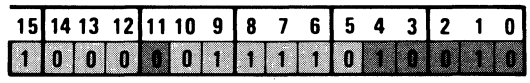


Loads the contents of the specified memory address into the B-register. The contents of the memory cell are not altered.

This instruction uses the alternate program map to fetch the operand. If the MEM is currently disabled, then a load directly from physical memory occurs.

This instruction will cause an MEM violation only if read protection rules are violated.

XMA TRANSFER MAPS INTERNALLY PER A



Transfers a copy of the entire contents (32 map registers) of the System map or the User map to the Port A map or

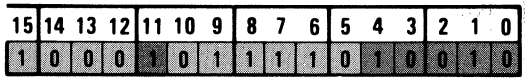
the Port B map as determined by the control word in the A-register:

Bit*	Significance
15	0 = System Map 1 = User Map
0	0 = Port A Map 1 = Port B Map

*Bits 14-1 are ignored.

This instruction will always generate an MEM violation when executed in the protected mode.

XMB TRANSFER MAPS INTERNALLY PER B



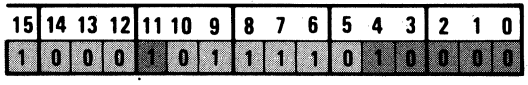
Transfers a copy of the entire contents (32 map registers) of the System map or the User map to the Port A map or the Port B map as determined by the control word in the B-register:

Bit*	Significance
15	0 = System Map 1 = User Map
0	0 = Port A Map 1 = Port B Map

*Bits 14-1 are ignored.

This instruction will always generate an MEM violation when executed in the protected mode.

XMM TRANSFER MAPS OR MEMORY



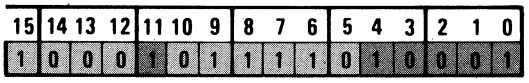
Transfers a number of words either from sequential memory locations to sequential map registers or vice versa. The A-register points to the first map register to be accessed and the B-register points to the first word of a group of words (table) in sequential memory locations. The X-register indicates the number of maps (0 to 127_{10}) to be transferred. If the content of the X-register is a positive integer, words are moved from memory to map registers; if the content is a negative integer, words are moved from map registers to memory.

Map registers are addressed as a contiguous space and a wraparound count from 127 to 0 can and will occur. It is the programmer's responsibility to avoid this error.

The contents of the maps are transferred in blocks of 16 registers or less. This instruction is interruptible only after each block has been completely transferred. The A-, B-, and X-registers are then reset to allow reentry at a later time. The X-register will always be zero at the completion of the instruction an the A- and B-registers will be advanced by the number of registers moved.

An attempt to load any map register in the protected mode will generate an MEM violation. An attempt to store map registers is allowed within the constraints of write protected memory.

XMS TRANSFER MAPS SEQUENTIALLY

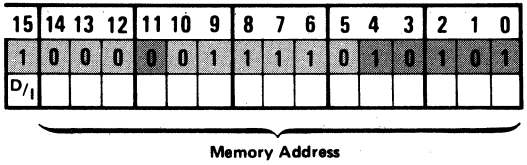


Transfers a number of words to sequential map registers. The A-register points to the first register to be accessed, the B-register contains the base quantity, and the X-register indicates the number of maps (0 to 127_{10}) to be loaded. If the contents of the X-register is a positive integer, the contents of the B-register will be used as the base quantity to be loaded into the first map register. The second register will be loaded with the base quantity plus one, the third register will be loaded with the base quantity plus two, and so forth up to the number of map registers specified in the X-register. If the contents of the X-register is less than or equal to zero, an effective NOP will occur, leaving the contents of the A-, B-, and X-registers unaltered.

This instruction is interruptible after each group of 16 registers has been transferred. The A-, B-, and X-registers are then reset to allow reentry at a later time. The X-register will always be zero at the completion of the instruction and the A- and B-registers will be advanced by the number of registers moved.

An attempt to load any map register in the protected mode will generate an MEM violation.

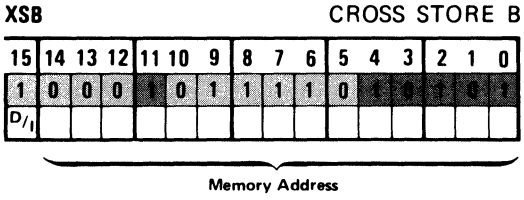
XSA CROSS STORE A



Stores the contents of the A-register into the addressed memory location. The previous contents of the memory cell are lost; the A-register contents are not altered.

This instruction uses the alternate program map for the write operation. If the MEM is currently disabled, then a store directly into physical memory occurs.

This instruction will always cause an MEM violation when executed in the protected mode.



Stores the contents of the B-register into the addressed memory location. The previous contents of the memory cell are lost; the B-register contents are not altered.

This instruction uses the alternate program map for the write operation. If the MEM is currently disabled, then a store directly into physical memory occurs.

This instruction will always cause an MEM violation when executed in the protected mode.

4-7. INSTRUCTION EXECUTION TIMES

Table 4-3 lists the execution times required for the various DMS instructions.

4-8. SAMPLE MAP LOAD/ENABLE ROUTINE

Table 4-4 provides a sample DMS map load and enable routine. This routine begins by loading 32 registers for the System map and 32 registers for the User map and continues by setting the Port A map to the area for User number one. The Port B map is then set to point into a new area where a third User's program would be loaded. Next, the Base Page Fence is set so that the System Fence value is used. Finally, the mapping functions of the DMS are enabled and program control is transferred to the System area beginning at address 1000_8.

4-9. ADDITIONAL DMS DEFINITIONS

The following paragraphs further define the terms "alternate map" and "protected mode" and contain definitive discussions for MEM violations and DCPC operation in a DMS environment.

Table 4-3A. Typical DMS Instruction Execution Times (M-Series)

INSTRUCTION	EXECUTION TIME (μs)
Dynamic Mapping System Group	
DJP, SJP	5.85[a]
DJS, SJS	6.50[a]
JRS	9.10 — 10.40[a]
LFA/B	3.57
MBF, MBI, MBW	6.50[b,c]
MWF, MWI, MWW	3.25[b]
PAA/B, SYA/B	47.125 — 47.80
PBA/B, USA/B	47.125 — 47.80
RSA/B	2.60
RVA/B	2.275
SSM	5.85[a]
UJP	5.525[a]
UJS	6.175[a]
XCA/B	6.175[a]
XLA/B, XSA/B	5.525[a]
XMA/B	15.275 — 16.575
XMM	9.75[e]
XMS	8.45[d]

a. Add 1.3 μs for each indirect address level.
b. Add 2.925 μs for each word moved.
c. Add 3.575 μs for last odd byte.
d. Add 0.975 μs for each word loaded into map register.
e. Add 1.3 μs for each word exchanged between maps and memory.

4-10. ALTERNATE MAP

If the system map is currently enabled, the user map is the alternate map. If the user map is currently enabled, the system map is the alternate map. The DCPC maps are never the alternate maps.

4-11. PROTECTED MODE

If the DMS and memory protect are enabled, the computer is in the protected mode. DMS will operate in the unprotected mode (DMS enabled, memory protect disabled), but none of the DMS safeguards will be operative.

4-12. MEM VIOLATIONS

The MEM violations are designed to safeguard DMS. The four types of violations are read protect, write protect, base page, and privileged instruction. Throughout the following paragraphs, references to logical memory refers to the memory address before mapping and references to physical memory refers to the memory address after mapping.

Table 4-3B. Typical DMS Instruction Execution Times (E-Series)

INSTRUCTION	EXECUTION TIME (μs)		
	Standard Performance Memory	High Performance Memory	Fault Control Memory
DJP,SJP	3.745	3.290	3.850
Each Indirect Level	1.330	0.875	1.400
DJS,SJS	4.410	3.710	4.480
Each Indirect Level		0.875	1.400
JRS	5.320 to 6.055	4.340 to 5.215	5.460 to 6.230
Each Indirect Level	1.330	0.875	1.400
LFA/B	2.170	2.100	2.240
Each Indirect Level	1.330	0.875	1.400
MBF	4.935 to 8.260	4.900 to 7.490	4.970 to 8.400
	+1.820/byte	+1.680/byte	+1.855/byte
MBI	4.760 to 8.120	4.725 to 7.385	4.795 to 8.225
	+1.820/byte	+1.680/byte	+1.855/byte
MBW	4.935 to 7.490	4.900 to 6.965	4.970 to 7.595
	+1.995/byte	+1.680/byte	+2.035/byte
MWF	2.765	2.520	2.800
	+1.820/word	+1.680/word	+1.855/word
MWI,MWW	2.590	2.520	2.625
	+1.820/word	+1.680/word	+1.855/word
PAA/B (load)	50.505	42.420	51.660
(store)	47.145	35.665	47.180
PBA/B (load)	50.680	42.595	51.835
(store)	47.320	35.840	47.425
SYA/B (load)	50.330	42.245	51.485
(store)	45.955	34.510	45.990
RSA/B	2.170	1.715	2.240
RVA/B	2.170	1.680	2.240
SSM	3.710	3.360	3.710
Each Indirect Level	1.330	0.875	1.400
UJP	3.500	3.010	3.640
Each Indirect Level	1.330	0.875	1.400
UJS	4.165	3.430	4.270
Each Indirect Level	1.330	0.875	1.400
USA/B (load)	50.400	42.345	51.555
(store)	46.025	34.580	46.060
XCA/B (no skip)	3.570	3.080	3.675
(skip)	4.235	3.500	4.375
Each Indirect Level	1.330	0.875	1.400
XLA/B	3.500	3.255	3.675
Each Indirect Level	1.330	0.875	1.400
XMA/B	28.980 to 29.750	28.735 to 29.255	29.015 to 29.855
XMM (from memory to map)	4.165	3.920	4.200
	+1.820/word	+1.575/word	+1.855/word
	+0.175/16 words	+0.175/16 words	+0.175/16 words
(from map to memory)	4.270	4.025	4.305
	+1.470/word	+1.330/word	+1.505/word
	+0.175/16 words	+0.175/16 words	+0.175/16 words
XMS	4.095	3.850	4.130
	+1.330/word	+1.330/word	+1.330/word
	+0.175/16 words	+0.175/16 words	+0.175/16 words
XSA/B	3.570	3.080	3.605
Each Indirect Level	1.330	0.875	1.400

Table 4-3C. Typical DMS Instruction Execution Times (F-Series)

INSTRUCTION	EXECUTION TIME (us)	
	High Performance Memory	High Performance Fault Control Memory
DJP,SJP	3.290	3.360
Each Indirect Level	0.875	0.980
DJS	3.710	3.850
Each Indirect Level	0.875	0.980
JRS	4.62 to 5.215	4.760 to 5.355
Each Indirect Level	0.875	0.980
LFA/B	2.100	2.100
Each Indirect Level	0.875	0.980
MBF	4.900 to 7.490 +1.680/byte	4.900 to 7.700 +1.680/byte
MBI	4.725 to 7.385 +1.680/byte	4.725 to 7.595 +1.680/byte
MBW	4.900 to 6.965 +1.750/byte	4.900 to 7.105 +1.820/byte
MWF	2.520 +1.680/word	2.590 +1.680/word
MWI	2.345 +1.680/word	2.415 +1.680/word
MWW	2.345 +1.750/word	2.415 +1.820/word
PAA/B (load)	42.420	44.730
(store)	35.665	36.890
PBA/B (load)	42.595	44.905
(store)	35.840	37.065
SYA/B (load)	42.245	44.555
(store)	34.510	34.580
RSA/B	1.715	1.820
RVA/B	1.680	1.820
SJS	4.130	4.235
Each Indirect Level	0.875	0.980
SSM	3.360	3.430
Each Indirect Level	0.875	0.980
UJP	3.010	3.080
Each Indirect Level	0.875	0.980
UJS	3.430	3.535
Each Indirect Level	0.875	0.980
USA/B (load)	42.315	44.625
(store)	34.580	34.650
XCA/B (no skip)	3.080	3.185
(skip)	3.500	3.675
Each Indirect Level	0.875	0.980
XLA/B	3.255	3.325
Each Indirect Level	0.875	0.980
XMA/B	28.735 to 29.225	28.805 to 29.295
XMM (from memory to map)	3.920 +1.575/word +0.175/16 words	3.990 +1.645/word +0.175/16 words
(from map to memory)	4.025 +1.330/word +0.175/16 words	4.095 +1.330/word +0.175/16 words
XMS	3.850 +1.330/word +0.175/16 words	3.920 +1.330/word +0.175/16 words
XSA/B	3.080	3.150
Each Indirect Level	0.875	0.980

If the computer is in the protected mode and bit 11, the read protect bit, of a system or user map register equals 1, any attempt by the system or user to read from the associated memory page causes a read protect violation and the read does not occur. If the computer is in the unprotected mode, the read occurs. In either case, bit 15 of the MEM violation register will be set to 1. For example, suppose the computer is in the protected mode and the system or user map register 3 contains 4043_8. Any attempt by the system or user to read from page 43_8 using map register 3 (i.e., read from physical addresses in the 106000_8 range), causes a read protect violation.

If the computer is in the protected mode and bit 10, the write protect bit, of a system or user map register equals 1, any attempt by the system or user to write onto the associated memory page causes a write protect violation and the write does not occur. If the computer is in the unprotected mode, the write occurs. In either case, bit 14 of the MEM violation register will be set to 1. For example, suppose the computer is in the protected mode and the system or user map register 3 contains 2043_8. Any attempt by the system or user to write onto page 43_8 using map register 3 (i.e., write onto physical addresses in the 106000_8 range), causes a write protect violation.

If the computer is in the protected mode, any attempt by the system or user to write onto the physical base page causes a base page violation and the write does not occur. If the computer is in the unprotected mode, the write occurs. In either case, bit 13 of the MEM violation register will be set to 1. For example, suppose the computer is in protected mode, the system or user map register 0 contains 0040_8, the base page fence is set at 1000_8, and bit 10 of the MEM status register equals 1 (i.e., logical addresses below the base page fence are mapped). If the system or user attempts to write to a logical memory address of 1500_8, MEM detects that the base page addresses above the base page fence are not mapped and begins to access physical memory address 1500_8. However, MEM then detects that a write to physical base page is being attempted which causes a base page violation and the write does not occur. If the computer was in the unprotected mode, the write would have occurred. In either case, bit 13 of the MEM violation register will be set to 1. If the system or user attempts to write to a logical memory address of 500_8, MEM detects that addresses below the base page fence are mapped and begins to access physical memory address $100000_8 + 500_8 = 100500_8$ where the write will occur providing standard memory protect is not violated. Note that standard memory protect checks the logical address (i.e., 500_8), not the physical address (i.e., 100500_8). Reading from logical or physical base page will not generate a base page violation. From the previous discussion, it can be seen that a DMS memory space has its base page in two pieces which may or may not be contiguous. Regardless, the total base page available for any DMS memory space is 1024 locations. The part of the physical base page accessible by all memory spaces is also referred to as the unmapped or shared part of the base page. Note that the logical addresses of 0 and 1 access the A- and B-registers, respectively.

Table 4-4. Sample DMS Load/Enable Routine

LABEL	OPCODE	OPERAND	COMMENTS
DMS	NOP		DMS Load/Enable Routine
	LDA	S.TABL	Load address of System Map Table
	LDB	U.TABL	Load address of User Map Table
	SYA		Load System Map from memory
	USB		Load User Map from memory
	LDA	A.TABL	Load address of Port A Map Table
	LDB	B.TABL	Load address of Port B Map Table
	PAA		Load Port A Map from memory
	PBB		Load Port B Map from memory
	LDA	FNC,I	Load fence value
	LFA		Load Base Page Fence register
	SJP	SYSTRT,I	Enable System Map and jump to operating system entry point
SYSTRT	OCT	1000	Operating system begins at 1000_8
FNC	DEF	SYSF	Points to System Fence
S.TABL	DEF	SYSTEM	Points to System Table
U.TABL	DEF	US01	Points to first User Table
A.TABL	DEF	US01	Points to first User Table
B.TABL	DEF	US03	Points to third User Table
SYSF	OCT	100	Fence for operating system
SYSTEM	OCT	0	System Map Table
	OCT	1	System Map Table
	OCT	2	System Map Table
.			
.			
.			
US01F	OCT	1000	Fence for first user
US01	OCT	40	First User Map Table
	OCT	41	First User Map Table
	OCT	42	First User Map Table
.			
.			
.			
US02F	OCT	2044	Fence for second user
US02	OCT	100	Second User Map Table
	OCT	101	Second User Map Table
	OCT	102	Second User Map Table
.			
.			
.			
US03F	OCT	0	Fence for third user
US03	OCT	200	Third User Map Table
	OCT	201	Third User Map Table
	OCT	202	Third User Map Table
.			
.			
.			

If the computer is in the protected mode, any attempt by the user to load into any MEM register, except the MEM address register, will cause a privileged instruction violation and the load will not occur. Any attempt by the system to load into any of the MEM map registers will cause a privileged instruction violation and the load will not occur. If the computer is in the unprotected mode, the load occurs. In either case, bit 12 of the MEM violation register will be set to 1. The system can always load into the MEM state register or MEM fence register. Under microprogrammed control the user can always load into the MEM state register or MEM fence register. The system or user can always load into the MEM address register, and can always read the MEM map registers. All MEM violations cause an interrupt to select code 5. Instruction SFS 5 will skip only for an MEM violation, allowing DMS interrupts to be differentiated from memory protect or parity error interrupts.

4-13. DCPC OPERATION IN A DMS ENVIRONMENT

DCPC activity disables the MEM violation logic. Therefore, the DCPC's can read or write physical memory without generating MEM violations. Note that mapping remains enabled during DCPC activity and that the base page partitioning is the same. For example, if a DCPC input transfer were aimed at logical memory addresses 0 to 77777_8, which happened to map to physical addresses 100000_8 to 177777_8, and the conditions cited in the base page examples prevailed, then the input data would be written into physical addresses 100000_8 to 100777_8, 1000_8 to 1777_8, and 102000_8 to 177777_8.

MICROPROGRAMMING

SECTION V

This section contains an introductory discussion of Hewlett-Packard's microprogramming techniques and development. For additional information, refer to the *HP M-Series Computer Microprogramming Reference Manual,* part no. 02108-90032, and the *HP E/F-Series Computer Microprogramming Reference Manual,* part no. 02109-90004.

5-1. THE MICROPROGRAMMED COMPUTER

The control section of a computer is the portion of the computer that directs and controls the other sections; i.e., the memory section, input-output section, and the arithmetic-logic section. In totally hardwired computers, the control section logic is normally "spread out" physically throughout the computer. This design approach makes it impossible to enhance the computer's instruction set without redesign. In contrast, HP 1000 computers have a fully microprogrammed control section, which means that the sequence in which the control functions are performed are made programmable through the use of a technique called microprogramming.

The action taken when any one of the F-Series base set of 128 assembly language instructions is executed is determined by a microprogram associated with the assembly language instruction (these microprograms reside in a special memory called control store); the control section oversees the translation and controls the execution of the microprogram. With this design approach, instruction set enhancements can be made by changing or adding to the set of microprograms that control the machine's execution. Many computers are microprogrammed; however, Hewlett-Packard has taken the concept one step further to offer the power of microprogramming to the user.

5-2. THE MICROPROGRAMMABLE COMPUTER

F-Series computer users can more fully take advantage of the computer's power by utilizing microprogramming. The microprogrammer has more instructions, a more flexible word format, more registers, and faster execution times to work with than does the assembly language programmer. The microinstruction word length is 24 bits which enables concurrent operations to be performed in a single instruction. Microprogrammers can access 12 scratch pad registers in addition to those available to the assembly language programmer and have up to 4096 (M-Series) or 16,384 (E/F-Series) 24-bit words of memory (termed control store) in which to store microprograms. Up to three levels of nested subroutines are possible in E- and F-Series computers. The microprogrammer works in a much faster environment than does the assembly language programmer for two reasons. One, since microinstructions have access to most of the internal parts of the computer's architecture, fewer memory fetches are required to accomplish most tasks. Two, the microinstruction execution time of 325 (M-Series) or 175 or 280 (E/F-Series) nanoseconds is much faster than the typical assembly instruction execution time of 1 to 2 microseconds.

These capabilities are easily taken advantage of by HP 1000 computer users through the extensive support provided by Hewlett-Packard. Some of the more important benefits of Hewlett-Packard's microprogramming are given in the following paragraphs.

5-3. CUSTOMIZED INSTRUCTIONS

Through the use of microprogramming, the computer's assembly language instruction set can be expanded with instructions tailored for specific applications. By adding special purpose instruction sets, the general purpose computer can be uniquely adapted for a certain job and thus become very efficient at that job. HP 1000 users can easily design their own instructions or purchase HP-supplied instruction sets such as the Dynamic Mapping System instructions. Applications that may be profitably microcoded include arithmetic calculations, I/O device driver programs, sorts and table searches, pseudo-DCPC operations, and special IBL orders.

Microprogramming is very similar to assembly language programming, although it is more powerful in many ways. Some knowledge of the internal structure of the computer is required, but once this knowledge is attained, the increased power and flexibility of microprogramming can ease the solution of many programming tasks. Microprograms are easily callable by assembly or higher level language programs. An extensive set of debugging aids, software analysis aids, and documentation is available to make microprogramming easy and efficient.

5-4. SYSTEM SPEED

Microprogramming often-used routines will typically decrease program execution time by factors of two to ten and sometimes by as much as twenty or more. Software routines can be made to execute at the hardware speeds of the microprogram environment and the additional regis-

ters available to the microprogrammer can serve to eliminate many time-consuming memory fetches.

5-5. MEMORY SPACE AND SECURITY

By converting software routines into microprograms, space in main memory that would normally be required for time-critical routines can be freed for other uses. The routines remain instantly callable, as opposed to routines stored in a peripheral device. Microprograms are also less accessible than conventional software which affords a higher degree of security to microcoded routines.

5-6. DEVELOPING MICROPROGRAMS

Developing microprograms is similar to developing assembly language programs; assembling and interactive debugging of microprograms is done with the aid of the HP Micro Assembler and Micro Debug Editor. Since the user will not normally want to microcode all of a certain program, some analysis is required to determine which segment(s) of the assembly language program can be most profitably converted to microcode. This analysis is easily done with the use of an HP contributed library program called the Activity Profile Generator (APG). The APG enables the user to determine where in a program the CPU is spending most of its time; by substituting this section of code with a microprogrammed subroutine that is callable by the assembly or higher level program, overall execution time may often be reduced.

Once the microprogrammer has determined what segment to implement in microcode, the microprogram is developed as shown in Figure 5-1. The Micro Assembler program (in main memory) is used to assemble the source microprogram into an object program. Then, the object microprogram is loaded into Writable Control Store (WCS) with the aid of the Micro Debug Editor program. Interactive debugging may be performed with the aid of the Debug Editor while the object microprogram resides in WCS.

When the microprogram is fully checked out, the user may choose to have his program reside permanently in programmable Read-Only Memory (pROM) or in WCS where it may be altered programmatically. Implementation in ROM is accomplished by programming the pROM's with a pROM writer and installing the programmed ROM's in the computer. The mask tapes shown in Figure 5-1 are required by the PROM writer and are generated by the software at the user's command. ROM-resident microprograms are permanent and do not have to be reloaded each time the computer is powered up; this implementation also prevents users from erroneously destroying the microprogram. The user who does not require such permanence for microprogram storage may execute his microcode from WCS. Microprograms used in this manner may be loaded with the WCS I/O utility routine and may be altered under program control to suit a variety of users. User-written microprograms are easily accessed by assembly or higher level programs. Once the microprogram is developed and loaded into control store, it may be called in a very similar manner to a software routine.

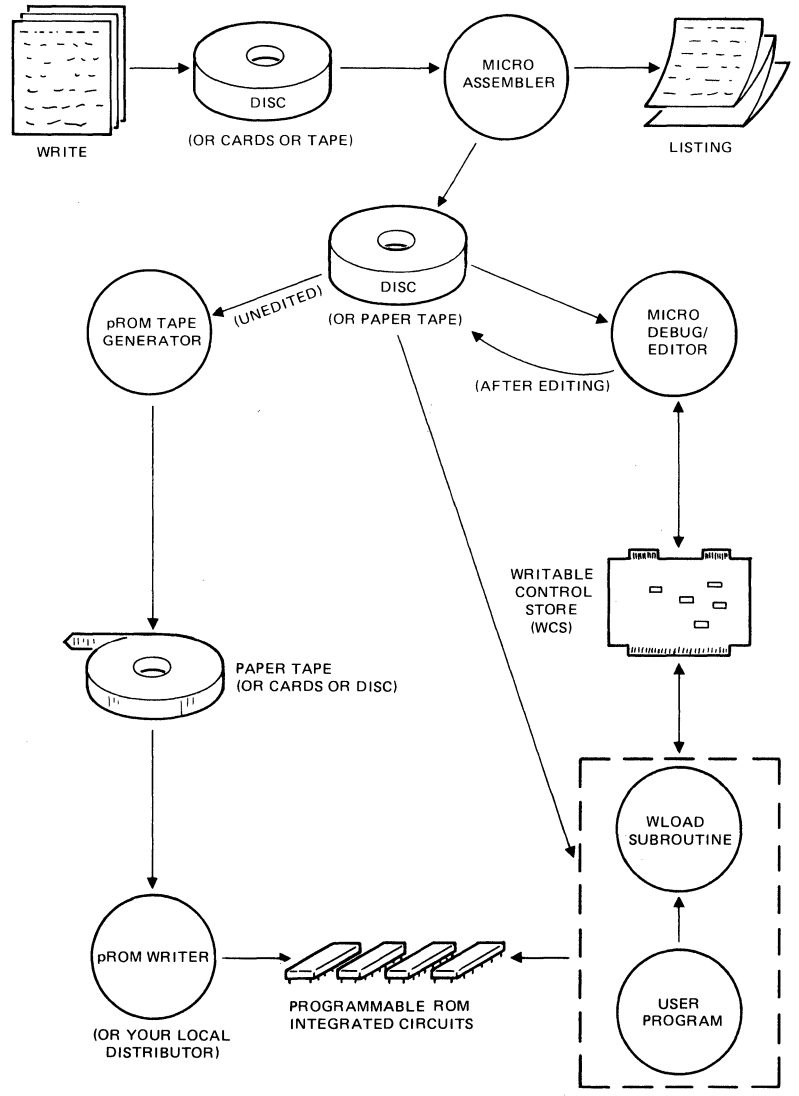
5-7. SUPPORT FOR THE MICROPROGRAMMER

Hewlett-Packard provides a comprehensive set of hardware and software manuals, and training courses to make user microprogramming easy to learn and implement. For permanent implementation of microprograms, programmed PROMs may be installed in the HP 12945A User ROM Control Store Board (M-Series), the HP 13304 Firmware Accessory Board (E/F-Series) or the HP 12791A Firmware Expansion Module (M/E/F-Series). The User ROM Control Store Board and the Firmware Accessory Board mount under the main CPU board of the M-Series and E/F-Series computers, respectively, and are used to house Hewlett-Packard standard or optional instruction sets such as the Dynamic Mapping System and the D/S 1000 Firmware instructions. Up to 8k 24-bit words of control store may be installed in the optional Firmware Expansion Module.

The HP 13197A 1k 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. Each WCS module consists of a single card which plugs into the I/O PCA cage, thus eliminating the need for extensive cabling or an additional power supply. A WCS card contains 1,024 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 processor.

Available microprogramming software includes the Micro Assembler and Micro Debug Editor as well as diagnostics, driver program, and I/O utility routine for use with the Writable Control Store module. These software aids operate under the Hewlett-Packard Real Time Executive (RTE) operating system.

A course is offered at HP facilities in Cupertino, California for customer training. Requiring only a knowledge of F-Series assembly language as a prerequisite, the course features in-depth coverage of microprogram development and implementation, and provides hands-on experience for the microprogrammer. The F-Series microprogrammer may also take advantage of other user-written microprograms via the HP Contributed Library, which contains many tested and documented microprograms.



NOTES:
1. MICROASSEMBLER OBJECT CODE MAY BE OUTPUT DIRECTLY TO DISC FILE AS WELL AS TO PAPER TAPE.
2. pROM TAPE GENERATOR (PTGEN) ACCEPTS ONLY UNEDITED OBJECT CODE.
3. OBJECT CODE MAY BE LOADED INTO WCS FROM DISC OR PAPER TAPE VIA WLOAD (WCS LIBRARY ROUTINE)
4. ALTERNATIVELY, MICROCODE MAY BE TRANSFERED DIRECTLY BETWEEN THE USER'S PROGRAM AND WCS WITH EXEC CALLS TO THE WCS DRIVERS.

7114-2

Figure 5-1. Microprogram Development Cycle

5-8. OPTIONAL INSTRUCTION SETS

5-9. DYNAMIC MAPPING SYSTEM

The Dynamic Mapping System (DMS) which is standard for the HP 2117F and optional for other HP 1000 Computers, gives the user the capability to address physical memory configurations larger than the standard 32,768 word limitation. The DMS provides a 20-bit-wide memory address bus which allows an addressing space of 1,048,576 words of main memory and allows the user to specify each 1,024-word page within physical memory to be read and/or write protected for program security. Separate memory translation maps provide isolation of system, user, DCPC channel 1, and DCPC channel 2.

The DMS consists of a Memory Expansion Module (MEM) and a Memory Protect PCA which plug into the memory PCA cage; microcode for implemewnting the additional 38 machine language instructions associated with the DMS is mounted in the Firmware Accessory Board.

5-10. FPP MICROPROGRAMMING (F-SERIES ONLY)

Information on directly microprogramming the Floating Point Processor (FPP) is given in the *HP E/F-Series Computer Microprogramming Reference Manual,* part no. 02109-90004.

5-11. CONCLUSION

Microprogramming is a very powerful tool that gives the user many advantages in terms of speed, flexibility, and program security. Experience has shown that microprogramming once learned, is in many cases much more flexible while being just as simple in concept as assembly language programming. Microprogramming does have its limitations however, and the potential user should examine very closely the extent of support provided by the computer manufacturer. Hewlett-Packard has by far sold and supported the greatest number of microprogrammable computers in the world, and provides world-wide customer support. Customer training courses and documentation have been refined from years of customer-contributed feedback and actual implementation is made easy through extensive software support packages and inexpensive hardware tools.

INTERRUPT SYSTEM

SECTION VI

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. Table 6-1 and 6-2 list the interrupt levels in priority order for the family of HP 1000 Computers.

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 VII.

Table 6-1. HP 2108M, HP 2109E and HP 2111F Interrupt Assignments

CHANNEL (Octal)	INTERRUPT LOCATION	ASSIGNMENT
04	00004	Power Fail Interrupt
05	00005	Memory Parity/Memory Protect/ DMS 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 6-2. HP 2112M, HP 2113E and HP 2117F Interrupt Assignments

CHANNEL (Octal)	INTERRUPT LOCATION	ASSIGNMENT
04	00004	Power Fail Interrupt
05	00005	Memory Parity/Memory Protect/ DMS Interrupt
06	00006	DCPC Channel 1 Completion Interrupt
07	00007	DCPC Channel 2 Completion Interrupt
10	00010	I/O Device (highest priority)
11 - 25	00011-00025	I/O Device (Mainframe)
26 - 47	00026-00047	I/O Device (Extender No. 1)
50 - 71	00050-00071	I/O Device (Extender No. 2)

6-1. POWER-FAIL INTERRUPT

The computer is equipped with power-sensing circuits. When primary line 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 6-3. The optional battery will supply enough power to preserve the contents of memory for a sustained line power outage of up to 2 hours.

Table 6-3. 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	4B	Shutdown
UP	LDA	SAVR	Was computer running
	SZA,RSS		when power failed?
	HLT	4B	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
	JMP	SAVP,I	Transfer control to program in execution at time of power failure
FENCE	OCT	2000	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

If the Dynamic Mapping System (DMS) is installed and a power failure occurs, the System Map is automatically enabled just prior to fetching the instruction in location 00004. Since all maps are disabled and none are considered valid upon the restoration of power, the power fail subroutine should include the necessary instructions to save as many maps as desired and restore them prior to enabling the DMS.

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 6-3.) 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 interrupt. Alternatively, if location 00004 contains a HLT instruction instead of a JSB instruction, the computer will halt and light the POWER FAIL 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 1000. 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. Pressing and holding the PRESET switch will force a halt when the LOCK/OPERATE switch is set to OPERATE.

The optional battery sustains the contents of memory when the line power is off. If the battery becomes discharged when the line power is off, the contents of memory will be lost. When power is restored, the computer will initiate a "Cold Start-up" and clear memory.

6-2. 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 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.

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 contain a JSB instruction referencing the entry point of a user-written memory protect subroutine, or alternatively 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.

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.

DST	ISZ	JLY	JMP	JPY	JSB
MVB	MVW	SAX	SAY	SBX	SBY
STA	STB	STX	STY		

6-3. MEMORY PROTECT/DMS INTERRUPT

The memory protect option provides the capability of protecting a selected block of memory of any size, from a settable fence address downward, against alteration or entry by programmed instructions.

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, an executive program residing in protected memory 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.

The optional DMS hardware includes additional memory protect features, which are enabled or disabled simultaneous with the memory protect hardware. When enabled by an STC 05 instruction, the DMS hardware provides the capability of read/write protecting memory on a 1024-word page basis. Included in the DMS are several privileged instructions which are not allowed when the memory protect logic is enabled. Upon detection of a violation, an interrupt to location 00005 is generated. Since the DMS will set the flag on channel 05, executing either an SFS 05 or an SFC 05 instruction will permit the programmer to know whether the DMS or memory protect interrupted.

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

1. The upper protected memory boundary address is loaded into the fence register from the A- or B-register by an OTA 05 or OTB 05 instruction, respectively. Memory addresses below but not including this addresses are protected.

2. Execution will be inhibited and an interrupt to location 00005 will occur if one of the following instructions either directly or indirectly modifies or enters a location in protected memory, or if any I/O instruction is attempted (including HLT but excluding those I/O instructions addressing select code 01).

3. Location 00002 is normally the lower boundary of protected memory. (Locations 00000 and 00001 are the A- and B-register addresses and may be freely addressed.) JMP, JLY, and JPY instructions may not reference the A- or B-register.

4. After three successive levels of indirect addressing, the memory protect logic will allow a pending I/O interrupt if the memory protect logic is installed.

5. Any instructions not mentioned in Step 2 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 2 is legal provided that the ultimate effective address is outside the protected memory area.

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 logical address of the error location.

Table 6-4 illustrates a sample memory protect, DMS, 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.

6-4. 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

Table 6-4. Sample Memory Protect, Parity Error, and DMS Subroutine

LABEL	OPERCODE	OPERAND	COMMENTS
MPEDM	NOP		Memory Protect/Parity Error/DMS Subroutine
	CLF	0B	Turn off interrupt system
	STA	SAVA	Save A-register contents
	STB	SAVB	Save B-register contents
	LIA	5B	Get contents of violation register
	CLF	5B	Turn off parity error interrupts
	SFC	5B	Check flag for DMS violation
	JMP	DMS	If flag is set, then DMS interrupted
	SSA		Check bit 15 of violation register
	JMP	PE	If bit 15 is set, then parity error occurred
	JMP	MP	If bit 15 is clear, then memory protect interrupted
MP	—		User's routine for memory protect violation
	—		
	—		
	etc.		
	—		
	—		
	JMP	REST	
PE	—		User's routine for parity error condition
	—		
	—		
	etc.		
	—		
	—		
	JMP	REST	
DMS	—		User's routine for DMS violation
	—		
	—		
	etc.		
	—		
	—		
	JMP	REST	
REST	LDA	SAVA	Restore A-register
	LDB	SAVB	Restore B-register
	STF	0B	Enable interrupt system
	STF	5B	Enable parity error interrupt
	STC	5B	Turn on memory protect
	JMP	MPEDM,I	Exit
SAVA	OCT	0	Storage for A
SAVB	OCT	0	Storage for B

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.

6-5. 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:

1. Interrupt system disabled or interface PCA interrupt disabled.

2. 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. After three successive levels of indirect addressing, the memory protect logic will allow a pending I/O interrupt if the memory protect logic is installed.

3. 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.

4. Optional dual-channel port controller in the process of transferring data.

5. Current instruction is one that may affect the priorities of I/O devices; e.g., STC, CLC, STF, CLF, SFS, and SFC. The interrupt in this case must wait until the succeeding instruction is executed. The SFS instruction used with the interrupt system on produces special conditions. Since the SFS instruction holds off interrupts until the next instruction is executed, if the next instruction clears the device flag, then it should also remove the interrupt request. Therefore, a CLF instruction should be used rather than appending, C (sets bit 9 in some I/O instructions) to the instruction.

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.

6-6. 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. As described in Section II, the central interrupt register contents can be accessed from the operator panel.

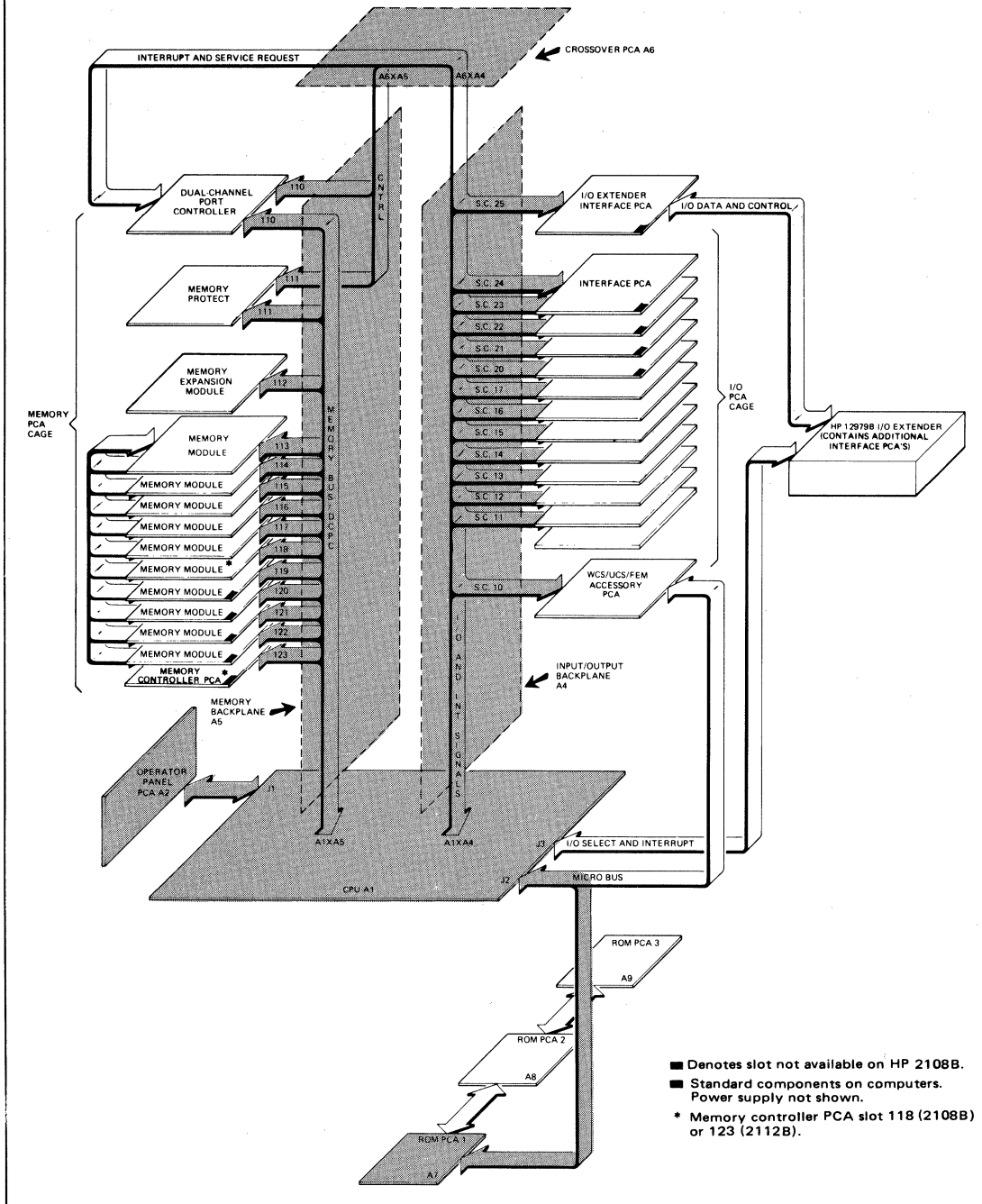
6-7. INTERRUPT SYSTEM CONTROL

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, the interrupt system is disabled.

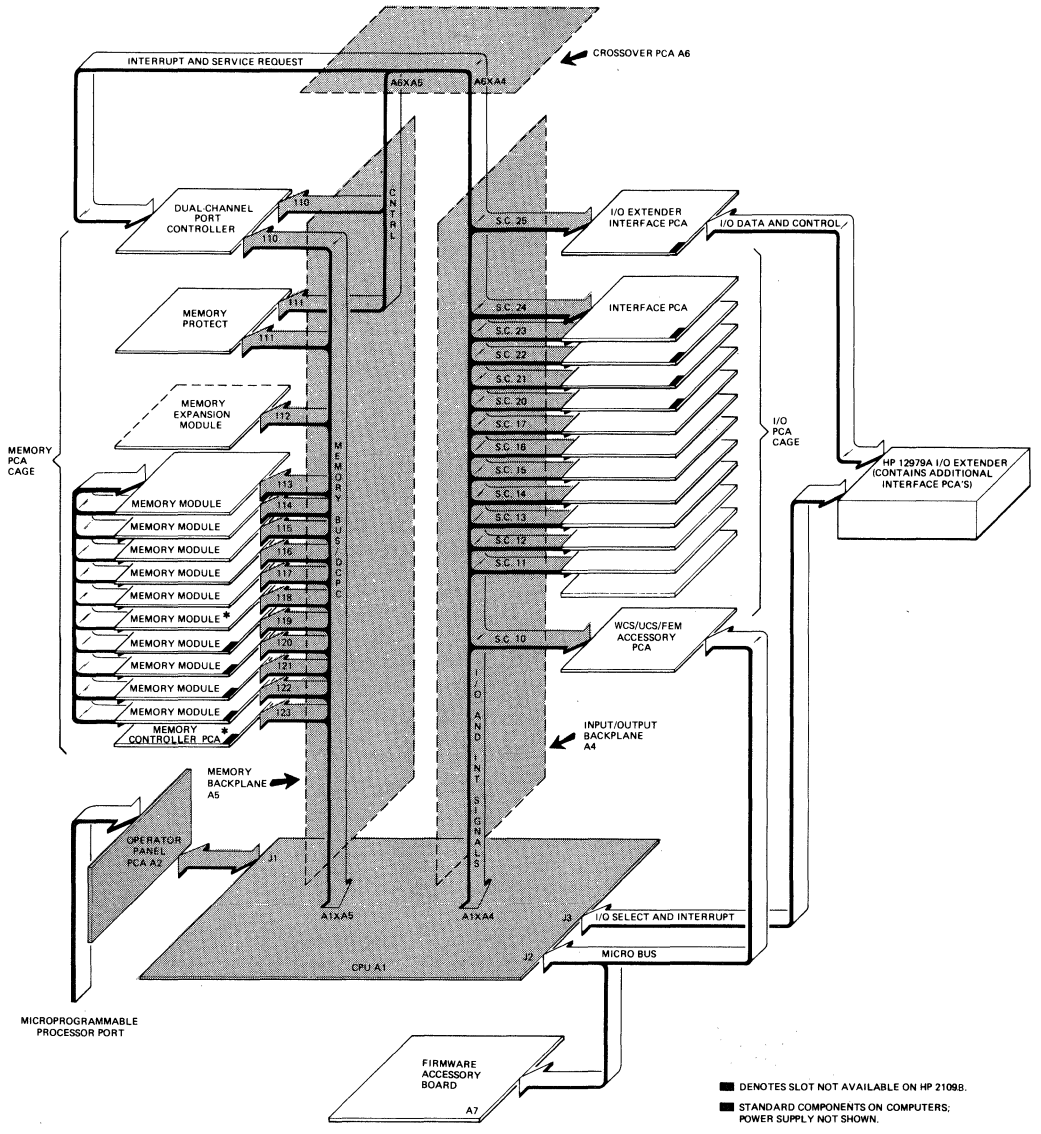
APPENDIX

A

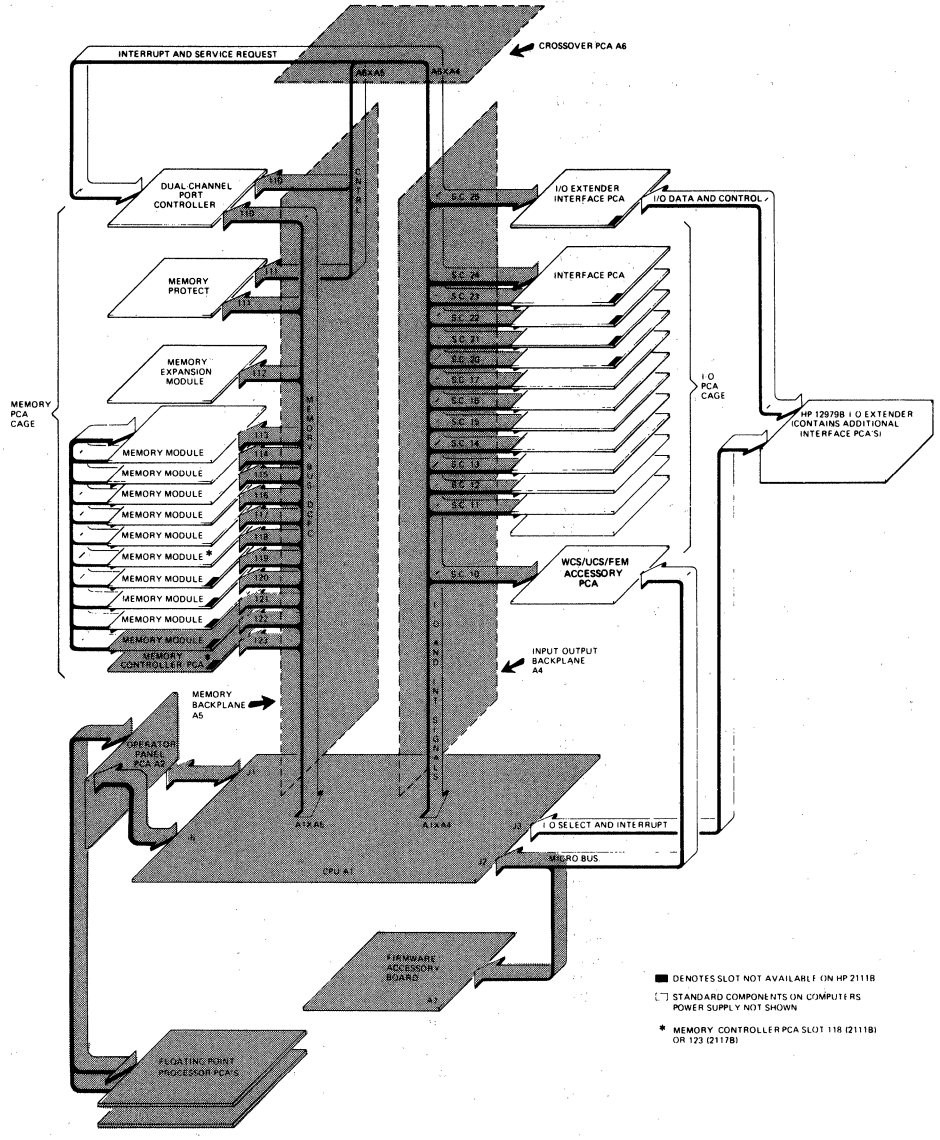
M-SERIES COMPUTER PHYSICAL LAYOUT



E-SERIES COMPUTER PHYSICAL LAYOUT



E-SERIES COMPUTER PHYSICAL LAYOUT

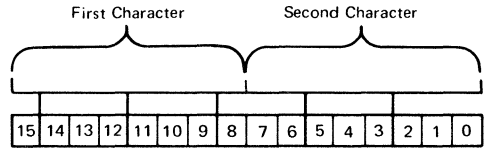


7700-202

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:

```
  177777
- 000556
  177221
     + 1
  177222
```

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.

$400_8 = 256_{10}$

$60_8 = 48_{10}$

$3_8 = 3_{10}$

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.

$5000_{10} = 11610_8$

$200_{10} = 310_8$

$20_{10} = 24_8$

$9_{10} = 11_8$

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	177634
-200	177470
-500	177014
-1000	176030
-2000	174060
-5000	166170
-10000	154360
-20000	130740
-32768	100000

EXAMPLE

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

$-500_{10} = 177014_8$

$-100_{10} = 177634_8$

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

$-9_{10} = 177767_8$

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
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.0625
32	5	0.03125
64	6	0.01562 5
128	7	0.00781 25
256	8	0.00390 625
512	9	0.00195 3125
1 024	10	0.00097 65625
2 048	11	0.00048 82812 5
4 096	12	0.00024 41406 25
8 192	13	0.00012 20703 125
16 384	14	0.00006 10351 5625
32 768	15	0.00003 05175 78125
65 536	16	0.00001 52587 89062 5
131 072	17	0.00000 76293 94531 25
262 144	18	0.00000 38146 97265 625
524 288	19	0.00000 19073 48632 8125
1 048 576	20	0.00000 09536 74316 40625
2 097 152	21	0.00000 04768 37158 20312 5
4 194 304	22	0.00000 02384 18579 10156 25
8 388 608	23	0.00000 01192 09289 55078 125
16 777 216	24	0.00000 00596 04644 77539 0625
33 554 432	25	0.00000 00298 02322 38769 53125
67 108 864	26	0.00000 00149 01161 19384 76562 5
134 217 728	27	0.00000 00074 50580 59692 38281 25
268 435 456	28	0.00000 00037 25290 29846 19140 625
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	1.000 000 000 000 000 000 00
12	1	0.063 146 314 631 463 146 31
144	2	0.005 075 341 217 270 243 66
1 750	3	0.000 406 111 564 570 651 77
23 420	4	0.000 032 155 613 530 704 15
303 240	5	0.000 002 476 132 610 706 64
3 641 100	6	0.000 000 206 157 364 055 37
46 113 200	7	0.000 000 015 327 745 152 75
575 360 400	8	0.000 000 001 257 143 561 06
7 346 545 000	9	0.000 000 000 104 560 276 41
112 402 762 000	10	0.000 000 000 006 676 337 66
1 351 035 564 000	11	0.000 000 000 000 537 657 77
16 432 451 210 000	12	0.000 000 000 000 043 136 32
221 411 634 520 000	13	0.000 000 000 000 003 411 35
2 657 142 036 440 000	14	0.000 000 000 000 000 264 11
34 327 724 461 500 000	15	0.000 000 000 000 000 022 01
434 157 115 760 200 000	16	0.000 000 000 000 000 001 63
5 432 127 413 542 400 000	17	0.000 000 000 000 000 000 14
67 405 553 164 731 000 000	18	0.000 000 000 000 000 000 01

MATHEMATICAL EQUIVALENTS

2^x IN DECIMAL

x	2^x	x	2^x	x	2^x
0.001	1.00069 33874 62581	0.01	1.00695 55500 56719	0.1	1.07177 34625 36293
0.002	1.00138 72557 11335	0.02	1.01395 94797 90029	0.2	1.14869 83549 97035
0.003	1.00208 16050 79633	0.03	1.02101 21257 07193	0.3	1.23114 44133 44916
0.004	1.00277 64359 01078	0.04	1.02811 38266 56067	0.4	1.31950 79107 72894
0.005	1.00347 17485 09503	0.05	1.03526 49238 41377	0.5	1.41421 35623 73095
0.006	1.00416 75432 38973	0.06	1.04246 57608 41121	0.6	1.51571 65665 10398
0.007	1.00486 38204 23785	0.07	1.04971 66836 23067	0.7	1.62450 47927 12471
0.008	1.00556 05803 98468	0.08	1.05701 80405 61380	0.8	1.74110 11265 92248
0.009	1.00625 78234 97782	0.09	1.06437 01824 53360	0.9	1.86606 59830 73615

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

n	$n \log_{10} 2$	$n \log_2 10$	n	$n \log_{10} 2$	$n \log_2 10$
1	0.30102 99957	3.32192 80949	6	1.80617 99740	19.93156 85693
2	0.60205 99913	6.64385 61898	7	2.10720 99696	23.25349 66642
3	0.90308 99870	9.96578 42847	8	2.40823 99653	26.57542 47591
4	1.20411 99827	13.28771 23795	9	2.70926 99610	29.89735 28540
5	1.50514 99783	16.60964 04744	10	3.01029 99566	33.21928 09489

MATHEMATICAL CONSTANTS IN OCTAL SCALE

$\pi = (3.11037\ 552421)_{(8)}$	$e = (2.55760\ 521305)_{(8)}$	$\gamma = (0.44742\ 147707)_{(8)}$
$\pi^{-1} = (0.24276\ 301556)_{(8)}$	$e^{-1} = (0.27426\ 530661)_{(8)}$	$\ln \gamma = -(0.43127\ 233602)_{(8)}$
$\sqrt{\pi} = (1.61337\ 611067)_{(8)}$	$\sqrt{e} = (1.51411\ 230704)_{(8)}$	$\log_2 \gamma = -(0.62573\ 030645)_{(8)}$
$\ln \pi = (1.11206\ 404435)_{(8)}$	$\log_{10} e = (0.33626\ 754251)_{(8)}$	$\sqrt{2} = (1.32404\ 746320)_{(8)}$
$\log_2 \pi = (1.51544\ 163223)_{(8)}$	$\log_2 e = (1.34252\ 166245)_{(8)}$	$\ln 2 = (0.54271\ 027760)_{(8)}$
$\sqrt{10} = (3.12305\ 407267)_{(8)}$	$\log_2 10 = (3.24464\ 741136)_{(8)}$	$\ln 10 = (2.23273\ 067355)_{(8)}$

OCTAL COMBINING TABLES

MEMORY REFERENCE INSTRUCTIONS

Indirect Addressing

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 A or B.
2. select 1 to 4 instructions, 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 instructions, 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 (2004)	SZA (2002)	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

Memory Reference

ADA	04(0XX)---
ADB	04(1XX)---
AND	01(0XX)---
CPA	05(0XX)---
CPB	05(1XX)---
IOR	03(0XX)---
ISZ	03(1XX)---
JMP	02(1XX)---
JSB	01(1XX)---
LDA	06(0XX)---
LDB	06(1XX)---
STA	07(0XX)---
STB	07(1XX)---
XOR	02(0XX)---

↑ Binary

Shift-Rotate

ALF	001700
ALR	001400
ALS	001000
ARS	001100
BLF	005700
BLR	005400
BLS	005000
BRS	005100
CLE	000040
ELA	001600
ELB	005600
ERA	001500
ERB	005500
NOP	000000
RAL	001200
RAR	001300
RBL	005200
RBR	005300
SLA	000010
SLB	004010

Alter-Skip

CCA	003400
CCB	007400
CCE	002300
CLA	002400
CLB	006400
CLE	002100
CMA	003000
CMB	007000
CME	002200
INA	002004
INB	006004
RSS	002001
SEZ	002040
SLA	002010
SLB	006010
SSA	002020
SSB	006020
SZA	002002
SZB	006002

Input/Output

CLC	1067--
CLF	1031--
CLO	103101
HLT	1020--
LIA	1025--
LIB	1065--
MIA	1024--
MIB	1064--
OTA	1026--
OTB	1066--
SFC	1022--
SFS	1023--
SOC	102201
SOS	102301
STC	1027--
STF	1021--
STO	102101

Extended Arithmetic

ASL	1000(01X)-
ASR	1010(01X)-
DIV	100400
DLD	104200
DST	104400
LSL	1000(10X)-
LSR	1010(10X)-
MPY	100200
RRL	1001(00X)-
RRR	1011(00X)

↑ Binary

*Assuming: no indirect addressing
no combined instructions
shifts taken in first position only
hold flag after I/O execution

†Not directly user callable.
Used by HP software.

Refer to preceding page for octal combining tables

Ext. Inst. Group

ADX	105746
ADY	105756
CAX	101741
CAY	101751
CBS	105774
CBT	105766
CBX	105741
CBY	105751
CMW	105776
CXA	101744
CXB	105744
CYA	101754
CYB	105754
DSX	105761
DSY	105771
ISX	105760
ISY	105770
JLY	105762
JPY	105772
LAX	101742
LAY	101752
LBT	105763
LBX	105742
LBY	105752
LDX	105745
LDY	105755
MBT	105765
MVW	105777
SAX	101740
SAY	101750
SBS	105773
SBT	105764
SBX	105740
SBY	105750
SFB	105767
STX	105743
STY	105753
TBS	105775
XAX	101747
XAY	101757
XBX	105747
XBY	105757

Floating Point

FAD	105000
FDV	105060
FIX	105100
FLT	105120
FMP	105040
FSB	105020
.FIXD	105104
.FLTD	105124
.TADD	105002
.TDIV	105062
.TFTD	105126
.TFTS	105122
.TFXD	105106
.TFXS	105102
.TMPY	105042
.TSUB	105022
.XADD	105001
.XDIV	105061
.XFTD	105125
.XFTS	105121
.XFXD	105105
.XFXS	105101
.XMPY	105041
.XSUB	105021

Scientific Inst. Set

ALOG	105322
ALOGT	105327
ATAN	105323
COS	105324
EXP	105326
SIN	105325
SQRT	105321
TAN	105320
TANH	105330
DPOLY	105331
/CMRT	105332†
/ATLG	105333
.FPWR	105334
.TPWR	105335

Fast FORTRAN

DBLE	105201
DDINT	105217
SNGL	105202
.BLE	105207
.CFER	105231
.DFER	105205
.ENTP	105224
.ENTR	105223
.FLUN	105226
.GOTO	105221
.NGL	105214
.PACK	105230
.PWR2	105225
$SETP	105227
.XCOM	105215
.XFER	105220
.XPAK	105206
..DCM	105216
..FCM	105232
..MAP	105222
..TCM	105233
.XADD	105213
.XSUB	105214
.XMPY	105203
.XDIV	105204
XADD	105207
XSUB	105210
XMPY	105211
XDIV	105212

Double Integer

.DAD	105014
.DCO	105204
.DDE	105211
.DDI	105074
.DDIR	105134
.DDS	105213
.DIN	105210
.DIS	105212
.DMP	105054
.DNG	105203
.DSB	105034
.DSBR	105114

Dynamic Mapping System

DJP	105732
DJS	105733
JRS	105715
LFA	101727
LFB	105727
MBF	105703
MBI	105702
MBW	105704
MWF	105706
MWI	105705
MWW	105707
PAA	101712
PAB	105712
PBA	101713
PBB	105713
RSA	101730
RSB	105730
RVA	101731
RVB	105731
SJP	105734
SJS	105735
SSM	105714
SYA	101710
SYB	105710
UJP	105736
UJS	105737
USA	101711
USB	105711
XCA	101726
XCB	105726
XLA	101724
XLB	105724
XMA	101722
XMB	105722
XMM	105720
XMS	105721
XSA	101725
XSB	105725

VECTOR INSTRUCTION SET OPCODES IN OCTAL

Single Precision

	OPCODE	SUB OPCODE
VABS	101462	
VADD	101460	000000
VDIV	101460	000060
VDOT	101465	
VMAB	101467	
VMAX	101466	
VMIB	101471	
VMIN	101470	
VMOV	101472	
VMPY	101460	000040
VNRM	101464	
VPIV	101461	
VSAD	101460	000400
VSDV	101460	000460
VSMY	101460	000440
VSSB	101460	000420
VSUB	101460	000020
VSUM	101463	
VSWP	101473	

Double Precision

	OPCODE	SUB OPCODE
DVABS	105462	
DVADD	105460	004002
DVDIV	105460	004062
DVDOT	105465	
DVMAB	105467	
DVMAX	105466	
DVMIB	105471	
DVMIN	105470	
DVMOV	105472	
DVMPY	105460	004042
DVNRM	105464	
DVPIV	105461	
DVSAD	105460	004402
DVSDV	105460	004462
DVSMY	105460	004442
DVSSB	105460	004422
DVSUB	105460	004022
DVSUM	105463	
DVSWP	105473	

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			D/E		*RS	001	
				A/B	0	D/E	R*L	010			D/E		R*L	010	
				A/B	0	D/E	R*R	011			D/E		R*R	011	
				A/B	0	D/E	*LR	100			D/E		*LR	100	
				A/B	0	D/E	ER*	101			D/E		ER*	101	
				A/B	0	D/E	EL*	110			D/E		EL*	110	
				A/B	0	D/E	*LF	111			D/E		*LF	111	
				NOP	000			000			000			000	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	ASG	000		A/B	1	CL*	01	CLE	01	SEZ	SS*	SL*	IN*	SZ*	RSS
				A/B		CM*	10	CME	10						
				A/B		CC*	11	CCE	11						

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	number			
				LSL		000	000			1	0	← of →			
				RRR		001	001			0	0	bits			
				RRL		000	001			0	0				

Notes: * = A or B, according to bit 11.
D/I, A/B, Z/C, D/E, H/C coded: 0/1.
**Second word is Memory Address.

†CLE: Only this bit is required.
‡SL*: Only this bit and bit 11 (A/B as applicable) are required.

BASE SET INSTRUCTION CODES IN BINARY (CONT)

EXTENDED INSTRUCTION GROUP	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
													JLY = 0			
													JPY = 1			
BYTE INSTRUCTIONS	1	0	0	0	1	0	1	1	1	1	1	1	0			
													LBT = 0	1	1	
													SBT = 1	0	0	
													MBT = 1	0	1	
													CBT = 1	1	0	
													SFB = 1	1	1	
BIT INSTRUCTIONS	1	0	0	0	1	0	1	1	1	1	1	1	1			
													SBS = 0	1	1	
													CBS = 1	0	0	
													TBS = 1	0	1	
WORD INSTRUCTIONS	1	0	0	0	1	0	1	1	1	1	1	1	1	1	1	
																CMW = 0
																MVW = 1

BASE SET INSTRUCTION CODES IN BINARY (CONT)

15	14 13 12	11 10 9	8 7	6 5 4	3	2 1 0
1	FLTPT 000	101	00	FAD 000 FSB 001 FMP 010 FDV 011 FIX 100 FLT 101 .XADD (F) 000 .XSUB (F) 001 .XMPY (F) 010 .XDIV (F) 011 .XFXS 100 .XFTS 101 .TADD 000 .TSUB 001 .TMPY 010 .TDIV 011 .TFXS 100 .TFTS 101 .FIXD 100 .FLTD 101 .XFXD 100 .XFTD 101 .TFTD 101 .TFXD 100	0	000 001 010 100 101 110

SCIENTIFIC, FAST FORTRAN AND DOUBLE INTEGER INSTRUCTION CODES IN BINARY

15	14 13 12	11 10 9	8 7 6	5 4 3	2 1 0
	SCIENTIFIC INST. SET				
1	000	101	011	0 1 0	TAN 000 SQRT 001 ALOG 010 ATAN 011 COS 100 SIN 101 EXP 110 ALOGT 111 TANH 000 DPOLY 001 /CMRT 010 /ATLG 011 .FPWR 100 .TPWR 101
15	**14 13 12**	**11 10 9**	**8 7 6**	**5 4 3**	**2 1 0**
	FAST FORTRAN				
1	000	101	010	0 0 0	DBLE 001 SNGL 010 .XMPY (M/E) 011 .XDIV (M/E) 100 .DFER 101 .XPAK 110 .BLE (F) 111
				0 0 1	XADD (M/E) 111 XSUB (M/E) 000 XMPY (M/E) 001 XDIV (M/E) 010 .XADD (M/E) 011 .XSUB (M/E) 100 .NGL (F) 100 .XCOM 101 ..DCM 110 DDINT 111
				0 1 0	.XFER 000 .GOTO 001 ..MAP 010 .ENTR 011 .ENTP 100 .PWR2 101 .FLUN 110 $SETP 111
				0 1 1	.PACK 000 .CFER (E/F) 001 ..FCM 010 ..TCM 011
	DOUBLE INTEGER				
1	000	101	000	001	.DAD 100
				011	.DSB 100
				101	.DMP 100
				111	.DDI 100
			001	001	.DSBR 100
				011	.DDIR 100
			010	000	.DNG 011
				001	.DCO 100 .DIN 000 .DDE 001 .DIS 010 .DDS 011

VIS PRIMARY AND SUB OPCODES IN BINARY

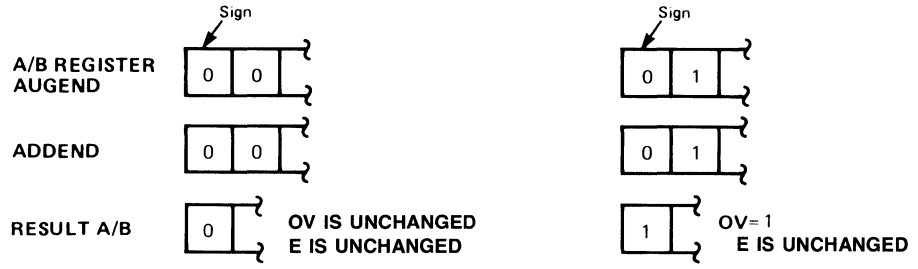
The opcode specifies the microcode entry point. The sub-opcode specifies the arithmetic function within a class of operations. The P-bit specifies the precision of the operation (P=0 for single and P=1 for double precision).

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VADD/DVADD	1	0	0	0	P	0	1	1	0	0	1	1	0	0	0	0
	0	0	0	0	P	0	0	0	0	0	0	0	0	0	P	0
VSUB/DVSUB	1	0	0	0	P	0	1	1	0	0	1	1	0	0	0	0
	0	0	0	0	P	0	0	0	0	0	0	1	0	0	P	0
VMPY/DVMPY	1	0	0	0	P	0	1	1	0	0	1	1	0	0	0	0
	0	0	0	0	P	0	0	0	0	0	1	0	0	0	P	0
VIDV/DVDIV	1	0	0	0	P	0	1	1	0	0	1	1	0	0	0	0
	0	0	0	0	P	0	0	0	0	0	1	1	0	0	P	0
VSAD/DVSAD	1	0	0	0	P	0	1	1	0	0	1	1	0	0	0	0
	0	0	0	0	P	0	0	1	0	0	0	0	0	0	P	0
VSSB/DVSSB	1	0	0	0	P	0	1	1	0	0	1	1	0	0	0	0
	0	0	0	0	P	0	0	1	0	0	0	1	0	0	P	0
VSMY/DVSMY	1	0	0	0	P	0	1	1	0	0	1	1	0	0	0	0
	0	0	0	0	P	0	0	1	0	0	1	0	0	0	P	0
VSDV/DVSDV	1	0	0	0	P	0	1	1	0	0	1	1	0	0	0	0
	0	0	0	0	P	0	0	1	0	0	1	1	0	0	P	0
VPIV/DVPIV	1	0	0	0	P	0	1	1	0	0	1	1	0	0	0	1
	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
VABS/DVABS	1	0	0	0	P	0	1	1	0	0	1	1	0	0	1	0
	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
VSUM/DVSUM	1	0	0	0	P	0	1	1	0	0	1	1	0	0	1	1
	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
VNRM/DVNRM	1	0	0	0	P	0	1	1	0	0	1	1	0	1	0	0
	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
VDOT/DVDOT	1	0	0	0	P	0	1	1	0	0	1	1	0	1	0	1
	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
VMAX/DVMAX	1	0	0	0	P	0	1	1	0	0	1	1	0	1	1	0
	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
VMAB/DVMAB	1	0	0	0	P	0	1	1	0	0	1	1	0	1	1	1
	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
VMIN/DVMIN	1	0	0	0	P	0	1	1	0	0	1	1	1	0	0	0
	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
VMIB/DVMIB	1	0	0	0	P	0	1	1	0	0	1	1	1	0	0	1
	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
VMOV/DVMOV	1	0	0	0	P	0	1	1	0	0	1	1	1	0	1	0
	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
VSWP/DVSWP	1	0	0	0	P	0	1	1	0	0	1	1	1	0	1	1
	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

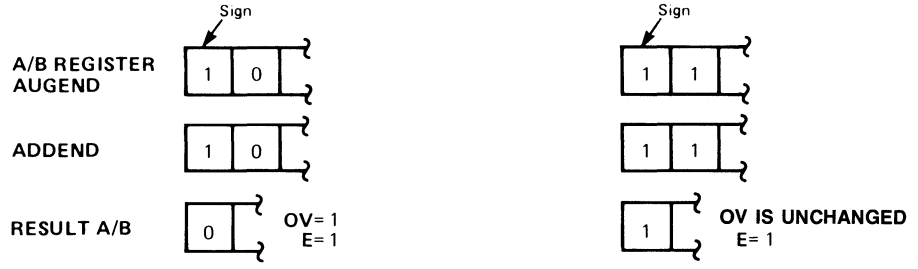
DYNAMIC MAPPING SYSTEM INSTRUCTION CODES IN BINARY

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DJP/DJS/UJP/UJS	1	0	0	0	1	0	1	1	1	1	0	1	1	D/U	1	P/S
JRS	1	0	0	0	1	0	1	1	1	1	0	0	1	1	0	1
LFA/LFB	1	0	0	0	A/B	0	1	1	1	1	0	1	0	1	1	1
MBI/MBF	1	0	0	0	1	0	1	1	1	1	0	0	0	0	1	I/F
MBW	1	0	0	0	1	0	1	1	1	1	0	0	0	1	0	0
MWF	1	0	0	0	1	0	1	1	1	1	0	0	0	1	1	0
MWI/MWW	1	0	0	0	1	0	1	1	1	1	0	0	0	1	I/W	1
PAA/PAB	1	0	0	0	A/B	0	1	1	1	1	0	0	1	0	1	0
PBA/PBB	1	0	0	0	A/B	0	1	1	1	1	0	0	1	0	1	1
RSA/RSB/RVA/RVB	1	0	0	0	A/B	0	1	1	1	1	0	1	1	0	0	S/V
SJP/SJS	1	0	0	0	1	0	1	1	1	1	0	1	1	1	0	P/S
SSM	1	0	0	0	1	0	1	1	1	1	0	0	1	1	0	0
SYA/SYB	1	0	0	0	A/B	0	1	1	1	1	0	0	1	0	0	0
USA/USB	1	0	0	0	A/B	0	1	1	1	1	0	0	1	0	0	1
XCA/XCB/XLA/XLB	1	0	0	0	A/B	0	1	1	1	1	0	1	0	1	L/C	0
XMA/XMB	1	0	0	0	A/B	0	1	1	1	1	0	1	0	0	1	0
XMM/XMS	1	0	0	0	1	0	1	1	1	1	0	1	0	0	0	M/S
XSA/XSB	1	0	0	0	A/B	0	1	1	1	1	0	0	1	1	0	1

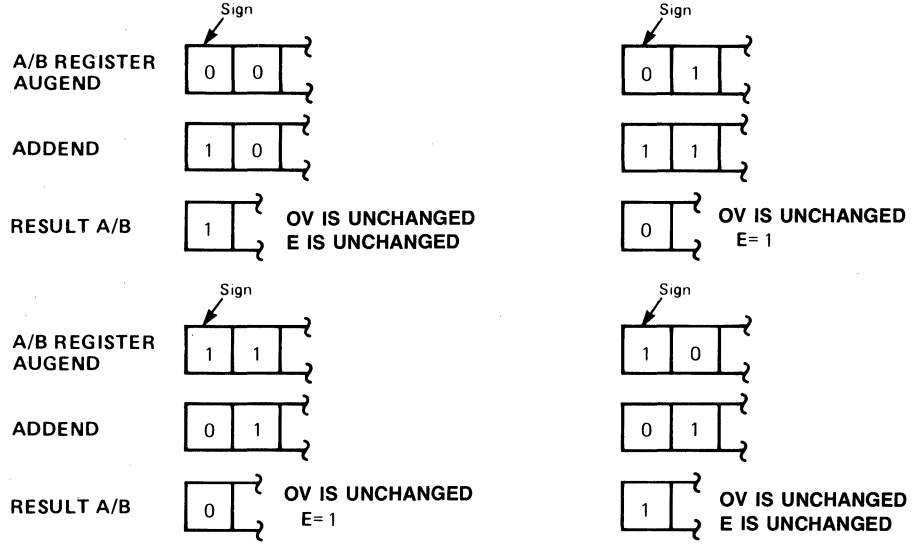
EXTEND AND OVERFLOW EXAMPLES



SAME SIGN (POSITIVE)



SAME SIGN (NEGATIVE)



DIFFERENT SIGNS

INTERRUPT AND I/O CONTROL SUMMARY

INST	S.C. 00	S.C. 01	S.C. 02	S.C. 03
STC	NOP	NOP	Prepares DCPC channel 1 to receive and store the block length in 2's complement form.	Prepares DCPC channel 2 to receive and store the block length in 2's complement form.
CLC	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	Loads A/B register with all zeros. (Equivalent to CLA/B instruction.)	Loads display 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	Equivalent to a NOP.	Merges display 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.	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).	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 and restores interrupt capability to lower priority functions.	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.
Flag FF sets automatically when power comes up. (No program control possible.)	Turns on parity error interrupt capability.	Aborts DCPC channel 1 data transfer.	Aborts DCPC channel 2 data transfer.	Sets PCA Flag FF.
Flag FF clears automatically when power fail occurs. (No program control possible.)	Turns off parity error interrupt capability and clears violation register bit 15.	Clears Flag FF on DCPC channel 1.	Clears Flag FF on DCPC channel 2.	Clears PCA Flag FF.
NOP	Skip if Dynamic Mapping System (DMS) interrupt.	Tests if DCPC channel 1 data transfer is complete.	Skip if DCPC channel 2 data transfer is complete.	Skip if I/O channel Flag FF is set.
Skip if power fail has occurred.	Skip if memory protect interrupt.	Tests if DCPC channel 1 data transfer is still in progress.	Skip if DCPC channel 2 data transfer is still in progress.	Skip if I/O channel Flag FF is clear.
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	Loads A/B register with all ones. (Equivalent to CCA/CCB instruction.)	Loads A/B register with all ones. (Equivalent to CCA/CCB instruction.)	Loads contents of PCA data 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.	Same as LIA/B 06 above.	Same as LIA/B 07 above.	Merges contents of PCA data buffer into A/B register.
NOP	Outputs first address of unprotected memory 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 data buffer.

PART II

HP 1000 M/E/F-SERIES COMPUTERS
I/O interfacing guide

CONTENTS

ILLUSTRATIONS

TABLES

INTRODUCTION

SECTION I

This manual is provided as an aid for design engineers and programmers to design and program special-purpose interfaces for the HP 1000 M-Series, E-Series and F-Series Computers. The content of this manual is presented as a supplement to your HP 1000 Series Computer Operating and Reference Manual. You should therefore have a thorough understanding of the applicable reference manual contents prior to reading this manual. It is also suggested you read this manual in its entirety and become completely acquainted with its contents before attempting to use the information presented in any one particular section.

Throughout this manual, the term HP 1000 is used to refer to all three models (M-, E- and F-Series Computers) when the information presented pertains equally to all three. A particular Series of HP 1000 Computer is referred to only if the information presented is unique to that Series.

1-1. INTRODUCTION TO INTERFACING

Interfacing a peripheral device with HP 1000 Computers involves both hardware and software. Except for the Microprogrammable Processor Port (MPP) interfacing discussed in Section VI, the hardware interface is accomplished by inserting one or more interface printed-circuit assemblies (PCA's) into easily accessible input/output (I/O) slots in the computer and connecting a cable between the interface PCA(s) and the peripheral device. As discussed in Section III, the computer provides a unique channel identification and service priority interrupt for every I/O channel used. Priority levels for the peripheral devices connected to the computer can be reassigned by simply changing the position of the interface PCA's in the computer I/O slots. (Specifications for I/O-type interface PCA's are discussed in Section V.) The software interface is accomplished by updating the existing computer I/O software system which may necessitate creating a new peripheral device driver.

1-2. HP 1000 M-SERIES, E-SERIES AND F-SERIES BASIC FEATURES AND DIFFERENCES

The HP 1000 M-, E- and F-Series Computers are microprogrammable, high-performance computers. The HP 1000 E-Series (HP 2109 and HP 2113) and F-Series (HP 2111 and HP 2117) Computers are enhanced versions of the HP 1000 M-Series (HP 2105, HP 2108 and HP 2112) Computers featuring faster system cycle and instruction execution times, faster I/O transfer rates and increased microprogram routine efficiency. The HP 1000 F-Series Computer combines the basic processing speed of the E-Series central processing unit with a hardware floating point processor, a Scientific Instruction Set and Fast FORTRAN Processor. Pertinent interfacing specifications for all three computers are presented in Table 1-1. (For more detailed specifications, refer to the applicable reference manuals listed in paragraph 1-5, Available Documentation). A discussion of control processor and I/O section timing for all three computers is contained in Section IV, Computer Timing.

The I/O systems for the HP 1000 M-, E- and F-Series Computers are generally compatible with each other. Wherever necessary, existing differences are discussed in this manual. The computer I/O system features a multilevel, vectored priority interrupt structure. There are 60 distinct interrupt levels, each of which has a unique priority assignment. Any I/O device can be selectively enabled or disabled under program control. The HP 2105 computer has four I/O channels in its mainframe; the HP 2108, HP 2109 and HP 2111 computers have nine; and the HP 2112, HP 2113 and HP 2117 have fourteen. The number of available I/O channels for the HP 1000 Computers can be increased by adding one or two HP 12979B I/O Extenders that provide 16 additional I/O channels each. All I/O channels are fully powered, buffered and bidirectional.

Data transfers between HP 1000 Computers and I/O devices can take place under program control or, for faster transfer rates, under Dual-Channel Port Controller (DCPC) control. The DCPC provides two direct links between computer memory and I/O devices and is program assignable to any two devices. DCPC data transfers occur on an I/O cycle-stealing basis and are independent of the I/O priority structure. For applications where even faster transfer rates are desirable, the HP 1000 E/F-Series Computers have special microprogrammed I/O capabilities that are discussed in Section VI, Advanced Interfacing Techniques.

1-3. USER INTERFACE REQUIREMENTS

This manual assumes that you wish to interface a device which is not a standard peripheral supplied by Hewlett-Packard along with its software I/O driver subroutine. Therefore, two objectives must be accomplished: some sort of general-purpose or special I/O interface PCA must be selected to plug into the computer and to accept the device

Table 1-1. HP 1000 Computers Interface Specifications

FEATURE	CAPABILITY
MAXIMUM MAINFRAME MEMORY SIZE IN BYTES: (Optional HP 12990A Memory Extender can be installed to add space and power for additional memory up to 1.152M bytes)	M-SERIES: HP 2105 To 64K; HP 2108 To 640K; HP 2112 To 1.28M E-SERIES: HP 2109 To 640K; HP 2113 To 1.28M F-SERIES: HP 2111 To 640K; HP 2117 To 1.28M
WORD SIZE:	16 bits

SYSTEM MEMORY CYCLE TIMES (nS):

	Cycle	Minimum	Typical	Maximum
(Standard Performance Memory and Controller, HP 2102B)	READ w/o DMS	560	595	665
	READ w/DMS	595	665	700
	WRITE	595	665	700
	REFRESH	595	665	700
(Standard Performance Fault Control Memory and Controller, HP 2102C)	READ w/o DMS	595	630	665
	READ w/DMS	630	700	730
	WRITE	595	630	665
	REFRESH	595	630	665
(High Performance Memory and Controller, HP 2102E, E- and F-Series Computers only)	READ w/o DMS	350	350	385
	READ w/DMS	385	420	455
	WRITE	350	350	385
	REFRESH	350	350	385
(High Performance Fault Control Memory and Controller, HP 2102H)	READ w/o DMS	386	421	456
	READ w/DMS	456	491	526
	WRITE	386	421	456
	REFRESH	386	421	456

INPUT/OUTPUT INSTRUCTION GROUP EXECUTION TIME (μS): (Depends on which I/O time period the instruction begins: T2, T3, T4, T5 or T6.)	M-SERIES:	2.59 To 3.89
	E/F-SERIES:	158 To 2.66
INTERRUPT LATENCY (μS):* (Non-DCPC environment)	M-SERIES:	85 (max.)
	E/F-SERIES:	45 (max.)

Table 1-1. HP 1000 Computers Interface Specifications (Continued)

FEATURE	CAPABILITY
DUAL-CHANNEL PORT CONTROLLER	
Number of Channels:	2
Word Size:	16 bits
Maximum Block Size:	32,768 words (65,536 bytes)
Transfer Rate (Mbytes/sec):	M-SERIES: (Maximum Input) 1.23 (Maximum Output) 1.23
	E/F-SERIES: (See following table; full bandwidth assumed, refer to Section 6.13)

	HP 2102B	Minimum	Typical	Maximum
(Standard Performance Memory)	Input w/DMS	1.884	1.950	2.098
	w/o DMS			
	Output w/DMS	1.626	1.676	1.782
	w/o DMS	1.672	1.778	1.938
	HP 2102C			
(Standard Performance Fault Control Memory)	Input w/DMS	1.946	1.018	2.096
	w/o DMS			
	Output w/DMS	1.586	1.626	1.726
	w/o DMS	1.670	1.724	1.780
	HP 2102E			
(High Performance Memory)	Input w/DMS	2.282	2.284	2.284
	w/o DMS			
	Output w/DMS	2.038	2.114	2.196
	w/o DMS	2.196	2.284	2.284
	HP 2102H			
(High Performance Fault Control Memory)	Input w/DMS	2.28	2.28	2.28
	w/o DMS			
	Output w/DMS	1.902	1.968	2.038
	w/o DMS	2.038	2.114	2.196

Table 1-1. HP 1000 Computers Interface Specifications (Continued)

FEATURE	CAPABILITY

DCPC LATENCY (CHANNEL 1, μS):**

	Typical	Worst Case
M-SERIES: Input Latency	2.22	2.93
Output Latency	2.54	3.25

E-SERIES: (See following table)

HP 2102B	Minimum	Typical	Maximum
(Standard Performance Memory)			
Input w/o DMS	2.73	2.91	3.15
w/DMS	2.84	3.12	3.26
Output w/o DMS	3.43	3.64	3.95
w/DMS	3.57	3.92	4.10
HP 2102C			
(Standard Performance Fault Control Memory)			
Input w/o DMS	2.98	3.12	3.26
w/DMS	3.09	3.33	3.46
Output w/o DMS	3.75	3.93	4.11
w/DMS	3.89	4.21	4.36
HP 2102E			
(High Performance Memory)			
Input w/o DMS	2.21	2.21	2.28
w/DMS	2.24	2.28	2.35
Output w/o DMS	2.75	2.75	2.84
w/DMS	2.80	2.87	2.98
HP 2102H			
(High Performance Fault Control Memory)			
Input w/o DMS	2.31	2.42	2.52
w/DMS	2.38	2.56	2.74
Output w/o DMS	2.84	2.94	3.05
w/DMS	2.98	3.16	3.33

A USER MICROCODE sequence of seven consecutive reads may provide conditions to produce the absolute worst case latency time. The absolute worst case latency times are as follows:

Memory System	Input	Output
HP 2102B	4.095 us	4.935 us
HP 2102C	5.125 us	6.031 us
HP 2102E	3.050 us	3.681 us

Table 1-1. HP 1000 Computers Interface Specifications (Continued)

FEATURE	CAPABILITY
MICROPROGRAMMABLE BLOCK I/O TRANSFERS (HP 1000 E-Series and F-Series only)	
Input (256 words or less):	2.28 Mbytes/sec
Output (256 words or less):	3.18 Mbytes/sec
Burst (16 words or less):	11.4 Mbytes/sec
MICROPROGRAMMABLE PROCESSOR PORT (MPP) I/O TRANSFERS (HP 1000 E-Series only)	
Burst (16 words or less):	11.4 Mbytes/sec (maximum)
Continuous:	3.18 Mbytes/sec (maximum)

NOTES: * Interrupt latency is defined as the time interval between the generation of an Interrupt Request (IRQ) signal by an I/O device and entry into the service routine.
** DCPC latency is defined as the time interval between the generation of a Service Request (SRQ) signal by an I/O device through the initiation of a DCPC channel 1 cycle to the actual completion of the I/O data transfer to or from the I/O interface PCA.

interface cable, and I/O software must be configured so that the computer can control the device. There are several possible methods of accomplishing these objectives. For hardware, the methods range from using available HP general-purpose interface PCA's to designing and building special interface PCA's from the drawingboard level. For software, writing and short assembly-language subroutine may suffice or Real-Time Executive (RTE) system drivers may have to be written. For software development information, refer to the applicable software system documentation listed in Table 1-3.

1-4. LEVELS OF HARDWARE INTERFACING

For purposes of this manual, the approaches to interfacing break down into three levels: Level 1 — Using HP General-Purpose Interface PCA's, Level 2 — Party-Line I/O, and Level 3 — Fabricating Interface PCA's.

Level 1 assumes that the specifications of off-the-shelf HP general-purpose interface PCA's are satisfactory to operate your device. These interface PCA's cover a wide range of applicability: receiving or transmitting signals with characteristics suitable for microcircuits, transistors, or relays. Appendix A of this manual contains a condensed, general description of the general-purpose interface kits available from Hewlett-Packard as of this printing. A data sheet providing the features, specifications, and a list of product support documentation and software either supplied with or available for the applicable interface kit is available at your nearest Hewlett-Packard Sales and Service Office. (A list of Sales and Service Offices is provided at the back of this manual.) Economics in design and manufacture can frequently be achieved by using these general-purpose interfaces. If a large number of devices, or devices of a special type are required to be serviced by the computer, or if exceptionally fast transfer rates are desired, Level 2 or Level 3 may have to be considered.

Level 2 provides a party-line method of servicing a large number of I/O devices. The number of devices serviceable by party-line I/O is dependent of the addressing word format you choose. Assuming seven bits are used for command and status information, eight bits would be left to address 256 devices. (One bit must be reserved for indirect addressing.) This is a typical example, but the quantity limit can vary by factors of the powers of two (128, 256, 512, etc.). A detailed discussion of party-line I/O is contained in Section VI, Advanced Interfacing Techniques.

Level 3 is the most fundamental level; designing and building an interface that permits the computer to service special devices or, for the HP 1000 E-Series and F-Series Computers, an interface that permits the faster microprogrammed I/O capabilities. Hewlett-Packard can furnish a breadboard interface PCA with the Flag and Control logic required by the computer's I/O section to facilitate these procedures. (Refer to Sections V and VI for more information.)

Table 1-2. HP 1000 Computers Available Current

MAXIMUM CURRENT AVAILABLE FOR MEMORY, ACCESSORIES AND I/O CARDS						
(A Model Power Supply)						
SUPPLY VOLTAGE	2105A	2108A	2109A	2112A	2113A	
+5V	12.8A	24.8A	24.6A	38.2A	38.0A	
−2V	5.0A	4.5A	4.5A	9.5A	9.5A	
+12V	1.0A	1.5A	1.5A	3.0A	3.0A	
−12V	1.0A	1.5A	1.5A	3.0A	3.0A	
(B Model Power Supply)						
SUPPLY VOLTAGE	2108B	2109B	2112B	2113B	2111F	2117F
+5V	38.8A	38.8A	38.8A	38.8A	21.9A	28.8A
−2V	4.0A	4.0A	4.0A	4.0A	6.0A	6.0A
+12V	2.5A	2.5A	2.5A	2.5A	2.5A	2.5A
−12V	2.0A	2.0A	2.0A	2.0A	2.0A	2.0A

The HP 1000 E-Series Computer has provisions for two types of microprogrammed I/O data transfers: transfers via a block I/O interface PCA connected to the I/O section and transfers via an interface PCA connected to the Microprogrammable Processor Port (MPP). As an aid toward determining whether block I/O or MPP transfers are best for your particular application, the following features and limitations should be considered:

1. Generally, MPP transfers are easier to microprogram and provide faster data transfer rates.

2. Block I/O transfers are more difficult to microprogram because the microcode must be written to simulate I/O instructions and the data transfer rates are slower because the I/O control instructions must be synchronized to I/O Section timing.

3. The MPP is totally independent on the I/O Section and, therefore, does not require the use of an I/O Section connector slot or select code.

4. Since MPP transfers are affected through bus drivers and receivers, the MPP has the capability of driving cables up to six feet (1.83 metres) in length.

5. Block I/O transfers do require the use of an I/O Section connector slot and do require a select code. Therefore, block I/O transfers can be used to combine the speed of microprogrammed I/O transfers with the capabilities of the interrupt system discussed in Section III, I/O System Fundamentals.

6. The MPP has no interrupt capability. Therefore, the computer must determine when the I/O device requires service by polling the device in the microprogram and then initiating the required data transfer. (Refer to Section IV, Computer Timing for more information.)

The HP 1000 F-Series Computer also has provisions for microprogrammed I/O data transfers. Because the MPP is dedicated to the floating point processor in the series of computers, the block I/O transfer method is the only one available to you for microprogrammed I/O applications.

At all levels, the user should keep in mind that Hewlett-Packard warranties and responsibilities apply only to those items produced and quality controlled by Hewlett-Packard. This manual is intended as a guide only, and the effectiveness of devices or programs created according to the recommendations outlined herein are purely the responsibility of the user.

1-5. AVAILABLE DOCUMENTATION

Supporting hardware documentation is provided with each Hewlett-Packard computer shipped to a customer. Hardware documentation is also supplied for optional and accessory add-ons as well as for off-the-shelf I/O interface PCA's. Basic hardware manuals for the HP 1000 Computers are listed in Table 1-3. Hardware manuals are also available for the I/O interface PCA's described in Appendix A of this manual. Consult your local Hewlett-Packard Sales and Service Office for additional hardware documentation related to the HP 1000 Computers. If your computer was supplied as part of an HP computer system,

Table 1-3. I/O Interface Related Reference Manuals

TITLE*	HP PART NUMBER
HARDWARE	
HP 1000 M-Series Computer Operating and Reference Manual	02108-90037
HP 1000 M-Series Computer Installation and Service Manual	02108-90035
HP 1000 E-Series Computer Operating and Reference Manual	02109-90014
HP 1000 E-Series Computer Installation and Service Manual	02109-90015
HP 1000 F-Series Computer Operating and Reference Manual	02109-90001
HP 1000 F-Series Computer Installation and Service Manual	02109-90002
MICROPROGRAMMING	
HP 1000 M-Series Computer RTE Microprogramming Reference Manual	02108-90032
HP 1000 E-Series and F-Series Computer Microprogramming Reference Manual	02109-90004
SOFTWARE	
Real-Time Executive Operating System Drivers and Device Subroutines	92200-93005
RTE-II Programming and Operating Manual	92001-93001
RTE-IV Programmer's Reference Manual	92067-90001
RTE-IVB Programmer's Reference Manual	92068-90004
RTE-M Programmer's Reference Manual	92064-90002
Driver Writing Manual	92200-93005

* For the purposes of this manual, HP 1000 is synonymous with 21MX.

a complete list of related hardware documentation is contained in the *Manual and Software Record* supplied with the system.

All software supplied with any HP computer system is supported by complete user documentation. General types of software manuals include language manuals, operating system manuals, software operating procedures, user manuals, applications manuals, and small program manuals. The *Manual and Software Record* supplied with each system lists all software furnished with the original equipment and provides an index to the software documentation. Software and software documentation supplied with standard HP I/O interface PCA's are listed in the individual data sheets. Reference manuals that contain basic information for writing system software drivers are listed in Table 1-3. Consult your local Hewlett-Packard Sales and Service Office for additional software documentation related to the HP 1000 Computers.

A complete list of microprogramming manuals available for the HP 1000 Computers is contained in Table 1-3.

COMPUTER CHARACTERISTICS

SECTION II

As an aid toward more successful I/O interface design, this section contains a general discussion of the HP 1000 Computer's operation and architecture. Unless otherwise specified, the contents of this section apply equally to the HP 1000 M-Series, HP 1000 E-Series and HP 1000 F-Series Computers.

2-1. COMPUTER OVERVIEW

As shown in Figure 2-1, the computer functionally consists of four major sections: a Control Processor Section, Main Memory Section, I/O Section, and Arithmetic/Logic Section. These four sections and the computer's Operator Panel are interconnected by a network of signal paths. Data processing programs and data are stored in the Main Memory Section. Parameters, status, commands, and computer results (data) are exchanged with external peripherals via the I/O Section. Mathematical functions such as add, subtract, and multiply and logical functions such as "and", "or", and shift are performed by the Arithmetic/Logic Section. The Operator Panel registers and switches provide direct operator communication. Each section operates under direction of the Control Processor Section by means of a microprogram. The Control Processor Section interprets the user's program stored in the Main Memory Section and directs the appropriate hardware in each of the other sections to perform the required operations. Control commands (or microinstructions) spell out which signal paths the data is to follow and what modifications or tests are to be performed.

Control and data paths between the computer's major sections and add-on accessories are provided by a bus system. The structure of the bus system is shown in Figure 2-2 which illustrates the main communication paths between major computer sections and accessories. The S-bus is a 16-bit, tri-state, TTL-compatible bus and is the major data transfer bus in the computer. The T-bus is a 16-bit, bi-state, TTL-compatible bus. The T-bus is a resultant data bus and is completely internal to the Arithmetic/Logic Section. The M-bus is a 16-bit, tri-state, TTL-compatible bus. The M-bus holds the address to be referenced by memory and is driven by CPU M-register or the DCPC memory address registers. The ME-bus is a 10-bit, tri-state, TTL-compatible bus. The ME-bus holds the upper-ten bits of the 20-bit expanded memory address bus and is driven by the Memory Expansion Module. For interfacing, the select code (SC) bus, interrupt address (IA) bus, and

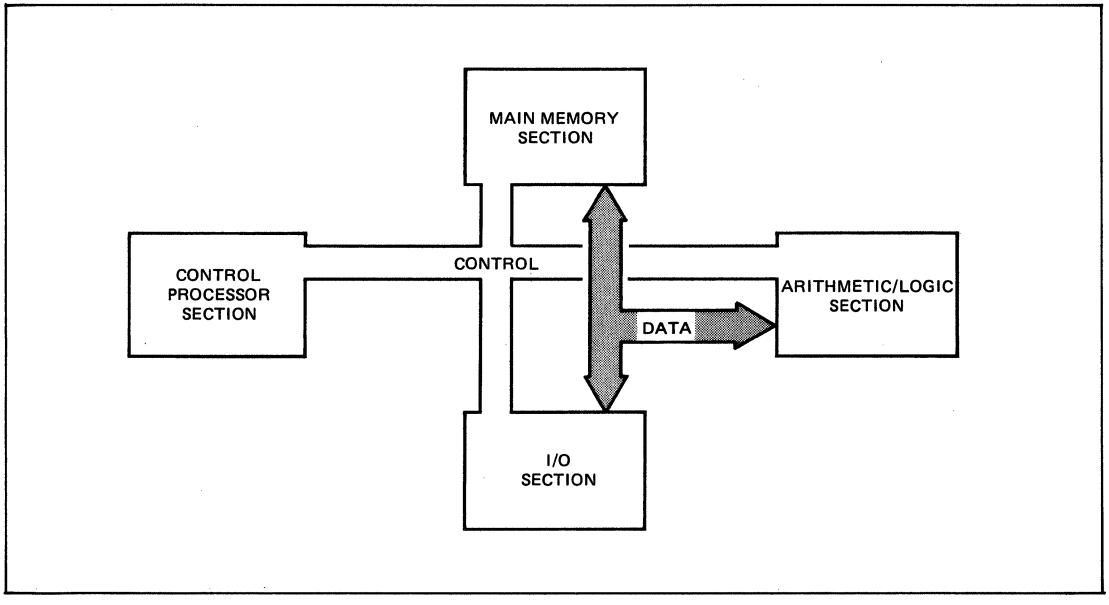


Figure 2-1. Computer Functional Sections

I/O bus are of prime importance. The select code bus is a 6-bit, CTL-compatible, control bus. The select code bus holds the select code for the I/O device being referenced by either the I/O Section or DCPC. The interrupt address bus is a 6-bit, open-collector, TTL-compatible control bus. The interrupt address bus holds the select code of any interrupt-requesting I/O interface PCA, memory protect or DCPC option. The I/O bus is a 16-bit, bi-directional, CTL-compatible, data communication bus for the I/O Section. All plug-in I/O interface PCA's transmit and receive data via the I/O bus. A more detailed discussion of the three I/O-related buses is contained in Sections III and IV of this manual. Block diagrams of the HP 1000 Computers is contained in Figure 2-3 at the end of this section.

2-2. CONTROL PROCESSOR SECTION

The Control Processor Section is the heart of the computer and contains the registers, control logic, control memory, and timing logic required to execute microprograms and fetch and execute programs stored in the Main Memory Section. This section initializes and controls, either directly or indirectly, the other computer sections. The primary tasks of the Control Processor Section are as follows:

a. Control the execution sequence of computer microprograms.

b. Decode microinstruction fields.

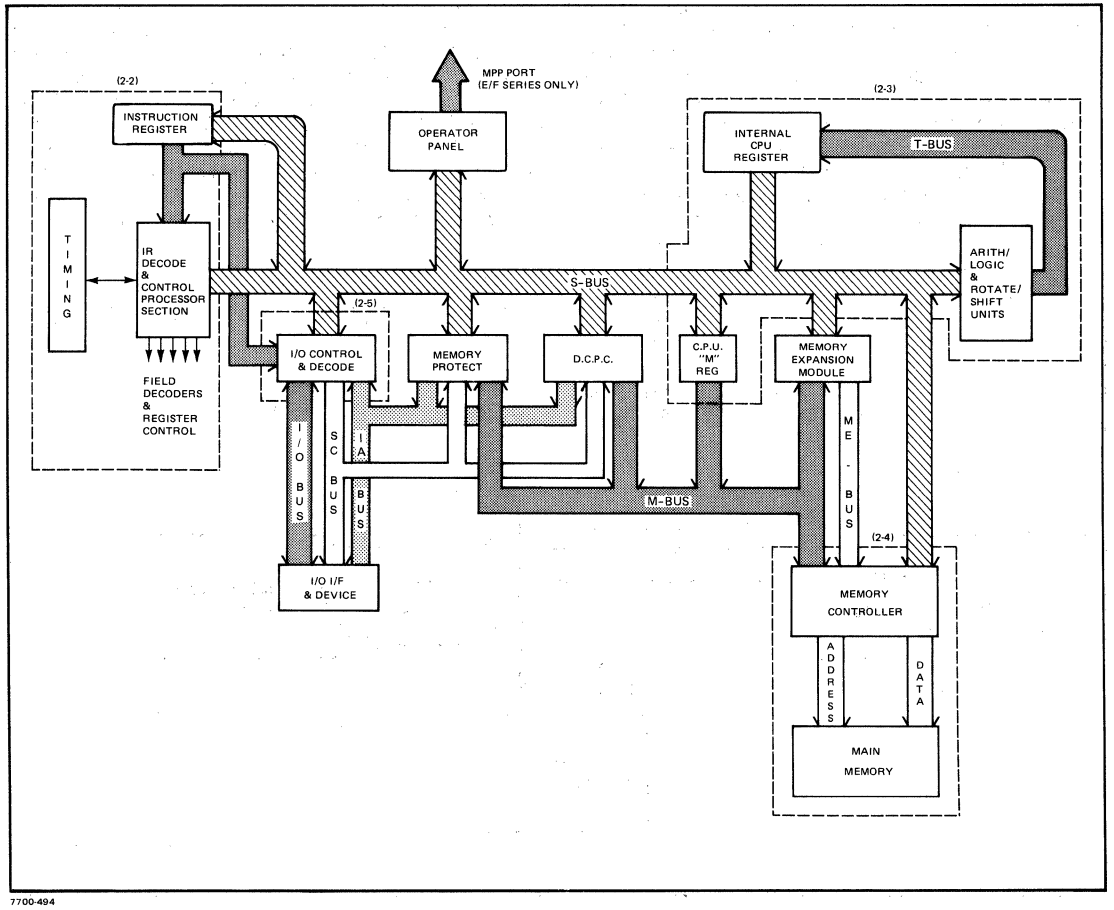
c. Control the computer data manipulations.

d. Initiate I/O signal sequences.

e. Control the Operator Panel.

f. Communicate with Memory Protect.

g. Provide system timing for all other computer sections.



7700-494

Figure 2-2. Computer Functional Block Diagram

h. Provide control processor synchronization with memory and I/O timing as required.

i. Provide effective execution of computer instructions.

2-3. ARITHMETIC/LOGIC SECTION

The Arithmetic/Logic Section contains all the computer's working registers and the necessary logic to perform arithmetic and logic operations on data. Resultant data is transferred between elements in this section via the T-bus. Data is transferred between this section and the rest of the computer via the S-bus. The primary tasks of the Arithmetic/Logic Section are as follows:

a. Provide temporary register storage of memory data.

b. Perform arithmetic and logical operations on data received from other computer registers and to modify and manipulate this data as instructed by the computer program.

c. Provide status indications of computed results as an operations aid (overflow, extend, and special flags).

2-4. MAIN MEMORY SECTION

The Main Memory Section consists of a memory controller and one or more memory module boards with which the controller is designed to operate. The memory module boards contain semiconductor memory arrays that are jumper selectable to various address spaces. The memory controller is the interface to and from the Main Memory Section and responds to read/write requests, generates proper timing and enabling signals for the memory modules, and controls memory refresh timing and addressing. The primary tasks of the Main Memory Section are as follows:

a. Sustain memory data. Since dynamic semiconductor memory is used, the memory module boards must be refreshed to retain stored data. The memory controller initiates and controls the refresh cycles to fit around read/write requests and Dual-Channel Port Controller (DCPC) cycles.

b. Respond to read and write requests. The memory controller generates the proper timing and enabling signals to perform read or write data transfers. The memory address is obtained from the M-bus and the addressed data is transferred on the S-bus.

c. Inhibit memory cycles upon receipt of violation flags from the Memory Protect. (Memory cycles received from the DCPC are not inhibited.)

2-5. I/O SECTION

Except for the MPP (E-Series computers only), the I/O Section provides the hardware link for communication between the computer and all peripheral devices (The MPP provides direct interfacing under microprogrammed control and is discussed in more detail in Section VI of this manual.) The I/O Section contains the I/O control and select code addressing logic, I/O bus control logic, interrupt control logic, and I/O interface PCA slots required to carry out computer initiated transfer operations and I/O device interrupting transfer operations. (Refer to Section III.) The primary tasks of the I/O Section are as follows:

a. Generate control signals for the I/O interface PCA's.

b. Provide data and status paths for the I/O interface PCA's.

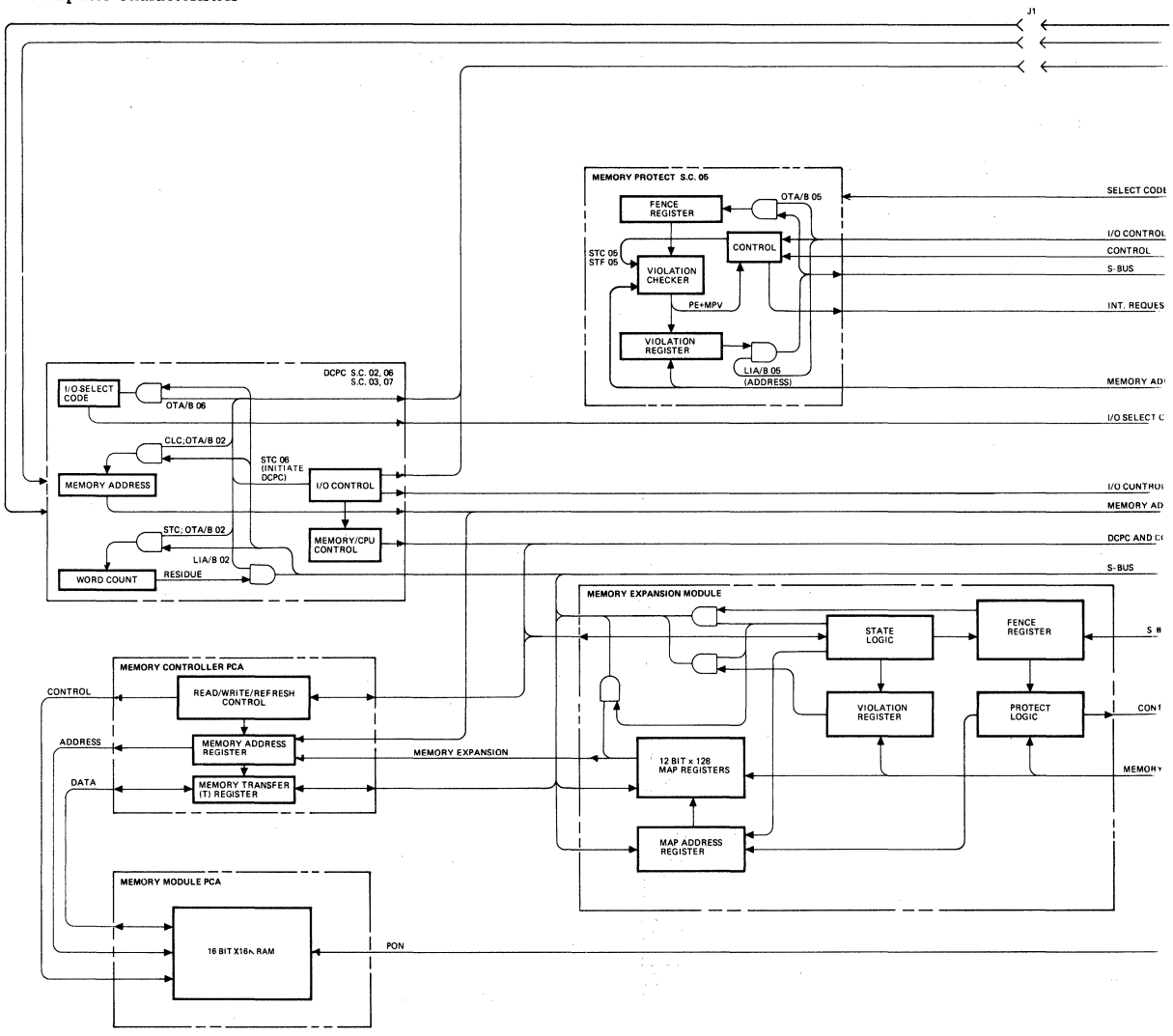
c. Determine select codes of interrupting peripheral devices.

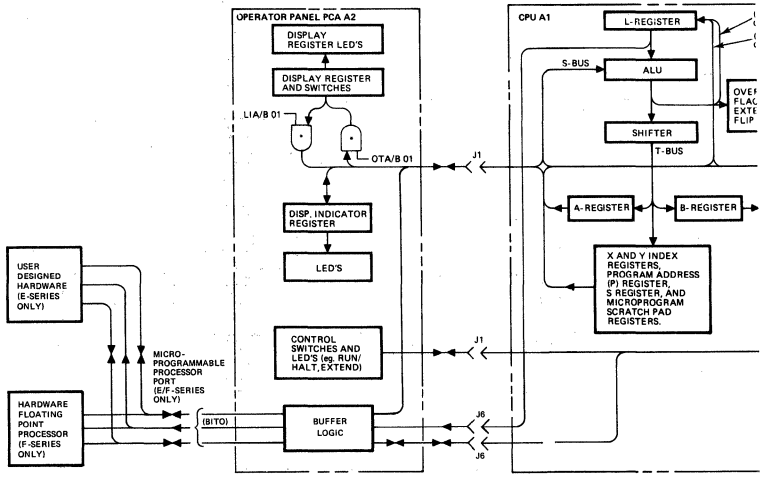
d. Resolve interrupt request priority conflicts.

e. Generate pending interrupt signals for the Control Section.

f. Provide control interface signals for special computer accessories such as the DCPC, Memory Protect, etc.

g. Provide communication lines for I/O extenders.





· THESE TRANSFERS ARE ALSO INITIATED BY MICROCONTROL
·· THESE TRANSFERS ARE CONTROLLED BY DCPC AND MICROCONTROL

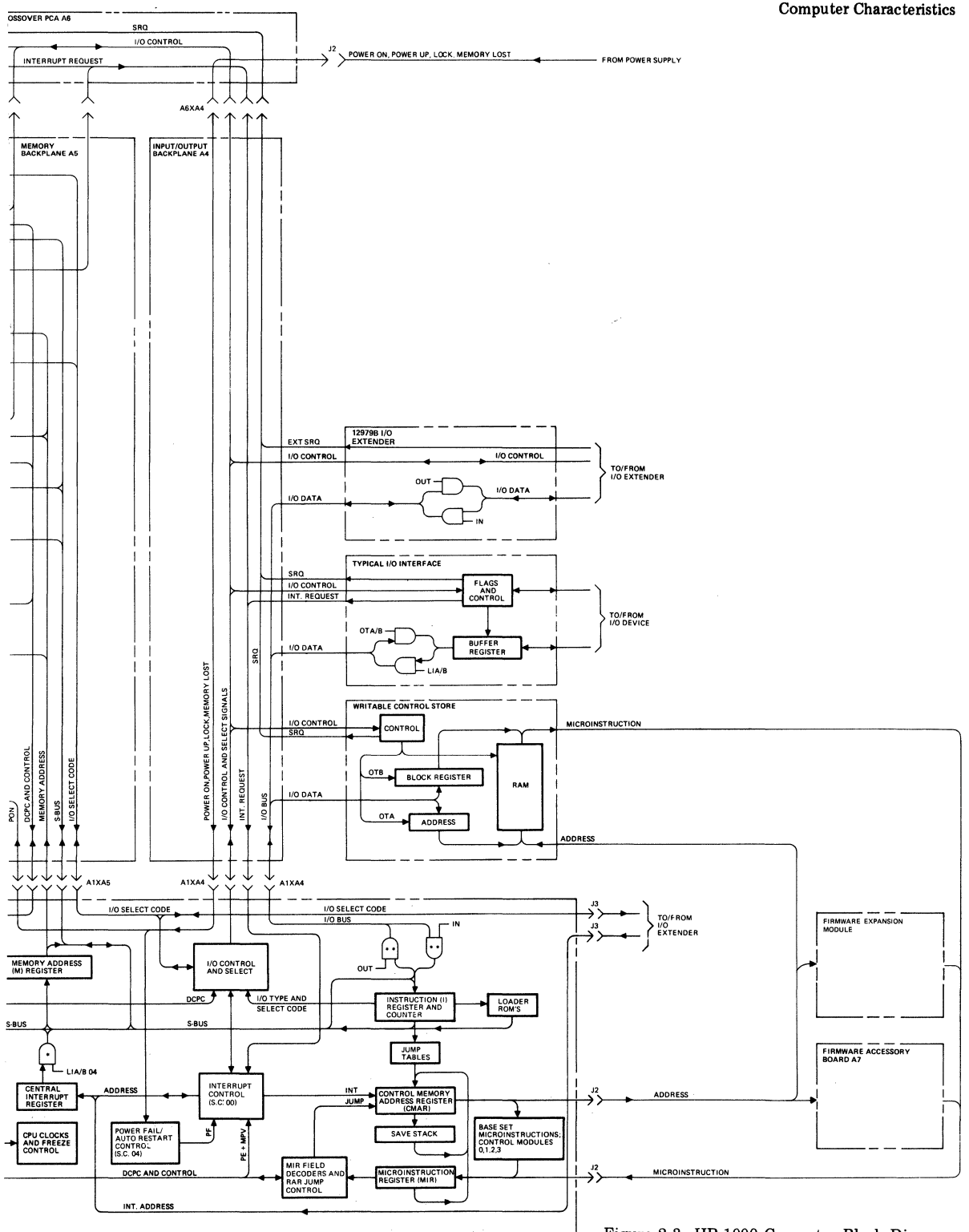


Figure 2-3. HP 1000 Computer Block Diagram

I/O SYSTEM FUNDAMENTALS

SECTION III

This section contains a general discussion of the HP 1000 Computer I/O system. Unless otherwise specified, the contents of this section apply equally to the HP 1000 M-Series, E-Series and F-Series Computers.

3-1. PURPOSE

The purpose of the I/O system is to transfer data between the computer and external peripheral devices. As shown in Figure 3-1, 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 via the I/O bus, S-bus, and CPU, and (3) between the A- or B-register and memory via the S-bus and memory controller. This three-step process also applies to an output transfer except in reverse order. This type of transfer, which is executed under machine instruction program control, allows the computer logic to manipulate the data during the transfer process.

As shown in Figure 3-1, data may also be transferred automatically under control of the Dual-Channel Port Controller (DCPC). Once the DCPC has been initialized, no programming is involved and the transfer is reduced to a two-step process: (1) between the external device and its interface PCA in the computer and (2) between the interface PCA and memory via the I/O bus, S-bus, and memory controller. The two DCPC channels are program assignable to operate with any two device interface PCA's. Since a DCPC transfer eliminates programmed loading and storing via the accumulators, the time involved is shorter than the programmed I/O method. Also, since DCPC operates on a cycle-stealing basis, instruction execution can occur concurrently with DCPC operation. Therefore, the DCPC is normally used with high-speed external devices capable of transferring data at the rates specified in Table 1-1. Additional information on DCPC I/O transfers is contained in paragraph 3-25.

In addition, Figure 3-1 shows that data may be transferred under microprogram control in the HP 1000 E-Series and F-Series Computers. Both the E-Series and F-Series Computers allow data transfers via a user-designed block I/O interface PCA connected to the I/O bus. For these microprogrammed block I/O operations, an input transfer occurs in two steps: (1) between the external device and its interface PCA in the computer, and (2) between the interface PCA and memory via the I/O bus, S-bus and memory controller. This two-step process also applies to an output transfer except in reverse order. The E-Series Computers permit I/O data transfers via a user-designed interface PCA connected to the Microprogrammable Processor Port (MPP). Design of the MPP permits external devices to be connected directly to the CPU and interfaced under microprogram control. For these transfers, an input transfer also occurs in two steps: (1) between the external device and its interface PCA, and (2) between the interface PCA and memory via the MPP, S-bus and memory controller. This two step process also applies to an output transfer except in reverse order. Microprogrammed I/O operations can be used with exceptionally fast external devices capable of transferring data at the rates specified in Table 1-1. Additional information on microprogrammed I/O transfers is contained in Section VI, Advanced Interfacing Techniques.

3-2. I/O SECTION CONTROL

3-3. GENERAL

Functionally, the I/O Section allows the computer to select and communicate with each of the I/O device's associated interface PCA(s) through I/O control and address logic and through direct bus wiring. The structure of the I/O Section also provides a means by which I/O devices can interrupt the computer program in order to be serviced by the computer. When more than one I/O device requests an interrupt, the computer processes the requests on a priority basis. Figure 3-1 illustrates the main sections of the computer concerned with the control of I/O operations. All sections shown are contained in the computer mainframe except for the I/O devices. Although the S-bus is represented as a single line, it actually consists of 16 individual hardwired lines.

3-4. I/O CONTROL INSTRUCTIONS

The I/O instructions generate signals that are translated into appropriate control, flag test and select code signals. The control signals generated by the CLC and STC instructions clear and set the interface PCA's Control flip-flop. In a similar fashion, the control signals generated by the CLF and STF instructions clear and set the interface PCA's Flag flip-flop. The flag test instructions, SFC and SFS, monitor the status of the associated I/O device flag. These control and flag test signals are routed to the appropriate I/O interface PCAs as determined by the select code signals. Each I/O slot is permanently assigned to an individual select code through the computer's hardware design. The select code bus allows each I/O interface PCA and associated I/O device to be individually addressed.

3-5. I/O DATA TRANSFER INSTRUCTIONS

Data transfer instructions initiate either an input or output data transfer between the computer and an I/O device

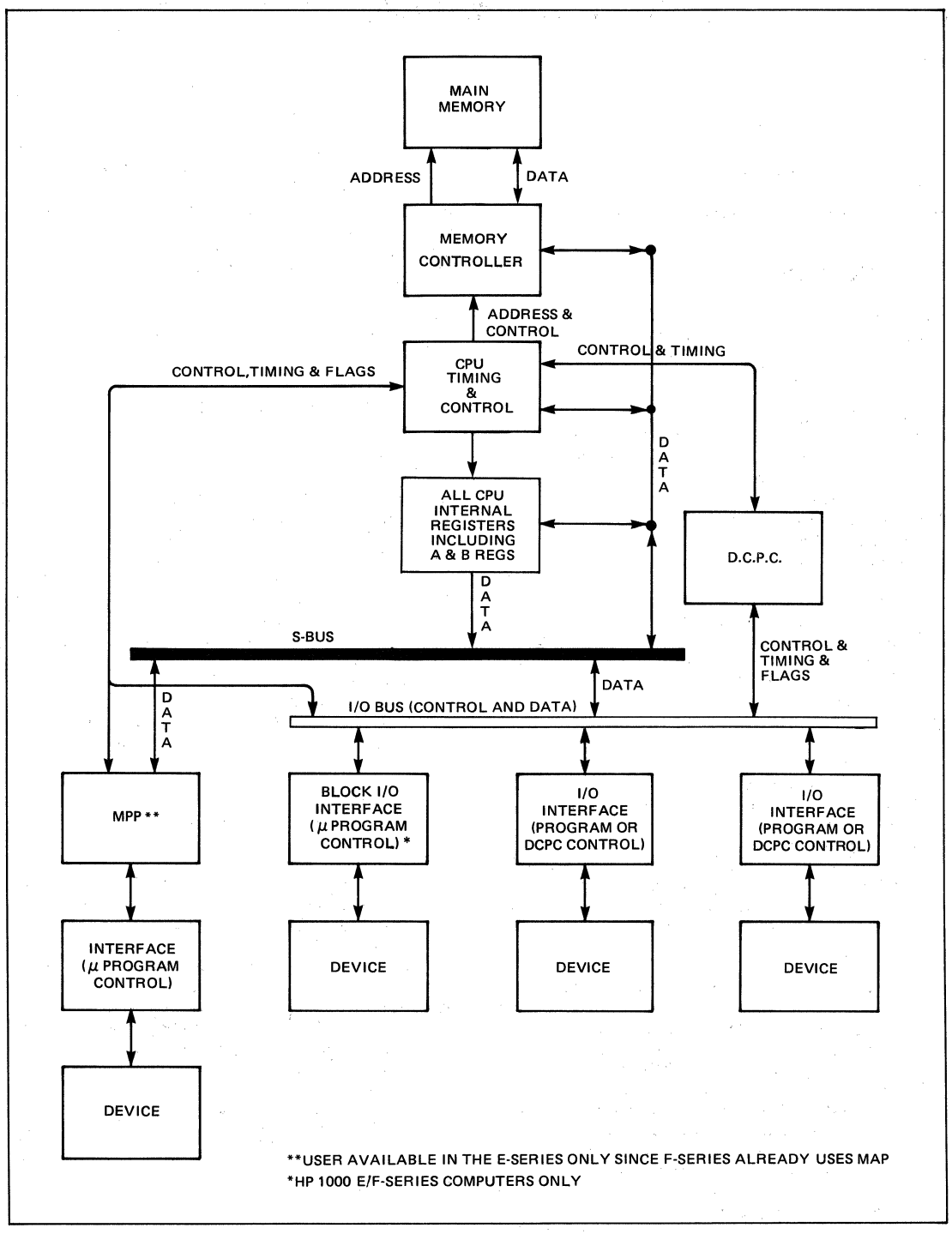


Figure 3-1. Input/Output System

by sending either an I/O Input (IOI) signal or an I/O Output (IOO) signal to the I/O interface PCA. The IOI signal clocks the appropriate interface PCA for input data as a result of a Load Into A (LIA), Load Into B (LIB), Merge Into A (MIA), or Merge Into B (MIB) instruction. Only the data from the interface PCA addressed by the select code is enabled. The data is clocked by IOI into either the A- or B-register via the I/O bus and S-bus. The IOO signal is applied to the interface PCA's as a result of an Output From A (OTA) or Output From B (OTB) instruction. This signal, when combined with the appropriate select code signal, clocks data from either the A- or B-register into the addressed interface PCA via the S-bus and I/O bus. IOI and IOO have the same function when operating under DCPC control except that both signals are generated by the DCPC hardware.

3-6. INTERRUPT REQUESTS

The instruction STF 00 enables the interrupt system. This system allows an external I/O device to request interrupt service. (The interrupt system is explained in more detail in paragraph 3-10.) To request an interrupt, the I/O device applies an Interrupt Request (IRQ) signal to the computer via its interface PCA. The I/O Section then determines the address of the interrupting device and causes an interrupt to the main computer program. The computer sets the Memory Address register (M-register) to the select code of the interrupting device. This memory location (or "trap cell") generally contains a Jump to Subroutine Indirect (JSB,I) instruction which transfers program control to the appropriate interrupt service routine. A typical interrupt service routine accepts input data from the I/O device or outputs data from the computer to the device. Upon completion of the interrupt service routine, program control is returned to the previously interrupted main program.

3-7. INTERFACE PCA'S

The interface PCA's provide data transfer channels between the computer and the external I/O devices, and provide control (via computer commands) for the I/O device's operation. The interface PCA's usually contain flag and interrupt logic circuits, and registers for temporary storage of data being transferred to or from the computer. Requirements for the use of the flag and interrupt logic circuits and the number of storage registers designed into the interface PCA depend on the type of I/O device to which it is connected. (Refer to Section V for a detailed discussion of interface PCA design.)

3-8. I/O TIMING

An I/O cycle is the time required to generate all I/O signals necessary to execute an I/O instruction. Each I/O cycle is divided into five T-periods designated T2, T3, T4, T5 and T6. The control processor provides the required timing for the I/O signals resulting from decoded I/O instructions and interrupt requests. Timing of the I/O signals in relation to the I/O time periods is discussed in Section IV. Two I/O time period signals, Enable Flag (ENF) corresponding to T2 and Set Interrupt Request (SIR) corresponding to T5, provide control signals for interrupt processing. I/O time period T2 (ENF) is used during I/O operations to synchronize the interfaces PCA's Flag flip-flop with the beginning of the computer's I/O cycle. I/O time period T5 (SIR) is used during interrupt processing to synchronize the setting of the interface PCA's Interrupt Request (IRQ) flip-flop.

3-9. I/O ADDRESSING

Each interface can be uniquely addressed through the use of select codes. A select code specifies one of 64 (decimal, 77 octal) possible I/O devices or functions. Select code signals corresponding to the two-digit octal address found in the select code field of an I/O instruction are transferred to the I/O slot of the selected interface PCA. This enables the device's interface PCA to respond to commands and/or transfer data to or from the I/O device. Select codes 00 through 07 (octal) are dedicated to specific functions and select codes 10 through 77 (octal) are used to address I/O devices. Table 3-1 lists the select codes and their assignments, and indicates the interrupt location (trap cell) corresponding to each select code. Figure 3-2 illustrates the I/O slots in the computer into which interface PCAs are installed.

Each I/O slot is assigned two select codes in order to service I/O devices that may be capable of performing two distinct functions. For example, if an I/O device is capable of responding to commands as well as inputting and/or outputting data, two separate select codes may be required. Since the two select codes are hardwired to each I/O slot, each interface PCA can be designed to respond to either a lower select code (LSC) or a higher select code (HSC) or both. The interface PCA assumes the select codes of the slot into which it is inserted.

The select code field of the I/O instructions generates the Select Code Most Significant Digit (SCM) and the Select Code Least Significant Digit (SCL) signals to determine which I/O slot is to be addressed. These signals are applied to the various I/O slots in the fashion illustrated in Figure 3-3. Note that the SCM(1) signal is applied to the most-significant digit input pins on the I/O slot connectors with select codes 10 through 17 (octal) and that SCM(2) is applied to the I/O slot connectors with select codes 20 through 27 (octal), and so on. The SCM(0) and SCL(0) through SCL(7) signals are used to form select codes 00 through 07 (octal). The functions of these select codes are given in Table 3-2. It should be noted that the SCM and SCL signals are applied to the same pins on all I/O slot connectors as follows:

a. Pin 14 — lower select code, most significant digit.

b. Pin 16 — lower select code, least significant digit.

Table 3-1. Select Code Assignments

SELECT CODE (OCTAL)	INTERRUPT MEMORY LOCATION (OCTAL)	ASSIGNMENT
00	None	Interrupt System Enable/Disable
01	None	Display Register or Overflow
02	None	DCPC Initialization Channel 1
03	None	DCPC Initialization Channel 2
04	00004	Power Fail Interrupt/Central Interrupt Register
05	00005	Parity Error Interrupt/Memory Protect Interrupt/Dynamic Mapping System Interrupt
06	00006	DCPC Channel 1 Completion Interrupt
07	00007	DCPC Channel 2 Completion Interrupt
10 thru 77	00010 thru 00077	I/O Devices

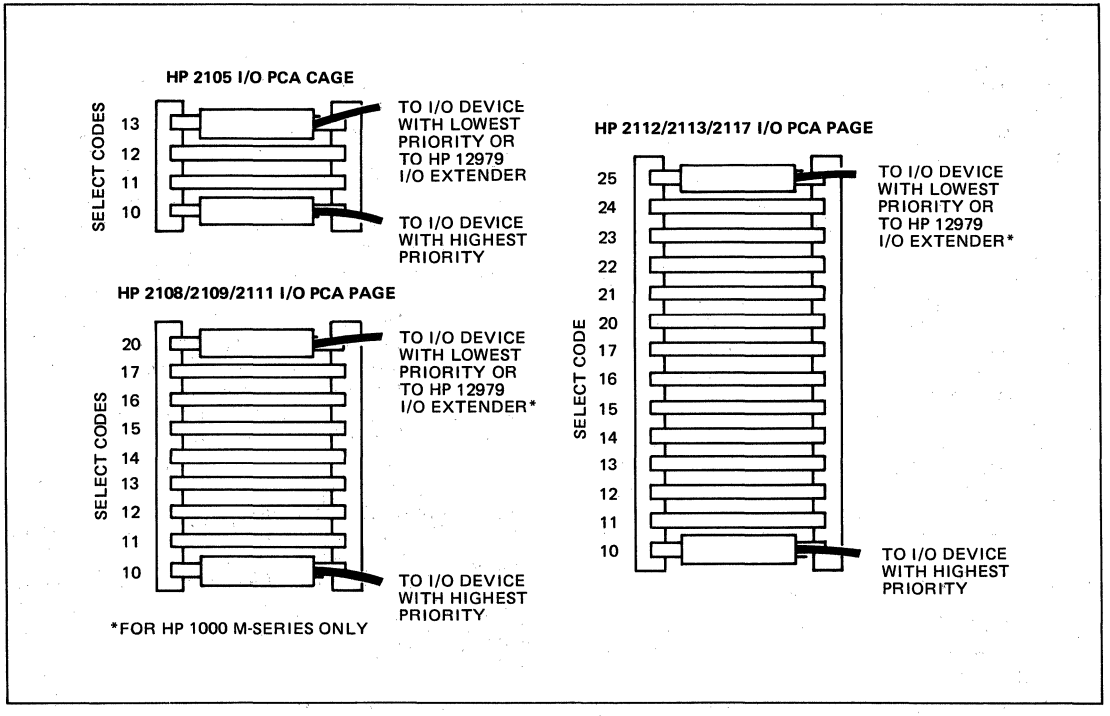
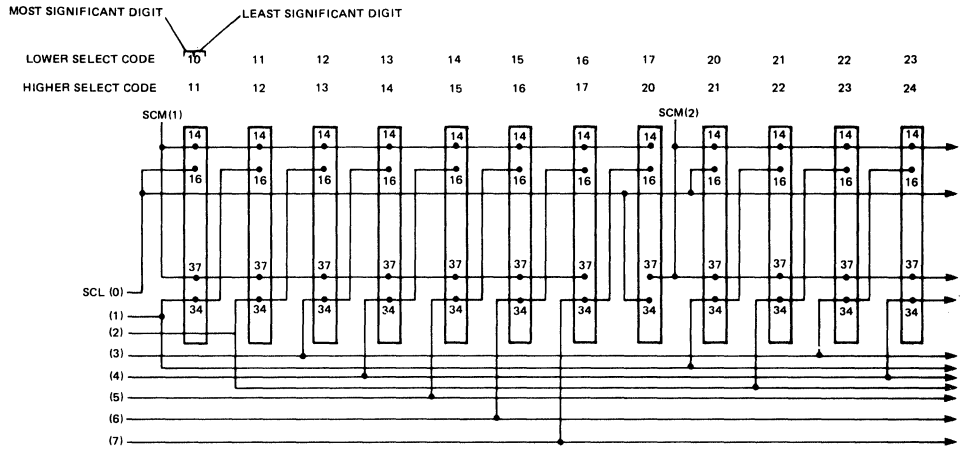


Figure 3-2. I/O Address Assignments



7700-495

Figure 3-3. I/O Slot Connector Select Code Wiring

c. Pin 37 — higher select code, most significant digit.

d. Pin 34 — higher select code, least significant digit.

3-10. INTERRUPT SYSTEM

The interrupt system provides the means for an external device to interrupt the program in progress. Generally, interrupts are used for interrupt data transfers, in which a background (main) program is interrupted when data is available for input or when additional output data can be accepted, or to signal the main program that an external event has occurred. An interrupt request occurs when the following conditions are met:

a. The interrupt system is enabled with a STF 00 instruction,

b. The specific I/O device's interface PCA Control flip-flop is set,

c. No instruction that affects priority (STF,CLF,STC or CLC) is in progress,

d. The priority network is not disabled by gaps between interface PCAs as discussed in paragraph 3-13.,

e. No higher priority I/O devices have met the conditions stated above, disabling the specified interface PCA, and

f. The specific I/O device's interface PCA Flag flip-flop is set.

Program control of the interrupt system is provided with the STF and CLF instructions. A Set Flag (STF) instruction with a select code of 00 (octal) enables the interrupt system. A Clear Flag (CLF) instruction with a select code of 00 (octal) disables the interrupt system. The interrupt system is disabled when power is initially applied to the computer. In addition, the Control Reset (CRS) signal is generated at power up which clears all interface PCA's Control flip-flop and the Power On Preset to I/O (POPIO) signal sets all Flag Buffer flip-flops. Therefore, to operate any device under interrupt system control, it is first necessary to clear the addressed I/O interface PCA's Flag Buffer and Flag flip-flops with a CLF sc instruction and to set the Control flip-flop with a STC sc instruction.

3-11. INTERRUPT PRIORITY

The interrupt system contains a priority network that establishes an orderly sequence of granting interrupt requests. The following paragraphs contain discussions of priority assignments, network operation, continuity, and instructions.

3-12. PRIORITY ASSIGNMENTS. A priority circuit on the I/O interface PCA's allows only one I/O device

Table 3-2. Interrupt and I/O Control Summary

INST	S.C. 00	S.C. 01	S.C. 02	S.C. 03	
STC	NOP	NOP	Prepares DCPC channel 1 to receive and store the block length in 2's complement form.	Prepares DCPC channel 2 to receive and store the block length in 2's complement form.	
CLC	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	Loads A/B register with all zeros. (Equivalent to CLA/B instruction.)	Loads display 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	Equivalent to a NOP.	Merges display 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.	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).	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 and restores interrupt capability to lower priority functions.	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.
	Flag FF sets automatically when power comes up. (No program control possible.)	Turns on parity error interrupt capability.	Aborts DCPC channel 1 data transfer.	Aborts DCPC channel 2 data transfer.	Sets PCA Flag FF.
	Flag FF clears automatically when power fail occurs. (No program control possible.)	Turns off parity error interrupt capability and clears violation register bit 15.	Clears Flag FF on DCPC channel 1.	Clears Flag FF on DCPC channel 2.	Clears PCA Flag FF.
	NOP	Skip if Dynamic Mapping System (DMS) interrupt.	Tests if DCPC channel 1 data transfer is complete.	Skip if DCPC channel 2 data transfer is complete.	Skip if I/O channel Flag FF is set.
	Skip if power fail has occurred.	Skip if memory protect interrupt.	Tests if DCPC channel 1 data transfer is still in progress.	Skip if DCPC channel 2 data transfer is still in progress.	Skip if I/O channel Flag FF is clear.
	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	Loads A/B register with all ones. (Equivalent to CCA/CCB instruction.)	Loads A/B register with all ones. (Equivalent to CCA/CCB instruction.)	Loads contents of PCA data 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.	Same as LIA/B 06 above.	Same as LIA/B 07 above.	Merges contents of PCA data buffer into A/B register.
	NOP	Outputs first address of unprotected memory 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 data buffer.

to interrupt the computer program regardless of the number of I/O devices requesting an interrupt. The priority network assigns the highest priority to select code 04 (octal) which is reserved for power fail interrupt and decreasing priority to select codes 05 through 77 (octal) as listed in Table 3-1. The interrupt priority assignments for each I/O interface PCA connector are fixed but, since any I/O interface PCA can be inserted into any connector slot, the interrupt priority of any I/O device can be easily changed simply by inserting the device's associated I/O interface PCA into another connector slot.

3-13. PRIORITY NETWORK OPERATION. Interrupt priority is established by a hardwired priority network on both the individual I/O interface PCA's and on each I/O connector slot. A simplified representation of the priority network is given in Figure 3-4. As illustrated, the interrupt priority decreases as the select code increases. Interrupts on select code 04 through 07 (octal) take priority over interrupts from I/O devices, with the Power Fail interrupt signal on select code 04 (octal) having the highest priority. When an I/O device requests an interrupt, the priority enable line is broken to lower priority devices.

The interrupt priority network is shown in more detail in Figure 3-5. An I/O device requests interrupt service by setting the Flag flip-flop on its associated interface PCA. If the IEN signal is asserted, the associated interface PCA's Control flip-flop is set and the priority enable line has not been broken, the device's interrupt service routine (ISR) will be executed. Figure 3-5 illustrates the operation of the priority network —

- Slot 10 passes priority (the PCA's Flag flip-flop is not set).
- Slot 11 drops priority to lower priority cards (Flag set).
- Slot 12 fulfills all conditions for an interrupt except that the priority enable line has been dropped at slot 11.
- Slot 13 contains a priority jumper PCA to continue the priority enable line to slot 14. In this case, the priority line has been previously dropped.

In this way, an interrupt service routine (ISR) for any I/O device can be interrupted by any other device with a higher priority. After the higher priority device has been serviced, the lower priority device's ISR can continue. As a result, several ISRs can be in an interrupted state at one time. Each ISR continues from its interrupted point when the next higher priority ISR is completed.

Interrupt priority can also be program controlled. Since an interrupt cannot occur unless the I/O interface PCA's Control flip-flop is set, all Control flip-flops on I/O interface PCAs with a higher priority than the one desired can be cleared by a CLC sc instruction. This prevents the higher priority I/O interface PCA's from requesting an interrupt and establishes the desired I/O interface PCA as the highest priority device. However, the I/O interface PCA's disabled by the CLC instruction must now be monitored for service by testing the state of their Flag flip-flops or by resetting the Control flip-flops with an STC sc instruction.

3-14. PRIORITY CONTINUITY. When an I/O interface PCA requests an interrupt, the PRL signal applied to the next lower priority I/O interface PCA as PRH goes false, which prevents that I/O interface PCA from requesting an interrupt. This sequence continues from PCA to PCA until the last (lowest priority) I/O interface PCA receives a false PRH signal.

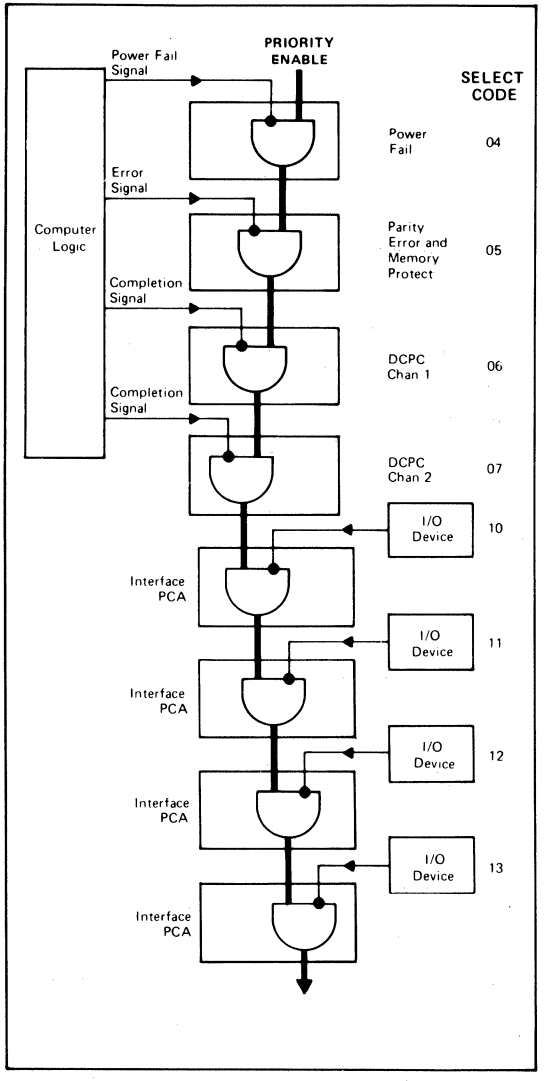
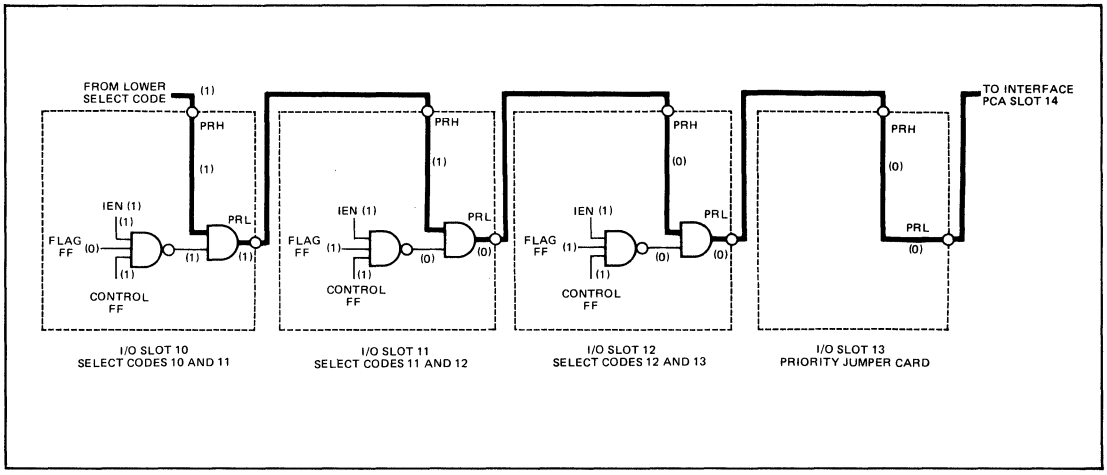


Figure 3-4. Priority Linkage



7700-496

Figure 3-5. Interrupt Priority Network

If an I/O interface PCA contains two separately addressable logic circuits such as input/output or command/data, continuity of the priority network must be ensured internally on the PCA as illustrated in Figure 3-6. There can be no gaps in the priority network if the system is to function properly. In addition, if a two-select-code PCA is used, only the higher select code of the next lower priority I/O slot is available. For example, slot 10 in Figure 3-6 contains an I/O interface PCA that uses both select codes 10 and 11. The next select code available (in slot 11) is select code 12. If the next lower priority PCA is not set up to use a higher select code (in this example 12), a jumper PCA must be inserted in slot 11 to ensure continuity of the priority network. The next lower priority PCA would be inserted into slot 12 in this example case.

3-15. INSTRUCTIONS AFFECTING PRIORITY. Four instructions (STC, CLC, STF, and CLF) affect the I/O interface PCA's priority network. Whether and I/O device can or cannot request an interrupt depends on whether its I/O interface PCA's Control flip-flop is set (STC) or cleared (CLC) and whether its Flag flip-flop is set (STF) or cleared (CLF). If an I/O device cannot request an interrupt, all succeeding lower priority devices assume a priority of one higher in the priority network. The four instructions (along with SFS and SFC) also inhibit all interrupts during the computer cycle in which they occur. This prevents interrupts from occuring during entry and exit from subroutines. Also, a combination of two of the four instructions is normally the next-to-last instruction in a service subroutine processing an interrupt. (The last instruction is a JMP,I instruction to return to the main program or to an address of another subroutine.) If another I/O device could interrupt the subroutine immediately after the combination of the two instructions and before the JMP,I instruction, the possibility would exist that the first I/O device could interrupt a second time before the JMP,I instruction. In this case, the first main program address or the other service subroutine address would be destroyed, preventing a return to the main program or to the other service subroutine.

3-16. INTERRUPT GENERATION

A detailed discussion of the sequence of events that take place during an interrupt request is contained in paragraphs 4-4 through 4-8 and paragraph 5-4 of this manual.

3-17. INTERRUPT PROCESSING

Initially, during an interrupt, the computer decrements the P-register to ensure that the proper location in the main program will be returned to after the interrupt is processed. Also, the computer places the service request address (always equal to the select code of the interrupting I/O device) into the M-register. This addresses the memory location having the same number as the service request address (select code). This location in memory is referred to as the "trap cell" and is reserved for one particular I/O device. For example, an I/O device specified by select code 10 (octal) will interrupt to (i.e., cause execution of the contents of) memory location 00010. The computer fetches the instruction in the trap cell which is usually a jump to a subroutine (JSB,I) instruction. (Any legal instruction can be placed in the trap cell.) The contents of the P-register plus one are then stored in the first location (X) of the subroutine. (Since the previous contents of the first memory location are destroyed when P+1 is stored, the first instruction of the subroutine should always be a no-operation (NOP) instruction or equivalent.) Next, the location of the subroutine (X+1) is placed in the P- and

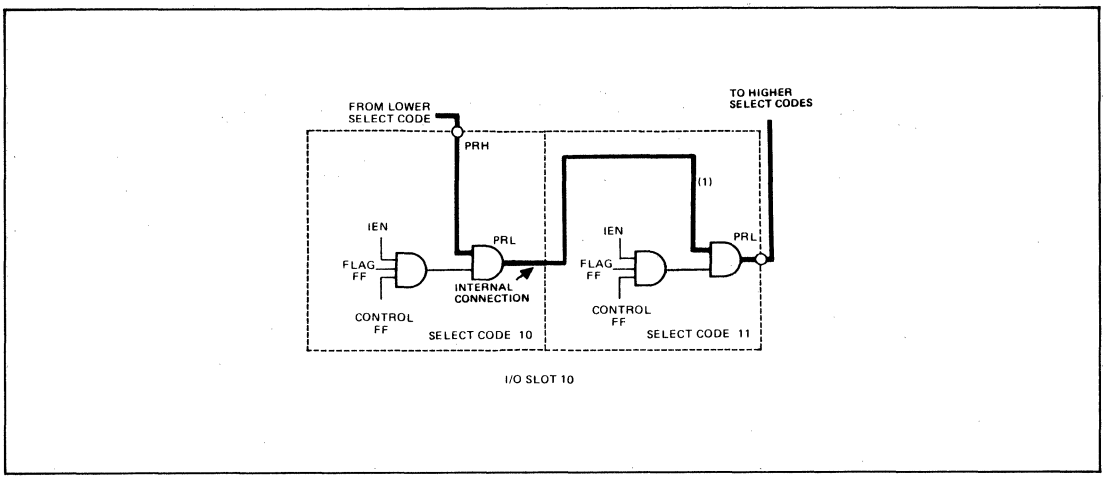


Figure 3-6. Internal Priority Continuation

M-registers and the computer resumes operation in the subroutine. Thus, the instruction stored in location X+1 is the first instruction of the subroutine to be executed. It should be noted that the contents of the working registers that were in use in the main program should be stored when entering the subroutine and restored before exiting from the subroutine. Exit from the subroutine is accomplished with a JMP,I instruction to location X. This places the address of the interrupted program instruction in the P- and M-registers and allows normal program operation to resume.

3-18. I/O DATA TRANSFERS

The following paragraphs describe how data is transferred between memory and I/O devices. A summary of I/O group instructions pertinent to the computer's interrupt and control functions is contained in Table 3-2. The sequences presented for noninterrupt and interrupt methods of data transfer are highly simplified in order to present an overall view without the involvement of software operating systems and device drivers. Certain I/O interface PCAs may require additional or different control and data transfer protocols. For additional information, refer to Sections IV through VI of this manual and to the documentation supplied with the appropriate software system or I/O subsystem.

3-19. NONINTERRUPT TRANSFERS

It is 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 the examples in Table 3-3. Each of these routines will transfer one word or character of data. It is also assumed that the interrupt system is disabled (STF 00 not previously given). The select codes used are for example purposes only.

Table 3-3. Noninterrupt Transfer Example Routines

INSTRUCTIONS	COMMENTS
INPUT ROUTINE	
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 ROUTINE	
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.

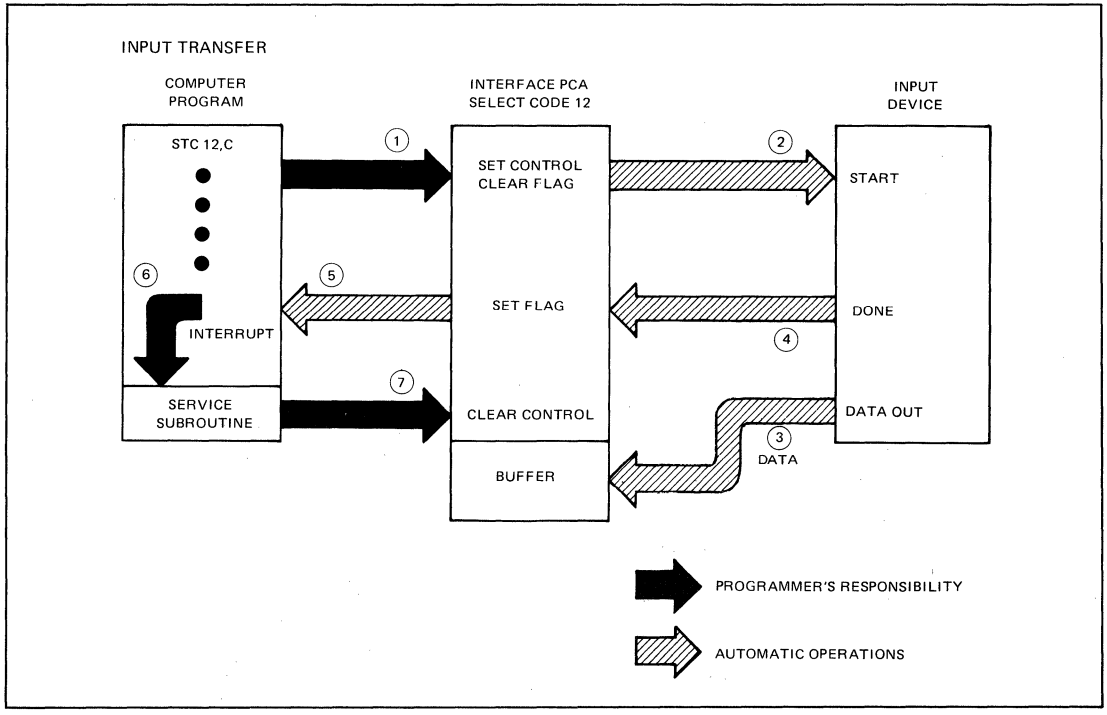
3-20. INPUT DATA. In the example in Table 3-3, operation begins with a programmed STC 12,C instruction which sets the Control flip-flop and clears the Flag flip-flop on the interface PCA. The computer then goes into a waiting loop, repeatedly checking the status of the PCA Flag flip-flop. Setting the Control flip-flop causes the PCA to output a Start (Device Command) signal to the I/O device. The Start signal causes the device to output a data character and then a Done signal to the PCA which sets the PCA Flag flip-flop. (A more detailed discussion of interface PCA operations is contained in Sections IV and V.) 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 then exits from the waiting loop and the LIA 12 instruction loads the device input data into the A-register.

3-21. OUTPUT DATA. The first step, in the example in Table 3-3, 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 discussed 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.)

3-22. INTERRUPT TRANSFERS

3-23. INPUT DATA. Figure 3-7 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 operation 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 sends out a data character (3) and then asserts the Done signal (4). (A more detailed discussion of the interface PCA operations is contained in Sections IV and V.)

The device Done signal sets the PCA Flag flip-flop which in turn generates an interrupt (5) 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 (accomplished in step 1).



0330-2

Figure 3-7. Input Data Transfer (Interrupt Method)

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 (JSB,I) 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 or CLF 12. This step (7) 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 at the completion of each data transfer operation. At the end of the subroutine, control is returned to the interrupted program via previously established linkages.

There are some cases in which it is desirable to use the noninterrupt I/O data transfer method while the interrupt system is enabled. For example, the interrupt system remains enabled while a noninterrupt transfer is being carried out to an I/O device in a privileged interrupt environment. When a noninterrupt input transfer is used with the interrupt system enabled in an HP E-Series or F-Series computer, certain programming considerations have to be taken into account. Care must be taken as to when the device flag is cleared since all I/O control instructions (SFC, SFS, STF, CLF, STC, and CLC) hold off interrupts until the following instruction is executed. The object here is to prevent the interface PCA on which the noninterrupt transfer is occurring from causing an interrupt. The usual sequence of instructions used to handle an input data transfer are the following:

```
SFS sc
JMP *-1
LIA sc,C
```

In this example, the SFS sc instruction senses that the device flag is set but holds of interrupts until after execution of the LIA sc,C instruction. Since the device flag is cleared by the LIA sc,C instruction, the interrupting device select code is not available to the Central Interrupt Register (CIR) when it is clocked. Instead, zero is loaded into the CIR. As with normal interrupt processing, the contents of the CIR determine the main memory address where the next instruction to be executed is found. Since the CIR contains zero, the contents of the A-register will be executed. This is, of course, undesirable because the A-register may contain input data or garbage at this point instead of the usual JSB,I instruction.

Proper operation can be ensured in one of two ways. The first consists of placing a CLF sc instruction before the LIA sc instruction. This technique eliminates the interrupt altogether. For example —

```
SFS 12
JMP *-1
CLF 12
LIA 12
```

The second method involves inserting an instruction (except an I/O control instruction) between the JMP *-1 and the LIA sc,C instructions and loading the corresponding trap cell with zero. In this case, the interrupt actually occurs after the inserted instruction and before the flag is cleared but program execution immediately returns to the LIA sc,C instruction. For example —

```
SFS 12
JMP *-1
NOP
LIA 12,C
```

3-24. OUTPUT DATA. Figure 3-8 illustrates the sequence of events required to output data using the interrupt method. Again, note the distinction between programmed instructions and automatic operations. It is assumed that the data to be transferred has been loaded from memory into the A-register and is in a form suitable for output. The interface PCA in this example is assumed to be in the I/O 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 interface PCA to output the buffered data (3) followed by a Start signal (4) to the device which writes (e.g., punches, stores, etc.) the data character and asserts the Done signal (5).

The device Done signal sets the PCA Flag flip-flop which in turn, generates an interrupt (6) provided that the interrupt system is on, priority is high, and the Control flip-flop is set (accomplished in Step 2). The interrupt causes the current computer program to be suspended and control is transferred to a service subroutine (7). It is the programmer's responsibility to provide the linkage (JSB,I) 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 can 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 or clear flag. This step (8) 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 linkage.

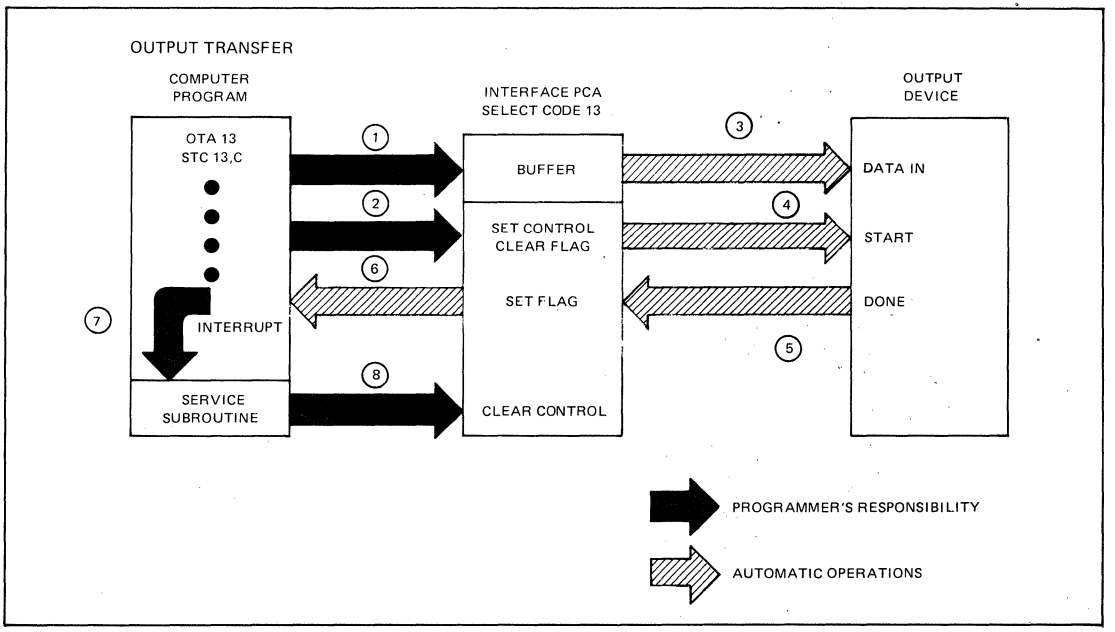
3-25. DUAL-CHANNEL PORT CONTROLLER

The Dual-Channel Port Controller (DCPC) accessory 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 transfer data at the rates specified in Table 1-1.

There are two DCPC channels, each of which may be separately assigned to operate with any I/O interface PCA. The two channels are capable of operating simultaneously in an interleaved fashion and, when in this mode, channel 1 has priority over channel 2. For the HP 1000 M-Series Computers, the combined maximum transfer rate for both channels operating simultaneously is 1.23 megabytes per second; the rate available on channel 2 is the rate difference between 1.23 and the actual operating rate of channel 1. For the HP 1000 E/F-Series Computers, the combined maximum transfer rate for both channels operating sinultaneously is as specified in Table 1-1 only if the channels are transferring data in the same direction. Channels transferring data in opposite directions will achieve an effective transfer rate between the specified input and output rates. Both channels do not have to be operating to achieve the full DCPC bandwidth. The DCPC hardware must steal every consecutive I/O cycle to attain the full DCPC bandwidth. The maximum transfer rates still remain at those previously mentioned. Since the memory cycle rate is somewhat faster than the I/O cycle rate, it is possible for the CPU to interleave memory cycles while the DCPC is operating.

Transfers via the DCPC are on a full-word basis; hardware packing and unpacking of bytes is 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 signaled 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 (SFS or SFC) for select code 06 or 07, or by interrogating the word count register with an LIA/B instruction 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 if the DCPC hardware is not stealing every cycle.

3-26. DCPC INPUT TRANSFER. Figure 3-9 and Table 3-4 illustrates the sequence of operations for a DCPC input transfer. A comoparison with the conventional interrupt method (Figure 3-7) shows that much more of the DCPC operation is automatic. Remember that the procedure in Figure 3-7 must be repeated for each word or character. In Figure 3-9, the automatic DCPC



0330-3

Figure 3-8. Output Data Transfer (Interrupt Method)

operation will input a block of data any size limited only by the available memory space. The initialization routine sets up the control registers on the DCPC (1) and issues the first start command (e.g., STC 12,C) directly to the interface PCA. The DCPC logic is now turned on and the computer program continues with other instructions.

Setting the Control and clearing the Flag flip-flop (2) causes the interface PCA to send a Start signal to the external device (3). The device goes through a read cycle and returns with a data word (4), then a Done signal (5). The Done signal sets the PCA Flag flip-flop which, regardless of priority, immediately instructs the DCPC logic to request an I/O cycle (6) and transfer a word into memory (7). 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 (8) and the DCPC logic generates a completion interrupt (9). The program control is now forced to a completion routine (10), the content of which is the programmer's responsibility.

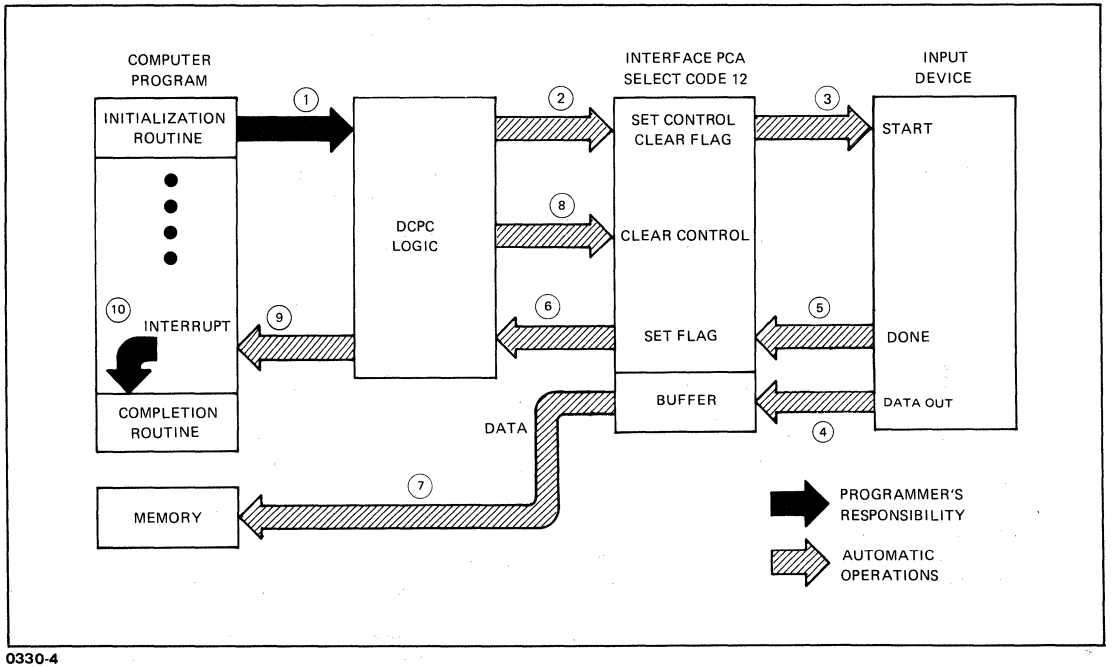
3-27. DCPC OUTPUT DATA TRANSFER. The components that make up a DCPC output transfer are the same as the DCPC input transfer: DCPC PCA, CPU memory system, CPU PCA, I/O interface PCA, and I/O device. The output transfer, however, identifies a specified block (buffer) of data in memory and transfers it to a specified I/O device.

In the case of most output transfers, full handshake protocol between the I/O interface and the I/O device is observed. This discussion assumes this protocol. Two major operations take place during each DCPC output cycle:

1. One 16 bit data word is transferred from memory through the I/O interface and presented to the I/O device's input registers.

2. A Device Command, initiated by a STC from the DCPC PCA through the I/O interface, is used as a "start signal" to tell the I/O device to latch the data into its input registers.

As the device completes the latching of each data word (one word per DCPC cycle), it returns a "flag" to the interface which indicates that the output process has been completed and that it is ready to accept another word. The flag from the device initiates the setting of the Flag flip-flop on the interface which in turn activates the Service Request (SRQ) signal to the DCPC module. SRQ when sensed by the DCPC, is responsible for initiating a DCPC cycle which transfers one data word. This process continues until the word count register located on the DCPC becomes zero, signaling that the entire specified block of memory data has been transferred to the I/O device. The DCPC communicates this completion condition to the CPU and can be interrogated on a "skip if flag set" basis (SFS 6, or SFS 7). If the interrupt system is enabled, this completion flag from either DCPC channel causes an interrupt to the respective DCPC trap cells (memory loca-



0330-4

Figure 3-9. DCPC Input Data Transfer

Table 3-4. DCPC Input Initialization Program

LABEL	OPCODE	OPERAND	COMMENTS
IN	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.
INPUT	STC	10B,C	Start input device.
	STC	6B,C	Activate DCPC Channel 1.
	SFS JMP	6B *−1	Wait while data transfer takes place or, if interrupt processing is used, 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}).

tions 6 or 7). The trap cell (memory location) may contain a JSB,I (call) to the desired DCPC Service Routine.

3-28. OUTPUT TRANSFER INITIALIZATION — Method 1. Due to the backward compatable nature of the DCPC option, the timing protocol necessitates that attention be paid to the method of coding the output transfer. As stated previously, there are two major operations involved in the transfer. The STC, which results in the Device Command, is always issued before the data is presented to the device in a DCPC output transfer. There are many methods of output initialization that depend upon the type of I/O device used. Several of these methods are discussed here.

The first method involves the use of a "timing/party line" flip-flop located on the I/O interface PCA. This flip-flop is used to delay the Device Command until the end of the DCPC cycle, allowing the data to be latched into the output registers of the I/O interface and stabilize at the input registers of the I/O device. A jumper located on the HP 12566 and 12930 Interfaces is used to select this flip-flop. If you wish use this method in conjunction with a custom-designed I/O interface, this flip-flop must be included. This method allows for the standard coding illustrated in Table 3-5. Notice that the actual DCPC setup sequence is common to both output and input transfers. A description of the output sequence follows (refer to Table 3-5 and Figure 3-10):

Table 3-5. DCPC Output Initialization Program — Method 1

LABEL	OPCODE	OPERAND	COMMENTS
ASGN2	LDA	CW1	Fetches control word 1 (CW1) from memory and loads it in A-register.
	OTA	7B	Outputs CW1 to DCPC Channel 2.
MAR2	CLC	3B	Prepares Memory Address Register to receive control word 2 (CW2).
	LDA	CW2	Fetches CW2 from memory and loads it in A-register.
	OTA	3B	Outputs CW2 to DCPC Channel 2.
WCR2	STC	3B	Prepares Word Count Register to receive control word 3 (CW3).
	LDA	CW3	Fetches CW3 from memory and loads it in A-register.
	OTA	3B	Outputs CW3 to DCPC Channel 2.
OUTPT	CLC	11B,C	Turn off the interface (idle the device).
	STF	11B	Assert SRQ to DCPC.
	STC	7B,C	Activate DCPC Channel 2.
	SFS JMP	7B *−1	Wait while data transfer takes place or, if interrupt processing is used, continue program.
.	.	.	
.	.	.	
.	.	.	
	HLT		Halt
CW1	OCT	100011	Assignment for DCPC Channel 2 (ASGN2); specifies I/O channel select code address (11_8), STC after each word is transferred, and CLC after final word is transferred.
CW2	OCT	000200	Memory Address Register control. DCPC Channel 2 (MAR2); specifies memory output operation and starting memory address (200_8).
CW3	DEC	−50	Word Count Register control. DCPC Channel 2 (WCR2); specifies the 2's complement of the number of character words in the block of data to be transferred (50_{10}).

1. The common initialization routine sets up the control registers on the DCPC.

2. Clear Control (CLC sc) to the interface assures that the Device Command can be asserted and also serves as a method to "idle" the device.

3/4. Setting the Flag flip-flop (STF sc) on the interface asserts the SRQ signal to initiate the transfer of the first word. At this time, DCPC has not been enabled by the appropriate instruction, thus SRQ remains active but does not cause an official DCPC cycle to take place. On an output transfer, the initial activation of SRQ through software is the only way to signal DCPC that an I/O device is prepared to receive data. If the STC sc is used, as for input transfers, it may cause the I/O device to accept erroneous data which is resident on the I/O interface's output registers at the time.

5. Setting the Control flip-flop and clearing the Flag flip-flop (STC sc,C) on the DCPC channel activates the beginning of the output transfer from memory to the I/O device. This allows the previous asserted SRQ to be sampled by the DCPC, then the first DCPC output cycle is requested and started.

6. The DCPC hardware initiates a STC sc,C to the interface near the beginning of the DCPC cycle. Since the "timing/party line" flip-flop is being used on the interface, the Device Command is delayed until the end of the cycle.

7/8. The data word from memory is latched into the I/O interface output registers and allowed to stabilize on the input registers of the I/O device. This takes place during the middle of the DCPC cycle.

9. The Device Command is issued by the combination of the STC sc,C issued earlier, but preserved on the Control flip-flop output which provides the input for the timing/party line flip-flop. The end of the DCPC cycle is used to clock the "timing/party line" flip-flop; the resulting signal is the Device Command.

10/11. The Device Command (start) causes the device to latch or take the data that is present on its input lines and returns a Device Flag. The setting of the Flag flip-flop asserts SRQ (4), resulting in the request of another DCPC cycle. Steps 4, 6, 7, 8, 9, 10, and 11 are now repeated under the specific direction of the DCPC hardware until the entire block of data words has been transferred to the I/O device from memory. At this time, the DCPC channel Flag is set thus generating an interrupt (12) and program control is transferred to the user-written completion routine (13). The SFS or SFC instructions can also be used to test the DCPC channel Flag (select codes 6 or 7) for transfer completion.

3-29. OUTPUT TRANSFER INITIALIZATION — Method 2. The second method should be used when the I/O interface design does not permit the use of a timing/party line flip-flop. In this case the output data is still allowed to stabilize before the Device Command latches the data into the input register of the device, however, the initialization software code is changed.

This method includes the issuing of the data word prior to the Device Command. For example, the data that is latched into the I/O interface output register at DCPC cycle n is latched into the input registers of the I/O device in DCPC cycle n+1. Thus it takes two DCPC cycles to transfer one data word as shown in Figure 3-11. Note that after the initialization of DCPC, memory data word two is valid at the I/O interface during DCPC cycle one, however, the same data is not transferred to the device until the STC command at DCPC cycle two. This type of interleaving scheme allows for standard programming initialization of the DCPC Control Registers such that no loss of time or cycles occurs during the entire DCPC block transfer.

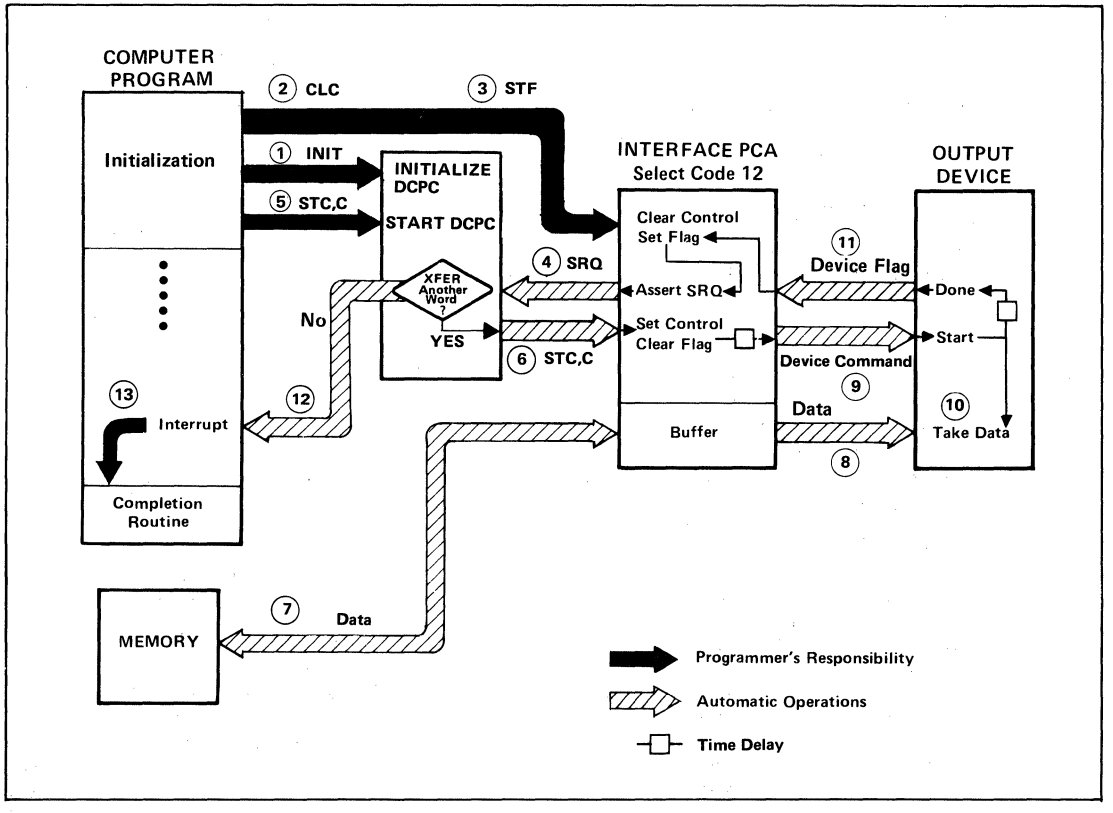
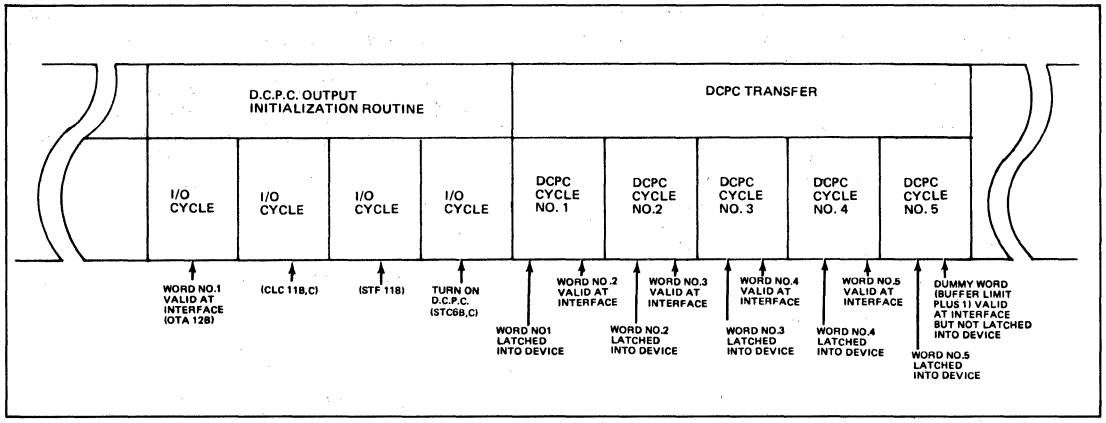


Figure 3-10. DCPC Output Data Transfer — Method 1



7700-497

Figure 3-11. Five Word DCPC Output Transfer — Method 2

The software initialization sequence for this method includes a programmed I/O output transfer (OTA sc) to the interface prior to turning on DCPC (Figure 3-11 and Table 3-6). This transfers the first word in the memory buffer to the I/O interface so that the STC command, at the beginning the first DCPC cycle, latches the first word into the device. On the last cycle of the transfer, the STC function is the only valid DCPC operation that occurs. The data word that is issued in the last cycle is not part of the specified memory buffer and is ignored, i.e., this word is never latched into the device since the DCPC word count is zero and the DCPC hardware has stopped further output transfer operations. Care should be taken to ensure that the DCPC control register is initialized to the second word in the block to be transferred since the first word is output via programmed I/O.

3-30. DCPC INITIALIZATION. The information required to initialize the DCPC (transfer direction, memory allocation, I/O channel assignment, and block length) is given by three control words. These three words must be addressed specifically to the DCPC. Figure 3-12 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

For DCPC output transfers, the clear control option of Control Word 1 (bit 13 = 1, Figure 3-12) is I/O-card dependent. A condition might exist such that the last data word is not received by the device because the device command is not issued. This can occur if the CLC signal overrides the STC signal (Control Word 1, bit 15 = 1), consequently eliminating device command for the last word transfer. The relationship between the STC signal, the CLC signal, and the device command determines whether the CLC option can be used.

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 1) or out of memory (logic 0). 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.

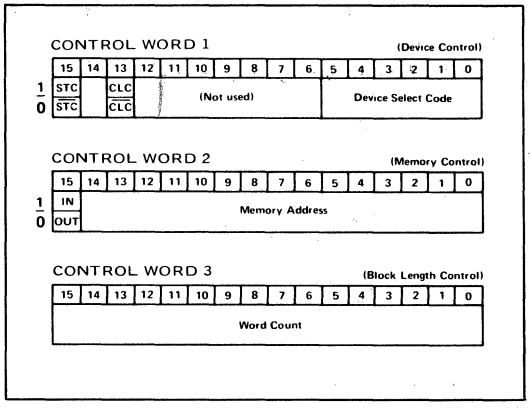


Figure 3-12. DCPC Control Word Formats

Table 3-6. DCPC Output Initialization Program — Method 2

LABEL	OPCODE	OPERAND	COMMENTS
ASGN2	LDA	CW1	Fetches control word 1 (CW1) from memory and loads it in A-register.
	OTA	7B	Outputs CW1 to DCPC Channel 2.
MAR2	CLC	3B	Prepares Memory Address Register to receive control word 2 (CW2).
	LDA	CW2	Fetches CW2 from memory and loads it in A-register.
	OTA	3B	Outputs CW2 to DCPC Channel 2.
WCR2	STC	3B	Prepares Word Count Register to receive control word 3 (CW3).
	LDA	CW3	Fetches CW3 from memory and loads it in A-register.
	OTA	3B	Outputs CW3 to DCPC Channel 2.
OUTPT	LDA	BUF1	Load A-register with first word to be transferred.
	OTA	12B	Output first word to I/O interface.
	CLC	11B,C	Turn off the interface (idle the device).
	STF	11B	Assert SRQ to DCPC.
	STC	7B,C	Activate DCPC Channel 2.
	SFS JMP	7B *−1	Wait while data transfer takes place or, if interrupt processing is used, continue program.
.	.	.	
.	.	.	
.	.	.	
	HLT		Halt
CW1	OCT	100011	Assignment for DCPC Channel 2 (ASGN2); specifies I/O channel select code address (11_8), STC after each word is transferred, and CLC after final word is transferred.
CW2	OCT	000200	Memory Address Register control. DCPC Channel 2 (MAR2); specifies memory output operation and starting memory address (200_8).
CW3	DEC	−50	Word Count Register control. DCPC Channel 2 (WCR2); specifies the 2's complement of the number of character words in the block of data to be transferred (50_{10}).

Table 3-4 contains the basic program sequence for outputting the control words to the DCPC. As shown in Table 3-4, 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 words 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. One important difference should be noted when doing a DCPC input operation from a disc or a drum. Due to the synchronous 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 3-5 illustrates a DCPC output transfer. Select codes 3 and 7 are used in conjunction with a channel 2 transfer. It should be noted that either channel can be used for input or output.

COMPUTER TIMING

SECTION IV

This section contains a general discussion of the HP 1000 Computers' timing schemes. Different timing schemes are employed by the HP 1000 M-Series and HP 1000 E/F-Series Computers and will, therefore, be discussed separately in the following paragraphs. Since timing for all computers is derived from a crystal-controlled oscillator in the control processor, a basic knowledge of control procesor timing is required to fully understand I/O Section timing. Timing cycles for the control processor are different from the timing cycles for other computer sections and these sections operate asynchronously until they must communicate with each other. Then, the control processor will inhibit (freeze) its operations until it is synchronized with the applicable computer section.

4-1. CONTROL PROCESSOR TIMING (HP 1000 M-SERIES)

A timing configuration diagram for the HP 1000 M-Series Computer is contained in Figure 4-1. As shown, control processor timing is derived from an 18.5-MHz crystal-controlled oscillator that clocks a three-stage ring counter approximately every 54 nanoseconds. The counter consists of a PA flip-flop (PA FF), PB flip-flop (PB FF), and PC flip-flop (PC FF) whose states are decoded to provide six 54-nanosecond periods designated P0, P1, P2, P3, P4, and P5 as shown in Figure 4-2. These six P-periods comprise one control processor cycle (one microcycle) and represent the time duration (325 nanoseconds) required by the computer to execute one microinstruction. Some decoded timing signals are continuous-running and others can be frozen by the control processor Freeze flip-flop. (The timing signals that can be frozen are designated in Figure 4-2.) If the execution of a microinstruction is dependent on synchronization between the control processor and some other computer section, the control processor will freeze certain clock signals to prevent execution of the microinstruction until the required synchronization has been completed. A freeze inhibits designated clock signals from the end of one P2 period to the end of the next P2 period. (Only one freeze signal can be issued per microcycle.) A freeze signal performs the following functions:

a. Prevents alteration of Arithmetic/Logic Section registers.

b. Prevents alteration of Read-Only-Memory (ROM) Instruction Register or ROM Address Register.

c. Prevents loading of Central Interrupt Register.

d. Prevents alteration of Overflow, Extend, and Flag flip-flops.

e. Isolates the control processor from the S-Bus.

f. Prevents the control processor from sending Read or Write signals to the Memory Section.

g. Prevents obtaining of data from the Memory Section.

h. Prevents initiation of an I/O cycle.

Although there are various conditions that determine if a freeze is required, only those conditions that affect the I/O Section will be discussed. There are two conditions when synchronous operation between the I/O Section and the control processor is required: (1) to clock the select code of the interrupting device into the Central Interrupt Register in order to issue an Interrupt Acknowledge (IAK) signal to the I/O interface PCA and (2) when the computer is ready to execute an I/O instruction. To issue an IAK signal, an internal Central Interrupt Register Enable (CIREN) signal is generated which initiates the freeze condition, clocks the Central Interrupt Register, and causes the generation of the IAK signal. The trailing edge of T6 from the I/O Section removes the freeze condition and terminates the IAK signal. Prior to the execution of an I/O instruction, an internal I/O Group Special (IOGSP) signal is generated which initiates the freeze condition. The trailing edge of T2 from the I/O Section terminates this freeze condition. Figure 4-2 illustrates the effect of a freeze condition on control processor clock generation.

4-2. CONTROL PROCESSOR TIMING (HP 1000 E- AND F-SERIES)

A timing configuration diagram for the HP 1000 E/F-Series Computer is shown in Figure 4-3. As shown, control processor timing is derived from a 28.5-MHz crystal-controlled oscillator that clocks a three-stage Gray counter every 35 nanoseconds. The counter is decoded from a PA FF, PB FF, and PC FF to provide either five or eight 35-nanosecond periods designated P1, P2, P3, E1, E2, E3, P4, and P5 as shown in Figure 4-4. These periods comprise one microcycle and represent the time duration (175 or 280 nanoseconds) required by the computer to execute one microinstruction. The HP 1000 E/F-Series Computers make use of variable-length microcycles and, because the I/O Section T-periods are also variable between non-IOG cycles, no attempt should be made to use I/O Section backplane signals as basic clocks.

The shortest time duration required to execute a microinstruction is 175 nanoseconds and is termed a short microcycle. (The short microcycle is the time duration for

which the Arithmetic/Logic Section is designed to operate.) When the SHORT signal (Figure 4-3) is low, the three-stage Gray code counter and decoder generates five 35-nanosecond periods designated P1, P2, P3, P4, and P5.

Since the HP 1000 E/F-Series Computer is user microprogrammable and since certain I/O interface PCA's may not be able to function properly with a control processor cycle time of less than 190 nanoseconds, during the execution of an I/O instruction, the control processor cycle time is extended to a duration of 280 nanoseconds which is termed a long microcycle. Control memory has an access time of approximately 140 nanoseconds worst case and the Control Section's Memory Address register is loaded only at the control processor cycle time period P3. Therefore, if a branch microinstruction is to be executed, only two control processor cycle time periods (P4 and P5) would remain to access memory during a short microcycle which is an insufficient amount of time. Hence, whenever a branch microinstruction is to be executed, an internal MICRO-BRANCH signal (Figure 4-3) is generated which in turn, at time P3, generates a high SHORT signal. When the SHORT signal is high, the three-stage Gray counter and decoder generates eight 35-nanosecond periods. The 35-nanosecond extend periods are designated E1, E2, and E3. Once the branch microinstruction is executed, the MICRO-BRANCH signal is terminated, the SHORT signal goes low, and the control processor cycle returns to the five-period cycle of P1 through P5. To ensure that all I/O interface PCA's have sufficient time to function properly, a high I/O Group Enable flip-flop (IOGEN FF) signal is generated during I/O operations to set the SHORT signal

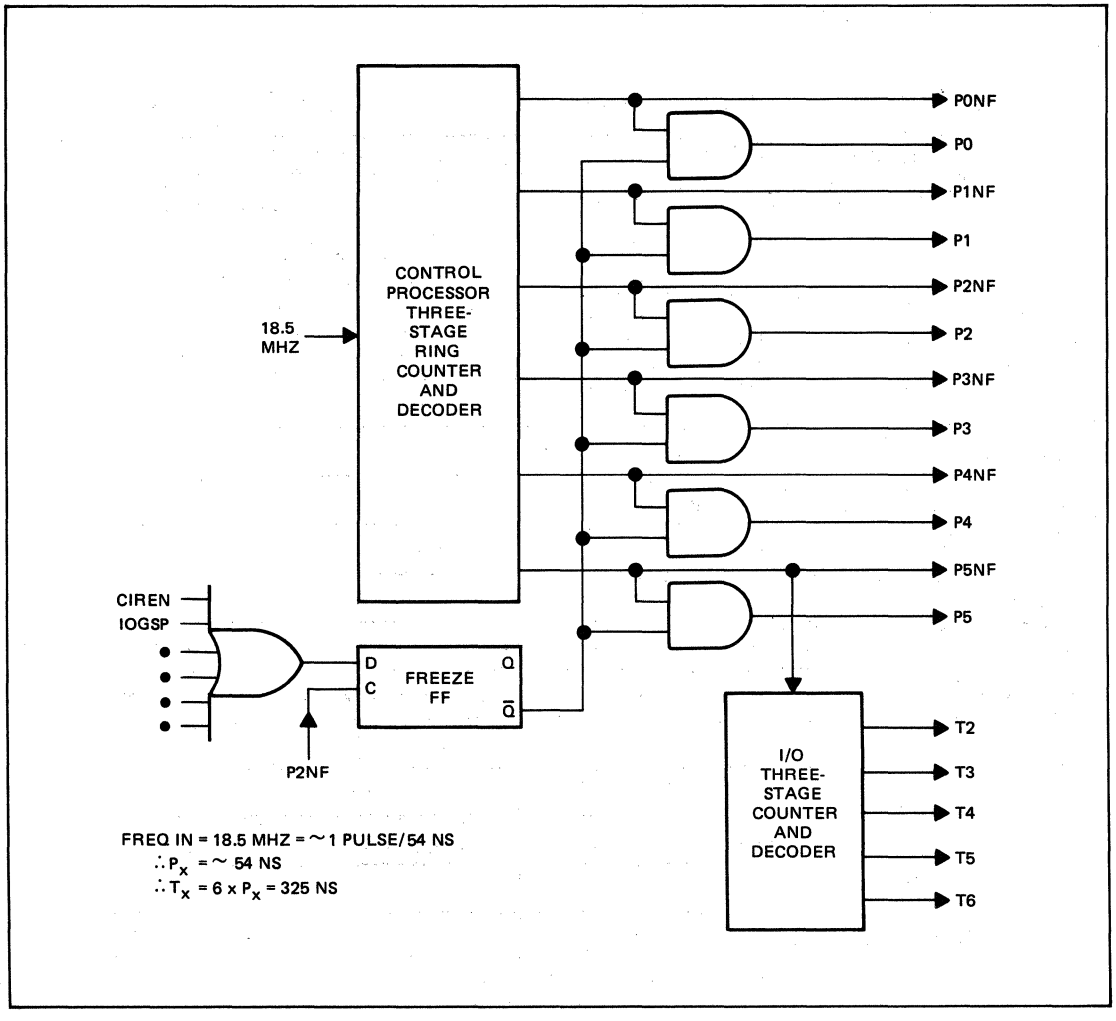


Figure 4-1. HP 1000 M-Series Computer Timing Configuration

high which, in turn, causes the three-stage Gray counter and decoder to divide by eight and produce a long microcycle of 280 nanoseconds (P1, P2, P3, E1, E2, E3, P4, and P5).

Like the HP 1000 M-Series Computers, the HP 1000 E/F-Series Computer control processor will freeze certain clock signals to prevent execution of a microinstruction until the required synchronization between the control processor and the applicable computer section has been accomplished. A freeze inhibits designated clock signals from the end of one P1 period to the end of the next P1 period. Figure 4-4 illustrates the effect of a freeze condition on control processor clock generation. Only one freeze condition can be issued per microcycle. A freeze signal performs the functions listed in paragraph 4-1(a) through 4-1(h). Although various conditions determine if a freeze is required, only those conditions that affect the I/O Section will be discussed. There are two conditions when synchronous operation between the control processor and the I/O Section is required: (1) to clock the Central Interrupt Register in order to issue an IAK signal to an I/O interface PCA and (2) when the computer is ready to execute an I/O instruction. To issue an IAK signal, an internal Interrupt Acknowledge Special (IAKSP) signal is generated which initiates the freeze condition (Figure 4-3), clocks the Central Interrupt Register, and causes the generation of the IAK signal. The trailing edge of T6 from the I/O Section removes the freeze condition and terminates the IAK signal. Prior to the execution of an I/O instruction, an internal IOGSP signal is generated which initiates the freeze condition. The trailing edge of T2 from the I/O Section terminates this freeze condition.

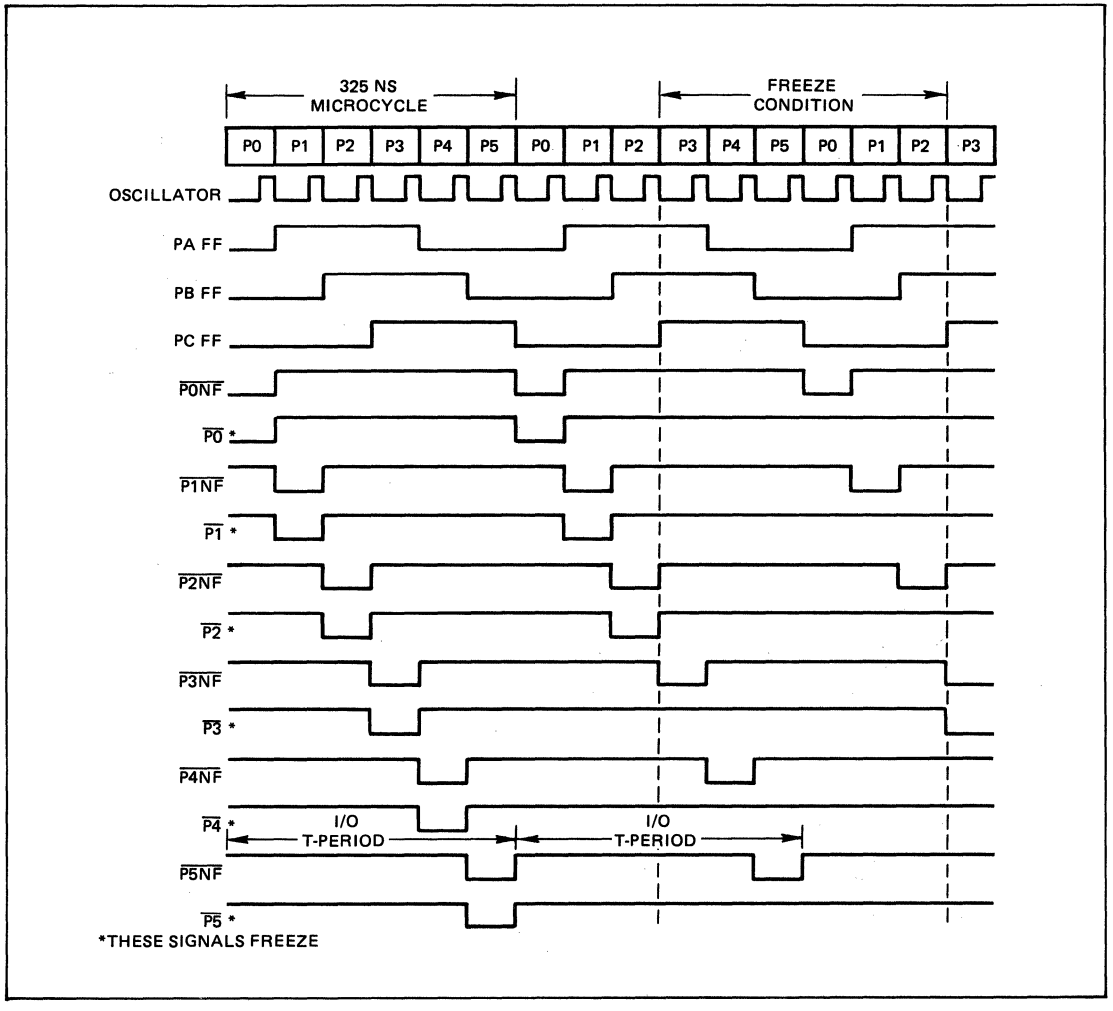


Figure 4-2. HP 1000 M-Series Control Processor Timing Diagram

In addition to the freeze condition, the HP 1000 E/F-Series Computers also employ a "pause" feature to suspend control processor timing. This feature permits asynchronous interface operations with memory. When the PAUSE signal (Figure 4-3) is high, the three-stage Gray counter and decoder operates as previously discussed. Whenever the PAUSE signal is low, however, a pause condition occurs and the microcycle is suspended at P3 period until the PAUSE signal again goes high. For example, if memory is busy (MBUSY signal high) and either the CPU or DCPC requests another memory operation, the PAUSE signal will go low at the next P3 period and the microcycle will be suspended at P3 until the memory is no longer busy and the MBUSY signal goes low. When MBUSY goes low, the PAUSE signal goes high and the microcycle will advance to either P4 or E1 period depending on the logic level of the SHORT signal as previously discussed. Therefore, as shown in Figure 4-4, a long microcycle duration can randomly vary from 280 to 630 nanoseconds and cannot be predicted unless the precise state of the computer is known (i.e., memory cycle time, memory operation, etc.). (It should be noted that during the execution of an I/O instruction, internal computer design guarantees a long microcycle duration of 280 nanoseconds for I/O periods T3, T4, and T5.)

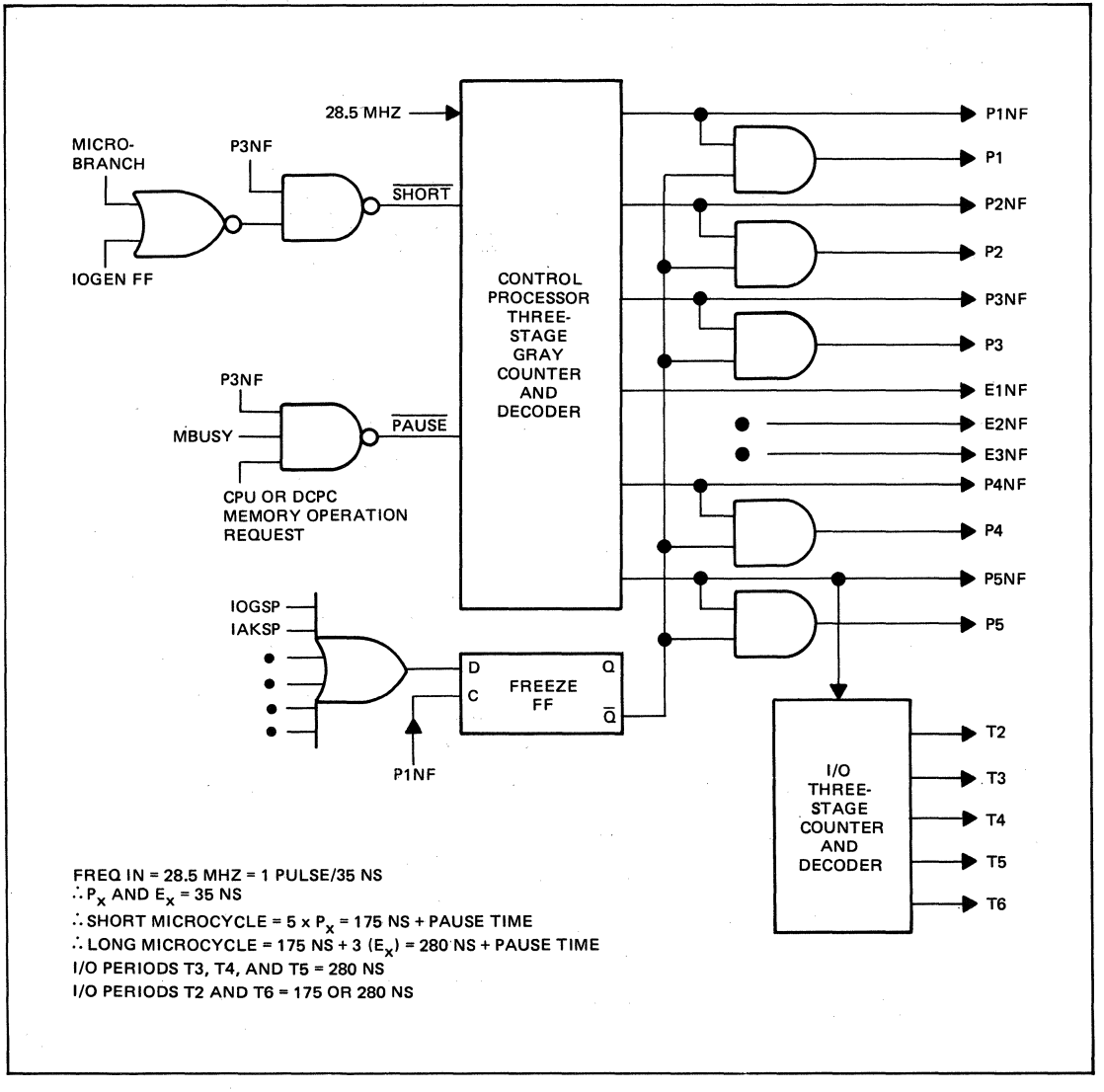
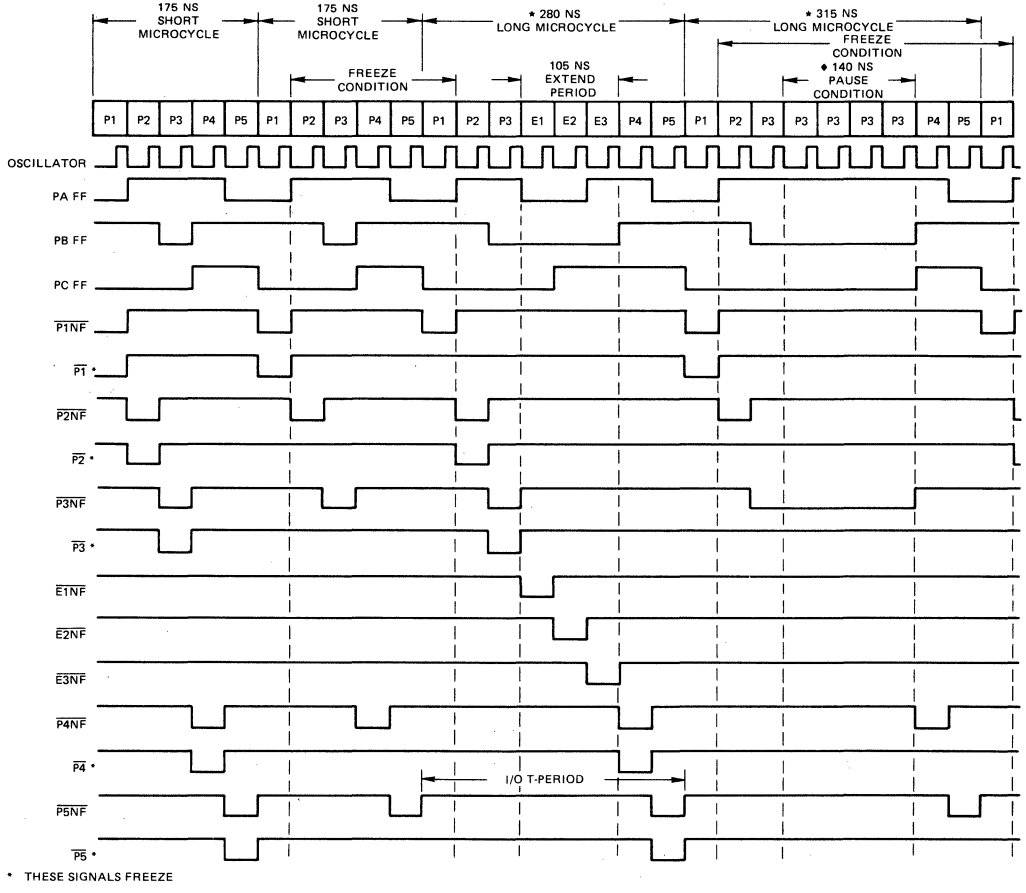


Figure 4-3. HP 1000 E/F-Series Computer Timing Configuration

4-3. I/O SECTION TIMING

As shown in Figures 4-1 through 4-4, I/O Section timing is derived from the control processor basic clock P5 non-freezable period (P5NF) that clocks the I/O three-stage counter and decoder. The counter is decoded from a TA flip-flop (TA FF), TB flip-flop (TB FF), and TC flip-flop (TC FF) to provide five T-periods designated T2, T3, T4, T5 and T6. These five T-periods comprise one I/O cycle and represent the required time to generate all the I/O signals that are required to execute an I/O instruction. For HP 1000 M-Series Computers, all T-periods are 325 nanoseconds and therefore, the duration of one I/O cycle is always 1.625 microseconds. For HP 1000 E/F-Series Computers, T-periods T3 through T5 are 280 nanoseconds each and T2 and T6 are either 175 or 280 nanoseconds. Therefore the duration of one E/F-Series I/O cycle is from 1.19 to 1.40 microseconds. Since all computer I/O cycles begin with T2 and end with T6, and since no I/O commands are generated at time T6 except Interrupt Acknowledge (IAK), the different I/O cycle time durations do not affect I/O compatibility between the HP 1000 M-Series and HP 1000 E/F-Series Computers or with existing HP 2100 Series interfaces. When an I/O cycle occurs, the various I/O signals are generated at the T-period times illustrated in Figure 4-5. Table 4-1 briefly defines all I/O signal

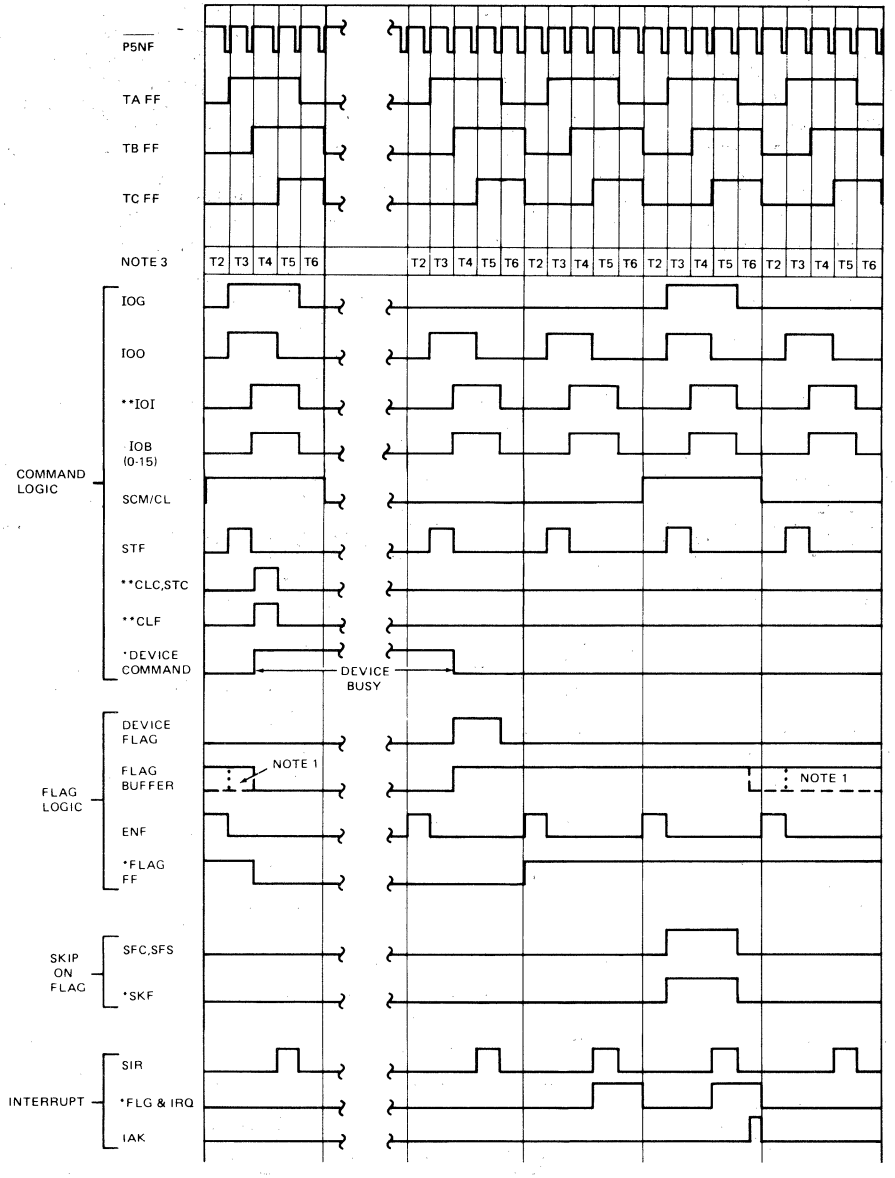


* THESE SIGNALS FREEZE

♦ VARIABLE FROM 35 – 455 NS/MICROCYCLE WITHOUT EXTEND PERIODS
VARIABLE FROM 35 – 280 NS/MICROCYCLE WITH MAXIMUM EXTEND PERIODS
* VARIABLE FROM 210 – 630 NS

0330-5

Figure 4-4. HP 1000 E/F-Series Control Processor Timing Diagram



NOTES:

1. BROKEN LINE FOR INTERRUPT OPERATION, SOLID LINE SKIP-ON-FLAG, AND DOTTED LINE FOR DCPC.
2. * INDICATES SIGNALS GENERATED BY INTERFACE PCA.
3. M-SERIES T-PERIODS=325 NS EACH; HP 1000 E/F-SERIES T3 THRU T5 =280 NS EACH, T2 AND T6=175 OR 280 NS.
4. ** INDICATES DIFFERENT T-PERIODS FOR DCPC. SEE FIGURE 4-8.

0330-6

Figure 4-5. I/O Section Timing Diagram

Table 4-1A. I/O Signal Definitions and Connector Pin Assignments (Numerical)

PIN NO. (ODD)	SIGNAL MNEMONIC AND DEFINITION		PIN NO. (EVEN)	SIGNAL MNEMONIC AND DEFINITION	
1	GND:	Ground	2	GND:	Ground
3	PRL:	Priority Low	4	FLGL:	Flag signal, Lower Select Code
5	SFC:	Skip if Flag is Clear	6	IRQL:	Interrupt Request, Lower Select Code
7	CLF:	Clear (reset) Flag flip-flop	8	IEN:	Interrupt Enable
9	STF:	Set Flag flip-flop	10	IAK:	Interrupt Acknowledge
11	T3:	I/O time period T3	12	SKF:	Skip on Flag
13	CRS:	Control Reset	14	SCM:	Select Code Most Significant Digit (Lower Address)
15	IOG:	I/O Group	16	SCL:	Select Code Least Significant Digit (Lower Address)
17	POPIO:	Power On Preset to I/O	18	IOB16:	I/O Bus input, bit 16 (M-Series only)
			18	BIOS:	"Not" Block I/O Strobe (E-Series only)
19	SRQ:	Service Request	20	IOO:	I/O data Output signal
21	CLC:	Clear (reset) Control flip-flop	22	STC:	Set Control flip-flop
23	PRH:	Priority High	24	IOI:	I/O data Input signal
25	SFS:	Skip if Flag is Set	26	IOB0:	I/O Bus input, bit 0
27	IOB8:	I/O Bus input, bit 8	28	IOB9:	I/O Bus input, bit 9
29	IOB1:	I/O Bus input, bit 1	30	IOB2:	I/O Bus input bit 2
31	IOB10:	I/O Bus input, bit 10	32	SIR:	Set Interrupt Request
33	IRQH:	Interrupt Request, Higher Select Code	34	SCL:	Select Code Least Significant Digit (Higher Address)
35	IOB0:	I/O Bus output, bit 0	36	+28V	
37	SCM:	Select Code Most Significant Digit (Higher Address)	38	IOB1:	I/O Bus output, bit 1
39	+5V		40	+5V	
41	IOB2:	I/O Bus output, bit 2	42	IOB4:	I/O Bus output, bit 4
43	+12V		44	+12V	
45	IOB3:	I/O Bus output, bit 3	46	ENF:	Enable Flag
47	−2V		48	−2V	
49	FLGH:	Flag signal, Higher Select Code	50	RUN:	Run
51	IOB5:	I/O Bus output, bit 5	52	IOB7:	I/O Bus output, bit 7
53	IOB6:	I/O Bus output, bit 6	54	IOB8:	I/O Bus output, bit 8
55	IOB11:	I/O Bus output, bit 11	56	IOB9:	I/O Bus output, bit 9
57	IOB12:	I/O Bus output, bit 12	58	IOB10:	I/O Bus output, bit 10
59	Not Used		60	IOB11:	I/O Bus input, bit 11
61	IOB13:	I/O Bus output, bit 13	62	EDT:	End Data Transfer (DCPC)
63	Not Used		64	IOB3:	I/O Bus input, bit 3
65	IOB14:	I/O Bus output, bit 14	66	PON:	Power On Normal
67	Not Used	(M-Series only)	68	Not Used	
67	BIOO:	"Not" Block I/O Output (E-Series only)			
69	−12V		70	−12V	
71	Not Used		72	Not Used	
73	SFSB:	Skip if Flag is Set Buffered (M-Series only)	74	IOB15:	I/O Bus output, bit 15
73	BIOI:	"Not" Block I/O Input (E-Series only)			
75	Not Used		76	Not Used	
77	IOB4:	I/O Bus input, bit 4	78	IOB12:	I/O Bus input, bit 12
79	IOB13:	I/O Bus input, bit 13	80	IOB5:	I/O Bus input, bit 5

Table 4-1A. I/O Signal Definitions and Connector Pin Assignments (Numerical) (Continued)

PIN NO. (ODD)	SIGNAL MNEMONIC AND DEFINITION	PIN NO. (EVEN)	SIGNAL MNEMONIC AND DEFINITION
81	IOB6: I/O Bus input, bit 6	82	IOB14: I/O Bus input, bit 14
83	IOB15: I/O Bus input, bit 15	84	IOB7: I/O Bus input, bit 7
85	GND: Ground	86	GND: Ground

Notes:
1. The following pins are connected together in pairs on each I/O backplane connector: 1 and 2; 39 and 40; 43 and 44; 47 and 48; 69 and 70; and 85 and 86.
2. Corresponding IOB bit lines are connected together on each I/O backplane connector (i.e., pins 26 and 35 are connected together, pins 29 and 38 are connected together, etc.).
3. Refer to Section VI of this manual for additional information on the block I/O signals on connector pins 18, 67, and 73.
4. Refer to Appendix B of this manual for more detailed signal descriptions.

Table 4-1B. I/O Signal Pin Assignments (Alphabetical)

PIN NO.	SIGNAL MNEMONIC AND DEFINITION	PIN NO.	SIGNAL MNEMONIC AND DEFINITION
67	BIOI: "Not" Block I/O Input (HP 1000 E- and F-Series only)	35	IOB0: I/O Bus Output, bit 0
		38	IOB1: I/O Bus Output, bit 1
73	BIOO: "Not" Block I/O Output (HP 1000 E- and F-Series only)	41	IOB2: I/O Bus Output, bit 2
		45	IOB3: I/O Bus Output, bit 3
18	BIOS: "Not" Block I/O Strobe (HP 1000 E- and F-Series only)	42	IOB4: I/O Bus Output, bit 4
		51	IOB5: I/O Bus Output, bit 5
21	CLC: Clear (reset) Control flip-flop	53	IOB6: I/O Bus Output, bit 6
7	CLF: Clear (reset) Flag flip-flop	52	IOB7: I/O Bus Output, bit 7
13	CRS: Control Reset	54	IOB8: I/O Bus Output, bit 8
62	EDT: End Data Transfer	56	IOB9: I/O Bus Output, bit 9
46	ENF: Enable Flag	58	IOB10: I/O Bus Output, bit 10
49	FLGH: Flag Signal, Higher Select Code		
4	FLGL: Flag Signal, Lower Select Code	60	IOB11: I/O Bus Output, bit 11
10	IAK: Interrupt Acknowledge	57	IOB12: I/O Bus Output, bit 12
8	IEN: Interrupt Enable	61	IOB13: I/O Bus Output, bit 13
26	IOB0: I/O Bus Input, bit 0	65	IOB14: I/O Bus Output, bit 14
29	IOB1: I/O Bus Input, bit 1	74	IOB15: I/O Bus Output, bit 15
30	IOB2: I/O Bus Input, bit 2	15	IOG: I/O Group
64	IOB3: I/O Bus Input, bit 3	24	IOI: I/O Data Input Signal
77	IOB4: I/O Bus Input, bit 4	20	IOO: I/O Data Output Signal
80	IOB5: I/O Bus Input, bit 5	33	IRQH: Interrupt Request, Higher Select Code
81	IOB6: I/O Bus Input, bit 6		
84	IOB7: I/O Bus Input, bit 7	6	IRQL: Interrupt Request, Lower Select Code
27	IOB8: I/O Bus Input, bit 8		
28	IOB9: I/O Bus Input, bit 9	66	PON: Power On Normal
31	IOB10: I/O Bus Input, bit 10	17	POPIO: Power On Preset to I/O
60	IOB11: I/O Bus Input, bit 11	23	PRH: Priority High
78	IOB12: I/O Bus Input, bit 12	3	PRL: Priority Low
79	IOB13: I/O Bus Input, bit 13	50	RUN: Run
82	IOB14: I/O Bus Input, bit 14	16	SCL: Select Code Least Significant Digit (Lower Address)
83	IOB15: I/O Bus Input, bit 15		
18	IOB16: I/O Bus Input, bit 16 (HP 1000 M-Series only)	34	SCL: Select Code Least Significant Digit (Higher Address)

Table 4-1B. I/O Signal Pin Assignments (Alphabetical) (Continued)

PIN NO.	SIGNAL MNEMONIC AND DEFINITION	PIN NO.	SIGNAL MNEMONIC AND DEFINITION
14	SCM: Select Code Most Significant Digit (Lower Address)	9	STF: Set Flag flip-flop
		11	T3: I/O Time Period T3
37	SCM: Select Code Most Significant Digit (Higher Address)	1,2,85,86	GND: Ground
		69,70	−12V
5	SFC: Skip if Flag is Clear	47,48	−2V
25	SFS: Skip if Flag is Set	39,40	+5V
73	SFSB: Sip if Flag is Set Buffered	43,44	+12V
32	SIR: Set Interrupt Request	36	+28V
12	SKF: Skip on Flag	59,63,68,	
19	SRQ: Service Request	71,72,75,	Not Used
22	STC: Set Control flip-flop	76	

mnemonics and identifies I/O connector pin number assignments for each signal. A more detailed description of the I/O signals is contained in Appendix B of this manual. (It should be noted that any individual interface PCA may not necessarily use all the signals listed in Table 4-1.) Interface PCA designers should also note that three T-periods (T2, T3, and T5) are buffered directly onto the I/O backplane and are unconditionally generated on backplane connector pin numbers 46, 11, and 32 respectively, once every I/O cycle. Period T2, renamed Enable Flag (ENF), is used to set interface PCA Flag flip-flops synchronously to begin interrupt priority resolution which ensures that no flag is set in the middle of some other I/O operation. Flags are to be set only during period T2 prior to generating I/O control signals. Period T5, renamed Set Interrupt Request (SIR), is used to set the Interrupt Request flip-flop on the interface PCA with the highest priority that is ready to interrupt.

4-4. TYPICAL APPLICATION

As an aid toward a better understanding of how the I/O Section timing scheme works, a typical application using the HP 12597A 8-Bit Duplex Register Interface PCA will be discussed in the following paragraphs. (A more detailed discussion of interface PCA basic element requirements and how to design your own interface PCA is contained in Section V.) As shown in Figure 4-6, the 8-bit duplex register interface PCA contains both input and output buffer storage for up to eight bits of control information, command information, or data. It also contains control logic to provide start and/or stop commands to the I/O device and flag logic to signal the computer when the I/O device is ready to perform its function.

4-5. SAMPLE PROGRAMS

Tables 4-2 and 4-3 contain sample programs that illustrate methods of programming the interface PCA using assembly language subroutines. (To satisfy the addressing requirements discussed in Section III, the interface PCA is arbitrarily assigned the select code of 15 octal.) The sample programs are designed only to exercise the interface PCA functions and do not apply to any specific I/O device. Table 4-2 provides sample programs for input and output operations. Table 4-3 provides a sample program for a combined input and output operation. (The use of a control word to the device and a status word from the device in Table 4-3 is for example purposes only. Data can be substituted in place of the control and/or status words without changing the programming techniques.)

4-6. FUNCTIONAL OPERATION

A flowchart illustrating the functional operation of the interface PCA during a typical I/O operation is contained in Figure 4-7. The programmed instructions shown on the flowchart are basically the same as those used in the combined I/O sample program contained in Table 4-3. The sequence of events illustrated in Figure 4-7 is as follows:

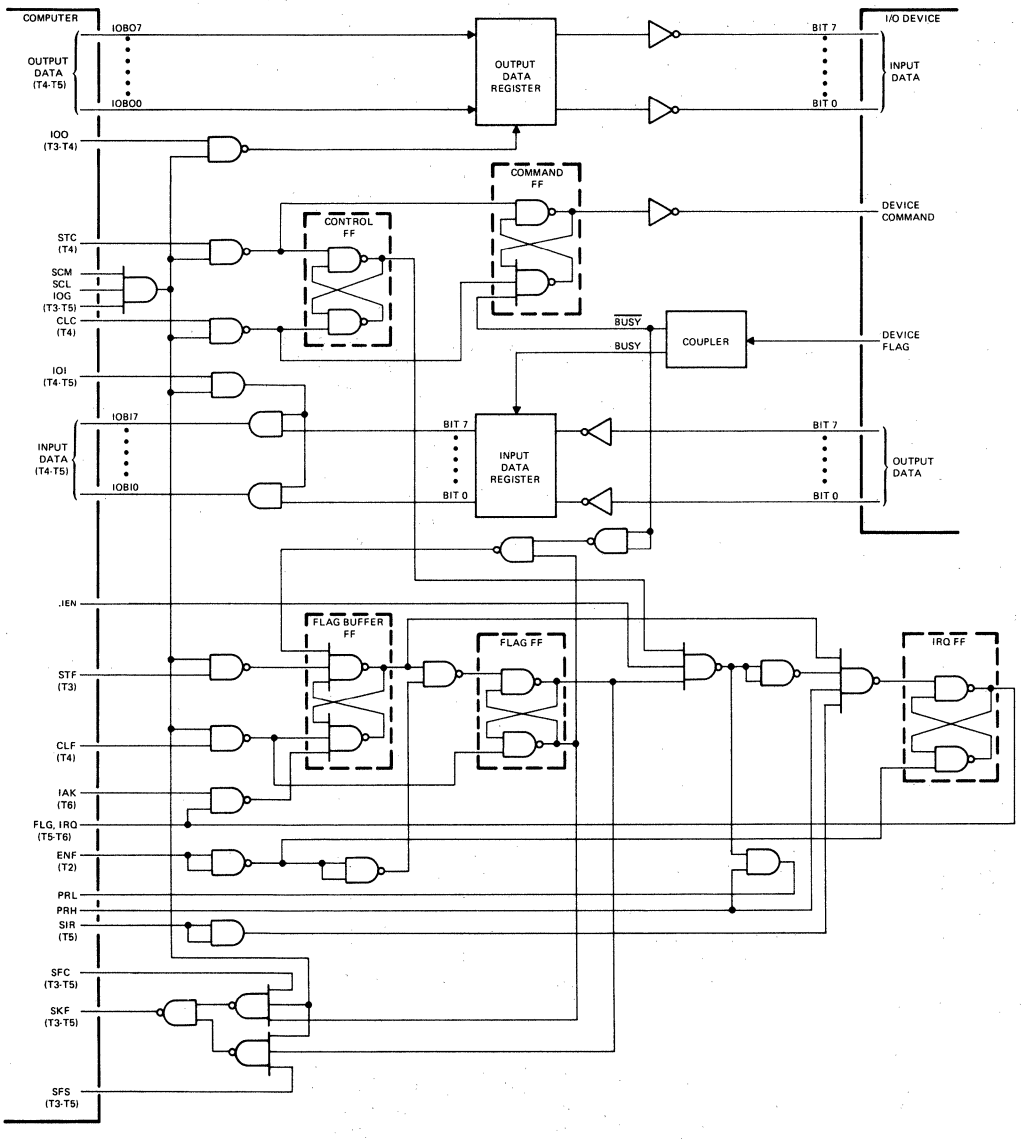
a. Data or control information is transferred from the computer's A-register to the interface PCA's output data register.

b. The I/O device is commanded to accept the data or control information and perform its function.

c. The computer waits for the I/O device to complete its operation.

d. The I/O device transfers a status word or data to the interface PCA's input data register and signals that its operation is complete.

e. The status word or data is transferred from the interface PCA's input data register to the computer's B-register.



0330-7

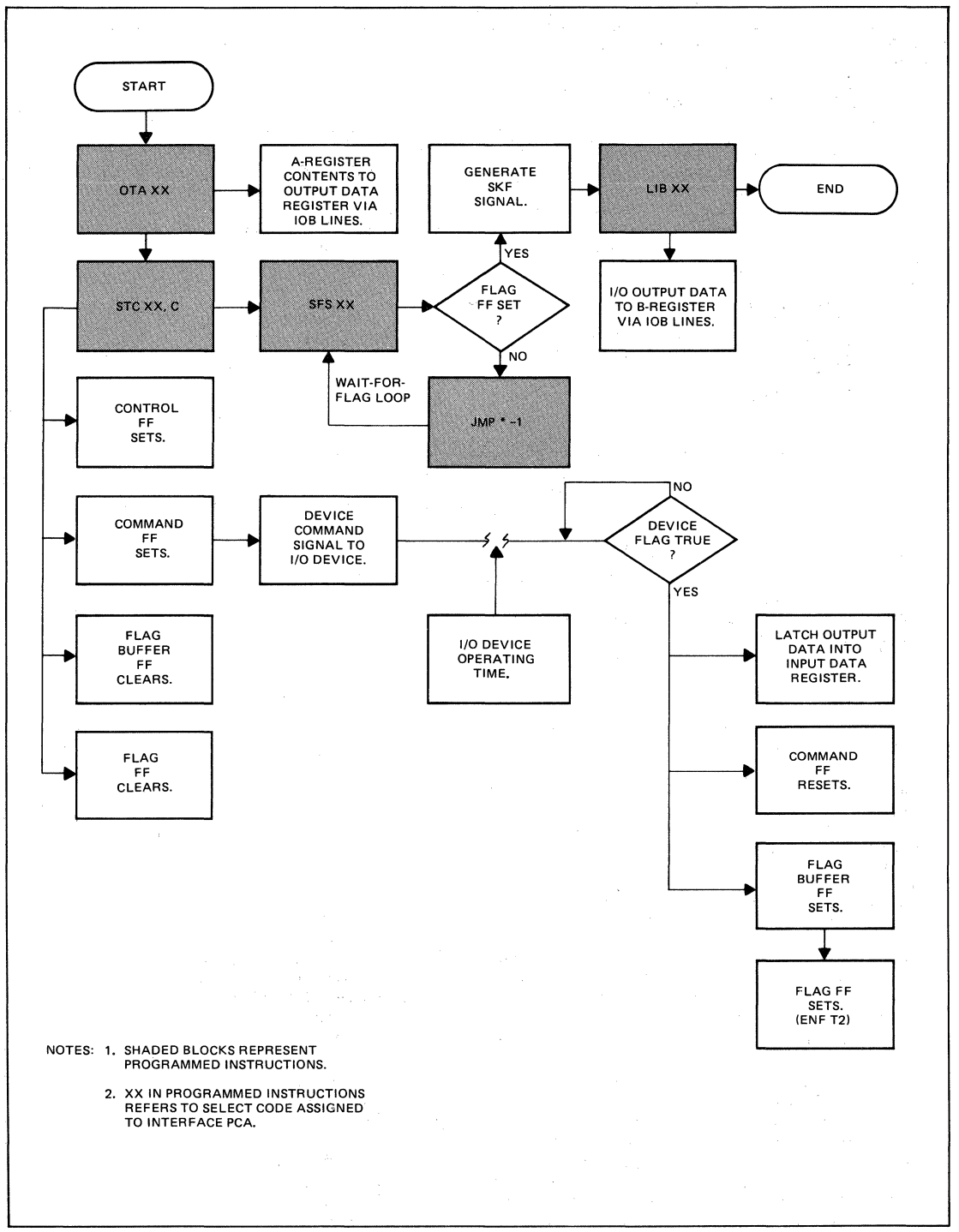
Figure 4-6. Duplex Register Interface PCA Simplified Logic Diagram

Table 4-2. Sample Input and Output Programs

```
0001                    ASMB,A,B,L,T
0002*
0003  00100                    ORG 100B
0004  00100 000000  START NOP
0005*
0006*  THE FOLLOWING ROUTINES ARE SAMPLES TO SHOW THE OPERATION OF THE
0007*  8-BIT DUPLEX REGISTER GENERAL PURPOSE INTERFACE.  THE INTERFACE
0008*  HAS BEEN ARBITRARILY ASSIGNED A SELECT CODE OF 15 OCTAL.
0009*
0010*       INPUT ROUTINE    THIS ROUTINE WILL START THE DEVICE, WAIT
0011*                        FOR THE DEVICE TO SUPPLY ONE 8-BIT WORD,
0012*                        AND PUT THAT WORD INTO THE COMPUTER'S
0013*                        A-REGISTER.
0014*
0015  00101 000000  INPT  NOP          ENTRY POINT.
0016  00102 103715       STC 15B,C    START DEVICE AND ENABLE FLAG LOGIC.
0017  00103 102315       SFS 15B      HAS DEVICE SUPPLIED A WORD?
0018  00104 024103       JMP *-1      NO. WAIT.
0019  00105 102515       LIA 15B      YES. PUT WORD IN A-REGISTER.
0020  00106 124101       JMP INPT,I   EXIT.
0021*
0022*
0023*       OUTPUT ROUTINE   THIS ROUTINE WILL WAIT FOR THE DEVICE TO
0024*                        SIGNAL THAT IT IS NOT BUSY, TRANSFER AN
0025*                        8-BIT WORD FROM THE A-REGISTER TO THE
0026*                        DEVICE, AND START THE DEVICE.
0027*
0028  00107 000000  OUTPT NOP          ENTRY POINT.
0029  00110 102315       SFS 15B      IS DEVICE READY?
0030  00111 024110       JMP *-1      NO. WAIT.
0031  00112 102615       OTA 15B      YES. PUT WORD IN OUTPUT REGISTER.
0032  00113 103715       STC 15B,C    START DEVICE AND ENABLE FLAG LOGIC.
0033  00114 124107       JMP OUTPT,I  EXIT.
0034*
0035*
0036                     END START
**  NO ERRORS*
```

Table 4-3. Sample Combined I/O Programs

```
0001                    ASMB,A,B,L,T
0002*
0003  00100                    ORG 100B
0004  00100 000000  START NOP
0005*
0006*  THE FOLLOWING ROUTINE IS A SAMPLE TO SHOW COMBINED INPUT/OUTPUT
0007*  CAPABILITIES OF THE 8-BIT DUPLEX REGISTER INTERFACE.  THE
0008*  INTERFACE HAS BEEN ARBITRARILY ASSIGNED A SELECT CODE OF 15 OCTAL.
0009*
0010*       THIS ROUTINE WILL TRANSFER AN 8-BIT CONTROL WORD FROM THE
0011*       A-REGISTER TO THE DEVICE, START THE DEVICE, AND TRANSFER AN
0012*       8-BIT STATUS WORD FROM THE DEVICE TO THE B-REGISTER WHEN
0013*       THE DEVICE OPERATION IS COMPLETE.
0014*
0015*
0016  00101 000000  I/O   NOP          ENTRY POINT.
0017  00102 102615       OTA 15B      PUT CONTROL WORD IN OUTPUT REGISTER.
0018  00103 103715       STC 15B,C    START DEVICE AND ENABLE FLAG LOGIC.
0019  00104 102315       SFS 15B      IS DEVICE OPERATION COMPLETE?
0020  00105 024104       JMP *-1      NO. WAIT.
0021  00106 106515       LIB 15B      YES. PUT STATUS WORD IN B-REGISTER.
0022  00107 124101       JMP I/O,I    EXIT.
0023*
0024*
0025                     END START
**  NO ERRORS*
```



0330-8

Figure 4-7. Interface PCA Functional Operation Flowchart

4-7. INPUT OPERATIONS. A combined STC and CLF instruction (STC XX,C) from the computer addressed to the select code of the interface PCA initiates the input of an 8-bit data or status word from the I/O device. As a result of this instruction, the interface PCA receives the IOG, SCM, SCL, STC, and CLF signals at the times specified in Figures 4-5 and 4-6. As shown in Figure 4-6, the STC signal sets the Command flip-flop which applies a Device Command (start) signal to the I/O device to initiate its input operation. Simultaneously, the CLF signal resets the Flag flip-flop to prevent an interrupt signal from being sent to the computer before the I/O device has transferred data to the interface PCA. (A detailed discussion of required flag and interrupt circuits is contained in Section V.)

When the I/O device is ready to transfer data to the interface PCA, it generates a Device Flag (done) signal which resets the Command flip-flop, sets the Flag flip-flop, and latches the eight bits of Output Data into the input data register. The timing of the flag logic at this point is dependent on the timing of the I/O device and is not related to the computer's I/O Section timing. The ENF signal generated by the computer every I/O cycle at time T2 is combined with an output signal from the interface PCA's Flag Buffer flip-flop to set the Flag flip-flop. This action synchronizes the effect of the Device Flag signal by allowing the Flag flip-flop to be set only at time T2. When the Flag flip-flop is set, the interface PCA flag logic generates the SKF signal that indicates to the computer that the I/O device has completed its operation and that the Output Data is in the input data register waiting to be transferred into the computer.

If the computer is programmed to wait for the Flag flip-flop to be set (e.g., an SFS instruction followed by a JMP* – 1 instruction), the resulting SFS signal gated with the set output of the Flag flip-flop generates the SKF signal as shown in Figure 4-6. It should be noted that the SKF signal can also be generated when the Flag flip-flop is reset by programming an SFC instruction. Either way, the state of the Flag flip-flop is monitored and the computer must be programmed accordingly.

If the computer interrupt system has been enabled by a programmed STF 00 instruction as listed in Table 4-4, the computer can be doing work in the program rather than waiting for the Flag flip-flop to be set. Then, when the I/O device completes its operation (Flag flip-flop set), the IEN signal is true, the Control flip-flop is set, and no device with a higher priority has requested an interrupt (the PRH signal is true), the interface PCA's IRQ flip-flop will be set at the following T5 time (SIR) which generates the FLG and IRQ signals. These signals are used by the computer to generate an interrupt request signal. After the interrupt is initiated, the next T2 time (ENF) resets the IRQ flip-flop and, if the PRH signal is still true, the following T5 time (SIR) sets it again. This time the resulting FLG and IRQ signals are used by the computer to encode the interrupt address. The interrupt address is stored in the memory address register and the IAK signal resets the interface PCA Flag Buffer flip-flop. The ENF signal resets the IRQ flip-flop. The Flag flip-flop remains set to inhibit lower priority interrupts by providing a false PRL signal. The next I/O cycle is controlled by the instruction stored at the interrupt location in computer memory. A CLF instruction must be programmed to reset the Flag flip-flop and enable lower priority interrupts just before leaving the interrupt subroutine.

As previously discussed, the Device Flag signal latches the Output Data from the I/O device into the input data register and the interface PCA logic generates the required signals to indicate to the computer that the information is waiting to be transferred. The computer will now accept the data from the input data register by outputting a programmed LIA, LIB, MIA, or MIB instruction addressed to the select code of the interface PCA. As a result of any one of these instructions, the IOG, SCM, and SCL signals are again applied to the interface PCA along with the IOI signal. The IOI signal gates the contents of the input data register onto data lines IOB0 through IOB7 and into the computer via the I/O bus. This completes one input operation with the 8-bit data or status word supplied by the I/O device now stored in the A- or B-register.

4-8. OUTPUT OPERATIONS. Output operations are essentially the same as input operations as far as the interface PCA is concerned. The primary differences are in the sequence of events and the use of the output data register instead of the input data register. Output operations require that the Flag flip-flop first be checked to ensure that the I/O device is not busy from some previous operation. The Flag flip-flop can be monitored by the SKF signal (interrupt system not enabled), or by the FLG and IRQ signals (interrupt system enabled) in the same manner as during input operations. If the I/O device is busy, the output operation must wait until the I/O device finishes and sets the Flag flip-flop as previously discussed.

After the computer has determined that the I/O device is not busy, the output operation can be initiated by either a programmed OTA or OTB instruction addressed to the select code of the interface PCA. As a result of either of these instructions, the IOG, SCM, and SCL signals are applied to the interface PCA along with the IOO signal which latches the 8-bit data or control word from data lines IOB0 through IOB7 into the output data register. The computer program must now issue a combined STC, CLF instruction (STC XX,C) to the interface PCA. The STC instruction sets the Command flip-flop which applies the Device Command signal to the I/O device indicating that data is available for transfer. The STC instruction also sets the Control flip-flop which provides the enabling signal for the interrupt control logic. The CLF instruction resets the Flag flip-flop to prevent an interrupt signal from being sent to the computer before the I/O device has accepted the data from the output data register and performed its operations. When the I/O device finishes, it returns the Device Flag signal to the interface PCA and sets the Flag flip-flop; the interface PCA then initiates an interrupt signal to the computer indicating that the I/O device is ready to accept additional information as previously discussed. This completes the output operation.

Table 4-4. Interrupt-Method Input Routine

INSTRUCTION	FUNCTION
STF 00	Enables interrupt system.
STC SC,C	Clears interface PCA's Flag flip-flop and starts I/O device operation.
JMP	Initiates jump to a different subroutine or program.
When a program interrupt occurs, a JSB,I instruction in the trap cell produces a program jump to the remainder of the I/O subroutine listed below.	
NOP	Entry point.
LIA SC	Transfers data from interface PCA into A-register.
STA XX	Transfers data from A-register into memory. (XX denotes assigned memory storage location.)

4-9. DUAL-CHANNEL PORT CONTROLLER (DCPC) TIMING

As discussed in Section III, the DCPC allows the user to initiate high-speed block word transfers between selected I/O devices and memory. The DCPC then controls the I/O device during the transfers, stealing memory and I/O cycles from the CPU, but not requiring CPU intervention until completion of the transfer. The DCPC is capable of stealing every consecutive I/O cycle. When the DCPC is operating, it takes priority over the CPU for both memory accesses and control of the I/O Section by generating all appropriate I/O signals. The CPU may not access memory or initiate an I/O cycle during a DCPC cycle. The DCPC data transfers are initiated by an initialization routine and then hardware controls the transfers automatically. No additional programming other than that discussed in Section III is required. Although the DCPC is designed to operate with I/O devices capable of handling high-speed data transmissions, it should be noted that it can be used with slower-speed devices if it is desirable to free these devices from program control.

DCPC timing is derived from the control processor crystal-controlled oscillator and is decoded into five T-periods designated T2, T3, T4, T5, and T6. These five T-periods comprise one DCPC cycle as shown in Figure 4-8. (For HP 1000 M-Series Computers, the duration of one DCPC cycle is 1.625 microseconds. For HP 1000 E/F-Series Computers, the duration of one DCPC cycle is typically from 0.875 to 1.16 microseconds.) Figure 4-8 provides a timing diagram for the DCPC during a DCPC cycle. The timing diagram shows both input and output control signals, but it should be noted that the operations require separate initialization routines as discussed in Section III. It should also be noted that some of the DCPC generated I/O instruction T-period times differ from standard I/O instruction times. (See Figure 4-5.) In addition to I/O instruction signals, Figure 4-8 also shows DCPC generated signals that prevent interference by the CPU and that help control data transfers when the DCPC is stealing I/O cycles. The definition and purpose of these signals are as follows:

SRQ: "Service Request". Used to notify computer that the I/O device is ready for a data transfer. Initiates DCPC cycle and prevents any further processing of programmed instructions by the computer until the DCPC cycle is complete.

DMAIOI: "Direct Memory Access I/O Input". Used to gate data on the I/O bus onto the S-bus during a DCPC input transfer.

IOG: "I/O Group". Used to enable DMA select code onto select code bus and indicates a "true" I/O instruction to the I/O interface.

IOI: "I/O Data Input". Used to gate data from the interface PCA onto the I/O bus during a DCPC input transfer.

DMALCH: "Direct Memory Access Latch". Used to hold data on the I/O bus through the completion of a DCPC output transfer. Latches the I/O bus onto itself.

DMAIOO: "Direct Memory Access I/O Output". Used to gate data on the S-bus onto the I/O bus during a DCPC output transfer.

IOO: "I/O Data Output". Used to gate data from the I/O bus to the interface PCA during a DCPC output transfer.

EDT: "End Data Transfer". Used to notify the I/O device that the number of words specified in the programmed block length have been transferred. Signifies the end of a DCPC transfer.

CLF: "Clear Flag". Used to clear (reset) interface PCA Flag flip-flop.

STC: "Set Control". Used to set interface PCA Control flip-flop.

CLC: "Clear Control". Used to clear (reset) interface PCA Control flip-flop.

SCM/SCL: "Select code most/Select code least". Used to determine the correct address of the I/O interface which communicates with the I/O device.

The DCPC cycle is initiated when the selected I/O device signals that it is ready for a data transfer with an SRQ signal (Figure 4-8) from its interface PCA at time T2. During input transfers, the IOI signal gates input data from the interface PCA onto the I/O bus and the DMAIOI signal transfers the data from the I/O bus onto the S-bus and into memory. At time T3, the CLF signal causes the SRQ signal to go false and the STC signal, if selected during initialization, restarts the I/O device for the next data transfer. The CLC signal, if selected during initialization, disables the I/O device at the end of the data block transfer. This completes the input DCPC cycle for the transfer of one block of data. During output transfers, the SRQ signal performs the same function previously discussed for input transfers. At time T3 during the DCPC cycle, the DMAIOO signal gates the data read from memory from the S-bus to the I/O bus. The DMALCH signal holds the data on the I/O bus until it is transferred to the interface PCA output data register by IOO. The CLF signal again causes the SRQ signal to go false and the STC signal, if selected during initialization, causes the I/O device to accept the data from the interface PCA. (Depending on I/O device characteristics and its associated interface PCA, the STC signal may or may not be required.) This completes the output DCPC cycle for one data word transfer. As previously discussed, the DCPC Word Count Register is incremented every DCPC cycle thereby effectively counting the number of words transferred into or out of memory. When the counted number of transferred words equals the number of words specified in the programmed block length, the Word Count Register generates a carry signal that initiates the DCPC interrupt logic which includes the generation of the EDT signal for the I/O device.

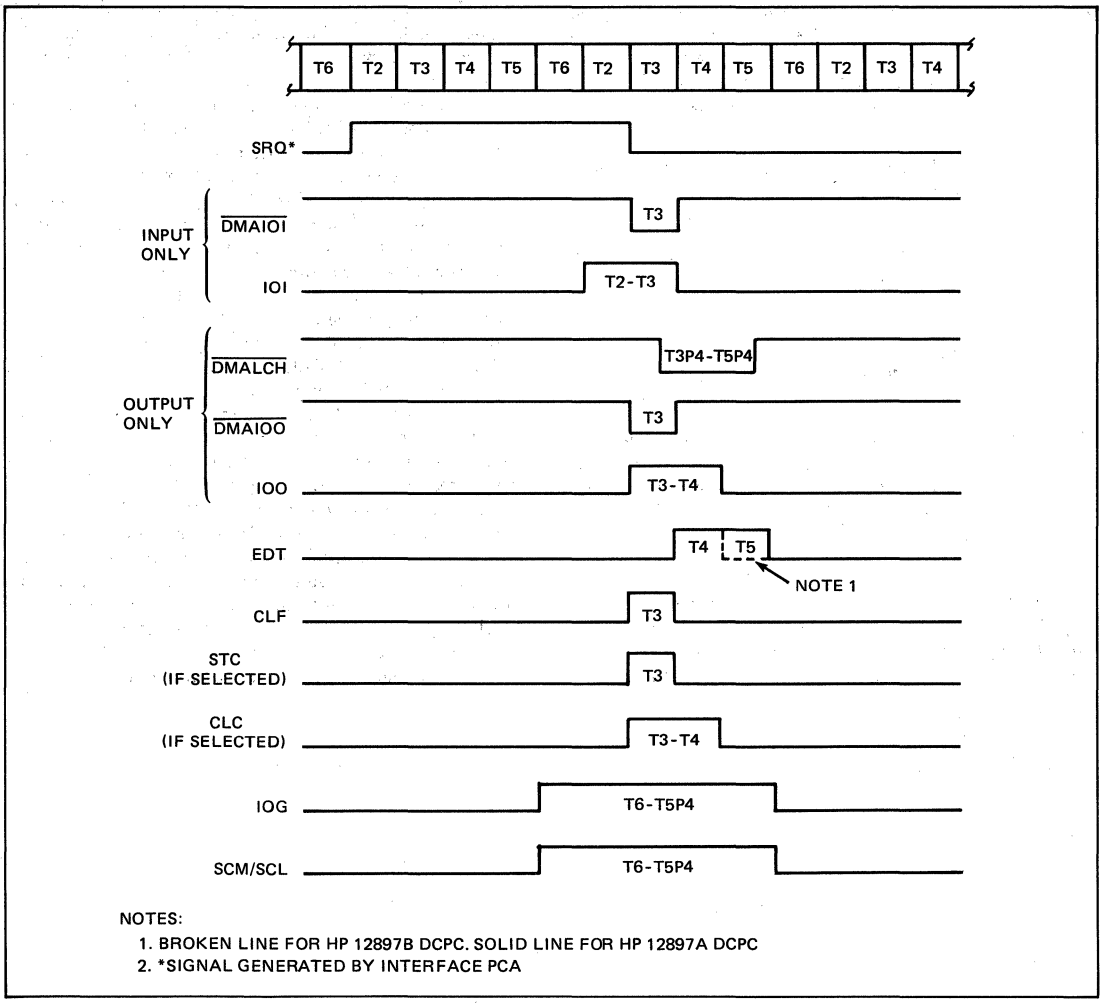


Figure 4-8. DCPC Timing Diagram

DESIGNING INTERFACE PCA'S

SECTION V

This section contains information for designing special purpose I/O interface PCA's. Unless otherwise specified, the contents of this section apply equally to the HP 1000 M-Series, E-Series and F-Series Computers.

5-1. INTRODUCTION

An I/O interface PCA must provide the circuits through which data can be transferred between the computer and an external I/O device. It must also provide the circuits required to control the I/O device from commands received from the computer. A typical interface PCA may contain as many as 16 buffers for temporary storage of data both to and from the I/O device. The number of buffers contained on a particular interface PCA depends on its associated I/O device. Some I/O devices require the capability of interrupting the computer program while for others, this capability is not necessary. Some I/O devices require control signals for the movement of tape, etc., and some require special timing signals. Some I/O devices require more than one interface PCA. There are many special cases in which unique types of controls or other criteria dictate the need to design and fabricate a special I/O interface PCA. Due to the very nature of special purpose interfacing, no detailed step-by-step procedures for the best design can be given. Only a study of the computer and I/O device mutual requirements can produce the ultimate design. Therefore, the information presented in this section should be used as guidelines around which to base your own interface design.

5-2. I/O SECTION INTERFACING

5-3. I/O INTERFACE PCA SPECIFICATIONS

The required dimension specifications for I/O interface PCA's are illustrated in Figure 5-1. The PCA shown in Figure 5-1 is a typical Hewlett-Packard interface PCA and is illustrated from the component side of the PCA. Unless otherwise specified, the dimensions shown in Figure 5-1 are symmetrical. The PCA thickness must be 0.059 ±0.006 inch (1.50 ±0.15 millimeters) although the computer's I/O backplane assembly connectors can accept a thickness up to 0.071 inch (1.80 millimeters). The center-to-center spacing of connector pins is 0.156 inch (3.96 millimeters).

One end of the interface PCA has 86 printed-circuit paths, 43 on each side of the PCA. This end of the PCA (usually designated P1) connects into the computer's I/O backplane assembly connector to transfer signals to and from the computer. The circuit path pin numbers on P1 correspond to the pin numbers on each of the I/O backplane assembly connectors. Odd-numbered pins 1 through 85 are on the component side of the PCA and even-numbered pins 2 through 86 are on the other side of the PCA. Consecutively-numbered pins are directly opposite each other on the PCA; e.g., pins 1 and 2 are on opposite sides of the PCA. Pin number assignments on this end of the PCA are identical for all I/O interface PCA's to permit the placement of any I/O interface PCA in any of the I/O backplane assembly connectors. A complete list of the signals assigned to the pin numbers on this end of the PCA is contained in Table 4-1. It should be noted that Table 4-1 lists all available pin assignments and that an individual interface PCA may not necessarily use all the signals listed.

The other end of the PCA shown in Figure 5-1 is usually designated J1 and typically has 48 printed-circuit paths, 24 on each side of the PCA. (The number of circuit paths for connector J1 is determined by the number of signal lines required for the I/O device.) The hood connector of the interconnecting cable between the interface PCA and its associated I/O device connects onto this end of the PCA. The circuit-path pin number positions on J1 correspond to the pin number positions on all standard HP interface cable hood connectors. Pins 1 through 24 are on the component side of the PCA and consecutively-lettered pins A through BB (letters G, I, O, and Q are omitted) are on the other side of the PCA. Pins 1 and A are on opposite sides of the PCA and pins 24 and BB are on opposite sides of the PCA. Pin assignments and signals between this end of the PCA and its I/O device are completely open to the discretion of the PCA's designer.

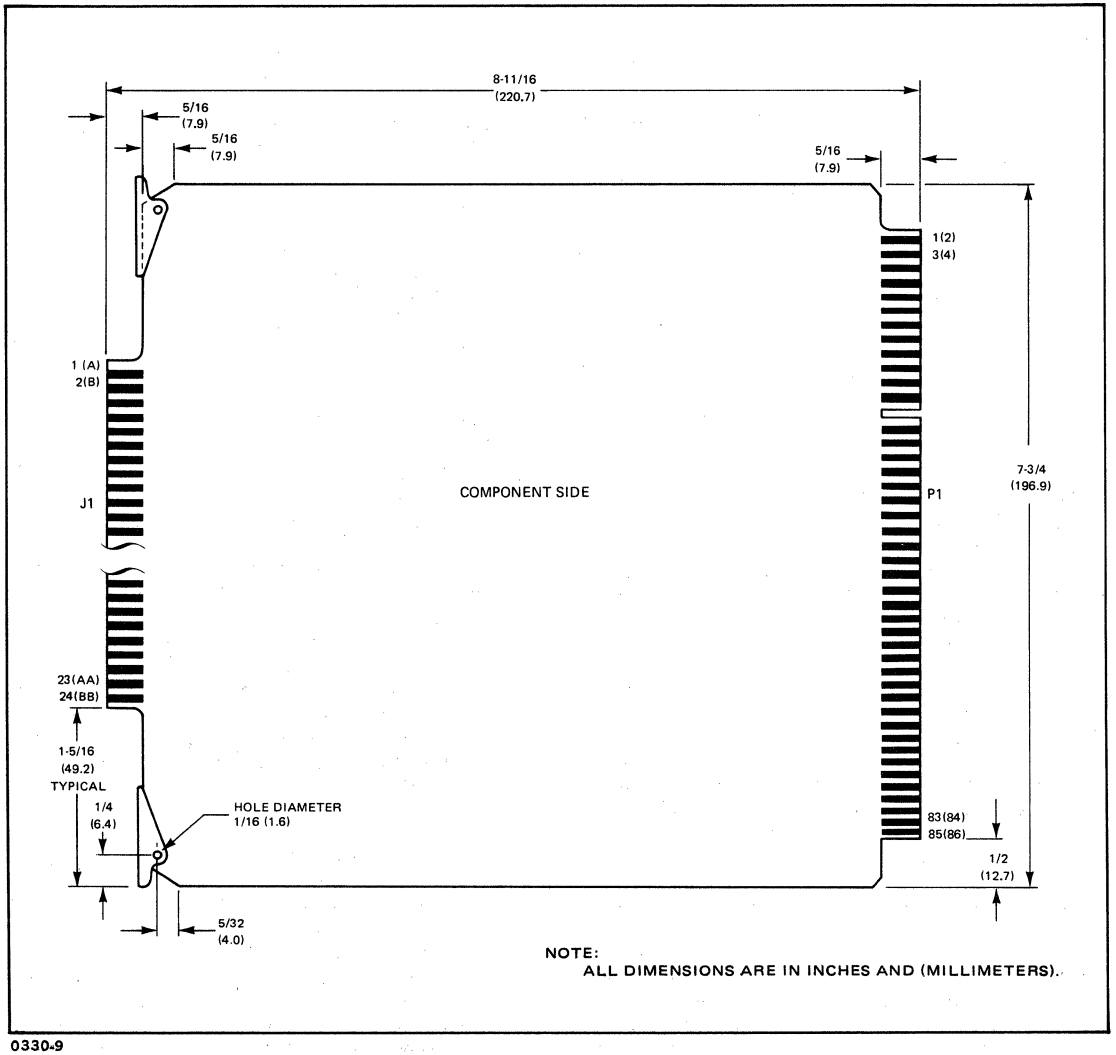
5-4. HP BREADBOARD INTERFACE KIT

To facilitate I/O interface PCA design, Hewlett-Packard can furnish a breadboard interface kit that includes a breadboard-type I/O interface PCA equipped with the TTL flag and interrupt circuits required to interface unique I/O devices with the HP 1000 Computers. The breadboard interface kit also includes a connector kit for fabricating the I/O device interface cable. The supplied interface PCA provides space for sixty 14-pin or 16-pin integrated circuit components, 11 of which are occupied by the flag and interrupt circuit components. The PCA also contains 12 test points (TP1 through TP12) to monitor the operation of the flag and interrupt circuits. The test points are defined in Table 5-1. The PCA circuit path pin numbers are compatible with the computer I/O backplane connector pin numbers listed in Table 4-1. (For additional information refer to the *HP 12620A Breadboard Interface Kit Operating and Service Manual,* part no. 12620-90001.)

A logic diagram of flag and interrupt circuits supplied on the HP 12620A Breadboard Interface Kit's interface PCA is contained in Figure 5-2. This diagram should be used as a guide when designing any interface PCA for an I/O device that requires flag and interrupt circuits. All integrated circuit components shown in Figure 5-2 are identified by reference designators (e.g., U25). Hewlett-Packard part numbers for these components and corresponding commercially available versions are listed in Table 5-2.

The following discussion describes the operational relationship between the flag and interrupt circuits illustrated in Figure 5-2 and the computer's I/O Section timing discussed in Section IV. The Flag Buffer flip-flop is set whenever a Device Flag signal from the I/O device is received at TP1 indicating that the device requires service. When set, the Flag Buffer flip-flop sets the Flag flip-flop when the ENF signal is received at time T2. With the Control flip-flop set by the STC signal from the computer program and the Flag flip-flop set, the Flag flip-flop disables the PRL signal to the lower priority (higher select code) devices at time T2. This prevents an interrupt by a lower priority I/O device.

If the priority at the interface is disabled by a device with higher priority via the PRH signal and if the interrupt



0330-9

Figure 5-1. I/O Interface PCA Dimensions

Table 5-1. Flag and Interrupt Circuit Test Point Definitions

TEST POINT	FUNCTION
TP1	Device Flag signal. Input signal is ground true. A ground sets the Flag Buffer flip-flop. Signal must remain true for at least 0.20 microsecond and must not exceed 2.5 microseconds.
TP2	ENF signal. Signal is positive true during time T2 and is used to gate Flag Buffer flip-flop output into Flag flip-flop.
TP3	SIR signal. Signal is positive true during time T5 and is used to enable inputs into IRQ flip-flop.
TP4	STC signal. Signal is ground true during a Set Control instruction addressed to the select code of the PCA.
TP5	CLC signal. Signal is ground true during a Clear Control instruction addressed to the select code of the PCA.
TP6	CRS signal. Signal is ground true at power turn-on, when computer front panel PRESET switch is pressed, or when a CLC 00 instruction is executed.
TP7	Flag Flip-Flop Set signal. Signal is positive true when Flag flip-flop is set.
TP8	Flag Flip-Flop Reset signal. Signal is positive true when Flag flip-flop is reset (clear).
TP9	Decoded Address signal. Signal is positive true when I/O instruction selects the PCA.
TP10 thru TP12	Signal ground for oscilloscope ground probe.

system is enabled (IEN signal), the interface has all of the correct conditions (Control, Flag Buffer, and Flag flip-flops are set) to create an interrupt to the CPU. The remaining qualifier is SIR signal at T5. The application of SIR causes the Interrupt Request (IRQ) flip-flop to be set. The FLG and IRQ interface lines (not to be confused with FLAG and IRQ flip-flop) become active as a result of the setting of the IRQ flip-flop. The CPU samples and records the FLG and IRQ line status during T5 thus setting up the proper flag conditions within the CPU control circuitry. (The only exception for the posting of the interrupt occurs during a DCPC cycle when the interrupt request from the interface would be ignored until the DCPC gives up control of memory and I/O.) The IRQ flip-flop is cleared at T2 thus remaining set for T5 and T6 only. As the CPU completes the currently executing assembly instruction or arrives at a conditional check for interrupts within the assembly instruction, the CPU Control recognizes the interrupt and issues an Interrupt Acknowledge (IAK) signal to the interface. The IAK will always be issued during the later part of T6. The minimum time for the interrupt to be recognized and acknowledged by the CPU Control is two I/O cycles as shown in Figure 4-5 in Section IV. If the interrupt is not recognized immediately due to a lengthy assembly instruction that is currently being executed, the IRQ flip-flop will continue to be set during T5/T6 by the SIR signal and cleared by each following T2 as shown in Figure 4-5. The IRQ flip-flop is cleared from T2 through T4 to provide a method of allowing an interface with a higher priority to activate its FLG and IRQ interrupt lines and be recorded within the CPU after SIR has set its IRQ flip-flop. In any case, after the IAK is sent, the CPU control is transferred to the memory location which corresponds to the interrupting select code (FLG and IRQ lines). This memory location, also referred to as a "trap cell", should contain JSB instruction so as to transfer control to a device or service subroutine. At the end of the subroutine a JMP, I through the beginning of the same subroutine will return control to the program that was executing at the time of the interrupt.

5-5. I/O INTERFACE PCA DESIGN

5-6. INITIAL CONSIDERATIONS. The first step in designing an interface PCA is to draw a logic diagram of the PCA. Therefore, what is needed first is a list of the functions that must be present on the PCA. To make up this list, a careful study of all interface requirements is necessary. Consider questions such as the following:

a. What kind of data registers are required? Will the register be used for output only (to I/O device), input only (from I/O device), or will two or more registers be required to handle both output and input operations? How many flip-flops will be required to store all bits? Are any registers required at all? This may be the case if, for example, the I/O device has its own storage facilities. In this case, only a row of gates with a strobe input for IOI and/or IOO may be required. In most cases, however, interface PCA storage capability is recommended for greater system flexibility.

b. What commands are required? The flag and control logic set and clear states normally provide for a command sequence as follows: start device (Control flip-flop set), device busy (Flag flip-flop clear), device operation complete (Flag flip-flop set), and stop device (Control flip-flop clear). Is this sequence adequate? Are other commands such as tape rewind, upper/lower-case shift, mode switching, etc., required? If so, a command register may be required to accept command words from the computer. The reverse situation of the computer being slaved to or commanded by the I/O device is also possible. In this case, an input command register may be required. Perhaps no control lines at all are required for the I/O device. On input, for example, a computer program may simply require the current value of a count-accumulating I/O device. The computer need not command the I/O device to read, and the I/O device need not have to inform the computer that data is ready for transfer. Conversely, on output, data may simply be presented to the I/O device without any accompanying commands. For example, this would be possible if the I/O device was a

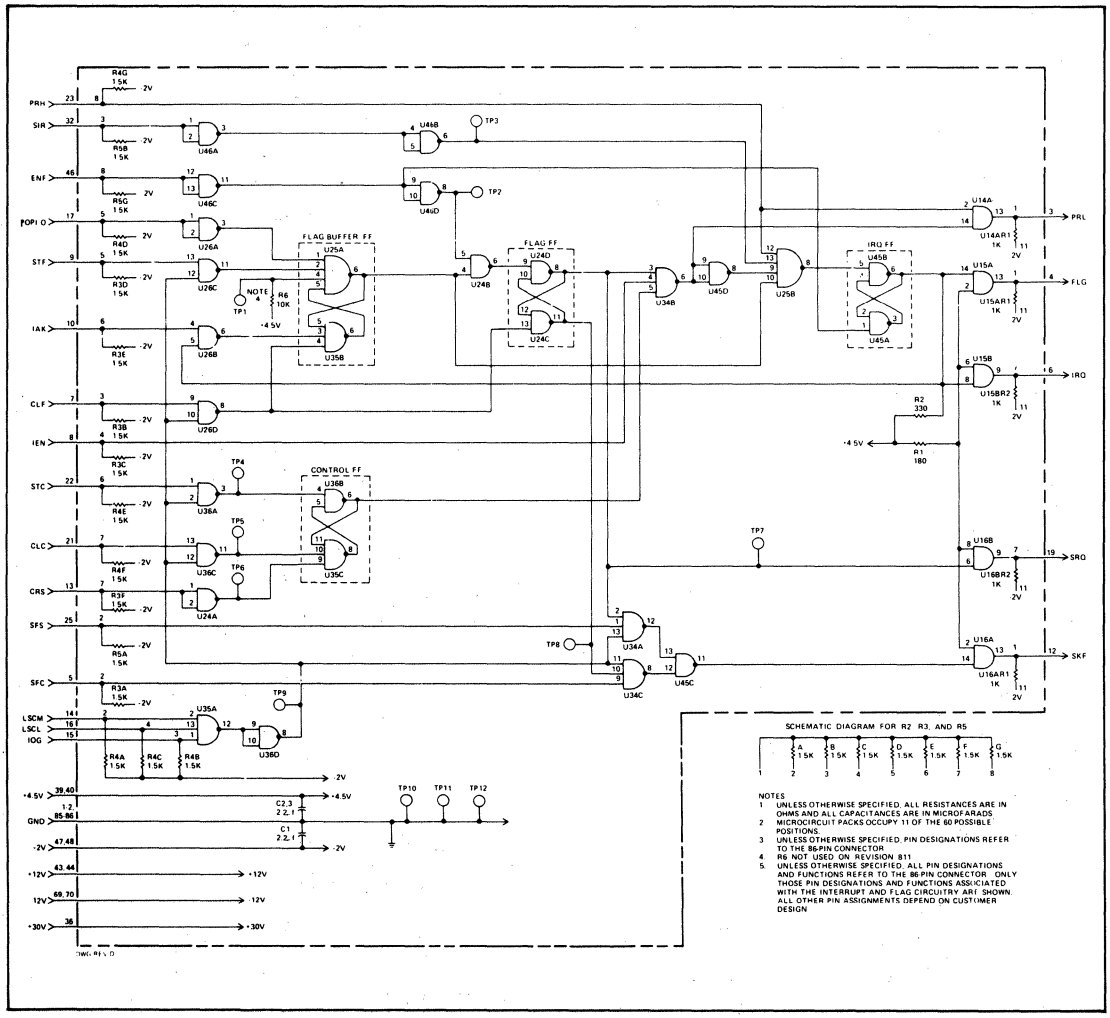


Figure 5-2. Flag and Interrupt Circuits Logic Diagram

Table 5-2. Flag and Interrupt Logic Component Identification

REFERENCE DESIGNATOR	HP PART NO.	COMMERCIAL EQUIVALENT
U14 thru U16	1820-1080	Signetics 8T13
U24,26,36,45,46	1820-0054	Texas Instruments SN4342
U25	1820-0069	Sprague Electric USN7420A
U34,35	1820-0068	Texas Instruments SN7410N

display unit or a device interlocked with some other program-synchronized device (e.g., a scanner-voltmeter relationship). In most cases, the recommended approach is to have the computer and I/O device completely interlocked so that each knows exactly what the other is doing.

c. Are multiple interface PCA's required? More than one interface PCA may be required if the involved logic is complex or if more than one address is required. It should be noted that one interface PCA can use two addresses, but to do so, the next higher I/O slot must be occupied by a priority jumper PCA.

d. How much work should be required of the interface PCA? Perhaps it may be more efficient to use the interface PCA merely to transmit data and command information to an intermediary device for the translation of complex operations that otherwise could not be physically designed into the interface PCA.

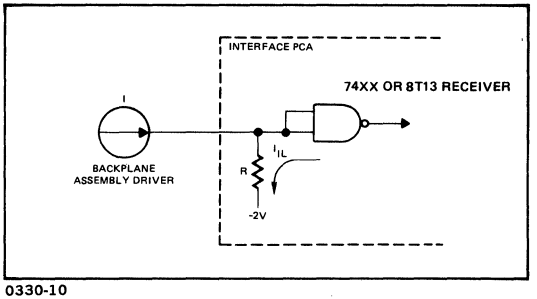
e. What type of logic is to be used? TTL integrated circuits are recommended for the logic design of the interface. Although the driving signals to the I/O bus require CTL characteristics they should be designed with TTL integrated circuits that exhibit CTL characteristics.

5-7. COMPUTER BACKPLANE ASSEMBLY REQUIREMENTS. The following paragraphs contain logic component selection rules and recommendations for designing interface PCA's.

5-8. Interface PCA Signal Receiving. All I/O signals from the computer backplane assembly are emitter-follower driven. A high signal state is indicated when the signal source drives current along the signal line. A low signal state is indicated when no current is driven.

A reveiw of available TTL logic families will indicate that the receiving gate cannot be a high-speed (H) or high-speed Schottky clamped (S) design because of the imposed low-value R restriction. Standard TTL receiving gates have been used for earlier I/O interface PCA designs and are still valid. However, for new designs, low-power Schottky (LS) components are recommended. These components require less power than standard TTL and are somewhat faster. Low-power Schottky Schmitt trigger gates are also recommended for backplane buffering and greater noise immunity. An accompanying R value of 4.7K ohms will place less demand on the backplane assembly drivers and reduce backplane switching current.

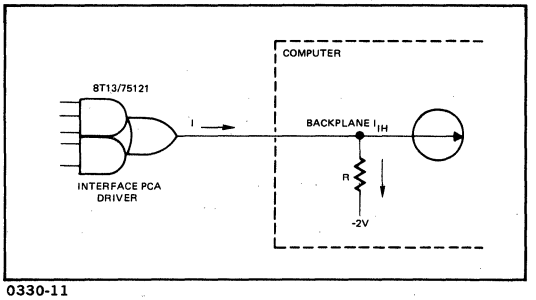
A valid interface PCA receiving circuit is either a TTL 74XX or 9XXX series gate. When receiving into TTL, a resistor pull to −2V is required to prevent input low current (I_{IL}) from the receiver from draining back onto the signal line and lifting it to a marginal logic level. (See Figure 5-3.) Compute the largest resistance value required to keep the signal line at a low potential as follows: $R = 2V/I_{IL}$ (I_{IL} obtained from manufacturers specifications). If an interface PCA is designed with multiple receiving devices in parallel, or using individual gates that have a large I_{IL}, a small value of R is required. Like earlier HP computers, the backplane assembly drivers can only source a finite guaranteeable current; therefore, the value of R must be restricted to a value of not less than 1.5K ohms.



0330-10

Figure 5-3. Valid Interface PCA TTL Receiver

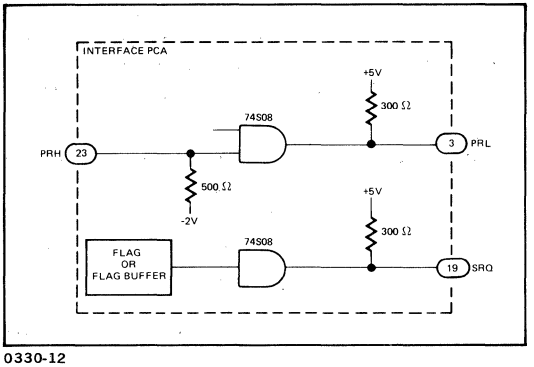
5-9. Interface PCA Signal Driving. Signals generated on the interface PCA for the computer must be capable of sourcing current through a resistive load to a −2V supply and must deliver an input high current (I_{IH}) adequate to turn on computer high-impedance input stage drivers to a logical "1" level as shown in Figure 5-4. These



0330-11

Figure 5-4. Valid Interface PCA Driver Circuit

signals are typically emitter-follower and OR-tieable, but the required current and characteristics of the line are dependent on the signal itself. CTL components (956 and 856 gates) should be avoided as drivers since these devices require high supply current (up to 70 mA per package) and have low switching thresholds (1.0V). A recommended component for signal driving is the 8T13 (HP part no. 1820-1080) or 75121 (Signetics Corp.) line drivers. These devices provide more than ample source current (greater than 60 mA) and sink current requirements are taken care of with the inclusion of a pull-down resistor to −2V (computed with the equation $R = 2V/I_{IL}$).

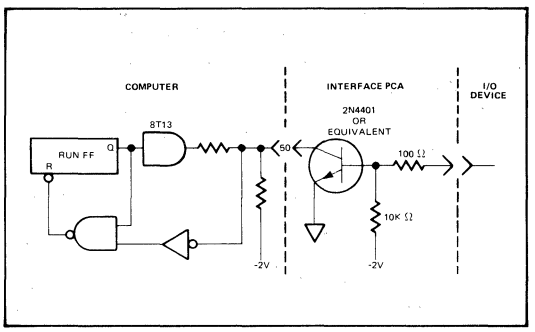


0330-12

Figure 5-5. Valid Interface PCA PRL and SRQ Line Drivers

5-10. I/O Run Signal. For HP 1000 M-Series Coomputers, the I/O backplane assembly Run signal (connector pin 50) is an emitter-follower image of the state of the computer's Run flip-flop. The signal is high when the computer is running and low when halted. For HP 1000 E/F-Series Computers, the Run signal line is a two-way communication link that can be used for remote control of an unattended or inaccessible computer. This feature is called "remote program load" and is discussed in the *HP 1000 E-Series Computer Operating and Reference Manual,* part no. 02109-90001 and the *HP 1000 F-Series Computer Operating and Reference Manual,* part no. 02111-90001. With this capability, an I/O device can pull down on the high Run line, reset the computer's Run flip-flop, and return the computer to its halt microroutines. To achieve this hardware capability inexpensively, a circuit similar to that shown in Figure 5-6 can be designed into the interface PCA. When the transistor's base is driven, its collector must sink up to 250 mA while it is in conflict with the 8T13 driving the Run line. The computer detects this conflict and clears its Run flip-flop.

5-11. I/O Microprogram Signals. For HP 1000 E/F-Series Computers, the I/O backplane assembly has three microprogram control signal lines available on connector pin numbers 18, 67, and 73. These lines are for use in microprogrammed I/O data transfer schemes discussed in Section VI. All three lines are ground-true and TTL-compatible.



0330-13

Figure 5-6. Remote Computer Halt Driver Circuit

5-12. POWER CONSIDERATIONS. Power available for I/O interfacing is listed in Table 1-1. Care should be taken when using custom-designed interface PCA's with high current drains not to exceed specified current limits. It should be remembered that the actual current available for I/O interfacing is the maximum current listed in Table 1-1 reduced by the current drain of any options or accessories installed in the computer.

When using voltages where two pins are provided, use both pins for contact. This decreases contact resistance and increases the current capacity of the connection. Due to contact capacity, the maximum current that can be used on any one interface PCA is 5.0A at +5.0V. When measuring current, always use an extender PCA and a clip-on ammeter.

When using certain combinations of I/O devices that have high current requirements, the computer's power suppy may not be adequate. If additional power is required, the HP 12979 I/O Extender must be used. The HP 12979 I/O Extender is a self-contained unit with a regulated power supply independent of the computer's power supply. Table 5-3 lists the additional current supplied by the HP 12979 I/O Extender.

Table 5-3. I/O Current Availability From I/O Extender

SUPPLY VOLTAGE	CURRENT
+5V	47.0A
−2V	5.0A
+12V	4.5A*
−12V	2.5A*
+28V	.25A

* 2.00A available to front and 2.5A available to rear I/O backplanes.

5-13. DEVELOPMENT CHECKLIST. The following paragraphs outline the sequence of steps that should be followed when developing an I/O interface PCA.

5-14. Draw Initial Logic Diagram. If flag and interrupt circuits are to be included on the interface PCA, use Figure 5-2 as a guide. If required, add an encoding flip-flop. For data and/or command word storage, add input receivers and storage registers and/or output storage registers and drivers. Refer to paragraphs 5-7 through 5-9 for receiver and driver selection information. For input and/or output storage register selection, the following type devices are recommended:

74LS374	Octal D-Type Register
74S374	Octal D-Type Register
74LS373	Octal D-Type Latch
74S373	Octal D-Type Latch
74LS175	Quad D-Type Register
74S175	Quad D-Type Register
74LS174	Hex D-Type Register
74S174	Hex D-Type Register

The exact logic configuration to be used of course will be determined by the associated I/O device's characteristics. To complete the logic diagram, add all required control lines and timing circuits.

5-15. Fabricate Working Model. Depending on the number of interface PCA's to be produced, the working model will be either the final product or a prototype. In either case, there are some basic considerations for layout that must be observed. Whether using a breadboard or creating original printed-circuit artwork, always keep signal paths as short as possible. Rework the layout as often as required to achive optimum signal path lengths. Etch a ground bus around the perimeter of the PCA. Lay out the +5V power bus in a grid pattern so that each integrated circuit receives power via the shortest possible direct path from connector pin numbers 39 and 40. Use 0.01 uF ceramic bypass capacitors liberally on the +5V and −2V power busses. No more than three integrated circuit packages should be served by one capacitor and no length of unbypassed power bus should exceed 3.0 inches (76.2 mm).

5-16. Test Working Model. Initially, test the interface PCA circuit paths for shorts with an ohmmeter. Next, connect the interface PCA into any available computer I/O slot with an extender PCA, connect the associated I/O device to the interface PCA, and energize both the computer and I/O device. Using Table 5-4 as a guide, write a simple noninterrupt program to operate the device and signal check the interface PCA with a logic probe. Note that the program listed in Table 5-4 assumes that the interface PCA is installed in the I/O slot assigned to select code 12 and that the XYZ device is an input device. This program will read one value from the XYZ device and then halt. The value read will be in the A-register for observation. Each time the computer's RUN switch is pressed, the XYZ device will complete a cycle and then halt.

If the XYZ device and interface PCA work properly in the noninterrupt data transfer mode, the next step is to check for proper operation in the interrupt mode. Using Table 5-5 as a guide, write a program that checks the interrupt capabilities of the interface PCA. When writing this type

Table 5-4. Interface PCA Test Program

LABEL	OPCODE	OPERAND	COMMENTS
	ORG	2000B	Page 1 origin.
XYZ	EQU	12B	Select code of XYZ device.
	CLF	00	Disable interrupt system.
LOOP	STC	XYZ,C	Start input device.
	SFS	XYZ	Is device busy?
	JMP	*−1	Yes, repeat previous instruction.
	LIA	XYZ	No, load input data into A-register.
	HLT		Halt.
	JMP	LOOP	Start program again.
	END		

Table 5-5. Interrupt Test Program

LABEL	OPCODE	OPERAND	COMMENTS
	ORG	12B	Set origin to 12B for JSB,I in the
	JSB	LINK,I	interrupt location.
LINK	DEF	SUBR	Interrupt subroutine address.
	ORG	2000B	Page 1 origin.
XYZ	EQU	12B	Select code of XYZ device.
START	STF	00	Enable interrupt system.
	LIB	1	Set S-register.
	SSB		Is S-register bit 15 set?
	RSS		
	JMP	*-3	No, stay in loop.
	ELB,		Yes, clear bit 15.
	CLE,		
	ERB		
	CMB,		Get count negative.
	INB		
	STB	CNTR	
	STC	XYZ,C	Start input device.
	INB		Simulated program in progress. Wait
	JMP	*-1	for interrupt.
	.		
	.		Subroutine to process interrupts.
	.		
SUBR	NOP		Subroutine entry.
	LIA	XYZ	Load input character into A-register.
	CLB		Time delay.
	ISZ	1	
	JMP	*-1	
	ISZ	CNTR	Increment counter. Finished?
	LIB	1	Get S-register.
	SSB		Is bit 15 set?
	JMP	*+5	Yes, go to exit.
	CLC	XYZ	No, clear device.
	HLT		Halt on interrupt.
	JMP	START	Get another request.
	STC	XYZ,C	Start input device again.
	JMP	SUBR,I	Return to interrupted point.

of program, the programmer must ensure that a known good instruction is stored in the device interrupt location. Any instruction can be placed in the interrupt location with the exception of JMP. The program listed in Table 5-5 uses JSB,I to illustrate an interrupt initiated transfer of control off the base page. The program uses bit 15 of the S-register as the controlling on/off switch and the remaining bits to control the number of cycles the XYZ device will make. After bit 15 is checked, a counter is set in the interrupt processing subroutine and the device cycle is initiated. When the XYZ device cycle is complete. an interrupt will occur and the coomputer will execute the JSB LINK,I instruction stored in the interrupt location. This instruction transfers control to the subroutine located on page 1 and loads the address of the interrupted instruction in the subroutine. The subroutine reads the data from the interface PCA into the A-register. Next, a small delay is programmed into the subroutine to allow S-register bit changes. If bit 15 is set, interrupts will be processed until the counter reaches zero at which time the simulated I/O request is satisfied. Before attempting to transfer data using the interrupt method, review the following information.

a. The interrupt system is enabled with an STF 00 instruction.

b. The interrupt priority linkage cannot be broken. All I/O channels with higher priority than the device being tested must be occupied or a special jumper PCA installed in place of any missing interface PCA.

c. No device with higher priority can be left with its interface PCA Control flip-flop set. This can create a problem similar to a missing interface PCA in the priority linkage. To eliminate this possibility, execute a CLC 00 instruction or manually press the computer's front panel PRESET switch.

d. When the computer is in the halt mode, the interrupt system is disabled. Therefore, the computer cannot be single-cycled through I/O operations that use the interrupt mode.

After using programs similar to those contained in Tables 5-4 and 5-5 to test the interface PCA, write a complete diagnostic program that will exercise every function of the interface PCA and associated I/O device. This program should test whether each function occured as commanded and report to the operator via coded halts or printed error messages whenever a function fails to occur.

5-17. Final Test and Production. Perform final checkout of the working model under all environmental conditions and, if required, update the logic diagram and layout drawing. If additional PCA's are to be produced from the working model, make final printed-circuit artwork and load new PCA's in accordance with the layout drawing. Using the diagnostic programs developed in paragraph 5-16, test each PCA with the computer and I/O device. Completely check each PCA for marginal signals and traces.

ADVANCED INTERFACING TECHNIQUES

SECTION VI

This section contains a general discussion of multiplexed and microprogrammed I/O techniques for the HP 1000 M-Series, E-Series and F-Series Computers. The method of multiplexing described in this section is party-line I/O. Two methods of microprogrammed I/O are also discussed: block I/O transfers and Microgrammable Processor Port transfers. The discussions of party-line I/O pertain to the HP 1000 M-, E- and F-Series Computers. The discussions of the microprogrammed block I/O transfers apply to the HP 1000 E- and F-Series Computers and the discussions of the Microprogrammable Processor Port transfers apply to the HP 1000 E-Series Computers only. In addition, this section contains information on the special requirements necessary to achieve high-speed DCPC transfers for the HP 1000 Computers.

6-1. PARTY-LINE I/O

If a large number of comparatively simple I/O devices are to be interfaced with the computer, it may be economically impractical to provide a separate interface PCA for each I/O device. In this case, party-line I/O provides a means of multiplexing (switching) each I/O device in turn to the computer using only those I/O signal lines normally assigned to a single I/O device. The I/O signal lines from the computer are bused to all I/O devices on the party line so that the I/O devices appear as a single device to the computer. Since the I/O signal lines are bused and identically available to all I/O devices on the party line, each I/O device must have its own controller. The controller must decode I/O device address and command information from the computer, send status information to the computer, and maintain overall control of its associated I/O device.

6-2. GENERAL INFORMATION

The party-line I/O described in the following paragraphs provides computer control for up to 256 I/O devices using only two interface PCA's. Each I/O device is connected party-line style to the interface PCA's through a user-designed controller. The number of devices that can be controlled is strictly a function of control word format. The two interface PCA's used in this technique are HP Microcircuit Interface PCA's, part no. 12566-60024. The two PCA's are identical and one each is supplied as part of an HP 12566B-001 Microcircuit Interface Kit. The 16-bit registers on the PCA's permit complete bidirectional data transfer. The I/O components on the PCA's permit the use of voltage levels in the 0 to +5V range for better noise margin and more desirable controller design by the user.

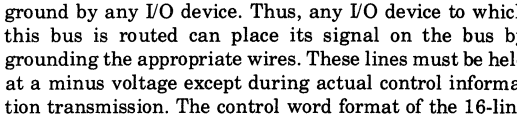
The two party-line interface PCA's plug into any two adjacent I/O slots in the computer and the remaining I/O slots can be used to interface standard I/O devices. The I/O slot positions occupied by the two PCA's establish the priority of the party line in relation to other I/O devices interfaced with the computer. Thus, the user can establish a party-line priority which is either higher or lower than the standard I/O devices connected to the computer. Party-line operations are performed at slower rates than operations using the standard I/O channels of the computer. Transfer rates of 40 kHz are possible under a noninterrupt mode, while rates are limited to 10 to 12 kHz under the interrupt mode. The latter rates are reduced mainly by software overhead time used in decoding party-line addresses.

6-3. PRINCIPLES OF OPERATION

As a guide toward implementing party-line I/O, the following paragraphs describe a typical party line with the addressing capability for 256 devices. This is a completely valid example, but it should be noted that this is only one of many ways that a party line can be implemented using two microcircuit interface PCA's. Another possible configuration would be the use of only one PCA, with possibly eight bits for data and eight bits for control.

In this discussion of party-line implementation, one of the two interface PCA's will be designated as the Control PCA and the remaining PCA as the Data PCA. It is assumed that the two PCA's are mounted in two adjacent I/O slots of the computer and that an interconnecting cable is connected between each of the PCA's and all I/O devices using the party line.

6-4. CONTROL PCA FUNCTIONS. The Control PCA contains a 16-bit input register and a 16-bit output register. For party-line applications, the register's input and output lines are connected together (bit 0 line of the input register connected to bit 0 line of the output register, etc.) with the interconnecting cable's 24-pin hood connector that mates with the PCA. Thus, both the input and output registers have access to a 16-line control bus. These lines are electrically designed to permit an "or-tie" to ground by any I/O device. Thus, any I/O device to which this bus is routed can place its signal on the bus by grounding the appropriate wires. These lines must be held at a minus voltage except during actual control information transmission. The control word format of the 16-line control bus is as shown below.



The 16-line bus from the Control PCA is available to each I/O device controller on the party line. Each controller must be capable of decoding I/O device addresses and commands received from the Control PCA and of sending status information back to the computer. The eight bits used for the device address permit an addressing capability of up to 256 I/O devices. The command and status bits are used as indicated in Table 6-1.

In addition to the 16 control bus lines, the Control PCA provides the user with an Encode line which is used to alert party-line I/O devices that an action is to be taken. A Device Flag line is also provided to permit party-line I/O devices to signal the computer that an action has been taken. A typical sequence of events to obtain status information from a party-line I/O device is as follows:

a. Load the computer A-register with a bit pattern specifying a party-line I/O device address and with bit 14 set to "1" to command the addressed I/O device to send status information back to the computer. (The I/O device address is determined by the user when the I/O device controller's address decoder is designed.)

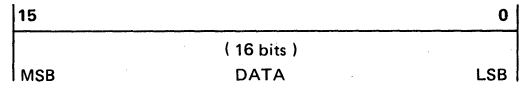
b. Output the contents of the A-register to the Control PCA. The bit pattern is now in the Control PCA's output register waiting to be placed on the control bus.

c. Execute an STC XX (XX is the select code of the Control PCA). This instruction gates the bit pattern stored in the output register onto the control bus. It also initiates the Encode signal which alerts the addressed party-line I/O device to take action.

d. The addressed I/O device's controller must now place the status of the device (bits 8 through 11) on the control bus and return the Device Flag signal to the Control PCA. The returned Device Flag signal causes the removal of the Encode signal and the address and command bits from the control bus. The Device Flag signal also causes the status bits to be gated into the Control PCA's input register.

e. Execute an LIA XX. This instruction loads the contents of the Control PCA's input register into the computer A-register to be checked by the user program. (The status code bit patterns are also determined by the user so that the bits from the I/O device's controller correspond to that which the user program expects to receive.

6-5. DATA PCA FUNCTIONS. Data PCA, like the Control PCA, also contains two 16-bit registers: one for input data and one for output data. The input and output lines from these registers are also connected together by the interconnecting cable's hood connector in the same manner as those on the Control PCA. Thus, both the input register and output register have access to a 16-line data bus. The entire 16 bits on the data bus are used for data transmission in the format shown below:



Two other lines are provided by the Data PCA: (1) an Encode (send/accept data) signal which is sent to the party line and (2) a Device Flag line which receives the data ready/taken signal from the party line. An STC instruction for the Data PCA causes a Data Enable signal to gate data from the Data PCA's output register onto the data bus. During this time, the Encode signal is initiated by the Data PCA which signals all I/O devices on the party line that data is available on the data bus. The I/O device that has been addressed by the computer can now accept the data. These signals keep data from the Data PCA off the data bus except when a data transfer is actually taking place under program control. Simultaneous placement of data onto the data bus by more than one I/O device is prevented since the party-line I/O devices can place data on the data bus only when commanded to do so by bit 12 from the Control PCA. The Flag signal received by the Data PCA gates data from the data bus into the Data PCA's input register during input operations. Thus, a party-line I/O device must send a Flag signal at the same time that is places data on the data bus for the computer.

Table 6-1. Command and Status Bit Assignments

BIT	ASSIGNMENT
8-11	Return status information from the I/O device to the computer.
12	If set to "1", the addressed I/O device is commanded to output data to the computer.
13	If set to "1", the addressed I/O device is commanded to accept data from the computer.
14	If set to "1", the addressed I/O device is commanded to output status to the computer.
15	Set to "1" under program control to indicate that the I/O device address will be used as an indirect address. (Used only during interrupt mode. The I/O device is not affected.)

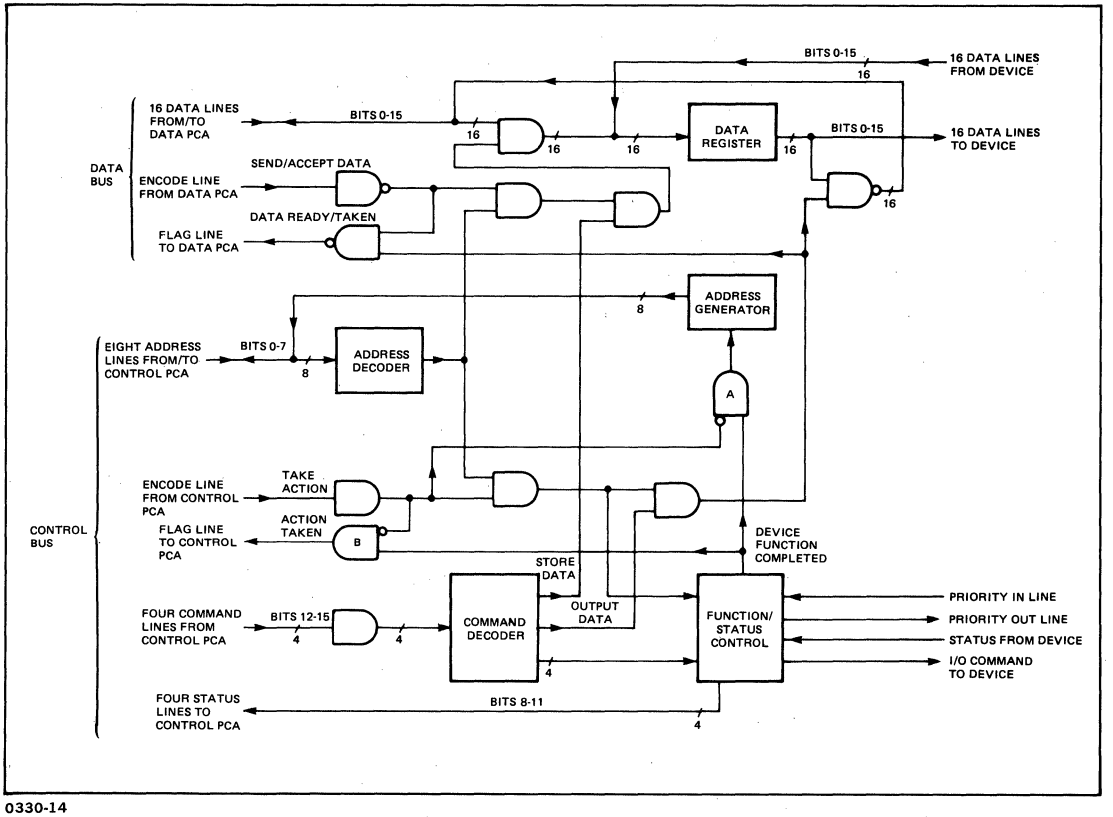
6-6. PARTY-LINE OPERATION DURING INTERRUPT MODE. Party-line I/O devices can be run using the computer interrupt system. One interrupt channel is provided for the party line and its associated I/O devices. The interrupt system of the party-line I/O devices must be established by the user. Each I/O device on the party-line has access to the Control PCA's Flag flip-flop via its Flag signal which initiates a computer interrupt request. The user must implement a priority chain through the I/O device controllers attached to the party-line so that only the highest priority I/O device of those devices requesting an interrupt has access to the control bus. After the higher priority I/O device is serviced, the next highest priority device is serviced, and so on. No more than one I/O device can request an interrupt at one time or the interrupt (I/O device) address will be erroneous.

At the same time that the I/O device returns its Flag signal requesting an interrupt, it must also place its identifying address on the designated control bus lines. An interrupt routine then reads this address and transfers control to the appropriate party-line I/O device interrupt routine. (The I/O device's controller must be designed to ensure that it does not place its identifying address on the control bus simultaneously with a computer address output.)

To ensure that the Data PCA does not interrupt the computer when data is gated in with the Flag signal from the I/O device, the PCA's Control flip-flop must be reset with a CLC instruction at all times other than during an output operation when the Data Enable and Encode signals are required. This inhibits the Data PCA from initiating an interrupt request.

6-7. CONTROLLER HARDWARE DESIGN

A logic diagram of a typical controller for an I/O device capable of both input and output operations is shown in Figure 6-1. This controller contains the maximum number of I/O lines available from the computer: eight address lines, 16 data lines, four command lines, and four status lines. The controller provides overall control of the party-line I/O device upon receipt of command information from the computer. It also provides device status information for the computer, alerts the computer when the commanded action has been completed, and properly maintains its position in the priority chain among all other party-line I/O devices.



0330-14

Figure 6-1. Typical I/O Device Controller Simplified Logic Diagram

As shown in Figure 6-1, an I/O operation with a party-line I/O device is initiated by a Command signal, the I/O device address, and Encode signals from both the Data and Control PCA's to the device controller. The Command Decoder informs the Function/Status Control of the action request, which then instructs the I/O device to perform its input/output function. The controller's Data Register provides buffer storage of data from or to the party-line data bus. The Command Decoder, in conjunction with the two Encode signals, determines whether data is to be applied to the I/O device or gated out of the computer. When the Function/Status Control has determined that the I/O device has completed its function, it sends a Flag signal to the Control PCA and simultaneously activates the Address Generator if the Control PCA's Encode signal has dropped. The Flag signal initiates an interrupt request to inform the computer that the I/O device function has been performed and that the device is ready to receive another command. The Address Generator identifies the requesting I/O device by placing its address on the lower eight lines of the control bus to the computer. Note that gates "A" and "B" in Figure 6-1 require that the Encode signal be removed before the Flag signal or device address is returned to the computer. This ensures that the I/O device address from the controller and the I/O device address from the computer are not placed simultaneously on the control bus.

During a computer input operation, the Command Decoder initiates the Flag signal and applies the controller's Data Register contents to the Data PCA. The Flag signal gates the data from the controller into the Data PCA's input register where it is available for the computer. The data bus Flag signal is not used during computer output operations. I/O device status information is obtained from the controller in the same manner as input data except that status information is sent to the Control PCA instead of the Data PCA. A Flag signal is returned to the Control PCA when status becomes available; a Flag signal is not returned to the Data PCA.

The Priority In and Priority Out lines form a simple chain running through all party-line I/O devices to establish priority when operating in the interrupt mode. When any I/O device interrupts the computer, the chain is broken and all lower priority devices are inhibited from interrupting until the original device has been serviced. Priority can be established in any manner the user desires to design the controllers, but a logical choice would be to have the highest priority devices be those requiring the highest data transfer rates.

6-8. INPUT PROGRAMMING USING NONINTERRUPT MODE

Programming a party-line I/O device noninterrupt input operation is similar to programming a noninterrupt input operation for a standard I/O device as previously discussed in Section III of this manual. A general noninterrupt input operation can be programmed as shown in Table 6-2. It should be noted that the program in Table 6-2 assumes that the I/O device is assigned a party-line address of 60 octal and that the Control and Data PCA's are installed in

Table 6-2. Party-Line NonInterrupt Input Routine

LABEL	OPCODE	OPERAND	COMMENTS
	CLF	00	Disable interrupt system.
	LDB	IADR	Load device address in lower eight bits of B-register and set bit 12 to "1" to command input operation (010060B).
	OTB	CNTL	Output B-register contents to Control PCA.
	STC	CNTL,C	Set Control PCA Control flip-flop to initiate Encode (take action) signal. Clear Flag flip-flop in preparation for recognition of Flag (action taken) signal from device.
	SFS	CNTL	Is Control PCA Flag flip-flop set (input ready)?
	JMP	*−1	No, repeat previous instruction.
	LIA	DATA	Yes, load Data PCA input register into A-register.
IADR	OCT	010060	Set IADR equal to device party-line address.
CNTL	EQU	10B	Set CNTL equal to Control PCA I/O slot select code.
DATA	EQU	11B	Set DATA equal to Data PCA I/O slot select code.

computer I/O slots assigned to select codes 10 and 11, respectively. To perform a looping operation (i.e., load several consecutive inputs into the computer), it is not necessary to repeat the first three instructions contained in Table 6-2 each time through the loop since these instructions are for initialization only.

6-9. OUTPUT PROGRAMMING USING NONINTERRUPT MODE

A general noninterrupt output operation can be programmed as shown in Table 6-3. The program in Table 6-3 assumes the same I/O device party-line address and interface PCA select codes as in the input program discussed in paragraph 6-8.

6-10. I/O PROGRAMMING USING INTERRUPT MODE

When programming party-line I/O devices using the interrupt mode, each I/O device's controller must be capable of sending an identifying address to the computer since several devices may be running simultaneously on the party line. These I/O device addresses must then be decoded by a user program to determine which specific interrupt routine should be entered. It should be noted that the necessary software overhead time spent in decoding these addresses causes data transfer rates to be slower than when operating in the noninterrupt mode. Actual transfer rate is dependent on the length of the interrupt routine, the time required to recognize an interrupt, and the time required to decode the address and process the data.

6-11. INTERRUPT PROCESSING. To initiate an interrupt request, the party-line I/O device controller sends a Flag signal to the Control PCA. It also places the requesting I/O device's address on the lower eight lines of the control bus. When the computer recognizes the interrupt request, it executes the instruction in the memory location corresponding to the Control PCA's I/O slot select code. This instruction is always a jump subroutine to a master interrupt subroutine designed to service party-line interrupts. The master interrupt subroutine causes a jump subroutine to the memory address corresponding to the address of the interrupting party-line I/O device. This memory address contains another indirect address that specifies the particular interrupt routine to service the requesting device.

6-12. SAMPLE PROGRAM. As shown in Table 6-4, only five instructions are required to initially program a party-line I/O device to input data to the computer using the interrupt mode. The program in Table 6-4 assumes the same I/O device party-line address and interface PCA select codes as in the noninterrupt input program discussed in paragraph 6-8. The program will continue until the I/O device is ready to input data. The I/O device then interrupts the computer by setting the Control PCA's Flag and IRQ flip-flops and placing its device address on the lower eight lines of the control bus. The computer executes the instruction in the memory location having an address corresponding to the Control PCA's I/O slot select code. This instruction is normally a jump subroutine indirect to a master interrupt subroutine designed to service all party-line interrupts. A sample master interrupt subroutine is contained in Table 6-5. It should be noted that the indirect address used in the JSB instruction of the program is actually the party-line address (60 octal) of the requesting I/O device. Thus, computer control will transfer to the address stored in memory location 60 octal. This is the actual starting place of the specific interrupt routine to service this particular I/O device. A sample specific interrupt routine for an input device is contained in Table 6-6.

6-13. DCPC TRANSFERS

As previously discussed in Sections III and IV, high-speed data transfer rates are achieved by effecting DCPC cycle steals; i.e., the DCPC channel receives a Service Request (SRQ) signal from the I/O interface PCA to request more data before the computer timing has advanced past the time period when a new DCPC cycle begins. Therefore, the time delay between the issuance of the I/O device's Flag (start) signal and the receipt of the SRQ signal by the computer is critical to effect successive I/O cycle steals. When designing an interface PCA or when selecting an HP general-purpose interface PCA for DCPC applications, particular attention must be paid to the timing of the interface PCA's flag and interrupt circuit SRQ signal (initiate DCPC cycle) and to when the I/O device's Flag signal is received.

Full DCPC bandwidth transfers assume that the DCPC option is stealing every CPU I/O cycle, that is, the Service Request (SRQ) signal is fully asserted at every consecutive sampling time on the DCPC PCA. The sampling times for the HP 1000 Computers are as follows:

M-Series:	T5P1, leading edge
E/F-Series:	T4P4, leading edge

The DCPC transfer rates presented in Table 1-1 are given under the full bandwidth assumption. To derive the transfer rates for less than full DCPC bandwidth, the number and access times of actual memory accesses involved in the execution of normal CPU instructions during I/O cycles in which DCPC does not have control must be taken into account. Therefore, one-half bandwidth figures may not necessarily equal the full bandwidth figures divided by two.

For applications in which transfer rates of less than one-half of the full DCPC bandwidth (approximately 616,666 bytes/second for the HP 1000 M-Series Computers and 860,000 bytes/second for the HP 1000 E/F-Series Computers) are sufficient, the flag and interrupt circuit (illustrated in Figure 5-2) will fulfill SRQ signal generation

Table 6-3. Party-Line NonInterrupt Output Routine

LABEL	OPCODE	OPERAND	COMMENTS
	CLF	00	Disable interrupt system.
	LDB	OADR	Load device party-line address in lower eight bits of B-register and set bit 13 to "1" to command an output operation.
	OTB	CNTL	Output B-register contents to Control PCA.
	STC	CNTL,C	Set Control PCA Control flip-flop to initiate Encode (take action) signal. Clear Flag flip-flop in preparation for recognition of Flag (action taken) signal from device.
	LDA	BUFF,I	Load output data word into A-register.
	OTA	DATA	Output A-register contents to Data PCA.
	STC	DATA,C	Set Data PCA Control flip-flop to gate output data onto data bus and to initiate an Encode signal.
	SFS	CNTL	Is Control PCA Flag flip-flop set (data received by device)?
	JMP	*−1	No, repeat previous instruction. Yes, continue program.
	•		
	•		
	•		
OADR	OCT	020060	Set OADR equal to device party-line address.
BUFF	DEF	BADDR	Define storage location of data word.
CNTL	EQU	10B	Set CNTL equal to Control PCA I/O slot select code.
DATA	EQU	11B	Set DATA equal to Data PCA I/O slot select code.

Table 6-4. Party-Line Interrupt Input Routine

LABEL	OPCODE	OPERAND	COMMENTS
	STF	00	Enable interrupt system.
	CLC	DATA	Clear Data PCA Control flip-flop to inhibit interrupt requests from Data PCA.
	LDB	IADR	Load device party-line address in lower eight bits of B-register and set bit 12 to "1" to command an input operation.
	OTB	CNTL	Output B-register contents to Control PCA.
	STC	CNTL,C	Set Control PCA Control flip-flop to initiate input operation.
IADR	OCT	010060	Set IADR equal to device party-line address.
CNTL	EQU	10B	Set CNTL equal to Control PCA I/O slot select code.
DATA	EQU	11B	Set DATA equal to Data PCA I/O slot select code.

Table 6-5. Master Interrupt Subroutine

LABEL	OPCODE	OPERAND	COMMENTS
MAST	NOP		Subroutine entry point. The return to the interrupted program is stored here by the jump subroutine instruction.
	STA	T1	Store contents of A-register in memory location T1.
	LDA STA	MAST T2	Load address to which program must return after interrupt request has been received into A-register.
	LIA	CNTL	Load device party-line address from Control PCA into A-register.
	IOR	B15	Set A-register bit 15 to "1" to allow device address to be used as an indirect address.
	JSB	A,I	Jump subroutine to specific device interrupt service subroutine using address contained in A-register.
T1	OCT	0	Storage location for A-register contents.
T2	OCT	0	Storage location for return address.
B15	OCT	100000	Mask to set bit 15 to "1".
A	EQU	0	Set A equal to memory location 0 (A-register).

Table 6-6. Specific Device Interrupt Subroutine

LABEL	OPCODE	OPERAND	COMMENTS
ENTR	NOP		Subroutine entry point. The address (location T1) for return to the master interrupt subroutine is stored here by the jump (JSB) instruction in the previous program (Table 6-5).
	STA	ADDR	Store device address (presently in A-register) in location ADDR.
	LDA STA	ENTR,I T1	Use address stored in ENTR to load and store original A-register contents previously stored in master interrupt subroutine.
	ISZ	ENTR	Increment address stored in ENTR, use it to load original return address from master interrupt subroutine, and store it as this routine's return address.
	LDA STA	ENTR,I ENTR	Gather necessary addresses and original register contents from master interrupt subroutine. (Frees master interrupt subroutine for processing other interrupt requests from party-line devices.)
	LIA	DATA	Load data from Data PCA into A-register.
	STC • • • •	CNTL,C	Set Control PCA Control flip-flop to re-enable party-line interrupt requests. (Causes new operation by addressed party-line device.) (Routine to process data received from party-line device.)

Table 6-6. Specific Device Interrupt Subroutine (Continued)

LABEL	OPCODE	OPERAND	COMMENTS
	LDA JMP	T1 ENTR,I	Interrupt process complete. Restore original A-register contents and jump back to interrupted main program.
ADDR	OCT	0	Storage location.
T1	OCT	0	Storage location.
CNTL	EQU	10B	Set CNTL equal to Control PCA I/O slot select code.
DATA	EQU	11B	Set DATA equal to Data PCA I/O slot select code.

timing requirements. As shown in Figure 6-2A, this circuit generates the SRQ signal at the first T2 time following the receipt of the Device Flag signal and, as such, can only initiate data transfers at a maximum rate of every other I/O cycle (half DCPC bandwidth). This circuit (Figure 6-2A) is used with most HP general-purpose PCA's, including the breadboard interface kit.

DCPC transfer rates of greater than half DCPC bandwidth can be brought about by several methods. The most appropriate method to use depends on three considerations: the series of HP 1000 Computer used, the I/O device speed, and whether or not handshake will be used. For the purposes of this discussion, these methods are presented in two categories: handshake and non-handshake transfers.

The first handshake method of achieving transfer rates of up to full DCPC bandwidth includes the use of an asynchronous flag circuit. The flag and interrupt circuit of Figure 6-2A can be either modified or designed similar to that shown in Figure 6-2B. This circuit generates the SRQ signal immediately upon receipt of the device flag signal, is not dependent on time T2 and, as such, can initiate data transfers at a maximum rate of up to every successive I/O cycle (full DCPC bandwidth) assuming that the I/O device is capable of generating a device flag signal that is fully asserted at the beginning of I/O cycle time T4. Propagation delays through the I/O device, the interface PCA and the cable *must* be taken into account. This method can be used with the HP 1000 M-Series Computers but must not be used with the HP 1000 E/F-Series Computers because of the shortened time duration of the I/O cycle T-periods. A buffering scheme is recommended for use with the E/F-Series Computers.

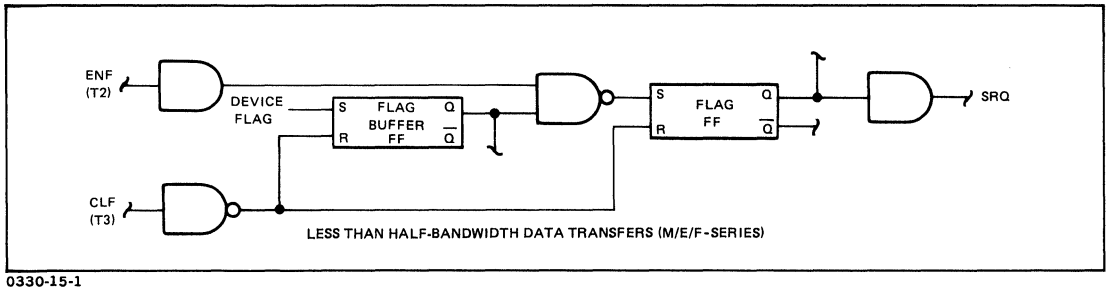
The buffering scheme is the second method of achieving the full DCPC bandwidth using handshake. Since the data transfer rate may be less than the DCPC latency time, more than one word can be received from the device before the DCPC gains control. Therefore, a buffering scheme is required to save data words until the channel is operative. The size of the buffer can be determined by dividing the worst case DCPC latency time by the data transfer time. For example, if the worst case DCPC latency time is 6 microseconds and words are transferred at 1 microsecond intervals, a 6 word buffer would be required.

$$\frac{6\ \mu\text{sec (latency)}}{1\ \mu\text{sec (transfer time/word)}} = 6 \text{ word buffer}$$

Additional words should be added for safety margin purposes.

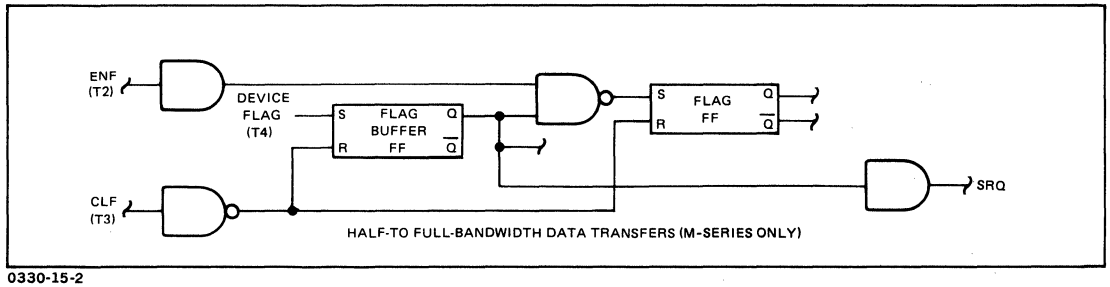
The diagram illustrated in Figure 6-2C provides the basic control logic and data handling to achieve DCPC transfers at full bandwidth with an E-Series or F-Series Computer. The design shown can be used in either the read or write mode and may utilize either DCPC channel. The design employs asynchronous logic to handle transfers to and from a 16 word First-In/First-Out memory (FIFO). This feature allows the data transfers to occur at the highest possible speed, limited only by the internal delays of the FIFO and the speed of the device. Synchronization with the CPU is achieved via DCPC cycle requests initiated by SRQ, which the DCPC samples during T4. The DCPC, upon receipt of a machine cycle, will issue I/O signals at particular T-periods. The timing diagram of Figure 4-8 illustrates the DCPC timing sequence.

Handshake need not be used to achieve the full DCPC bandwidth. However, the preprequisite to using non-handshake transfers is that the I/O device must be capable of transferring data at rates *faster* than the full DCPC bandwidth (approximately 1.23 Mbytes/sec with the HP 1000 M-Series and 2.28 Mbytes/sec with the E/F-Series Computers). The DCPC must receive a fully asserted and stable SRQ signal by P4(E/F) or P1(M) microcycle of time period T4(E/F) or T5(M) of the current I/O cycle in order to steal the next I/O cycle. In the E/F-Series, microcycle P3 of T4 can be extended for the first SRQ only. Therefore, T4 P-periods are short microcycles. Two methods of ensuring a fully asserted SRQ signal for the correct sampling times of the current I/O cycle involve either tieing SRQ high through the use of a jumper as illustrated in Figure 6-2C or tieing the Device Command signal directly back to the Device Flag line on the interface PCA. Again, both methods assume that the I/O device is capable of handling data as fast as the interface PCA can present or take it.



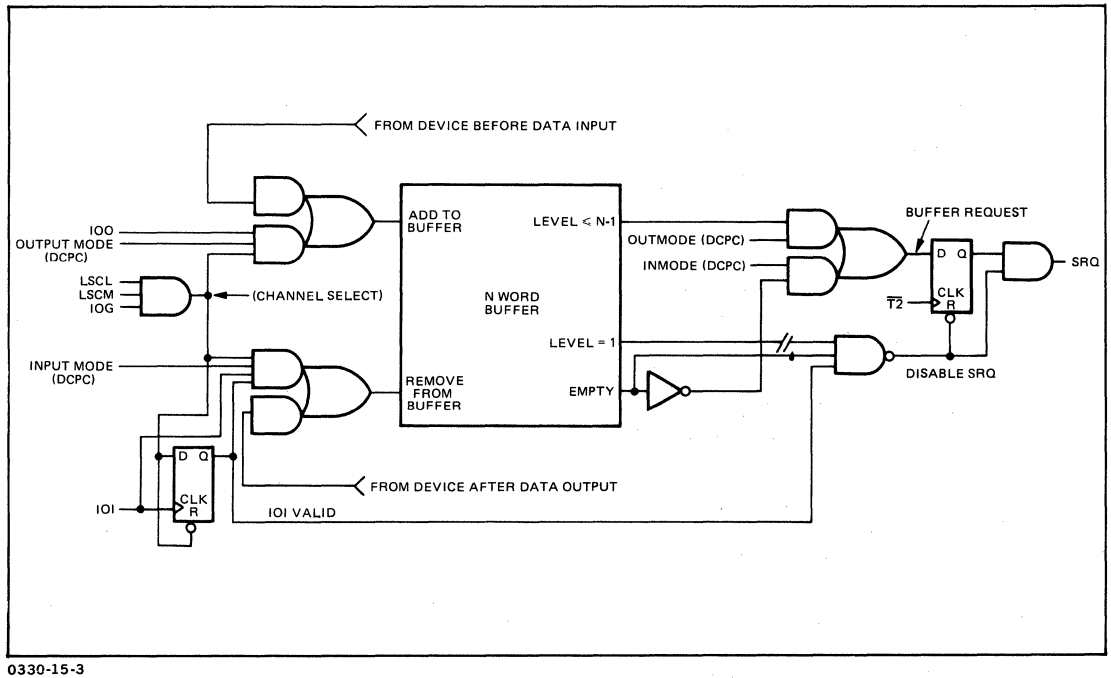
0330-15-1

Figure 6-2A. Critical DCPC Interface Design (Half-Bandwidth)



0330-15-2

Figure 6-2B. Critical DCPC Interface Design (M-Series, Full-Bandwidth)



0330-15-3

Figure 6-2C. Critical DCPC Interface Design (E/F-Series, Full-Bandwidth)

The non-handshake methods suffice for applications in which the I/O device does not generate the necessary signals to effect a complete handshake transfer. However, the buffering scheme using handshake is recommended for applications in which the I/O device does generate the signals necessary for a complete handshake.

6-14. MICROPROGRAMMED I/O

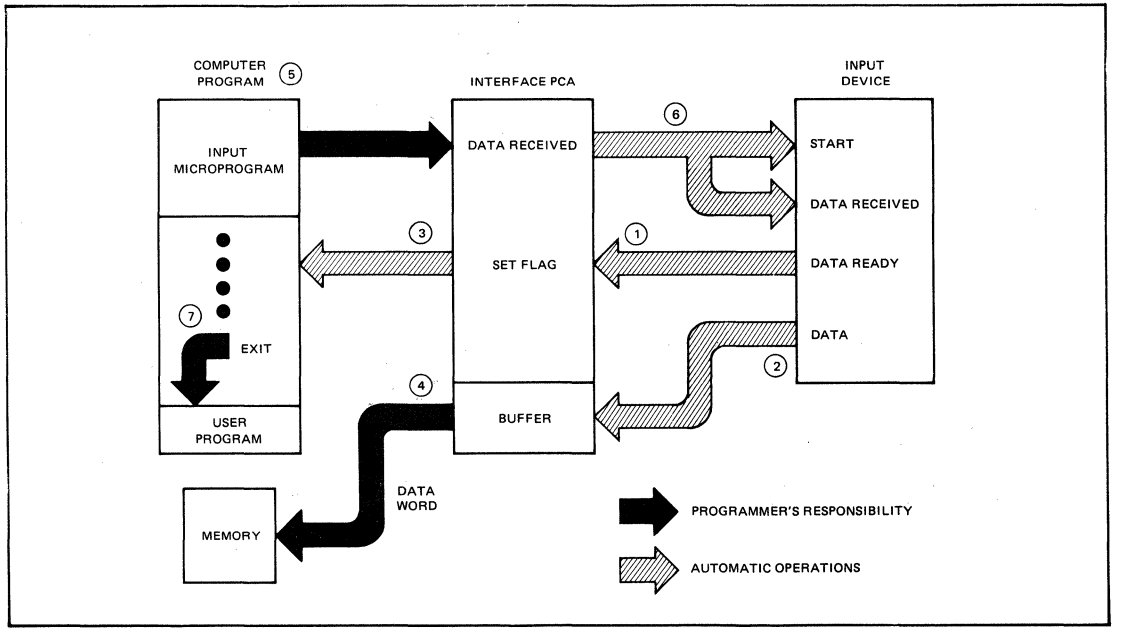
The HP 1000 Computers have a user-microprogrammable control processor which allows the user to expand the computer's instruction set so that the computer can perform specific functions more efficiently. Through the use of microprogramming, computer execution time of often-used routines can be greatly reduced. An introduction to Hewlett-Packard's microprogramming techniques and development is contained in the *HP 1000 E-Series Computer Operating and Reference Manual,* part no. 02109-90001 and the *HP 1000 F-Series Coomputer Operating and Reference Manual,* part no. 02111-90001. Complete information on how to prepare, load and execute microprograms is contained in the *HP 1000 E-Series Computer Microprogramming Reference Manual,* part no. 02109-90004. A thorough understanding of the contents of these manuals must be obtained prior to attempting any microprogrammed I/O operations. Specifically, this manual contains information pertaining to microprogrammed block I/O transfers via the computer's I/O Section for both the E-Series and F-Series Computers and a general discussion of the use of the E-Series Computer's Microprogrammable Processor Port (MPP) for I/O applications.

A listing illustrating the forming and execution of microprogrammed I/O instructions is contained in Table 6-7. It should be noted that this table is provided for example purposes only to illustrate the I/O functions performed by particular microinstructions. As is evident, the sequences of microinstructions provided emulate assembly language I/O instructions. This type of microprogrammed I/O generally does not produce appreciable transfer rate increases. However, it does allow you to develope custom I/O instructions. Two types of microprogrammed I/O that do effect appreciable transfer rate increases are Microprogrammed Block I/O transfers and Microprogrammable Processor Port transfers.

6-15. MICROPROGRAMMED BLOCK I/O TRANSFERS

Microprogrammed block I/O (MBIO) transfers via the I/O Section provide the capability of high-speed data transfers between the computer and a peripheral device in an asynchronous manner (with respect to T-periods) at rates up to 1.59 million words per second as specified in Table 1-1. This capability provides a higher bandwidth than DCPC for some applications and, in addition, provides for special-purpose I/O operations such as byte packing, etc.

Figure 6-3 illustrates the sequence of operations for a MBIO input transfer. The sequence of events are as follows. The input device outputs a Data Ready signal (1) and



0330-16

Figure 6-3. Microprogrammed Block I/O Input Data Transfer

Table 6-7. Forming and Executing Microprogrammed I/O Instructions

LABEL	OP/ BRCH	SPCL	ALU/ MOD/ COND	STR	S-BUS/ OPRD/ ADDRESS	COMMENTS
CIR				L	CIR	Places the select code of the I/O device stored in the Central Interrupt Register into the L-register via the S-bus.
STC (T2) (T3) (T4) (T5) (Refer to Note1.)	IMM	L4 IOG NOP NOP NOP	CMLO IOR	S8 S8 IRCM	303B S8 S8	Generates the binary number required to form an STC 0,C and combines the STC 0,C with the select code stored in the L-register to form an STC SC,C. The CPU is frozen until the next I/O Section T2 time period and the STC SC,C is executed at the following T4 time period.
LIA/B (T2) (T3) (T4) (T5)	IMM	 IOG NOP NOP	CMHI IOR	S4 S4 IRCM S5	376B S4 S4 IOI	Generates the binary number required to form an LIA/B 0 and combines the LIA/B 0 with the select code stored in the L-register to form an LIA/B SC. The CPU is frozen until the next I/O Section T2 time period, the LIA/B is executed, and data from the selected I/O device is gated from the I/O bus into Scratch Register S5 at the following T5 time period.
OTA/B (T2) (T3) (T4) (T5)	IMM	L1 IOG NOP NOP	CMLO IOR	S9 S9 IRCM IOO IOO	77B S9 S9 S5 S5 S5	Generates the binary number required to form an OTA/B 0 and combines the OTA/B 0 with the select code stored in the L-register to form an OTA/B SC. The CPU is frozen until the next I/O Section T2 time period, the OTA/B is executed, and the data from Scratch Register S5 is gated from the S-bus to the selected I/O device at the following T4-T5 timeperiods. (Refer to Note 2.)
IAK		IAK				Freezes the CPU until the next I/O Section T6 time period, loads the select code of the interrupting I/O device into the Central Interrupt Register, and generates an IAK signal for the I/O device.

OP = Operation Field
MOD = Modifier Field
OPRD = Operand Field
BRCH = Branch Field
COND = Condition Field
SPCL = Special Field
STR = Store Field

NOTES:
1. Parenthetical T-periods listed in the LABEL column are for reference purposes only to illustrate the use of NOP micro-orders to synchronize the microprogram with I/O Section timing. Any non-freezable micro-order can be used in place of the NOP.
2. Micro-order IOO is not the same as the I/O Section IOO signal. In order for an output transfer to be accomplished properly, micro-order IOO must be issued at I/O Section T4-T5 time period and the IOO signal must be present at T3-T4 time period.

a Data Word (2) to the interface PCA. The Data Ready signal sets the interface PCA flag and clocks the data word through the input buffer. The computer recognizes the Set Flag signal (3) generated by the interface PCA, writes the first data word (4) into the memory starting address specified in the microprogram, and immediately sends a Data Received signal (5) to the interface PCA. The interface PCA outputs the data received signal which is recognized by the input device as a Start signal (6). The process now repeats back to the beginning of this paragraph to transfer the next word. After the specified number of words (up to 256) have been transferred into memory, the microprogram (7) returns control to the next instruction.

As an aid toward a better understanding of how microprogrammed block I/O transfers can be accomplished, some typical transfer schemes will be discussed in the following paragraphs. Each of the discussed I/O transfers will use the standard I/O backplane connector signals and the three special microprogrammed block I/O signals ($\overline{\text{BIOI}}$, $\overline{\text{BIOO}}$, and $\overline{\text{BIOS}}$) also available on each I/O backplane connector. (I/O backplane connector pin number assignments for all I/O signals are listed in Table 4-1.) Figure 6-4 is a flow diagram for microprogrammed block I/O transfers that illustrates which microinstructions generate the required I/O transfer signals and a typical I/O interface PCA that can be designed to use these signals to transfer data. MBIO timing diagrams are provided in Figure 6-5. It should be noted that the flag and interrupt circuit design discussed in Section V must be modified as shown in Figure 6-6 for microprogrammed block I/O transfers.

The circuits shown in Figures 6-4, 6-6A, 6-6B and 6-6C synchronize the SKF signal to decrease the probability of an oscillatory input driving the SKF signal due to inadequate set-up time at the first flip-flop. The synchronization is necessary to ensure a relatively hazard-free SKF signal for the CPU to test. Figure 6-5 illustrates the amount of time available to enable input or output data to or from the interface PCA. Any delay times in the user interface PCA must meet these specifications.

An alternate addressing scheme should be employed for MBIO transfers. Instead of addressing the card via select code signals generated by the lower six bits of the IR (Instruction Register), the state of the control flip-flop should determine addressing. The control flip-flop is set on the MBIO interface card with an STC sc,C instruction in assembly code before the microroutine is called. It is important to have the MBIO interface PCA addressed when $\overline{\text{BIOS}}$ (P4) occurs, since $\overline{\text{BIOS}}$ is an additional qualifier for the $\overline{\text{BIOI}}$ and $\overline{\text{BIOO}}$ signals. When DCPC is used in conjunction with MBIO, DCPC takes control of the select code bus at P4 (the same time that $\overline{\text{BIOS}}$ occurs) preceding the DCPC cycle. Proper select code bus signals are not generated because the addressing qualifier is disabled when $\overline{\text{BIOS}}$ is enabled. The alternate addressing scheme eliminates competition for the select code bus and allows proper signal generation.

Other MBIO restrictions arise due to DCPC and memory refresh. Since MBIO and DCPC share the same I/O bus, MBIO can contaminate DCPC data if a DCPC freeze occurs with an IOI or IOO micro-order in the MIR (Micro Instruction Register). In the first case, control of the S-Bus is returned to the CPU during T4 of a DCPC output cycle. The decoders are re-enabled and the $\overline{\text{BIOI}}$ signal is generated. $\overline{\text{BIOI}}$ enables data from the MBIO interface PCA onto the I/O bus at the same time that DMALCH holds DCPC output data on the I/O bus. In the second case, IOO in the store field enables the S-Bus data onto the I/O bus for the length of the DCPC input or output cycle. For both cases, the resultant data is the inclusive-OR of the two sources. This can be avoided by placing a dummy read or RJ30 micro-order one or two microinstructions before the IOI or IOO micro-orders. This prevents loading IOI or IOO microinstructions into the MIR because the read or RJ30 micro-order freezes the CPU prior to the DCPC cycle.

A microprogramming restriction arises when the $\overline{\text{BIOI}}$ signal is used for a handshake acknowledgement or for an "increment buffer pointer" signal. Consider the following line of microcode:

```
WRITE PASS TAB IOI
```

Since this is a write micro-order, the machine freezes on this line if it concides with refresh or DCPC. Before freezing, the CPU performs the IOI into tab transfer. This transfer is performed a second time after the freeze. In the case that $\overline{\text{BIOI}}$ is used to bump the buffer pointer, a new word is transferred and the previous word is lost. In the case that $\overline{\text{BIOI}}$ is used for a handshake acknowledgement, there will be two acknowledgements for a one word transfer. The solution to this problem is to pass the data with IOI into a scratch pad register as shown in the following segment of microcode:

```
PASS S1 IOI
WRITE PASS TAB S1
```

In addition, the CPU counter (CNTR) cannot be used as the word count register in an MBIO input transfer because the I/O bus is disabled from driving the S-Bus whenever the select code (lower 6 bits of the CNTR) is less than seven.

For example purposes, assume that a block of data from the memory of a "master" computer is to be transferred into the memory of a "slave" computer under microprogram control. First, a driver circuit similar to that shown in Figure 6-6A must be designed on an HP Breadboard Interface PCA and the PCA inserted into the "master" computer's I/O slot. Second, a receiver circuit similar

to that shown in Figure 6-6B must be designed on another HP Breadboard Interface PCA and this PCA inserted into the "slave" computer's I/O slot. Third, the two interface PCA's J1 connectors must be interconnected with a cable assembly. Fourth, a microprogrammed block I/O output routine similar to that contained in Table 6-8 must be developed for the "master" computer and a microprogrammed block I/O input routine similar to that contained in Table 6-9 must be developed for the "slave" computer.

Table 6-8. Block I/O Output Microprogram

LABEL	OP/ BRCH	SPCL	ALU/ MOD/ COND	STR	S-BUS/ OPRD/ ADDRESS	COMMENTS
ENTRY		IOFF				Disable normal interrupts
	READ*		PASS	S11	M	Save return address
			INC	PNM	A	Pick up buffer address
			PASS	IOO	B	Output word count
	JMP	CNDX	ALZ		RETURN	
LOOP	JMP	CNDX	HOI		ABORT	Test for emergency interrupt.
	READ		PASS			
	JMP	CNDX	SKPF	RJS	*−2	Flag set? Yes; process transfer output word
		PRST	INC	PNM	P	
			PASS	IOO	TAB	
			DEC	B	B	
	JMP	CNDX	ALZ	RJS	LOOP	All words transferred? Yes, return.
RETURN	READ	ION	INC	PNM	S11	
	RTN					
ABORT			•			Save word count, buffer address, etc. Set P = S11 and jump to base set HORI routine.
			•			
			•			
	JMP				HORI	

* Dummy read.

Table 6-9. Block I/O Input Microprogram

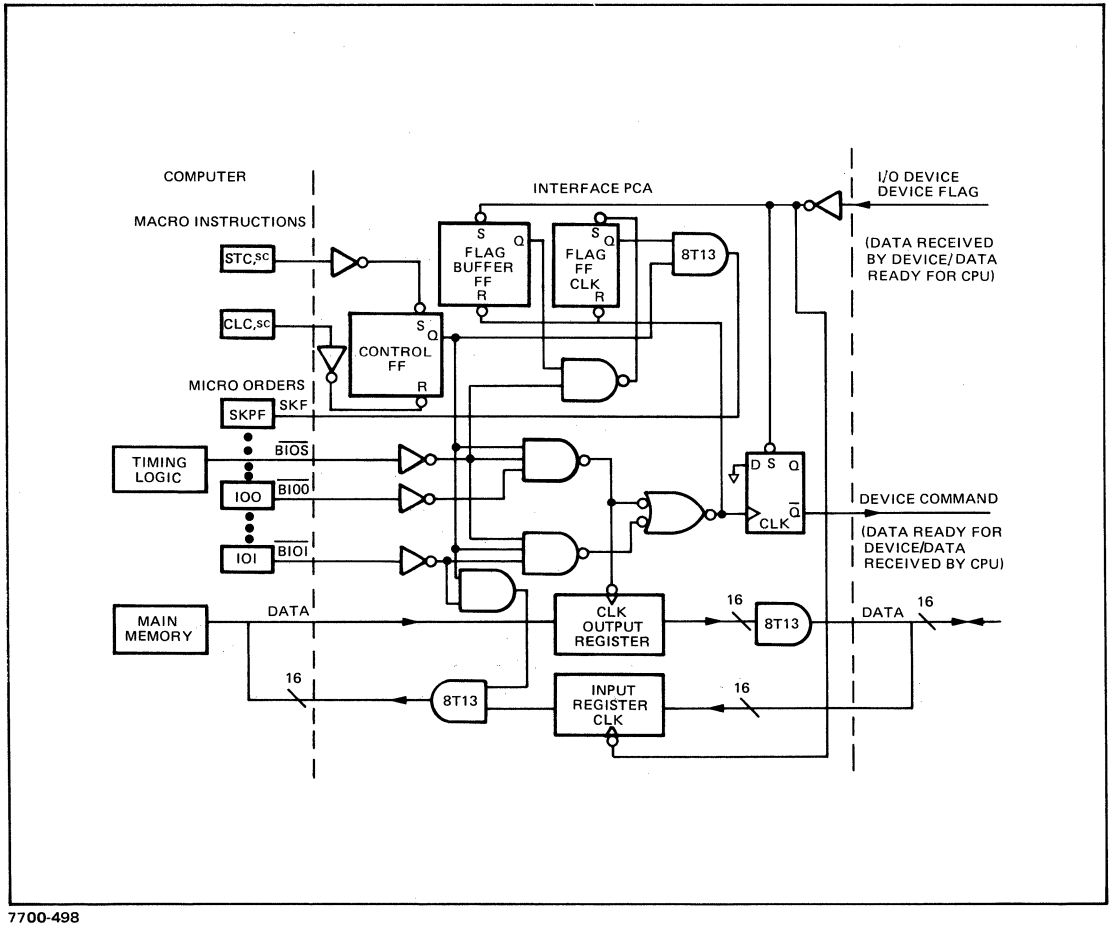
LABEL	OP/ BRCH	SPCL	ALU/ MOD/ COND	STR	S-BUS OPRD/ ADDRESS	COMMENTS
ENTRY		IOFF	DEC	S11	P	Disable normal interrupts
	READ*		PASS	P	A	Save return address
			PASS	B	IOI	Pick up buffer address and word count.
	JMP	CNDX	ALZ		RETURN	Word count = 0?
			PASS			
LOOP	JMP	CNDX	HOI		ABORT	Test for emergency interrupt. Flag set? Yes, process
	JMP	CNDX	SKPF	RJS	*−2	transfer Pick up word, Write it into memory.
	READ*		INC	PNM	P	
			PASS	S1	IOI	
	WRITE		PASS	TAB	S1	
			DEC	B	B	
	JMP	CNDX	ALZ	RJS	LOOP	All words transferred? Yes, return.
RETURN	READ	ION	INC	PNM	S11	
	RTN					
ABORT			•			Save word count, buffer address, etc. Set P = S11 and jump to base set HORI routine.
			•			
			•			
	JMP				HORI	

* Dummy read.

The program listed in Table 6-8 assumes the buffer address (starting address of memory where the block of words is stored) is in the A-register and the word count is in the B-register. The output program of Table 6-8, being associated with the "master" computer, starts the data transfer. Due to the master computer's interface PCA circuitry (Figure 6-6A) the flag signal is present on entry into the microroutine to meet the SKPF condition to start the transfer. The "slave" computer fields the first transfer request, via an interrupt from the slave computer's interface PCA. The slave's input microprogram (Table 6-9) is entered with the A-register containing the buffer address. The word count is picked up from the slave's interface PCA and the data transfer loop is entered. Note that all read micro-orders marked with an asterisk (*) are dummy reads which are necessary due to possible DCPC or memory refresh interaction. Also note that the MBIO programs test for emergency interrupts (i.e., power fail and parity error) and leave the MBIO routine if an emergency interrupt occurs. If an emergency interrupt branch is taken in either computer, the other computer will hang up waiting for the failing computer's handshake signals. Therefore, when implementing MBIO transfers, this situation should be handled in either hardware or software.

The programs listed in Table 6-8 and 6-9 are capable of transferring data at 1.9 Mbytes per second for input transfers and 2.04 Mbytes per second for output transfers. Both transfer rates assume DCPC is inactive, therefore, all machine cycles are available to the CPU.

Higher transfer rates can be achieved if SKF synchronization is not required. Table 6-10 contains programs to accomplish higher transfer rates. Assuming DCPC is inactive, data transfer rates of 2.28 Mbytes for input and 3.17 Mbytes for output can be achieved.



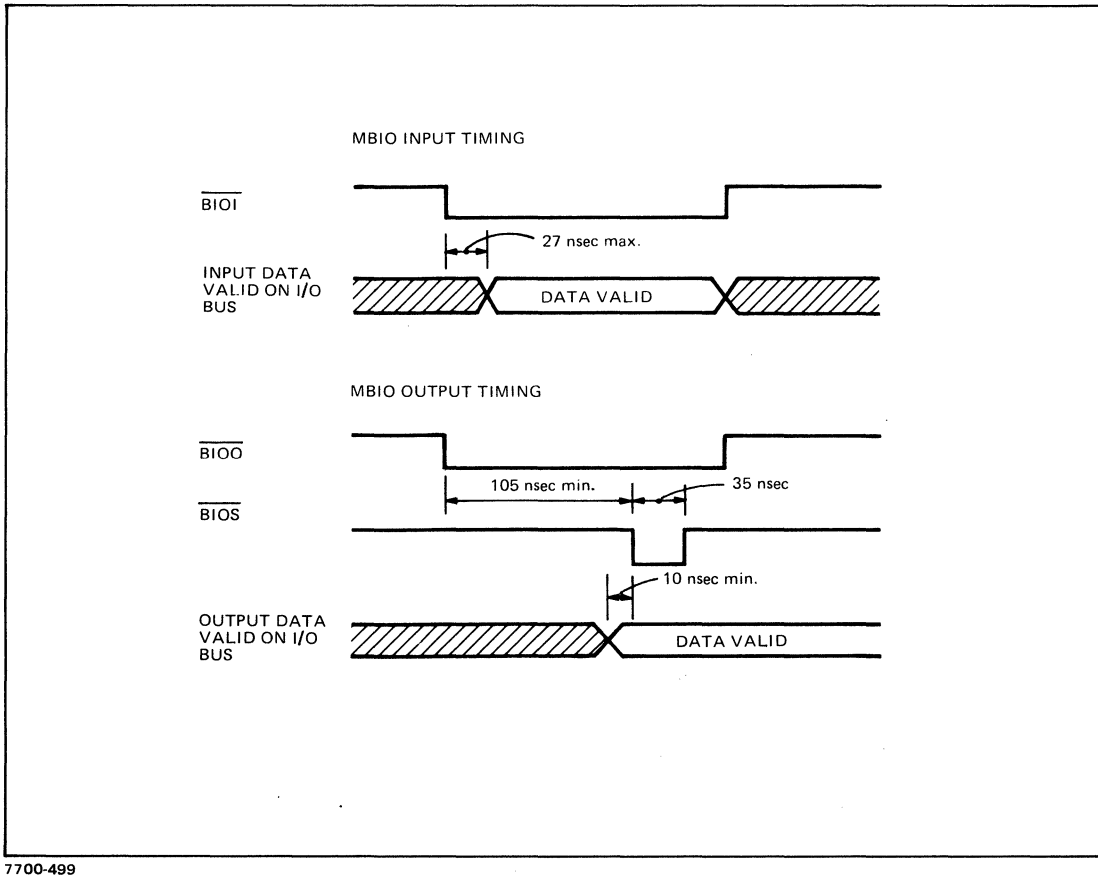
7700-498

Figure 6-4. Microgrammed Block I/O Flow Diagram

Table 6-10. High Speed MBIO Transfer

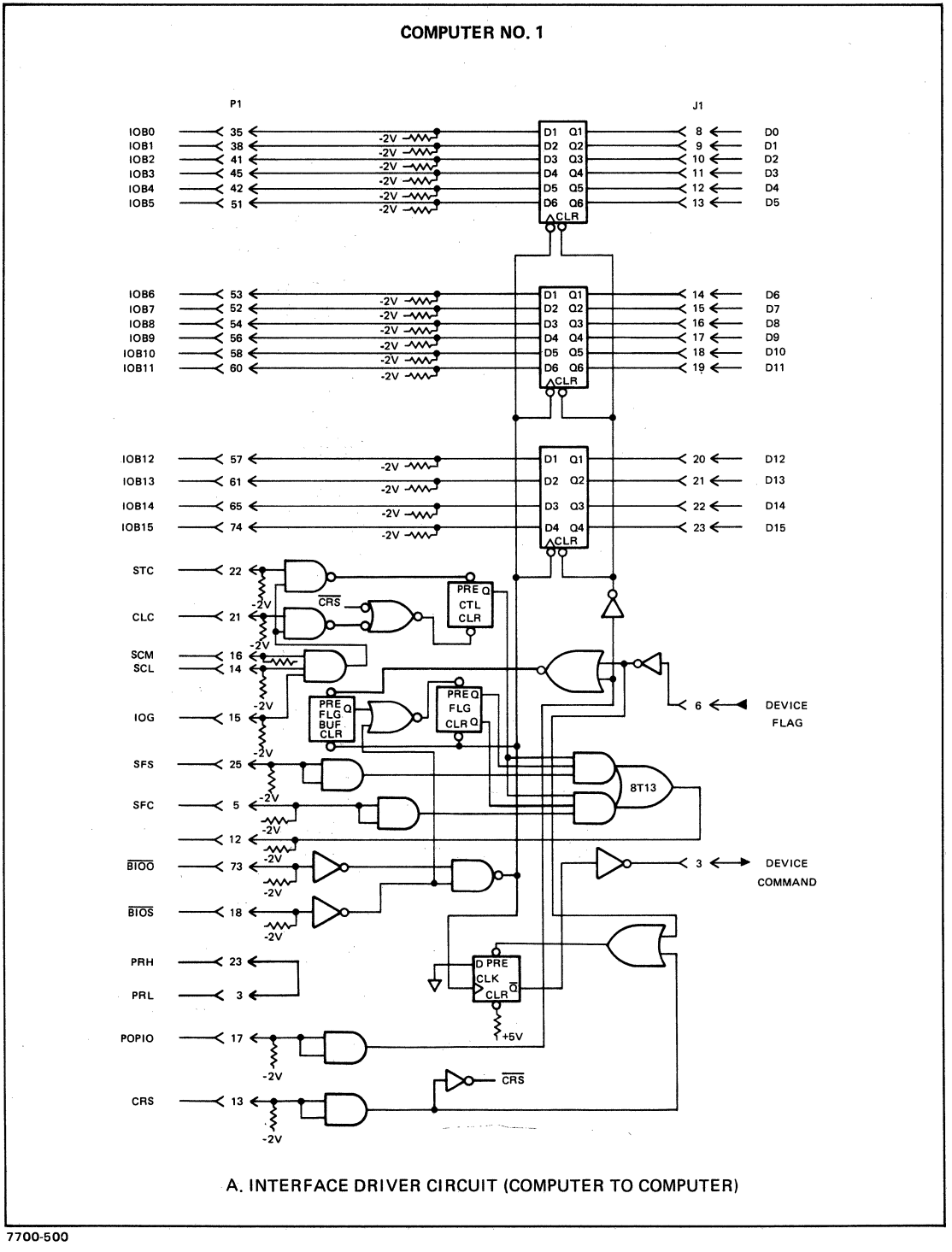
LABEL	OP/ BRCH	SPCL	ALU/ MOD/ COND	STR	S-BUS/ OPRD/ ADDRESS	COMMENTS
INPUT	READ*	INC	PNM	P		
		DCNT	PASS	S1	IOI	Input data word
	WRITE		PASS	TAB	S1	Write into memory
	JMP	CNDX	CNT8	RJS	INPUT	Continue or stop.
OUTPUT		PRST	INC	PNM	P	
	READ	DCNT	PASS	IOI	TAB	Read next word, output present word. Continue or
	JMP	CNDX	CNT8	RJS	OUTPUT	stop.

* Dummy read.



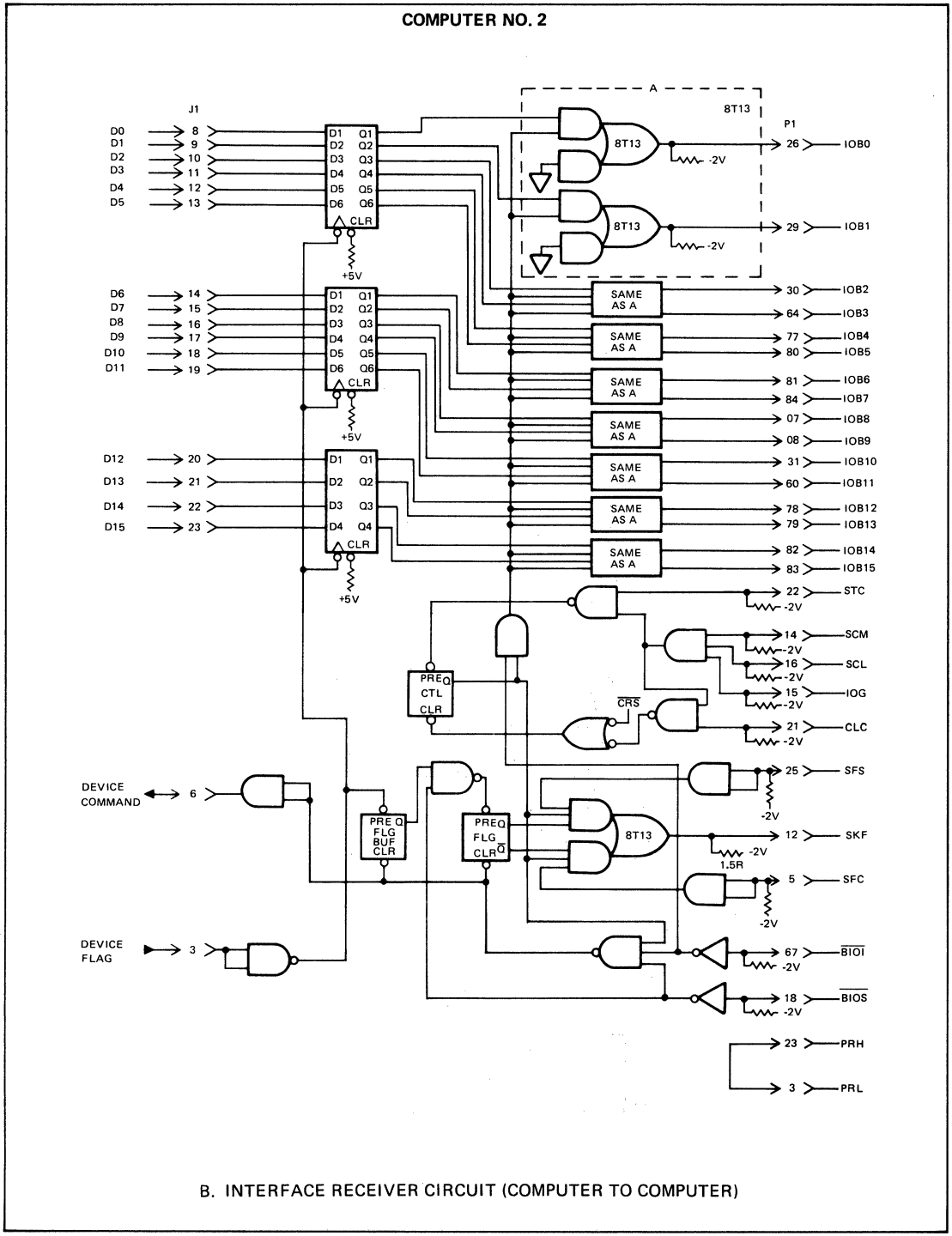
7700-499

Figure 6-5. MBIO Timing Considerations



7700-500

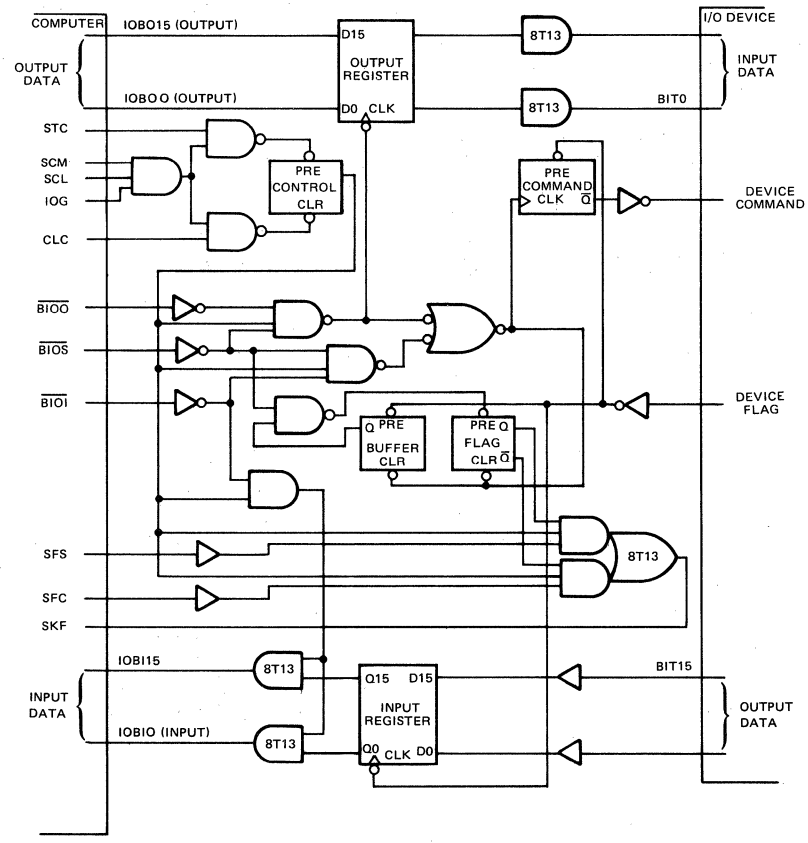
Figure 6-6A. Typical Microprogrammed Block I/O Interface Circuits



B. INTERFACE RECEIVER CIRCUIT (COMPUTER TO COMPUTER)

7700-501

Figure 6-6B. Typical Microprogrammed Block I/O Interface Circuits



C. DUPLEX REGISTER MBIO INTERFACE (COMPUTER TO I/O DEVICE)

7700-502

Figure 6-6C. Typical Microprogrammed Block I/O Interface Circuits

6-16. MICROPROGRAMMABLE PROCESSOR PORT INTERFACING

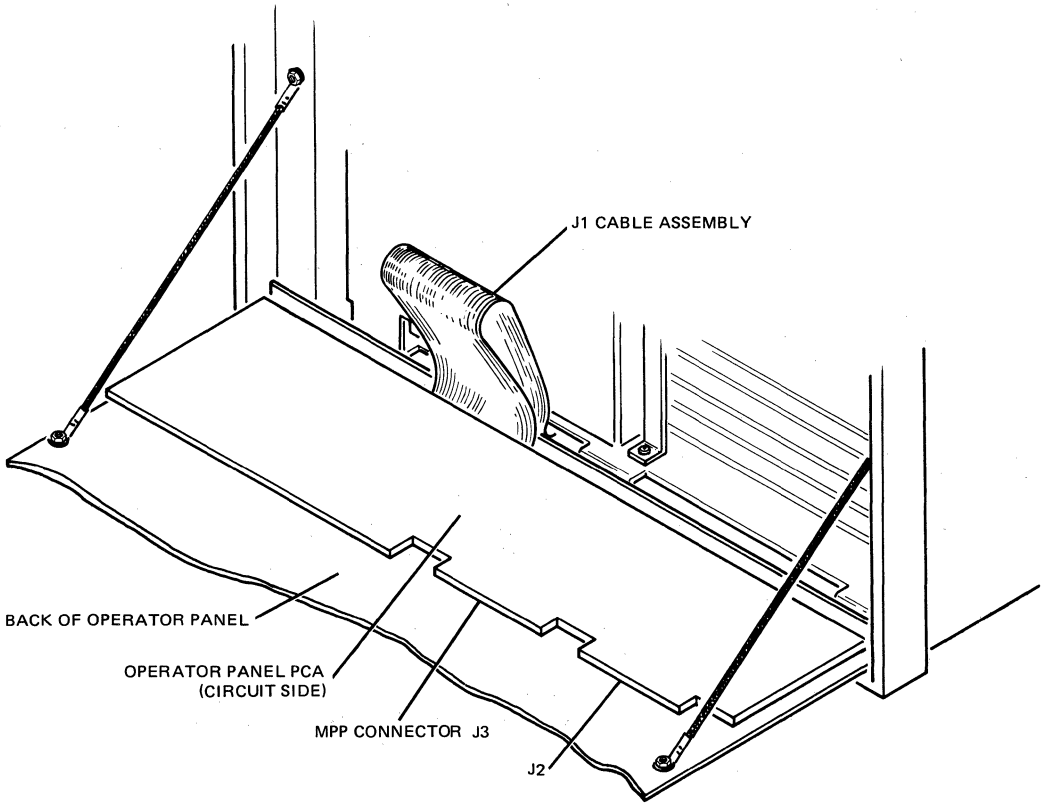
Although it is not part of the computer's I/O Section, the Microprogrammable Processor Port (MPP) permits external hardware to be connected directly to the CPU of the E-Series Computer and can be interfaced under fast and direct microprogrammed control. Some typical applications with the MPP include computer-to-computer communication, adaptation of specialized performance hardware (e.g., floating point hardware), and a low-cost, high-speed or special I/O channel. Access to the computer is accomplished by fabricating an interconnecting cable assembly consisting of standard flat cable and mating connectors. This cable assembly connects between the external device or interface PCA and the computer's MPP connector J3 located behind the computer's operator panel on Operator Panel PCA A2 as shown in Figure 6-7. (Maximum interconnecting cable assembly length is restricted to 6.0 feet (1.8 meters.) It should be noted that the MPP employs tri-state logic and that compatible devices must be used for interface design. Some recommended devices are as follows:

74S240	Octal Inverter
74S241	Octal Buffer
74S373	Octal Latch
74S374	Octal Flip-Flop

Signal definitions and connector pin assignments for MPP connector J3 are contained in Table 6-10. A timing diagram for the MPP signals is contained in Figure 6-8. It should be noted that the actual use of the MPP signals must be determined by the user. When a use for a particular signal is stated in Table 6-11, it is a suggested use only and is not restrictive. As previously stated, it is imperative that the user be completely acquainted with the contents of the *HP 1000 E-Series Computer Microprogramming Reference Manual,* part no. 02109-90004 before attempting to use the MPP. Note that the same microprogramming restriction applies to MPP transfers that applies to MBIO transfers. Namely, the data on the MPP must be passed into a scratchpad register and then the scratchpad register is passed into the T-register. This assures that data words are not lost and that only one handshake acknowledgement occurs per transfer.

For reference purposes, a listing illustrating how to form and execute a microprogrammed MPP I/O transfer is contained in Table 6-11. It should be noted that Table 6-11 is provided for example purposes only and that actual transfer instruction formats will vary depending on the external I/O device's specifications and the user's application.

The microprogram listed in Table 6-11 inputs data words in a burst manner via the MPP and stores the data in main memory. The listed microprogram is interruptible before the word burst begins, but is not interruptible during the burst. Any interrupts that occur during the word burst transfer will be serviced at the end of the microprogram. The listed microprogram has a maximum transfer rate of approximately 500 kilowords/second in a typical DCPC environment and of approximately 1500 kilowords/second in a non-DCPC environment. The microprogram listed in Table 6-11 assumes that the external device contains a data buffer large enough to hold the specified word burst and that the positive word count (number of words to be transferred) has been previously entered into the A-register. (In order to obtain the transfer rates specified above, the listed microprogram is limited to 256 words. Word bursts greater than 256 can be transferred; however, this requires a second word counter which requires additional microinstructions. This in turn decreases the data transfer rate.) The listed microprogram also assumes that the buffer address (starting address of memory where data is to be stored) has been previously entered into the B-register. It should be noted that it is the programmer's responsibility to precede the microprogram listed in Table 6-11 with the required main memory/control memory linkage.



0330-20

Figure 6-7. MPP Connector Location

Table 6-11. MPP Connector J3 Signal Definitions and Connector Pin Assignments

PIN NO.	SIGNAL MNEMONIC AND DEFINITIONS	
1**	$\overline{\text{STOV}}$:	"Not" Set Overflow. This is a ground-true signal used to set the control processor's Overflow flip-flop. Processor loading is 14.0 mA.
3*	PP5:	Processor Port P5. This is a positive-true signal derived from CPU freezeable P5 (Figure 4-4) used to synchronize data flow between the computer and the external device.
5	MPPIO11:	Buffered S-bus bit 11. (Refer to Note 1.)
7*	PP2SP:	Processor Port "2" Special. This is a user-defined, positive-true signal that goes high when micro-order MPP2 is in the Special Field of a microinstruction. Permitted load is 6.0 mA.
9**	$\overline{\text{MPP}}$:	"Not" Microprogrammable Processor Port. This is a user-defined, ground-true signal that can be sensed by the control processor when micro-order MPP is in the Condition Field of a micro-instruction. This signal must be asserted throughout the microinstruction cycle. Processor loading is 2.0 mA.
11*	$\overline{\text{PP1SP}}$:	"Not" Processor Port "1" Special. This is a user-defined, ground-true signal that goes low when micro-order MPP1 is in the Special Field of a microinstruction. Permitted load is 6.0 mA.
13	MPPIO12:	Buffered S-bus bit 12
15	MPPIO13:	Buffered S-bus bit 13
17	MPPIO14:	Buffered S-bus bit 14
19	MPPIO15:	Buffered S-bus bit 15
21	MPPIO08:	Buffered S-bus bit 8
23	MPPIO09:	Buffered S-bus bit 9
25	MPPIO07:	Buffered S-bus bit 7
		Refer to Note 1.
27*	MPBST:	Microprogrammable Processor Port "B" Store. This is a positive-true signal that goes high when micro-order MPPB is in the Store Field of a microinstruction. Can be used to strobe data on the S-bus into the external device. Permitted load is 6.0 mA.
29*	$\overline{\text{PLR0}}$:	"Not" Processor Port L-Register Bit 0. This is a ground-true signal that can be used as an address line to the external device. Permitted load is 6.0 mA. The buffered signal is true whenever LR0 (least significant bit of L-register) is true.
31*	PIRST:	Processor Port Instruction Register Store. This is positive-true signal that goes high when micro-order IRCM is in the Store Field of a microinstruction. Can be used by external device for recognition of special instructions. Permitted load is 6.0 mA.
33	MPPIO06:	Buffered S-bus bit 6
35	MPPIO05:	Buffered S-bus bit 5
37	MPPIO04:	Buffered S-bus bit 4
39	MPPIO03:	Buffered S-bus bit 3
		Refer to Note 1.
41*	MPBEN:	Microprogrammable Processor Port "B" Enable. This is a positive-true signal that goes high when micro-order MPPB is in the S-Bus Field of a microinstruction. Can be used to load data from the external device. Permitted load is 6.0 mA.
43	MPPIO02:	Buffered S-bus bit 2
45	MPPIO01:	Buffered S-bus bit 1
47	MPPIO00:	Buffered S-bus bit 0
49	MPPIO10:	Buffered S-bus bit 10
		Refer to Note 1.

NOTES:
1. All S-bus signals are bidirectional.
2. All even-numbered pins (2 through 50) are connected to ground.
* Signal generated by computer for external device.
** Signal generated by external device for computer.

Table 6-12. MPP Word Burst Input Microprogram

LABEL	OP/ BRCH	SPCL	ALU/ MOD/ COND	STR	S-BUS/ OPRD/ ADDRESS	COMMENTS
			• • •			
BURSTIN			DEC	S3	P	Stores contents of P-register in Scratch Register S3 for reentry point.
				CNTR	A	Stores the positive word count in the Instruction Register.
	JMP	CNDX	ALZ PASS		DONE	
WAIT	JMP	CNDX	HOI		INTRPT	Any interrupts pending? Yes; jump to interrupt microroutine. No; continue.
		NOP				I/O device's data ready (i.e., $\overline{\text{MPP}}$ signal true)? No; repeat previous instruction. Yes; continue.
	JMP	CNDX	MPP	RJS	WAIT	
			INC	PNM	B	Loads starting buffer address in M-register and loads next buffer address in P-register.
				S1	MPPB	Performs memory protect check of M-register address for memory protect fence or DMS violation. Strobes data from the I/O device onto the S-bus (MPBEN signal true) and writes the data into main memory address specified by contents of M-register.
BURST	WRTE	MPCK		TAB	S1	
		DCNT	INC	PNM	P	Decrements the Instruction Register (word count), loads next buffer address from P-register into M-register and increments the P-register.
	JMP	CNDX	CNT8	RJS	BURST	Word count zero? No; jump to BURST. Yes; continue.
DONE				B	P	Begins exit routine by reading next instruction from main memory address specified in M-register (original P-register contents). B-register contains last buffer address. A-register contains all zeros indicating that the word burst is complete.
	READ		INC	PNM	S3	
		RTN		A	CNTR	
INTRPT			PASS	P	S3	Store microprogram reentry address into P-register and exit to Halt-Or-Interrupt Microroutine.
	JMP				6	
			• • •			

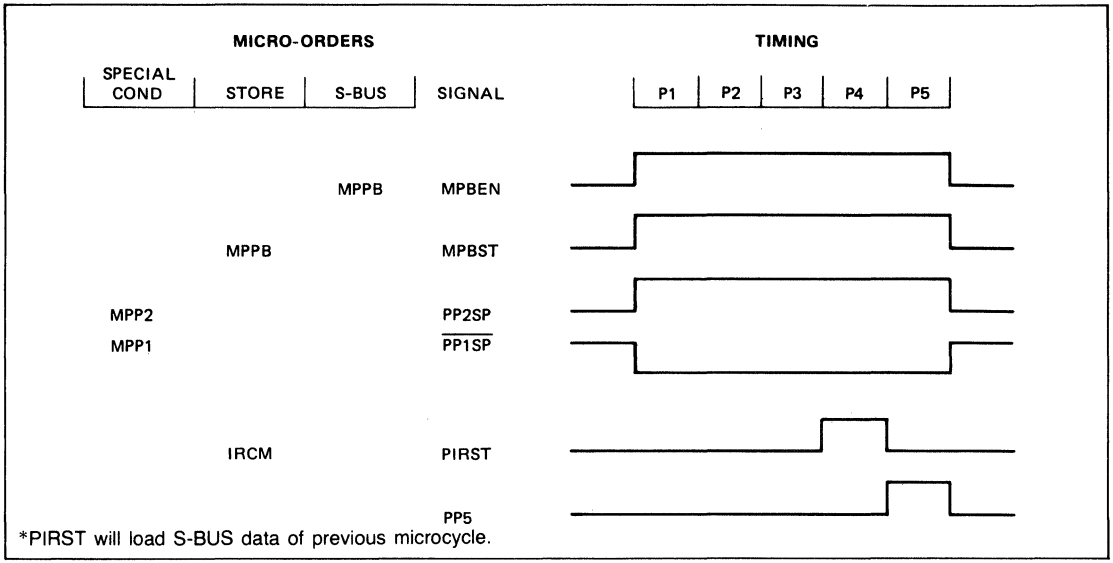
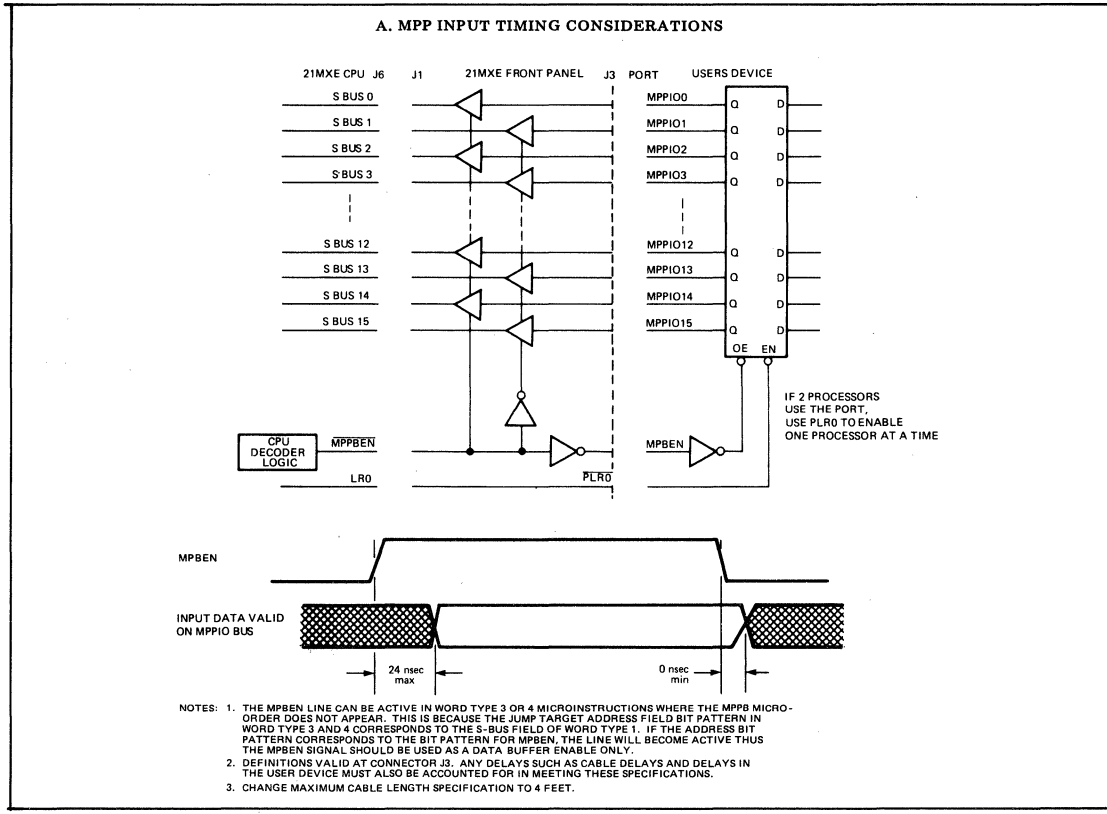


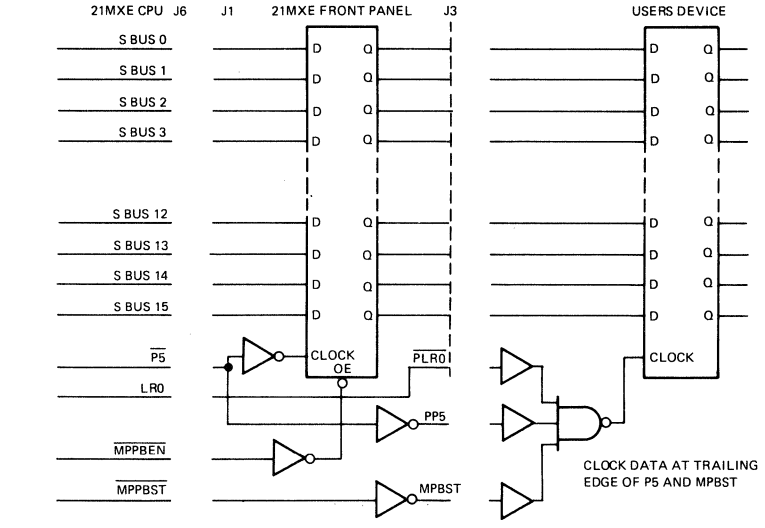
Figure 6-8. MPP Timing Diagram

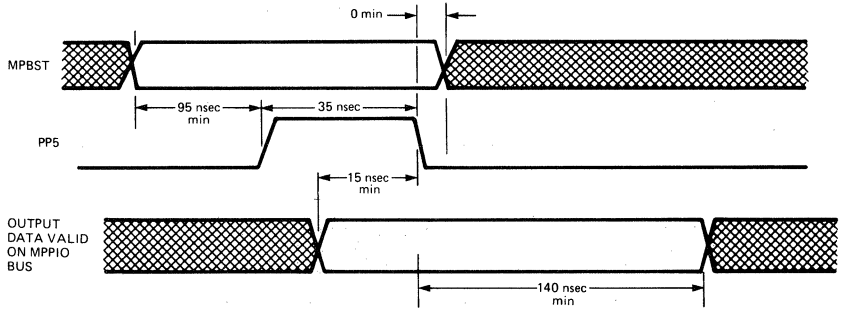


0330-21

Figure 6-9. MPP Timing Considerations (Sheet 1 of 4)

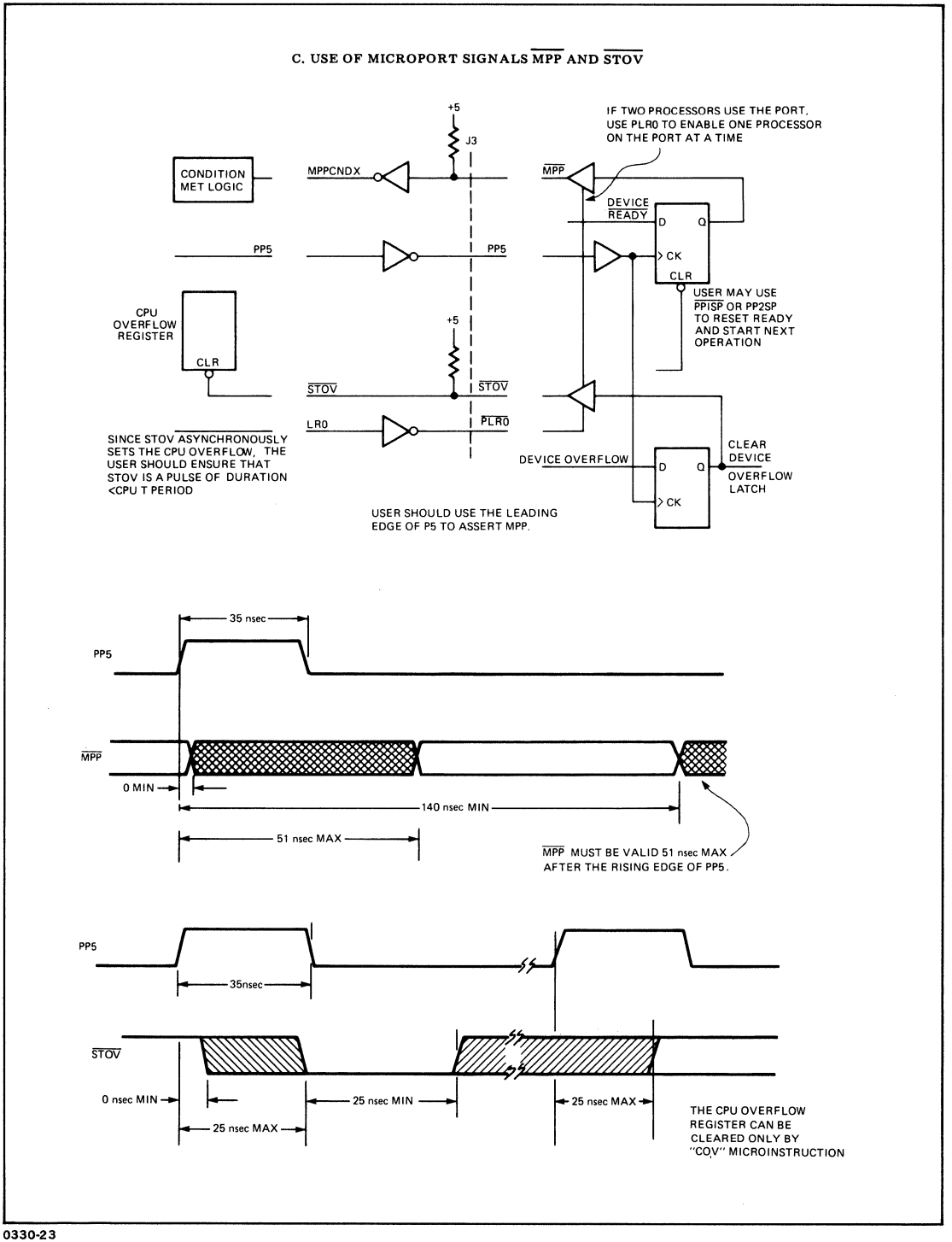
B. MPP OUTPUT TIMING CONSIDERATIONS





0330-22

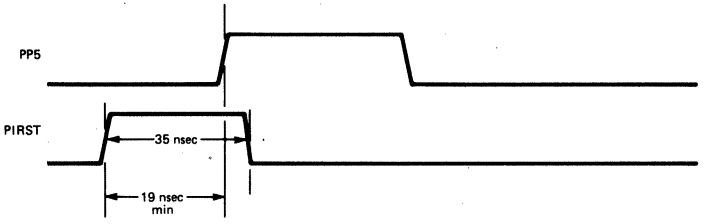
Figure 6-9. MPP Timing Considerations (Sheet 2 of 4)



0330-23

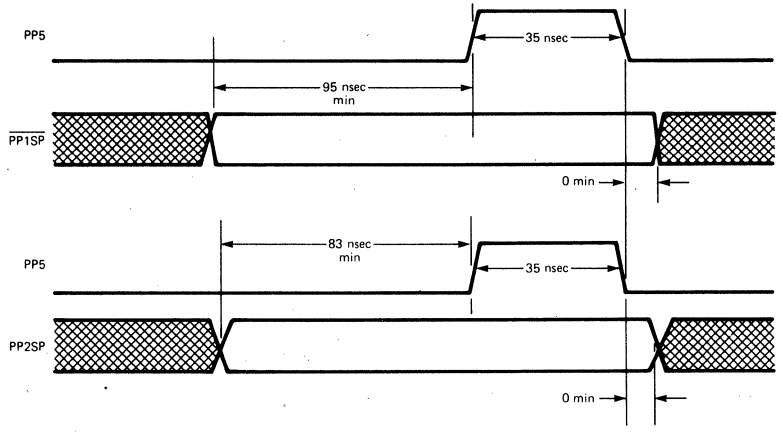
Figure 6-9. MPP Timing Considerations (Sheet 3 of 4)

D. PIRST TIMING



0330-24

E. $\overline{\text{PP1SP}}$ AND PP2SP TIMING



NOTE:
THERE IS A DECODER SETTLING TIME AT THE BEGINNING OF EVERY MICROCYCLE DURING WHICH TIME $\overline{\text{PP1SP}}$ AND PP2SP ARE INDETERMINATE. THE SETUP TIME INDICATES THE START OF THE STABLE PERIOD. IF THE INDETERMINATE PERIOD POSES DIFFICULTY, $\overline{\text{MPP1SP}}$ AND PP2SP CAN BE QUALIFIED WITH PP5.

0330-25

Figure 6-9. MPP Timing Considerations (Sheet 4 of 4)

HP INTERFACE KITS

APPENDIX A

This appendix contains condensed general descriptions of general-purpose interface kits currently available from Hewlett-Packard. Unless otherwise specified, the interface kits listed are compatible with the HP 1000 M-Series, E-Series and F-Series Computers. Additional information for these interface kits is available at any of the Hewlett-Packard Sales and Service Offices listed at the back of this manual.

A-1. HP 12531C BUFFERED TELEPRINTER INTERFACE KIT

The HP 12531C interfaces HP 1000 Computers to HP 2752A and HP 2754B Teleprinters using current loop signals. Optional features permit interfacing to a variety of EIA compatible devices, including Bell 103 Type data sets or equivalent (manual mode only). Five jumper-selectable data transfer rates (110, 220, 440, 880, and 1760 bits per second) are available. A second jumper permits control of the data transfer rate up to a maximum of 2400 bits per second by an external clock from the associated I/O device.

A-2. HP 12531D TERMINAL INTERFACE KIT

The HP 12531D interfaces HP 1000 Computers to a variety of terminal devices. The standard interface permits interfacing with I/O devices using current loop signals. Optional features permit interfacing to a variety of EIA compatible devices, including Bell 103 Type data sets or equivalent (manual mode only), HP 2640A Interactive Display Terminals, and HP 2644A Mini Data Stations. Five jumper-selectable data transfer rates (150, 300, 600, 1200, and 2400 bits per second) are available. A second jumper permits control of the data transfer rate up to a maximum of 9600 bits per second by an external clock from the associated I/O device.

A-3. HP 12551B RELAY REGISTER INTERFACE KIT

The HP 12551B interfaces HP 1000 Computers to external devices that require floating contact closures. The standard interface provides 16 floating contact closures that can be used to control one device or, subdivided in any combination, to control several devices. The opening and closing of each set of relay contacts is under computer program control and the voltages switched through the relay contacts can differ from each other and from computer ground by as much as 100V peak. The relay contacts can be connected in series, parallel, or series-parallel, with or without diode isolation. An optional feature permits data to be read back into the computer from the interface's storage register.

A-4. HP 12554A 16-BIT DUPLEX REGISTER INTERFACE KIT

The HP 12554A interfaces HP 1000 Computers to a variety of digital I/O devices. The interface permits 16-bit input, output, or combined input/output operations between a computer and its associated I/O device by providing a 16-bit input storage register, a 16-bit output storage register, and all required control and interrupt logic.

A-5. HP 12555B DIGITAL-TO-ANALOG CONVERTER INTERFACE KIT (M-SERIES ONLY)

The HP 12555B interfaces HP 1000 M-Series Computers to a variety of external I/O analog devices. The interface receives 16-bit binary words from the computer, divides each 16-bit word into two 8-bit bytes, and stores these words in two 8-bit registers. Outputs from the two 8-bit registers are scaled to provide two analog output voltages that are used as the X- and Y-axis input signals to the external analog device. The magnitude of each analog output voltage is given by 10N/255, where N is the decimal value represented by the combination of bits in each group of eight bits from the computer. For conventional oscilloscopes, the analog output signals are regenerated every 20 milliseconds to refresh the display. For storage type I/O devices, an Erase signal is generated to remove a previously generated display. Positive or negative blanking signals are also generated that can be connected to the Z-axis input of the device to provide the display after the interface circuits have stabilized. The interface also accepts a Device Flag signal from the external device that indicates when the device is ready to receive new data.

A-6. HP 12556B 40-BIT OUTPUT REGISTER INTERFACE KIT (M-SERIES ONLY)

The HP 12556B interfaces with HP 1000 M-Series Computers and has a 40-bit output capacity. Its capabilities

include driving digital recorders such as the HP 5055A and HP 5050B or equivalent, driving program input lines of stimulus or measuring instruments, and driving control panel indicators or control lines. The interface's 40-bit output register offers two jumper-selectable output modes; ASCII and binary. In ASCII mode, the register assembles the BCD position of ASCII characters from six words in computer memory. In binary mode, the register assembles the output from three words in computer memory.

A-7. HP 12560A DIGITAL PLOTTER INTERFACE KIT

The HP 12560A interfaces HP 1000 Computers to the California Computer Products (CALCOMP) Model 565 or 563 Digital Incremental Plotter. The interface provides control, interrupt, and output logic circuits for computer program control of the plotter. When properly programmed, the interface accepts any combination of parallel six bits from the computer and applies these six bits to the plotter for control of the drum, pen, and pen carriage assemblies. Drive capability for either the Model 565 or 563 is jumper selectable.

A-8. HP 12566B MICROCIRCUIT INTERFACE KIT

The HP 12566B interfaces HP 1000 Computers to a variety of digital measurement devices with DTL/TTL output voltage levels. The interface permits 16-bit input and output information flow between the computer and its associated I/O device at data speeds much greater than can be achieved with discrete components.

A-9. HP 12587B ASYNCHRONOUS DATA SET INTERFACE KIT (M-SERIES ONLY)

The HP 12587B interfaces HP 1000 M-Series Computers to common carrier data transmission equipment or, as an optional feature, to a computer terminal. During transmit operations, the interface converts parallel data output from the computer into serial data that is compatible with a data set or computer terminal. During receive operations, the interface converts serial data output from a data set or computer terminal into parallel data for computer input. The interface can provide asynchronous communications at speeds up to 3110 bits per second. The data rates are jumper-selectable and programmable functions include character size, parity generation, parity checking, and the selection of one or two stop bits.

A-10. HP 12589A AUTOMATIC DIALER INTERFACE KIT (M-SERIES ONLY)

The HP 12589A interfaces the HP 1000 M-Series Computers to a Bell Auxiliary Data Set 801 Automatic Calling Unit. The automatic calling unit permits the computer to dial telephone numbers under program control to access a remote terminal for data transmission. Automatic calling can be used with either asynchronous or synchronous interface kits.

A-11. HP 12597A 8-BIT DUPLEX REGISTER INTERFACE KIT

The HP 12597A interfaces HP 1000 Computers to a variety of external I/O digital devices. The interface permits 8-bit input, output, or combined input/output operations between a computer and its associated I/O device by providing an 8-bit input storage register, an 8-bit output storage register, and all required control and interrupt logic.

A-12. HP 12604B DATA SOURCE INTERFACE KIT

The HP 12604B interfaces HP 1000 Computers to a variety of digital measurement devices. The interface provides a 32-bit capacity and, as such, can transfer up to eight BCD digits from counters, digital voltmeters, etc. to the computer. The interface employs referenced capacitive coupling to accommodate input logic voltage levels from −100V to +100V.

A-13. HP 12618A SYNCHRONOUS DATA SET INTERFACE KIT (M-SERIES ONLY)

The HP 12618A interfaces HP 1000 M-Series Computers to data communication networks equipped with high-speed synchronous data sets such as the Bell 201 Type or equivalent. Using fully independent transmit and receive channels, the interface can operate in either half or full duplex mode at data transfer rates up to 9600 bits per second. Under program control, the interface also provides selection of parity generation and checking, a synchronization character, character size, and a special character recognition/interrupt capability.

A-14. HP 12620A BREADBOARD INTERFACE KIT

The HP 12620A is a single plug-in I/O interface PCA that contains the standard HP flag and interrupt circuits required by the HP 1000 Computers. The interface permits

users to design and add to the PCA special circuits required to interface unique I/O devices to HP computers.

A-15. HP 12880A TERMINAL INTERFACE KIT

The HP 12880A interfaces HP 1000 Computers to console type terminals and provides jumper-selectable data transfer rates up to a maximum of 9600 bits per second. The standard interface is supplied with a cable suitable for connecting to most EIA terminals. An optional cable permits direct connections to the HP 2640 Interactive Display Terminal and to the HP 2644A Mini Data Station.

A-16. HP 12889A HARDWIRED SERIAL INTERFACE KIT (M-SERIES ONLY)

The HP 12889A enhances the data communication capability of the HP 1000 M-Series Computers. The interface enables high-speed, asynchronous, long distance, point-to-point data transfer between two HP 1000 M-Series Computers. The interface operates in any of four data handling modes; program to program, program to DCPC, DCPC to program, and DCPC to DCPC.

A-17. HP 12920B ASYNCHRONOUS MULTIPLEXER

The HP 12920B interfaces HP 1000 Computers to provide multiplexed I/O capability for up to 16 communications devices at programmable data rates up to 2400 bits per second. The standard interface provides multiplexed I/O capability for up to 16 Bell 103 Type data sets or bit serial EIA RS232 compatible terminals such as teleprinters, card readers, or similar devices. Optional features provide for up to 16 Bell 202 Type data sets or up to eight Bell 801 Type automatic dialers. All input and output channels are independent so that full duplex and split-speed devices can be interfaced.

A-18. HP 12930A UNIVERSAL INTERFACE KIT

The HP 12930A interfaces HP 1000 Computers to a wide variety of external I/O devices. Programmable switches provide the versatility required for the interface to accommodate most I/O interface requirements. The interface's dual channel design provides the capability for transferring large blocks of data over distances up to 500 feet. Optional features provide a choice of either ground-true or positive-true TTL logic in place of the standard differential logic.

A-19. HP 12936A PRIVILEGED INTERRUPT FENCE ACCESSORY

The HP 12936A is a single printed-circuit assembly that can be installed in one of the I/O slots in an HP 1000 Computer to provide a programmable I/O barrier between high and low priority I/O devices. It contains all required circuitry to control both the generation of interrupts and to inhibit the priority line to lower priority devices under program control.

A-20. HP 12966A BUFFERED ASYNCHRONOUS DATA COMMUNICATIONS INTERFACE

The HP 12966A interfaces HP 1000 Computers to asynchronous, bit-serial, EIA RS232C compatible data sets or terminals. The interface permits the selection of parity generation and checking (even, odd, or none), selection of character size (five or eight bits), selection of number of stop bits (one or two), and a selection of data transfer rates from 50 to 9600 bits per second all under program control.

A-21. HP 12967A SYNCHRONOUS COMMUNICATIONS INTERFACE (M-SERIES ONLY)

The HP 12967A interfaces HP 1000 M-Series Computers to any EIA RS232C compatible data set and provides half-duplex, synchronous, bit-serial, data communications at transfer rates up to 20,000 bits per second. Under program control, the interface permits the selection of parity generation and checking and the ability to transfer data under either program control or DCPC control. Character size is fixed at eight bits and a switch selectable synchronization character permits automatic synchronization of incoming data.

A-22. HP 12968A ASYNCHRONOUS COMMUNICATIONS INTERFACE

The HP 12968A interfaces HP 1000 Computers to EIA RS232C compatible, asynchronous data sets and terminals and provides half-duplex, asynchronous, bit-serial data communications at transfer rates up to 9600 bits per second. The HP 12968A is an economical, low-power version of the HP 12966A and is identical in every respect to the HP 12966A except that it has a two-character buffer and no special character capability.

A-23. HP 59310B HP-IB INTERFACE KIT

The Hewlett-Packard Interface Bus (HP-IB) is Hewlett-Packard's implementation of IEEE Standard 488-1975,

Digital Interface for Programmable Instrumentation. The HP 59310B interfaces the HP 1000 Computers to the HP-IB which provides a two-way digital communications structure for one or more instruments with ASCII-compatible interfaces. The interface makes the following bus functions available to the computer: listen and talk functions, serial or parallel poll identification, controller clearing, and four types of interrupt flagging. Data transfers are byte-serial and bit-parallel (8-bit bytes). Data transfers can be accomplished under either program or DCPC control.

A-24. HP 91000A PLUG-IN 20 KHZ ANALOG-TO-DIGITAL INTERFACE SUBSYSTEM

The HP 91000A is a complete computer-controlled data aquisition subsystem that can be connected into an I/O slot of either an HP 1000 Computer. Once installed, the interface can, under program control, scan the outputs of multiple external analog devices, convert the analog signals into 12-bit two's complement binary representation, and return the binary number to the computer for processing. Jumper selections permits the interface to be configured to accept either 8 differential or 16 single-ended analog signals in the range of + 10.235 to − 10.240 volts.

A-25. HP 91200B TV INTERFACE KIT

The HP 91200B interfaces HP 1000 M-Series Computers to both black-and-white and color television monitors. Under program control, the interface generates a composite video signal to provide displays that combine both graphic images and alphanumeric characters on television monitors. The interface is compatible with either American or European standard broadcast TV scan rates and, in addition, can supply non-standard scan rates to optimize its operation with television monitors operating with 60-Hz vertical rates.

I/O SIGNAL DEFINITIONS

APPENDIX B

This appendix contains a list of definitions for the signals available on the I/O backplane. The list is arranged in signal mnemonic, alphabetical order. Connector pin number assignments for the signals are contained in Table 4-1. The signals are generated at the T-period times illustrated in Figures 4-5 and 4-8. Program control of the signals and how they interrelate are discussed in Sections III and IV.

$\overline{\text{BIOI}}$: "Not" Block I/O Input. Used to strobe data from the I/O interface PCA into the computer during microprogrammed I/O transfers. (Refer to BIOS.) The BIOI signal is true when micro-order IOI is in the S-Bus Field of a microinstruction and no micro-order IOG is in the two preceding Special Fields.

$\overline{\text{BIOO}}$: "Not" Block I/O Output. Used to strobe data from the computer into the I/O interface PCA during microprogrammed I/O transfers. (Refer to BIOS.) The BIOO signal is true when micro-order IOO is in the Store Field of a microinstruction and no micro-order IOG is in the two preceding Special Fields.

$\overline{\text{BIOS}}$: "Not" Block I/O Strobe. Used in conjunction with BIOI and BIOO signals during microprogrammed I/O transfers. During output transfers, data is valid on the I/O bus at the trailing edge of BIOS. During input transfers, data must be enabled onto the I/O bus at BIOI time. BIOI•BIOS verifies completion of the input transfer.

CLC: Clear Control flip-flop. Used to clear addressed I/O interface PCA's Control and Command flip-flops. The CLC signal is generated by a CLC instruction.

CLF: Clear Flag flip-flop. Used to clear addressed I/O interface PCA's Flag Buffer and Flag flip-flops. The CLF signal is generated by a CLF instruction.

CRS: Control Reset. Used to clear all I/O interface PCA's Control flip-flops. The CRS signal can be generated by either a CLC instruction addressed to select code 00, a false PON signal, or by pressing the Operator Panel PRESET switch.

EDT: End Data Transfer. Used during DCPC transfers to notify the I/O device that a data transfer is complete. The EDT signal is generated when the number of transferred words counted by the DCPC Word Count Register equals the number of words specified in the programmed block length.

ENF: Enable Flag. Used during I/O operations to time the setting of all I/O interface PCA's Flag flip-flops. The ENF signals is generated by a buffered T2 time signal.

FLG: Flag. Used in conjunction with the addressed I/O interface PCA's IRQ signal to initiate an interrupt for an I/O device. The FLG signal is generated when the addressed I/O interface PCA receives a combination of programmed I/O control signals from the computer and a Device Flag signal from the I/O device. This signal is also used to define the SCM octal digit for the interrupt address.

IAK: Interrupt Acknowledge. Used to clear the addressed I/O interface PCA's Flag Buffer flip-flop to prevent a second interrupt from occurring for the same IRQ and FLG signals. The IAK signal is generated after the computer has encoded the interrupt address and is under control of the instruction stored in the trap cell.

IEN: Interrupt Enable. Used to enable or disable all I/O interface PCA's IRQ flip-flops. The IEN signal is controlled by STF and CLF instructions addressed to select code 00.

IOG: I/O Group. Used in conjunction with SCM and SCL signals to enable the addressed I/O interface PCA. The IOG signal is generated whenever an I/O group instruction is initiated.

IOI: I/O Data Input. Used to strobe data from the addressed I/O interface PCA into the computer. The IOI signal is generated by either an LIA, LIB, MIA, or MIB instruction.

IOO: I/O Data Output. Used to strobe data from the computer into the addressed I/O interface PCA. The IOO signal is generated by either an OTA or OTB instruction.

IRQ: Interrupt Request. Used in conjunction with the addressed I/O interface PCA's FLG signal to initiate an interrupt for an I/O device. The IRQ signal is generated when the addressed I/O interface PCA receives a combination of programmed I/O control signals from the computer and a Device Flag signal from the I/O device. This signal is also used to define the SCL octal digit for the interrupt address.

PON: Power On Normal. Used as a master reset signal for the entire computer and, when false, generates the CRS and POPIO signals for all I/O interface PCA's. A false PON signal is generated when power is initially applied to the computer.

POPIO: Power On Preset to I/O. Used to set all I/O interface PCA's Flag Buffer flip-flops. The POPIO signal is generated by either a false PON signal or by pressing the Operator Panel PRESET switch.

PRH: Priority High. Used in conjunction with the PRL signal to maintain the priority chain between all I/O interface PCA's. The PRH signal is high whenever no I/O interface PCA's with a higher priority are requesting an interrupt.

PRL: Priority Low. Used in conjunction with the PRH signal to maintain the priority chain between all I/O interface PCA's. The PRL signal is high whenever the I/O interface PCA is not requesting an interrupt and no I/O interface PCA's with a higher priority are requesting an interrupt.

RUN: Run. For HP 1000 M-Series Computers, the RUN signal reflects the state of the CPU Run flip-flop. For HP 1000 E-/F-Series Computers, the RUN signal can be used for remote control of an unattended or inaccessible computer. (Refer to paragraph 5-10.)

SCL: Select Code Least Significant Digit. Used in conjunction with the SCM signal to determine which I/O interface PCA is to receive an I/O instruction. The SCL, SCM, and IOG signals must all be true in order to enable an I/O interface PCA. The SCL signal is generated by decoding bits 2 — 0 of an I/O instruction into an octal digit.

SCM: Select Code Most Significant Digit. Used in conjunction with the SCL signal to determine which I/O interface PCA is to receive an I/O instruction. The SCM, SCL, and IOG signals must all be true in order to enable an I/O interface PCA. The SCM signal is generated by decoding bits 5 — 3 of an I/O instruction into an octal digit.

SFC: Skip if Flag is Clear. Used in conjunction with the SKF signal to test if the addressed I/O interface PCA's Flag flip-flop is clear. The SFC signal is generated by an SFC instruction.

SFS: Skip if Flag is Set. Used in conjunction with the SKF signal to test if the addressed I/O interface PCA's Flag flip-flop is set. The SFS signal is generated by an SFS instruction.

SIR: Set Interrupt Request. Used during interrupt processing to time the setting of the I/O interface PCA's IRQ flip-flop. The SIR signal is generated by a buffered T5 time signal.

SKF: Skip on Flag. Used in conjunction with the SFS and SFC signals to indicate the state (set or clear) of the addressed I/O interface PCA's Flag flip-flop. The SKF signal is generated when the addressed I/O interface PCA's Flag flip-flop is set and the SFS signal is true or when the Flag flip-flop is clear and the SFC signal is true.

SRQ: Service Request. Used during DCPC operations to initiate a DCPC cycle. The SRQ signal is generated whenever the addressed I/O interface PCA's Flag flip-flop is set indicating that the associated I/O device is ready for a data transfer.

STC: Set Control flip-flop. Used to set addressed I/O interface PCA's Control and Command flip-flops. The STC signal is generated by an STC instruction.

STF: Set Flag flip-flop. Used to set addressed I/O interface PCA's Flag Buffer flip-flop. The STF signal is generated by an STF instruction.

SALES OFFICES

Arranged alphabetically by country

ANGOLA
Telectra
Empresa Técnica de Equipamentos Eléctricos, S.A.R.L.
R. Barbosa Rodrigues, 41-l°DT.°
Caixa Postal, 6487
Luanda
Tel: 35515/6

ARGENTINA
Hewlett-Packard Argentina S.A.
Santa Fe 2035, Martinez
6140 **Buenos Aires**
Tel: 792-1239, 798-6086
Telex: 122443 AR CIGY

Biotron S.A.C.I.y M.
Avda. Paseo Colon 221
9 piso
1399 **Buenos Aires**
Tel: 30-4846/1851/8384
34-9356/0460/4551
Telex: (33) 17595 BIO AR

AUSTRALIA

AUSTRALIA CAPITAL TERR.
Hewlett-Packard Australia Pty. Ltd.
121 Wollongong Street
Fyshwick, 2609
Tel: 804244
Telex: 62650

NEW SOUTH WALES
Hewlett-Packard Australia Pty. Ltd.
31 Bridge Street
Pymble, 2073
Tel: 4496566
Telex: 21561

QUEENSLAND
Hewlett Packard Australia Pty. Ltd.
5th Floor
Teachers Union Building
495-499 Boundary Street
Spring Hill, 4000
Tel: 2291544

SOUTH AUSTRALIA
Hewlett-Packard Australia Pty. Ltd.
153 Greenhill Road
Parkside, 5063
Tel: 2725911
Telex: 82536

VICTORIA
Hewlett-Packard Australia Pty. Ltd.
31-41 Joseph Street
Blackburn, 3130
Tel: 89-6351
Telex: 31024 MELB

WESTERN AUSTRALIA
Hewlett-Packard Australia Pty. Ltd.
141 Stirling Highway
Nedlands, 6009
Tel: 3865455
Telex: 93859

AUSTRIA
Hewlett-Packard Ges.m.b.H.
Wehlistrasse 29
P.O. Box 7
A-1205 **Vienna**
Tel: 35-16-21-0
Telex: 13582/135066

Hewlett-Packard Ges.m.b.H.
Wehlistrasse, 29
A-1205 **Wien**
Tel: 35-16-21
Telex: 135066

BAHRAIN
Medical Only
Wael Pharmacy
P.O. Box 648
Bahrain
Tel: 54886, 56123
Telex: 8550 WAEL GJ

Al Hamidiya Trading and Contracting
P.O. Box 20074
Manama
Tel: 259978, 259958
Telex: 8895 KALDIA GJ

BANGLADESH
The General Electric Co. of Bangladesh Ltd.
Magnet House 72
Dilkusha Commercial Area
Motijheil, Dacca 2
Tel: 252415, 252419
Telex: 734

BELGIUM
Hewlett-Packard Benelux S.A./N.V.
Avenue du Col-Vert, 1, (Groenkraaglaan)
B-1170 **Brussels**
Tel: (02) 660 50 50
Telex: 23-494 paloben bru

BRAZIL
Hewlett-Packard do Brasil I.e.C. Ltda.
Alameda Rio Negro, 750
Alphaville
06400 **Barueri** SP
Tel: 429-3222

Hewlett-Packard do Brasil I.e.C. Ltda.
Rua Padre Chagas, 32
90000-**Pôrto Alegre**-RS
Tel: 22-2998, 22-5621

Hewlett-Packard do Brasil I.e.C. Ltda.
Av. Epitacio Pessoa, 4664
22471-**Rio de Janeiro**-RJ
Tel: 286-0237
Telex: 021-21905 HPBR-BR

CANADA

ALBERTA
Hewlett-Packard (Canada) Ltd.
11620A - 168th Street
Edmonton T5M 3T9
Tel: (403) 452-3670
TWX: 610-831-2431

Hewlett-Packard (Canada) Ltd.
210, 7220 Fisher St. S.E.
Calgary T2H 2H8
Tel: (403) 253-2713
TWX: 610-821-6141

BRITISH COLUMBIA
Hewlett-Packard (Canada) Ltd.
10691 Shellbridge Way
Richmond V6X 2W7
Tel: (604) 270-2277
TWX: 610-925-5059

MANITOBA
Hewlett-Packard (Canada) Ltd.
380-550 Century St.
St. James,
Winnipeg R3H 0Y1
Tel: (204) 786-6701
TWX: 610-671-3531

NOVA SCOTIA
Hewlett-Packard (Canada) Ltd.
P.O. Box 931
800 Windmill Road
Dartmouth B3B 1L1
Tel: (902) 469-7820
TWX: 610-271-4482

ONTARIO
Hewlett-Packard (Canada) Ltd.
1020 Morrison Dr.
Ottawa K2H 8K7
Tel: (613) 820-6483
TWX: 610-563-1636

Hewlett-Packard (Canada) Ltd.
6877 Goreway Drive
Mississauga L4V 1M8
Tel: (416) 678-9430
TWX: 610-492-4246

Hewlett-Packard (Canada) Ltd.
552 Newbold Street
London N6E 2S5
Tel: (519) 686-9181
TWX: 610-352-1201

QUEBEC
Hewlett-Packard (Canada) Ltd.
275 Hymus Blvd.
Pointe Claire H9R 1G7
Tel: (514) 697-4232
TWX: 610-422-3022

FOR CANADIAN AREAS NOT LISTED:
Contact Hewlett-Packard (Canada) Ltd. in Mississauga.

CHILE
Jorge Calcagni y Cia. Ltda.
Arturo Burhle 065
Casilla 16475
Correo 9, **Santiago**
Tel: 220222
Telex: JCALCAGNI

COLOMBIA
Instrumentación
Henrik A. Langebaek & Kier S.A.
Carrera 7 No. 48-75
Apartado Aéreo 6287
Bogotá, 1 D.E.
Tel: 269-8877
Telex: 44400

Instrumentación
H.A. Langebaek & Kier S.A.
Carrera 63 No. 49-A-31
Apartado 54098
Medellin
Tel: 304475

COSTA RICA
Científica Costarricense S.A.
Avenida 2, Calle 5
San Pedro de Montes de Oca
Apartado 10159
San José
Tel: 24-38-20, 24-08-19
Telex: 2367 GALGUR CR

CYPRUS
Kypronics
19 Gregorios Xenopoulos Street
P.O. Box 1152
Nicosia
Tel: 45628/29
Telex: 3018

CZECHOSLOVAKIA
Hewlett-Packard
Obchodni zastupitelstvi v CSSR
Pisemny styk
Post. schranka 27
CS 118 01 **Praha** 011
CSSR

Vyvojova a Provozni Zakladna
Vyzkumnych Ustavu v Bechovicich
CSSR-25097 **Bechovice u Prahy**
Tel: 89 93 41
Telex: 12133

Institute of Medical Bionics
Vyskumny Ustav Lekarskej Bioniky
Jedlova 6
CS-88346 **Bratislava-Kramare**
Tel: 44-551
Telex: 93229

DENMARK
Hewlett-Packard A/S
Datavej 52
DK-3460 **Birkerod**
Tel: (02) 81 66 40
Telex: 37409 hpas dk

Hewlett-Packard A/S
Navervej 1
DK-8600 **Silkeborg**
Tel: (06) 82 71 66
Telex: 37409 hpas dk

ECUADOR
CYEDE Cia. Ltda.
P.O. Box 6423 CCI
Av. Eloy Alfaro 1749
Quito
Tel: 450-975, 243-052
Telex: 2548 CYEDE ED

Medical Only
Hospitalar S.A.
Casilla 3590
Robles 625
Quito
Tel: 545-250

EGYPT
I.E.A.
International Engineering Associates
24 Hussein Hegazi Street
Kasr-el-Aini
Cairo
Tel: 23 829
Telex: 93830

SAMITRO
Sami Amin Trading Office
18 Abdel Aziz Gawish
Abdine-Cairo
Tel: 24932

EL SALVADOR
IPESA
Bulevar de los Heroes 11-48
Edificio Sarah 1148
San Salvador
Tel: 252787

ETHIOPIA
Abdella Abdulmalik
P.O. Box 2635
Addis Ababa
Tel: 11 93 40

FINLAND
Hewlett-Packard Oy
Revontulentie, 7
SF-02100 **Espoo** 10
Tel: (90) 455 0211
Telex: 121563 hewpa sf

FRANCE
Hewlett-Packard France
Zone d'activites de Courtaboeuf
Avenue des Tropiques
Boite Postale 6
91401 **Orsay**-Cédex
Tel: (1) 907 78 25
TWX: 600048F

Hewlett-Packard France
Chemin des Mouilles
B.P. 162
69130 **Ecully**
Tel: (78) 33 81 25
TWX: 310617F

Hewlett-Packard France
20, Chemin de La Cépière
31081 **Toulouse**
Le Mirail-Cédex
Tel: (61) 40 11 12

Hewlett-Packard France
Le Ligoures
Place Romée de Villeneuve
13100 **Aix-en-Provence**
Tel: (42) 59 41 02
TWX: 410770F

Hewlett-Packard France
2, Allee de la Bourgonette
35100 **Rennes**
Tel: (99) 51 42 44
TWX: 740912F

Hewlett-Packard France
18, rue du Canal de la Marne
67300 **Schiltigheim**
Tel: (88) 83 08 10
TWX: 890141F

Hewlett-Packard France
Immeuble péricentre
rue van Gogh
59650 **Villeneuve** D'Ascq
Tel: (20) 91 41 25
TWX: 160124F

Hewlett-Packard France
Bâtiment Ampère
Rue de la Commune de Paris
B.P. 300
93153 **Le Blanc Mesnil**-Cédex
Tel: (01) 931 88 50
Telex: 211032F

Hewlett-Packard France
Av. du Pdt. Kennedy
33700 **Merignac**
Tel: (56) 97 01 81

Hewlett-Packard France
Immeuble Lorraine
Boulevard de France
91035 **Evry**-Cédex
Tel: 077 96 60
Telex: 692315F

Hewlett-Packard France
23 Rue Lothaire
57000 **Metz**
Tel: (87) 65 53 50

GERMAN FEDERAL REPUBLIC
Hewlett-Packard GmbH
Vertriebszentrale Frankfurt
Berner Strasse 117
Postfach 560 140
D-6000 **Frankfurt** 56
Tel: (06011) 50041
Telex: 04 13249 hpffm d

Hewlett-Packard GmbH
Technisches Büro Böblingen
Herrenberger Strasse 110
D-7030 **Böblingen,**
Württemberg
Tel: (07031) 667-1
Telex: 07265739 bbn

Hewlett-Packard GmbH
Technisches Büro Düsseldorf
Emanuel-Leutze-Str. 1 (Seestern)
D-4000 **Düsseldorf**
Tel: (0211) 5971-1
Telex: 085/86 533 hpdd d

Hewlett-Packard GmbH
Technisches Büro Hamburg
Kapstadtring 5
D-2000 **Hamburg** 60
Tel: (040) 63804-1
Telex: 21 63 032 hphh d

Hewlett-Packard GmbH
Technisches Büro Hannover
Am Grossmarkt 6
D-3000 **Hannover** 91
Tel: (0511) 46 60 01
Telex: 092 3259

Hewlett-Packard GmbH
Technisches Büro Nürnberg
Neumeyerstrasse 90
D-8500 **Nürnberg**
Tel: (0911) 52 20 83
Telex: 0623 860

Hewlett-Packard GmbH
Technisches Büro München
Eschenstrasse 5
D-8021 **Taufkirchen**
Tel: (089) 6117-1
Telex: 0524985

Hewlett-Packard GmbH
Technisches Büro Berlin
Kaithstrasse 2-4
D-1000 **Berlin** 30
Tel: (030) 24 90 86
Telex: 018 3405 hpbln d

GREECE
Kostas Karayannis
8 Omirou Street
Athens 133
Tel: 32 30 303/32/37 731
Telex: 21 59 62 RKAR GR

GUAM
Guam Medical Supply, Inc.
Suite C, Airport Plaza
P.O. Box 8947
Tamuning 96911
Tel: 646-4513

GUATEMALA
IPESA
Avenida Reforma 3-48
Zona 9
Guatemala City
Tel: 316627, 314786, 66471-5, ext. 9
Telex: 4192 Teletro Gu

HONG KONG
Hewlett-Packard Hong Kong Ltd.
11th Floor, Four Seas Bldg.
212 Nathan Rd.
Kowloon
Tel: 3-697446 (5 lines)
Telex: 36678 HX

Medical/Analytical Only
Schmidt & Co. (Hong Kong) Ltd.
Wing On Centre, 28th Floor
Connaught Road, C.
Hong Kong
Tel: 5-455644
Telex: 74766 SCHMX HX

INDIA
Blue Star Ltd.
Sahas
414/2 Vir Savarkar Marg
Prabhadevi
Bombay 400 025
Tel: 45 78 87
Telex: 011-4093

Blue Star Ltd.
Band Box House
Prabhadevi
Bombay 400 025
Tel: 45 73 01
Telex: 011-3751

Blue Star Ltd.
Bhavdeep
Stadium Road
Ahmedabad 380 014
Tel: 43922
Telex: 012-234

Blue Star Ltd.
7 Hare Street
Calcutta 700 001
Tel: 23-0131
Telex: 021-7655

Blue Star Ltd.
Bhandari House
91 Nehru Place
New Delhi 110 024
Tel: 682547
Telex: 031-2463

Blue Star Ltd.
T.C. 7/603 'Poornima'
Maruthankuzhi
Trivandrum 695 013
Tel: 65799
Telex: 0884-259

Blue Star Ltd.
11 Magarath Road
Bangalore 560 025
Tel: 55668
Telex: 0845-430

Blue Star Ltd.
Meeakshi Mandiram
XXXXV/1379-2 Mahatma Gandhi Rd.
Cochin 682 016
Tel: 32069
Telex: 085-514

Blue Star Ltd.
1-1-117/1 Sarojini Devi Road
Secunderabad 500 033
Tel: 70126
Telex: 0155-459

Blue Star Ltd.
133 Kodambakkam High Road
Madras 600 034
Tel: 82057
Telex: 041-379

ICELAND
Medical Only
Eliding Trading Company Inc.
Hafnarnvoli - Tryggvagötu
P.O. Box 895
IS-**Reykjavik**
Tel: 1 58 20/1 63 03

INDONESIA
BERCA Indonesia P.T.
P.O. Box 496/Jkt.
Jln. Abdul Muis 62
Jakarta
Tel: 349255, 349886
Telex: 46748 BERSIL IA

BERCA Indonesia P.T.
P.O. Box 174/Sby.
23 Jln. Jimerto
Surabaya
Tel: 42027

IRELAND
Hewlett-Packard Ltd.
Kestrel House
Clanwilliam Place
Lower Mount Street
Dublin 2, Eire

Hewlett-Packard Ltd.
2C Avongberg Ind. Est.
Long Mile Road
Dublin 12
Tel: 514322/514224
Telex: 30439

Medical Only
Cardiac Services (Ireland) Ltd.
Kilmore Road
Artane
Dublin 5, Eire
Tel: (01) 315820

Medical Only
Cardiac Services Co.
95A Finaghy Rd. South
Belfast BT10 0BY
GB-Northern Ireland
Tel: (0232) 625566
Telex: 747626

ISRAEL
Electronics Engineering Div. of Motorola Israel Ltd.
16, Kremenetski Street
P.O. Box 25016
Tel-Aviv
Tel: 38973
Telex: 33569, 34164

ITALY
Hewlett-Packard Italiana S.p.A.
Via G. Di Vittorio, 9
20063 **Cernusco Sul Naviglio** (MI)
Tel: (2) 903691
Telex: 334632 HEWPACKIT

Hewlett-Packard Italiana S.p.A.
Via Turazza, 14
35100 **Padova**
Tel: (49) 664888
Telex: 430315 HEWPACKI

Hewlett-Packard Italiana S.p.A.
Via G. Armellini 10
1-00143 **Roma**
Tel: (06) 54 69 61
Telex: 610514

Hewlett-Packard Italiana S.p.A.
Corso Giovanni Lanza 94
I-10133 **Torino**
Tel: (011) 659308
Telex: 221079

Hewlett-Packard Italiana S.p.A.
Via Principe Nicola 43 G/C
I-95126 **Catania**
Tel: (095) 37 05 04
Telex: 970291

Hewlett-Packard Italiana S.p.A.
Via Nuova san Rocco A Capadimonte, 62A
80131 **Napoli**
Tel: (081) 710698

Hewlett-Packard Italiana S.p.A.
Via Martin Luther King, 38/111
I-40132 **Bologna**
Tel: (051) 402394
Telex: 511630

JAPAN
Yokogawa-Hewlett-Packard Ltd.
29-21, Takaido-Higashi 3-chome
Suginami-ku, **Tokyo** 168
Tel: 03-331-6111
Telex: 232-2024 YHP-Tokyo

Yokogawa-Hewlett-Packard Ltd.
Chuo Bldg., 4th Floor
4-20, Nishinakajima 5-chome
Yodogawa-ku, Osaka-shi
Osaka, 532
Tel: 06-304-6021
Telex: 523-3624

Yokogawa-Hewlett-Packard Ltd.
Sunitomo Seimei Nagaya Bldg.
11-2 Shimosasajima-cho,
Nakamura-ku, **Nagoya,** 450
Tel: 052 571-5171

Yokogawa-Hewlett-Packard Ltd.
Tanigawa Building
2-24-1 Tsuruya-cho
Kanagawa-ku
Yokohama, 221
Tel: 045-312-1252
Telex: 382-3204 YHP YOK

Yokogawa-Hewlett-Packard Ltd.
Mito Mitsui Building
105, 1-chome, San-no-maru
Mito, Ibaragi 310
Tel: 0292-25-7470

Yokogawa-Hewlett-Packard Ltd.
Inoue Building
1348-3, Asahi-cho, 1-chome
Atsugi, Kanagawa 243
Tel: 0462-24-0452

Yokogawa-Hewlett-Packard Ltd.
Kumagaya Asahi
Hachijuni Building
4th Floor
3-4, Tsukuba
Kumagaya, Saitama 360
Tel: 0485-24-6563

JORDAN
Mouasher Cousins Co.
P.O. Box 1387
Amman
Tel: 24907/39907
Telex: SABCO JO 1456

KENYA
ADCOM Ltd., Inc.
P.O. Box 30070
Nairobi
Tel: 331955
Telex: 22639

Medical Only
International Aeradio (E.A.) Ltd.
P.O. Box 19012
Nairobi Airport
Nairobi
Tel: 336055/56
Telex: 22201/22301

Medical Only
International Aeradio (E.A.) Ltd.
P.O. Box 95221
Mombasa

KOREA
Samsung Electronics Co., Ltd.
4759 Shingil-6-Dong
Yeong Deung POU
Seoul
Tel: 833-4122, 4121
Telex: SAMSAN 27364

KUWAIT
Al-Khaldiya Trading & Contracting
P.O. Box 830-Safat
Kuwait
Tel: 42 4910/41 1726
Telex: 2481 Areeg kt

LUXEMBURG
Hewlett-Packard Beneluz S.A./N.V.
Avenue du Col-Vert, 1 (Groenkraaglaan)
B-1170 **Brussels**
Tel: (02) 660 5050
Telex: 23 494

MALAYSIA
Hewlett-Packard Sales (Malaysia) Sdn. Bhd.
Suite 2.21/2.22
Bangunan Angkasa Raya
Jalan Ampang
Kuala Lumpur
Tel: 483680, 485653

Protel Engineering
P.O. Box 1917
Lot 259, Satok Road
Kuching, **Sarawak**
Tel: 53544

MEXICO
Hewlett-Packard Mexicana, S.A. de C.V.
Av. Periférico Sur No. 6501
Tepepan, Xochimilco
Mexico 23, D.F.
Tel: 905-676-4600
Telex: 017-74-507

Hewlett-Packard Mexicana, S.A. de C.V.
Rio Volga #600
Col. Del Valle
Monterrey, N.L.
Tel: 78-32-10

MOROCCO
Dolbeau
81 rue Karatchi
Casablanca
Tel: 3041 82
Telex: 23051/22822

Gerep
2, rue d'Agadir
Boite Postal 156
Casablanca
Tel: 272093/5
Telex: 23 739

MOZAMBIQUE
A.N. Goncalves, Ltd.
162, 1° Apt. 14 Av. D. Luis
Caixa Postal 107
Maputo
Tel: 27091, 27114
Telex: 6-203 NEGON Mo

NETHERLANDS
Hewlett-Packard Benelux N.V.
Van Heuven Goedhartlaan 121
P.O. Box 667
1181KK **Amstelveen**
Tel: (20) 47 20 21
Telex: 13 216

NEW ZEALAND
Hewlett-Packard (N.Z.) Ltd.
4-12 Cruickshank Street
Kilbirnie, Wellington 3
P.O. Box 9443
Courtney Place
Wellington
Tel: 877-199

Hewlett-Packard (N.Z.) Ltd.
P.O. Box 26-189
169 Manukau Road
Epsom, **Auckland**
Tel: 687-159

Analytical/Medical Only
Northrop Instruments & Systems Ltd.,
Sturdee House
85-87 Ghuznee Street
P.O. Box 2406
Wellington
Tel: 850-091
Telex: NZ 31291

Northrup Instruments & Systems Ltd.
Eden House, 44 Khyber Pass Rd.
P.O. Box 9682, Newmarket
Auckland 1
Tel: 794-091

Northrup Instruments & Systems Ltd.
Terrace House, 4 Oxford Terrace
P.O. Box 8388
Christchurch
Tel: 64-165

NIGERIA
The Electronics Instrumentations Ltd.
N6B/770 Oyo Road
Oluseun House
P.M.B. 5402
Ibadan
Tel: 461577
Telex: 31231 TEIL NG

The Electronics Instrumentations Ltd.
144 Agege Motor Road, Mushin
P.O. Box 481
Mushin, **Lagos**

NORWAY
Hewlett-Packard Norge A/S
Ostendalen 18
P.O. Box 34
1345 **Osteraas**
Tel: (02) 1711 80
Telex: 16621 hpnas n

Hewlett-Packard Norge A/S
Nygaardsgaten 114
P.O. Box 4210
5013 Nygaardsgaten,
Bergen
Tel: (05) 21 97 33

PANAMA
Electrónico Balboa, S.A.
Aparatado 4929
Panama 5
Calle Samuel Lewis
Edificio "Alfa," No. 2
Ciudad de Panama
Tel: 64-2700
Telex: 3483103 Curundu, Canal Zone

PERU
Compañía Electro Médica S.A.
Los Flamencos 145
San Isidro Casilla 1030
Lima 1
Tel: 41-4325
Telex: Pub. Booth 25424 SISIDRO

PAKISTAN
Mushko & Company Ltd.
Oosman Chambers
Abdullah Haroon Road
Karachi-3
Tel: 511027, 512927
Telex: 2894

SALES OFFICES
Arranged alphabetically by country (cont.)

Mushko & Company, Ltd.
10, Bazar Rd.
Sector G-6/4
Islamabad
Tel: 28264

PHILIPPINES
The Online Advanced Systems Corporation
Rico House
Amorsolo cor. Herrera Str.
Legaspi Village, Makati
P.O. Box 1510
Metro **Manila**
Tel: 85-35-81, 85-34-91, 85-32-21
Telex: 3274 ONLINE

RHODESIA
Field Technical Sales
45 Kelvin Road North
P.O. Box 3458
Salisbury
Tel: 705231 (5 lines)
Telex: RH 4122

POLAND
Biuro Informacji Technicznej Hewlett-Packard
Ul Stawki 2, 6P
PL00-950 **Warszawa**
Tel: 39 59 62, 39 51 87
Telex: 81 24 53

PORTUGAL
Telectra-Empresa Técnica de Equipamentos Eléctricos S.a.r.l.
Rua Rodrigo da Fonseca 103
P.O. Box 2531
P-**Lisbon** 1
Tel: (19) 68 60 72
Telex: 12598

Medical Only
Mundinter
Intercambio Mundial de Comércio S.a.r.l.
P.O. Box 2761
Avenida Antonio Augusto de Aguiar 138
P-**Lisbon**
Tel: (19) 53 21 31/7
Telex: 16691 munter p

PUERTO RICO
Hewlett-Packard Inter-Americas
Puerto Rico Branch Office
Calle 272,
#203 Urb. Country Club
Carolina 00630
Tel: (809) 762-7255
Telex: 345 0514

QATAR
Nasser Trading & Contracting
P.O. Box 1563
Doha
Tel: 22170
Telex: 4439 NASSER

ROMANIA
Hewlett-Packard Reprezentanta
Bd.n. Balcescu 16
Bucuresti
Tel: 15 80 23/13 88 85
Telex: 10440

SAUDI ARABIA
Modern Electronic Establishment (Head Office)
P.O. Box 1228, Baghdadiah Street
Jeddah
Tel: 27 798
Telex: 40035
Cable: ELECTA JEDDAH

Modern Electronic Establishment (Branch)
P.O. Box 2728
Riyadh
Tel: 62596/66232
Telex: 202049

Modern Electronic Establishment (Branch)
P.O. Box 193
Al-Khobar
Tel: 44678-44813
Telex: 670136
Cable: ELECTA AL-KHOBAR

SINGAPORE
Hewlett-Packard Singapore (Pte.) Ltd.
6th Floor, Inchcape House
450-452 Alexandra Road
P.O. Box 58
Alexandra Post Office
Singapore 9115
Tel: 631788
Telex: HPSG RS 21486

SOUTH AFRICA
Hewlett-Packard South Africa (Pty.), Ltd.
Private Bag Wendywood,
Sandton, Transvaal, 2144
Hewlett-Packard Centre
Daphne Street, Wendywood,
Sandton, 2144
Tel: 802-5111/25
Telex: 8-4782

Hewlett-Packard South Africa (Pty.), Ltd.
P.O. Box 120
Howard Place,
Cape Province, 7450
Pine Park Centre, Forest Drive,
Pinelands,
Cape Province, 7405
Tel: 53-7955 thru 9
Telex: 57-0006

SPAIN
Hewlett-Packard Española, S.A.
Calle Jerez 3
E-**Madrid** 16
Tel: (1) 458 26 00 (10 lines)
Telex: 23515 hpe

Hewlett-Packard Española S.A.
Colonia Mirasierra
Edificio Juban
c/o Costa Brava, 13
Madrid 34

Hewlett-Packard Española, S.A.
Milanesado 21-23
E-**Barcelona** 17
Tel: (3) 203 6200 (5 lines)
Telex: 52603 hpbe e

Hewlett-Packard Española, S.A.
Av Ramón y Cajal, 1
Edificio Sevilla, planta 9°
E-**Sevilla** 5
Tel: 64 44 54/58

Hewlett-Packard Española S.A.
Edificio Albia II 7° B
E-**Bilbao** 1
Tel: 23 83 06/23 82 06

Hewlett-Packard Española S.A.
C/Ramon Gordillo 1
(Entlo.)
E-**Valencia** 10
Tel: 96-361.13.54/361.13.58

SRI LANKA
Metropolitan Agencies Ltd.
209/9 Union Place
Colombo 2
Tel: 35947
Telex: 1377METROLTD CE

SUDAN
Radison Trade
P.O. Box 921
Khartoum
Tel: 44048
Telex: 375

SURINAM
Surtel Radio Holland N.V.
Grote Hofstr. 3-5
P.O. Box 155
Paramaribo
Tel: 72118, 77880

SWEDEN
Hewlett-Packard Sverige AB
Enighetsvägen 3, Fack
S-161 **Bromma** 20
Tel: (08) 730 05 50
Telex: 10721
Cable: MEASUREMENTS Stockholm

Hewlett-Packard Sverige AB
Frötallsgatan 30
S-421 32 **Västra Frölunda**
Tel: (031) 49 09 50
Telex: 10721 via Bromma office

SWITZERLAND
Hewlett-Packard (Schweiz) AG
Zürcherstrasse 20
P.O. Box 307
CH-8952 **Schlieren-Zürich**
Tel: (01) 7305240
Telex: 53933 hpag ch
Cable: HPAG CH

Hewlett-Packard (Schweiz) AG
Château Bloc 19
CH-1219 **Le Lignon-Geneva**
Tel: (022) 96 03 22
Telex: 27333 hpag ch
Cable: HEWPACKAG Geneva

SYRIA
General Electronic Inc.
Nuri Basha-Ahnaf Ebn Kays Street
P.O. Box 5781
Damascus
Tel: 33 24 87
Telex: 11215 ITIKAL
Cable: ELECTROBOR DAMASCUS

Medical only
Sawah & Co.
Place Azmé
B.P. 2308
Damascus
Tel: 16 367-19 697-14 268
Telex: 11304 SATACO SY
Cable: SAWAH, DAMASCUS

Suleiman Hilal El Mlawi
P.O. Box 2528
Mamoun Bitar Street, 56-58
Damascus
Tel: 11 46 63
Telex: 11270
Cable: HILAL DAMASCUS

TAIWAN
Hewlett-Packard Far East Ltd.
Taiwan Branch
Bank Tower, 5th Floor
205 Tun Hau North Road
Taipei
Tel: (02) 751-0404 (15 lines)

Hewlett-Packard Far East Ltd.
Taiwan Branch
68-2, Chung Cheng 3rd. Road
Kaohsiung
Tel: (07) 242318-Kaohsiung

Analytical Only
San Kwang Instruments Co., Ltd.
20 Yung Sui Road
Taipei
Tel: 3615446-9 (4 lines)
Telex: 22894 SANKWANG

TANZANIA
Medical Only
International Aeradio (E.A.), Ltd.
P.O. Box 861
Dar es Salaam
Tel: 21251 Ext. 265
Telex: 41030

THAILAND
UNIMESA Co. Ltd.
Elcom Research Building
2538 Sukumvit Ave.
Bangchak, Bangkok
Tel: 39-32-387, 39-30-338

TRINIDAD & TOBAGO
CARTEL
Caribbean Telecoms Ltd.
P.O. Box 732
69 Frederick Street
Port-of-Spain
Tel: 62-53068

TUNISIA
Tunisie Electronique
31 Avenue de la Liberte
Tunis
Tel: 280 144

Corema
1 ter. Av. de Carthage
Tunis
Tel: 253 821
Telex: 12319 CABAM TN

TURKEY
TEKNIM Company Ltd.
Riza Sah Pehlevi
Caddesi No. 7
Kavaklidere, **Ankara**
Tel: 275800
Telex: 42155

Teknim Com., Ltd.
Barbaros Bulvari 55/12
Besikyas, **Istanbul**
Tel: 613 546
Telex: 23540

E.M.A.
Muhendislik Kollektif Sirketi
Mediha Eldem Sokak 41/6
Yuksel Caddesi
Ankara
Tel: 17 56 22

Telex: 43933 Yilmaz Ozyurek
Milli Mudafaa Cad 16/6
Kizilay
Ankara
Tel: 25 03 09 - 17 80 26
Telex: 42576 OZEK TR

UNITED ARAB EMIRATES
Emitac Ltd. (Head Office)
P.O. Box 1641
Sharjah
Tel: 354121/3
Telex: 8136

Emitac Ltd. (Branch Office)
P.O. Box 2711
Abu Dhabi
Tel: 331370/1

UNITED KINGDOM
Hewlett-Packard Ltd.
King Street Lane
Winnerah, Wokingham
Berkshire RG11 5AR
GB-England
Tel: (0734) 784774
Telex: 84 71 78/9

Hewlett-Packard Ltd.
Fourier House,
257-263 High Street
London Colney
St. Albans, Herts
GB-England
Tel: (0727) 24400
Telex: 1-8952716

Hewlett-Packard Ltd.
Trafalgar House
Navigation Road
Altrincham
Cheshire WA14 1NU
GB-England
Tel: (061) 928 6422
Telex: 668068

Hewlett-Packard Ltd.
Lygon Court
Hereward Rise
Dudley Road
Halesowen,
West Midlands, B62 8SD
GB-England
Tel: (021) 501 1221
Telex: 339105

Hewlett-Packard Ltd.
Wedge House
799, London Road
Thornton Heath
Surrey, CR4 6XL
GB-England
Tel: (01) 684-0103/8
Telex: 946825

Hewlett-Packard Ltd.
14 Wesley St
Castleford
Yorks WF10 1AE
Tel: (0977) 550016
TWX: 5557335

Hewlett-Packard Ltd.
Tradax House
St. Mary's Walk
Maidenhead
Berkshire, SL6 1ST
GB-England

Hewlett-Packard Ltd.
Morley Road
Staplehill
Bristol, BS16 4QT
GB-England

Hewlett-Packard Ltd.
South Queensferry
West Lothian, EH30 9TG
GB-Scotland
Tel: (031) 331 1188
Telex: 72682

UNITED STATES

ALABAMA
700 Century Park South, Suite 128
Birmingham 35226
Tel: (205) 822-6802

P.O. Box 4207
8290 Whitesburg Dr.
Huntsville 35802
Tel: (205) 881-4591

ARIZONA
2336 E. Magnolia St.
Phoenix 85034
Tel: (602) 273-8000

2424 East Aragon Rd.
Tucson 85706
Tel: (602) 273-8000

***ARKANSAS**
Medical Service Only
P.O. Box 5646
Brady Station
Little Rock 72215
Tel: (501) 376-1844

CALIFORNIA
1579 W. Shaw Ave.
Fresno 93771
Tel: (209) 224-0582

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

5400 West Rosecrans Blvd.
P.O. Box 92105
World Way Postal Center
Los Angeles 90009
Tel: (213) 970-7500
TWX: 910-325-6608

3939 Lankershim Boulevard
North Hollywood 91604
Tel: (213) 877-1282
TWX: 910-499-2671

3200 Hillview Av
Palo Alto, CA 94304
Tel: (408) 988-7000

646 W. North Market Blvd.
Sacramento 95834
Tel: (916) 929-7222

9606 Aero Drive
P.O. Box 23333
San Diego 92123
Tel: (714) 279-3200

363 Brookhollow Dr.
Santa Ana, CA 92705
Tel: (714) 641-0977

3003 Scott Boulevard
Santa Clara 95050
Tel: (408) 988-7000
TWX: 910-338-0518

454 Carlton Court
So. San Francisco 94080
Tel: (415) 877-0772

***Tarzana**
Tel: (213) 705-3344

COLORADO
5600 DTC Parkway
Englewood 80110
Tel: (303) 771-3455

CONNECTICUT
47 Barnes Industrial Road
Barnes Park South
Wallingford 06492
Tel: (203) 265-7801

FLORIDA
P.O. Box 24210
2727 N.W. 62nd Street
Ft. Lauderdale 33309
Tel: (305) 973-2600

4080 Woodcock Drive #132
Brownett Building
Jacksonville 32207
Tel: (904) 398-0663

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

P.O. Box 12826
Suite 5, Bldg. 1
Office Park North
Pensacola 32575
Tel: (904) 476-8422
110 South Hoover Blvd.
Suite 120
Tampa 33609
Tel: (813) 872-0900

GEORGIA
P.O. Box 105005
450 Interstate North Parkway
Atlanta 30348
Tel: (404) 955-1500
TWX: 810-766-4890

Medical Service Only
***Augusta** 30903
Tel: (404) 736-0592

P.O. Box 2103
1172 N. Davis Drive
Warner Robins 31098
Tel: (912) 922-0449

HAWAII
2875 So. King Street
Honolulu 96826
Tel: (808) 955-4455

ILLINOIS
211 Prospect Rd.
Bloomington 61701
Tel: (309) 663-0383

5201 Tolliview Dr.
Rolling Meadows 60008
Tel: (312) 255-9800
TWX: 910-687-2260

INDIANA
7301 North Shadeland Ave.
Indianapolis 46250
Tel: (317) 842-1000
TWX: 810-260-1797

IOWA
2415 Heinz Road
Iowa City 52240
Tel: (319) 351-1020

KENTUCKY
10170 Linn Station Road
Suite 525
Louisville 40223
Tel: (502) 426-0100

LOUISIANA
P.O. Box 1449
3229-39 Williams Boulevard
Kenner 70062
Tel: (504) 443-6201

MARYLAND
7121 Standard Drive
Parkway Industrial Center
Hanover 21076
Tel: (301) 796-7700
TWX: 710-862-1943

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 Hills 48024
Tel: (313) 476-6400

724 West Centre Ave.
Kalamazoo 49002
Tel: (616) 323-8362

MINNESOTA
2400 N. Prior Ave.
St. Paul 55113
Tel: (612) 636-0700

MISSISSIPPI
322 N. Mart Plaza
Jackson 39206
Tel: (601) 982-9363

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

1024 Executive Parkway
St. Louis 63141
Tel: (314) 878-0200

NEBRASKA
Medical Only
7101 Mercy Road
Suite 101
Omaha 68106
Tel: (402) 392-0948

NEVADA
***Las Vegas**
Tel: (702) 736-6610

NEW JERSEY
Crystal Brook Professional Building
Route 35
Eatontown 07724
Tel: (201) 542-1384

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

NEW MEXICO
P.O. Box 11634
Station E
11300 Lomas Blvd., N.E.
Albuquerque 87123
Tel: (505) 292-1330
TWX: 910-989-1185

156 Wyatt Drive
Las Cruces 88001
Tel: (505) 526-2484
TWX: 910-9983-0550

NEW YORK
6 Automation Lane
Computer Park
Albany 12205
Tel: (518) 458-1550
TWX: 710-444-4961

650 Perinton Hill Office Park
Fairport 14450
Tel: (716) 223-9950
TWX: 510-253-0092

No. 1 Pennsylvania Plaza
55th Floor
34th Street & 8th Avenue
New York 10001
Tel: (212) 971-0800

5858 East Molloy Road
Syracuse 13211
Tel: (315) 455-2486

1 Crossways Park West
Woodbury 11797
Tel: (516) 921-0300
TWX: 510-221-2183
Tel: (513) 671-7400

NORTH CAROLINA
5605 Roanne Way
Greensboro 27409
Tel: (919) 852-1800

OHIO
Medical/Computer Only
9920 Carver Road
Cincinnati 45242
Tel: (513) 891-9870
16500 Sprague Road
Cleveland 44130
Tel: (216) 243-7300
TWX: 810-423-9430

962 Crupper Ave.
Columbus 43229
Tel: (614) 436-1041

330 Progress Rd.
Dayton 45449
Tel: (513) 859-8202

OKLAHOMA
P.O. Box 32008
6301 N. Meridan Avenue
Oklahoma City 73112
Tel: (405) 721-0200

9920 E. 42nd Street
Suite 121
Tulsa 74145
Tel: (918) 665-3300

OREGON
17890 S.W. Lower Boones Ferry Road
Tualatin 97062
Tel: (503) 620-3350

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

111 Zeta Drive
Pittsburgh 15238
Tel: (412) 782-0400

SOUTH CAROLINA
P.O. Box 6442
6941-0 N. Trenholm Road
Columbia 29206
Tel: (803) 782-6493

TENNESSEE
8906 Kingston Pike
Knoxville 37919
Tel: (615) 691-2371

3070 Directors Row
Directors Square
Memphis 38131
Tel: (901) 346-8370

***Nashville**
Medical Service Only
Tel: (615) 244-5448

TEXAS
4171 North Mesa
Suite C110
El Paso 79902
Tel: (915) 533-3555

P.O. Box 42816
10535 Harwin St.
Houston 77036
Tel: (713) 776-6400

***Lubbock**
Medical Service Only
Tel: (806) 799-4472

P.O. Box 1270
201 E. Arapaho Rd.
Richardson 75081
Tel: (214) 231-6101

205 Billy Mitchell Road
San Antonio 78226
Tel: (512) 434-8241

UTAH
2160 South 3270 West Street
Salt Lake City 84119
Tel: (801) 972-4711

VIRGINIA
P.O. Box 9669
2914 Hungary Spring Road
Richmond 23228
Tel: (804) 285-3431

Computer Systems/Medical Only
Airport Executive Center
Suite 302
5700 Thurston Avenue
Virginia Beach 23455
Tel: (804) 460-2471

WASHINGTON
Bellefield Office Pk.
1203 - 114th Ave. S.E.
Bellevue 98004
Tel: (206) 454-3971
TWX: 910-443-2446

P.O. Box 4010
Spokane 99202
Tel: (509) 535-0864

***WEST VIRGINIA**
Medical/Analytical Only
4604 Mac Corkle Ave., S.E.
Charleston 25304
Tel: (304) 925-0492

WISCONSIN
150 South Sunny Slope Road
Brookfield 53005
Tel: (414) 784-8800

FOR U.S. AREAS NOT LISTED:
Contact the regional office nearest you:
Atlanta, Georgia . . . **North Holly-wood,** California . . . **Rockville,** Maryland . . . **Rolling Meadows,** Illinois.
Their complete addresses are listed above.

USSR
Hewlett-Packard Representative Office
USSR
Pokrovsky Boulevard 4/17-kw 12
Moscow 101000
Tel: 294.20.24
Telex: 7825 hewpak su

YUGOSLAVIA
Iskra Commerce, n.sol.o.
Zastopstvo Hewlett-Packard
Obilicev Venac 26
YU 11000 **Beograd**
Tel: 636-955
Telex: 11530

Iskra Commerce, n.sol.o.
Zastopstvo Hewlett-Packard
Miklosiceva 38/VII
YU-61000 **Ljubljana**
Tel: 321-674, 315-879
Telex: 31583

URUGUAY
Pablo Ferrando S.A.C.el.
Avenida Italia 2877
Casilla de Correo 370
Montevideo
Tel: 40-3102
Telex: 702 Public Booth
Para Pablo Ferrando

VENEZUELA
Hewlett-Packard de Venezuela C.A.
P.O. Box 50933
Caracas 105
Los Ruices Norte
3a Transversal
Edificio Segre
Caracas 107
Tel: 239-4133 (20 lines)
Telex: 25146 HEWPACK

ZAMBIA
R.J. Tilbury (Zambia) Ltd.
P.O. Box 2792
Lusaka
Tel: 73793

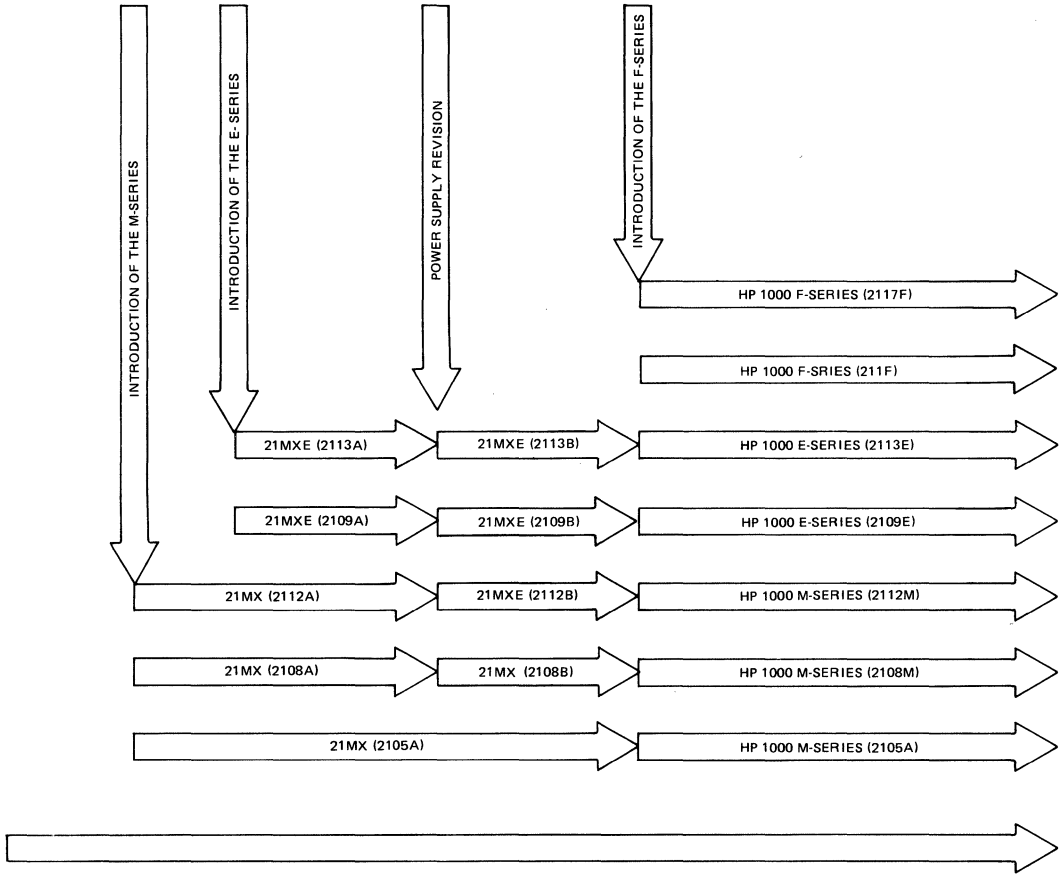
MEDITERRANEAN AND MIDDLE EAST COUNTRIES NOT SHOWN, PLEASE CONTACT:
Hewlett-Packard S.A.
Mediterranean and Middle East Operations
35, Kolokotroni Street
Platia Kefallariou
GR-Kifissia-**Athens,** Greece
Tel: 8080359/429
Telex: 21-6588
Cable: HEWPACKSA Athens

SOCIALIST COUNTRIES NOT SHOWN, PLEASE CONTACT:
Hewlett-Packard Ges.m.b.H.
Handelskai 52
P.O. Box 7
A-1205 **Vienna,** Austria
Tel: (0222) 35 16 21 to 27
Cable: HEWPAK Vienna
Telex: 75923 hewpak a

OTHER AREAS NOT LISTED, CONTACT:
Hewlett-Packard Intercontinental
3495 Deer Creek Road
Palo Alto, California 94304
Tel: (415) 856-1501
TWX: 910-373-1267
Cable: HEWPACK Palo Alto
Telex: 034-8300, 034-8493

Hewlett-Packard S.A.
7, rue du Bois-du-Lan
P.O. Box
CH-1217 Meyrin 2 - **Geneva**
Switzerland
Tel: (022) 82 70 00
Cable: HEWPACKSA Geneva
Telex: 2 24 86

***Service Only**
2-15-80



HP 1000 SERIES NAMING HISTORY

7700-504

NOTICE

The information contained in this document is subject to change without notice.

HEWLETT-PACKARD MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

MANUAL PART NO. 5955-0282
Printed in U.S.A. MARCH 1980

HEWLETT
PACKARD