88973500

CONTROL DATA CORPORATION

CONTROL DATA® MICRO-PROGRAMMABLE COMPUTER FAMILY

1700 ENHANCED PROCESSOR WITH CORE MEMORY

HARDWARE REFERENCE MANUAL

REVISION RECORD

REVISION	DESCRIPTION
01 (10/74)	Preliminary Release
A (8/75)	Manual released. This manual obsoletes all previous editions.
B (9/77)	Manual revised; includes Engineering Change Order 14967 which corrects storage reference instructions. Page 4-16 is revised.

Publication No. 88973500

REVISION LETTERS I, O, Q AND X ARE NOT USED

Printed in the United States of America

Address comments concerning this manual to:
Control Data Corporation
Publications and Graphics Division
4455 Eastgate Mall
LaJolla, Calif. 92037
or use Comment Sheet in the back of this manual.

LIST OF EFFECTIVE PAGES

New features, as well as changes, deletions, and additions to information in this manual, are indicated by bars in the margins or by a dot near the page number if the entire page is affected. A bar by the page number indicates pagination rather than content has changed.

PAGE	REV	PAGE	REV	PAGE	REV	PAGE	REV	PAGE	REV
Cover	--								
Title Page	--								
Revision Record	B								
iii/iv	B								
v thru viii	A								
1-1 thru 1-8	A								
2-1 thru 2-6	A								
3-1 thru 3-7	A								
4-1 thru 4-15	A								
4-16	B								
4-17 thru 4-29	A								
5-1 thru 5-3	A								
6-1	A								
6-2	A								
7-1 thru 7-4	A								
A-1 thru A-3	A								
B-1 thru B-4	A								
C-1 thru C-8	A								
Index-1 thru Index-6	A								
Comment Sheet	B								
Envelope	--								
Back Cover	--								

PREFACE

The micro-programmable (MP) computer emulates the 1700 family of computers. Readers of this document should be familiar with the CONTROL DATA® 1700 series computers and their associated hardware. The MP is upward-compatible and has an enhanced instruction capability.

Additional information on Control Data software applicable to the MP system will be found in the following publications:

Description	Publication No.
1700 Computer System Codes	60163500
1700 MSOS Version 4 Reference Manual	60361500
1700 MSOS 4 MS FORTRAN Version 3A/B	60362000
Micro Processor Reference Manual	88973400
CCP Support Software 1 MICRO Assembler Reference Manual	88988800
CCP Support Software 1 MACRO Assembler Reference Manual	88988900

CONTENTS

APPENDIXES

INDEX

FIGURES

TABLES

SYSTEM DESCRIPTION 1

INTRODUCTION

The 1700 enhanced processor computer is a special configuration of the CONTROL DATA® MP family of parallel mode, stored program, digital processors. It is dedicated to perform as a CDC® 1700-compatible digital computer. The MP uses micro programming to perform 1700 language programs.

This manual describes the basic as well as the optional characteristics of the MP. It covers the hardware, general operating procedures, and the MP instruction repertoire.

The basic MP configuration consists of:

- Micro processor with 1700 transform
- Micro memory
- Macro memory
- I/O interface
- Power supply

Various standard options, such as a card reader and a line printer, are available for the MP. The user may also use the micro memory and I/O to perform nonstandard 1700 functions to achieve even greater flexibility with the MP processor.

A listing of the general characteristics of the MP is contained in table 1-1.

FUNCTIONAL CHARACTERISTICS

The micro-programmable computer is a multilevel processor, which uses a semiconductor ROM and a special hardware function (transform) to emulate a CDC 1700 computer. The macro memory unit contains 1700 language programs (called macro instructions). The multilevel processor differs from the conventional processor, as shown in figure 1-1. The MP operation is controlled by a program (micro program) in the semiconductor memory (referred to as micro memory). The micro program reads 1700 macro instructions from macro memory and decodes them for execution in the micro processor. The semiconductor memory is several times faster than the macro memory. The transform aids in decoding and program execution. Therefore, the MP uses special micro-programming techniques to emulate an enhanced CDC 1700 system for lower cost, smaller size, and equal or better speed.

PHYSICAL CHARACTERISTICS

The MP is modularly designed with standard TTL MSI components and commercial construction.

The standard chassis, shown in figure 1-2, is 18.5 inches high by 17.5 inches wide by 12 inches deep. The chassis includes cooling fans. The standard chassis back panel has the I/O wiring for the 1700 A/Q and 1700 A/Q-DMA. However, it may also contain specialized I/O for the user. Wiring details are included in the system wirelist provided for the unit. A front cover panel is provided on the chassis for maximum cooling.

Power requirements for the MP vary with the user's application. CDC provides power supplies of ±5, ±12, and ±15 volts with input power requirements of 115 vac, 50 or 60 Hz. Physical dimensions for a power supply chassis are 8.75 inches high by 17.5 inches wide by 16.0 inches deep. Cooling fans for logic and power supply chassis require 115 vac, 50 or 60 Hz.

A typical MP circuit card, shown in figure 1-3, is 11 by 14 inches and has 204 input/output contacts.

The MP chassis has a pre-wired location for an optional panel interface card. The maintenance panel is a 16-inch by 4.5-inch printed circuit board, connected by a flexible cable to the panel interface card. The panel contains controls and LED indicators for manually controlling the MP at the micro level. The panel interface card also provides an interface to ASCII RS232-compatible consoles (full-duplex interface). The normal configuration for the 1700 enhanced processor uses the I/O-TTY module for manual operator interface.

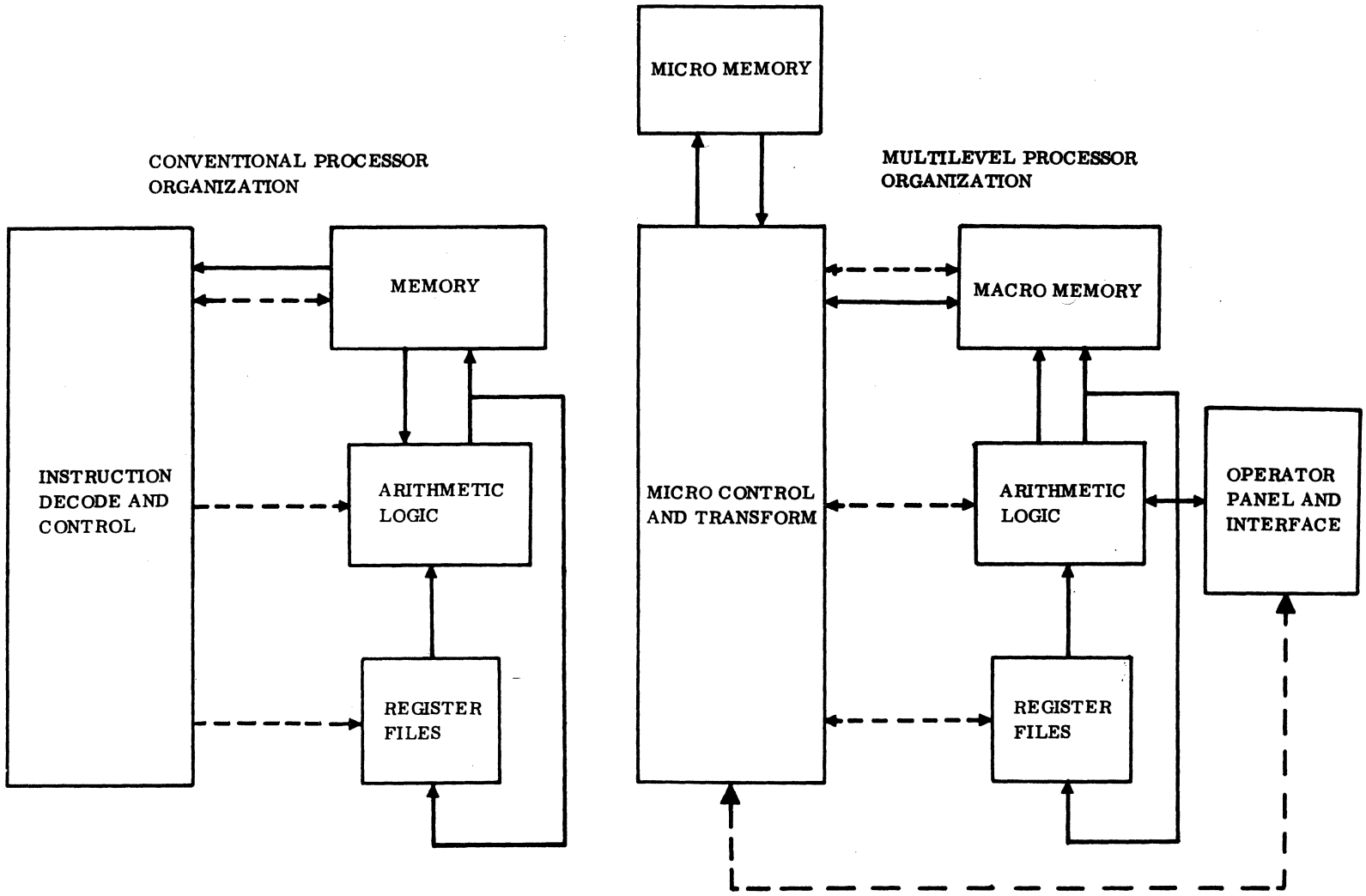
TABLE 1-1. MP GENERAL CHARACTERISTICS

Basic Configuration	
Processor	
Type	General-purpose, micro-programmable digital processor
Organization	Register oriented or file oriented.
Word length	16 bit
Micro-instruction word	32-bit format; two micro instructions per micro-memory address
Micro-memory type	Semiconductor read/write memory (RAM) and/or read only memory (ROM)
Micro-memory size	512 words in 64-bit increments (on transform); maximum of 4,096 additional words available.
Micro-memory access time	70 nanoseconds
Arithmetic	Binary with dynamic selection of ones or twos complement mode Up to four parallel unrelated operations are possible in one micro instruction
Macro-instruction execution time	Approximately the same as a 1700 computer with 900 μsec memory cycle time (for detailed timing, see Appendix C).
Macro Memory	
Requirement	Variable, according to application
Type	Core memory: available in 8K stacks, with a maximum of 32K; the main chassis has a 16-bit format. Parity and protect bits are available in the standard stack.
Core speed	Read: 600 nanoseconds cycle time† Write: 700 nanoseconds cycle time†
Direct memory access	Four I/O ports are wired for DMA devices; one can be a CDC 1700 DSA (QSE feature).
Input/Output (I/O)	
Interfaces	Teletypewriter Display terminal (RS232-C compatible)

†The shortest possible time between successive operations.

TABLE 1-1. MP GENERAL CHARACTERISTICS (Continued)

Basic Configuration (Continued)	
Mechanical	
Hardware	Modular
Construction	RETMA 19-inch, rack mountable
Dimensions	Logic Chassis: Height — 18.5 inches (47 cm) Width — 17.5 inches (44.5 cm) Depth — 16.0 inches (40.64 cm) Power Supply Chassis: Height — 8.75 inches (22.25 cm) Width — 17.5 inches (44.5 cm) Depth — 16.0 inches (40.64 cm)
Weight	Logic Chassis: 40 pounds (approximately) (18 kg) Power Supply: 50 pounds (approximately) (45 kg)
Input power	115 volts, 50/60 Hz
Miscellaneous Features	Real-time clock Auto-data transfer Enhanced 1700 instruction repertoire
Standard Options	
Input/Output (I/O)	
Interfaces	Maintenance panel CDC 1700 A/Q-DMA (TTL level) RS232-C compatible console
Operator input device	Teletypewriter ASR/KSR 33/35 CDC conversational display terminals (RS232-C compatible)



NOTES: 1. DOTTED LINES ARE CONTROL SIGNALS.
2. SOLID LINES ARE INSTRUCTIONS AND DATA FLOW.

Figure 1-1. Digital Processor Organizations

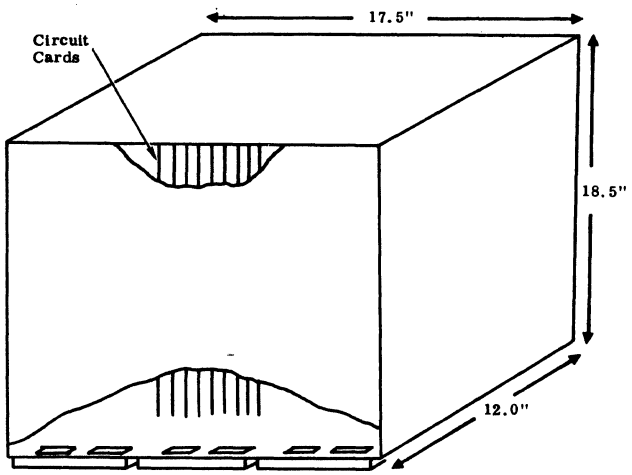


Figure 1-2. MP Standard Chassis

The MP will operate in computer rooms, general offices, and industrial environments. It will operate at temperatures of 40° F to 120° F (4.5° C to 48.8° C), withstand a maximum temperature gradient of 0.2° F per minute or at a rate that precludes condensation, and a relative humidity of 10 to 90 percent. Non-operating environment extends the temperature range from -30° F to 150° F (-35° C to 65° C) and a maximum thermal gradient not to exceed 20° F per hour or at a rate that precludes condensation. Storage temperatures with proper packaging protection may range from -60° F to 160° F and relative humidity from 2 to 98 percent with temperature cycles of not more than 60° F per hour or at a rate that will preclude condensation. The user should note that these ranges cover only the micro processor; peripheral equipments may require more stringent environmental controls.

MAJOR SYSTEM COMPONENT DESCRIPTION

Figure 1-4 shows the chassis layout for the MP equipment; figure 1-5 is the functional block diagram.

MICRO PROCESSOR

The MP enhanced processor consists of an arithmetic card (ALU), a status mode interrupt card (SMI), control cards 1 and 2, and a 1700 transform module. The micro-processor cards are interconnected through the basic backpanel wiring. Special user options will require additional wiring.

TRANSFORM

The transform hardware is packaged as a separate module and is specially designed for the MP application. The MP has a 512-word, 64-bit ROM micro memory on the transform module.

Functioning as the hardware portion of the macro-instruction decode process, the transform causes the micro program to form program branches, sets various parameters, and performs arithmetic or logical operations. It provides the micro program with the capability of selecting patterns of bits from the data transmission paths to form the micro-memory addresses that sequence the micro program.

MICRO MEMORY

The MP contains a 512-word micro memory on the 1700 transform board. It also has two card slots for additional micro-memory or special algorithms if required by the user. The slots are interconnected to the micro processor through the backpanel and are accessible only by the micro processor.

MACRO MEMORY (CORE) AND MEMORY INTERFACE

The core macro memory consists of memory stacks and an interface card. The memory stacks are configured in 8K increments of 20 bits: 1 parity, 1 protect, 1 protect parity, 1 unused, and 16 data bits. The stacks are mounted on standard 11 x 14-inch circuit boards, with each stack requiring two card spaces in the chassis.

Data flow is in 16-bit word format, with a maximum of 32K possible in the basic chassis. A direct memory access (DMA) channel is included in the memory interface as well as the parity and program protect generation and checking. The DMA for the MP can provide access for four external DMA devices through a port to the macro memory.

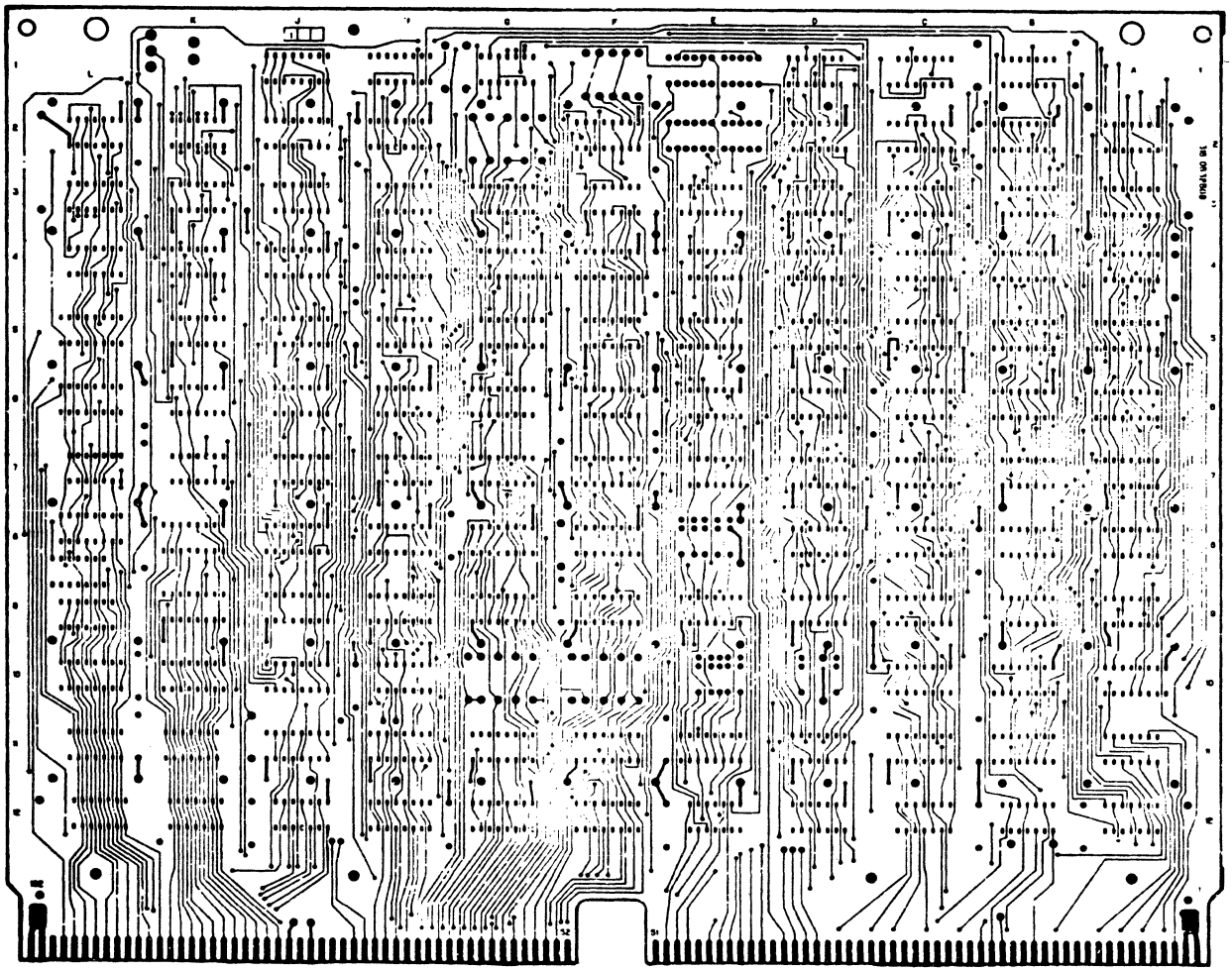


Figure 1-3. Typical MP Circuit Card

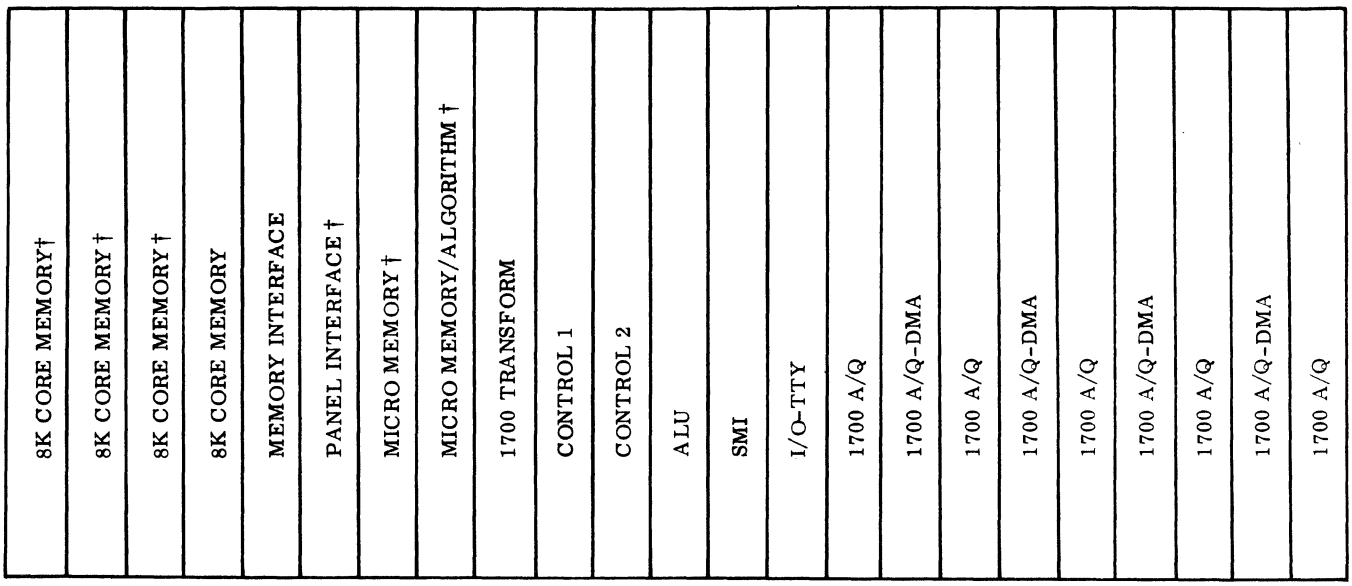


Figure 1-4. Typical MP Chassis Layout

I/O-TTY INTERFACE

The standard operator interface to the MP is through the I/O-TTY module. It can interface with Teletype Corporation Model ASR/KSR 33/35 Teletype or the Control Data RS232-C compatible conversational display terminals. A TTL bus is available in the I/O-TTY module for interfacing the controller cards in the main chassis to the micro processor.

EXTERNAL I/O INTERFACE

The main chassis for the MP includes nine slots for external I/O devices (in addition to the I/O capability of the I/O-TTY module). As shown in figure 1-4, four slots are prewired for 1700 A/Q-DMA Channels and five slots are prewired for 1700 A/Q channels. These may be used with standard CDC equipment or for special user applications.

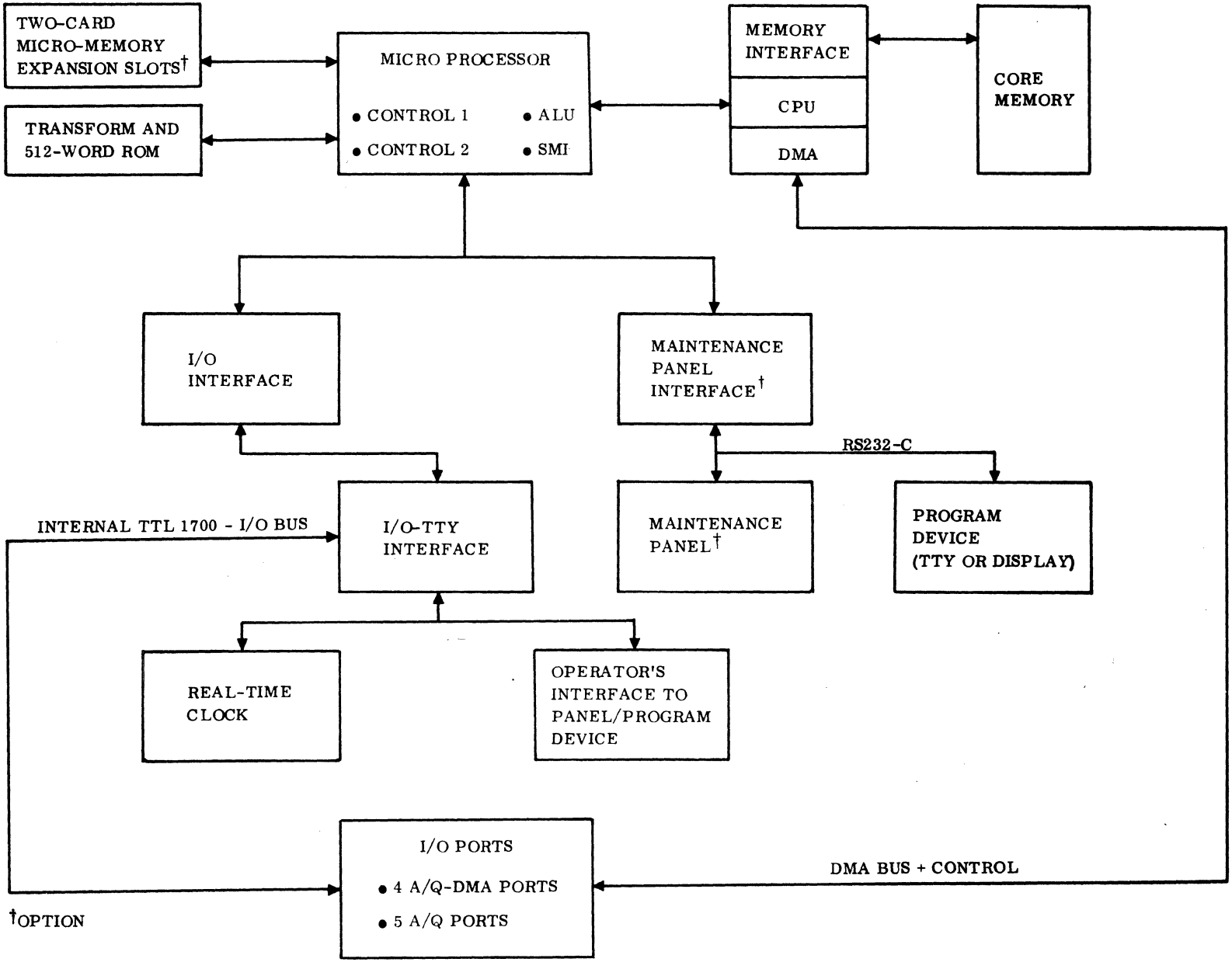


Figure 1-5. MP Functional Block Diagram

GENERAL DESCRIPTION

The micro-programmable computer will emulate a CDC 1700 computer system. It can perform all 1700 functions, utilizing an expanded instruction set with interfacing capabilities to 1700 Series peripherals. Figure 2-1 shows a block diagram of the MP system. The basic MP configuration includes the micro processor, macro memory, I/O interface, and the operator's interface. The flexible design of the system permits the user to incorporate his own equipment or to upgrade the MP with additional micro memory, I/O capability, or a special hardware algorithm module.

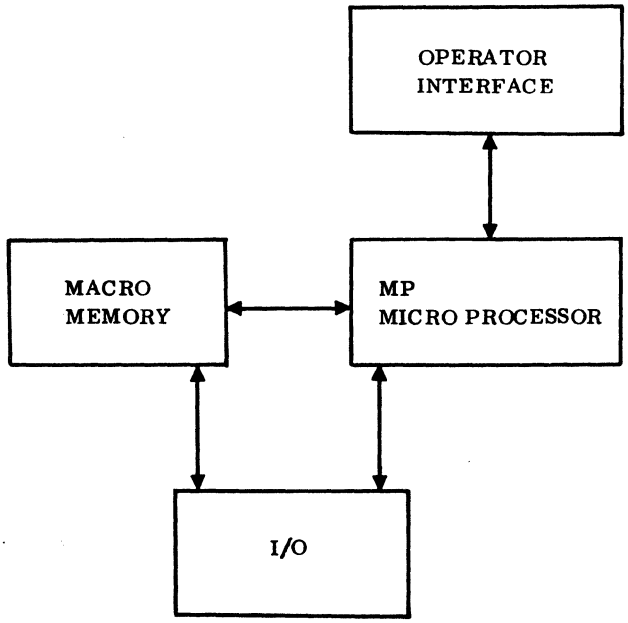


Figure 2-1. MP Block Diagram

MICRO PROCESSOR

The MP central processing unit (CPU) is a special configuration that consists of an ALU module, an SMI module, two control modules, and the standard MP transform module. Detailed organization of the MP is shown in figure 2-2. This diagram shows MP registers interconnected primarily by selectors. A selector is a multiplexer that transfers one of several inputs to an output. They are either one, eight, 12, 16, or 32 bits wide.

TRANSFORMS AND THE TRANSFORM MODULE

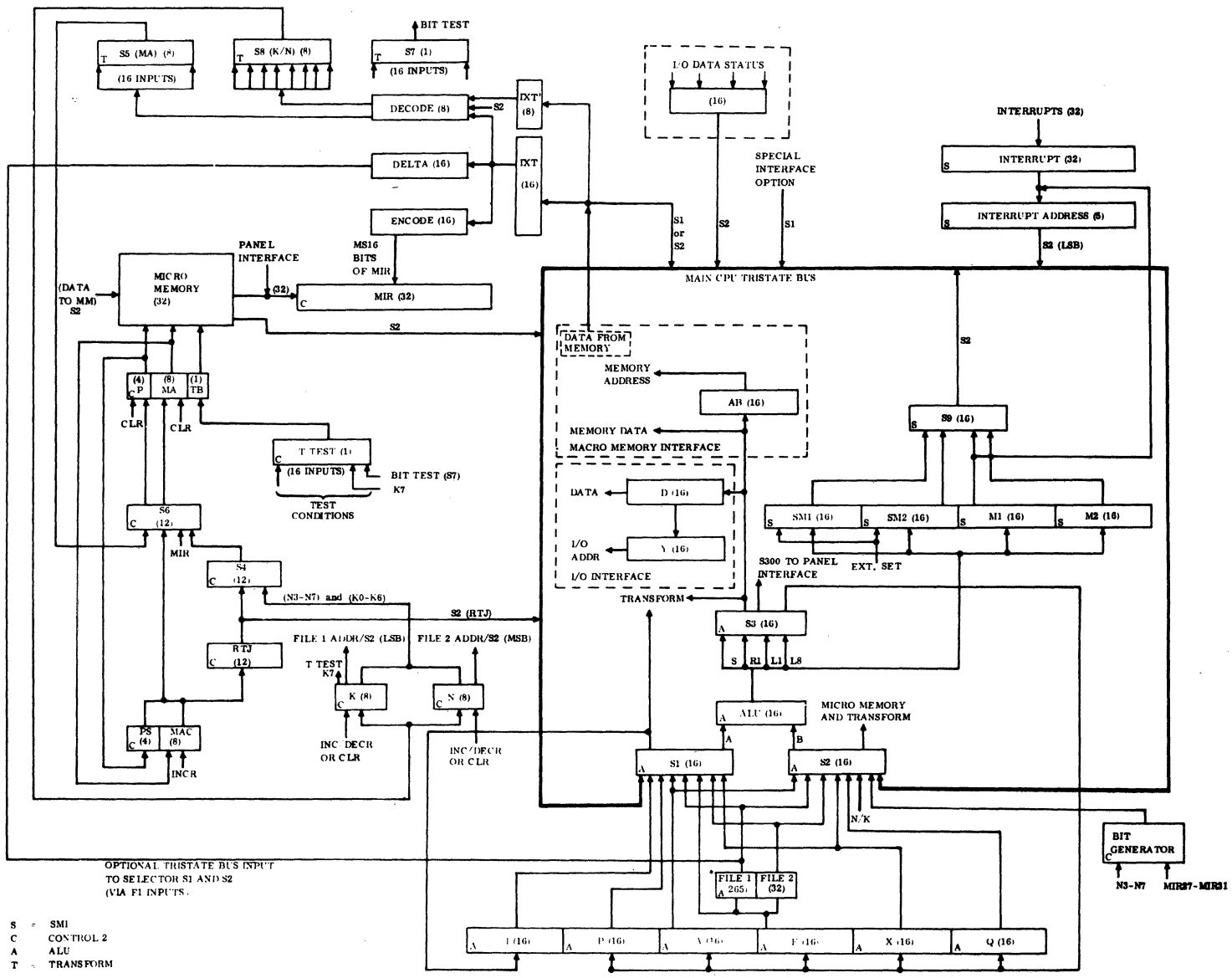
Transforms enable quick and efficient decoding of an emulated instruction. A transform can be designed to extract bits from a register or registers, shift the bits to the required position, and add a base address or constant bits. This result can then be transferred to the micro-memory address register (transform jump) or to the K or N register (transform register load). For example, when a 1700 instruction is read from macro memory, one micro-instruction transform jump transfers control to one of 108 micro-memory locations. Without the transform feature the above operation would require many micro instructions.

The transform hardware is packaged in a separate module and is implemented using three selectors. The transform module includes 1,024 micro instructions (512 words) in ROM. The majority of these instructions are used to execute the 1700 emulator. The ROM also contains instructions for the panel interface simulation via the I/O-TTY board.

ARITHMETIC/LOGICAL UNIT (ALU) AND DATA TRANSFER ORGANIZATION

The ALU provides the arithmetic and logical capabilities of the MP. This unit combines two input words of the system word length. These two inputs are combined according to the function code specified in the micro instruction. The result is immediately available at the output of the ALU for possible shifting via selector S3 and delivery to the destination register, memory interface, panel interface, and I/O. The unshifted output of the ALU is delivered to the SM and mask registers. The ALU output can be ignored on an operation. The results of the ALU operation regarding sign, zero, and magnitude (by means of carryout test) are available to the test bit logic for instruction sequencing.

The data transfer organization of the MP provides for storing data in one of six working registers and two



*File 1 is furnished as an option.

Figure 2-2. Detailed Block Diagram of 1700 Enhanced Processor

files, and for selecting data for processing through the ALU. ALU results are transferred back to one of the registers or out of the organization to control external equipment.

The primary data registers are I, P, A, F, X, and Q.

The following are brief descriptions of the primary registers. Table 3-3 contains a comparison of the MP registers with 1700 registers.

- I Register — A word-length register whose only input and output is the selector S1. This register should not be confused with the 1700 I register (location $00FF_{16}$).
- P Register† — A word-length, general-purpose register that receives data from the ALU and provides output to S1. Normally it is used to hold the software instruction counter.
- A Register† — A word-length, general-purpose register that receives data from the ALU and provides output to S1. The A register is mechanized as a shifting register, and can be shifted left or right without using the ALU. The A register may also be combined with the Q register to form a double-length shifting register that operates independently of the ALU.
- F Register — A word-length, general-purpose register that receives data from the ALU and provides data to S1 or S2 as ALU input. This register is also used as the file entry register and contains information written into the files when they are used as the destination of an ALU operation.
- X Register — A word-length, general-purpose register that receives data from the ALU and provides data to S1 or S2.
- Q Register† — A word-length, general-purpose register that receives data from the ALU and provides output to S2. The Q register is mechanized as a shifting register. It may be shifted left or right in conjunction with the A register without using the ALU.

Other major portions of the standard MP are:

- File 2 — A 32-word scratchpad file that may be used as a general-purpose, word-sized register. It delivers its output to S1 and S2; data input is provided by the F register. File 2 is reserved for the emulator, except for registers R1, R2, R3, and R4, which are available to the 1700 programmer through enhanced instructions.
- Bit Generator (BG) — The BG circuit generates one bit at any position in a word as input to the B side of the ALU. Control to drive the bit generator is derived from either the micro instruction (bits 27 to 31) or the lower five bits of the N register. Control is usually obtained from the micro instruction. A bit setting in an SM register determines the input that will drive the bit generator.
- Status/Mode Register (SM) — The SM register allows the micro program to control the mode of operation and also allows the micro program to examine the status of certain internal and external conditions. The MP can access one of two SM registers, SM1 and SM2.

  The SM register module contains 16 bits of SM1 and 16 bits of SM2. All 32 bits of an SM module can be set or reset by the micro program by transferring information to the SM register from the output of the ALU. Master clear will also clear SM1 and SM2.
- Interrupts and Mask Register — The interrupt system is implemented as a sampled data system at the micro-program level, instead of a true vectored interrupt system as used in conventional computers.

  The mask register enables the micro processor to disable/enable interrupts. The MP can access two mask registers, M1 or M2. For each mask bit there is a corresponding bit in the interrupt register.

  M1 is available to the 1700 programmer through the DMI instruction, while M2 (referred to as M) is available through the basic inter-register instruction (see section 4).

  Interrupts are identified by their corresponding mask bits, which are assigned to control the interrupt recognition. The bits in the mask registers are identified as follows:

  - Mask Register 1 (M1): M100 through M115
  - Mask Register 2 (M2): M200 through M215

†Available to the 1700 programmer.

Interrupt addresses are generated by the interrupt address encoder, according to the assignments given in table 2-1.

TABLE 2-1. MP MASK REGISTER/INTERRUPT ADDRESSES

Mask Bit	Interrupt Address Mask Register 1	
M100	15	Lowest Priority (M1)
M101	14	
M102	13	
M103	12	
M104	11	
M105	10	
M106	09	
M107	08	
M108	07	
M109	06	
M110	05	
M111	04	
M112	03	
M113	02	
M114	01	
M115	00	Highest Priority (M1)
	Interrupt Address Mask Register 2	
M200	31	Lowest Priority (M2)
M201	30	
M202	29	
M203	28	
M204	27	
M205	26	
M206	25	
M207	24	
M208	23	
M209	22	
M210	21	
M211	20	
M212	19	
M213	18	
M214	17	
M215	16	Highest Priority (M2)

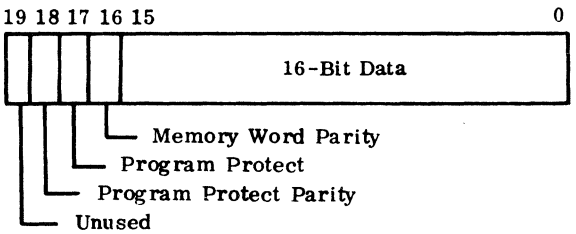
Note: The interrupt address generated is the same as its priority level; i.e., the highest priority interrupt generates a 0 interrupt address and the lowest priority interrupt generates a 31 interrupt address.

The interrupt priorities correspond to the interrupt address generated; that is, interrupt address 00 is associated with the highest priority interrupt line and interrupt address 31 is associated with the lowest priority interrupt line. For example, an interrupt associated with M112 would have priority over an interrupt associated with M111, and an interrupt address of 3 would be developed by the interrupt address encoder.

- K Register — An eight-bit counter that can be cleared, incremented, or decremented. It is used to address file 1 in addition to any program usage as a counter.
- N Register — An eight-bit counter that may be cleared, incremented, or decremented. It is used to address file 2, control shifts, control the scale operations, and may be used as an iteration counter that controls micro-instruction execution.
- N/K Register — The N and K registers may be combined to provide operand addresses outside the current operating micro page.
- File 1 — An optional file of 256 general-purpose, word-sized registers that are addressed by the contents of the K register. The output of the addressed file is delivered to S1 and S2 and thus to the A and B side of the ALU on demand. This file 1 input to selectors S1 and S2 is a submultiplexed input to the ALU. Thus, depending on the state of status mode bit (SM111), either file 1 or transform data can be selected as either an A or B input to the ALU.

MACRO MEMORY

Macro memory for the MP consists of 8K core memory stacks and an interface card. The interface card provides the control and interfacing required for MP/memory function and peripheral (DMA) equipment/memory functions. The 8K memory stacks are in 20-bit format:



The parity and program protect bits are generated and tested in the interface card. One interface card will handle up to four stacks (32K) in the main MP chassis.

Minimum memory cycle time is 600 ns, which is defined as the shortest possible time between successive read operations in macro memory. Minimum macro memory cycle time is 700 ns for write operations.

MACRO MEMORY CONFIGURATION

The macro memory configuration is shown in figure 2-3.

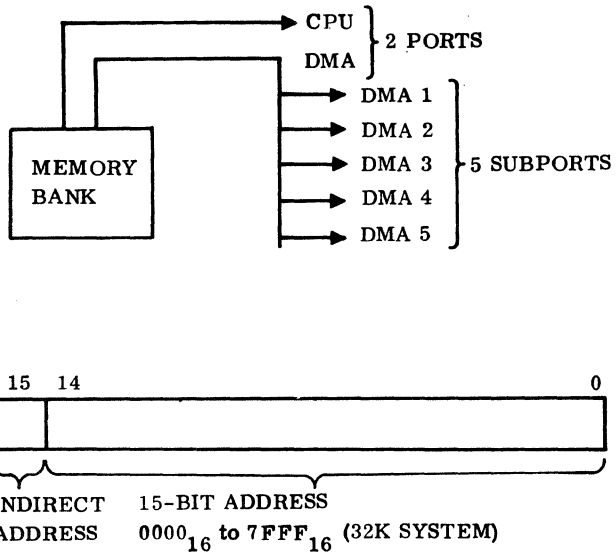


Figure 2-3. Macro Memory Configuration

The core memory configuration (for 8K to 32K) is a one-bank, two-port memory. One bank signifies that only one reference may take place at one time. Two ports provide two independent data and control paths to the memory; either port may request memory independent of any operation underway on the other port. The ports are CPU and DMA (direct memory access).

I/O-TTY MODULE

Figure 2-4 illustrates major signal flow paths to and from the I/O-TTY module.

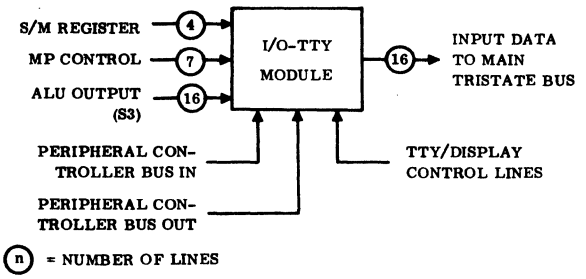


Figure 2-4. Major Signal Flow Paths of I/O-TTY Module

This module includes the following components:

- Real-Time Clock — In conjunction with the micro code it appears as a 1700 peripheral to the macro-level programmer.
- I/O Teletypewriter/Display Control — This controller is an integral part of the module. It interfaces to Teletype Corporation ASR/KSR 33/35 teletypes and to the Control Data RS232-C compatible conversational display terminals.
- Internal Peripheral Controller Bus — Provides all I/O data lines, interrupts, and control signals necessary to generate, in conjunction with the micro code, an internal CDC 1700 A/Q (input/output) bus. This TTL-level bus is intended to interface with controllers located in the basic MP chassis.
- Panel Interface Simulation — A logic section that is required when a panel/program device is used for operator input in the panel mode.

The MP is interfaced to the I/O module as follows:

- ALU Output — All output data and address information is provided from the output of the ALU via S3.
- SM Register — All commands to peripheral controllers are generated by micro code manipulation of the MP status mode register.
- MP Control — Timing and control information for controlling internal I/O module data gating is provided from the MP control signals.

- Interrupts — Interrupts from peripheral controllers (within the basic chassis) are wired directly from the peripheral controller module to the MP.
- Input Data and Peripheral Response Signals — All of these are provided to the MP on the main CPU tristate bus.
- Real-Time Clock — An integral part of the I/O module, the real-time clock appears as a 1700 peripheral to the macro-level software. Two functions are available to the macro-level program: Enable Limit Interrupt and Disable Limit Interrupt. Two status bits are also available to the macro-level program: Limit Interrupt and Lost Count.

The user may use his own design for I/O interfacing to facilitate use of special hardware.

MAINTENANCE INTERFACE/MAINTENANCE PANEL

The maintenance panel interface is an optional circuit module available for manual interface to the micro processor. The panel interface provides interfaces for a maintenance panel or for an RS232-C compatible console that has full-duplex serial ASCII characteristics. A card slot is prewired for the panel interface card control, and data lines tie directly into the control cards and to the ALU.

This section discusses the operating procedure for the micro-programmable computer in general terms. Since each user will have a different equipment application and setup, it is recommended that the user evaluate and develop his own operating procedure. The following sections present a general outline for startup and shutdown actions. Included is a description of the normal operator's interface to the MP.

STARTUP

The following startup sequence is a suggested outline:

1. Power-On Switch. Turn the MP power-on switch to the ON position.
2. Peripheral Power On Sequence. Turn on all peripherals and auxiliary power units.

EMULATOR OR MACRO-PROGRAM DEADSTART

1. Master clear the machine.
2. Place the emulator or macro-program deadstart deck in the reader.
3. Press ESCAPE on the panel/program device.
4. Depress the deadstart switch.

SHUTDOWN

De-energize all peripherals. Position the power-on switch to the OFF position.

SYSTEM FAILURE

After a system failure, follow the startup procedure and deadstart/autoload for restart.

MSOS AUTOLOAD

1. Master clear
2. Depress the autoload button for the mass storage controller
3. Press ESCAPE on the panel/program device
4. Type K31002800:
5. Type I@
6. After the initial MSOS messages, press ESCAPE on the panel/program device
7. Set the program protect by typing J28@
8. Input data/time on the panel/program device and continue

OPERATOR INTERFACE FOR THE MP

The normal MP configuration will include a CRT display unit as the panel/program device. The panel/program device is connected to the MP through the I/O-TTY card. It will function as a panel interface or a program (input/output) device.

FUNCTION CONTROL REGISTER (FCR)

The function control register (table 3-1) is the basic means of communication between the MP and the panel/program device in the panel interface mode. The eight hexadecimal digits (32 bits) of the FCR can be grouped as follows (0 is highest order):

Display:	Digits 0 and 1
Machine Modes:	Digits 2 to 5
Machine Status:	Digits 6, 7

The display digits determine which individual registers of two groups of registers (identified in table 3-2) can be displayed and/or modified. Digits 2 to 5 of the FCR are used to set such conditions as selective stop/on/off, step/run mode, etc.

TABLE 3-1. FUNCTION CONTROL REGISTER (FCR)

Bit		Digit	Bit Definition
(LSB) 31	1F	7	Overflow
30	1E		Not Protected Instruction
29	1D		Protect Fault
28	1C		Parity Error
27	1B	6	Interrupt System Active
26	1A		Auto-Restart Enabled
25	19		Micro Running
24	18		Macro Running
23	17	5	Not used
22	16		Not used
21	15		Enable Auto Display
20	14		Enable Console Echo
19	13	4	Enable Micro Memory Write
18	12		Multilevel Indirect Addressing Mode
17	11		Not used
16	10		Suppress Console Transmit
15	0F	3	0 0 Breakpoint Off
14	0E		0 1 Instruction Reference BP
			1 0 Storage Operand BP
			1 1 All References BP
13	0D		BP Interrupt (BP Stop if Clear)
12	0C		Micro BP, Step, Go, Stop (Macro if Clear)
11	0B	2	Step
10	0A		Selective Stop
09	09		Selective Skip
08	08		Protect Switch
07	07	1	DISPLAY 1
06	06		
05	05		
04	04		
03	03	0	DISPLAY 0
02	02		
01	01		
(MSB) 00	00		

Status Only

TABLE 3-2. DISPLAY CODE DEFINITIONS

Code		Display 1	Display 0
0	0 0 0 0	FCR	F2 (Addressed by N)
1	0 0 0 1	P†	N (MSBs)††
2	0 0 1 0	I	K (LSBs)††
3	0 0 1 1		X
4	0 1 0 0	A†	Q
5	0 1 0 1	MIR	F
6	0 1 1 0	BP/P-MA (Display Only)	F1 { Addressed by K Enabled by SM111
7	0 1 1 1	P-MA (Display Only)	MEM
8	1 0 0 0	SM1	
9	1 0 0 1	M1	$\overline{\text{RTJ}}$
A	1 0 1 0	SM2	
B	1 0 1 1	M2	
C	1 1 0 0		MM
D	1 1 0 1	A*	
E	1 1 1 0	X*	
F	1 1 1 1	Q*	

† Used to address macro memory. Automatically incremented after each memory reference.

†† The combined contents of these two registers are used to address micro memory. The K register is automatically incremented after each memory reference. The N register does not automatically increment.

The two least significant digits (6, 7) of the FCR are set by the MP and indicate the machine status, such as overflow on/off, macro storage parity error, protect fault, etc.

NOTES

1. Bits 14_{16} and 15_{16} of the FCR (Enable Console Echo and Enable Auto Display) are mutually exclusive; that is, the operator may select one or the other, but not both simultaneously.
2. Digit 3 of the FCR (bits $0C_{16}$ to $0F_{16}$), Breakpoint, is applicable only if the user has the optional maintenance panel and panel interface card.
3. Unassigned display codes (table 3-2) should be assumed to be undefined.
4. Selecting BP or P/MA (table 3-2) will result in both BP and P-MA being displayed. BP is the leftmost 16 bits and P-MA is the rightmost 16 bits. BP can be modified only if BP is selected; P-MA cannot be modified in either case.
5. Selecting N or K (table 3-2) will result in both N and K being displayed. N is the left eight bits and K is the right eight bits. However, when N is selected only the N register can be modified; when K is selected only the K register can be modified.

AUTO-DISPLAY

When auto-display is enabled, the register selected by the control code and display code will be output to the operator's interface and continuously updated (assuming the operator's interface contains a display and not a teletypewriter). With auto-display enabled, depressing a terminator (:, G or @) with no characters preceding it will cause a go signal.

PANEL INTERFACE CONTROL COMMANDS

The control commands used in the panel interface mode include: H, I, J, K, L, @, :, G, and ?. Control commands H through L identify the type of data or operation entered or returned. The at symbol (@), the colon (:), and G all perform an entry termination function. The @ will also cause the operator's interface to go from the panel interface mode to program (A/Q) mode. The question mark, ?, generates a master clear.

A normal entry consists of one control character H through L; two, four, or eight hexadecimal digits 0 through F; and a terminating entry (: or G), in that order.

A normal response consists of the control character identifying the data that follows and four or eight hexadecimal digits. If a transmission or operator error occurs on the entry, an asterisk (*) precedes the control character and the function control register is unconditionally displayed with the last legal control character. All entries except the ? cause a response, unless bit 10_{16} (Suppress Console Transmit) of the FCR is set. The following are examples of the control functions. The colon (:) is used as the terminating entry.

- Master Clear — A master clear can be generated in several ways:
  - A power on master clear
  - The MC button on the maintenance panel
  - A signal from a peripheral controller
  - A question mark from a panel device (programmers console)

  NOTE

  Baud rate compatibility between the panel device and the machine must exist for ? master clear.

- Stop/Go Control — The following entry will cause a go:

  I: (Initiate)

This is a micro go if bit 12 of the FCR is set. It is both a micro and macro go if bit 12 of the FCR is clear.

The I control function may also be used to set a bit in the FCR.

The following entry will cause a stop:

H: (Halt)

This is a micro stop if bit 12 of the FCR is set. It is a macro stop if bit 12 of the FCR is Clear.

The response to a start or stop entry is a display of the FCR.

The H control function may also be used to clear a specific bit in the FCR. The entry

H14:

would clear bit 14_{16} in the FCR and the response would be a display of the updated FCR.

NOTE

The clear and set capabilities of the H and I control functions are not available in the panel simulation mode.

- J Control Function — The J control function is used to replace the contents of the function control register in a digit mode. While it may be used to change the value of any FCR digit, it is generally used to change digits 0 and 1. The value of Display 0 and Display 1 specifies which MP parameter is displayed on display requests, or entered on enter requests (refer to table 3-3). J functions always consist of J followed by two hexadecimal digits and a terminator (:, G, or @). The first hexadecimal digit specifies the FCR digit 0 through 5 and the second hexadecimal digit specifies the value the digit is to assume, 0 through F.

The function code

J14:

TABLE 3-3. MP/1700 REGISTER CORRESPONDENCE

MP	1700
P	P
A	A
Q	Q
X	(P) (i.e., next instruction) (display only)
I	I (see notes 1 and 2) (display only)
F2(1)	R1
F2(2)	R2
F2(3)	R3
F2(4)	R4
F2(5)	Q (display only)
F2(6)	A (display only)
F2(7)	I (see notes 1 and 2)
M2	M

NOTE: To change I:
1. Change location $00FF_{16}$
2. Change F2(7)

would set FCR digit 1 to 4 (select the A register), and the response would be a display of the updated FCR.

The J code is also used to alternately display the upper and lower 16 bits of a 32-bit register on the 16-bit maintenance panel display.

In the panel simulation mode, J: will result in the display of the entire FCR register. There is no upper/lower mode.

- K Control Function — The K control function is used to display or enter data into the parameter specified by Display 1. The K function uses two formats. The first format is a request to display the parameter specified by Display 1:

  K:

  The second format is an enter data request. The data is entered into the parameter specified by Display 1. It consists of K followed by four or eight hexadecimal digits, followed by a terminator (:, G, or @). The hexadecimal digits are the data to be entered. For example:

  - To display the P register, type:

    J11: Set Display 1 to P register (FCR Digit 1 = 1_{16}).

    K: Display parameter selected in Display 1.

  - To enter $14FE_{16}$ into the breakpoint register, type:

    J16: Set Display 1 to BP register (FCR Digit 1 = 6_{16}).

    K14FE: Enter data into parameter selected in Display 1.

- L Control Function — The L function is operationally the same as the K function, except that it is associated with Display 0.

NOTE

When macro memory is displayed or entered, the register selected in Display 1 is the macro memory address. The Display 1 selection must be the P or A register. This register is incremented by 1 after the display. In the panel simulation mode, the Display 1 selection must be the P register. When micro memory is displayed or entered, the K register is the eight least significant bits of the address, and the N register provides the remaining bits. The K register is incremented by 1 after the display.

- Breakpoint (BP) — There are two types of breakpoint: micro and macro. If bit 12 of the FCR is set, micro breakpoint is selected. If bit 12 is clear, macro breakpoint is selected. In the panel simulation mode there is no micro or macro breakpoint capability.

  Bits 14 and 15 of the FCR are used to select three types of macro BP:

Bit 14	Bit 15	
0	0	Breakpoint not selected
0	1	Instruction reference BP
1	0	Store operand BP
1	1	All references BP

A macro breakpoint occurs if the breakpoint register is equal to the macro memory address and the select conditions are met. For example:

J16: Set display 1 to breakpoint register.

K0050: Set breakpoint register to 0050_{16}.

J31: Set macro mode and breakpoint on instruction reference.

A stop will occur after the instruction at macro location 50_{16} is executed.

If bit 13 of the FCR is set, an interrupt occurs when the breakpoint conditions are met rather than a stop.

For a micro breakpoint, P/MA is compared to the lower 12 bits of the breakpoint register. In addition, the upper/lower selection (32-bit select) is compared to bit 13 of the breakpoint register. If all bits are equal and the combination of FCR bits 14 and 15 is not zero, then a micro stop occurs. If FCR bit 14 is set, then a comparison of FCR bit 13 and the upper/lower selector is not required.

- Auto Display — When auto display is enabled, the register selected by the control and display codes will be output to the operator's interface and continuously updated as long as the interface is a display terminal and not a teletypewriter. Depressing a terminator (:, G, or @) with no characters preceding it will cause a go signal, which is useful for stepping through a micro or macro program.

NOTE

Auto-display mode and echo mode should never be selected simultaneously. In other words, FCR bits 20 and 21 should be mutually exclusive.

PANEL/PROGRAM MODE COMMANDS

Commands for use in the program mode are escape (ESC) and manual interrupt. The ESC command will cause the panel/program device to go from program mode to panel interface mode. It sets the reserve status line, which will indicate to the software that the panel/program device is busy if the macro program would attempt to reference it.

The manual interrupt is generated by a control G (BELL) command. It is used instead of a console manual interrupt button.

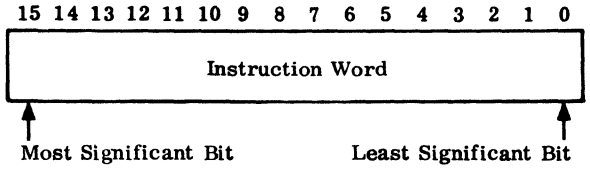
The command for use in panel mode is the @ symbol. It will generate a release reserve as it causes the panel/program device to enter into the program mode from the panel mode. Selecting the @ during program mode will be accepted as a normal ASCII character with no special function.

I/O OPERATIONS

With the exceptions specified in the program mode commands, the program mode is to be used as standard operator data interface to the MP for I/O.

INSTRUCTION FORMAT

The MP computer instruction word shown in the following example consists of 16 bits, numbered right to left as 0 to 15, with the leftmost bit, 15, being the most significant and the rightmost bit, 0, being the least significant.



Hexadecimal (base 16) notation is used in this computer.

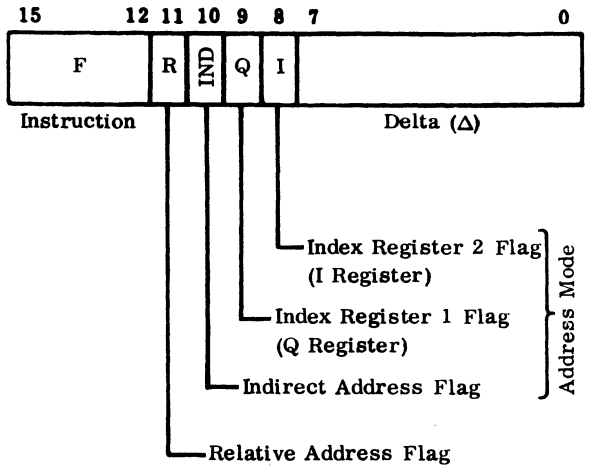
The MP computer is composed of a basic and an enhanced instruction set. The basic set is 1700-compatible and is divided into storage reference, register reference, inter-register, skip, and shift instructions. The enhanced instruction set is divided into the enhanced storage reference, field reference, enhanced inter-register, enhanced skip, decrement and repeat, and miscellaneous instructions.

BASIC INSTRUCTION SET

STORAGE REFERENCE

The storage reference instructions shown in the following illustration contain three fields: instruction, address mode, and delta. The instruction field contains the operation code.

The address mode field contains flags for indexing, indirect addressing, and relative addressing. The delta field is a signed eight-bit address modifier in which the most significant bit is the sign bit. Storage reference instructions have the following format:



Five types of addresses and/or address methods are created by these instructions:

- Instruction Address — The address of the instruction being executed; also called P
- Indirect Address — A storage address that contains an address rather than an operand
- Base Address — The operand address after all indirect addressing but before modification by the index registers. The base address is the effective address when no indexing is specified.
- Effective Address — The final address of the operand. At certain times the effective address equals the operand for read-operand type instructions (refer to table 4-1).
- Indexing — The computer has two index registers. Index register 1 is the Q register; index register 2 is storage location $00FF_{16}$ (I register). The base address may be modified by either or both of the index registers. If the index 1 flag is set, the contents of the Q register are added to the base address.

TABLE 4-1. STORAGE REFERENCE INSTRUCTION ADDRESSING

Mode	Binary 11 10 9 8	Hex.	Δ Delta	Effective Address	Address of Next Instruction
8-Bit Absolute	0000	0	≠0	Δ	P+1
	0001	1		Δ+(00FF)	
	0010	2		Δ+(Q)	
	0011	3		Δ+(Q)+(00FF)	
8-Bit Absolute Indirect††	0100	4		(Δ)	
	0101	5		(Δ)+(00FF)	
	0110	6		(Δ)+(Q)	
	0111	7		(Δ)+(Q)+(00FF)	
8-Bit Relative	1000	8		P+Δ	
	1001	9		P+Δ+(00FF)	
	1010	A		P+Δ+(Q)	
	1011	B		P+Δ+(Q)+(00FF)	
8-Bit Relative Indirect††	1100	C		(P+Δ)	
	1101	D		(P+Δ)+(00FF)	
	1110	E		(P+Δ)+(Q)	
	1111	F		(P+Δ)+(Q)+(00FF)	
Absolute Constant	0000	0	=0	P+1	P+2
	0001	1		(P+1)+(00FF)†	
	0010	2		(P+1)+(Q)†	
	0011	3		(P+1)+(Q)+(00FF)†	
16-Bit Storage††	0100	4		(P+1)	
	0101	5		(P+1)+(00FF)	
	0110	6		(P+1)+(Q)	
	0111	7		(P+1)+(Q)+(00FF)	
16-Bit Relative	1000	8		P+1+(P+1)	
	1001	9		P+1+(P+1)+(00FF)	
	1010	A		P+1+(P+1)+(Q)	
	1011	B		P+1+(P+1)+(Q)+(00FF)	
16-Bit Relative Indirect††	1100	C		(P+1+(P+1))	
	1101	D		(P+1+(P+1))+(00FF)	
	1110	E		(P+1(P+1))+(Q)	
	1111	F		(P+1+(P+1))+(Q)+(00FF)	

†Effective address is the operand for read-operand type instructions.

††Multilevel only in 32K mode

to form the effective address. If the index register 2 flag is set, the contents of storage location $00FF_{16}$ (I register) are added to the base address to form the effective address. If both index register flags are set, the contents of Q are added to the base address; then the contents of $00FF_{16}$ are added to the result to form the effective address. B (for both) is used for indexing both the Q and I register. Indexing occurs after completion of indirect addressing.

The computer uses the 16-bit ones complement adder during indexing operations. Consequently, the index register contents are treated as signed quantities (bit 15 is the sign bit).

The storage reference instructions (refer to table 4-1) have eight different types of addressing modes: eight-bit absolute, eight-bit absolute indirect, eight-bit relative, eight-bit relative indirect, absolute constant, 16-bit storage, 16-bit relative, and 16-bit relative indirect.

- Eight-Bit Absolute (address mode bits = 0, 1, 2, or 3) — Both relative and indirect flags are set to 0 and delta is not set to 0. The base address equals delta. Delta has no sign bit. The contents of the index registers, when specified, are added to the base address to form the effective address.
- Eight-Bit Absolute Indirect (address mode bits = 4, 5, 6, or 7) — The relative address flag is set to 0, the indirect flag is set to 1, and delta is not set to 0. The eight-bit value of delta is an indirect address. Delta is a magnitude quantity for this operation (no sign bit).
- Eight-Bit Relative (address mode bits = 8, 9, A, or B) — The relative flag is set to 0, and delta is not set to 0. The base address is equal to the instruction address P plus the value of delta with sign extended. The contents of the index registers, when specified, are added to the base address to form the effective address.
- Eight-Bit Relative Indirect (address mode bits = C, D, E, or F) — Both relative and indirect flags are set to 1. If delta is not set to 0, the value of the instruction address P plus the value of delta with sign extended is an indirect address. If bit 15 of the contents of this indirect address is 0, the contents of this indirect address is the base address. If bit 15 of the contents of the indirect address is set when the computer is in 32K mode, another indirect address is indicated.
- Absolute Constant (address mode bits = 0, 1, 2, or 3) — Both relative and indirect flags and delta are set to 0.

  When the address mode bits are set to 0, P + 1 is the effective address. When the address mode bits are set to 1, 2, or 3, the contents of P + 1 plus the contents of one or both index registers form the effective address. The effective address is taken as the operand for read-operand type instructions.
- 16-Bit Storage (address mode bits = 4, 5, 6, or 7) — The relative address flag and delta are set to 0 and the indirect flag is set to 1. The contents of location P + 1 is an indirect address. When the base address is formed (indirect addressing complete), the contents of one or both index registers, if specified, are added to form the effective address.
- 16-Bit Relative (address mode bits = 8, 9, A, or B) — The relative address flag is set to 1, and the indirect address flag and delta are set to 0. If no indexing is specified, the instruction address P + 1 plus the contents of location P + 1 form the base address or effective address. If indexing is specified, the contents of the specified index register(s) are added to the base address to form the effective address.
- 16-Bit Relative Indirect (address mode bits = C, D, E, or F) — Both relative and indirect flags are set to 1. In 65K mode, the contents of P + 1 + (P + 1)† is the base address. Then the contents of the index registers, when specified, are added to the base address to form the effective address. In 32K mode, P + 1 + (P + 1) = base address if bit 15 of (P + 1) is 0; if 1, then P + 1 + (P + 1) forms an indirect address. This process continues until bit 15 = 0.

Table 4-2 shows all the addressing possibilities for storage reference instructions that may be obtained through combinations of flag bits.

REGISTER REFERENCE

Register reference instructions (refer to table 4-3) use the address mode field for the operation code. These

† ( ) Denotes contents of expression

Length & type of instructions

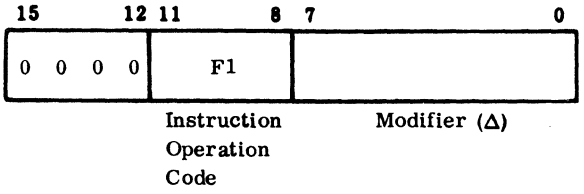
TABLE 4-2. STORAGE REFERENCE INSTRUCTIONS

Instruction	Mnemonic	Description
Unconditional Jump F = 1	JMP	Effective address specifies the location of the next instruction
Multiply Integer F = 2	MUI	Multiply the contents of the storage location specified by the effective address in the A register. The 32-bit product replaces the contents of Q and A with the most significant bits in the Q register. Ones complement arithmetic is used.
Divide Integer F = 3	DVI	Divide the combined contents of the Q and A registers by the contents of the effective address. The Q register contains the most significant bits before execution. The quotient is in the A register and the remainder is in the Q register at the end of execution. The overflow indicator is set if the magnitude of the quotient is greater than the capacity of the A register. Once set, the overflow indicator remains set until a skip on overflow instruction is executed.
Store Q F = 4	STQ	Store the contents of the Q register in the storage location specified by the effective address. The contents of Q are not changed.
Return Jump F = 5	RTJ	Replace the contents of the storage location specified by the effective address with the address of the next consecutive instruction. The address stored in the effective address will be P + 1 or P + 2, depending on the addressing mode of RTJ. The contents of P are then replaced with the effective address + 1.
Store A F = 6	STA	Store the contents of the A register in the storage location specified by the effective address. The contents of A are not altered.
Store A, Parity to A F = 7	SPA	Store the contents of the A register in the storage location specified by the effective address. Set the A register to 0001_{16} if the parity bit of the word stored in the effective address is set. If the parity bit is not set, set the A register to 0000.
Add to A F = 8	ADD	Add the contents of the storage location specified by the effective address to the contents of the A register. Ones complement arithmetic is used. The overflow indicator will be set if the magnitude of the sum is greater than the capacity of the A register. Once set, the overflow indicator will remain set until a skip on overflow instruction is executed.
Subtract from A F = 9	SUB	Subtract the contents of the storage location specified by the effective address from the contents of the A register. Ones complement arithmetic is used. The overflow operation is the same as in ADD.

TABLE 4-2. STORAGE REFERENCE INSTRUCTIONS (Continued)

Instruction	Mnemonic	Description
And with A F = A	AND	Form the logical product, bit-by-bit, of the contents of the storage location specified by the effective address and the contents of the A register. The result replaces the contents of A.
Exclusive OR with A F = B	EOR	Form the logical difference (exclusive OR), bit-by-bit, of the contents of the storage location specified by the effective address and the contents of the A register. The results replace the contents of the A register.
Load A F = C	LDA	Load the A register with the contents of the storage location specified by the effective address. The contents of the storage location are not altered.
Replace Add One in Storage F = D	RAO	Add 1 to the contents of the storage location specified by the effective address. The contents of A and Q are not changed. Ones complement arithmetic is used. Operation on overflow is the same as in ADD.
Load Q F = E	LDQ	Load the Q register with the contents of the storage location specified by the effective address. The contents of the storage location are not altered.
Add to Q F = F	ADQ	Add the contents of the storage location specified by the effective address to the contents of the Q register. Ones complement arithmetic is used. Operation on overflow is the same as in ADD.

instructions are identified by 0s in the upper four bits of an instruction and the F1 instruction operation code (address mode field) cannot be a one, eight, or 15:



INTER-REGISTER

Inter-register instructions (F1 = 8) are identified by an 8 in the address mode field and a 0 in the instruction mode field. These instructions (table 4-4) cause data from certain combinations of origin registers to be sent through the adder to any combination of destination registers. Various operations, selected by the adder control lines, are performed on the data as it passes through the adder. The inter-register instruction format is:

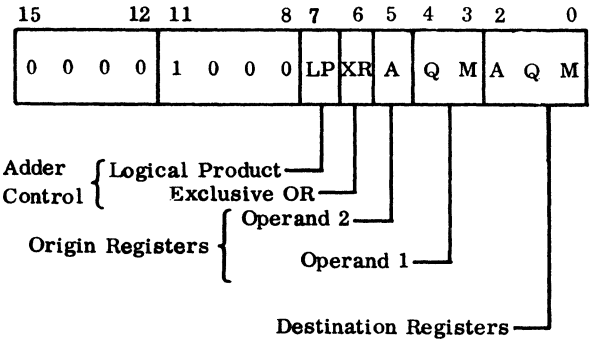


TABLE 4-3. REGISTER REFERENCE INSTRUCTIONS

Instruction	Mnemonic	Description
Selective Stop F1 = 0 Δ = 0	SLS	If this instruction is executed when the STOP switch is on, the machine is stopped. When the switch is off, the instruction becomes a pass.
Input to A F1 = 2	INP	Read one word from an external device into the A register. The word in the Q register selects the sending device. If the device sends a reply, the next instruction comes from P + 1. If the device sends a reject, the next instruction comes from P + 1 + Δ, where Δ is an eight-bit signed number including sign. An internal reject causes the next instruction to come from P + Δ.
Output from A F1 = 3	OUT	Output one word from the A register to an external device. The word in the Q register selects the receiving device. If the device sends a reply, the next instruction comes from P + 1. If the device sends a reject, the next instruction comes from P + 1 + Δ, where Δ is an eight-bit signed number including sign. An internal reject causes the next instruction to come from P + Δ.
Increase A F1 = 9	INA	Replace the contents of A with the sum of the initial contents of A and delta. Delta is treated as a signed number with the sign extended into the upper eight bits. Operation on overflow is the same as in ADD.
Enter A F1 = A	ENA	Replace the contents of the A register with the eight-bit delta, sign extended.
No Operation F1 = B Δ = 0	NOP	
Enter Q F1 = C	ENQ	Replace the contents of Q with the eight-bit delta, sign extended.
Increase Q F1 = D	INQ	Replace the contents of Q with the sum of the initial contents of Q and delta. Delta is treated as a signed number with the sign extended into the upper eight bits. Operation on overflow is the same as in ADD.
Enable Interrupt† F1 = 4 Δ = 0	EIN	Activate the interrupt system. The interrupt system must be active and the mask bit set for an interrupt to be recognized.

†These instructions are only legal when the PROGRAM PROTECT switch is off, or the instructions themselves are protected. If an instruction is illegal, it becomes a Selective Stop and an interrupt on Program Protect Fault is possible (if selected).

TABLE 4-3. REGISTER REFERENCE INSTRUCTIONS (Continued)

Instruction	Mnemonic	Description
Inhibit Interrupt† F1 = 5 Δ = 0	IIN	De-activate the interrupt system.
Set Program Protect† F1 = 6 Δ = 0	SPB	Set the program protect bit in the address specified by Q.
Clear Program Protect† F1 = 7 Δ = 0	CPB	Clear the program protect bit in the address specified by Q.
Exit Interrupt State† F1 = E	EXI	Exit from an interrupt state specified by delta. This instruction reads the address containing the return address, resets the overflow indicator according to bit 16, activates the interrupt system, and jumps to the return address.

†These instructions are only legal when the PROGRAM PROTECT switch is off, or the instructions themselves are protected. If an instruction is illegal, it becomes a Selective Stop and an interrupt on Program Protect Fault is possible (if selected).

The origin registers are considered as operands. There are two kinds:

- Operand 1 may be one of the following:

  $FFFF_{16}$ (bit 5 = 0)

  The contents of A (bit 5 = 1)

- Operand 2 may be one of the following:

  $FFFF_{16}$ (bit 4 = 0 and bit 3 = 0)

  The contents of M (bit 4 = 0 and bit 3 = 1)

  The contents of Q (bit 4 = 1 and bit 3 = 0)

  The OR, bit-by-bit, of the contents of Q and M (bit 4 = 1 and bit 3 = 1)

The following operations are possible (refer to table 4-5 for examples of all possible four-bit operands):

- LP = 0 and XR = 0 — The data placed in the destination register(s) is the arithmetic sum of operand 1 and operand 2. The overflow indicator operates the same as in ADD.
- LP = 1 and XR = 0 — The data placed in the destination registers is the logical product, bit-by-bit, of operand 1 and operand 2.
- LP = 0 and XR = 1 — The data placed in the destination registers is the exclusive OR, bit-by-bit, or operand 1 and operand 2.
- LP = 1 and XR = 1 — The data in the destination registers is the complement of the logical product, bit-by-bit, of operand 1 and operand 2.

SKIP

Skip instructions (F1=1) are identified by a 1 in the address mode field and a 0 in the instruction mode field:

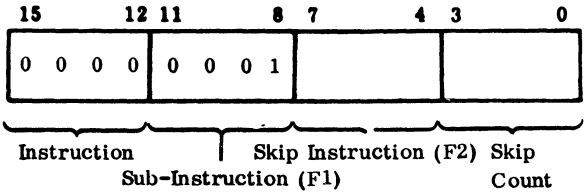


TABLE 4-4. INTER-REGISTER INSTRUCTIONS

Description	Mnemonics	Bit 7 6 5 4 3
Set to Ones	SET	1 0 0 0 0
Clear to Zero	CLP	0 1 0 0 0
Transfer A	TRA	1 0 1 0 0
Transfer Q	TRQ	1 0 0 1 0
Transfer Q or M	TRB	1 0 0 1 1
Transfer Complement A	TCA	0 1 1 0 0
Transfer Complement M	TCM	0 1 0 0 1
Transfer Complement Q	TCQ	0 1 0 1 0
Transfer Complement Q or M	TCB	0 1 0 1 1
Transfer Arithmetic Sum A, M	AAM	0 0 1 0 1
Transfer Arithmetic Sum A, Q, or M	AAB	0 0 1 1 1
Transfer Arithmetic Sum A, Q	AAQ	0 0 1 1 0
Transfer Exclusive OR A, M	EAM	0 1 1 0 1
Transfer Exclusive OR A, Q	EAQ	0 1 1 1 0
Transfer Exclusive OR A, Q, or M	EAB	0 1 1 1 1
Transfer Logical Product A, M	LAM	1 0 1 0 1
Transfer Logical Product A, Q	LAQ	1 0 1 1 0
Transfer Logical Product A, Q, or M	LAB	1 0 1 1 1
Transfer Complement Logical Product A, M	CAM	1 1 1 0 1
Transfer Complement Logical Product A, Q	CAQ	1 1 1 1 0
Transfer Complement Logical Product A, Q, or M	CAB	1 1 1 1 1

TABLE 4-5. INTER-REGISTER INSTRUCTION TRUTH TABLE

Operand 1	Operand 2	LP = 0 XR = 1	LP = 1 XR = 0	LP = 1 XR = 1	LP = 0 XR = 0
0	0	0	0	1	Arithmetic Sum
0	1	1	0	1	
1	0	1	0	1	
1	1	0	1	0	

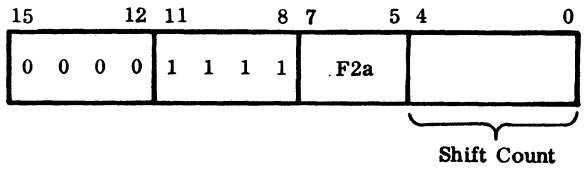
Notes:
1. Register transfers can be accomplished with LP = 0, XR = 0, and by making operand 1 or operand 2 equal to $FFFF_{16}$.
2. Without destroying either operand, magnitude comparisons can be done with LP = 0, XR = 0, no destination register selected, and by testing the overflow indicator.
3. Complementing registers can be done with LP = 0, XR = 1, and making operand 1 and operand 2 equal to $FFFF_{16}$.

When the skip condition is met, the contents of the skip count + 1 is added to P to obtain the address of the next instruction (e.g., when the skip count is 0, go to P + 1). When the skip condition is not met, the address of the next instruction is P + 1 (skip count ignored). The skip count does not have a sign bit.

The skip instructions are listed in table 4-6.

SHIFT

Shift instructions are identified by a 15 in the address mode field and a 0 in the instruction mode field. These instructions shift A or Q, or QA left or right for the number of places specified by the five-bit shift count. The sign is extended on right shifts. Left shifts are end-around. This instruction has the following format:



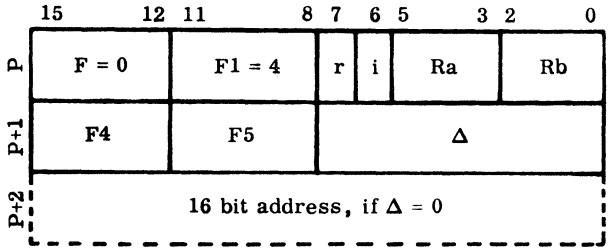
The shift instructions (F2a) are listed in table 4-7.

ENHANCED MP INSTRUCTIONS

Instruction formats for enhancements to the 1700 instruction repertoire are upward-compatible with the existing 1700 computers. They make use of previously undefined instruction formats.

ENHANCED STORAGE REFERENCE

These instructions have the following format:



The enhanced storage reference instructions are identified when the F field is 0, the F1 field is equal

TABLE 4-6. SKIP INSTRUCTIONS

Instruction	Mnemonic	Description
Skip if A = +0	SAZ	F2 = 0
Skip if A ≠ +0	SAN	F2 = 1
Skip if A = +	SAP	F2 = 2
Skip if A = -	SAM	F2 = 3
Skip if Q = +0	SQZ	F2 = 4
Skip if Q ≠ +0	SQN	F2 = 5
Skip if Q = +	SQP	F2 = 6
Skip if Q = -	SQM	F2 = 7
Skip if switch is set	SWS	F2 = 8
Skip if switch is not set	SWN	F2 = 9
Skip on overflow. SOV clears the overflow indicator.	SOV	F2 = A
Skip on no overflow	SNO	F2 = B
Skip on storage parity error. SPE clears the storage parity error signal and indicator.	SPE	F2 = C
Skip on no storage parity error	SNP	F2 = D
Skip on program protect fault†	SPF	F2 = E
Skip on no program protect fault†	SNF	F2 = F

†The program protect fault is set by:

1. A nonprotected instruction attempting to write into a protected address.
2. A protected instruction executed immediately following a nonprotected instruction, except when an an interrupt has caused the instruction sequence.
3. Execution of any nonprotected instruction that attempts to alter the interrupt system.

The program protect fault is cleared when an SPF or SNF is executed. The program protect fault cannot be set if the program protect system is disabled.

TABLE 4-7. SHIFT INSTRUCTIONS

Instruction Name	Mnemonic	Description
Q Right Shift	QRS	F2a = 1
A Right Shift	ARS	F2a = 2
Long Right Shift (QA)	LRS	F2a = 3
Q Left Shift	QLS	F2a = 5
A Left Shift	ALS	F2a = 6
Long Left Shift (QA)	LLS	F2a = 7

to 4, and the r, i, Ra, and Rb fields are not all 0. (If these fields are all 0, the instruction is an EIN.) This instruction is made up of two (or three, if delta is 0) words.

The enhanced storage reference instructions are similar to the basic storage references in that they contain four parts: instruction field (F4), instruction mode field (F5), addressing mode fields (delta, r, i, and Ra), and register Rb. Two operands (A and B) are specified for executing the instruction.

The F4 field determines the instruction (e.g., add, subtract, etc.). The F5 field determines the instruction mode:

F5 = 0	Word processing; register destination
1	Word processing; memory destination
2	Character processing; register destination
3	Character processing; memory destination

NOTE

F5 is not used for subroutine jumps and subroutine exit. The register/memory destination bit of F5 is not used for compare instructions (see below).

The addressing mode fields contain four fields:

1. Delta determines eight- or 16-bit addressing. If delta is 0, a third word will be required to specify a 16-bit address.
2. Flag r is the relative address flag.
3. Flag i is the indirect address flag.
4. Register Ra is the index register.

The addressing modes are similar to the basic storage instructions. The basic set allows indexing by one or two registers (I and Q); while the enhanced set allows indexing by any one of seven registers (1, 2, 3, 4, Q, A, or I). Table 4-8 specifies the addressing modes, the effective address, and the address of the next instruction.

The addressing mode fields determine the effective address for operand A. Register Rb and the instruction mode field (F5) determine the address for operand B. Note that for character addressing, the effective address (operand A and register Rb) are combined to ascertain the actual character effective address (refer to the character instructions in table 4-9). Operand B is always the A register for character addressing.

CAUTION

For character addressing, selection of absolute (r = 0), no indirect (i = 0), no index register (Ra = 0), and no character register (Rb = 0) will result in an EIN instruction.

Any unspecified combinations of F4, F5, and Rb are reserved for future expansion.

The following definitions apply to the description of addressing modes:

- Instruction Address — The address of the instruction being executed, also called P.
- Indirect Address — A storage address that contains an address rather than an operand. Note that there is no multilevel indirect addressing for enhanced storage reference instructions.
- Base Address — The operand address after all indirect addressing but before modification by an index register. The base address is the effective address if no indexing is specified.
- Effective Address — The final address of the operand.
- Indexing — If specified, the contents of register Ra is added to the base address to form the effective address. Indexing occurs after addressing is completed.

  The computer uses the 16-bit ones complement adder during indexing operations. Consequently, the index register contents are treated as signed quantities (bit 15 is the sign bit).
- Registers — Registers Ra and Rb are defined as follows:

Register	Value
None	0
1	1
2	2
3	3
4	4
Q	5
A	6
I	7

Enhanced storage reference instructions (table 4-8) have the following types of addressing modes:

- Eight-Bit Absolute — (r = 0, i = 0, and $\Delta \neq 0$) — The base address equals delta and the sign bit of delta is not extended. The contents of index register Ra, when specified, are added to the base address to form the effective address.
- Eight-Bit Absolute Indirect (r = 0, i = 1, and $\Delta \neq 0$) — The eight-bit value of delta is an

TABLE 4-8. ENHANCED STORAGE REFERENCE INSTRUCTION ADDRESSES

Addressing Mode	Delta	r	i	Ra	Effective Address (EA)	Address of Next Instruction
8-Bit Absolute	$\Delta \neq 0$	0	0	0	Δ	P + 2
		0	0	1	Δ + (1)	P + 2
		0	0	2	Δ + (2)	P + 2
		0	0	3	Δ + (3)	P + 2
		0	0	4	Δ + (4)	P + 2
		0	0	5	Δ + (Q)	P + 2
		0	0	6	Δ + (A)	P + 2
		0	0	7	Δ + (I)	P + 2
8-Bit Absolute Indirect	$\Delta \neq 0$	0	1	0	(Δ)	P + 2
		0	1	1	(Δ) + (1)	P + 2
		0	1	2	(Δ) + (2)	P + 2
		0	1	3	(Δ) + (3)	P + 2
		0	1	4	(Δ) + (4)	P + 2
		0	1	5	(Δ) + (Q)	P + 2
		0	1	6	(Δ) + (A)	P + 2
		0	1	7	(Δ) + (I)	P + 2
8-Bit Relative†	$\Delta \neq 0$	1	0	0	P + 1 + Δ	P + 2
		1	0	1	P + 1 + Δ + (1)	P + 2
		1	0	2	P + 1 + Δ + (2)	P + 2
		1	0	3	P + 1 + Δ + (3)	P + 2
		1	0	4	P + 1 + Δ + (4)	P + 2
		1	0	5	P + 1 + Δ + (Q)	P + 2
		1	0	6	P + 1 + Δ + (A)	P + 2
		1	0	7	P + 1 + Δ + (I)	P + 2
8-Bit Relative Indirect†	$\Delta \neq 0$	1	1	0	(P + 1 + Δ)	P + 2
		1	1	1	(P + 1 + Δ) + (1)	P + 2
		1	1	2	(P + 1 + Δ) + (2)	P + 2
		1	1	3	(P + 1 + Δ) + (3)	P + 2
		1	1	4	(P + 1 + Δ) + (4)	P + 2
		1	1	5	(P + 1 + Δ) + (Q)	P + 2
		1	1	6	(P + 1 + Δ) + (A)	P + 2
		1	1	7	(P + 1 + Δ) + (I)	P + 2

†For these addressing modes, delta is sign extended.

Note: ( ) Denotes contents of expression.

TABLE 4-8. ENHANCED STORAGE REFERENCE INSTRUCTION ADDRESSES (Continued)

Addressing Modes	Delta	r	i	Ra	Effective Address (EA)	Address of Next Instruction
Absolute Constant	Δ = 0	0	0	0	P + 2	P + 3
		0	0	1	P + 2 + (1)	P + 3
		0	0	2	P + 2 + (2)	P + 3
		0	0	3	P + 2 + (3)	P + 3
		0	0	4	P + 2 + (4)	P + 3
		0	0	5	P + 2 + (Q)	P + 3
		0	0	6	P + 2 + (A)	P + 3
		0	0	7	P + 2 + (I)	P + 3
16-Bit Storage	Δ = 0	0	1	0	(P + 2)	P + 3
		0	1	1	(P + 2) + (1)	P + 3
		0	1	2	(P + 2) + (2)	P + 3
		0	1	3	(P + 2) + (3)	P + 3
		0	1	4	(P + 2) + (4)	P + 3
		0	1	5	(P + 2) + (Q)	P + 3
		0	1	6	(P + 2) + (A)	P + 3
		0	1	7	(P + 2) + (I)	P + 3
16-Bit Relative	Δ = 0	1	0	0	P + 2 + (P + 2)	P + 3
		1	0	1	P + 2 + (P + 2) + (1)	P + 3
		1	0	2	P + 2 + (P + 2) + (2)	P + 3
		1	0	3	P + 2 + (P + 2) + (3)	P + 3
		1	0	4	P + 2 + (P + 2) + (4)	P + 3
		1	0	5	P + 2 + (P + 2) + (Q)	P + 3
		1	0	6	P + 2 + (P + 2) + (A)	P + 3
		1	0	7	P + 2 + (P + 2) + (I)	P + 3
16-Bit Relative Indirect	Δ = 0	1	1	0	(P + 2 + (P + 2))	P + 3
		1	1	1	(P + 2 + (P + 2)) + (1)	P + 3
		1	1	2	(P + 2 + (P + 2)) + (2)	P + 3
		1	1	3	(P + 2 + (P + 2)) + (3)	P + 3
		1	1	4	(P + 2 + (P + 2)) + (4)	P + 3
		1	1	5	(P + 2 + (P + 2)) + (Q)	P + 3
		1	1	6	(P + 2 + (P + 2)) + (A)	P + 3
		1	1	7	(P + 2 + (P + 2)) + (I)	P + 3

Note: ( ) denotes contents of expression

indirect address. The sign bit of delta is not extended. The content of this address in low core (addresses 0001_{16} to $00FF_{16}$) is the base address. The contents of index register Ra, when specified, are added to the base address to form the effective address.

- Eight-Bit Relative (r = 1, i = 0, and $\Delta \neq 0$) — The base address is equal to the instruction address plus one, P + 1, plus the value of delta with sign extended. The contents of index register Ra (when specified) are added to the base address to form the effective address.

  If no indexing takes place, the addresses that can be referenced in the eight-bit relative mode are restricted to the program area. Delta is eight bits long, thus the computer references a location between P - $7E_{16}$ and P + 80_{16} inclusive.

- Eight-Bit Relative Indirect (r = 1, i = 1, and $\Delta \neq 0$) — The address of the second word of the instruction, P + 1, plus the value of delta with sign extended is an indirect address. The content of this address is the base address. The contents of index register Ra, when specified, are added to the base address to form the effective address.

- Absolute Constant (r = 0, i = 0, and $\Delta = 0$) — The address of the third word of the instruction, P + 2, is the base address. The contents of the index register Ra, when specified, are added to the base address to form the effective address. Thus, when Ra is not specified, the contents of P + 2 is the value of the operand.

  Note that there is no immediate operand condition (i.e., indexing is specified and the instruction is a read-operand type) as there is for basic storage reference addressing.

- 16-Bit Storage (r = 0, i = 1, and $\Delta = 0$) — The base address equals the contents of P + 2. The contents of index register Ra, when specified, are added to the base address to form the effective address.

- 16-Bit Relative (r = 1, i = 0, and $\Delta = 0$) — The base address equals the contents of P + 2 plus P + 2. The contents of index register Ra, when specified, are added to the base address to form the effective address.

- 16-Bit Relative Indirect (r = 1, i = 1, and $\Delta = 0$) — The address of the third word of the instruction, P + 2, plus the contents of the third word of the instruction is an indirect address. The content of this address is the base address. The contents of the index register Ra, when specified, are added to the base address to form the effective address.

The instruction descriptions are given in table 4-9.

TABLE 4-9. ENHANCED STORAGE REFERENCE INSTRUCTIONS

Instruction	Mnemonic	Description
Subroutine/Jump Exit F4 = 5 F5 = 0 Rb = 0	SJE	Replace the contents of P with the effective address. This instruction can be used as a jump or subroutine exit. For example, if $\Delta = 1$ and Ra has been set up by a previous subroutine jump (see below), control will be returned following that subroutine jump. Note that subroutine jumps save the address of the instruction, rather than the next instruction, so the subroutine jump exit may be a two-word instruction ($\Delta \neq 0$) rather than three. For example, the following program makes a subroutine jump at location 1000. Register A will contain 1002 upon

TABLE 4-9. ENHANCED STORAGE REFERENCE INSTRUCTIONS (Continued)

Instruction	Mnemonic	Description
		entry to the subroutine SUB. Upon completion, SUB will exit to location 1003. 1000 0446 SJA+ SUB 1001 5000 1002 2000 1003 . . . . . . 2000 SUB . . . . 2020 0430 SJE- 1, A 2021 5001 CAUTION Since Rb = 0, a selection of absolute (r = 0), no indirect (i = 0), and no index register (Ra = 0) will result in an EIN instruction.
Subroutine Jump F4 = 5 F5 = 0 Rb = 1, 2, 3, 4, 5, 6, or 7 r = 1, 2, 3, 4, Q, A, or I	SJr	Load register r with the address of the last word of this instruction (i.e., P + 1 for $\Delta \neq 0$; P + 2 for $\Delta = 0$). The contents of P are then replaced with the effective address.
Add Register F4 = 8 F5 = 0 Rb = 1, 2, 3, 4, 5, 6, or 7 r = 1, 2, 3, 4, Q, A, or I	ARr	Add (using ones complement arithmetic) the contents of the storage location specified by the effective address to the contents of register r. Operation on overflow is the same as for the ADD instruction. The contents of storage are not altered.
Subtract Register F4 = 9 F5 = 0 Rb = 1, 2, 3, 4, 5, 6, or 7 r = 1, 2, 3, 4, Q, A, or I	SBr	Subtract (using ones complement arithmetic) the contents of the storage location specified by the effective address from the contents of register r. Operation on overflow is the same as for the ADD instruction. The contents of storage are not altered.
AND Register F4 = A F5 = 0 Rb = 1, 2, 3, 4, 5, 6, or 7 r = 1, 2, 3, 4, Q, A, or I	ANr	Form the logical product (AND), bit-by-bit, of the contents of the storage location specified by the effective address and the contents of register r. The result replaces the contents of register r. The contents of storage are not altered.

TABLE 4-9. ENHANCED STORAGE REFERENCE INSTRUCTIONS (Continued)

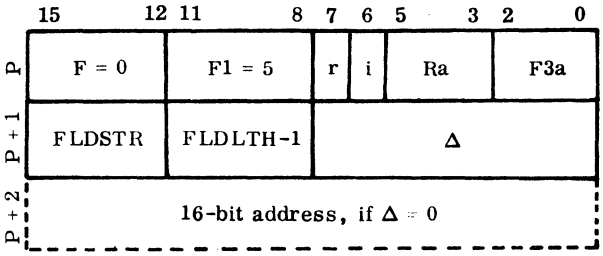
Instruction	Mnemonic	Description
AND Memory F4 = A F5 = 1 Rb = 1, 2, 3, 4, 5, 6, or 7 r = 1, 2, 3, 4, Q, A, or I	AMr	Form the logical product (AND), bit-by-bit, of the contents of the storage location specified by the effective address and the contents of register r. The result replaces the contents of the storage location specified by the effective address. The original contents of the storage location (specified by the effective address) replace the contents of the A register. The contents of register r are not altered unless r is the A register. Memory is locked until completion of the instruction. This instruction is useful for communication between MPs via memory.
Load Register F4 = C F5 = 0 Rb = 1, 2, 3, 4, 5, 6, or 7 r = 1, 2, 3, 4, Q, A, or I	LRr	Load register r with the contents of the storage location specified by the effective address. The contents of storage are not altered.
Store Register F4 = C F5 = 1 Rb = 1, 2, 3, 4, 5, 6, or 7 r = 1, 2, 3, 4, Q, A, or I	SRr	Store the contents of register r in the storage location specified by the effective address. The contents of register r are not altered.
Load Character to A F4 = C F5 = 2	LCA	Load bits A00 through A07 with a character from the storage location specified by the sum of the effective address and bits 1 to 15 of register Rb. Register Rb bit 0 set to 0 specifies the left character (bits 8 to 15) of the storage location; bit 0 set to 1 specifies the right character (bits 0 to 7). Bits A08 through A15 are cleared to zero. The contents of storage are not altered.
Store Character from A F4 = C F5 = 3	SCA	Store the contents of bits A00 through A07 into a character of the storage location specified by the sum of the effective address and bits 1 to 15 of register Rb. If bit 0 of register Rb is set to 0, the left character (bits 8 to 15) of the storage location is specified; if bit 0 is set to 1 the right character (bits 0 to 7) is specified. The contents of register A and other storage characters are not altered.
OR Register F4 = D F5 = 0 Rb = 1, 2, 3, 4, 5, 6, or 7 r = 1, 2, 3, 4, Q, A, or I	ORr	Form the logical sum (inclusive OR), bit-by-bit, of the contents of the storage location specified by the effective address and the contents of register r. The result replaces the contents of register r. The contents of storage are not altered.
OR Memory F4 = D F5 = 1 Rb = 1, 2, 3, 4, 5, 6, or 7 r = 1, 2, 3, 4, Q, A, or I	OMr	Form the logical sum (inclusive OR), bit-by-bit, of the contents of the storage location specified by the effective address and the contents of register r. The result replaces the contents of the storage location specified by the effective address. The original contents of the storage location

TABLE 4-9. ENHANCED STORAGE REFERENCE INSTRUCTIONS (Continued)

Instruction	Mnemonic	Description
		(specified by the effective address) replaces the contents of register A. The contents of register r are not altered unless r is the A register. Memory is locked until completion of the instruction. This instruction is useful for communication between MPs via memory.
Compare Register Equal F4 = E F5 = 0 Rb = 1,2,3,4,5,6, or 7 r = 1,2,3,4,Q,A, or I	CrE	Skip one location if the contents of register r and the contents of the storage location specified by the effective address are equal, bit-by-bit. If they are not, execute the next instruction. The contents of register r and storage are not altered.
Compare Character Equal F4 = E F5 = 2	CCE	Skip one location if the contents of bits 0 to 7 of register A and the character of the storage location specified by the sum of the effective address and bits 1 to 15 of register Rb are equal, bit-by-bit. If they are not, execute the next instruction. If bit 0 of register Rb is set to 0, the left character (bits 8 to 15) of the storage location is specified; if bit 0 is set to 1, the right character (bits 0 to 7) is specified. The contents of register A and storage are not altered. CAUTION Each compare instruction assumes that a one-word instruction follows it.

FIELD REFERENCE

These instructions have the following format:



Field reference instructions are identified when the F field is 0, the F1 field is equal to 5, and the r, i, Ra, and F3a fields are not all 0. (If these fields are all 0, the instruction is an IIN.)

Field reference instructions contain four parts: operation field (F3a), addressing mode fields (Δ, r, i, and Ra), FLDSTR, and FLDLTH-1 fields. The F3a field determines the operation (e.g., load, store). The addressing mode fields are defined exactly as the enhanced storage reference instructions. Refer to table 4-10 for descriptions of these instructions.

FLDSTR defines the starting bit of the field. For example, FLDSTR = 0 indicates that the field starts at bit 0. FLDLTH-1 defines the length of the field minus one. FLDLTH-1 = 0 indicates that the field is one bit long. If FLDLTH-1 = 0, the field reference instructions become bit reference instructions.

A field starts at the bit specified by FLDSTR and includes the contiguous FLDLTH bits to the right of that bit. No field may cross a word boundary (i.e., FLDSTR-FLDLTH-1 must be greater than or equal to 0). If FLDSTR = 0, the field length must be one bit long (FLDLTH-1 = 0).

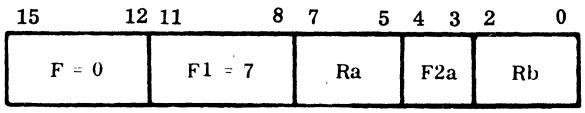
Note that F3a = 0, F3a = 1, and FLDSTR-FLDLTH-1 < 0 are reserved for future expansion.

TABLE 4-10. FIELD REFERENCE INSTRUCTIONS

Instruction	Mnemonic	Description
Skip if Field Zero F3a = 2	SFZ	Skip one location if the contents of the specified field of the storage location identified in the effective address are 0 (all bits are 0). If the contents are not 0, execute the next instruction.
Skip if Field Not Zero F3a = 3	SFN	Skip one location if the contents of the specified field of the storage location field identified in the effective address are nonzero (not all bits are 0). If the contents are zero execute the next instruction. CAUTION Each skip field instruction assumes that a one-word instruction follows it.
Load Field F3a = 4	LFA	Load register A, right justified, with the contents of the specified field of the storage location field identified in the effective address. All other bits of register A are cleared to 0. The contents of storage are not altered.
Store Field F3a = 5	SFA	Store the contents of the field from register A, right justified, into the specified field of the storage location identified in the effective address. All other storage bits are unchanged. Memory is locked until completion of the instruction. The contents of A are not altered.
Clear Field F3a = 6	CLF	Clear the specified field of the storage location specified by the effective address to all 0s. All other storage bits are unchanged. Memory is locked until completion of the instruction.
Set Field F3a = 7	SEF	Set the specified field of the storage location identified in the effective address to all 1s. All other storage bits are unchanged. Memory is locked until completion of the instruction.

ENHANCED INTER-REGISTER

These instructions have the following format:



Enhanced inter-register instructions are identified when the F field is 0, the F1 field is 7, and the F2a, Ra, and Rb fields are not all 0. (If these fields are all 0, the instruction is CPB.)

Enhanced inter-register instructions (similar to the basic) inter-register instructions, such as TRA Q) contain three parts: operation field (F2a) and two register fields (Ra and Rb). The F2a field determines the operation (e.g., transfer). The Ra and Rb fields specify two operands.

Note that F2a = 1, F2a = 2, F2a = 3, Ra = 0, and Rb = 0 are reserved for future expansion.

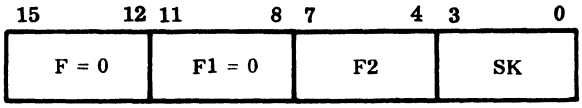
The instruction description is:

Transfer Register XFr R
F2a = 0
Ra = 1, 2, 3, 4, 5, 6, or 7
r = 1, 2, 3, 4, Q, A, or 1

Transfer the contents of register r to register R. Note that R = 1, 2, 3, 4, Q, A, or I implies that Rb = 1, 2, 3, 4, 5, 6, or 7.

ENHANCED SKIP

The skip instructions have the following format:



Enhanced skip instructions are identified when the F and F1 fields are both 0, and the F2 and SK fields are not both 0. (If these fields are both 0, the instruction is an SLS.)

Enhanced skip instructions (similar to the basic skips; such as SAZ) contain two parts: operation field (F2) and skip count (SK). The F2 field determines the operation (i.e., skip on register 1, 2, 3, or 4 if zero, nonzero, positive, or negative). The skip count specifies how many locations to skip if the skip condition is met.

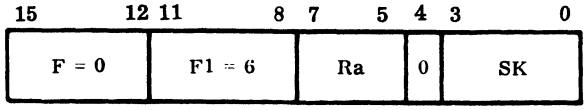
When the skip condition is met, the skip count plus one is added to the P register to obtain the address of the next instruction (e.g., when the skip count is one, go to P + 2). When the skip condition is not met, the address of the next instruction is P + 1 (skip count ignored). The skip count does not have a sign bit.

If F2 = 0 (S4Z), the skip count cannot be 0 because the instruction would be an SLS.

The instruction descriptions are given in table 4-11.

DECREMENT AND REPEAT

These instructions have the following format:



Decrement and repeat instructions are specified when the F field is 0, the F1 field is 6, bit 4 is 0, and the Ra and SK fields are not both 0. (If these fields are both 0, the instruction is a SPB.)

Decrement and repeat instructions contain two parts: register field (ra) and skip count (SK). The register field specifies which register is to be decremented by one and checked for the skip condition. The skip count specifies how many locations to repeat (go backwards) if the skip condition is met.

When the skip condition is met, the skip count is subtracted from the P register to obtain the address of the next instruction (e.g., when the skip count is one, go to P - 1). When the skip condition is not met, the address of the next instruction is P + 1. The skip count does not have a sign bit.

Note that Ra = 0 and bit 4 = 1 are reserved for future expansion.

TABLE 4-11. ENHANCED SKIP INSTRUCTIONS

Instruction	Mnemonic	Description
Skip if Register Zero F2 = 0, 4, 8, or C r = 4, 1, 2, or 3	SrZ SK	Skip if register r is a positive 0 (all bits are 0).
Skip if Register Nonzero F2 = 1, 5, 9, or D r = 4, 1, 2, or 3	SrN SK	Skip if register r is not a positive 0 (not all bits are 0).
Skip if Register Positive F2 = 2, 6, A, or E r = 4, 1, 2, or 3	SrP SK	Skip if register r is positive (bit 15 is 0).
Skip if Register Negative F2 = 3, 7, B, or F r = 4, 1, 2, or 3	SrM SK	Skip if register r is negative (bit 15 is a 1).

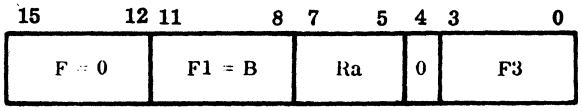
The instruction description is:

Decrement and Repeat if Positive DrP SK
Ra = 1, 2, 3, 4, 5, 6, or 7
r = 1, 2, 3, 4, Q, A, or I

Decrement the contents of register r by one. Operation on overflow is the same as for the ADD instruction. Repeat (go backwards) SK locations if the contents of register r are positive (bit 15 is 0), otherwise execute the next instruction.

MISCELLANEOUS

Miscellaneous instructions have the following format:



Miscellaneous instructions are specified when the F field is 0, the F1 field is equal to a decimal 11 (hexadecimal B), bit 4 is 0, and Ra and F3 fields are not both 0. (If these fields are both 0, the instruction is an NOP.) All of the miscellaneous instructions are privileged instructions; i.e., if they are executed by an unprotected program, they will cause a program protect violation.

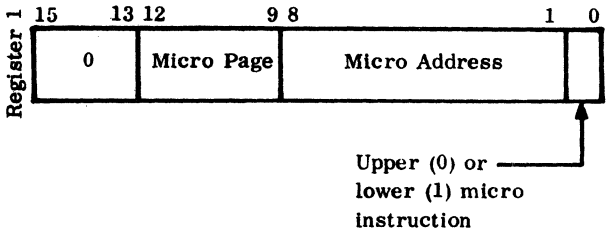
Miscellaneous instructions contain two parts: operation field (F3) and register field (Ra).

If Ra is nonzero, the F3 operation field can select up to 16 miscellaneous instructions with register Ra used to specify an operand. If Ra is 0, the F3 operation field can select up to 15 more miscellaneous instructions without any explicit operand specified.

All the miscellaneous instruction descriptions are given in table 4-12. Those instructions that require more detail are described below.

The miscellaneous instruction formats are:

1. Load Micro Memory



Initially, the Q register contains the number of 32-bit micro-memory instructions to be transferred (if Q = 0, no instructions will be transferred). Register 1 contains the starting address of micro memory. Register 2 contains the starting address of MP macro memory.

The most significant bit (15) of the contents of the starting address will be transferred to the most significant bit of the first micro instruction. The least significant bit (0) of the contents of the starting address plus one will be transferred to the least significant bit. This instruction is interruptible after storing each 32-bit micro memory instruction, and when registers 1, 2, and Q are incremented/decremented to allow the instruction to be restarted after any interruption. When the instruction is completed, these registers will contain the following rather than their original values:

Q ← 0
R1 ← (R1)i+(Q)i
R2 ← (R2)i+2*(Q)i

Where: i is the initial value before execution.

2. Set/Sample Output or Input

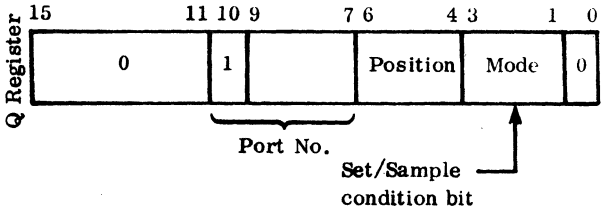


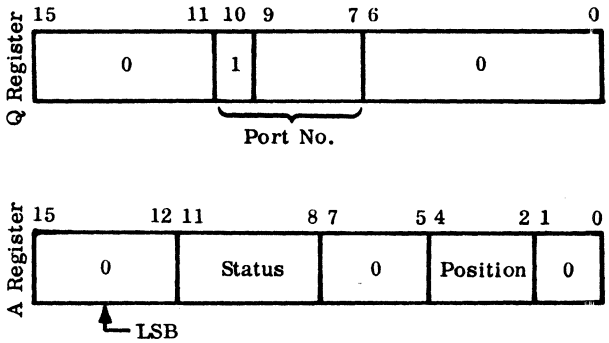
TABLE 4-12. MISCELLANEOUS ENHANCED INSTRUCTIONS

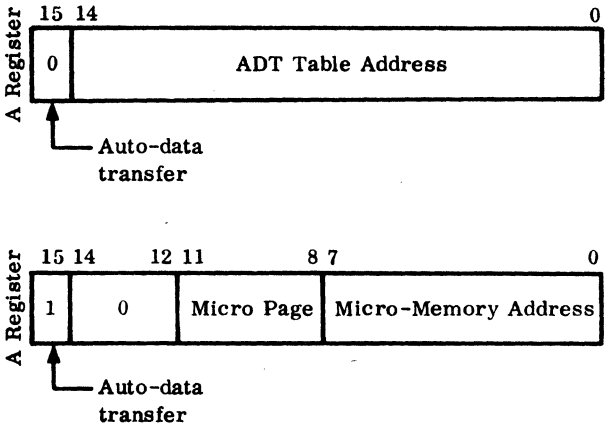
Instruction	Mnemonic	Description
Load Micro Memory F3 = 1 Ra = 0	LMM	Load a 32-bit micro-memory instruction into read/write micro memory from 16-bit MP macro memory. (For read-only micro memory or no micro memory, no operation is executed.)
Load Registers F3 = 2 Ra = 0	LRG	Registers 1, 2, 3, 4, Q, A, I, M, and the overflow indicator are loaded with the contents of nine storage locations, beginning at a storage location specified by the contents of the contents of the next location, P + 1. The contents of the nine storage locations will not be altered and the next instruction will be executed at location P + 2 (i.e., the LRG instruction is a two-word instruction). Refer to figure 4-1.
Store Registers F3 = 3 Ra = 0	SRG	Registers 1, 2, 3, 4, Q, A, I, M, and the overflow indicator are stored into nine storage locations specified by the contents of the next location, P + 1, incremented by a decimal 10. The contents of the registers will not be altered and the next instruction will be executed at location P + 2 (i.e., the SRG instruction is a two-word instruction). Refer to figure 4-2.
Set/Sample Output or Input F3 = 4 Ra = 0	SIO	Set one word from register A for output to an external device. The word in register Q selects the receiving device. For input, one word from an external device is sampled (input) to register A. The word in register Q selects the sending device.
Sample Position/Status F3 = 5 Ra = 0	SPS	Sample (input) to the A register the position and status of a M05 device, which has caused an MP macro interrupt. The word in the Q register selects the device. This instruction also provides for clearing the M05-generated MP macro interrupt.
Define Micro Interrupt F3 = 6 Ra = 0	DMI	Define the use of one of the 12 available micro interrupts. (The use of micro interrupts 12 through 15 is restricted for internal use.) This instruction allows a micro interrupt to be enabled/disabled and defined for auto-data transfer (ADT) or special usage.
Clear Breakpoint Interrupt F3 = 7 Ra = 0	CBP	Clear the MP macro breakpoint interrupt. This interrupt occurs when the following conditions are true: macro breakpoint is externally selected, macro breakpoint interrupt option is externally selected, the MP recognizes a breakpoint condition and generates an MP macro breakpoint interrupt because of b.
Generate Character Parity Even F3 = 8 Ra = 0	GPE	Set or clear bit 7 of the A register so that bits 0 to 7 have an even parity. The other bits in the A register are not altered.
Generate Character Parity Odd F3 = 9 Ra = 0	GPO	Set or clear bit 7 of the A register so that bits 0 to 7 have an odd parity. The other bits in the A register are not altered.

TABLE 4-12. MISCELLANEOUS ENHANCED INSTRUCTIONS (Continued)

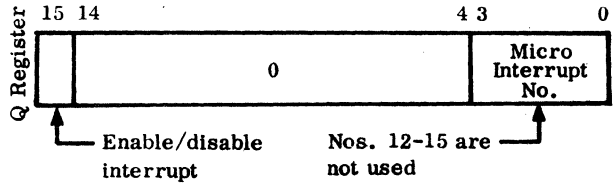
Instruction	Mnemonic	Description
Scale Accumulator F3 = A Ra = 0	ASC	Shift the A register left (end-around) until bits 14 and 15 of the A register are different. Upon completion of the instruction, register 1 will contain the number of spaces that the A register was shifted. (This number may range from 0 to 14.) If the A register is ±0 (0000 or FFFF$_{16}$), no shift has been done and register 1 will contain -0 (FFFF$_{16}$).
Load Upper Unprotected Bounds F3 = 0 Ra = 1, 2, 3, 4, 5, 6, or 7 R = 1, 2, 3, 4, Q, A, or I	LUB R	Load the upper unprotected bounds register from the contents of register R.
Load Lower Unprotected Bounds F3 = 1 Ra = 1, 2, 3, 4, 5, 6, or 7 R = 1, 2, 3, 4, Q, A, or I	LLB R	Load the lower unprotected bounds register from the contents of register R.
Execute Micro Sequence F3 = 2 Ra = 1, 2, 3, 4, 5, 6, or 7 R = 1, 2, 3, 4, Q, A, or I	EMS R	Transfer machine control to the upper micro instruction of the page/micro-memory address in bits 0 to 15 of register R. A section of micro memory is assumed to have been previously loaded.

3. Sample Position Status





4. Define Micro Interrupt



When bit 15 of the A register is set to a 1, a jump is made to the upper micro instruction of the page/micro memory in bits 0 to 14. A section of micro memory is assumed to have been previously loaded, and it must process the micro interrupt properly and return control to the current macro instruction address (P) by jumping to the lower micro instruction of micro-memory address 3E$_{16}$ in micro page zero. Registers P, A, Q

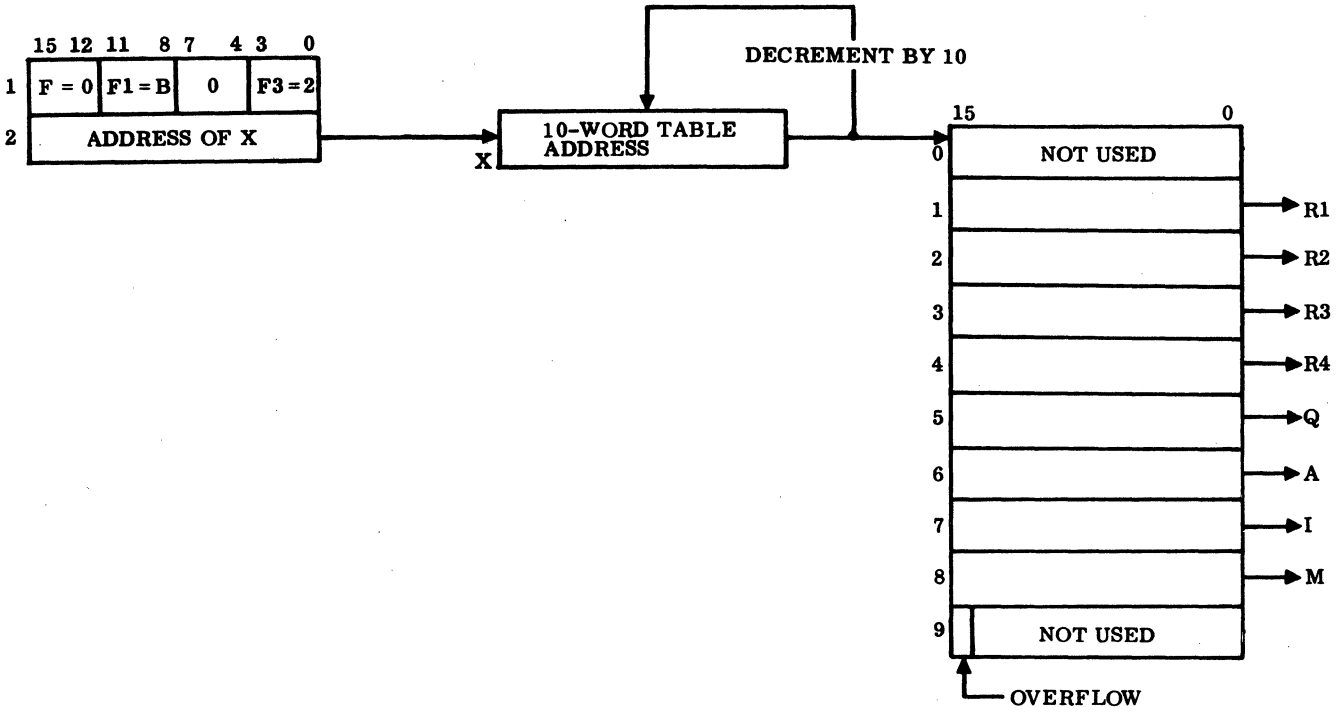


Figure 4-1. LRG Instruction

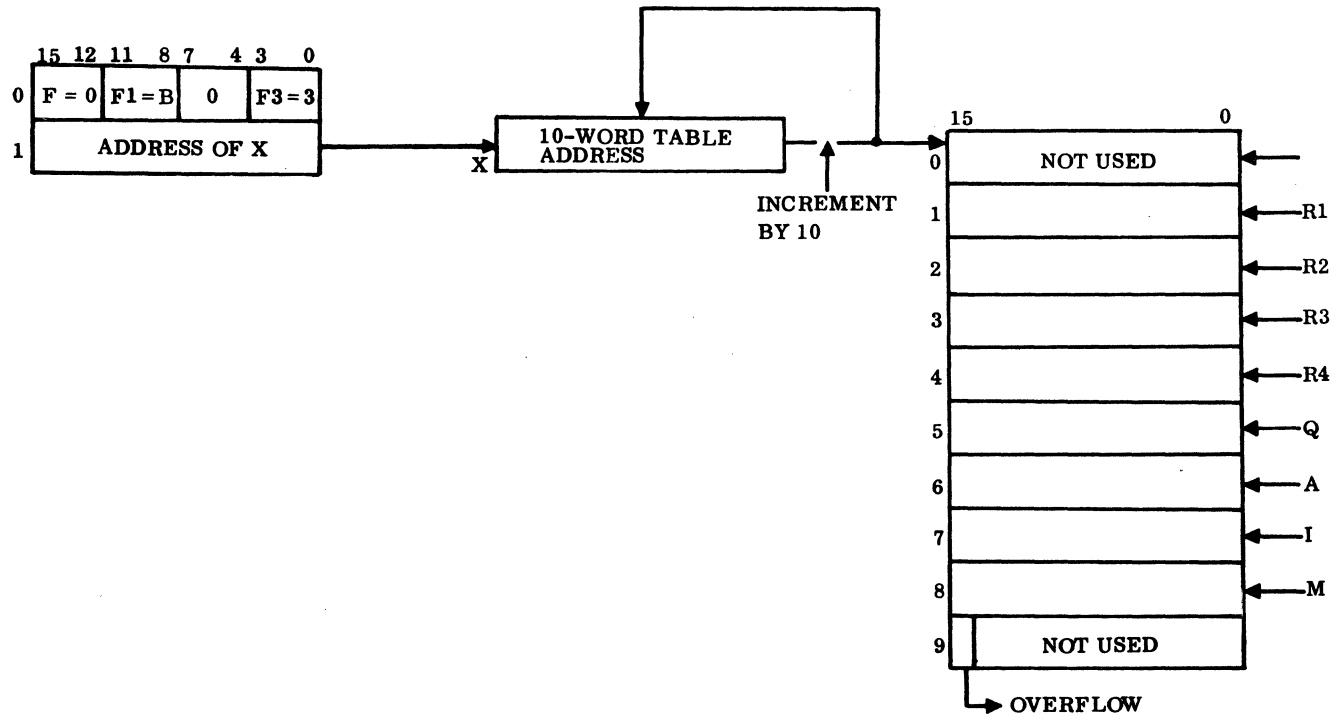


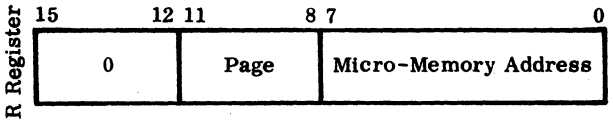
Figure 4-2. SRG Instruction

and all of file 2 should not be altered, and return must be within 12.5 microseconds.

CAUTION

The MP micro function, SUB-, must not be used. Extreme caution should be exercised in using this option, since it provides an escape from the 1700 emulation being performed.

5. Execute Micro Sequence



An EMS to non-existant micro memory may cause an indeterminate result. Control should be returned to the next macro-instruction address (P + 1) by jumping to the lower micro instruction of micro-memory address $3E_{16}$ in micro page zero. Registers P, A, Q, and all of file 2 should not be altered, and return must be within 12.5 microseconds (or the micro sequence must be interruptible).

CAUTION

Extreme caution should be exercised in using this option, since it provides an escape from the 1700 emulation being performed.

AUTO-DATA TRANSFER

Auto-data transfer (ADT) provides for pseudo direct memory transfers of data blocks to or from a device. At the macro level, the transfer appears as a direct memory access (DMA) transfer. At the micro level, the 1700 emulator processes each data interrupt and inputs or outputs the next data in a singular fashion. Thus, ADT takes less time than input/output via the INP, OUT, or SIO instructions, but more time than a true DMA transfer.

To accomplish an ADT for a particular device, perform the following:

1. The device and its controller must adhere to the auto-data transfer specifications in the Micro-programmable Computer I/O Specification.

2. The macro programmer must execute a DMI instruction. This command specifies where the block of data is, how long it is, the direction (input/output), and the device's address.

3. The ADT operation is then initiated by an INP, OUT, or SIO instruction as specified by the particular device.

While the ADT operation is in progress, the emulator is executing instructions. After each instruction is executed, interrupts are checked. When the particular ADT micro interrupt becomes the highest active interrupt, the next data is input or output. After the interruption, the next instruction is executed, except when another interrupt is active.

When the ADT operation is completed (or if there is an error), a macro interrupt is generated. The macro programmer may then disable the ADT micro interrupt or initiate another ADT operation to or from the device. For MOS devices, an SPS instruction must be performed to clear the macro interrupt.

The following are the four types of ADT tables specified by DMI instructions.

1. ADT Table for a Single A/Q Device:

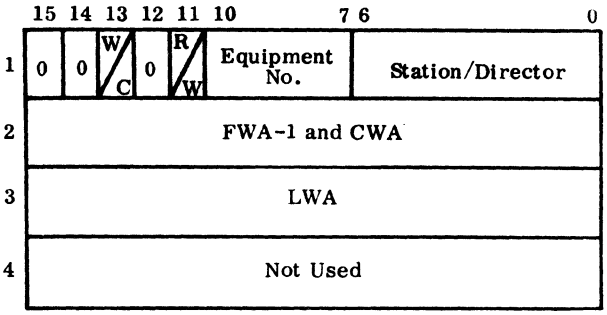
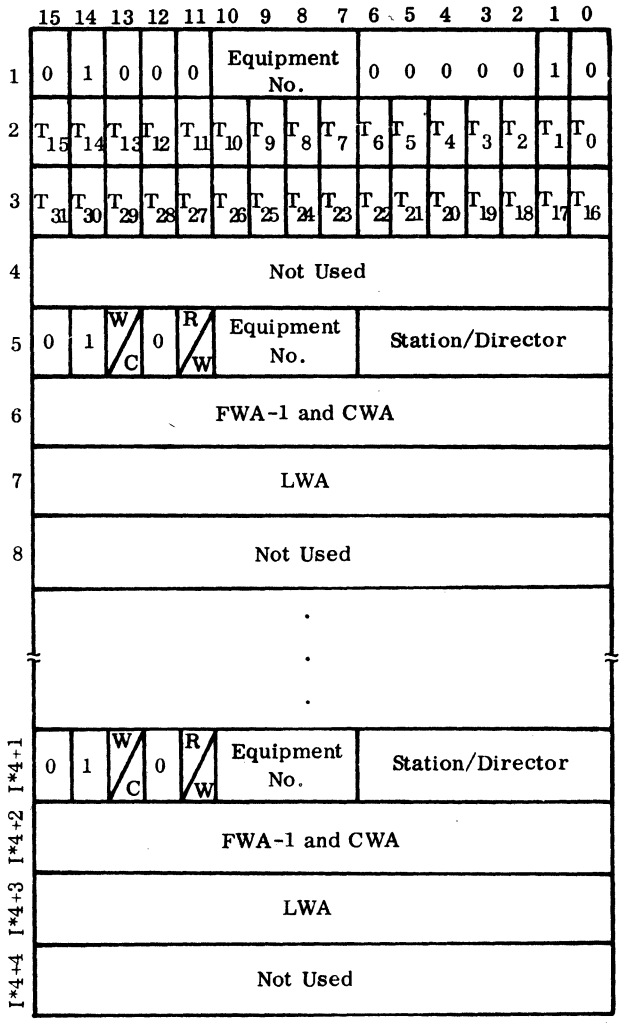


Table 4-13 gives a detailed description of these four words.

TABLE 4-13. ADT TABLE FOR A SINGLE A/Q DEVICE

Word	Bits	Description
1	15 14 12	Must be set to 0.
	13	0 Word operation; data is transferred one word at a time. Normally, a total of (CWA - FWA + 1) words will be transferred. 1 Character operation; data is transferred one character (eight bits) at a time. On input, the first character will be stored in the most significant half (bits 15 to 8) of the current words address; the second character in the least significant half (bits 7 to 0). Subsequent pairs of characters will be output from the most significant half of the current word address; the second character from the least significant half. Normally a total of 2 × (CWA - FWA + 1) characters will be transferred.
	11	0 A read ADT operation 1 A write ADT operation
	10 through 7	The equipment number of the device. This number can not conflict with any MO5 I/O port numbers.
	6 through 0	The station/director bits of the device to execute the ADT operation. These bits should specify a data (not a status/function) transfer.
2		Initially set to the first word address less one (FWA - 1) of the data block to be transferred. This word is used as the current word address (CWA) as the ADT operation is in progress and points to the last word read or stored. Each time a word (or two characters) is transferred, CWA is incremented. Specifically, CWA can be used to ascertain whether all the data was transferred after the ADT operation was completed (if CWA = LWA, all the data has been transferred).
3		The last word address (LWA) of the data block to be transferred
4		Reserved for future use; must be set to 0.

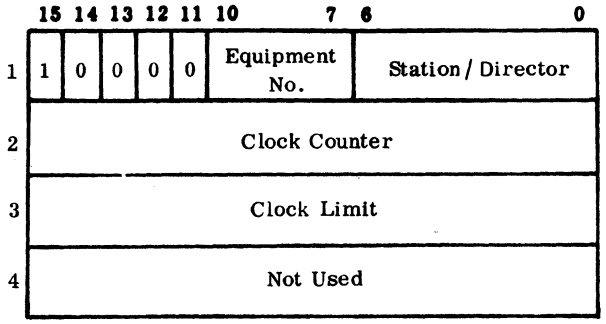
2. ADT Table for Multiple A/Q Devices:



This type of ADT table consists of I*4+4 words, where I is the number of multiple A/Q devices (up to 32) on one micro interrupt.

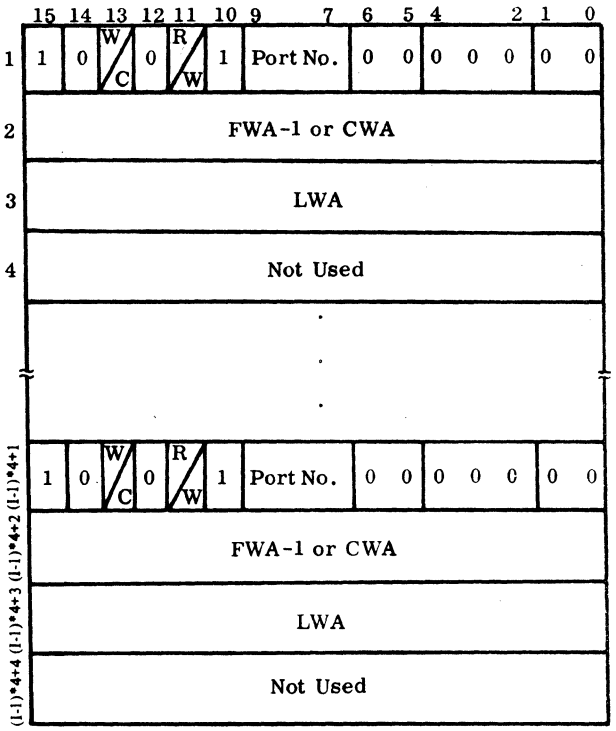
Table 4-14 provides detailed descriptions of these words.

3. ADT Table for the Clock.



Detailed descriptions of this type are given in table 4-15.

4. ADT Table for Single or Multiple M05 Devices



The ADT table for this type consists of (I-1)*4+4 words, where I is the number of M05 devices (up to 8) on one micro interrupt. Detailed descriptions of this type are given in table 4-16.

TABLE 4-14. ADT TABLE FOR MULTIPLE A/Q DEVICES

Word	Bits	Description
1	15	Must be 0
	14	Must be 1
	13 through 11	Must be 0
	10 through 7	The equipment number of the device. This number can not conflict with any M05 I/O port numbers.
	6 through 2	The maximum station (or channel) number; equivalent to the number of multiple A/Q devices less one on a wire interrupt. Station numbers must be contiguous. Certain peripheral devices have specific parameters for these bits; refer to the peripheral controller reference manual.
	1	Must be 1
	0	Must be 0
2 3		Contain termination bits for the 32 devices. Initially, they must be all 0. When a macro interrupt occurs, one or more of these bits will be set to 1 to indicate that one or more ADT operations have terminated. Thus $T_7 = 1$ indicates that the seventh device has terminated its ADT operation. After receipt, the bit should be cleared via an instruction that locks memory (e.g., a CLF instruction).
4		Reserved for future use; must be 0.
5 6 7 8 †		Defined the same as a single A/Q device, except for bit 14 of the first word (I*4+1), which must be 1. Refer to table 4-13.

†Words I*4+1, I*4+2, I*4+3, and I*4+4, where $2 \leq I \leq 32$

TABLE 4-15. ADT TABLE FOR THE CLOCK

Word	Bits	Description
1	15	Must be 1
	14 through 11	Must be 0
	10 through 7	The equipment of the clock; must be set to 1.
	6 through 0	The station/director bits of the clock, which is always equal to 70_{16}. (Thus, word 1 should equal $80F0_{16}$.)
2		Initially set to 0; whenever the clock has been enabled, the clock counter will be incremented every 3 1/3 milliseconds.
3		The clock limit, which is interpreted as a multiple of 3 1/3 milliseconds. When the clock counter equals the clock limit and the macro-clock interrupt is enabled, the macro-clock interrupt will occur. Thus, if the clock limit is five, the clock interrupt is 16 2/3 milliseconds, or 60 times a second. To continue the process, the clock counter should be reset to 0, or the limit counter should be incremented by its original value (e.g., 5). In the latter method, the clock counter can function as an elapsed time counter. Note that if the macro-clock interrupt is not answered the clock limit will still continue to be incremented.
4		Reserved for future use; must be 0.

TABLE 4-16. ADT TABLE FOR SINGLE OR MULTIPLE M05 DEVICES

Word	Bits	Description
1	15	Must be 1
	14	Must be 0
	13	Defined the same as a single A/Q device.
	12	Must be 0
	11	0 A read ADT operation 1 A write ADT operation
	10 through 7	The part number of the device. (Bit 10 is always set to 1.) Part numbers are analogous to the A/Q I/O equipment numbers and thus cannot conflict with them.
	6, 5	Must be 0
	4 through 2	The maximum position number; equivalent to the number of multiple M05 devices, less one, on a wire interrupt. Position numbers must be contiguous (i.e., 0 to I-1).
2†		Initially set to the first word address (FWA-1) of the data block to be transferred; this word is used as the current word address (CWA) as the ADT operation is in progress and points to the last data word read or stored. Each time a word (or two characters) is transferred, CWA is incremented. Specifically, CWA can be used to ascertain whether all the data was transferred after the ADT operation was completed (i.e., if CWA = LWA, all data has been transferred).
3†		The last word address (LWA) of the data block to be transferred
4†		Reserved for future use; must be 0.

† Words (I-1)*4+1, (I-1)*4+2, (I-1)*4+3, and (I-1)*4+4, where $2 \leq I \leq 8$, are defined in the same manner as words 1, 2, 3, and 4, respectively.

This system enables the program to establish an interrupt priority so that an interrupt of high priority can interrupt the machine while it is processing an interrupt of a lower priority. The return path to the interrupted program is clearly established and saved.

INTERRUPT TRAP LOCATIONS

Trap locations are established for each interrupt line. They are in the range of addresses 0100 through 013C. These addresses are reserved for interrupts unless that particular interrupt is not being used. The assignment for each interrupt state or line is shown in table 5-1.

MASK REGISTER

The mask register is the enable for each interrupt state or line. Bit 0 of the mask register corresponds to the interrupt line 0, bit 1 to line 1, etc. To enable an interrupt line, its corresponding bit in the mask register must be set. The mask register is set by the inter-register instruction.

PRIORITY

The computer program controls the interrupt priority by establishing an interrupt mask for each interrupt state, which enables all higher priority interrupts and

TABLE 5-1. INTERRUPT STATE DEFINITIONS

Interrupt State	Value of Δ to Exit State	Location of Return Address	Location of First Instruction after Interrupt Occurs
00	00	0100	0101
01	04	0104	0105
02	08	0108	0109
03	0C	010C	010D
04	10	0110	0111
05	14	0114	0115
06	18	0118	0119
07	1C	011C	011D
08	20	0120	0121
09	24	0124	0125
10	28	0128	0129
11	2C	012C	012D
12	30	0130	0131
13	34	0134	0135
14	38	0138	0139
15	3C	013C	013D

disables all lower priority interrupts. When an interrupt state is entered, the mask for that state is placed in the mask register. Therefore, there may be up to 16 levels of priority. It is possible to change priority during execution of a program.

INTERNAL INTERRUPTS

Interrupts are also generated by certain conditions arising within the computer. These are called internal interrupts. If such a condition occurs, it generates interrupt 00 (interrupt mask bit 00). Normally, internal interrupts are assigned the highest priority. The internal interrupts are:

- Storage Parity Error
- Program Protect Fault
- Power Failure

OPERATION

The computer can distinguish between up to 16 (1 internal, 15 external) macro interrupts. Each of these interrupts has its respective address to which control is transferred when the interrupt is recognized.

When the computer is processing a particular interrupt, it will be defined as being in that interrupt state (state 00 through 15). Thus, the interrupts and their respective bits in the interrupt mask register are numbered 00 through 15. An interrupt in bit 7 will put the computer in interrupt state 7, etc.

Before the computer can recognize any interrupt, the mask bit for that interrupt must be set and the interrupt system must be activated. The mask register may be set by an inter-register command and the interrupt system can be activated by an enable interrupt command.

When an interrupt is recognized, the computer automatically stores the return address in the storage location reserved for that interrupt state. If 32K multilevel indirect mode has been selected, bit 15 of the storage location is set or cleared to record the current state of the overflow indicator. If 65K multilevel indirect mode has been selected, all 16 bits are required to save the return address. Thus, the program must check for an overflow condition with an SOV or SNO instruction and record this condition for restoration of the overflow indicator. In both 32K and 65K modes the interrupt system is de-activated and control is transferred when the interrupt occurs. In 32K mode the overflow is cleared, while in 65K mode the SOV or SNO instruction must first be executed. The program then stores all registers, including the mask register, in addresses reserved for this interrupt state and loads the mask register with the mask to be used in this state. The 1s in the mask indicate the interrupts that have a higher priority than the interrupt being processed. The mask should not have a 1 in the position of the interrupt being processed; this would lose the return link. The program then activates the interrupt system and processes the interrupt.

The computer exits from an interrupt state when the program inhibits the interrupt and restores the registers (including the mask register). After loading the register, the program executes the exit interrupt command with delta equal to the lower eight bits of the base address of the interrupt state. This command reads the storage location where the return address is stored. The overflow indicator is set or cleared as specified by bit 16. The interrupt system is activated and control is transferred to the return address.

EXAMPLE

The following listing and sample program steps apply if there were five different possible interrupts and three levels of priority:

Interrupt 01		High priority
02	}	Mid-priority
05	}	Mid-priority
03	}	Low priority
04	}	Low priority

Bit	5 4 3 2 1 0	
Mask 1	1 1 1 1 1 1	Mask used for main program
Mask 2	1 0 0 1 1 1	Mask used for state 03, 04
Mask 3	0 0 0 0 1 1	Mask used for state 02, 05
Mask 4	0 0 0 0 0 1	Mask used for state 01

Main Program	State 02 Program
Set mask register to Mask 1	Store registers
Enable interrupt	Set mask to Mask 3
.	Enable interrupt
.	.
.	.
	.

```
                                  Inhibit interrupt
                                  Replace registers
                                  Exit interrupt 02

State 01 Program                  State 03 Program

Store registers                   Store registers
Set mask to Mask 4                Set mask to Mask 2
Enable interrupt                  Enable interrupt
      .                                 .
      .                                 .
      .                                 .
Inhibit interrupt                 Inhibit interrupt
Exit interrupt 01                 Replace registers
                                  Exit interrupt 03

State 04 Program                  State 05 Program

Store registers                   Store registers
Set mask to Mask 2                Set mask to Mask 3
Enable interrupt                  Enable interrupt
      .                                 .
      .                                 .
      .                                 .
Inhibit interrupt                 Inhibit interrupt
Replace registers                 Replace registers
Exit interrupt 04                 Exit interrupt 05
```

The MP computer has a program protect system to protect a program in the computer from any other nonprotected program also in the computer. The system is built around a program protect bit contained in each word of storage. If the bit is set, the word is an operand or an instruction of the protected program. All operand and instruction locations of the protected program must have the program protect bit set. None of the instructions or operands of the nonprotected program can have the program protect bit set.

Whenever a violation of the program protect system, other than a direct storage access violation, is detected, the program protect fault flip-flop is set and an internal interrupt is generated. A violation indicates that the nonprotected program has attempted an operation that could harm the protected program.

PROGRAM PROTECT VIOLATIONS

The following are the program protect violations:

- A nonprotected instruction attempts to write in a protected storage location. The contents of the storage location are not changed.
- An attempt is made to write into a protected storage location via external storage access when a nonprotected instruction was the ultimate source of the attempt. The contents of the storage location are not changed.
- An attempt is made to execute a protected instruction following execution of a nonprotected instruction. The protected instruction is executed as a nonprotected selected stop instruction. However, it is not a violation if an interrupt caused this sequence of instructions.
- An attempt is made to execute the following instructions when they are not protected: any interregister instructions with bit 0 = 1, EIN, IIN, EXI, SPB, CPB, or any miscellaneous instructions (0Bxx). Those instructions become a nonprotected selective stop instruction under these circumstances.

Program protect is enabled by setting bit 8 in the function control register. If this bit is not set, then none of the above violations are recognized, with the exception of the external storage access protect violation.

STORAGE PARITY ERRORS AS RELATED TO PROGRAM PROTECTION

If a nonprotected instruction is attempting to write into storage and a storage parity error is present or occurs, the word in storage is not altered and a Storage Parity Error interrupt is enabled.

If a protected instruction is attempting to write into storage and a storage parity error occurs, the word is written into storage and a Storage Parity Error interrupt is enabled.

If the computer attempts to execute a SPB or CPB instruction and a storage parity error occurs, these become Pass instructions and a storage parity error interrupt is enabled.

SET/CLEAR PROGRAM PROTECT BIT

The program protect instructions (SPB or CPB) and the bounds instructions (LUB and LLB) are the only way in which the program protect bit may be set or cleared in each word of storage.

PROGRAMMING REQUIREMENTS

The following program requirements must be met:

- The program package that handles all interrupts for the nonprotected program must be completely checked out. This program must also be part of the protected program.
- The protected program must be a completely checked-out program.

PERIPHERAL EQUIPMENT PROTECTION

All peripheral equipment that is essential to the operation of the protected program must have a bit in the FCR to designate if the device is protected. If the bit is set, the peripheral device responds with a reject to all nonprotected commands (except status request) addressed to it. All protected commands have a normal response. If the bit is not set, the peripheral device responds in the normal manner to protected and nonprotected commands.

The standard assignments for MP I/O devices are listed in table 7-1. Descriptions of the panel/program device and clock follow.

PANEL/PROGRAM DEVICE

When referencing the panel/program devices, the Q register should contain either 0090_{16} or 0091_{16} according to the following table:

	Computer Instruction	
Q Register	Output from A	Input to A
0090	Write	Read
0091	Director Function (1)	Director Status (2)

DIRECTOR FUNCTION (1):

Bit (A Register)	Function	Operation
00	Clear Controller	Clear all interrupt requests. Clear busy, interrupt, data, alarm, and manual interrupt conditions. Select read mode. Connect printer. Any interrupt request bit will take precedence over this function. Clear Controller is also used in conjunction with bits 11, 12, 14, and 15.
01	Clear Interrupt	Clear all interrupt requests and the manual interrupt. Any interrupt request bit will take precedence over this function.
02	Data Interrupt Request	Send an interrupt signal whenever a data status is active.
03	End-of-Operation Interrupt Request	Send an interrupt signal when the controller is not busy. In the EOP state the controller will accept a mode change.
04	Alarm Interrupt Request	Send an interrupt signal when the Lost Data Status is active.
05	Not used	
06	ADT Mode	Auto-data transfer operation
07	Not used	
08	Select Write Mode	An output operation; does not clear the alarm status.
09	Select Read Mode	An input operation
10	Connect Printer	Select a mode of operation in which the printer (with the paper tape punch, when used) and the tape reader (when used) are both connected to the controller. Data read from the paper tape in this mode will also be printed (and punched).
11	Not used	
12	Not used	
13	Disconnect Printer	Select a mode of operation in which the printer (and paper tape punch when used) is disconnected from the controller. Data read from paper tape is not printed (or punched). This mode allows

TABLE 7-1. STANDARD MP ASSIGNMENT OF EQUIPMENT CODES, MACRO/MICRO INTERRUPT LINES

Device Type	Equipment Code, Macro/Micro Interrupt Line
Program Protect,† Memory Parity, Power Failure	0
Low-Speed I/O (CRT/TTY)	1
Drums	2
Disks (Storage Module Disk)	3
Line Printers (1742-30)	4
Communication Equipment	5
Communication Equipment	6
Magnetic Tape (Cassette)	7
Real-Time Clock (equipment is same as CRT/TTY, -1)	8
	9
	10
Card Readers (1729-3)	11
Firmware Panel Input††	12
Firmware Panel Output††	13
Hardware Panel Request††	14
Macro Instruction Step††	15

†Macro-interrupt line only
††Micro-interrupt line only

Bit (A Register)	Function	Operation
		non-ASCII codes and binary information to be transmitted to the computer.
14	Not used	
15	Not used	

All nonconflicting functions may be performed simultaneously. Select write mode and select read mode are rejected when the controller is busy. Other functions are always performed. When several functions are issued simultaneously and some of them can be performed, the output from the A instruction exists normally (Reply), but those functions that should be rejected are not performed. When none of the functions can be performed, the output from the A instruction is rejected.

DIRECTOR STATUS (2):

Bit (A Register)	Status	Description
00	Ready	Unit is ready.
01	Busy	Read mode — The controller is in the process of receiving a character from the TTY/CDT, or the holding register contains data for transfer to the

Bit (A Register)	Status	Description
		computer. The busy status will drop upon completion of the data transfer. Write mode — The data register contains data and is in the process of transferring it to the TTY/CDT. The busy status will drop when the transfer is completed.
02	Interrupt	An interrupt condition exists in the controller.
03	Data	Read mode — The holding register contains data for transfer to the computer. The data status will drop when the transfer is completed. Write mode — The controller is ready to accept another character from the computer.
04		Always the inverse of busy (bit 01)
05	Alarm	Parity error or lost data or field error (no stop bit when expected) occurred.
06	Lost Data	The holding register contained data for transfer to the computer, and the TTY/CDT began to send a new sequence.
07	Parity Error	A parity error occurred
08	Release	Release reserve interrupt; this interrupt is generated when the CRT or TTY has been reserved for the panel interface and is returned.
09	Read Mode	The controller is conditioned for input operation.
10	Reserved	Indicates if the TTY or CRT is currently assigned to the panel interface and is unavailable to the TTY controller.
11	Manual Interrupt	A manual interrupt has occurred.
12	Not used	Always 0
13	Not used	Always 0
14	Not used	Always 0
15	Not used	Always 0

REAL-TIME CLOCK

The real-time clock is an integral part of the I/O module and is designed to appear as a 1700 peripheral to the macro-level software. Two functions are available to the macro-level program: Enable/Disable Limit Interrupt and Enable/Disable Clock. Also available to the macro-level program are two status bits: Limit Interrupt and Lost Count.

The Enable Clock and Limit Interrupt functions are selected by performing a write to the real-time clock (W = 0, E = 1, and S = 7) and setting the two least significant bits of the Q register equal to 1 ($Q = 00F3_{16}$). The enable Clock and Limit Interrupt functions are disabled in the same manner with the two least significant bits of the Q register equal to 0 ($Q = 00F0_{16}$). Either case will clear an existing limit interrupt and clear the status of the real-time clock.

The real-time clock status is obtained by an input from the real-time clock (W = 0, E = 1, and S = 7). Status is returned in the A register with bit 15 (when true) indicating a lost count, and bit 14 (when true) indicating a limit interrupt. The rest of the bits in the A register are undefined.

The limit interrupt that is received by a macro-level program is dependent on the appropriate M register bit (M08, 1700 convention) being set and the macro interrupt enabled, as with all other 1700 peripherals.

The emulator is capable of receiving a micro interrupt from the real-time clock every 3 1/3 milliseconds

(based on a crystal oscillator). This interrupt is enabled anytime the DMI instruction has defined the micro interrupt (INT08) and the clock has been enabled as indicated above.

The value stored for the clock limit in the ADT table for the clock detemines when the emulator generates the macro-level interrupt. If the emulator becomes overloaded with higher priority micro interrupts and the micro interrupt for the clock has not been cleared before another 3 1/3 millisecond count occurs (and the limit interrupt has been selected), then the lost count status bit will be set. This will cause a Limit Interrupt to occur with the lost count status bit set.

The real-time clock is always ready, and reads or writes are never rejected.

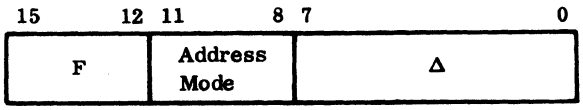
A FIELD	In a micro instruction the A field specifies the operand source to be sent to the ALU from selector 1.
A REGISTER	General-purpose register
AB	Address buffer register; main memory address register.
ALU	Arithmetic/logical unit; performs arithmetic and logical operations on two operands received from the two selectors.
AUTOLOAD	Process whereby macro memory is loaded from an external input device via the memory DMA port.
B FIELD	In a micro instruction the B field specifies the operand source to be sent to the ALU from selector 2.
BG	Bit generator; allows a word containing all 0s except one bit at any position; used for masking or arithmetic operations.
C FIELD	Constant field micro-instruction field; may contain constants, micro-memory addresses, or other codes, depending on format of micro instruction.
CPU	Central processing unit; consists of micro memory, control section, arithmetic section, and I/O.
D REGISTER	Data register on MP I/O card
D FIELD	Destination field/micro-instruction field; specifies the destination for results of the operation performed by ALU.
DEADSTART	Optional logic that allows read/write micro memory to be loaded from external input device.
EMULATION	Process combining hardware and firmware design in which one processor (emulator) executes programs designed for a different processor, even though one-to-one hardware correspondence does not exist.
1700 ENHANCED PROCESSOR	A 16-bit MP processing element operating as an enhanced CDC 1700 computer.
F FIELD	Function field micro-instruction field; specifies the operation to be performed by ALU, shift, or scale of A or A/Q registers.
F REGISTER	General purpose register
FILE 1	Optional register file addressed by the contents of the K register.
FILE 2	Register file typically addressed by contents of N register.
FIRMWARE	General term for combination of micro instructions used in micro program to perform a certain operation.
I REGISTER	Storage location $00FF_{16}$ used for 1700 instruction indexing. Also a general-purpose MP register used in 1700 simulation.
IXT	I register external on the transform
I/O	Input/output
K REGISTER	Eight-bit counter that can be cleared, incremented, or decremented under micro-instruction control. Also used to address file 1.

M FIELD	Mode field micro-instruction field; specifies the addressing mode to be used to obtain the next micro-instruction pair from micro memory.
MA REGISTER	Micro-memory address register; holds the micro-memory address of the current micro-instruction pair.
MA TRANSFORM	Micro-memory address transform
MAC	Memory address counter; holds the address of the next sequential micro-instruction pair.
MACRO MEMORY	Core memory used by the MP for storage of operands, etc.
MASK REGISTER	Used to control internal and external interrupt processing.
MICRO INSTRUCTION	A 32-bit micro memory instruction that controls all operations throughout system
MICRO MEMORY	High-speed semiconductor memory that contains micro programs
MICRO PROGRAM	A set of micro instructions stored in micro memory.
MIR	The micro-instruction register; holds the micro instruction being executed.
MM	Micro memory
MP	Micro processor; the basic micro-programmable processor, which can be configured in many forms/ applications.
N REGISTER	An eight-bit counter that can be cleared, incremented, or decremented under micro-instruction control; also used to address file 2.
P REGISTER	General-purpose register used to hold the main memory address of software instruction being executed if the MP is configured as emulator.
PROGRAM PROTECT	Optional logic which, when enabled, prevents unprotected programs and I/O users from changing contents of protected areas of macro memory.
Q REGISTER	General-purpose register used in multiplication and division operations.
RTJ REGISTER	Return jump register; holds the micro-memory address to which control will return at completion of a subroutine.
S FIELD	A special field in the micro-instruction field; specifies the operation to be performed in parallel with the ALU operation.
S1, S2, ETC.	Selector 1, selector 2, etc.
SELECTOR	A multiplexer that allows one of several sources of data to be selected for transfer from one location in the MP organization to another under control of the micro instruction.
SM	Status/mode register; contains flag bits and status/mode bits. Flag bits are set under micro instruction control to enable certain internal MP operations. Status/mode bits indicate internal or external conditions (e.g., memory parity error).
SMI CARD	Status mode interrupt module; contains the status/mode registers, mask registers, and interrupt registers for the micro processor.
T FIELD	Test field in micro-instruction field; specifies if the upper or lower micro instruction of next micro-instruction pair is to be executed.

TRANSFORM MATRIX	Selects bits from various sources in the MP organization and translates them into micro-memory address in the MA register, or transfers them to the K or N register.
X REGISTER	General-purpose processor register
Y REGISTER	Address register on the MP (1700) I/O card

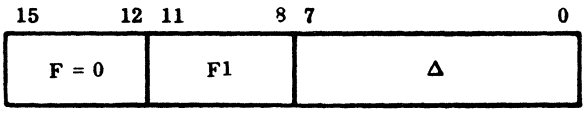
BASIC INSTRUCTIONS

Storage Reference



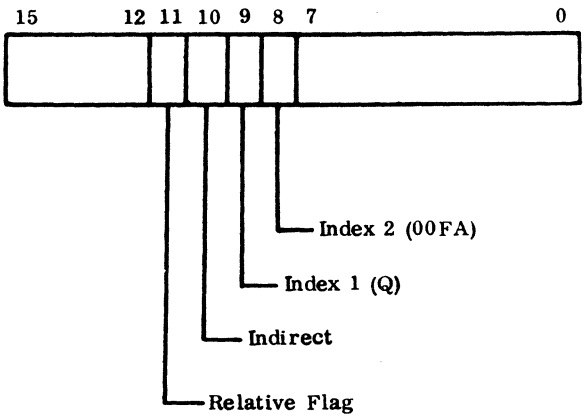
F = 1	JMP
F = 2	MUI
F = 3	DVI
F = 4	STQ
F = 5	RTJ
F = 6	STA
F = 7	SPA
F = 8	ADD
F = 9	SUB
F = A	AND
F = B	EOR
F = C	LDA
F = D	RAO
F = E	LDQ
F = F	ADQ

Register Reference



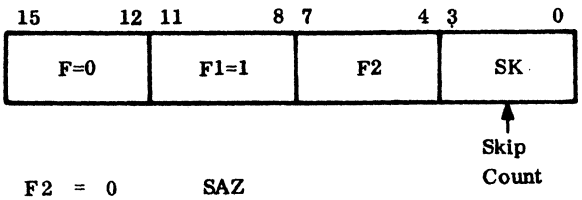
F1 = 0	SLS (Δ=0)
F1 = 1	SKIP
F1 = 2	INP
F1 = 3	OUT
F1 = 4	EIN (Δ=0)
F1 = 5	IIN (Δ=0)
F1 = 6	SPB (Δ=0)
F1 = 7	CPB (Δ=0)
F1 = 8	Inter-register
F1 = 9	INA
F1 = A	ENA
F1 = B	NOP (Δ=0)
F1 = C	ENQ
F1 = D	INQ
F1 = E	EXI
F1 = F	SHIFT

Address Mode



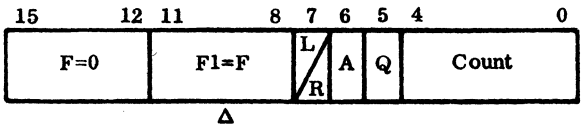
Eight-Bit Absolute:	**0 to 3, Δ≠0**
Eight-Bit Absolute Indirect:	**4 to 7, Δ≠0**
Eight-Bit Relative:	**8 to 11, Δ≠0**
Eight-Bit Relative Indirect:	**12 to 15, Δ≠0**
Absolute Constant:	**0 to 3, Δ=0**
16-Bit Storage:	**4 to 7, Δ=0**
16-Bit Relative:	**8 to 11, Δ=0**
16-Bit Relative Indirect:	**12 to 15 Δ=0**

Skip Format



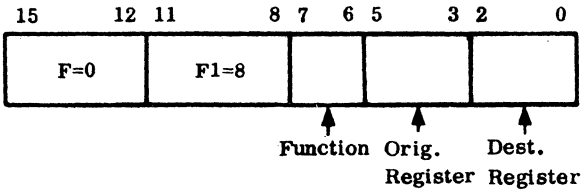
F2 = 0	SAZ
F2 = 1	SAN
F2 = 2	SAP
F2 = 3	SAM
F2 = 4	SQZ
F2 = 5	SQN
F2 = 6	SQP
F2 = 7	SQM
F2 = 8	SWS
F2 = 9	SWN
F2 = A	SOV
F2 = B	SNO
F2 = C	SPE
F2 = D	SNP
F2 = E	SPF
F2 = F	SNF

Shift Format



010xxxxx	ARS
001xxxxx	QRS
011xxxxx	LRS
110xxxxx	ALS
111xxxxx	LLS

Inter-Register Format

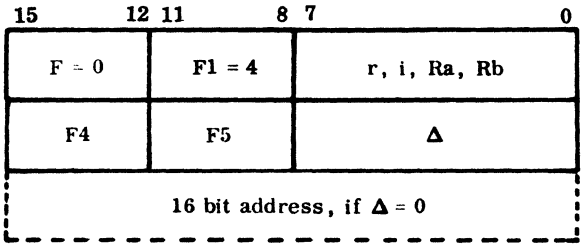


76543	
10000	SET
01000	CLP
10100	TRA
10010	TRQ
10011	TRB
01100	TCA
01001	TCM
01010	TCQ

01011	TCB
00101	AAM
00111	AAB
00110	AAQ
01101	EAM
01110	EAQ
01111	EAB
10101	LAM
10110	LAQ
10111	LAB
11101	CAM
11110	CAQ
11111	CAB

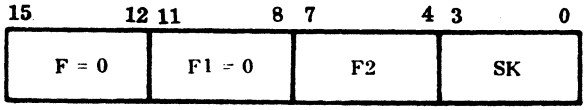
ENHANCED INSTRUCTIONS

Enhanced Storage Reference (r, i, Ra, Rb ≠ 0)



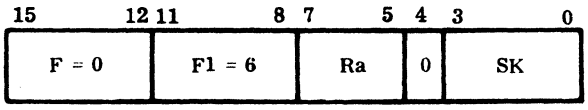
F4 = 5, F5 = 0, Rb = 0	SJE
F4 = 5, F5 = 0, Rb ≠ 0	SJr
F4 = 8, F5 = 0, Rb ≠ 0	ARr
F4 = 9, F5 = 0, Rb ≠ 0	SBr
F4 = A, F5 = 0, Rb ≠ 0	ANr
F4 = A, F5 = 1, Rb ≠ 0	AMr
F4 = C, F5 = 0, Rb ≠ 0	LRr
F4 = C, F5 = 1, Rb ≠ 0	SRr
F4 = C, F5 = 2	LCA
F4 = C, F5 = 3	SCA
F4 = D, F5 = 0, Rb ≠ 0	ORr
F4 = D, F5 = 1, Rb ≠ 0	OMr
F4 = E, F5 = 0, Rb ≠ 0	CrE
F4 = E, F5 = 2	CCE

Enhanced Skip Instruction



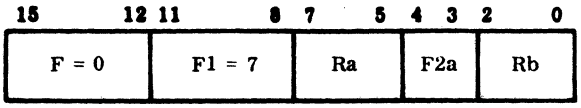
F2 = 0, 4, 8, or C r = 4, 1, 2, or 3	SrZ SK
F2 = 1, 5, 9, or D r = 4, 1, 2, or 3	SrN SK
F2 = 2, 6, A, or E r = 4, 1, 2, or 3	SrP SK
F2 = 3, 7, B, or F r = 4, 1, 2, or 3	SrM SK

Decrement and Repeat



Ra = 1 to 7	DrP SK

Enhanced Inter-Register

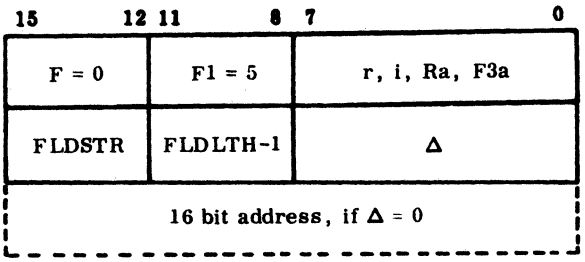


F2a = 0, Ra = 1-7 XFr R
Rb = 1-4, Q, A, I

NOTE

Ra, Rb (0 to 7) = 0, 1, 2, 3, 4, Q, A, I

Field Reference



F3a = 2 SFZ

F3a = 3 SFN

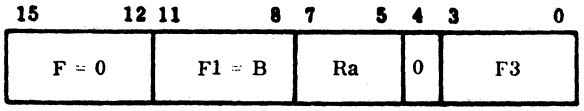
F3a = 4 LFA

F3a = 5 SFA

F3a = 6 CLF

F3a = 7 SEF

Miscellaneous



F3 = 1, Ra = 0 LMM

F3 = 2, Ra = 0 LRG

F3 = 3, Ra = 0 SRG

F3 = 4, Ra = 0 SIO

F3 = 5, Ra = 0 SPS

F3 = 6, Ra = 0 DMI

F3 = 7, Ra = 0 CBP

F3 = 8, Ra = 0 GPE

F3 = 9, Ra = 0 GPO

F3 = A, Ra = 0 ASC

F3 = 0, Ra = r LUB

F3 = 1, Ra = r LLB

F3 = 2, Ra = r EMS

When calculating the instruction execution times listed on the following pages, the user must consider the following parameters:

1. For basic storage reference instructions, add the following addressing mode time to the execution time given in this appendix.

F1	Delta	Additional Execution Time	F1	Delta	Additional Execution Time
0	≠ 0	.00	0	= 0	.29
1		.00	1		.51
2		.00	2		.51
3		.29	3		.51
4		.80	4		.80
5		.80	5		.80
6		.80	6		.80
7		.80	7		.80
8		.00	8		.80
9		.29	9		.80
10		.29	10		.80
11		.57	11		1.07
12		.80	12		1.57
13		.80	13		1.57
14		.80	14		1.57
15		.80	15		1.57

2. For an enhanced storage reference field instruction, add the following address mode time to the execution time given in this appendix.

Addressing Mode	Delta	Additional Execution Time
0	≠ 0	.00
1		.67
2		.28
3		.78

Addressing Mode	Delta	Additional Execution Time
0	= 0	.22
1		.72
2		.72
3		1.50

3. Add .72 microseconds for the following instructions:

   SJI

   ARJ

   SBI

   ANI

   LRI

   ORI

Mnemonic	Definition	Execution Times (μsec)	OP Code			
AAB	Transfer Arithmetic Sum A, Q+M	1.74, 1.91, 2.08, 2.25	0	8	3	8—F
AAM	Transfer Arithmetic Sum A, M	1.74, 1.91, 2.08, 2.25	0	8	2	8—F
AAQ	Transfer Arithmetic Sum A, Q	1.10, 1.34, 1.51, 1.54†	0	8	3	0—7
ADD	ADD A	1.76	8	0 to F	Δ	
ADQ	ADD Q	1.76	F	0 to F	Δ	
ALS	A Left Shift	1.62 + .056 * N	0	F	C/D	0—F
AMr	AND Memory	5.68	0	4	0 to F	0 to F
			A	1	Δ	
AND	AND with A	1.62	A	0 to F	Δ	
ANr	AND Register	5.40	0	4	0 to F	0 to F
			A	0	Δ	
ARr	Add Register	5.40	0	4	0 to F	0 to F
			8	0	Δ	
ARS	A Right Shift	1.62 + .056 * N	0	F	4/5	0—F
ASC	Accumulator Scale	2.88 + .056 * N	0	B	0	A
CAB	Transfer Complement Logical Product A, Q+M	1.63, 1.80, 1.96, 2.13†	0	8	F	8—F
CAM	Transfer Complement Logical Product A, M	1.63, 1.80, 1.96, 2.13†	0	8	E	8—F
CAQ	Transfer Complement Logical Product A, Q	1.18, 1.34, 1.34, 1.51†	0	8	F	0—7
CBP	Clear Breakpoint Interrupt	2.19	0	B	0	7
CCE	Compare Character Equal	6.14	0	4	0—F	0—F
			E	2	Δ	

†For inter-register instructions, the first execution time is for A or Q register destinations, the second time is for M, A and M, or Q and M registers, the third time is for A and Q, and the fourth time for A and Q and M.

Mnemonic	Definition	Execution Times (μsec)	OP Code			
CLF	Clear Field	6.64	0 0 to F	5 0 to F	+ Δ	6
CLR	Clear to Zero	1.18, 1.34, 1.34, 1.51†	0	8	4	0 to 7
CPB	Clear Program Protect	1.72	0	7	0	0
CrE	Compare Register Equal	5.23, 5.46††	0 E	4 0	0 to F Δ	0 to F
DMI	Define Micro Interrupt	3.43	0	B	0	6
DrP	Decrement and Repeat	2.22†††	0	6	††††	0 to F
DVI	Divide Integer	10.48	3	0 to F	Δ	
EAB	Transfer Exclusive OR A, Q, M	1.63, 1.80, 1.96, 2.13†	0	8	7	8 to F
EAM	Transfer Exclusive OR A, M	1.63, 1.80, 1.96, 2.13†	0	8	6	8 to F
EAQ	Transfer Exclusive OR A, Q	1.18, 1.34, 1.34, 1.51†	0	8	7	0 to 7
EIN	Enable Interrupt	1.40	0	4	0	0
EMS	Execute Micro Sequence	6.20†††††	0	B	r, o	2
ENA	Enter A	.95	0	A	Δ	
ENQ	Enter Q	.95	0	C	Δ	
EOR	Exclusive OR with A	1.62	B	0 to F	Δ	
EXI	Exit Interrupt State	1.85	0	E	Δ	

†For inter-register instructions, the first execution time is for A or Q register destinations, the second time is for M, A and M, or Q and M registers, the third time is for A and Q, and the fourth time for A and Q and M.

††For compare instructions, the first execution time listed is for unequal conditions, and the second time is for equal conditions.

†††Add .62 microseconds for the DIP instruction.

††††2, 4, 6, 8, A, C, E

†††††Plus micro-sequence time

Mnemonic	Definition	Execution Times (μsec)	OP Code			
GPE	Generate Character Parity Even	3.92	0	B	0	8
GPO	Generate Character Parity Odd	3.92	0	B	0	9
IIN	Inhibit Interrupt	1.40	0	5	0	0
INA	Increase A	.95	0	9	Δ	
INP	Input to A	3.77 min. 15.63 max†	0	2	Δ	
INQ	Increase Q	.95	0	D	Δ	
JMP	Jump	1.17	1	0 to F	Δ	
LAB	Transfer Logical Product A, Q+M	1.63, 1.80, 1.96, 2.13††	0	8	B	8 to F
LAM	Transfer Logical Product A, M	1.63, 1.80, 1.96, 2.13††	0	8	A	8 to F
LAQ	Transfer Logical Product A, Q	1.10, 1.34, 1.34, 1.51††	0	8	B	0 to 7
LCA	Load Character to A	5.80	0 C	4 2	0 to F Δ	0 to F
LDA	Load A	1.62	C	0 to F	Δ	
LDQ	Load Q	1.62	E	0 to F	Δ	
LFA	Load Field	6.19 + .056 * N	0 0 to F	5 0 to F	0 to F Δ	4 or C
LLB	Load Lower Unprotected Bounds	2.14	0	B	r, o	1
LLS	Long Left Shift	2.30 + .056 * N	0	F	E/F	0 to F
LMM	Load Micro Memory	2.42 + 3.5 * N	0	B	0	1
LRG	Load Registers	13.80	0	B	0	2

†Maximum time is for internal reject

††For inter-register instructions, the first execution time is for A or Q register destinations, the second time is for M, A and M, or Q and M registers, the third time is for A and Q, and the fourth time for A and Q and M.

Mnemonic	Definition	Execution Times (μsec)	OP Code			
LRr	Load Register	5.40	0 C	4 0	0 to F Δ	0 to F
LRS	Load Right Shift	2.30 + .056 * N	0	F	6/7	0 to F
LUB	Load Upper Unprotected Bounds	2.14	0	B	r, o	0
MUI	Multiply Integer	5.62 min. 7.49 max.	2	0 to F	Δ	
NOP	No Operation	1.17	0	F	0 to 1	0 to F
OMr	OR Memory	5.68	0 D	4 1	0 to F Δ	0 to F
ORr	OR Register	5.40	0 D	4 0	0 to F Δ	0 to F
OUT	Output from A	3.49 min. 15.63 max.†	0	3	Δ	
QLS	Q Left Shift	1.96 + .056 * N	0	F	A or B	0 to F
QRS	Q Right Shift	1.96 + .056 * N	0	F	2 or 3	0 to F
RAO	Replace Add 1 in Storage	2.22	D	0 to F	Δ	
RTJ	Return Jump	1.69	5	0 to F	Δ	
SAM	Skip if A = -	1.23, 1.52††	0	1	3	0 to F
SAN	Skip if A ≠ +0	1.23, 1.52††	0	1	1	0 to F
SAP	Skip if A = +	1.23, 1.52††	0	1	2	0 to F
SAZ	Skip if A = +0	1.23, 1.52††	0	1	0	0 to F
SBr	Subtract Register	5.47	0 9	4 0	0 to F Δ	0 to F
SCA	Store Character from A	6.53	0 C	4 3	0 to F Δ	0 to F

†Maximum time is for internal reject

††For skip instructions, the first execution time is for no skip, and the second is for skip.

Mnemonic	Definition	Execution Times (μsec)	OP Code			
SEF	Set Field	6.64	0 0 to F	5 0 to F	0 to F Δ	7 or F
SET	Set to 1s	1.18, 1.34, 1.34, 1.51†	0	8	8	0 to 7
SFA	Store Field	7.15 * .056N	0 0 to F	5 0 to F	0 to F Δ	5 or D
SFN	Skip if Field Nonzero	5.85, 6.08††	0 0 to F	5 0 to F	0 to F Δ	3 or B
SFZ	Skip if Field Zero	5.85, 6.08††	0 0 to F	5 0 to F	0 to F Δ	2 or A
SIO	Set/Sample Output or Input	3.88	0	B	0	4
SJE	Subroutine Jump Exit	4.50	0 5	4 0	0 to F Δ	0 to F
SJr	Subroutine Jump	4.67	0 5	4 0	0 to F Δ	0 to F
SLS	Select Stop	1.35	0	0	0	0
SNF	Skip on No Program Protect Fault	1.17, 1.46††	0	1	B	0 to F
SNO	Skip on No Overflow	1.17, 1.46††	0	1	F	0 to F
SNP	Skip on No Storage Parity Error	1.35, 1.46††	0	1	D	0 to F
SOV	Skip on Overflow	1.17, 1.46††	0	1	A	0 to F
SPA	Store A, Parity to A	2.18	7	0 to F	Δ	
SPB	Set Program Protect	1.72	0	6	0	0
SPE	Skip on Storage Parity Error	1.35, 1.46††	0	1	C	0 to F
SPF	Skip on Program Protect Fault	1.17, 1.46††	0	1	E	0 to F

†For inter-register instructions, the first execution time is for A or Q register destinations, the second time is for M, A and M, or Q and M registers, the third time is for A and Q, and the fourth time for A and Q and M.

††For skip instructions, the first execution time is for no skip, and the second is for skip.

Mnemonic	Definition	Execution Times (μsec)	OP Code			
SPS	Sample Position Status	3.94	0	B	0	5
SQM	Skip if Q = -	1.23, 1.52†	0	1	7	0 to F
SQN	Skip if Q ≠ +0	1.23, 1.52†	0	1	5	0 to F
SQP	Skip if Q = +	1.23, 1.52†	0	1	6	0 to F
SQZ	Skip if Q = +0	1.23, 1.52†	0	1	4	0 to F
SRG	Store Registers	12.59	0	B	0	3
SrM	Skip if Register Negative	1.91, 2.02†	0	0	3, 7 B, F	0 to F
SrN	Skip if Register Nonzero	1.91, 2.02†	0	0	1, 5 9, D	0 to F
SrP	Skip if Register Positive	1.91, 2.02†	0	0	2, 6 A, E	0 to F
SRr	Store Register	5.51	0 C	4 1	0 to F Δ	0 to F
SrZ	Skip if Register Zero	1.91, 2.02†	0	0	0, 4 8 C	0 to F
STA	Store A	1.69	6	0 to F	Δ	
STQ	Store Q	1.69	4	0 to F	Δ	
SUB	Subtract	1.76	9	0 to F	Δ	
SWN	Skip if Switch not Set	1.12, 1.40†	0	1	9	0 to F
SWS	Skip if Switch Set	1.12, 1.40†	0	1	8	0 to F
TCA	Transfer Complement A	1.18, 1.34, 1.34, 1.51††	0	8	6	0 to 7
TCB	Transfer Complement Q+M	1.46, 1.62, 1.79, 1.96††	0	8	5	8 to F

†For skip instructions, the first execution time is for no skip, and the second is for skip.

††For inter-register instructions, the first execution time is for A or Q register destinations, the second time is for M, A and M, or Q and M registers, the third time is for A and Q, and the fourth time for A and Q and M.

Mnemonic	Definition	Execution Times (μsec)	OP Code			
TCM	Transfer Complement M	1.46, 1.62, 1.79, 1.96†	0	8	4	8 to F
TCQ	Transfer Complement Q	1.18, 1.34, 1.34, 1.51†	0	8	5	0 to 7
TRA	Transfer A	1.18, 1.34, 1.34, 1.51†	0	8	A	0 to 7
TRB	Transfer Q+M	1.46, 1.62, 1.79, 1.96†	0	8	9	8 to F
TRM	Transfer M	1.46, 1.62, 1.79, 1.96†	0	8	8	8 to F
TRQ	Transfer Q	1.18, 1.34, 1.34, 1.51†	0	8	9	0 to 7
XFr	Transfer Register	2.47††	0	7	0, 1 to 7	1 to 7

†For inter-register instructions, the first execution time is for A or Q register destinations, the second time is for M, A and M, or Q and M registers, the third time is for A and Q, and the fourth time for A and Q and M.

††Add .67 microseconds for XFI instruction.

INDEX

COMMENT SHEET

MANUAL TITLE CDC® Micro-Programmable Computer Family

1700 Enhanced Processor with Core Memory Hardware Reference Manual

PUBLICATION NO. 88973500 REVISION B

FROM NAME: _____

BUSINESS
ADDRESS: _____

COMMENTS: This form is not intended to be used as an order blank. Your evaluation of this manual will be welcomed by Control Data Corporation. Any errors, suggested additions or deletions, or general comments may be made below. Please include page number.

CUT ALONG LINE

STAPLE

STAPLE

FOLD

FIRST CLASS
PERMIT NO. 333

LA JOLLA, CA.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION
PUBLICATIONS AND GRAPHICS DIVISION
4455 EASTGATE MALL
LA JOLLA, CALIFORNIA 92037

CUT ALONG LINE

FOLD

STAPLE

STAPLE