74375500

CONTROL DATA CORPORATION

CDC® 2550-2(MOS) HOST COMMUNICATIONS PROCESSOR

2554-16/-32 MEMORY EXPANSION MODULES

2556-2/-3/-4 COMMUNICATIONS LINE EXPANSION UNITS

2558-1 COMMUNICATIONS COUPLER

2571-1 PERIPHERAL CONTROLLER

SYSTEM DESCRIPTION
FUNCTIONAL DESCRIPTIONS
OPERATING INSTRUCTIONS
INSTRUCTION DESCRIPTIONS
PROGRAMMING INFORMATION

HARDWARE REFERENCE MANUAL

REVISION RECORD

REVISION	DESCRIPTION
01	Preliminary Edition
(10-13-75)	
A	Manual released. This edition obsoletes all previous editions.
(6-17-77)	

Publication No.
74375500

Address comments concerning this manual to:
Control Data Corporation
Publications & Graphics Division
3519 West Warner Avenue
Santa Ana, California 92704

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	-								
ii thru xii	A								
1-1 thru 1-17	A								
2-1 thru 2-44	A								
3-1 thru 3-32	A								
4-1 thru 4-53	A								
5-1 thru 5-73	A								
A-1 thru A-4	A								
B-1	A								
C-1 thru C-3	A								
D-1 thru D-4	A								
E-1, E-2	A								
F-1 thru F-9	A								
G-1, G-2	A								
H-1	A								
I-1	A								
J-1	A								
Index-1,-2	A								

PREFACE

This manual describes the functional, operational and programming characteristics of the CONTROL DATA® 2550-2 (MOS) Host Communications Processor (HCP). The manual also provides information on seven products which are used to expand the 2550-2 HCP capabilities. These products are the 2554-16 and 2554-32 Memory Expansion Modules; 2556-2, 2556-3 and 2556-4 Communications Line Expansion Units; 2558-1 Communications Coupler, and 2571-1 Peripheral Controller.

The related publications listed below are available through the CDC Literature Distribution Services, Minneapolis, Minnesota.

Publication	Publication Number
2550-2 (MOS) Host Communications Processor - Hardware Maintenance Manual	74701000
2550 Series Host Communications Processor - Site Preparation Manual	74641200
2560-1/-2/-3 Synchronous Communications Line Adapters - Hardware Reference and Maintenance Manual	74700700
2561-1 Asynchronous Communications Line Adapter - Hardware Reference and Maintenance Manual	74700900
1714 Computer System - Reference Manual	60364900
Basic Microprogrammable Processor - Hardware Maintenance Manual	3945140
CW212-A I/O TTY Controller - Hardware Maintenance Manual	96728900
DE402-A/1700 Transform with Micromemory - Hardware Maintenance Manual	96728700
AA109-A/B 1700 Enhanced Processor with MOS Memory and Interface - Hardware Maintenance Manual	96768600
FA104-A Tape Cassette Controller - Hardware Maintenance Manual	96711900

2558-2 Communications Coupler 3000L - Hardware Reference/ Maintenance Manual	60470400
6671/6676 Emulation Coupler - Hardware Reference/Maintenance Manual	74849600

CONTENTS

APPENDIX

INDEX

FIGURES

TABLES

SYSTEM DESCRIPTION 1

INTRODUCTION

The 2550-2 HCP is a medium-size, digital, data communications system that serves as a communications front-end to a CONTROL DATA® 6000, CYBER 70, CYBER 170, or a lower 3000 series host computer. The 2550-2 HCP also interfaces with selected terminal devices. See block diagram, figure 1-1.

HOST INTERFACING CAPABILITY

The 2550-2 HCP normally interfaces with one host computer. However, by adding a 2558-1 communications coupler to the system, the HCP can interface either with a second host computer, or with another channel of the same host.

The 3000L coupler is available to interface the 2550-2 HCP with the CDC 3000L series processor. See preface.

The 2550-100 emulation module and the 2558-2 emulation coupler used with the 2550-2 HCP emulates multiples of the 6671 and 6676 data set controllers without host software modifications. See figure 1-1.

COMMUNICATIONS INTERFACING CAPABILITY

The 2550-2 HCP provides multiplexer capabilities to interface as many as 32 communications line adapters (CLAs). This interfacing capability can be expanded to as many as 128 CLAs by adding 2556-2, -3, and -4 communications line expansion (CLE) units. Each CLE configuration provides multiplexer capacity to interface 32 additional CLAs. A variety of CLA types are available for use with the HCP, as required. See table 1-1.

TABLE 1-1. COMMUNICATIONS LINE EXPANSION OPTIONS

Increment	Configuration	Number of CLAs to be Interfaced
1	Basic 2550-2 HCP	1-32
2	Add 2556-2 CLE	33-54
3	Add 2556-3 CLE	65-96
4	Add 2556-4 CLE	97-128

MP17 COMMUNICATIONS PROCESSOR

The 2550-2 HCP contains an MP17 communications processor (CP). The MP17 CP is a modified MP16 computer frame and associated modules. The MP17 transform module (and 1700 emulation controlware stored on it) effects the conversion to MP17. Several special purpose modules were added to accommodate the 2550 HCP configuration.

MAIN MEMORY CAPACITY

The main MOS memory for the 2550-2 HCP holds 32,768 16-bit words. This capacity can be expanded to 49,152 words by addition of a 2554-16 memory expansion module or to 65,536 words by addition of a 2554-32 memory expansion module.

COMMUNICATIONS CONSOLE

A communications console is required to operate the HCP, and the user can select from either a number of commercially-available teletypewriters (TTYs) or cathode ray tube (CRT) conversational display terminals for this purpose. The 713-10 CRT or the 1711-4, -5 and 1713-4, -5 TTYs are compatible with the 2550-2 HCP.

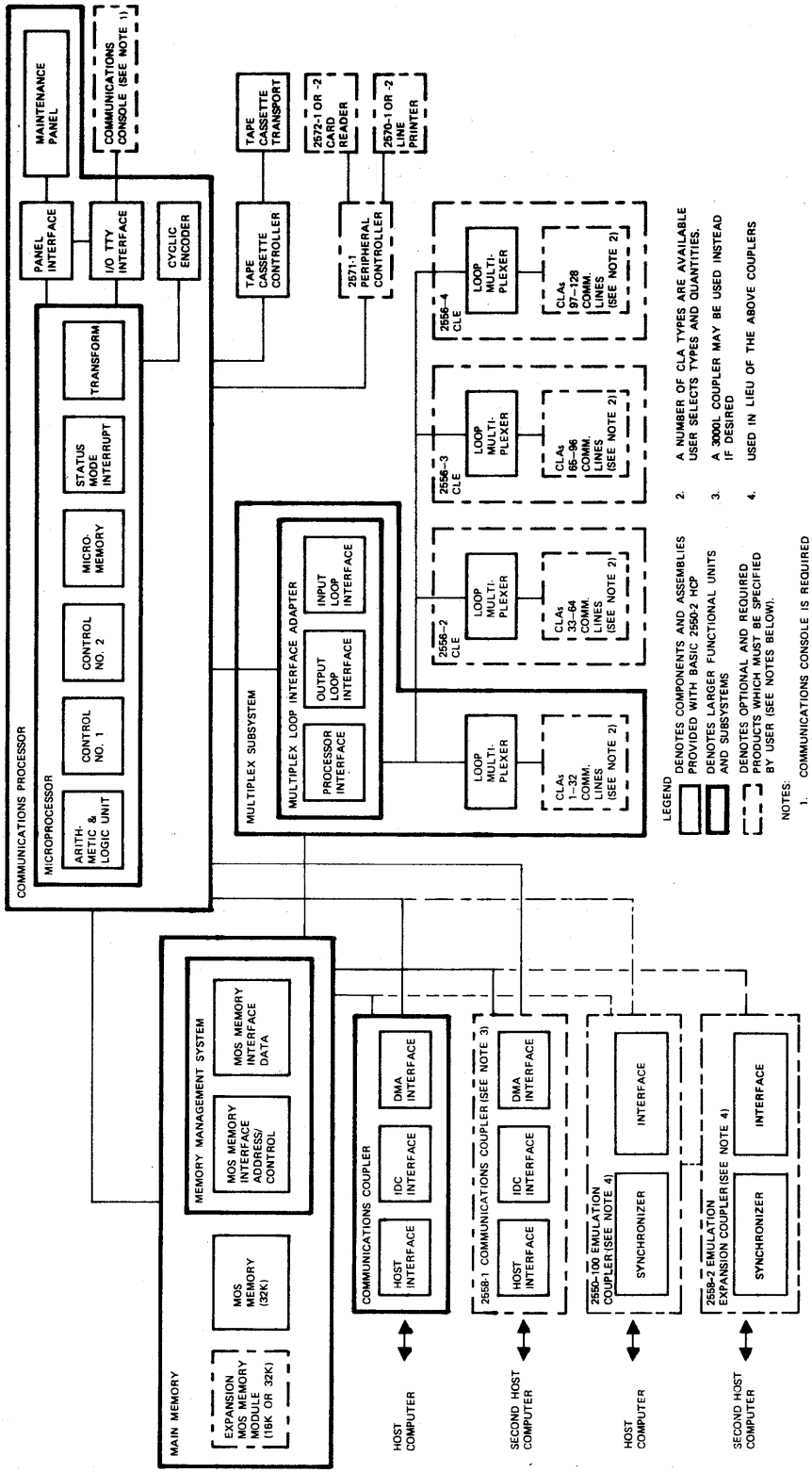


Figure 1-1. 2550-2 Host Communications Processor, Block Diagram

The 713-10 Conversational Display Terminal is a desktop unit containing a CRT display and a keyboard. The unit operates at variable speeds up to 1200 characters per second (9600 baud).

The 1711-4 and -5 TTYs are capable of operating in the keyboard send-receive (KSR) mode only. The 1713-4 and -5 TTYs contain paper-tape punches and decoders, and are capable of automatic send-receive (ASR) operations, as well as KSR operation.

OPTIONAL PERIPHERALS

The addition of a 2571-1 Peripheral Controller to the system enables the HCP to receive input from a card reader and provide output to a line printer. Either a 2572-1, or 2572-2 Card Reader may be used with the 2550-2 HCP. These units are identical except that the 2572-1 Card Reader operates at a speed of 300 cards per minute, while the 2572-2 configuration reads 600 cards per minute. The 2550-2 HCP is also compatible with either the 2570-1 or the 2570-2 Line Printers. The 2570-1 unit employs a drum-printing mechanism and prints 300 lines per minute; the 2570-2 configuration contains a train-printing mechanism and prints 1200 lines per minute.

PHYSICAL CHARACTERISTICS

A front view of the HCP is shown as figure 1-2, and a side view as figure 1-3.

The major system components of the 2550-2 HCP are listed below:

1. Communications Processor
2. Main Memory
3. Multiplex Subsystem
4. Tape Cassette Transport and Controller
5. Communications Coupler
6. Wired Cabinet Assembly
7. Communications Console
8. Optional Peripherals and Controller

COMMUNICATIONS PROCESSOR

The communications processor (CP) contains the following assemblies and components:

1. Microprocessor
2. Maintenance Panel
3. Maintenance Panel Interface
4. I/O TTY Interface
5. Cyclic Encoder

Microprocessor

The microprocessor contains the following six printed circuit cards which are located in the slots indicated within the CP card cage assembly. See figure 1-4.

Card Slot	Printed Circuit Card Nomenclature
M	Arithmetic and Logic Unit (ALU)
P	Control No. 1
N	Control No. 2
T	Micromemory
L	Status Mode Interrupt
R	1700 Transform

Each of these circuit cards is 11 in. x 14 in. (28 cm x 36 cm). These cards contain the circuitry which performs the various arithmetic and logic operations which control the flow of data through the HCP system.

Maintenance Panel

As shown in figure 1-2, the maintenance panel is located above the CP card cage assembly in Bay 0. Maintenance panel controls and indicators are described in section 3. The maintenance panel is illustrated in figure 3-1.

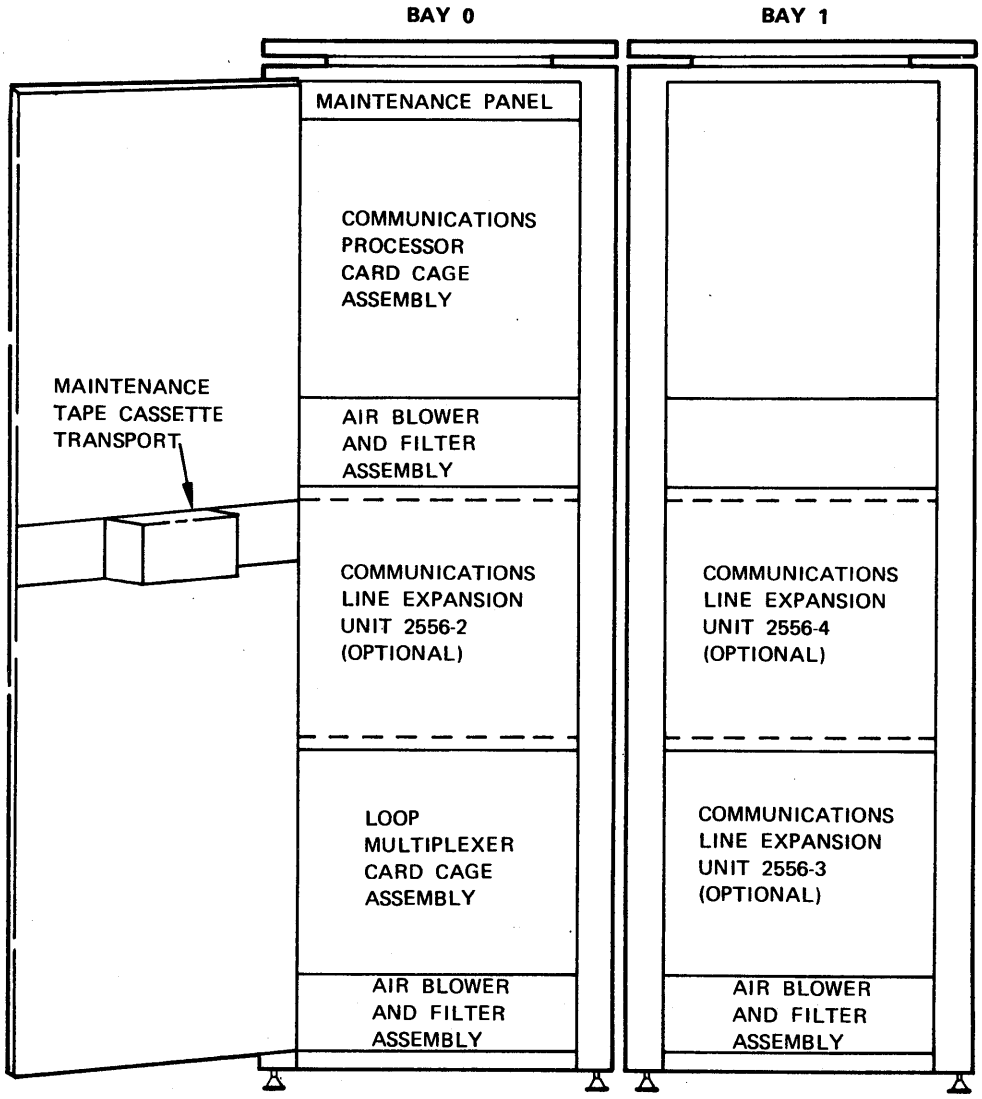


Figure 1-2. 2550-2 Host Communications Processor, Front View

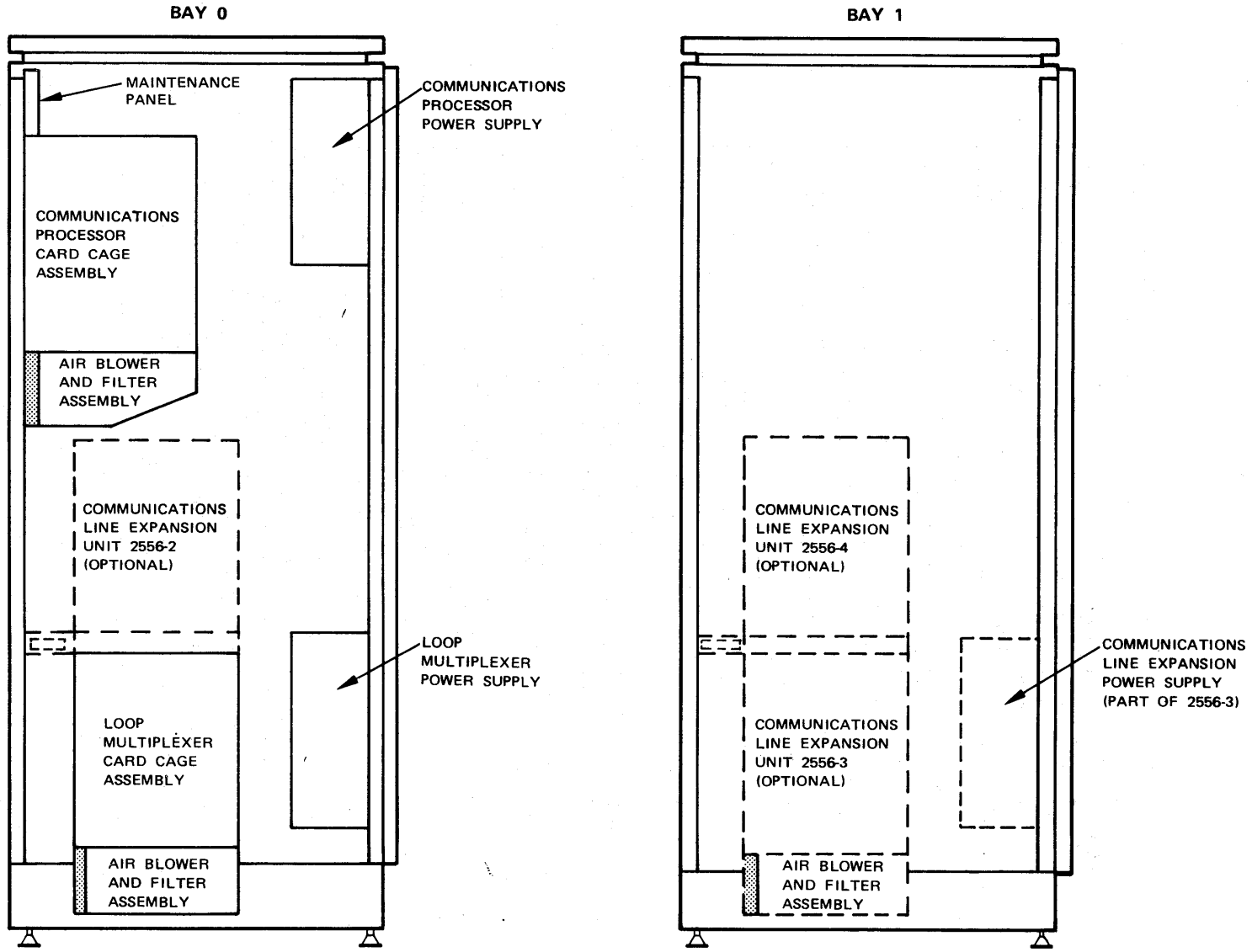


Figure 1-3. 2550-2 Host Communications Processor, Side View

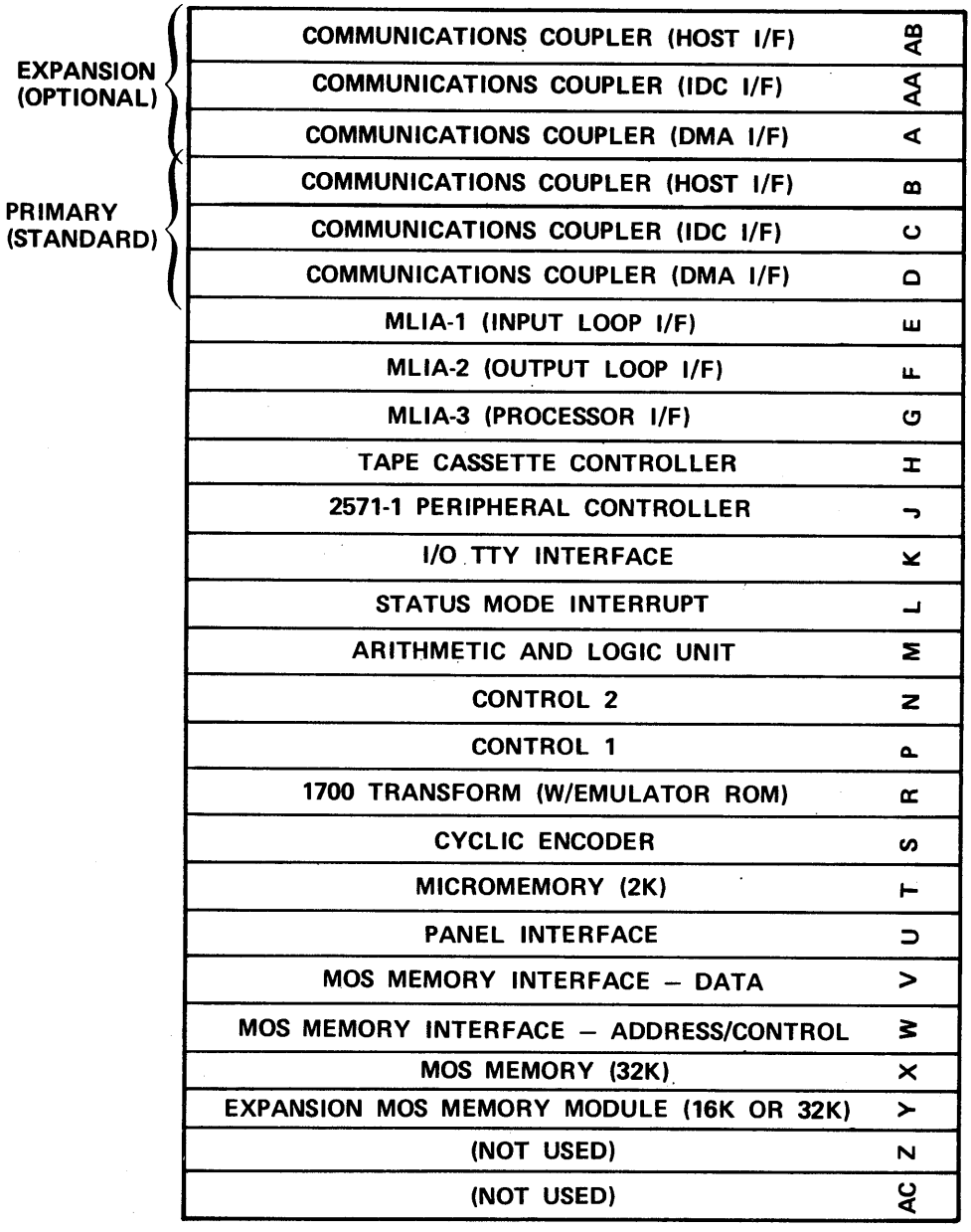


Figure 1-4. Communications Processor Card Cage Assembly

Panel I/F, I/O TTY I/F, and Cyclic Encoder

Three additional printed circuit cards are used in the CP for the basic 2550-2 configuration. These cards are 11 in. x 14 in. (28 cm x 36 cm), and are located in the slots indicated within the card cage assembly. See figure 1-4.

Card Slot	Printed Circuit Card Nomenclature
U	Panel Interface
K	I/O TTY Interface
S	Cyclic Encoder

The panel and I/O TTY interface circuit cards provide the interfaces for the maintenance panel and the communications console to the CP. The cyclic encoder generates check characters.

MAIN MEMORY

Main memory, for the basic 2550-2 HCP configuration, consists of three printed circuit cards which are installed in the slots indicated within the CP card cage assembly. See figure 1-3.

Slot	Circuit Card
V	MOS Memory Interface - Data
W	MOS Memory Interface - Address/Control
X	MOS Memory (32K)

The MOS memory circuit card provides 32,768 words of main memory. The other two circuit cards (known as the Memory Management System) provide the interface between main memory and the microprocessor. All of these circuit cards are 11 in. x 14 in. (28 cm x 36 cm).

Main memory capacity can be increased by adding either of the MOS memory expansion circuit cards listed in table 1-2.

TABLE 1-2. MOS MEMORY CAPACITY

Control Data Product No.	Added Memory Capacity	Total System Memory Capacity
2554-16	16,384	49,152
2554-32	32,768	65,536

NOTE

If a 2554-16 memory expansion module has been installed in a system, it must be removed prior to adding the 2554-32 memory expansion module.

COMMUNICATIONS COUPLER

In the basic 2550-2 HCP configuration, the communications coupler consists of the following three printed circuit cards which are located in the indicated slots of the CP card cage assembly. See figure 1-4.

Card Slot	Printed Circuit Card Nomenclature
B	Host Interface
C	IDC Interface
D	DMA Interface

The expansion 2558-1 Communications Coupler consists of the following three printed circuit cards which are located in the indicated slots of the CP card cage assembly. See figure 1-3.

Card Slot	Printed Circuit Card Nomenclature
AB	Host Interface
AA	IDC Interface
A	DMA Interface

6671/6676 EMULATION COUPLER

The 6671/6676 emulation coupler requires one 2550-100 emulation module which contains two circuit cards. See figure 1-5. The emulation coupler expansion is accomplished by addition of a 2558-2 emulation coupler which also contains two circuit cards. The 2550-100 interface and synchronizer circuit cards form the primary 6671/6676 emulation coupler and are located in slots C and B. The 2558-2 circuit cards located in card slots AA and AB form the expansion 6671/6676 emulation coupler.

MULTIPLEX SUBSYSTEM

The multiplex subsystem comprises the following major system components:

1. Multiplex Loop Interface Adapter (MLIA)
2. Loop Multiplexer (LM)
3. Communications Line Adapters (CLAs)

Multiplex Loop Interface Adapter

The multiplex loop interface adapter (MLIA) comprises the following three printed circuit cards which are located in the slots indicated in the CP card cage assembly (see figure 1-4):

Card Slot	Printed Circuit Card Nomenclature
E	Input Loop Interface
F	Output Loop Interface
G	Processor Interface

Each of these circuit cards is 11 in. z 14 in. (28 cm x 36 cm).

Loop Multiplexer and Communications Line Adapters

The loop multiplexer (LM) is located in the second circuit card slot from the right, in the LM card cage assembly as shown in figure 1-6. The 16 circuit card slots to the left of the LM can be used for CLAs. A typical CLA installation is shown in figure 1-4; however, the total number of CLAs used, as well as the types of CLAs, is determined by the user.

All configurations of the communications line expansion units (2556-2, -3, -4) have the same placement of LM and CLA cards as that described above for the LM card cage assembly.

The LM and CLA circuit cards are 11 in. x 14 in. (28 cm x 36 cm). A narrow (7/8-inch [2.2 cm]-wide) control panel is mounted on the front-facing edge of each card, which contains controls and indicators.

The LM printed circuit card contains a power (PWR) switch and the following four light-emitting diode (LED) indicators:

1. IN LOOP CLK
2. IN LOOP DATA
3. OUT LOOP CLK
4. OUT LOOP DATA

The function of the switch and indicators is described in section 3 of this manual.

The CLA controls and indicators vary according to the type(s) of CLAs selected, and the user should refer to the appropriate CLA manual(s) (see Preface).

TAPE CASSETTE TRANSPORT AND CONTROLLER

The tape cassette transport is mounted on the inner surface of the door for Bay 0 as shown in figure 1-1.

The length of the transport is approximately 7.5 in. (19 cm); this is the length of the transport surface which contacts the door.

The height of the transport is approximately 5.4 in. (13.7 cm) with the top cover closed. The depth of the transport is 5.1 in. (13 cm); this is

EXPANSION (OPTIONAL)	2558-2 EMULATION COUPLER SYNCHRONIZER	AB
	2558-2 EMULATION COUPLER INTERFACE	AA
	(NOT USED)	A
PRIMARY (STANDARD)	2550-100 EMULATION COUPLER SYNCHRONIZER	B
	2550-100 EMULATION COUPLER INTERFACE	C
	(NOT USED)	D
	MLIA-1 (INPUT LOOP I/F)	E
	MLIA-2 (OUTPUT LOOP I/F)	F
	MLIA-3 (PROCESSOR I/F)	G
	TAPE CASSETTE CONTROLLER	H
	2571-1 PERIPHERAL CONTROLLER	J
	I/O TTY INTERFACE	K
	STATUS MODE INTERRUPT	L
	ARITHMETIC AND LOGIC UNIT	M
	CONTROL 2	N
	CONTROL 1	P
	1700 TRANSFORM (W/EMULATOR ROM)	R
	CYCLIC ENCODER	S
	MICROMEMORY (2K)	T
	PANEL INTERFACE	U
	MOS MEMORY INTERFACE – DATA	V
	MOS MEMORY INTERFACE – ADDRESS/CONTROL	W
	MOS MEMORY (32K)	X
	EXPANSION MOS MEMORY MODULE (16K OR 32K)	Y
	(NOT USED)	Z
	(NOT USED)	AC

Figure 1-5. CP Card Cage Configuration for 2550-100/2558-2 Emulation Couplers

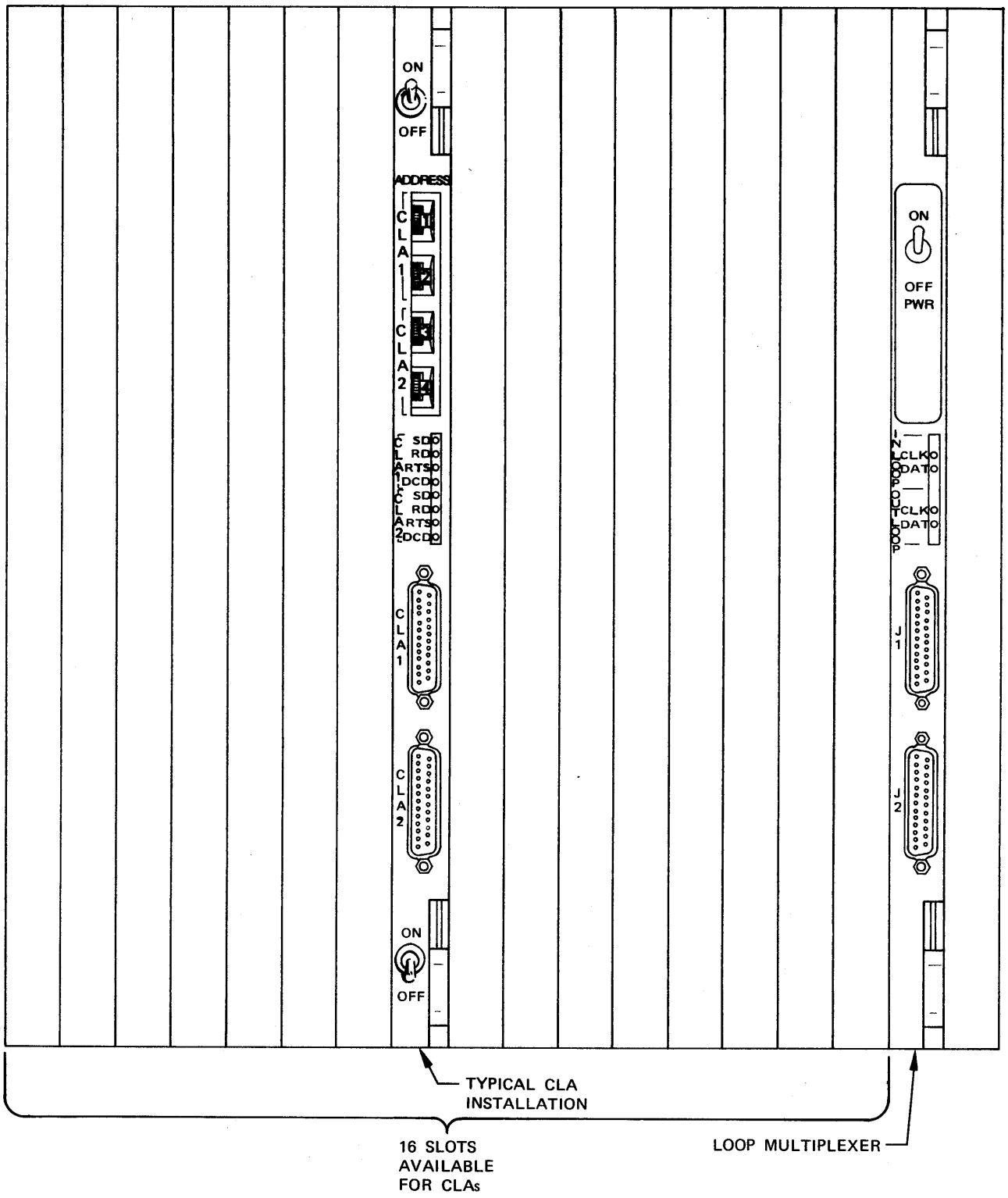


Figure 1-6. Location of Loop Multiplexer and Typical Communications Line Adapter

the dimension of the transport which extends into the cabinet when the HCP door is closed.

The tape cassette controller is an 11 in. x 14 in. (28 cm x 36 cm) circuit card which is installed in slot H of the CP card cage assembly. See figure 1-4.

WIRED CABINET ASSEMBLY

The following major system components are installed in the wired cabinet assembly:

1. Power Supplies
2. Air Blower and Filter Assemblies
3. Cabling

POWER SUPPLIES

As shown in figure 1-3, two power supplies are provided with the basic 2550-2 HCP, and a third is provided as a part of the optional 2556-3 CLE product. These power supplies are identified as follows:

1. Communications Processor Power Supply
2. Loop Multiplexer Power Supply
3. Communications Line Expansion Power Supply (part of optional 2556-3 CLE)

All of the power supplies are mounted on hinges, and can be swung outward (from the rear of the HCP) to provide access to circuit breakers, fuses, and voltage-adjustment controls. All three power supplies have the following dimensions:

Height:	17.6 in. (45 cm)
Width:	15.3 in. (39 cm)
Depth:	8.4 in. (21 cm)

AIR BLOWER AND FILTER ASSEMBLIES

As shown in figure 1-3, two air blower and filter assemblies (ABFAs) are provided with the basic 2550-2 HCP, and a third is provided as part of the optional 2556-3 CLE product. These ABFAs are identified as follows:

1. Communications Processor ABFA
2. Loop Multiplexer ABFA
3. Communications Line Expansion ABFA (part of optional 2556-3 CLE)

CABLING

The cabling for the 2550-2 HCP is described in detail in the Site Preparation Manual (see preface).

POWER REQUIREMENTS

Refer to the Site Preparation Manual (see preface) for 2550-2 HCP power requirements.

FUNCTIONAL CHARACTERISTICS

A block diagram of the 2550-2 HCP is presented in figure 1-1. This diagram shows the major functional units and subsystems of the basic 2550-2 HCP, the components and assemblies of these larger functional units, and the optional and required products which the user must specify by product number. The HCP macroinstruction set is compatible with the Control Data 1700 instruction set, and also includes a set of enhanced instructions.

The functional features of the HCP are listed in table 1-3.

COMMUNICATIONS PROCESSOR

Microprocessor

The communications processor (CP) is capable of either emulating the CDC 1700 Series processor or directly executing microcode. The CP contains arithmetic and logical elements, a series of general purpose registers, two files, a status mode register, 16 macro and 16 micro interrupts

TABLE 1-3. HCP FUNCTIONAL FEATURES

MAIN MEMORY
Capacity: 32,768 words (expandable to 65,536 words)
Word Length: 16 bits (plus a parity bit and a protect bit)
Read-access time: 550 nanoseconds
Memory addressing modes: 8
Memory word and region protection
Memory parity detection
Direct memory access (4 users)
External CPU access
Automatic interlaced refresh
MICROMEMORY
Capacity: 3,072 words (2,048 read/write; 1,024 read only emulator)
Word length: 32 bits
Read-access time: 168 nanoseconds
INTERRUPTS
Macrointerrupts: 16 (correspond to CDC 1700 interrupts)
Macrointerrupts: 16 (internal to this processor)
REGISTERS: 15
FILES
File 1: 256 words
File 2: 32 words

(with masks), a 1700 transform and emulator and 3072 32-bit words of memory plus necessary timing and control.

Also included is an internal data channel (capability for interfacing peripherals utilizing either an NCR MO5 protocol or a CDC 1700 AQ protocol. To provide timing for communications protocols, a real-time clock generates a continuous series of pulses at a 300 pps rate. Special algorithm hardware provides a capability for checking/generating a wide variety of LRC/CRC error check characters for messages in either direction. Integral to the CP are two sets of files accessible directly by the microprocessor.

As an enhanced emulative microprocessor, the CP executes microprograms; either those emulation routines residing in the 1024 fixed (read only) portion of micromemory or those real-time character-processing routines residing in the 2048 alterable read-write portion. When emulating, macroinstructions residing in main memory in 1700 format are emulated (not executed). The net effect is comparable; however, the implementation is totally different.

Multiplex subsystem operations generally are controlled by the real-time character-processing microcode. Communication between the emulator and the real-time microcode is via the microinterrupts.

The basic execution time for a micromemory instruction is 168 nanoseconds; however, longer times are required for some types of instructions. A typical main memory read-access requires 550 nanoseconds.

Maintenance Panel and Interface

The maintenance panel provides the following capabilities:

1. Starting the processor
2. Stopping the processor
3. Selecting any two registers (listed in table 3-2) for display
4. Modifying the contents of any register
5. Displaying the contents of main memory
6. Modifying the contents of main memory

7. Setting the macro or micro-breakpoint

NOTE

All of these capabilities also are available at the operator communications console.

Use of this panel is described in detail in section 3 of this manual.

Communications Console and Interface

The communications console can be either a CRT or TTY. In each case, the console serves as the operator's primary point of interface with the system. The communications console can function in either of two modes: I/O mode or panel mode. The mode of operation can be selected either by the software or by the operator using the communications console keyboard.

In the I/O mode, all communication between the console and the MP is routed only through the I/O TTY interface. The console thus operates as a standard peripheral device on the I/O bus.

In the panel mode, the console is used essentially as a maintenance panel. Operating modes can be switched from the communications console.

Cyclic Encoder

The primary function of the cyclic encoder is to compute cyclic redundancy checksum (CRC) and longitudinal redundancy checksum (LRC) characters in minimum time. These characters are used in error code generation and verification of both incoming and outgoing data to the terminals. Choice of encoding polynomials is controlled by software.

Main Memory

Main memory provides storage for macroprograms, tables, and data buffers needed to perform the special communications tasks of the HCP. Up to 64K words can be directly addressed by the processor and up to 128K words can be directly addressed by the host (e.g., CYBER 70/170) via the communications coupler.

Memory can be expanded up to 128K (92K if error correction features are included). Beyond 64K locations are addressed through the use of a memory paging system whereby memory is functionally divided into 2K pieces. Any combination of 32 registers constitutes the effective memory when either of two sets of paging are selected. As currently implemented, the 2550 operational software does not use the paging feature.

Each memory location contains a word-parity check-bit to provide one form of error control. Another form of error control is the memory protect system of which two types are provided. In the first, a protect bit is associated with each word location and can be set under program control to provide word-by-word protection. The second consists of an upper and lower bounds register used to protect on a contiguous block basis. The protect can be manually selected or disabled.

Main memory modules are interconnected to preclude simultaneous read operations from two or more locations in main memory. However, there are three ports for main memory. These ports are identified as input/output (I/O), direct memory access (DMA), and external. The I/O port serves as the normal access path for the CP to main memory. The DMA port provides access paths for the multiplex subsystem, and for direct memory-to-memory transfers from the host computer's peripheral processing unit (PPU) via the coupler. The external port is not used in single-processor application.

MULTIPLEX SUBSYSTEM

The multiplex subsystem contains the hardware, firmware, and software elements which provide the data and control paths for information flow between communications lines and user program software. Many of the line/protocol dependent functions that were performed by fixed hardware in previous systems are implemented in common alterable firmware/software allowing reduced development and recurring hardware costs for each line and protocol that is added to the communications system.

The primary task performed by the multiplex subsystem is to receive data from many communications lines and distribute the characters to line-related input buffers, and to obtain characters from line-related output buffers and distribute these characters to communications lines. The subsystem provides for special table-driven and dynamically controlled processing on each character as it is being distributed from a communications line. Circuit, modem and subsystem status are detected and transferred in the form of work demands to user programs for processing. Control information received from user programs in the form of commands to the subsystem is decoded and executed within one or more subsystem element.

The multiplexing scheme used in the multiplex subsystem design is based on a demand-driven multiplex loop concept. The multiplex subsystem gathers input data and status from, and distributes output data and control to, many communications lines on a real-time basis. The design provides an economical method of connecting many communications lines to a controlling processor while using a minimum of the processor resources to manage the mechanism.

The prime functional elements of the multiplex subsystem are as follows:

1. Multiplex loop interface adapter (MLIA) which provides the movement of data demands, data and supervision between the input and output loops and the controlling processor.

2. Loop multiplexers (LM) which provide access to the input and output loops by the CLAs.

3. Communications line adapters (CLAs) which provide character assembly/disassembly and communications line/modem interface.

Although not a part of the standard 2550-2 product, provisions are included for redundant operation of the multiplex subsystem.

Loop Multiplexer

The loop multiplexer performs the function of loading and unloading the input and output loops and communicating with the CLAs (up to 32 per LM). On input, demands are sensed, requests are queued and placed on the loop. Likewise, output data streams are examined for local delivery and corresponding output messages are given to the CLA.

The LM presents the data from the output loop to the CLA in parallel form. The LM also receives data in parallel form from the CLA, serializes this data and presents the data to the input loop. The LM operates at a speed of 20×10^6 bps (controlled by the MLIA). Operation of the LM is completely automatic and requires no direct program control.

Communications Line Adapter

The CLA adapts the varying requirements of the attached modems or terminals to the standard LM interface. Several CLA types are available, each with differing electrical and functional characteristics. In conjunction with parameter and mode control registers which are set under program control, the CLAs interface with a wide variety of modem or terminal types. Differences between modem types are accommodated by providing

a series of CLA-to-modem and CLA-to-terminal cables in which the standard CLA interface is transformed, as required, for the specific application. Refer to the corresponding CLA hardware reference manual for further information (see preface).

TAPE CASSETTE TRANSPORT AND CONTROLLER

The tape cassette transport is used for loading diagnostic programs. Also the transport is used to load the operating system software when the downline load capability is unavailable. The transport operates at a speed of 7.5 inches (19 cm) per second and recording density is 800 bits per inch (315 bits per cm). The transfer rate of data to and from the transport is 6 KHz. Rewind speed is approximately 50 inches (127 cm) per second. The transport utilizes Philips style cassettes.

The tape cassette controller serves as the access path for all communications between the transport and the CP. The controller provides input to, and receives output from, the transport at a rate of 750 characters per second (8 bits per character) in European Computer Manufacturers Association (ECMA) compatible format.

The tape cassette controller provides dual interfaces to the CP. The controller can be used as a standard peripheral, or as a dead-start device. The controller can be used to automatically provide input to the HCP as described in the Tape Cassette Autoload procedure in section 3.

OPTIONAL PERIPHERALS AND CONTROLLER

Should the user require a punched-card reader as a source of input, or should a line printer be required to provide output data, the 2571-1 Peripheral Controller must be added to the HCP. All communications between the CP and these peripherals are routed via the peripheral controller. The controller is capable of receiving data from a card reader at a maximum rate of 600 cards/min., and transmitting output data to a line printer at a rate of 1200 lines/min.

COMMUNICATIONS COUPLER

The primary purpose of the communications coupler is to provide a link between the host computer and the communications processor (CP). The coupler provides the means for direct memory-to-memory transfers between the host computer's peripheral processing unit (PPU) and the CP at a 1-MHz character rate. The unit is controlled by both PPU and CP software commands. The coupler also enables the PPU to start, clear and stop the CP. In addition to single word transfers, the coupler provides block transfers in either direction (e.g., downline load, upline dump) plus the ability to store and retrieve data via the standard buffer format.

MAIN DATA PATHS

The main data paths through the system are illustrated in figure 1-7. The microprogrammable processor (microprocessor) controls data flow between the host computer(s) and the terminals and/or modems. Data to and from the host computer(s) is routed via the communications coupler(s) to the microprocessor.

Data enroute to the terminals or modems is transmitted from the microprocessor to the MLIA, to the LM(s) via the output loop, and to the appropriate CLA interfacing the destination terminal or modem. Data enroute to the microprocessor is transmitted via the source terminal or modem to its associate CLA, to the LM, to the MLIA via the input loop, and to the microprocessor. Output data demands from a CLA are similarly carried inward via the input loop mechanism.

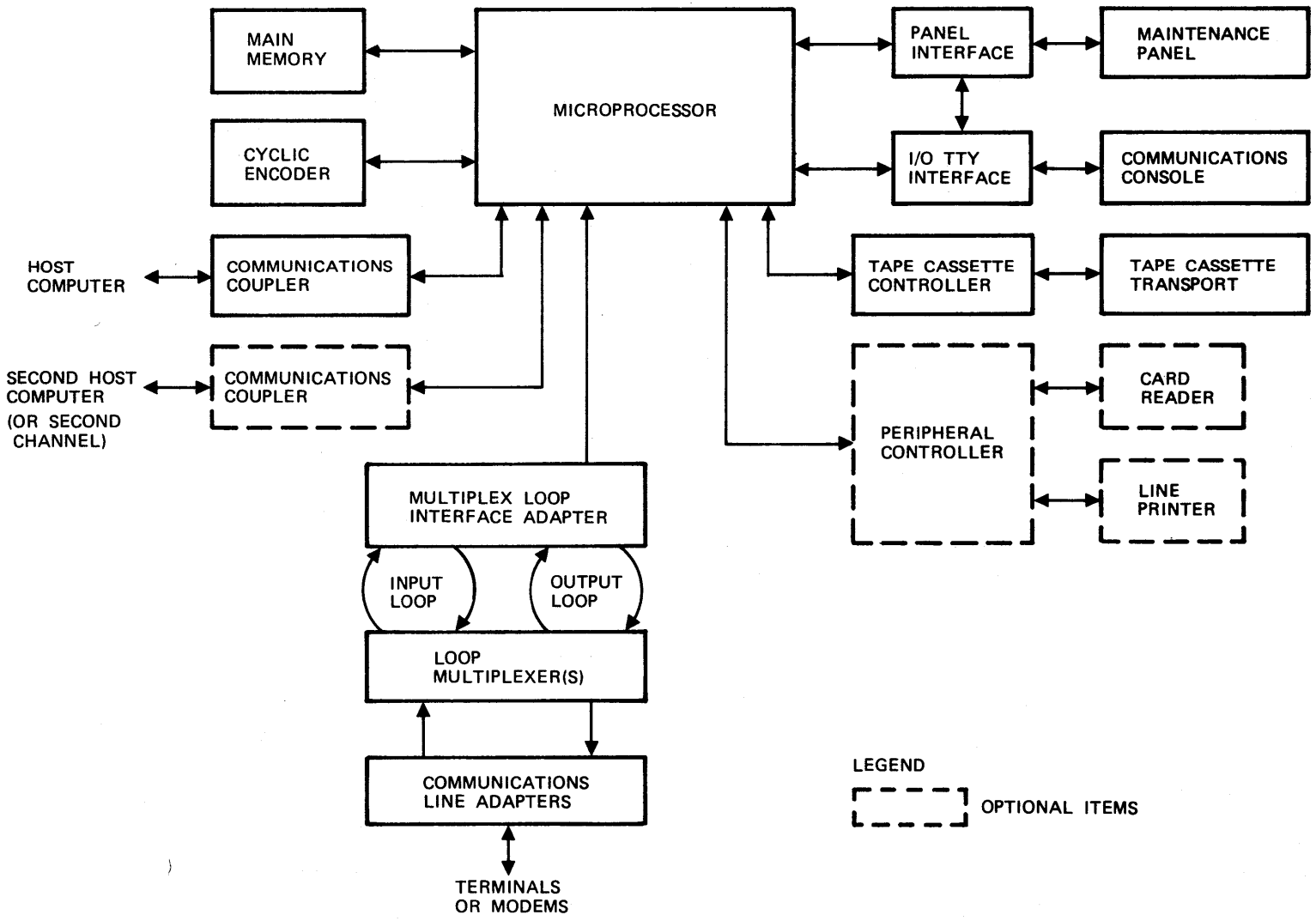


Figure 1-7. 2550-2 HCP Main Data Paths

The cyclic encoder receives instructions and data from the microprocessor to generate check characters. The cyclic encoder transmits these check characters (CRC and LRC) to the microprocessor as required by the protocol.

Main memory interfaces the communications processor through two ports: Input/Output (I/O) and Direct Memory Access (DMA). The microprocessor uses I/O, and the communications coupler(s) and multiplex subsystem use DMA.

The tape cassette transport serves as the primary source of operational and diagnostic programs for the HCP.

The tape cassette controller provides the access path for all communications between the transport and the CP.

All communication between the microprocessor and the maintenance panel is routed via the panel interface logic. The communications console can function in two modes: I/O and panel. In the I/O mode, all communications between the microprocessor and the communications console are routed via the I/O TTY I/F logic. In the panel mode, input data is routed from the console, to the I/O TTY I/F logic, and through the panel I/F logic to the microprocessor; conversely, output data from the microprocessor is routed to the panel I/F logic, and through the I/O TTY I/F logic to the console. The console thus acts as either a standard I/O peripheral device or as a maintenance panel (selectable under software control).

FUNCTIONAL DESCRIPTIONS

2

Functions of the major components of the 2550-2 HCP as shown in figure 1-2 are described in this section. The major components are as follows:

1. Communications Processor
2. Main Memory
3. Multiplex Subsystem
4. Tape Cassette Transport and Controller
5. Communications Coupler
6. Peripheral Controller

COMMUNICATIONS PROCESSOR

The communications processor (CP) provides buffer storage for input and output data, storage and execution of various firmware and software programs, and provides the hardware interfaces for both the host computer and the multiplexing subsystems.

The five functional units of the CP are as follows:

1. Microprocessor
2. Maintenance Panel
3. Maintenance Panel Interface
4. I/O TTY Interface
5. Cyclic Encoder

MICROPROCESSOR

The controlling element within the CP is a small-scale, stored-program, parallel-mode digital computer called a microprocessor (MP). See figure 2-1. The MP utilizes firmware (microinstructions) and software (macroinstructions) to control the multiplexing subsystem, communications coupler, plus peripheral devices. A Direct Memory Access (DMA) channel accepts both input and output data traffic from the host computer plus input traffic from the multiplexing subsystem. An Internal Data Channel (IDC) serves as the main access path for the multiplexing subsystem and the various peripherals. The IDC handles data traffic in both directions, and operates under microinstruction control.

The HCP contains two separate memory units which are identified as main memory and micromemory. Micromemory has smaller capacity, but faster access and cycle times. Each micromemory location contains a 64-bit double-word.

Main memory stores the macroinstruction program (and data), and micromemory stores the microinstruction program. Macroinstructions are read from main memory sequentially by a portion of the microprogram. The macroinstruction is then decoded into a series of microoperations by a combination hardware/controlware (firmware) technique. The hardware portion of the macroinstruction decode is called a transform; the controlware (firmware) portion is called an emulator (in the case of standard 1700-type instructions). The 1700 emulator consists of a series of fixed microinstructions contained in a Read Only Memory (ROM). The transform/emulator process causes the microprogram to form program branches, sets parameters, and performs arithmetic and logical operations.

A transformed/emulated macroinstruction is performed as if it were one or more standard stored microinstruction(s). Numerous microinstructions may be required to execute one macroinstruction. Macroinstructions can be decoded either with or without the use of the transform feature; however, the use of transforms significantly reduces the number of microinstructions required and increases overall performance.

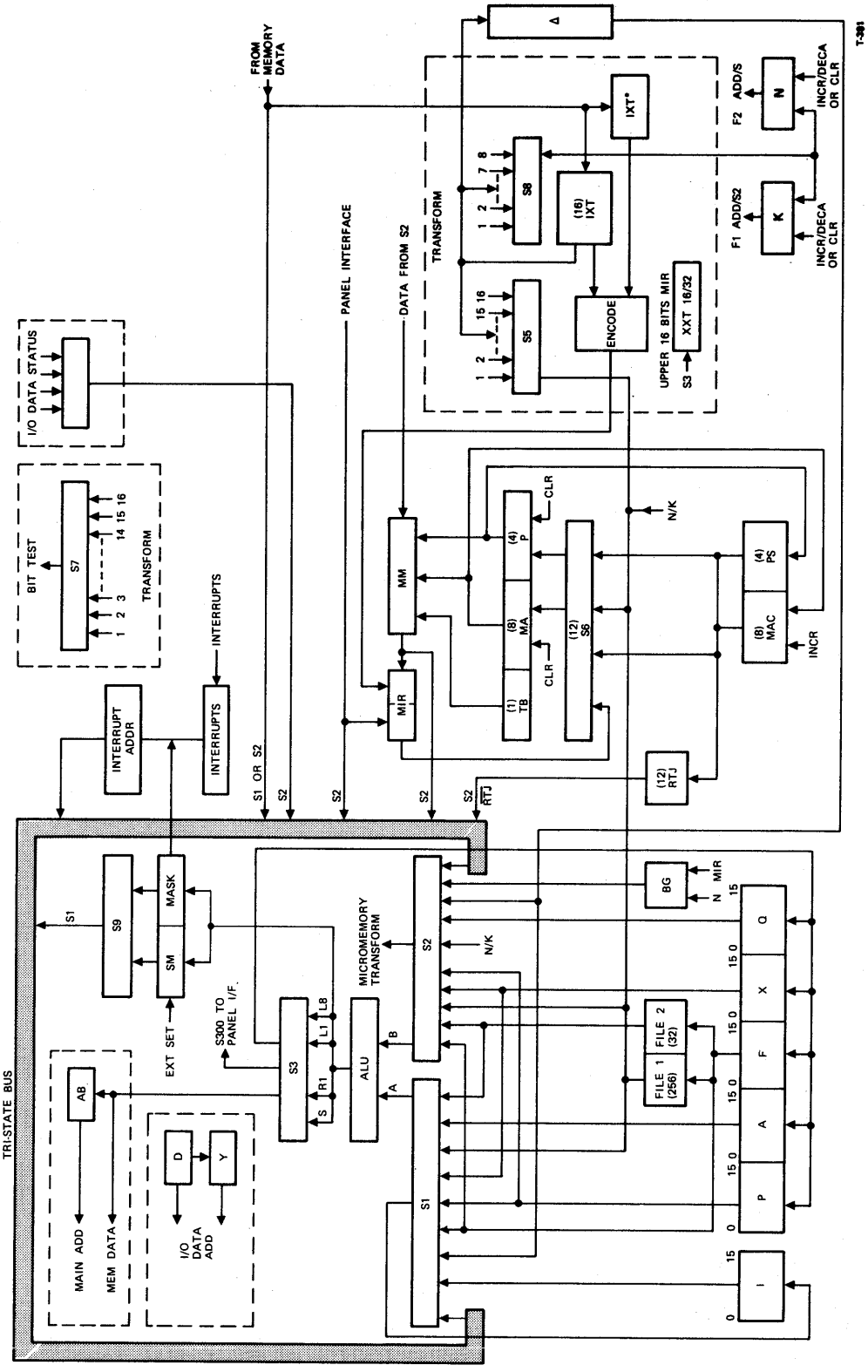


Figure 2-1. Detailed Microprocessor Block Diagram

Arithmetic operations are performed by the Arithmetic and Logical Unit (ALU). The two inputs (A and B) are connected to a variety of input sources (e.g., I, P, A, F, X, and Q registers, Files 1, 2, bit generator, N/K register, Δ, tristate bus) via selectors S1 and S2, respectively. Output of the ALU (as is, shifted right one bit, shifted left one bit, or shifted left eight bits depending on selector S3) then appears in the designed general-purpose registers (P, A, F, X, Q). Files 1 and 2 are addressed by registers K and N, respectively.

Control is exercised through microinstructions from the micromemory (MM). Addressing is controlled by a four-bit page, eight-bit memory address, and a one-bit, T-bit from the current microinstruction.

Operating modes are established by setting/clearing bits in the Status Mode (SM) register. Interrupts are selected via priority logic reading through an interrupt mask register.

Micromemory

The MP micromemory is an integrated circuit, random access, semiconductor memory which contains two sections:

1. Nonviolatable/alterable Read Only Memory (ROM)

2. Alterable Random Access Memory (RAM)

NOTE

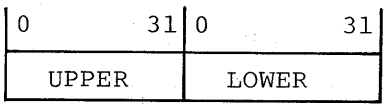
The terms RAM and ROM are used since this is their commonly referenced designation. A more accurate statement is that the RAM is a randomly-accessed read/write memory, and the ROM is a randomly-accessed read-only memory.

READ ONLY MEMORY (INCLUDING 1700 EMULATOR)

Instructions can be read only from the ROM section of micromemory.

There are two pages of ROM. The storage capacity of each page is 512 instruction words of 32 bits each. However, it should be noted that the words are coupled in pairs of 64 bits; thus the capacity of each page is sometimes referred to as 256 instruction word-pairs; and the total two-page ROM capacity is 512 instruction word-pairs of 64 bits.

Each instruction word-pair contains upper and lower addressing as shown below.



ROM contains the macroinstruction set (1700) emulator microprogram. For a detailed description of the interrupt system, program protection and parity error processing provided by the 1700 Emulator, refer to Chapter 4 of the 1714 Computer System Reference Manual listed in the Preface. Although ROM is functionally a part of micromemory, it is physically located on the transform circuit card.

RANDOM ACCESS MEMORY

Instructions can be written into, as well as read from, the RAM section of micromemory.

RAM is used in applications that require the MP to be reprogrammed and during program development. RAM can be loaded from an external device by writing into main memory (e.g., from coupler, cassette), and then by writing into RAM from main memory.

There are four pages of RAM. The storage capacity of each page is the same as that previously described for ROM.

The total capacity of RAM is 1024 instruction word-pairs of 64 bits. The microprograms that control the multiplexing subsystem are stored in RAM micromemory.

MICROMEMORY CONTROL

Control of the micromemory is provided by the following registers:

1. Page/Micromemory Address Register
2. Memory Address Counter
3. Page Storage Register
4. Microinstruction Register
5. Return Jump Register

Page/Micromemory Address Register

The micromemory is addressed by the Page/Micromemory Address (P/MA) register which is divided into two separate units: P and MA. The P portion of the P/MA register provides page control; it consists of four bits, which can specify as many as 16 pages of micromemory (each page consists of 256 instruction word-pairs, or a total of 512 microinstructions). The MA portion of the register consists of 8 bits, which specify one of the 256 microinstruction pairs within the page that is to be the source for the next pair of microinstructions. The T-field of the current microinstruction is used to select either the upper or lower microinstruction of a pair as the next microinstruction to be executed. Microinstruction sequencing is designed to prevent automatic overflow of addressing from the MA portion to the P portion of the register. Any transfer of control between pages is initiated by a Page-jump operation, MA-transform operation, or clear-page operation.

The P portion of the P/MA register can be set by microinstructions from the Page Storage (PS) register or the Return Jump (RTJ) register, or by an output from the transform module. The MA portion of the register can be set by microinstructions from the Memory Address Counter (MAC) or the RTJ register, or by an output from the transform module. The MA portion of the register has no sequencing capability; this is provided by the MAC. The combination N and K registers can provide addresses to the P/MA register for micromemory Read or Write operand references.

Memory Address Counter

The Memory Address Counter (MAC) is an 8-bit device which is used to determine the next location within a page which follows the current location specified by the MA register. In operation, the contents of the MA register are transferred to the MAC at each micromemory reference; then the contents of the MAC are incremented by one, to point to the next location. Depending on the sequencing operation specified in the next microinstruction, the MAC may or may not be used to obtain the next microinstruction. Sequencing of the MAC is such that location 0 within a page follows location 255 on the previous page.

Page Storage Register

The Page Storage (PS) register is a 4-bit holding register which is used to determine the page for the next instruction. In operation, the contents of the P portion of the P/MA register are transferred to the PS register at each micromemory reference. Depending on the sequencing operation specified in the microinstruction, the PS register may or may not be used to obtain the next microinstruction.

Microinstruction Register

The Microinstruction Register (MIR) is a 32-bit register which is used to hold a microinstruction during execution. Microinstructions are normally entered into the MIR from micromemory; either the upper or lower 32 bits of the contents of the micromemory location are gated to the MIR based on the value of the test bit (determined during the preceding microinstruction). A test bit of zero specifies the upper microinstruction, and a test bit of one specifies the lower microinstruction.

Microinstructions can also be transferred into the MIR via the mainte-

nance panel interface by the HCP system operator. A transform design option allows partial microinstructions to be transferred from the transform module to the upper 16 bits of the MIR for specialized applications.

Return Jump Register

The Return Jump (RTJ) register is a 12-bit device which is used to store the location of the next microinstruction pair at any time, when specified by a microinstruction. When this operation is specified, the contents of MAC are incremented, and the contents of the PS register and the MAC are stored in the RTJ register. The contents of the RTJ register are unchanged until the next command is given to save a new address. This saving of the next instruction pair location is independent of any actual transfer of control. The output of the RTJ register can be gated to the P/MA register to perform the return operation, or may be read into the organization of the MP through selector S2.

Transform

The transform feature provides the microprogram with the capability of selecting any pattern of bits from the data transmission paths of the MP. These bit patterns are used to form micromemory addresses which sequence the microprogram. The bit patterns are also used to set the contents of the K and N registers, and the upper 16 bits of the MIR.

The 1700 transform printed circuit card provides the following functions: Selectors 5, 7 and 8; IXT and IXT*, Transform Encode; and Transform Micromemory. This card contains memory protect violation logic, decimal correction logic, and a delta field generator for the lower eight bits of each data word.

SELECTOR S5

Selector S5 is an 8-bit wide, 2-to-16-position selector which is used to form micromemory addresses. A maximum of 16 different micromemory address (MA) instructions can thus be specified. The MA which is transformed specifies one of the 256 64-bit microwords within a page. The T-field code specifies which 32-bit microinstruction is to be executed. If SM207 is set to one, the transform address will be to the page denoted in the S1 field of the microinstruction. If SM207 is set to zero, the transform address will reside in the same page as the microinstruction currently being executed. The lower (least significant) eight bits of the output of S2 will represent MA transform number 7 (j=0111 in C field).

SELECTOR S8

Selector S8 is an 8-bit wide, 2-to-8 position selector which is used to select a source for loading either the K or N registers. Two of the sources are fixed; KN transform 6 is the output of the lower (least significant) eight bits of the MIR; KN transform 0 is the lower eight bits of S2.

MACROINSTRUCTION REGISTER

A specialized transform of the upper (most significant) 16 bits of the MIR register is implemented by the macroinstruction register (identified as IXT). This feature allows the upper 16 bits of the MIR to be loaded directly with information which has been encoded from the IXT register. This provides control of the MP arithmetic function during macroemulation. Read Next Instruction (RNI) cycles are also provided for special applications.

SELECTOR S7

Selector S7 is a 16-bit wide, 2-to-16-position selector. This selector is used to test a specific condition in order to determine which of the microinstructions in the next pair should be executed. This is accomplished by placing a Bit Upper (BTU) command in the T field of the microinstruction. The lower five bits of the C field (j) determine which of the 32 bits is to be tested. If the tested bit is a one, the upper microinstruction is executed; otherwise, the lower microinstruction is executed.

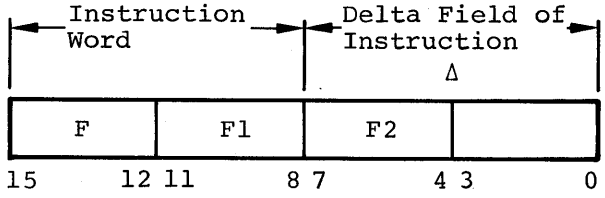
TEST BITS

Bits that can be tested include all of the bits which are normally available for use in creating transforms, and bits associated with any specialized I/O logic. The following MP outputs are available for monitoring by the transform module:

1. S1
2. S2
3. SM registers
4. Data registers on specialized transform module
5. MIR
6. Main memory read data bus
7. Bits 19 and 24 through 31 of micromemory read data

TRANSFORM OPERATION

The format for the transform operation is shown below:



The F field is normally utilized for the operation code. If F=0, the operation code will be placed in the F1 field.

The contents of the Arithmetic and Logic Unit (ALU) are loaded into the Instruction (I) register. Data is provided, four bits at a time, via the transform unit so that it can be read by microprogram.

If field F1=0001, a skip instruction is recognized; in this case, the contents of field F2 will indicate how far the skip is to be executed. This action also increments the P register. Control of the ALU is provided by micromemory.

ROM 1700 EMULATOR

As previously stated in the micromemory subsection, the ROM 1700 Emulator microprogram is physically located on the transform circuit card. It should also be noted that this portion of micromemory is a programmable ROM (PROM), which is "burned-in" at the time of manufacture, and cannot be changed without replacing the integrated circuit component. Some machines may contain non-programmable memory units (ROM) in which the pattern is installed at the time of semiconductor manufacture.

Arithmetic and Logical Unit

The Arithmetic and Logical Unit contains the A, F, I, P, Q, and S registers, and also provides S1, S2 and ALU Look-Ahead functions. The ALU provides File 1, consisting of 16 bits in each of 256 locations, and File 2, which has 16 bits in each of 32 locations.

The ALU provides the basic capabilities for arithmetic and logical operations within the MP. The unit combines two input words: the A input (provided by S1) and the B input (provided by S2). These two inputs are combined according to the function code specified in the microinstruction; the result is delivered

simultaneously to the output of the ALU for use by any of the following:

1. a specified destination register
2. SM register
3. MASK register
4. memory bus
5. panel interface

It is also possible for the MP to ignore the output of the ALU during a given operation. The results of ALU operations, i.e., sign, zero, and magnitude, are available to the test-bit logic for instruction sequencing.

Arithmetic operations can be performed using either one's or two's complement arithmetic; the desired mode is selected via the microprogram, which controls the states of the applicable bits in the SM register.

DATA TRANSFER

The data transfer logic of the MP provides for storing data in six working registers and two files. This data can then be selected by one of two selectors for further processing through the ALU. The ALU outputs can be transferred back to one of the registers, or out of the data transfer logic to control external circuits.

The six working registers serve as primary data devices, and are identified as the I, P, A, F, X, and Q registers. The two files serve as storage registers and are identified as file 1 and file 2. The two selectors are identified as S1 and S2. Their function is to select the outputs of the above devices for application to the ALU or other logic. A bit generator (BG) is also utilized in the data transfer logic. The following paragraphs describe the function of these 11 units in more detail.

I Register

The I register is a word-length register which is used primarily to hold the emulated software instruction being executed by the MP. It is also available as an input to selector S1 and, therefore, to the A input of the ALU.

Data is entered into the I register from the output of S1. This connection is provided so that data can be transferred from memory directly to the I register; in addition, some other operation can be simultaneously performed on the memory data, or the memory data can be to another register through the ALU.

P Register

The P register is a word-length register which receives data from the ALU, and its output is also provided via S1 to the A input of the ALU. This is a general purpose register; however, it normally is used to contain the software instruction counter for the emulation of a host computer.

A Register

The A register is a word-length, general purpose register which receives data from the ALU and provides an output via S1 to the A input of the ALU. This unit is mechanized as a shifting register which can be shifted left or right without ALU control. The A register can also be combined with the Q register to form a double-length shifting register, which can also operate independently of the ALU.

F Register

The F register is a word-length, general purpose register which can be read into either the A or B input

of the ALU. This register is also used as the file entry register; it contains the information written into file 1 or file 2 when these files are used as the destination of an ALU operation.

X Register

The X register is a word-length general purpose register which can be read into either the A or B input of the ALU.

Q Register

The Q register is a word-length, general purpose register which receives data from the ALU and provides an output via S2 to the B input of the ALU. The Q register is mechanized as a shifting register; it can be shifted left or right, in conjunction with the A register, without the use of ALU control.

File 1

File 1 comprises 256 general purpose, word-sized registers which are addressed by the contents of the K register. On demand, the output of the addressed file is delivered via S1 and S2 to the A and B inputs, respectively, of the ALU. An SM bit is also provided for selecting either the output of file 1, or the output of the transform module, as the input to S1 or S2.

File 2

File 2 is a 32-word file which is addressed by the lower five bits of the N register. It is used primarily as a source of constants, but can also be used as a general purpose, word-sized register which delivers its output via S1 and S2 to the A and B inputs, respectively, of the ALU.

Selector S1

Selector S1 provides for the selection of one of nine inputs for delivery to the ALU, or for delivery to the I register if directed by microinstruction. The selector can also provide data for transforms.

The nine inputs to selector S1 are:

1. Input bus
2. I register
3. P register
4. F register
5. File or external input (transform)
6. X register
7. File 2
8. A register
9. File 1

The input bus provides for delivery of data from the following sources to S1:

1. Data from the external main memory
2. Output of the SM registers or the interrupt mask registers
3. Output of the cyclic encoder

Selector S2

Selector S2 provides for the selection of one of nine inputs for delivery to the B input of the ALU. In addition, this unit provides for the transfer of information to the transform module and to micromemory.

The nine inputs to selector S2 are:

1. F register
2. File 1 or external input (transform)
3. X register
4. File 2
5. Q register
6. N and K registers
7. Bit generator
8. Input bus
9. P register

The input bus provides for submultiplexing of data for input to S2. The input bus receives data from the following sources:

1. Main memory
2. Interrupt address
3. Micromemory
4. I/O system, maintenance panel interface
5. RTJ register
6. I/O data status

Bit Generator

The bit generator (BG) can provide one bit, at any bit position in a word, as input to the B side of the ALU. Bits are numbered from left to right as bits 0 through 15. The bit generator is controlled either by a microinstruction (bits 27 to 31) or by the lower five bits of the N register. The selection of control method for the BG is determined by setting a bit in the status mode (SM) register.

NOTE

Three bit-numbering plans are used within the communications processor. Basic MP registers, selectors, bit generators and microinstructions are numbered left to right with the left-most bit numbered as zero. Typical register numbering is of the form: 0 → 7, 0 → 15, 0 → 31, or 0 → 63, as appropriate.

As seen at the software/firmware level, however, standard 1700-type numbering is used. Thus main memory contents, register contents, etc., are usually numbered right to left, 15 ← 0. Wherever possible in this document, the 1700 numbering plan will be used.

The 1700-type numbering plan is used with all CDC A/Q-channel peripherals. Devices using the NCR-type numbering plan are identified by a slightly different version in which the bits are numbered right to left, 16 ← 1.

Unless otherwise noted, 1700-type numbering (15 ← 0) can be assumed for all macro-level operations, and all machine micro-level operations use basic MP (0 → n) numbering.

Status Mode Interrupt

The status mode interrupt (SMI) logic consists of two status mode registers, two interrupt holding registers, two interrupt mask registers, two priority encoders, plus associated selectors and other hardware elements.

STATUS MODE REGISTERS

The status mode (SM) registers are designated as SM1 and SM2. Each register contains 16 bits, and the bits are numbered from left to right as follows:

Register	Bit Numbers
SM1	SM100 to SM115
SM2	SM200 to SM215

The bit assignments for all 32 SM bits are listed in table 2-1.

The SM bits can be divided into three general functional classifications as follows:

1. Operating Mode Bits
2. Event Occurrence Bits
3. Control and Timing Driver Bits

Operating Mode Bits

The operating mode bits are listed in table 2-2. The following paragraphs describe the functions of the operating mode bits.

One's Complement Arithmetic

If status mode bit SM101 is set to one, the ALU performs addition and subtraction in one's complement arithmetic. If SM101 is set to zero, the ALU performs these operations in two's complement arithmetic.

TABLE 2-1. STATUS MODE REGISTER BIT ASSIGNMENTS

Bit	Function
SM100	Not used (double precision)
SM101	One's Complement Arithmetic
SM102	Enable Bit Generator Input from N Register
SM103	Not used (split adder)
SM104	Macro Breakpoint Occurred (use with INT31/)
SM105	1700 Protect Fault Occurred
SM106	Enable Macro Interrupt System
SM107	Not used (enable decimal arithmetic)
SM108	Main Memory Parity Error Occurred
SM109	Enable Micromemory Halt
SM110	Overflow
SM111	Enable File 1 (Read/Write)
SM112	Not used (enable binary overflow)
SM113	Not used (enable R/W MM via transform)
SM114	Delay Enabling Macro Interrupts
SM115	MLIA Busy
SM200	Enable ADT Mode
SM201	Strobe or Read Data from MO5 Device
SM202	Write Data to AQ Device
SM203	Terminate I/O Transfer
SM204	Enable Autoload
SM205	Enable MUX Subsystem Priority 3 Interrupt
SM206	Not used
SM207	Enable page selection with transform
SM208	Not used
SM209	Enable the unprotected instruction followed by a Protected Instruction Check
SM210	Write Data to Panel Device
SM211	Read Data to Panel Device
SM212	Write Data to FCR Register
SM213	Enable 1700 Enhanced Transform
SM214	Enable Console Control
SM215	Enable Macro Instruction Run

TABLE 2-2. OPERATING MODE BITS

Bit	Function
SM101	One's Complement Arithmetic
SM102	Enable Bit Generator Input from N Register
SM106	Enable Macro Interrupt System
SM109	Enable Micromemory Halt
SM111	Enable File 1 (Read/Write)
SM114	Delay Enabling Macro Interrupts
SM200	Enable ADT Mode
SM204	Enable Autoload
SM205	Enable MUX Subsystem Priority 3 Interrupt
SM207	Enable Page Selection with Transform
SM209	Enable the Unprotected Instruction Followed by a Protected Instruction Check
SM213	Enable 1700 Enhanced Transform
SM214	Enable Console Control
SM215	Enable Macro Instruction Run

Enable Bit Generator Input From N Register

If status mode bit SM102 is set to one, the bit generator (BG) is controlled by the lower five bits of the N register. If SM102 is set to zero, BG is controlled by the lower five bits of the microinstruction.

Enable Macro Interrupt System

If status mode bit SM106 is set to one, the macro interrupt system of the MP is activated, and the interrupt (INTU) test logic can examine the interrupt system.

If SM106 is set to zero, the interrupt system is disabled and the INTU test logic always receives a reply of "no interrupt present".

Enable Micromemory Halt

If status mode bit SM109 is set to one, any microinstruction with a HALT code in the S field stops the operation of the MP on completion of the microinstruction. If SM109 is set to zero, the HALT code in the S field of any microinstruction is ignored.

Enable File 1 (Read/Write)

If status mode bit SM111 is set to one, the output of File 1 is delivered to Arithmetic and Logical Unit (ALU). This overrides any selection of an external source.

If SM111 is set to zero, the output of File 1 is disabled as an input source and the transform module enabled at the same position as defined for File 1.

Note that this mode bit must be set to one in order to read/write File 1.

Delay Enabling Macro Interrupts

If status mode bit SM114 is set to one, the enabling of macro interrupts is delayed until the next GITMAK/XT

microinstruction rather than enabling interrupts on the first GITMAK/XT. If SM114 is set to zero, the enabling of macro interrupts is not delayed. This bit is utilized by the EIN macroinstruction (part of the emulator).

Enable ADT Mode

If status mode bit SM200 is set to one, the Automatic Data Transfer (ADT) mode of operation is enabled. If SM200 is set to zero, the ADT mode is disabled.

Enable Autoload

If status mode bit SM204 is set to one, the autoload mode is selected for the console interface, and whichever deadstart device is READY begins loading information. If SM204 is set to zero, the autoload mode is disabled at the console interface.

Enable MUX Subsystem Priority 3 Interrupt

If status mode bit SM205 is set to one, the multiplex subsystem priority 3 interrupt is enabled. If SM205 is set to zero, this interrupt is disabled.

Enable Page Selection with Transform

If status mode bit SM207 is set to one, an MA transform can be executed across a page boundary by setting the desired page in the S field of the microinstruction. In this case, normal S field operations are disabled.

If SM207 is set to zero, an MA transform is limited to the page in which the microinstruction containing the MA transform command resides. The S field of the microinstruction is decoded for normal operation.

Enable the Unprotected Instruction Followed by a Protected Instruction Check

If status mode bit SM209 is set to one, an unprotected instruction, followed by a protected instruction check, is enabled. If SM209 is set to zero, an unprotected instruction, followed by a protected instruction check, is disabled.

Enable 1700 Enhanced Transform

If status mode bit SM213 is set to one, a 1700 enhanced transform is enabled. If SM213 is set to zero, a 1700 enhanced transform is disabled.

Enable Console Control

If status mode bit SM214 is set to one, the output of the console interface is pre-enabled to the microinstruction register (MIR). If the MP is Master-Cleared, the console interface is enabled to the MIR.

If SM214 is set to zero, the output of the micromemory is pre-enabled to the MIR.

Enable Macro Instruction Run

If status mode bit SM215 is set to one, the running of macroinstructions is enabled. If SM215 is set to zero, the running of macroinstructions is disabled.

Event Occurrence Bits

The event occurrence bits are listed in table 2-3. The function of these status mode bits is to store a bit signifying the occurrence of an event. These bits typically serve as a source for a corresponding interrupt line.

TABLE 2-3. EVENT OCCURRENCE BITS

Bit	Event Occurred
SM104	Macro Breakpoint Occurred (use with INT31/)
SM105	1700 Protect Fault Occurred
SM108	Main Memory Parity Error Occurred
SM110	Overflow
SM115	MLIA Busy

Control and Timing Driver Bits

The control and timing driver bits are listed in table 2-4. These bits drive control and timing lines (under firmware control) for I/O operations.

TABLE 2-4. CONTROL AND TIMING DRIVER BITS

Bit	Function
SM201	Strobe or Read Data from MO5 device
SM202	Write Data to AQ device
SM203	Terminate I/O Transfer
SM210	Write Data to Panel device
SM211	Read Data to Panel device
SM212	Write Data to FCR register

Of the 32 bits in the two SM registers, eight are designated as flag bits, and the remaining 24 are designated as external bits.

Flag Bits

Flag bits are set and reset by Set Flags (SETF) and Clear Flags (CLRF) commands, respectively, in a microinstruction. The flag bits are not cleared by a processor master clear, and no external input is provided to these bits.

Sixteen flag bits are available, although only eight are presently addressed by SM register bits as shown in table 2-5.

TABLE 2-5. STATUS MODE REGISTER FLAG BITS

Flag Bit Nos.	SM Bit Nos.
0 to 3	SM100 to SM103
4 to 7	(not used)
8 to 11	SM200 to SM203
12 to 15	(not used)

External Bits

The 24 external SM bits are set by external signals, and are cleared by a master clear signal from the MP.

INTERRUPT HOLDING REGISTERS

The interrupt holding registers are designated as I1 and I2. Register I1 provides internal microinterrupts; register I2 provides external macrointerrupts. Each register contains 16 bits, and the bits are numbered from left to right as follows:

Register	Bit Numbers
I1	I100 to I115
I2	I200 to I215

The priority levels of the interrupt bits are in numerical order. Bit I100 has the highest priority in register I1; bit I115 has the lowest priority. Similarly, bit I200 has the highest priority in register I2, and bit I215 has the lowest.

The bit assignments for registers I1 and I2 are listed in table 2-6.

TABLE 2-6. INTERRUPT BIT ASSIGNMENTS

Bit	Function
I100	MLIA (Output Data Demand)
I101	Communications Console (TTY or CRT)
I102	MLIA (Line Frame)
I103	(unassigned)
I104	2571/2570 Line Printer Controller
I105	(unassigned)
I106	(unassigned)
I107	Tape Cassette Controller
I108	Real Time Clock
I109	(unassigned)
I110	(unassigned)
I111	2571/2572 Card Reader Controller
I112	1700 Emulator
I113	1700 Emulator
I114	1700 Emulator
I115	1700 Emulator
I200	Power Fail/Memory Parity
I201	Communications Console (TTY or CRT)
I202	MLIA (Error)
I203	Multiplex Subsystem Priority 3 (source is SM205)
I204	2571/2570 Line Printer Controller
I205	CYBER Coupler 2 (2558-1)
I206	CYBER Coupler
I207	Tape Cassette Controller
I208	Real Time Clock
I209	1732/608-609 Mag Tape Controller (QSE)
I210	1740-1742 Line Printer Controller (QSE)
I211	2571/2572 Card Reader Controller
I212	MLIA (Same as I100)
I213	MLIA (Same as I102)
I214	(unassigned)
I215	Macro Breakpoint

MASK REGISTERS

The mask registers are designated as M1 and M2. Each register contains 16 bits, and the bits are numbered from left to right as follows:

Register	Bit Numbers
M1	M100 to M115
M2	M200 to M215

The mask register bits correspond to the interrupt lines having the same number.

PRIORITY ENCODERS

The function of the two priority encoders is to look at interrupt lines (thru the two mask registers) and produce a number equal to the highest priority interrupt currently active. This interrupt will then be serviced by the processor.

Real-Time Clock

The real-time clock is designed to appear as a 1700 peripheral to the macrolevel software. Two functions are available to the macrolevel program: Enable Limit Interrupt and Disable Limit Interrupt. Two status bits are also available to the macrolevel program: Limit Interrupt and Lost Count.

The limit interrupt being received by a macrolevel program is dependent on the appropriate Mask register bit being set and the macro interrupt enabled as with all other 1700 peripherals.

Microlevel code is capable of receiving a data interrupt from the real-time clock every 3-1/3 milliseconds (derived from a crystal oscillator). This interrupt is enabled anytime that the corresponding mask bit is set.

The determination of when the microcode generates the macrolevel interrupt is dependent on the emulator. To generate the limit interrupt, it is only necessary for the microinstruction to set SM bit Terminate before clearing the data interrupt. If this is done, the data interrupt will be set, providing the limit interrupt has been enabled by the microcode.

The data interrupt is cleared by setting SM bits Auto-Data and Strobe with the I/O Y-register. In the case where it is desired to clear the data interrupt without generating the Limit interrupt, it is necessary to clear SM bit Terminate before clearing the data interrupt.

If the data interrupt has not been cleared before another 3-1/3 millisecond count occurs (and the limit interrupt has been selected), 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.

Breakpoint

There are two types of breakpoint (BP): micro and macro. If bit 12 of the function control register (FCR) is set, micro BP is selected. If bit 12 is clear, macro BP is selected.

Three types of macro BP are selected by bits 14 and 15 of the FCR as follows:

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

A macro BP occurs if the breakpoint register is equal to the main memory address and the select conditions are met.

For micro BP, 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 comparison of FCR bit 13 and the upper/lower selector is not required.

Parity

Each main memory word consists of 19 bits as follows:

16 data word bits,
1 data word parity bit,
1 protect bit, and
1 protect parity bit.

The memory parity error function is used to indicate that either a word or protect parity error has occurred.

A word parity error indicates that one of the memory word bits is wrong or the parity bit is wrong. Word parity error has no effect on memory operation other than setting the parity error line.

A protect parity error (PPE) indicates that either the protect bit or the protect parity bit is wrong. When a PPE occurs and CPU PROTECT is true during a write operation, actual protect bit (DO18) is stored back in memory protect bit (DI18) and the complement of the actual protect bit is stored in memory protect parity bit (DI19). When a PPE occurs and CPU PROTECT is false, the actual bits (DO18 and DO19) are both restored in DI18 and DI19 respectively.

MAINTENANCE PANEL

The maintenance panel can be used to enter operational and diagnostic program commands and also displays output data (via LED indicators). Its purpose is to enable operator control of the system at the HCP itself, rather than from the communications console which may be remotely located.

Since the maintenance panel consists primarily of a number of controls and indicators, the functional description for this unit is included in Section 3, Operating Instructions. The Controls and Indicators Subsection describes the function of each of the components of the maintenance panel; and the Operating Procedures Subsection gives specific examples of their use. Tables 3-1 and 3-2, in Section 3, define the function control register (FCR), which is closely related to the functional description of the maintenance panel.

MAINTENANCE PANEL/COMMUNICATIONS CONSOLE INTERFACE

All communications between the microprocessor (MP) and the maintenance panel are routed through the panel interface (see figure 2-2).

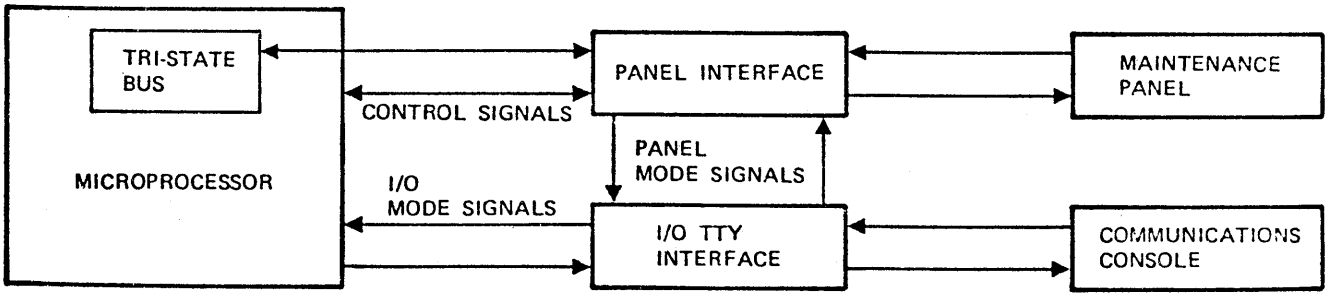


Figure 2-2. Maintenance Panel/Communications Console Interface Block Diagram

The communications console serves as the operator's primary point of interface with the system. Either a teletypewriter (TTY) or cathode ray tube (CRT) conversational display can be used as the communications console, provided that the selected device has two-way, simultaneous, serial (full duplex), ASCII, odd parity characteristics. The communications console interface is RS232-compatible.

The communications console can function in either of two modes: I/O and panel.

In the I/O mode, all communication between the console and the MP is routed only through the I/O TTY interface, and the console operates as a standard peripheral device.

In the panel mode, the console essentially functions as a maintenance panel; and in this case, input from the console is routed through the I/O TTY interface to the panel interface, and then to the MP. Output from the MP is routed via the panel interface, through the I/O TTY interface, and on to the console.

Panel Interface

The panel interface provides all of the control functions commonly associated with digital computer systems. These functions can be performed equally well through the maintenance panel or the communications console (operating in panel mode).

The panel interface is used to develop MP instructions, and it also directs execution of these instructions by the MP. The instructions are transferred directly to the microinstruction registers. The panel interface develops four basic MP instructions:

1. ALU Panel I/F
2. P or K Register Increment
3. Store Operator Data in Selected Destination
4. Read Operator Selected Source

The panel interface also provides the communications interface for the following:

1. TTY Interface Port (RS232)
2. CRT Display Port
3. TTL Port (compatible with serial ASCII TTY)
4. Function Control Register (32 bit)
5. Breakpoint Register
6. Console Control Codes (H, I, J, K, L, M, N) Interpretation (see FCR description, Section 3)

I/O TTY Interface

The I/O TTY Interface includes the TTY interface, and the I/O control for both CDC A/Q-type and NCR M05-type operations.

Either a TTY or CRT display can be attached provided that it functions as an asynchronous RS232-compatible device. A device having similar functional characters but operating at TTL levels, such as the tape cassette (dead-start portion), can also be connected. ASCII character rates up to 9600 bps can be accommodated.

In addition to the status mode bits that generate timing as previously noted, operations of the Internal Data Channel (IDC) are controlled by two registers. The D register contains the data to be written on the IDC. The Y register, loaded through the D register, contains address information.

CYCLIC ENCODER

The primary function of the cyclic encoder is to compute Cyclic Redundancy Checksum (CRC) and Longitudinal Redundancy Checksum (LRC) characters. The unit, operating under microinstruction control, can use any polynomial of sixteenth degree or less as a generator to compute a checksum on any message character of from one to eight bits in length.

CRC/LRC polynomials are stored in read only memory (ROM) on the cyclic encoder. The cyclic encoder operates in a serial mode, computing a partial checksum for each message character it receives, one bit at a time.

After a checksum is computed for either an inbound or outbound message character, the results of the computation are accessible by the program until another message character is issued to the cyclic encoder; after which time a new checksum is available and the previous one is either destroyed (inbound) or stored (outbound) in the cyclic encoder but is no longer accessible by the program.

Information is transferred to the cyclic encoder from the output of selector S1 in the microprocessor; and data is transferred out of the cyclic encoder via the input bus to selector S1. Transfer in both directions occurs under microinstruction (firmware) control.

The cyclic encoder computes a checksum on an 8-bit message character in less than 760 nanoseconds following execution of a load X* microinstruction. Computation time decreases by 61 nanoseconds per bit for message characters less than eight bits.

Information is transferred to the cyclic encoder from selector S1 and stored in one of two registers designated A* or X*. Both registers are loadable only, and cannot be used as utility registers since their outputs are not available from the card.

A* Register

The A* register is a 16-bit register used to store the initial value or previous partial checksum for inbound messages, and the initial value to be used with the first character of an outbound message.

The A* word format for a 16th degree generator polynomial is shown in figure 2-3.

Bit 8 corresponds to the highest order coefficient of the CRC polynomial for any length generator polynomial. If a generator polynomial of less than 16th degree is used for CRC computation, the unused bits of the CRC word must be set to zero.

For example, if a generator polynomial of 12th degree is used, bit 8 corresponds to the X^{11} coefficient of the CRC polynomial and bits 7-4 must be set to zero when A* is loaded.

The A* word format for a 12th degree generator polynomial is shown in figure 2-4.

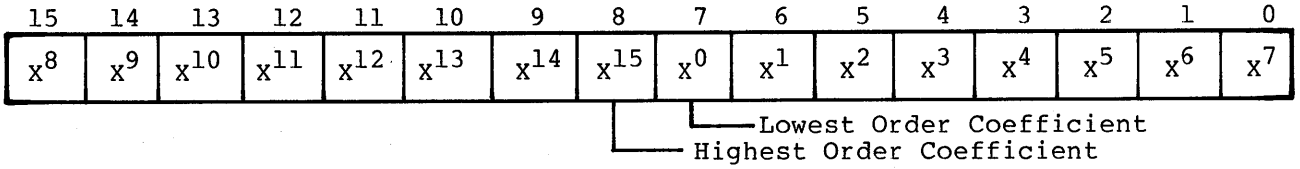


Figure 2-3. A* Word Format (11th Degree Generator Polynomial)

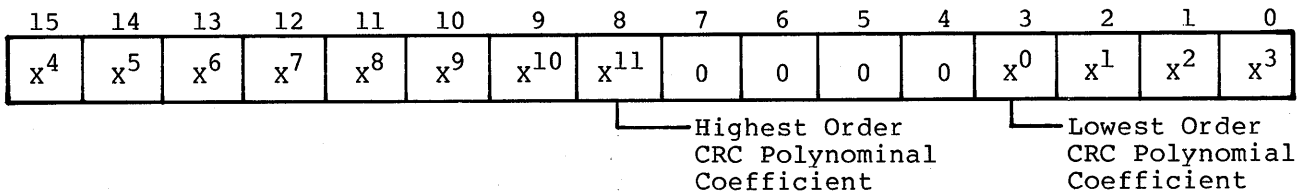


Figure 2-4. A* Word Format (12th Degree Generator Polynomial)

A* must be loaded for each inbound message character and for the first character of an outbound message. The initial value, usually zero, is dependent on the CRC protocol which in some cases can be other than zero.

X* Register

The X* register is a 16-bit register used to store:

1. the message character for which a partial sum is to be computed
2. the ROM address of the generator polynomial
3. the number of bits to be used in the character
4. the shift direction (out of X*).

The X* register is also used to indicate whether the character is inbound or outbound.

The X* word has the format shown in figure 2-5.

CRC Format

The results of a checksum computation are gated to the input bus to selector S1 under microinstruction control, but only the results of the last CRC computation are available.

The CRC word format for a 16th degree generator polynomial is shown in figure 2-6.

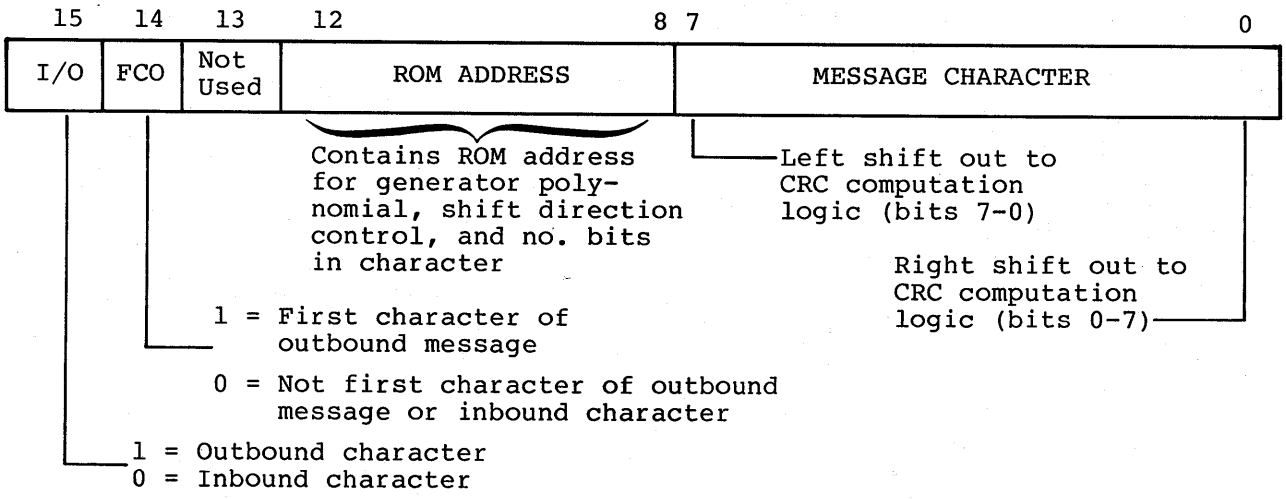


Figure 2-5. X* Word Format

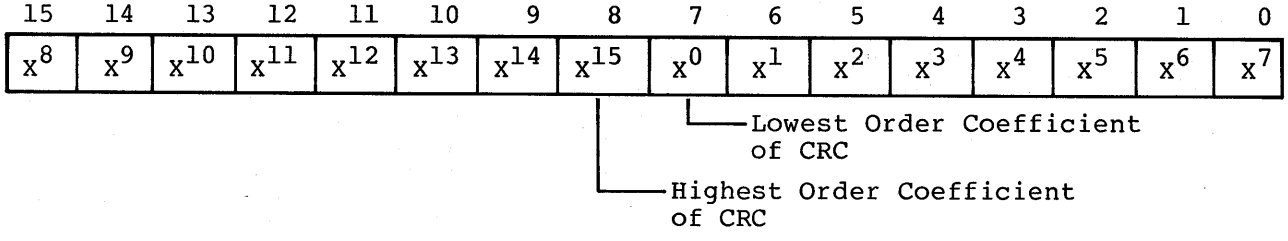


Figure 2-6. CRC Word Format (16th Degree Generator Polynomial)

Bit 8 corresponds to the highest order bit of the CRC polynomial. If a generator polynomial of less than 16th degree has been used for the CRC computation, then only the bits of the CRC word corresponding to the degree of the generator polynomial minus one are valid. The remaining bits are set to zero. For example, if a 12th degree generator polynomial has been used, the CRC is contained in bits 8 through 15 and bits 0-3. Bits 4-7 are set to 0. The CRC word format for a 12th degree generator polynomial is shown in figure 2-7.

Sequence of Operation

Checksum computation starts when the X* register is loaded. For all inbound message characters and for the first character of an outbound message, A* must be loaded before the X* register.

When a load X* microinstruction is executed, the generator polynomial to be used, the number of bits in the character, and the shift direction control bit are provided as outputs from the ROM location specified by X* bits 8-12. Bit 15 is then examined to determine the source of information for the checksum calculation. If bit 15=0, indicating an inbound message character, the contents of A* register are used. If bit 15=1, then bit 14 is examined. If bit 14=1, then the contents of A* register are used. If bit 14=0, the contents of S0 register are used for the computation.

Once a serial checksum computation has started, bits 0-7 are shifted out of the X* register (starting with bit 0 if right shift or bit 7 if left shift specified) through the serial encoder logic. The number of bits shifted out is controlled by the number of bits/character from the ROM.

At the completion of a checksum computation for each outbound message character, the partial sum is stored in register S0 to be used for the next outbound message character. The contents of S0 are not accessible by the program. The results of a partial sum calculation, if required, must be taken before the next microinstruction to load X* is executed.

MAIN MEMORY

Main memory can be assembled in three different configurations:

1. In the basic 2550-2 (MOS) HCP, main memory consists of a MOS memory circuit card having a capacity of 32,768 words.

2. If the user requires as many as 49,152 words of memory, a Control Data 2554-16 Memory Expansion Module can be added to the system; this module is a "depopulated" MOS memory circuit card having a capacity of 16,384 words.

3. If the user requires as many as 65,536 words of memory, a Control Data 2554-32 Memory Expansion Module can be added to the system; this module is a "fully populated" MOS memory circuit card having a capacity of 32,768 words. If a 2556-16 Memory Expansion Module has been previously installed in a system, it must be

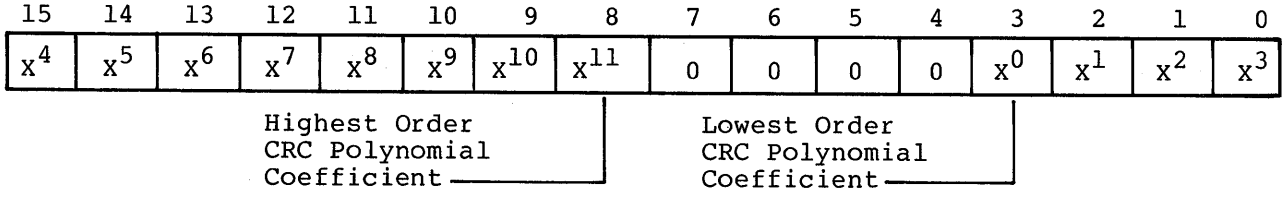


Figure 2-7. CRC Word Format (12th Degree Generator Polynomial)

removed prior to installing the 2554-32 Module.

In each of the preceding configurations, each memory word consists of 18 bits; however, only 16 bits are available to the user. The remaining bits are used to implement memory parity and program protect functions.

Each 32,768-word MOS memory circuit card contains 144 N-channel, Metal Oxide Semiconductor (MOS) chips. Each chip contains 4,096 bits of dynamic, random access memory (RAM). Each card contains eight basic 4K x 18 memory building blocks arranged in two 16K x 18 memory segments. The card also contains all the necessary buffering for the data paths, address inputs, and all clocks and enables for the memory chips.

Presently, there are two different types of memory elements used in the MOS memory circuit cards. There is a fast-access element and a slow-access element. Individual circuit cards contain all of one type, but circuit cards containing different types can be used in the same system. MP memory control logic senses the presence of a slow-access assembly and modifies memory timing accordingly. Timing differences for the two elements are presented in table 2-7.

The main memory modules comprise a one-bank system; that is, the modules are interconnected to preclude simultaneous read operations from two or more locations in main memory. However, there are two ports for main memory, which provide independent data and control paths to memory. These ports are identified as Input/Output (I/O) and Direct Memory Access (DMA).

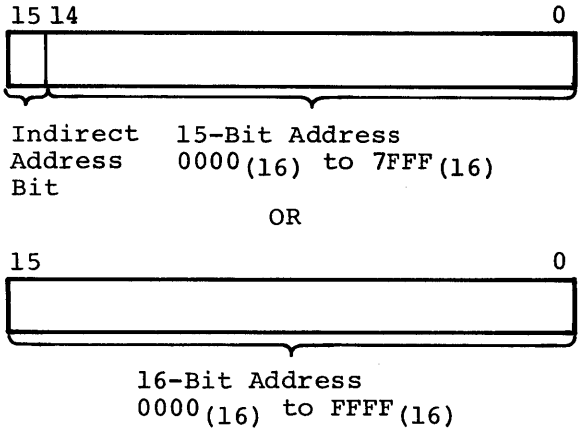
The I/O port serves as the normal access path for the CP to main memory. The DMA port is divided into four subports (identified as DMA 1 through DMA 4). These subports provide the access paths for the multiplexing subsystem, and for direct memory-to-memory transfers from the host computer's peripheral processing unit (PPU) via the communications coupler.

The DMA port allows the DMA peripheral to use all of the 16 memory address bits whenever a memory reference is made. The I/O port, however, allows the use of either 15-bit or 16-bit memory addresses depending upon the state of the multilevel indirect mode bit of the function control register. The standard 2550-2 HCP uses only the 16-bit mode.

TABLE 2-7. TIMING DIFFERENCE FOR FAST-ACCESS AND SLOW-ACCESS MOS MEMORY ELEMENTS

Function	Slow-Access Element	Fast-Access Element
Row Address Strobe (RAS) Access Time	330 ns max.	280 ns max.
Column Address Strobe (CAS) Access Time	195 ns max.	170 ns max.
Random Read or Write Cycle Time	500 ns min.	475 ns
Random Read/ Modify/Write Cycle Time	700 ns min. plus Δ MOD	675 ns min. plus Δ MOD
Refresh Time	2 ms max.	2 ms max.

The address formats for main memory are as follows:



MEMORY MANAGEMENT SYSTEM

Within main memory, user requests are coordinated, processed, and controlled by the Memory Management System consisting of two circuit cards: MOS Memory Interface-Data and MOS Memory Interface-Address/Control. Six user requests can be accommodated: CPU (lowest priority), DMA (4), and Refresh (highest priority). DMA is further subdivided by use of a four-position scanner, thus enabling a total of six sources (including refresh) to access the memory.

Each of the 64 row addresses must be accessed once every two milliseconds (minimum) to maintain the data. These accesses are automatically inserted approximately once every 30 microseconds. Refresh thus constitutes less than two percent of the available memory cycle.

Memory can be accessed by the processor in either full-cycle or split-cycle mode. Selection of the mode to be used is determined by whether or not the protect system is needed. When greater throughput efficiency is needed (the usual case), split-cycle mode is selected, wherein the MOS memory will execute write operations directly. However, when the state of the protect bit must be examined prior to any write operation (such as might be used in a TRACE routine), only full-cycle operation can be used.

MOS Memory Interface - Data

This logic section controls the selection of various data sources, and the distribution of data to various users for both local and external operation. For write operations, the CPU or DMA source is selected for the 16 data bits plus the protect bit. Odd parity is generated over the 17 bits to provide the full 18-bit input. Similarly data from memory is sent to either CPU or DMA, as appropriate. In addition, both memory parity and protect status are tested and made available to the using program. If enabled, the protect logic tests the protect bit associated with a memory cell and determines whether or not a requested write operation can occur.

Upper and lower bounds logic is also provided to augment the protect system. Addresses can be loaded into the upper and lower bounds registers. Any time an attempt is made to access a location lower than the contents of the lower bounds register, or higher than the contents of the upper bounds register, a protect fault occurs, effectively preventing access to the specified location.

MOS Memory Interface - Address/Control

Just as the data interface logic selects the 18 data lines to be used by MOS memory, this logic selects the particular set of command and response lines which correspond to the selected source.

Sixteen bit addresses are selected from either CPU or DMA input lines. These lines are then decoded to match the addressing organization of the MOS memory circuit cards. The lower order 12 bits are grouped into two groups of six bits, and presented alternately to address the specific row and column within each memory chip. The next two bits

are decoded to select one of four blocks of 4K-chips in the array. Finally, high-order bits are decoded to designate which 16K array is enabled (a circuit card can physically contain either one or two 16K arrays).

MULTIPLEX SUBSYSTEM

The multiplex subsystem contains the hardware, firmware, and software elements which provide the data and control paths for information flow between communication lines and user program software. Many of the line/protocol dependent functions that were performed by fixed hardware in previous systems are implemented in common alterable firmware/software allowing reduced development and reoccurring hardware costs for each line and protocol that is added to the communications system.

The purpose of the multiplex subsystem is to gather data from the various terminal devices and deliver this data to the CP, and to receive data from the CP for distribution to the appropriate terminal devices. In performing this data movement (e.g., from terminal to CP) all unique signal characteristics not requiring knowledge of the meaning of the bits are stripped out. These signal characteristics include: data rate, synchronous or asynchronous transmission, parity or no parity, number of bits (5, 6, 7 or 8), and serial or parallel form. All information arrives in CP memory in bit-parallel, right-justified form. Control codes, error control and other housekeeping bits are removed.

Similarly, information from the CP is formatted and conditioned sufficiently to permit transmission to any modem or local peripheral. Differences which require knowing the meaning of the bit (e.g., protocol, code type, etc.) are handled by CP software.

The heart of the multiplex subsystem comprises two high-speed digital data circuits called multiplex loops (see figure 1-7). The input loop handles data enroute to the CP from the terminal devices. The output loop handles traffic from the CP to the terminals. The loops also resolve simultaneous input requests.

The multiplex subsystem is demand-driven. When action is required by a terminal, a demand service request is generated and transmitted to the CP via the input loop.

The key to the ability of the multiplex subsystem to operate in a demand-oriented fashion lies in the nature of the multiplex loop. The multiplex loop can be compared to a railroad track, and the Multiplex Loop Interface Adapter (MLIA) and the Loop Multiplexer (LM) which it connects can be compared to stations on the railroad. Data on the multiplex loop are analogous to the passengers.

The primary functional elements of the mechanism are the communications line adapters (CLAs) that accomplish character assembly/disassembly and terminal interface; the input loop which transports data demands, data and status from the CLAs to the CP; the output loop which transports data and control from the CP to the CLAs; an MLIA which controls the movement of data demands, data and supervision between the loops and the CP; and the LMs which provide access to the loops by the CLAs.

Multiplex action on the input loop occurs when the MLIA issues a loop-end control signal. As the signal propagates around the loop, each LM in turn has an opportunity to put data and/or supervision from the CLAs on the loop. LMs holding information from the CLAs, monitor the loop for the loop end signal, place the information on the loop, and replace the loop end signal. The MLIA transfers the information received on the input loop to buffer storage for processing by the CP.

The MLIA takes addressed information from buffer storage under direction of the CP and transmits this information on the output loop. The LM receives the addressed information blocks and presents the address to all CLAs. The CLA recognizing its address is selected and the information is transferred from the output loop to the CLA.

LOOP TRANSMISSION

Data transmission on the multiplex loops is in serial form at a fixed rate of 20 MHz. Every twelfth bit is a cell frame marker that defines a 12-bit loop cell. The cell frame marker is followed by a cell identification field (3 bits); its contents define the meaning of the remaining field (eight bits) of the loop cell.

A contiguous group of loop cells starting with a CLA address and ending with a Cyclic Redundancy Checksum (CRCS) is called a line frame. A line frame contains data and/or supervision related to a single CLA. Line frames are made up of four different types of loop cells; address, data, supervision and (CRCS). Line frames carry data and supervision loop cells on the loops between the CLAs and the CP.

The CLA address loop cell carries Output Data Demands (ODDs) as well as a unique binary number (address) of the CLA which is independent of the CLA's physical location within the system. An ODD is a request from a CLA to the CP for data to be sent to the CLA for subsequent transmission on the communication line by the CLA.

Data loop cells contain up to eight bits of information (one character) that is to be transmitted to, or has been received from, communication lines. The number of data characters per line frame is limited by the hardware to a maximum of 16, but best performance is obtained by sending only one instance of data and/or supervision per line frame. This will normally be one data and two supervision characters if supervision is present at all. Supervision takes the form of commands outbound and status inbound.

INPUT SEQUENCE

Data from remote terminals are received in serial fashion over the communications lines and placed into single character buffers in the CLA. When an entire character is received, the CLA notifies the LM by signalling on a special line associated with each CLA. At this time the CLA buffer is acting as a waiting room for the passengers (data) wishing to board the train to the MLIA.

Empty trains are originated at the MLIA. The trains, called input loop batches, are composed of loop cells which are the cars of the train in the analogy. A loop cell contains 12 bits which comprise eight bits of cell information, three bits of cell identification and a 1-bit frame marker. The frame marker designates the start of a cell.

As an input train passes an LM which has input data to be forwarded to the CP, the LM marks the first empty frame as a CLA address by setting the appropriate cell identification, and places the CLA address in the eight information bit locations. The subsequent empty cell is marked as data and the data from the CLA is placed in the information bit locations. If a CLA has supervisory information it is placed in subsequent cells. The next empty cell is labeled as a Cyclic Redundancy Checksum (CRCS) cell; the information bit locations are filled with a special checking code.

The train returns to the MLIA which utilizes the CRCS to verify the incoming data. If the data is incorrect, the next loop batch identifies the data that was incorrectly received, and the MLIA initiates error recovery procedures. If the input data is found to be

correct, the MLIA transmits the data to the CP along with the associated address. The MLIA initiates action to start the next train on the input loop as soon as the previous one returns.

The above sequence is repeated for all other CLAs connected to that LM; then, CLAs connected to adjacent LMs are handled until there are either no data bits left in the CLAs, or no empty cells left in the train.

OUTPUT SEQUENCE

Data output is accomplished in a manner functionally similar to input. In this case, trains of data originated by the MLIA on the multiplexer output loop (usually in response to ODDs) are not empty. Here, the output loop batches are composed of cells which specify the destination and content of a message, as well as certain supervisory information regarding the transmission.

The destination portion of the data transmission is in the form of a loop cell which contains, in its information field, the address of the CLA which is to receive the data. The content portion of the data transmission is carried in the information field of the loop cell which immediately follows the CLA address cell. The third portion of the data transmission consists of one or more cells (usually two) identified as supervisory cells. The information contained in these cells is used for CLA command functions which are described later in this section.

Finally, there is a CRCS cell signifying the end of the activity for that CLA. The output loop batch may contain output data and commands for a number of CLAs.

Once the data flow from the CLA to the remote terminal has begun, the CLA initiates a demand signal when more data is required. The CLA signals the LM in a manner similar to that for announcing the presence of data. The LM then uses an empty loop cell in the next input train comprises two cells. The first cell contains the ODD signal and the CLA address. The second cell contains a CRCS for error checking. Upon receipt of the ODD command, the MLIA in turn notifies the CP to transmit the next character.

This concludes an overview of the functions of the multiplexing subsystem. The following three subsections describe, in much greater detail, the functions of the three principal hardware components of the multiplexing subsystem: MLIA, LM and CLA.

MULTIPLEX LOOP INTERFACE ADAPTER

The multiplex loop interface adapter (MLIA) provides the interface between the CP and the multiplex loop.

The MLIA is designed to gather input data from as many as eight LMs. Since each LM can interface with as many as 32 CLAs, each MLIA is capable of receiving data from as many as 256 CLAs. However, the maximum number of CLAs used in the 2550-2 HCP is 128.

The MLIA performs limited editing and checking functions on the input data, and deposits the data directly into a large circular buffer in main memory via the direct memory access (DMA) channel.

The MLIA receives ODD signals from the CLAs and forwards these to the CP as high-priority interrupts. The MLIA then receives output data from the CP via the internal data channel (IDC), and distributes the data to the LMs. The MLIA provides first-in, first-out (FIFO) buffering for input data, ODD signals and output data.

Input Functional Sequence

Input functions are performed by the MLIA in the following sequence.

1. The MLIA generates and transmits clock, loop end, empty and null cells on the output side of the input loop to solicit information from the CLAs.

2. It monitors the input side of the input loop for:

   a. Clock
   b. Loop synchronization
   c. Information placed on the loop by the LMs that must be stored in memory
   d. Loop end cells for error control

3. The MLIA checks the CRCS character, and stores each line frame received on the input loop in the CP circular input buffer (CIB) via the DMA channel.

4. After receipt of the loop end cell on the input side of the input loop, another empty loop batch is generated immediately. This provides the maximum rate of loop-end generation without allowing more than one loop end cell on the loop at any one time.

Output Functional Sequence

The following is the sequence of output functions performed by the MLIA:

1. The MLIA detects ODD signals received from the CLAs on the input loop and provides for special priority processing of the ODDs.

2. Processing of the ODDs requires that the MLIA obtain the next character from the CP and transmit this character on the output loop. (The ODD processing function is controlled by the firmware.)

3. The MLIA transmits line frames on the output loop on command from the CP.

   a. Output line frames contain CLA address, data characters, CLA commands (if any) and CRCS.

   b. The CRCS is generated for each line frame by the MLIA and is transmitted as the last cell of each line frame.

4. The MLIA monitors the input side of the output loop for loop synchronization and loop batch timeout, which are used for error recording and diagnostics.

MLIA Partitioning

The MLIA is functionally partitioned into three parts:

1. Processor Interface
2. Input Loop Interface
3. Output Loop Interface

PROCESSOR INTERFACE

The processor interface controls the MLIA's communication with both the interrupt data channel (IDC) and direct memory access (DMA) channel of the CP.

Three operating parameter words are received from the CP via the IDC: flag, null and empty (FNE). These FNE parameters establish the amount of information that can be accumulated during each scan of the input loop. Two other words define the length and location of the CP's CIB where the incoming data is to be placed. Operational commands, such as ODD limit, are also received via the IDC channel.

The MLIA processor interface consists of the following eight function elements:

1. ODD Limit and CIB Length Parameters Registers - The ODD

limit parameter sets the threshold for the number of ODD signals that the CP allows to be held in the MLIA. When this threshold is reached, the input loop train is temporarily halted. The CIB length parameter sets the size of the CIB in the CP. This parameter operates under firmware control.

2. CIB Location Address Parameter Registers - The CIB address parameter sets the locations within the CP memory that can be used for input storage. This parameter operates under firmware control.

3. FNE Parameters Register - The following FNE parameters operate under firmware control:

   a. The F parameter sets the quantity and types of flags that may be used on the input and output loop.

   b. The N parameter sets the number of null cells allowed on the input loop and changes with traffic load.

   c. The E parameter sets the number of empty cells allowed on the input loop and changes with the traffic load.

4. Status Registers - The status registers report the operational status of the MLIA to the CP.

5. Input Loop Launch Control - The input loop launch control is used to control the generation of the input loop train, and is controlled by the ODD limit, CIB length and FNE parameters.

6. CIB Address Generator - The CIB address generator receives input parameters from the CIB length and CIB location registers, and chooses the location within the CIB for storing blocks of information. This generator operates under firmware control.

7. Last Frame Position Pointer - The last frame position (LFP) pointer is only changed after a complete block of information has been loaded into the CIB by the MLIA.

8. Interface Buses - Finally, the processor interface circuit card contains three buses which provide the proper electrical interfaces for the exchange of operating parameters.

INPUT LOOP INTERFACE

The input loop interface controls operation of the input loop which gathers ODDs and other incoming data. A serial, 20-megabit-per-second line is connected sequentially to each LM, and then returns to the input side of this logic. The loop is continually being scanned for any inbound traffic (ODDs, data or supervision).

The following is the functional sequence for the MLIA input loop interface.

1. Cell Generator - Starting here, the loop end cell, followed by empty and null cells, is generated. The loop end timer is activated and the train starts around the loop.

2. Parallel-to-Serial Converter - When the train arrives at this converter, the 12-bit cells are converted to a 20-megabit-per-second serial stream and sent via a clock and a data cable to the string of LMs and CLAs.

3. Loop Multiplexers - If an LM has information from any of its 32 CLAs, the LM posts a flag; the LM then searches the loop for a loop end cell followed by at least one empty cell. When the LM detects this combination of cells, the LM removes the loop end cell and starts replacing the empty and null cells with data and/or supervision.

Groups of cells are divided into line frames. Line frames start with a CLA address and end with a CRCS. A line frame contains data and/or supervision related to a single CLA. As the loop travels through an LM, each CLA associated with that LM can place one line frame on the loop. The LM replaces the loop end cell on the end of the train, and passes the loop batch on to the next LM.

The LM next in line then searches for the loop end cell followed by at least one empty cell. If this combination is detected, the LM removes the end cell, loads data on the loop, replaces the end cell, and passes the loop batch on. Each LM is serviced in turn until all have been serviced, or the loop has no more empty cells. If an LM has no data for the loop progress is not inhibited. The amount of traffic that can be accepted by each loop batch is controlled by the CP via the FNE parameter in the MLIA.

4. Serial-to-Parallel Converter - After the last LM has been serviced the loop batch is sent back to the MLIA and converted from serial to parallel form.

   a. Sync Detector - Here the data is checked for synchronization, format, sequence, etc., to ensure a high degree of traffic integrity.

   b. CRCS Check - Here the error control characters are checked to verify proper transmission.

5. Edit Logic - Edit logic strips off the ODDs and sends them to the ODD FIFO buffer with a flag for high-priority interrupt. The remainder of the data is sent to the main input FIFO buffer.

6. Main Input FIFO Buffer - The main input FIFO buffer can store 64 16-bit words, and is flagged with an interrupt (input line frame interrupt) whenever a complete line frame has been placed in the CIB. Its main purpose is to eliminate the need for the firmware comparing the CIB read and write pointers in order to detect recent activity. Data is extracted from the main FIFO buffer as required by the CP.

7. Output Data Demand FIFO Buffer - ODDs are removed from the data train by the edit logic and sent to the ODD FIFO buffer. Presence of ODDs is flagged with a high priority interrupt (ODD interrupt). The ODD FIFO buffer can store eight characters containing eight bits each, and the buffer is serviced within 20 microseconds under firmware control.

8. Loop End Detector - Loop end cells are sent to the loop end detector. The loop end timer, started with the launching of the train, then stops and orders a new loop batch to be generated. If it appears that the MLIA input buffers may be overrun, the order to form a new loop batch is delayed by firmware control until the probability of overrun is reduced.

OUTPUT LOOP INTERFACE

The output loop interface controls the operation of the output loop, which distributes outgoing information to the LMs. A serial, 20-megabit-per-second line connects sequentially to each LM and then to the input side of the output loop interface logic. Information to be sent is received via the IDC and accumulated in the output buffer. As soon as all messages that are to be distributed during the next loop scan have been received (usually one), the characters are converted to serial form; error control and housekeeping

characters are added and the entire batch is sent out on the loop. This starts the loop timer. The timer stops when the message is returned to the MLIA via the output loop. Should a problem occur on the loop and the message does not return to the MLIA within the allotted time, the timer generates an error signal. During periods of no traffic, empty cells are sent to ensure loop continuity and maintain synchronization.

The following paragraphs describe the functional sequence for the MLIA output loop interface.

Loop and empty cells are continuously generated and transmitted. As soon as the loop end cell returns on the loop, another cell is transmitted even if data is not available in the output buffer; this ensures loop continuity and synchronization.

When the CP has output information to be transmitted, the data is routed from the IDC to the output buffer. The output buffer can store 16 words of 16 bits each; the buffer holds information until the next loop train is started. The parallel-to-serial converter places the output data on the serial loop. CRCS cells are added for each CLA message and accumulated for error checking. The loop batch is then transmitted to the LMs.

The output section of each LM receives contiguous loop cells from the MLIA or the preceding LM on the loop. The LM utilizes empty cells to synchronize with the data stream. When synchronized, the LM looks for a CLA address as frame boundary, and passes loop cells within the frame to the CLAs. All CLAs connected to the LM are connected on a common bus, and each CLA has a unique address.

When the LM has received a CLA address loop cell, it presents the cell to all CLAs on the common bus. The CLA with the corresponding address is selected and connects itself to the output bus. The following loop cells, containing data and/or commands, are then transferred from the output loop to the selected CLA; transfer continues until the LM detects the CRCS cell which signals the end of the line frame. The LM checks the received CRCS against the CRCS accumulated for the line frame, and if a difference is detected, the selected CLA is notified of the error prior to deselection. The LM then passes the loop on to the next LM and back to the MLIA.

The loop batch then returns to the MLIA and is converted back to parallel form. Synchronization is checked, errors (if any) are accumulated, and loop time is stopped. An order to start a new loop batch is then initiated. Had the train not been received back, the timer would have timed out, signalling an error condition.

LOOP MULTIPLEXER

The loop multiplexer (LM) serves as the interface between the CLAs and the input and output loops. The LM accepts data in parallel from the CLAs, serializes this data, and presents it to the input loop. The LM also assembles loop cells from the output loop, and presents data in parallel form to the CLAs.

Input Section

The input section of the LM receives contiguous loop cells from the preceding LM or from the MLIA. When not synchronized to the data stream, the LM passes all information received to the next LM or to the MLIA. The clock signal is reconstructed to specification when passed sequentially to the next element in the system.

The LM utilizes the unique combination of a loop end cell and a subsequent empty cell to acquire synchronization. When synchronized, the LM accepts data from each CLA that has information for the system; the LM then outputs this information to the input loop, in a line frame.

At each frame boundary, the LM examines the input loop for an empty cell. If the empty cell is detected, the LM places the next CLA frame on the loop. If an empty cell is not detected, the LM drops synchronization.

All CLAs plug into a common bus and control-line structure. Each CLA has a unique address and may be plugged into any CLA slot. Each CLA can inform the LM, via one of 32 lines, that it has information for the system. The LM can inform each CLA, via one of 32 control lines, that it can place information on the CLA bus in response to strobe signals. When a CLA has been selected, the LM accepts information from that CLA until either an information-end signal from that CLA is received, or 16 cells have been processed. When one of the above occurs, the LM deselects that CLA, and selects another CLA. When all CLAs requesting access to the loop have been serviced, the LM drops synchronization.

INPUT FUNCTIONAL SEQUENCE

The following is the sequence of functions for the input section of the LM:

1. The input-loop data stream is examined at each bit time for a 24-bit code consisting of a loop end cell followed by an empty cell. Upon detection of this bit pattern, the LM is cell-synchronized to the data stream. The loop end cell is removed from the data stream and one empty cell replaces the loop end.

2. When synchronized, additional storage of information-available signals from the CLAs is inhibited, and only those CLAs with stored-information-available signals can be selected. These CLAs are serviced on a priority basis determined by their slot position in the multiplexer card cage assembly. CLAs positioned closest to the LM circuit card have the highest priority. Frames placed on the loop by the LM are continuous with no empty or null cells between frames.

3. The selected CLA inhibits changes in its input information storage registers during selection. Provision is made in the CLA to capture critical status during selection.

4. Information is presented to the LM by the CLA in 11-bit bytes. These are identical to bits 1 through 11 in the loop cell. Address information is presented first, followed by up to 15 other bytes of information. Null cells may be placed among these bytes and are included in the byte count.

5. The LM replaces each cell received on the loop with the information received from the CLAs. When the CLA informs the LM that the last byte has been presented, the LM concludes the frame with a CRCS character.

6. The LM examines the cell re-received on the input loop prior to the CRCS output to determine the presence or absence of an empty cell.

   a. Presence of an empty cell causes the LM to select the highest priority CLA not yet serviced. This process continues until all CLAs with stored requests have been serviced once.

   b. Absence of an empty cell causes the LM to halt CLA selection, and output the CRCS cell of the last frame with a contiguous loop end cell.

7. When all stored requests have been serviced, the LM places a loop end cell and an empty cell on the input loop, and then drops synchronization.

INPUT ERROR PROTOCOLS

The LM utilizes the following four protocols to ensure the integrity of data on the input loop:

1. Cyclic Redundancy Checksum
2. Restart Loop Cell
3. CLA Overrun

1. Cyclic Redundancy Checksum

The input section generates a CRCS character for each frame placed on the loop. Accumulation of a CRCS character begins with the first bit of the line frame, and ends with the fourth bit of the CRCS cell in the line frame. The LM generates the CRCS character by treating the serial information as a long binary number and dividing that number by a generator binary code, using modulus two arithmetic. The generator is 111000011, $X^8+X^7+X^6+X+1$ (in polynomial notation).

2. Restart Loop Cell

The MLIA notifies the LM of an input batch received in error via a restart loop cell. The restart loop cell is placed on the loop immediately following detection of the bad batch. The LM detects the restart loop cell, and notifies the CLA which last placed information on the loop, of the error.

3. CLA Overrun

If a CLA places more than 16 bytes of information on the bus, the LM assumes an error has occurred and disconnects that CLA. The CRC is added to the data stream as in a normal frame termination.

Output Section

The output section of the LM receives contiguous loop cells from the preceding LM or from the MLIA. The output section always passes all loop cells received to the next LM or to the MLIA. The clock signal is reconstructed to specification when passed sequentially to the next element in the system. The LM utilizes empty cells to synchronize itself to the data stream. When synchronized, the LM looks for frame boundaries, and passes loop cells within the frame to the CLAs. The LM checks the CRCS character received with each frame.

When the LM has accepted the address of a frame, it presents this address to all CLAs. The CLA with the corresponding address then connects itself to the output bus. The selected CLA accepts the information directed to it until disconnected by the LM. If a CRCS error is detected by the LM, it is reported to the CLA prior to termination.

OUTPUT FUNCTIONAL SEQUENCE

The following is the sequence of functions for the output section of the LM:

1. The LM examines the output loop data stream each bit time for the presence of an empty cell bit pattern. Upon empty cell detection the LM is cell-synchronized and begins a search for an address code in the identification field of each loop cell.

2. When the LM locates an address code, it presents the address with a select signal to all CLAs in the multiplexer card cage assembly. The CLA with the same address as that presented connects itself to the bus, and is prepared to accept 11-bit bytes from the LM. These bytes are identical to those contained in the loop cells that are part of the frame intended for the selected CLA.

3. When the LM receives an identification code indicating the presence of a CRCS character, it

halts byte transfer to the CLA bus. The CRCS character is compared to that accumulated over all preceding bits of the received frame, and the results reported to the CLA.

4. After CRCS verification, the LM again searches for address codes in the received loop cells.

OUTPUT ERROR PROTOCOLS

The LM utilizes the following two protocols, Cyclic Redundancy Checksum and CLA Overrun, to ensure the integrity of data on the output loop.

1. Cyclic Redundancy Checksum

   The output section accumulates a CRCS character for every line frame received on the output loop. Accumulation begins with the first bit of the line frame and ends with the fourth bit of the CRCS cell. The CRCS character is generated by treating the serial information as a long binary number and dividing that number by a generator binary code, using modulus 2 arithmetic. The generator is 111000011 or $X^8 + X^7 + X^6 + X + 1$ (in polynomial notation).

2. CLA Overrun

   If more than 16 cells are received by the LM in a given frame, the LM reports an error to the selected CLA and disconnects that CLA.

COMMUNICATIONS LINE ADAPTER

The function of the communications line adapter (CLA) is to provide the interface between the LM and a terminal with or without a modem. On input, the CLA serves as a serial-to-parallel converter assembling the serial data at the signalling rate of the terminal, and transferring the data to the LM in an 8-bit byte. On output, the CLA functions as a parallel-to-serial converter receiving the data characters in bytes of eight bits from the LM and outputting them serially at the signalling rate of the terminal.

In order to minimize the number of CLA types required, the CLA hardware is designed to accommodate most circuit characteristics under software/firmware control. Each circuit's individual characteristics are handled, for the most part, by associated software which detects the specifics and sets parameter registers within a particular CLA to handle them. Software-selected elements include: circuit speed, code set, mode of operation and, for synchronous circuits - the synchronization pattern(s), and for asynchronous circuits - different input and output speeds, parity detect and generation, and number of stop bits per character. This software detection is possible by means of an algorithm associated with the arrival of the first few characters of a data transmission.

The only remaining circuit variables are whether the circuit is synchronous or asynchronous, and if synchronous, the signal level standard to which the circuit interfaces. The terms synchronous and asynchronous refer to the way in which the data is transmitted: synchronously with a clock, or without reference to a clock (but at a fixed rate and with synchronization elements appended).

A number of CLA types, which have differing functional and/or electrical interface characteristics, are available. In addition, the CLAs constitute a product line which is under continuing development. Therefore, it is recommended that the user refer to the Hardware Reference and Maintenance manual(s) for the functional descriptions of the specific CLA type(s) selected for his system. See preface.

TAPE CASSETTE CONTROLLER

The tape cassette controller serves as the access path for all communications between the transport and the CP. The controller provides input to, and receives output from, the transport at a rate of 750 characters per second (8-bit characters). This yields a data transfer rate of 6 KHz which is compatible with the transport. The controller is capable of providing data to the CP at a 9600 baud rate.

INTERFACE

The tape cassette controller interfaces are the following: CP Interface, Deadstart Interface, Transport Interface.

CP Interface

The cassette controller interfaces to the internal I/O, IDC channel of the CP. The controller may be status- or interrupt-driven at the macro level, and data transfers may also take place in the auto-data transfer (ADT) mode at the micro level. The data path between the CP and the controller is eight bits wide.

Deadstart Interface

The cassette controller deadstart interface provides the means for the tape cassette transport to begin reading data in response to an operator-initiated command at the maintenance panel or the communications console. Only the read function is performed in the deadstart mode of operation. The data is ASCII, asynchronous, binary, serial-by-bit, at the 9600 baud rate. The maximum character transfer rate is 750 characters per second. Each character is composed of one start bit, seven binary bits (data), one parity bit, and at least one stop bit.

Transport Interface

Signal transfer between the controller and the transport is provided by a cable having a maximum length of 10 ft (3m). The cable connects the backpanel directly behind the controller card to the transport.

ADDRESSING AND OPERATIONS

The equipment address is defined by the E field of the Q register (Q10-07). This address is specified by selecting the positions on a 4-segment switch located on the controller card. The W field of the Q register (Q15-11) must be zero. Standard equipment address for the tape transport is hex seven (0111).

Input and output operations are specified by a combination of the D field of Q (Q00), and the I/O instructions received from the MP as shown in table 2-8, and in the word format shown below.

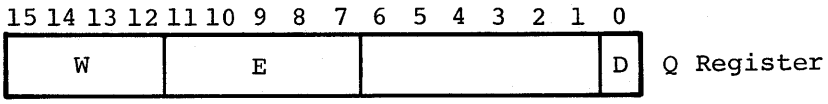


TABLE 2-8. TAPE CASSETTE CONTROLLER OPERATION SELECTION

D Field (Q00)	Microprocessor Instruction	
	Output from A (OUT)	Input to A (INP)
0	Write data transfer	Read data transfer
1	Control function	Status

DATA TRANSFER

Whenever it is desired to read information from a tape cassette or write information onto a tape cassette, an orderly dialogue occurs on the I/O channel (IDC) which results in a series of actions by the controller and/or transport. These are summarized in the following paragraphs.

Write

Upon insertion of the cassette into the transport and closing the lid, the Ready status (and possibly the Side B status, depending on which side of the cassette in in the active position) occurs, and a Rewind sequence is automatically initiated (assuming that the Auto-Rewind feature has not been disabled with the switch furnished on the controller board for this purpose).

The tape rewinds to the transparent leader, stops and winds forward until the Beginning of Tape (BOT) hole is sensed. Tape motion ceases, BOT status occurs, and the READY lamp on the maintenance panel lights.

A function command is issued directing initiation of write motion (typically with Data Request and End of Operation interrupts requested). Forward tape motion begins at 7.5 inches (19 cm) per second; at the end of a 50-millisecond interval, the tape is up to speed. The tape will move approximately 1.6 inches (4 cm) (or 0.8 inch (2 cm) for all records after the first). Since the transport write data line was held steady, an Inter-Record Gap (IRG) is written on the tape.

After receipt of the write motion command, the Data Request status bit is set (accompanied by a macrointerrupt, if requested). If an ADT mode, a micro (data) interrupt also occurs.

As soon as the character is obtained and stored, the Data Request (and corresponding interrupt) drops.

At the conclusion of the IRG interval, the preamble (10101010) is written. Starting with the LSB (zero), the bits are sent one at a time through the biphase encoding logic (where bit clock transitions are added) and to the transport write circuitry.

Following the issuance of the preamble, the first data character is written (LSB first) as described above. The data is also sent through the Cyclic Redundancy Checksum (CRC) generator for accumulation.

Transference of the character from output buffer storage to the transport register empties buffer storage. Data Request status goes set again (accompanied by a macrointerrupt, if requested). If in ADT mode, a micro (data) interrupt also occurs. This action occurs each time during the data portion of a write operation that the output buffer storage becomes empty.

As soon as a character is obtained and stored, the Data Request (and corresponding interrupt) drops. Any time that the output buffer storage is empty at the time its contents should be moved to the transport, the controller declares an "underrun" to exist. Occurrence of a data underrun defines the end of the data portion of a block. During ADT operation receipt of a Terminate signal defines the end of the data transfer, micro (data) interrupts are disabled, and an "underrun" condition develops which initiates shutdown as in normal A/Q operation.

At this time, the accumulated CRC becomes the source for the phase-encoding logic for the next 16 bits. Following this, the 8-bit postamble is written. At this point the biphase encoding logic is turned off (i.e., the Write Data line is held steady).

Tape motion continues for a fixed delay time and the transport is then commanded to stop. During these two

intervals, IRG is being written. The delay time inserts a sufficiently long interval to ensure that the proper length IRG is written, consistent with the transport stop dynamics. End of Operation status is then set and the unit becomes not busy.

Refer to the following section for Read-After-Write check operation.

Read

As stated under Write, the cassette is inserted, the lid is closed and the Rewind sequence is initiated.

After the BOT point has been reached (as indicated by the status bit), Side B status checked (to determine whether the cassette has been loaded correctly), a function command is issued directing initiation of read motion (typically with Data Available and End of Operation interrupts requested). Forward tape motion begins (7.5 inches per second/19 cm per second). The tape comes up to speed and sometime later (approximately 70 msec after the tape is at proper speed), data transitions begin arriving from the transport read circuits.

These data transitions become the read data. The transitions are also used to synchronize bit timing to the recovered read data streams.

Information passes through the more than twenty-four bits (3 characters) of serial bit storage. This delay is needed for CRC and postamble detection (described later). Character synchronization is obtained from the eight bits of the preamble.

When the preamble has filled the last eight bits of the input storage, character synchronization is obtained and the preamble is discarded. As the next character (first data character) reaches the last eight bits of the serial input storage, however, it is broadside-loaded into the (8-bit parallel) input buffer storage. Anytime the input buffer storage is filled, Data Available status (and interrupt, if requested) is set. If in ADT, a micro (data) interrupt also occurs. This signals the MP to input a character. Meanwhile, data continues to shift through the serial bit storage under control of the recovered read data bit clock. Data Available status (and the interrupts, if requested) clears whenever the MP empties the input buffer storage.

As each character reaches the end of the serial bit storage, it is moved into the buffer storage where it remains until emptied by the MP. If the input buffer storage is not empty at the time that the next character needs to be moved to the input buffer storage, an overrun condition is signalled and the corresponding status bit (and alarm interrupt, if requested) is set.

This action of shifting data, filling and then emptying the buffer storage continues until a sensor on the transport read data line detects a lack of transitions, indicating that the ending IRG has been reached. It is only when this IRG is sensed that the three preceding characters can be identified as the CRC and postamble (rather than data). The CRC check logic is located on the serial bit storage so that at the next character boundary, the data and CRC have been checked. A test for all zeros is made in the CRC check; sensing of any nonzero condition sets the CRC error status (and alarm interrupt, if requested). No more Data Available status (or interrupts) are set since the CRC and postamble are discarded.

As soon as the CRC has been checked, and the end of record (IRG) detected, the transport is halted and the End of Operation status (and interrupts, if requested) are set.

The same CRC check sequence described above occurs during the read-after-write phase of any write data operation (timing is identical). Note

that during any Search Tape Mark, Backspace, or Read command, if the tape reads undefined bit patterns or fully erased tape, tape motion is stopped in the first blank area after:

1. eight or more data transitions, or
2. 2.7 seconds, or
3. 20 inches (0.5 m) of blank tape.

Echo

Echo mode places the entire controller data path in a loopback mode and tests the controller in lieu of the transport. This is primarily a maintenance feature.

Operation starts as in a write operation and terminates as in a read operation.

Data interrupts occur for both Data Request and Data Available (under the same conditions as described individually under Read and Write).

Owing to the inherent time delays through the controller, several Data Request interrupts are received prior to the first Data Available interrupt. After that, they occur alternately until the CP stops sending data. The remaining Data Available interrupts are then received. Since the program cannot distinguish between the Data Request and Data Available conditions using the signal Data Request/Available, a separate status line is provided which is true only for Data Available conditions.

Upon reaching the end of block, Echo mode is reset.

Any motion commands that are included in the output function to select Echo mode are ignored.

Auto Data Transfer

Auto Data Transfer (ADT) mode provides for pseudo direct memory transfers of data blocks to or from the tape cassette controller over the IDC. At the macro level, the transfer appears as a direct memory/storage access (DMA/DSA) transfer; however, at the micro level, the 1700 emulator processes each data (micro) interrupt, and inputs or outputs the next word of data in a singular fashion. Thus ADT takes less time to transfer a block of data than input/output via the INP, OUT, or SIO instructions, but without the need for special DMA circuitry.

The controller performs ADT operation whenever the ADT mode bit is set. Depending on whether write motion or read motion has been requested, data write or read operations occurs as previously described except for the occurrence of microinterrupts. A/Q channel operations trigger Data Request/Available macro (program) interrupts (if enabled). During ADT operations, Data Request/Available simultaneously triggers a micro (data) interrupt in addition to the macro (program) interrupt. Owing to machine interrupt servicing priorities, the micro (data) interrupt is seen first and thus it is the one serviced. The A/Q channel then provides/accepts a character. This sequence continues until a termination signal (STERM) is received. Receipt of this signal inhibits generation of all further data interrupts (macro or micro) for the remainder of the block.

Since Data Request/Available interrupts are no longer being generated, data no longer is provided/accepted by the A/Q channel and the block terminates normally as previously described. As soon as tape motion ceases, End of Operation macro interrupt is generated with End of

Operation status set. Note that during an ADT Read operation, the alarm status (and macro interrupt, if enabled) occurs if the entire record has not been read.

Deadstart

When plugged into an I/O slot that is wired for deadstart operations, prerecorded cassettes can be read into the CP via the 9600 bps asynchronous serial panel interface. To be chosen as the deadstart device (many controllers could be attached to these same signals), one and only one device can be Ready. In the tape transport, with power applied and a cassette mounted, Ready is obtained by closing the lid. (Conversely, the transport is made Not Ready by opening the lid or by removing the cassette.)

Tape positioning is strictly under control of the Auto-Rewind or External Rewind features. If the Auto-Rewind feature is enabled (at the controller board), then the tape is automatically positioned to BOT (load point) when the cassette is loaded and the lid of the transport is closed. If the Auto-Rewind feature is disabled, then no tape positioning occurs during the deadstart procedure. If the External Rewind signal is generated, then the tape is automatically positioned to BOT (load point).

Once the cassette is ready for the operator to initiate the deadstart operation, the "READY" indicator (which is mounted on the maintenance panel) lights.

Operation begins when the Deadstart switch is pressed. This action causes the CP to generate the Deadstart signal, which controls the entire deadstart operation.

While the Deadstart signal is active, the DEADSTART ACTIVE indicator (which is mounted on the maintenance panel) is also lit.

As soon as the Deadstart signal becomes true, the controller initiates a read motion command and the operations begin. As soon as data characters become available from the end of the serial bit storage, they are parallel-transferred to another serial bit storage in which start and stop bits are appended and the storage serial output is shifted out at a 9600 bps rate to the panel interface.

Operations continue through one or more records until the Deadstart signal drops (due to programming control). At that time, any data remaining in the current record is transferred and the transport comes to an orderly stop in the IRG after the current record.

In case of any errors detectable by the controller, the transfer comes to a normal termination and appropriate status set (as in any other read operation).

Note that the cassette READY indicator goes out and the DEADSTART ACTIVE indicator remains on after the tape has ceased all motion whenever the controller has detected a CRC or Format error during deadstart operations. Also note that deadstart routines cannot be written in the region of tape between the End of Tape (EOT) hole and clear trailer.

During any deadstart operation, controller status is reported as Ready and Busy.

Reject Conditions

The following is a summary of the conditions that can cause the controller to generate an external Reject.

1. Protect Violation
2. Illegal (all zeros) Function Command
3. Illegal Echo Command - trying to select Echo mode while the controller is busy. Note that this

means any other function commands that are issued after the controller is in the Echo mode (to clear and re-enable interrupts, for example) must not include function bit A05, or else a Reject will occur.

4. Illegal ADT Command - trying to issue an ADT command without a Read or Write Motion command and without EOP interrupt enabled.

5. Mode Conflict - Echo and ADT modes simultaneously selected.

6. Motion Command when Controller is Busy

7. Motion Command when Unit is Not Ready

8. Illegal Motion Command - attempting a Write Tape Mark, Write, or Erase Motion command when Write is not enabled (when file is protected).

9. Write Data Transfer without a Data Request

10. Read Data Transfer without a Data Available

TAPE CASSETTE TRANSPORT

The tape cassette transport can be used as a program-load or data-capture device. The transport is capable of both reading and writing on Philips-type, magnetic tape, digital data cassettes. The transport contains servo motors, servo amplifiers, read-write circuitry, read-after-write head, and the necessary control circuits for processing data and controlling the tape.

Tape speed in forward or reverse is approximately 50 inches (127 cm) per second. Recording density is 800 bits per inch (bpi) (315 bits per cm) at 7.5 inches (19 cm) per second. The transfer rate of data to and from the transport is 6 KHz. The read/write head is a dual-gap, single-track head. The dual head allows read-after-write as a check on whether data is written correctly. The transport is equipped with an interlock switch on the lid which disables all functions when the lid is open. The cassette tape cartridge can be inserted with either side up. The capstan motor runs whenever power is applied and a cassette has been properly inserted. The cartridge has a notch which is sensed by the transport to determine which side (A or B) is up. The tape in the cassette cartridge is 0.15 inches (0.38 cm) wide and a minimum of 282 feet (86 m) long.

The transport can be operated in five basic modes of operation:

1. Forward Read
2. Forward Write
3. Reverse
4. Rewind
5. Idle

COMMUNICATIONS COUPLER

The primary purpose of the communications coupler is to provide a link between the host computer and the communications processor (CP). The coupler provides the means for direct core-to-core transfers between the host computer's peripheral processing unit (PPU) and the CP at a 1-MHz character rate. The unit is controlled by both PPU and CP software commands. The coupler also enables the PPU to start, clear and stop the CP.

As shown in figure 2-8, the communications coupler has three interfaces: the host computer's PPU data channel, the CP's Internal Data Channel (IDC) and the CP's Direct Memory Access (DMA) channel.

Information can be transferred between any two interfaces, and between any interface and various data and address registers within the coupler. These transfers occur during the following coupler operations:

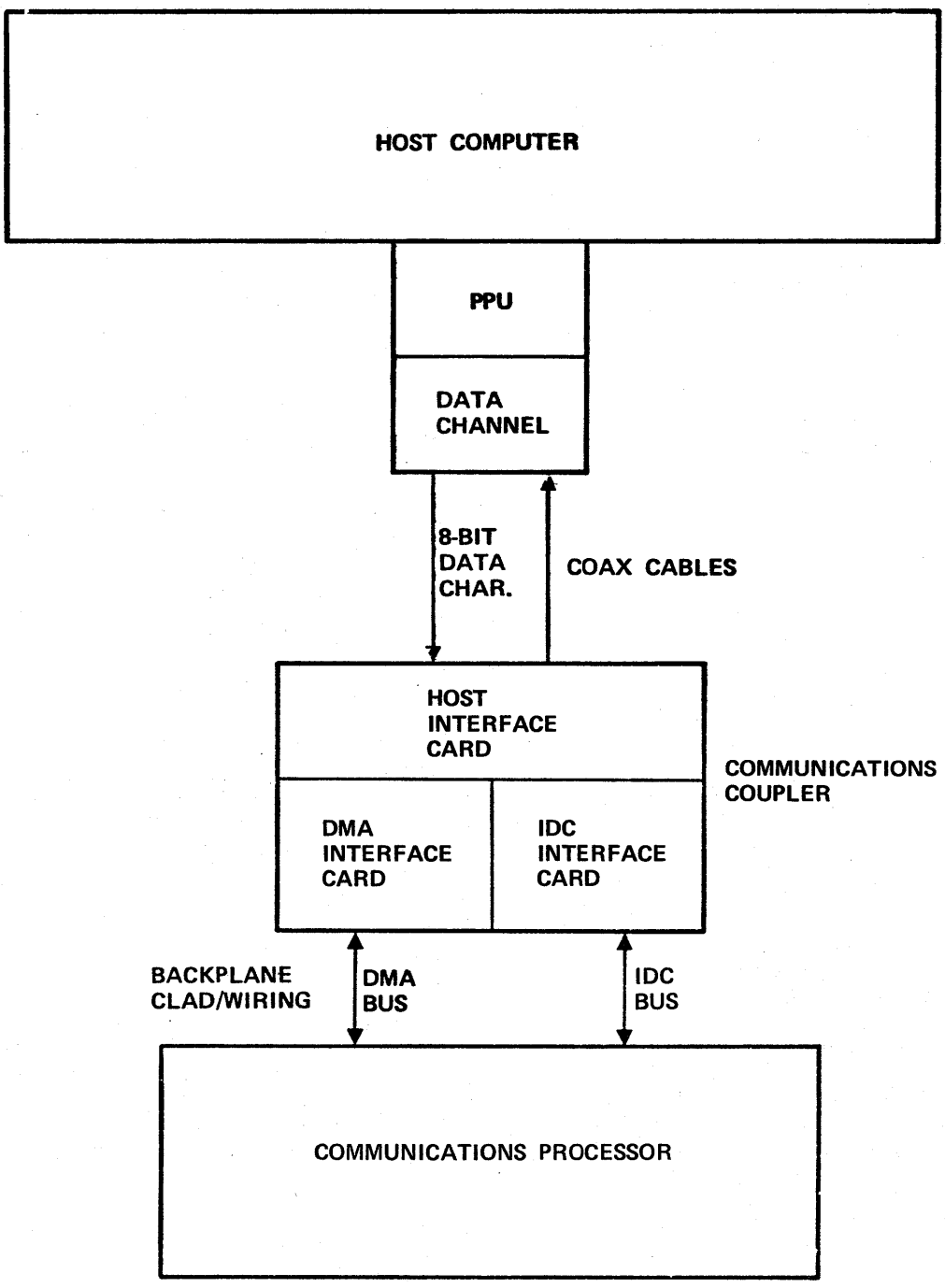


Figure 2-8. Communications Coupler Block Diagram

1. Load coupler registers from PPU
2. Load coupler registers from CP
3. Monitoring status coupler registers from PPU
4. Monitoring status coupler registers from CP
5. Output data characters from PPU
6. Output data characters from CP
7. Input data characters to PPU
8. Input data characters to CP
9. Output program characters from PPU (downline load)
10. Input program characters to PPU (upline dump).

NOTE

In this context, data refers to a series of characters stored in one (or more, if chained) formatted buffers; program refers to a single contiguous block of unformatted memory contents.

Bidirectional transfers using the stated means involve three general types of transfers: single word, multiple word, and formatted data. Single word transfers can be further subdivided into operations and housekeeping. This latter group consists of loading and/or reading internal coupler registers.

Communication between CP and host is through two unidirectional, software-controlled, single-word register transfers. Although their software-controlled nature makes them available for any purpose (the coupler is basically a transparent device), standard usage is to send work requests, status requests, etc., from the host via the order-word register. The host fills this register; the register being full interrupts the CP which then reads the register and takes action as directed by the resident software. Information from the CP to the host utilizes the CP status register (another free-form register). The CP fills the register; the register being full sets a bit in the coupler status word (which the host reads periodically). The host then reads the CP status word.

Multiple word transfers are accomplished in the program mode. Blocks of any size can be moved from host to CP, such as in down-line program loading (a typical CP operation). Similarly, blocks can be moved from CP to host as in up-line dumping. Prior to either operation, the starting address in CP memory is placed in the Memory Address Register (MAR) via a single word transfer from either machine.

Data transfers refer to the movement of characters, in either direction, to or from one (or more) chained, formatted buffer areas. All CP buffers use the same format: first word contains first and last character displacement pointer (referenced from the buffer start point); second word contains various flag bits; last word is the address of the next buffer in the chain (if any); and intervening words are available for characters. Buffer length is established by the CP loading the Buffer Length register; typical buffer lengths are 16, 32, and 64 words (software selectable). The Memory Address Register is loaded with the beginning address of the first buffer, and the coupler fetches (or stores) through the use of the pointer, flags, and chain address(es), until all the information has been moved.

Although these three items constitute the main coupler operations, additional commands have been included, primarily to support diagnostic and maintenance activities.

Structurally the coupler consists of a large number of registers and parts situated on a large data bus. The CYBER channel interface and the DMA interface are each buffered (bidirectionally); all IDC interface operates in real time and is unbuffered. Since all three interfaces can be active simultaneously, bus access requests are resolved via a priority scheme: IDC (highest), DMA, and host (lowest). Once priority is obtained, bus operations take 200 nsec./request except IDC which waits until the transfer to the CP is complete before releasing the bus.

HOST COMPUTER INTERFACE

The communications coupler operates on the 6000, CYBER 70, or CYBER 170 channel - alone or in conjunction with another 2550 system. As such it occupies a "dedicated" channel.

Transfers of orders, status or register parameters utilize the full 12-bit channel width. Data and program transfers, on the other hand, pass 8-bit characters with flags appearing in the upper four bit positions.

Since the channels do not use an interrupt system, the host periodically reads coupler status. Bits within the status word identify the occurrence of CP activities requiring attention of the host.

All CYBER channel signal regeneration occurs on the board which also contains an ON-LINE/OFF-LINE switch (accessible from the front when covers are removed). Placing the unit off-line permits off-line diagnostics to be run on the CP without interfacing with other channel operations.

INTERNAL DATA CHANNEL INTERFACE

The communications coupler operates as an MO5-type peripheral on the internal data channel (IDC). All single word CP transfers use this channel; it is also used for diagnostic purposes in lieu of the CYBER channel (input test, output test operations). Standard equipment address assignment is 'C' (1100), optional coupler equipment address assignment is 'D' (1101).

Several registers and other points can be loaded and/or read in the following paragraphs.

Memory Address Registers

Reading and writing of either data or program on the DMA channel is controlled by a pair of memory address registers, zero and one (MAR0, MAR1). MAR0 is a 9-bit register and holds the most significant addressing bits; MAR1 is an 8-bit register and holds the least significant addressing bits. The effective memory address capability is thus 17-bits (of which only the lower 16 are used in the 2550).

Instructions referring to the MAR (input or output) reference only the lower 16 bits. Access to the seventeenth bit is obtained through the use of the (MAR0) instruction, followed by a MAR instruction.

When operating in the Program mode (input or output), the MAR always points to the desired memory location. However, in the Data mode (input or output), the MAR points to the location of the first word of the buffer; addressing sent to the DMA is the sum of the (MAR) + Δ, where Δ represents a modifier selected for the type of word being accessed.

This modifier is derived either from a constant, zero or one, or from the contents of one or more 8-bit registers: First Character Displacement (FCD) Present Character Displacement (PCD), Last Character Displacement (LCD), and Buffer Length (BL). The PCD is initially loaded with FCD, incremented for each character until PCD = LCD; since PCD is a character address, PCD/2 is used to modify MAR. BL contains the buffer length minus one, used to

locate the chain address to the next buffer FWA.

Information coming from the DMA is stored in two 8-bit registers: FDMAR0 (upper byte) and FDMAR1 (lower byte). These registers are over-written at each access, but not cleared, and thus retain the last DMA output. Flag Mux and Flag Register contain the upper four bits of the host word on output and the upper four bits to the host on input.

The Orderword Register is a 12-bit communications register typically loaded by the host and read by the CP. The CP Status Register is a 16-bit communications register typically loaded by the CP and read (lower 12-bits only) by the host. Other modes are available for diagnostic purposes.

DIRECT MEMORY ACCESS INTERFACE

The communications coupler operates as a low-priority device (compared to the MLIA) on the Direct Memory Access (DMA) Channel. Information is transferred during output program, output data/output test, input program, and input data/input test operations.

Transfers during either input or output program consist entirely of data movements on a word (2-character) basis. Capabilities within the coupler perform the pack/unpack operations to drive the character-oriented host interface. Since the host channel operates at a 1 megacharacter-per-second rate, the DMA channel operates at a 500 kiloword-per-second rate, or one access every two microseconds (average).

Data (or test) operations involve several steps:

Input

1. Fetch and save FCD/LCD; set PCD - FCD.
2. Fetch and save flags.
3. Fetch two characters from CP memory, increment PCD.
4. Send two characters to host.
5. Continue character transfers until all characters sent, exit.
6. If buffer emptied (PCD = LCD) and not last buffer (F3 = 0), fetch chain address (FWA of next buffer) from CP and save in MAR.
7. Resume entire sequence until all data transferred.

Output:

1. Fetch and save FCD/LCD; set PCD = FCD.
2. Test FCD/LCD. If FCD > LCD, fetch chain address of next buffer, save in MAR, and resume; else continue.
3. Fetch two characters from host, assemble into word; increment PCD.
4. Store word in CP memory.
5. Continue character transfer until buffer filled (PCD - LCD), all characters stored (F3 = 1), or until host signals termination of present buffer (F2 - F0 $\neq$ 0).
6. Store FCD/PCD as FCD/LCD in CP memory.
7. Store flags in CP memory
8. If all data transferred, exit; else fetch chain address from CP, and save in MAR.
9. Resume entire sequence until all data transferred.

Data and test operations are essentially identical except that test mode uses the IDC for data in lieu of the host.

PERIPHERAL CONTROLLER

The Control Data 2571-1 Peripheral Controller is an optional product which provides the necessary controls for using a card reader as an input device, and/or a line printer as an output device for the 2550-2 HCP.

CONTROLLER/MICROPROCESSOR INTERFACE

The peripheral controller interfaces to the CP as two standard A/Q channel equipments, a card reader controller and a line printer controller, except that both are contained on the same circuit board and thus require only one I/O slot. The card reader appears as equipment B (1011) and the line printer appears as equipment 4 (0100).

Data transfers are controlled by three means:

1. Status. Continual status read operations can be used to determine when the next transfer can occur.

2. Interrupt. The controller can be configured, under program control, to provide a macrointerrupt whenever the next transfer can occur.

3. ADT. Automatic Data Transfer mode operation can be established wherein the data operations are controlled by firmware in response to ADT microinterrupts.

Standard interrupt assignments are as follows: card reader macro-11, micro-11; line printer macro-4, micro-4.

Four fundamental operations occur. Write data (line printer), read data (card reader), write director (control) function, and read status. Each of these are decoded and passed on to the respective peripheral for action (the controller is essentially transparent). Specific functional operation is discussed in the succeeding sections.

A test mode exists in which line printer data is looped back to become card reader data. This is discussed further in Section 5.

This interface also supports operation of the card reader as a deadstart device. (The card reader/line printer slot must contain the optional deadstart wiring). One and only one deadstart device can be READY at one time; since the deadstart system does not use any form of addressing, any READY device will be dead-started. In a 2550 system containing both cassette and card reader (wired for deadstart), the cassette must be not ready (lid open) to use the card reader, and the card reader must be not ready to use the cassette. To avoid speed overrun, every other column on a deadstart card is left blank (slow-speed reader). Refer to section 3 for operation of the deadstart system.

CONTROLLER/LINE PRINTER INTERFACE

Printers connected to this interface use the preprint paper motion option. Functionally the printer operates as follows:

1. First data character is interpreted as a spacing control character. As soon as it is received, paper motion begins.

2. Succeeding characters are loaded into a memory contained with the printer.

3. When all characters have been loaded, a PRINT command is issued. Contents of printer memory cause the designated character to print on one line at the current position of the paper.

4. At the conclusion of the print cycle, characters are again accepted and the cycle repeats until all information has been printed.

5. Printer conditions are reported back to the controller on separate lines. These conditions control transfer of data characters, set status and/or create interrupts (if requested).

The 1200-LPM chain printer (2570-2) contains an image memory filled from the A/Q channel. Whenever the start pushbutton has been pressed (and no fault condition exists) after the fill image pushbutton has been pressed on the printer, then the next 288 characters from the A/Q channel are treated as type array codes and loaded into the image memory. A status bit marks occurrence of the image load operation.

There are five interrupts available from the line printer controller: Data, EOP, Alarm, Common, and ADT. The common interrupt is active whenever any of the data, EOP, or alarm interrupts are active (logical OR). The five line printer interrupts are brought to an interrupt selection matrix where any two (of the five) may be jumpered to either of two CPU interrupt lines. Standard interrupt assignments place the common interrupt on the macro-interrupt line and the ADT interrupt on the microinterrupt line.

CONTROLLER/CARD READER INTERFACE

Either the 2572-1 or 2572-2 card reader may be connected to this interface. Functionally the card reader operates as follows:

1. A director function is issued to initiate a read cycle for one card. Card motion begins.

2. As each Hollerith character passes the read station, the column image is placed on the twelve read lines (bit 0 = '9', bit 1 = '8', bit 2 = '7',... bit 9 = '0', bit 10 = '11', bit 11 = '12', bits 12-15 = zero).

3. Columnar data is then transferred into the CP, typically under control of the data (macro) interrupt or the ADT (micro) interrupt.

4. The process continues until all columns are read. At the time of the 81st column, EOP (if requested) occurs.

5. Card reader conditions are reported back to the controller on separate lines. These control transfer of data characters, set status and/or create interrupts (if requested).

There are five interrupts available from the card reader controller: Data, EOP, Alarm, Common, and ADT. The common interrupt is active whenever any of the data, EOP, or alarm interrupts are active (logical OR). The five card reader interrupts are brought to an interrupt selection matrix where any two (of the five) may be jumpered to either of two interrupt lines. Standard interrupt assignments place the common interrupt on the macro interrupt line and the ADT interrupt on the micro interrupt line.

CONTROLS AND INDICATORS

The communications console and the maintenance panel contain the majority of the controls and indicators which are used to operate the HCP. The communications console and maintenance panel provide the following basic functions:

1. Master Clear
2. Halt memory operations
3. Start memory operations
4. Step mode
5. Run mode
6. Display memory
7. Load memory
8. Display register
9. Load register
10. Stop, on condition met.

In order to effectively operate the system, it is essential that the user be familiar with the function control register, which is closely related to the operation of the communications console and maintenance panel.

FUNCTION CONTROL REGISTER

The function control register (FCR) provides the basic means of communication between the microprocessor (MP) and the maintenance panel and communications console. It replaces the traditional front panel switches with a series of software-controlled switches and indicators capable of being set or read locally or remotely. It is this approach which provides the capability for remote, unattended operation.

The FCR stores and defines the functions that are to be performed by the MP. The contents of the FCR can be displayed at the console or the maintenance panel in response to an operator's command. The contents thus displayed can then be changed by operator commands from the console or maintenance panel. The maintenance panel is thus "local" operation and the console "remote" operation, but both provide the same level of control.

The FCR is a 32-bit register which is divided into eight digits of four bits each as shown in table 3-1. Bit 00 is the most significant bit (MSB), and bit 31 is the least significant bit (LSB).

The machine status digits (6-7) are set by the CP and indicate machine status such as overflow, macro running, micro running, main storage parity error, protect fault, etc. The machine mode digits (2-5) of the FCR are used to set such conditions as selective stop/on/off, step/run mode, etc. The display digits (0-1) determine which individual registers of two groups of registers (identified in table 3-2) can be displayed or modified.

A typical FCR setting is 442008C0 (A/Q reference) or 712008C0 (main memory reference).

FUNCTION CONTROL REGISTER BIT DEFINITIONS

The following paragraphs describe the bits in the FCR (bit numbers are in decimal notation).

1. Overflow (Bit 31) - this bit is set when an overflow occurs during the execution of an arithmetic operation by the ALU.

2. Protected Instruction (Bit 30) - this bit is set when the CP is executing a Protected instruction, i.e., one in which the protect bit is set.

TABLE 3-1. FUNCTION CONTROL REGISTER - BIT DEFINITIONS AND DIGIT FUNCTIONS

Bit Number		Bit Definition	Digit Number	Digit Function
Decimal	Hexa-Decimal			
31	1F	Overflow	(LSB)	Machine Status
30	1E	Protected Instruction	7	
29	1D	Protect Fault		
28	1C	Parity Error		
27	1B	Interrupt System Active		
26	1A	Auto Restart Enabled	6	
25	19	Micro Running		
24	18	Macro Running		
23	17	undefined	5	Machine Modes
22	16	undefined		
21	15	Enable Auto Display		
20	14	Enable Console Echo		
19	13	Enable Micromemory Write†	4	
18	12	Multilevel Ind. Add. Mode		
17	11	undefined		
16	10	Suppress Console Transmit		
15	0F	Macrobreakpoint	3	
14	0E			
13	0D	BP Interrupt (BP Stop if Clear)		
12	0C	Micro BP, Step, Go, Stop (Macro if Clear)		
11	0B	Step	2	
10	0A	Selective Stop		
09	09	Selective Skip		
08	08	Protect Switch		
07	07	Display 1 (K Function)	1	Display
06	06			
05	05			
04	04			
03	03	Display 0 (L Function)	0	
02	02			
01	01			
00	00		(MSB)	

†Enable Micromemory Write is a hard-wired function, and is not controllable via the FCR.

TABLE 3-2. FUNCTION CONTROL REGISTER - DISPLAY CODE DEFINITIONS

Code		Display 1 (K Function)	Display 0 (L Function)
0	0 0 0 0	FCR	F2 (addressed by N)
1	0 0 0 1	P†	N (MSDs)††
2	0 0 1 0	I	K (LSDs)††
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	F1 (addressed by K, enabled by SM111)
7	0 1 1 1	BP/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 main memory. Automatically incremented after each memory reference.

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

NOTES

The notation: 14_{16} means the number 14 in the hexadecimal (base 16) numbering system.

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.

Unassigned display codes are undefined.

Selection of BP or P-MA causes both BP and P-MA to be displayed. BP consists of the leftmost 16 bits and P-MA consists of the rightmost 16 bits. BP can be modified only if BP is selected; P-MA cannot be modified in either case.

Selection of N or K causes both N and K to be displayed. N is the left 8 bits and K is the right 8 bits. However, when N is selected, only the N register can be modified; when K is selected, only the K register can be modified.

If the program protect line indicates that the instruction is protected, then for all commands (function, status, and data transfers) a reply response is enabled. If the program protect line indicates that the instruction is unprotected, then for each function and data transfer command, a reject is generated; however, a status command is never rejected.

3. Protect Fault (Bit 29) - this bit is set when an attempt is made to access a protected area in main memory from a nonprotected instruction (and the protect system is enabled).

4. Parity Error (Bit 28) - this bit is set when the main memory interface logic detects a parity error while reading data from main memory.

5. Interrupt System Active (Bit 27) - this bit is set when the interrupt system is available for use (regardless of the contents of the interrupt mask register).

6. Auto-Restart Enabled (Bit 26) - this bit is set when automatic program restart is desired after a power failure.

7. Micro Running (Bit 25) - this bit is set when the emulator function (microprogram) is operating in the MP. The bit is 0 when the emulator is halted.

8. Macro Running (Bit 24) - this bit is set when a macroprogram is being executed in the MP. It should be noted that Bit 25 (micro running) must also be set in order for macroprograms to be executed.

9. Enable Auto Display (Bit 21) - this bit is set to allow viewing of a given register (selected by digits 0 and 1) while the CP is running. Auto Display mode is used in conjunction with the maintenance panel to verify that certain registers (e.g., P, A, Q) are changing; it is not normally used with the console.

10. Enable Console Echo (Bit 20) - this bit is set to allow viewing of data being entered by the operator at the communications console (as well as machine responses). It is usually used with teletypewriter input to provide a record of all messages; it is sometimes used with the CRT console and almost never with the maintenance panel itself.

11. Multilevel Indirect Address Mode (Bit 18) - this bit is set when a maximum of 32K words can be accessed in main memory and the MSB denotes indirect addressing. When this bit is 0, 65K words can be accessed. The 32K mode corresponds to CDC-1704 operation; the 65K mode corresponds to 1714, et al, operation and is the commonly used mode (some diagnostics use 32K mode).

12. Suppress Console Transmit (Bit 16) - this bit is set when operational mode transmission from the console to the MP is inhibited. It should be noted that maintenance mode transmission by the console is permitted when this bit is set.

13. Breakpoint (Bits 15, 14) - these bits are used in combination to select three types of breakpoint (BP) as follows:

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

If bit 12 is set, these are all micro breakpoints, else all are macro breakpoints.

14. Breakpoint Interrupt (Bit 13) - when this bit is set, and a breakpoint is reached, the program does not halt immediately, but halts upon reaching an interrupt instruction.

15. Micro BP, Step, Go, Stop (Bit 12) - when this bit is set, the CP is operating in the micro BP, step, go, stop mode only. When this bit is 0, the CP is operating in the macro mode for these functions.

16. Step (Bit 11) - when this bit is set, the program operates in a stepping mode. The operator must trigger the execution of each instruction individually.

17. Selective Stop (Bit 10) - when this bit is set, halt instructions cause instruction execution to cease. When this bit is not set, halt instructions have no affect on program execution.

18. Selective Skip (Bit 09) - when this bit is set, program executions for "skip switch on" are executed. When this bit is not set, program executions for "skip switch not on" are executed.

19. Protect Switch (Bit 08) - when this bit is set, it is possible to write into protected areas in main memory, assuming the protect system is enabled. If the protect system is not enabled, this switch has no affect.

NOTE

All of these actions occur because of controlware operation and not because of any direct hardware control.

MAINTENANCE PANEL

The maintenance panel controls and indicators are described in figure 3-1 and tables 3-3 and 3-4.

COMMUNICATIONS CONSOLE

The communications console is the operator's primary point of interface with the HCP. Either a cathode ray tube (CRT) conversational display or a teletypewriter (TTY) device may be used. Figure 3-2 presents the controls and indicators (including keyboard) for a typical TTY, and figure 3-3 illustrates a typical CRT conversational display. For a complete description of the controls and indicators for these units, the user should refer to the appropriate manual for the communications console selected for the system.

I/O TTY INTERFACE CIRCUIT CARD

The I/O TTY Interface circuit card is illustrated in figure 3-4. This circuit card contains a four-element switch assembly. Each element is a

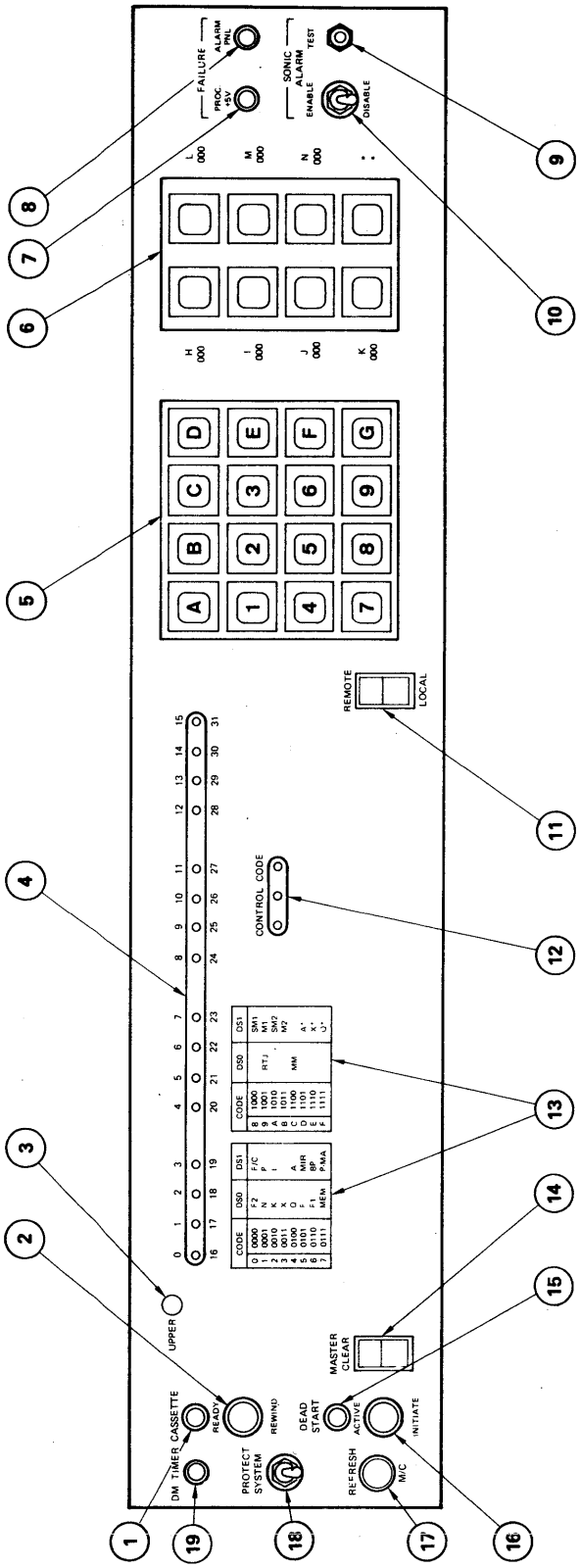


Figure 3-1. Maintenance Panel - Controls and Indicators

TABLE 3-3. MAINTENANCE PANEL CONTROLS AND INDICATORS

Item	Panel Nomenclature	Type	Function
1	CASSETTE REWIND Switch	Momentary Pushbutton	Used to actuate the rewind mode of operation in the tape cassette transport.
2	CASSETTE READY Indicator	LED	Lights when tape cassette transport is ready. This condition exists with a cassette installed, door closed, transport not rewinding and no CRC error. A cassette must be ready to initiate dead-start.
3	UPPER Indicator	LED	Operates in conjunction with the data display indicators. Indicates whether display represents contents of the upper or lower bits in a register. Primary use is the display of the FCR.
4	Data Display Indicators	LED	These 16 indicators are used to display the contents of a selected register in the MP. For registers containing 32 bits, the UPPER indicator is lit when bits 0 thru 15 are being displayed. When the UPPER indicator is not lit, bits 16 thru 31 are displayed.
5	Data Entry Switches	Momentary Pushbutton	These 16 switches are used to enter data in hexadecimal (16-bit) form. These are labeled 0 thru 9 and A thru F.
6	Control Character Switches	Momentary Pushbutton	These eight switches are used to enter control characters H, I, J, K, L, M, N, and the colon (:). (See table 2-4 for Control Character functions.)
7	FAILURE ALARM PNL Indicator	LED	Not Used
8	FAILURE ALARM PNL Indicator	LED	Not Used

TABLE 3-3. MAINTENANCE PANEL CONTROL AND INDICATORS (CONTD)

Item	Panel Nomenclature	Type	Function
9	SONIC ALARM TEST Switch	Momentary Pushbutton	Not Used
10	SONIC ALARM ENABLE/DISABLE Switch	Two-position Toggle	Not Used
11	REMOTE/LOCAL Switch	Two-position Rocker	Used to select either the maintenance panel or the communications console as the point of control for the HCP. When, the switch is set to the local position, the maintenance panel is enabled; when the switch is set to remote, the communications console is enabled.
12	CONTROL CODE Indicator	LED	These three indicators present the last character entered via the control character switches in the control codes listed; a one denotes a lighted LED. Once entered, a control code remains set until replaced by another control code or Master Clear (see table 2-4 for Control Codes).
13	Function Control Definition Table	Decal	Provides operator assistance information such as FCR definition and control character definition.
14	MASTER CLEAR Switch	Momentary Rocker	Switch used to clear memory in CP and peripheral controller.
15	DEAD START ACTIVE Indicator	LED	Used to indicate that the HCP is ready to receive data from a card reader or the tape cassette transport, and the dead-start sequence is in process.
16	DEAD START INITIATE Switch	Momentary Pushbutton	Used to actuate the reading of data by a card reader or the tape cassette transport. Selection of the device to be used is determined by the READY state of the peripherals; only one dead-start peripheral can be READY when using this switch.
17	REFRESH M/C	Momentary Pushbutton	Not used.

TABLE 3-3. MAINTENANCE PANEL CONTROLS AND INDICATORS (CONTD)

Item	Panel Nomenclature	Type	Function
18	PROTECT SYSTEM	Two-position Toggle	Switch is normally set to down position (system not protected). In up position designated portions of memory are protected from over-write during certain diagnostic routines.
19	DM TIMER	LED	Not used.

TABLE 3-4. CONTROL CHARACTER CODE/FUNCTIONS

Character	Display	Function
H (Master-Cleared State	000	When used alone (followed by a colon terminator), causes program execution to halt. The halt affects macroprogram executing or microprogram execution, depending on the setting of bit 12 of the FCR. When used preceding a two-digit number (followed by a colon terminator), causes the corresponding bit of the FCR to be reset.
I	001	When used alone (followed by a colon terminator), causes program execution to be initiated. Affects macroprogram execution or microprogram execution, depending on the setting of bit 12 of the FCR. When used preceding a two-digit number (followed by a colon terminator), causes the corresponding bit of the FCR to be set.
J	010	When used alone (followed by a colon terminator), causes the UPPER indicator to change state, permitting upper and lower portions of the FCR, for example, to be displayed. When used preceding a two-digit number (followed by a colon terminator), causes four bits of the FCR referenced by the first digit (0 thru 5) to be replaced by the value of the second digit.
K	011	When used alone (followed by a colon terminator), causes the contents of the register specified by the Display 1 portion of the FCR to be displayed. When used preceding a 4- or 8-hex digit number (followed by a colon terminator), causes the contents of the register by the FCR Display 1 character to be replaced by the specified digits.
L	100	Operationally the same as K function except that it is associated with Display 0.
M	101	Not presently used.
N	110	Not presently used.

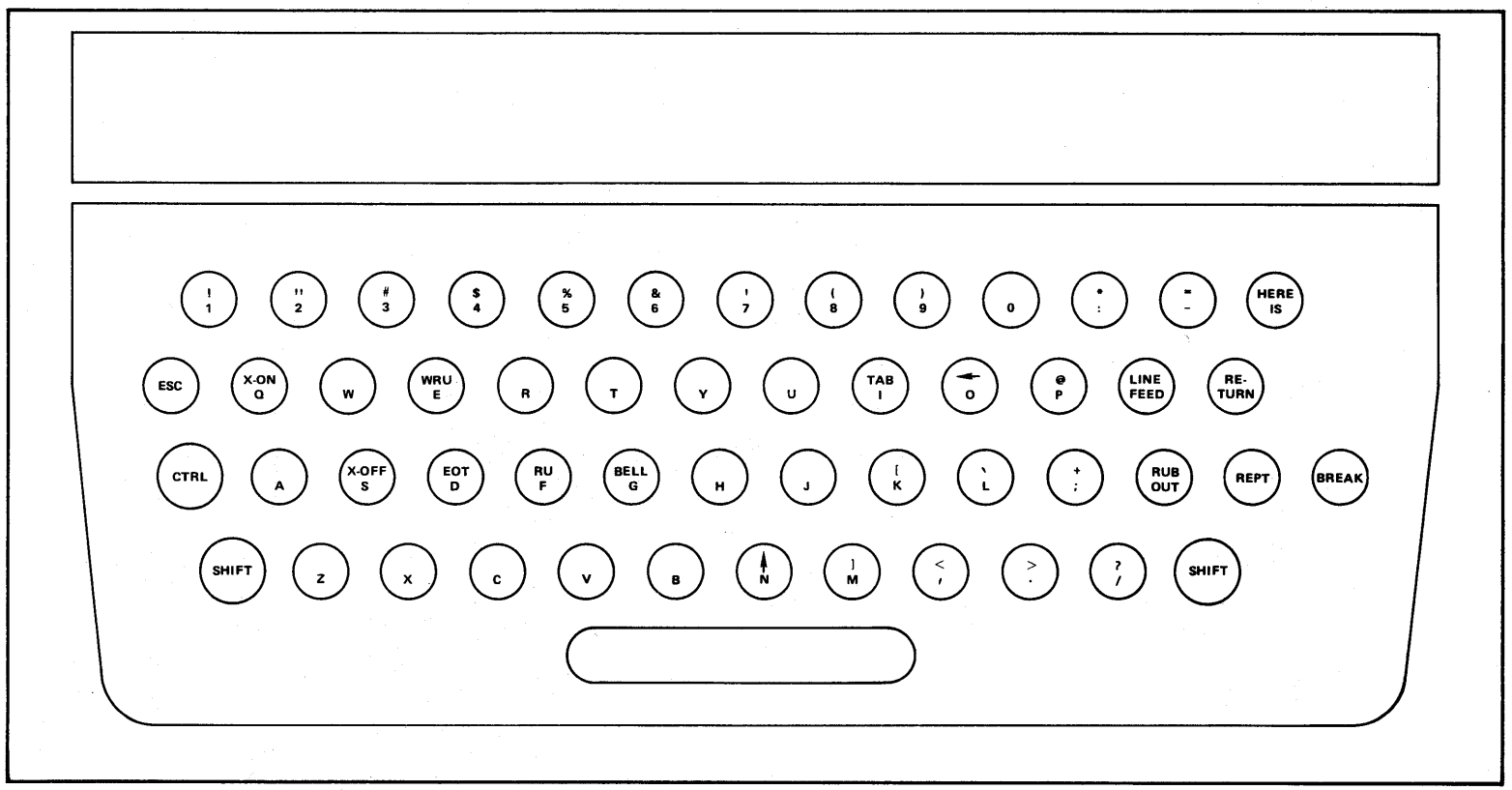
TABLE 3-4. CONTROL CHARACTER CODE/FUNCTIONS (CONTD)

Character	Display	Function
ERROR	111	This code can be used as one indicator that a program failed to load properly.
:	-	Used to terminate all entries. (Note that when using the console in lieu of the maintenance panel, three terminating characters can be used - colon, G, or @.) The @ character also causes control to switch from panel mode to I/O mode.

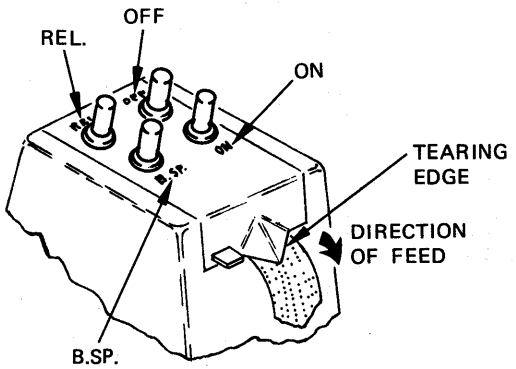
NOTE

When main memory is displayed or entered, the register selected in Display 1 is the main memory address. The Display 1 selection must be the P register. This register is incremented by 1 after the display.

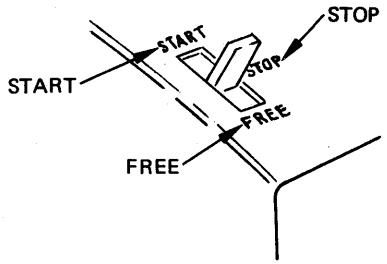
When micromemory is displayed or entered, the K register is the least significant 8 bits of the address, and the N register provides for the remaining bits. The K register is incremented by 1 after the display.



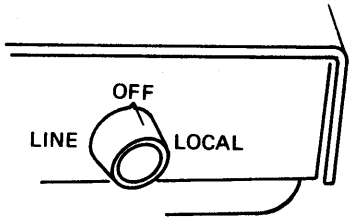
KEYBOARD



AUTOMATIC/MANUAL TAPE PUNCH CONTROLS

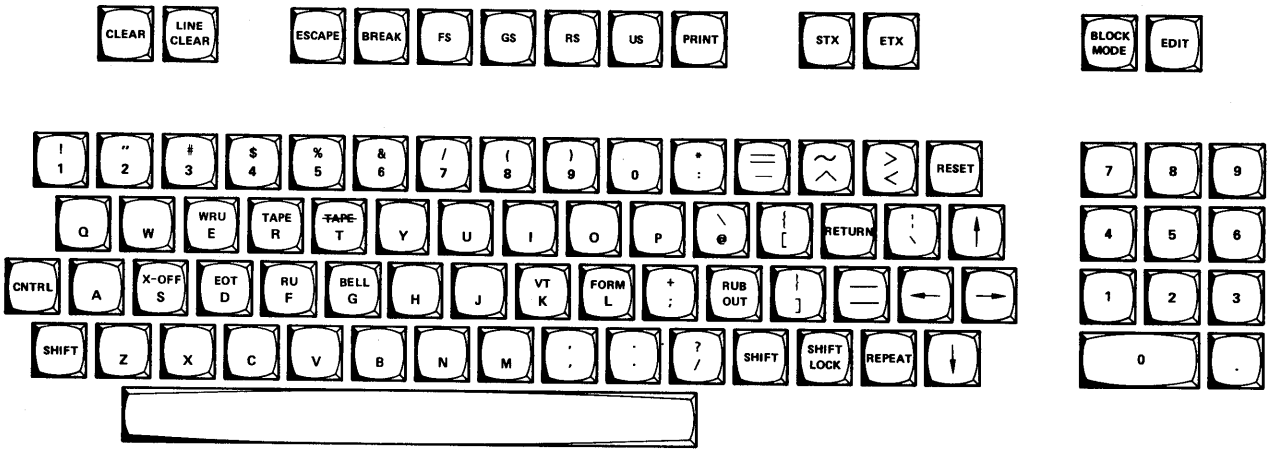


MANUAL READER CONTROLS

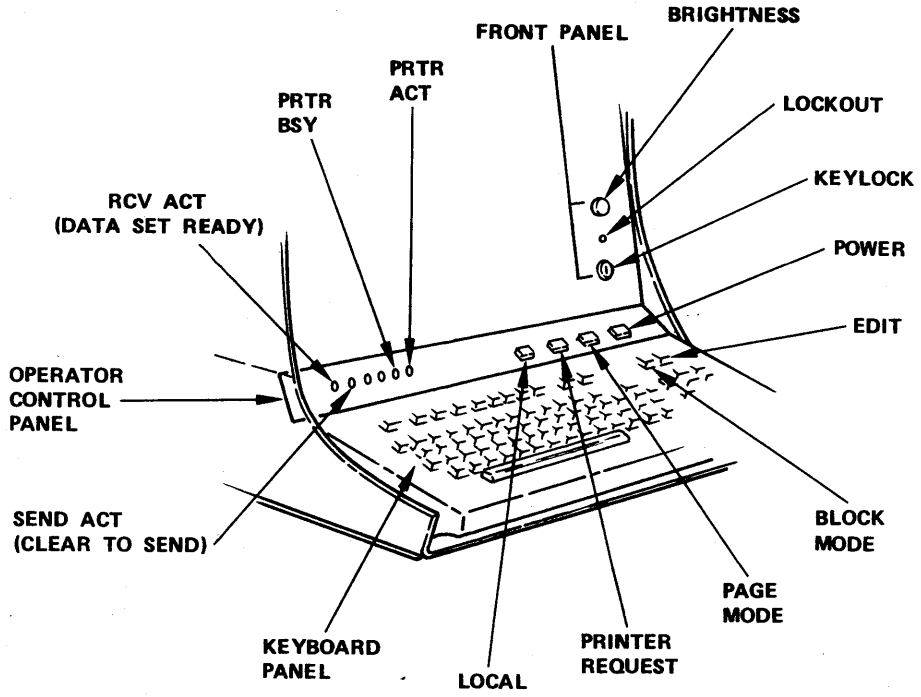


SET CONTROL KNOB

Figure 3-2. Typical Teletypewriter - Controls and Indicators



KEYBOARD



OPERATOR CONTROLS

Figure 3-3. Typical Cathode Ray Tube Conversational Display - Controls and Indicators

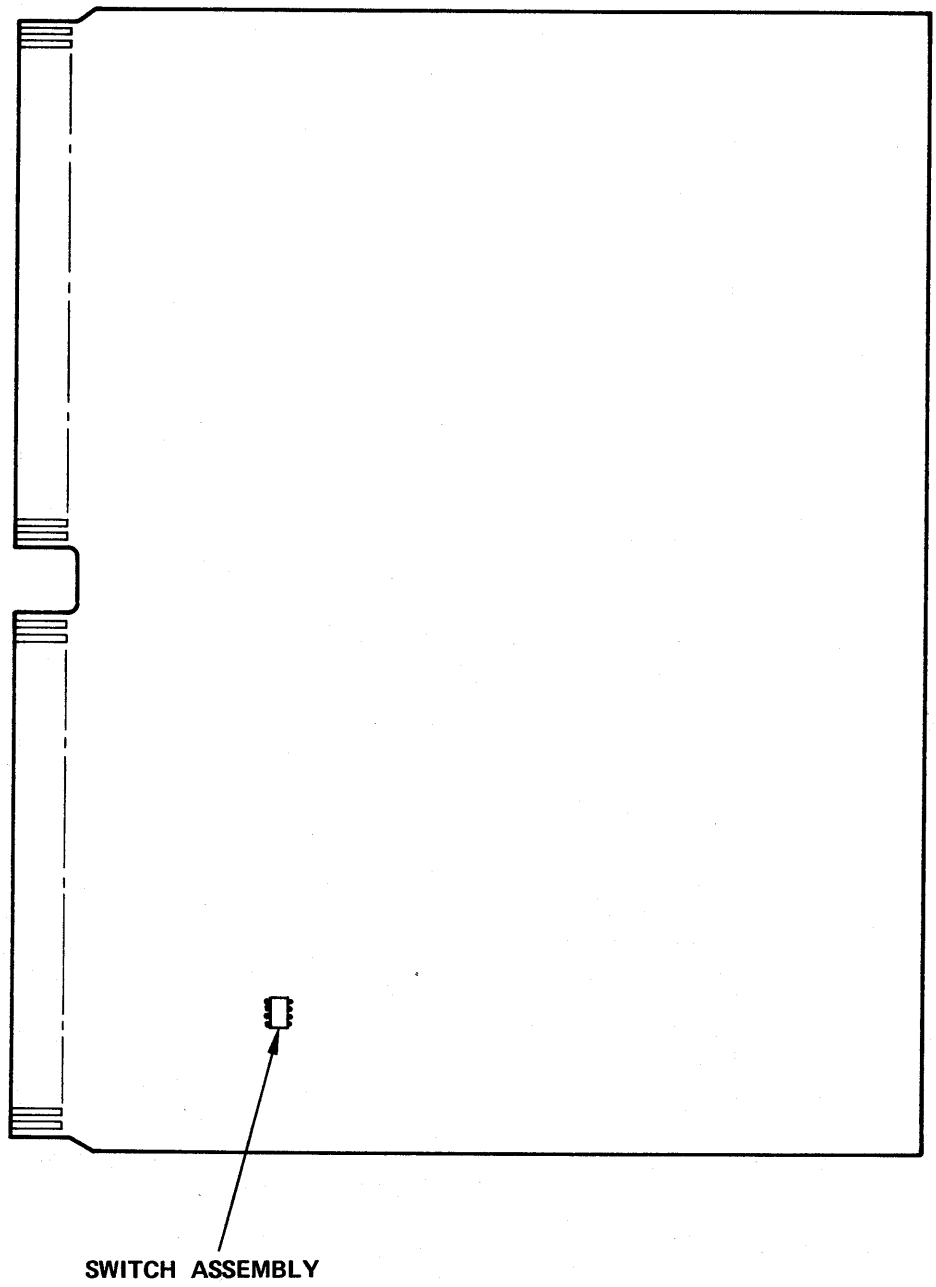


Figure 3-4. I/O TTY Interface Circuit Card - Controls

two-position switch. The switches are used to select the desired baud rate as shown in table 3-5.

TABLE 3-5. SWITCH SETTINGS - I/O TTY INTERFACE CIRCUIT CARD

Dead Start*		Program Select**		Baud Rate
1	2	3	4	
On	On	On	On	110
On	Off	On	Off	300
Off	On	Off	On	1200
Off	Off	Off	Off	9600

*Input Device: Card Reader, 9600; Tape Cassette, 9600.

**I/O Console: TTY, 110; CRT, 110-9600.

NOTE

The switches are set at the time the HCP is installed, and are not reset under normal operating conditions. This information is presented for reference only.

CYCLIC ENCODER

The Cyclic Encoder circuit card is illustrated in figure 3-5. This circuit card contains a single LED indicator which indicates the presence of +5-volt power (when lit).

LM PRINTED CIRCUIT CARD

A narrow vertical control panel is attached to the front-facing edge of each LM printed circuit card (refer to figure 3-6). This panel contains a power switch, four LED indicators, and a circuit legend as described below:

1. PWR Switch. This two-position toggle switch is used to control electrical power to the LM printed circuit card only.

2. IN LOOP CLK Indicator. When lit, this LED indicates that the input loop clock signal is receiving bit clocking from the Multiplex Loop Interface Adapter (MLIA) or preceding LM.

3. IN LOOP DATA Indicator. When lit, this LED indicates that the input loop is receiving data cells from the MLIA or preceding LM.

4. OUT LOOP CLK Indicator. When lit, this LED indicates that the output loop clock signal is receiving bit clocking from the MLIA or preceding LM.

5. OUT LOOP DATA Indicator. When lit, this LED indicates that the output loop is receiving data cells from the MLIA or preceding LM.

6. CIRCUIT LEGEND. This area is provided to permit each user to identify the circuits connected to the CLAs by insertion of small labels. There are no standard markings; customer may use as necessary to meet his particular needs.

NOTE

The four LED indicators are of particular use during maintenance activities should some form of loop problem occur since they provide visual assurance of loop continuity.

MULTIPLEX LOOP INTERFACE ADAPTER

All three of the circuit cards which comprise the multiplex loop interface adapter (MLIA) contain LED indicators.

MLIA – Processor Interface Circuit Card

The MLIA-Processor Interface circuit card is illustrated in figure 3-5. This circuit card contains a single LED indicator which indicates the presence of +5V electrical power (when lit).

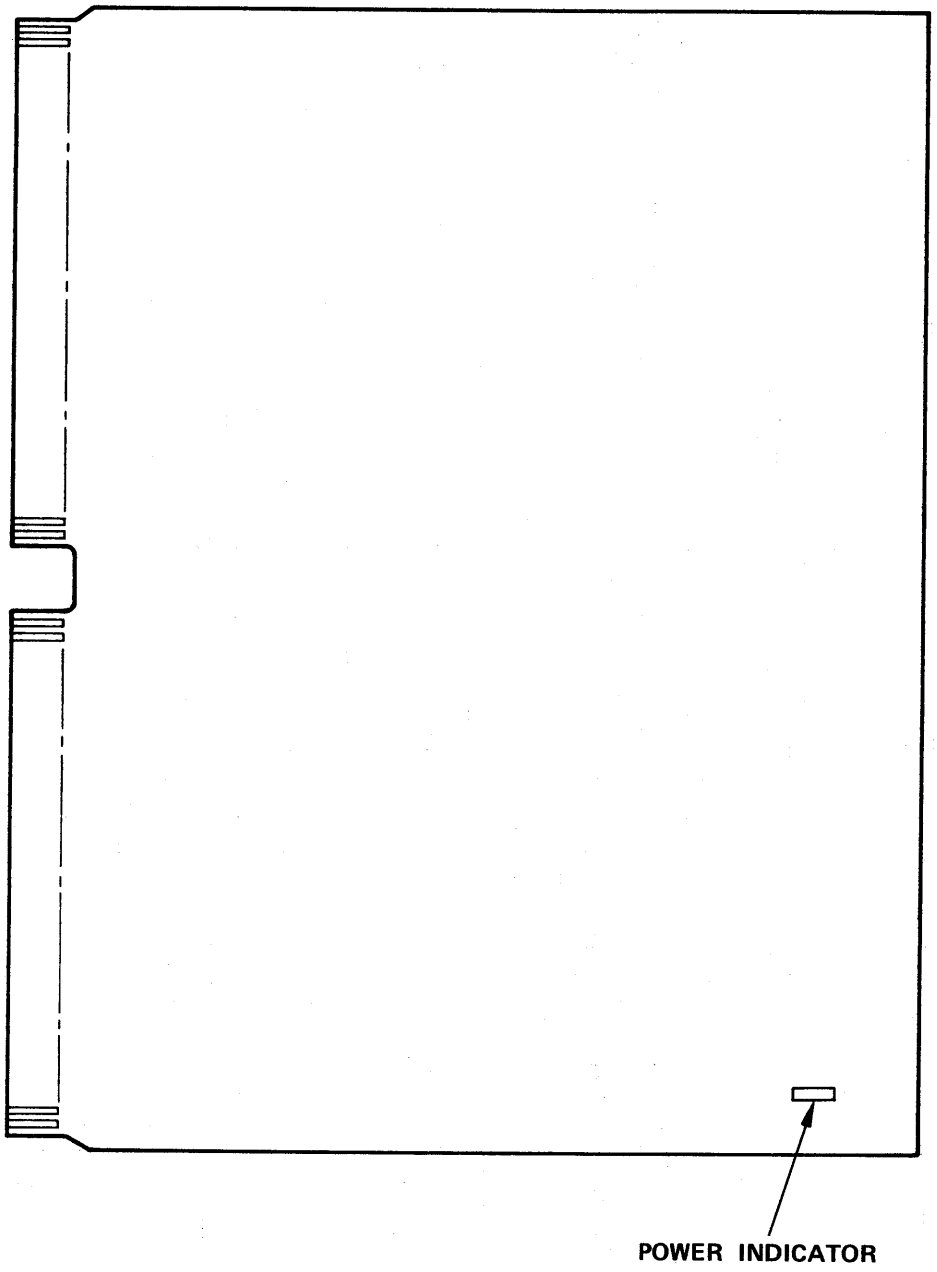


Figure 3-5. Typical Circuit Card - Power Indicator

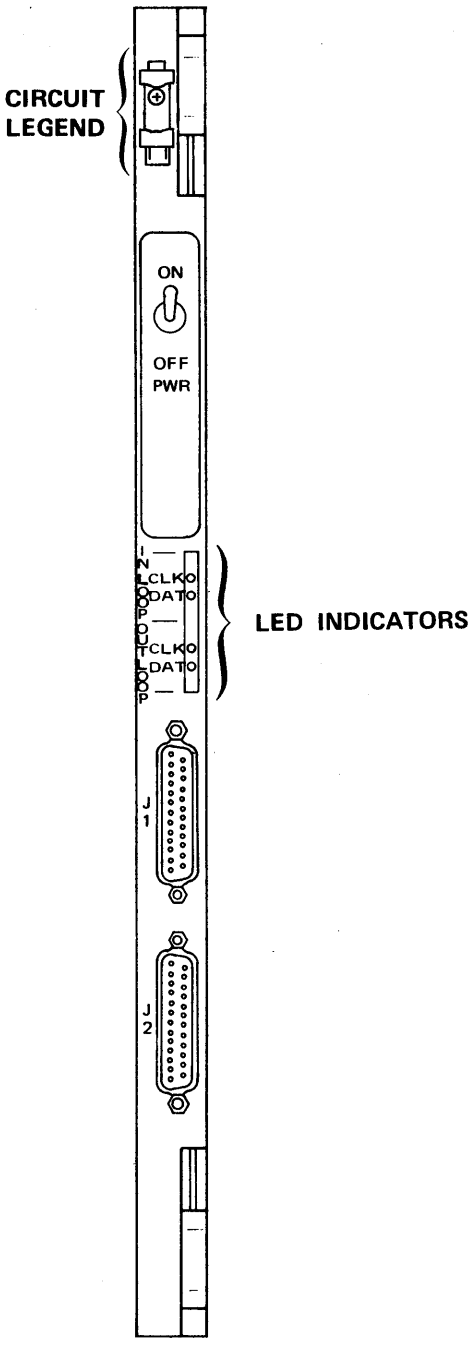


Figure 3-6. Loop Multiplexer Circuit Card - Controls and Indicators

MLIA – Input Loop Interface Circuit Card

The MLIA-Input Loop Interface circuit card is illustrated in figure 3-5. This circuit card contains a single LED indicator which indicates the presence of +5V electrical power (when lit).

MLIA – Output Loop Interface Circuit Card

The MLIA-Output Loop Interface circuit card is illustrated in figure 3-6. This circuit card contains the following five LED indicators (the numbered items below correspond to the numbered callouts in figure 3-7):

1. MLIA RUNNING. When lit, this LED indicates that the MLIA is not reset.

2. OUTPUT ON. When lit, this LED indicates the presence of data and clock signals on the output loop.

3. INPUT ON. When lit, this LED indicates the presence of data and clock signals on the input loop.

4. INPUT BUSY. When lit, this LED indicates either the presence of loop batches on the input loop, or the absence of a valid Restart Loop End following an error.

5. POWER. When lit, this LED indicates the presence of +5V electrical power.

COMMUNICATIONS COUPLER

The communications coupler, host interface circuit card is illustrated in figure 3-8. This circuit card contains the following controls and indicators:

1. CLOCK ADJUSTMENT POTENTIOMETER. This potentiometer can be used to adjust the phase of the clock signal provided by the host computer. This control is adjusted during HCP installation and normally needs no readjustment. It should be adjusted only by qualified maintenance personnel.

2. ONL CLK PRESENT Indicator. When lit, this LED indicates that the on-line clock signal from the host computer is available. The LED continues to indicate the availability of the on-line clock, even if the ON-LINE/OFF LINE switch is set to the OFF-LINE position. When lit, this LED also indicates the presence of +5V electrical power. The indicator is not lit if either the on-line clock is not available, or if electrical power is not present. It is a valuable indication of cable integrity back to the host.

3. ON-LINE/OFF-LINE Switch. This two-position toggle switch is used to connect the HCP to the enabled host data channel. When the switch is set to the ON-LINE (up) position, the HCP is connected to the host channel. When the switch is set to the OFF-LINE (down) position, the communication links with the host data channel are disabled. Normally the switch is in the ON-LINE position except when off-line diagnostics are being run.

4. SWITCH Assembly. This is an eight-element switch assembly; each element is a two-position rocker switch. Elements 1, 2 and 3 are used to select an equipment code (address) for the communications coupler; elements 4, 5, 6 and 7 are not used; and element 8 is used for parity. Table 3-6 presents the functions for these switches. This switch assembly is set at the time the HCP is installed, and is not reset under normal conditions.

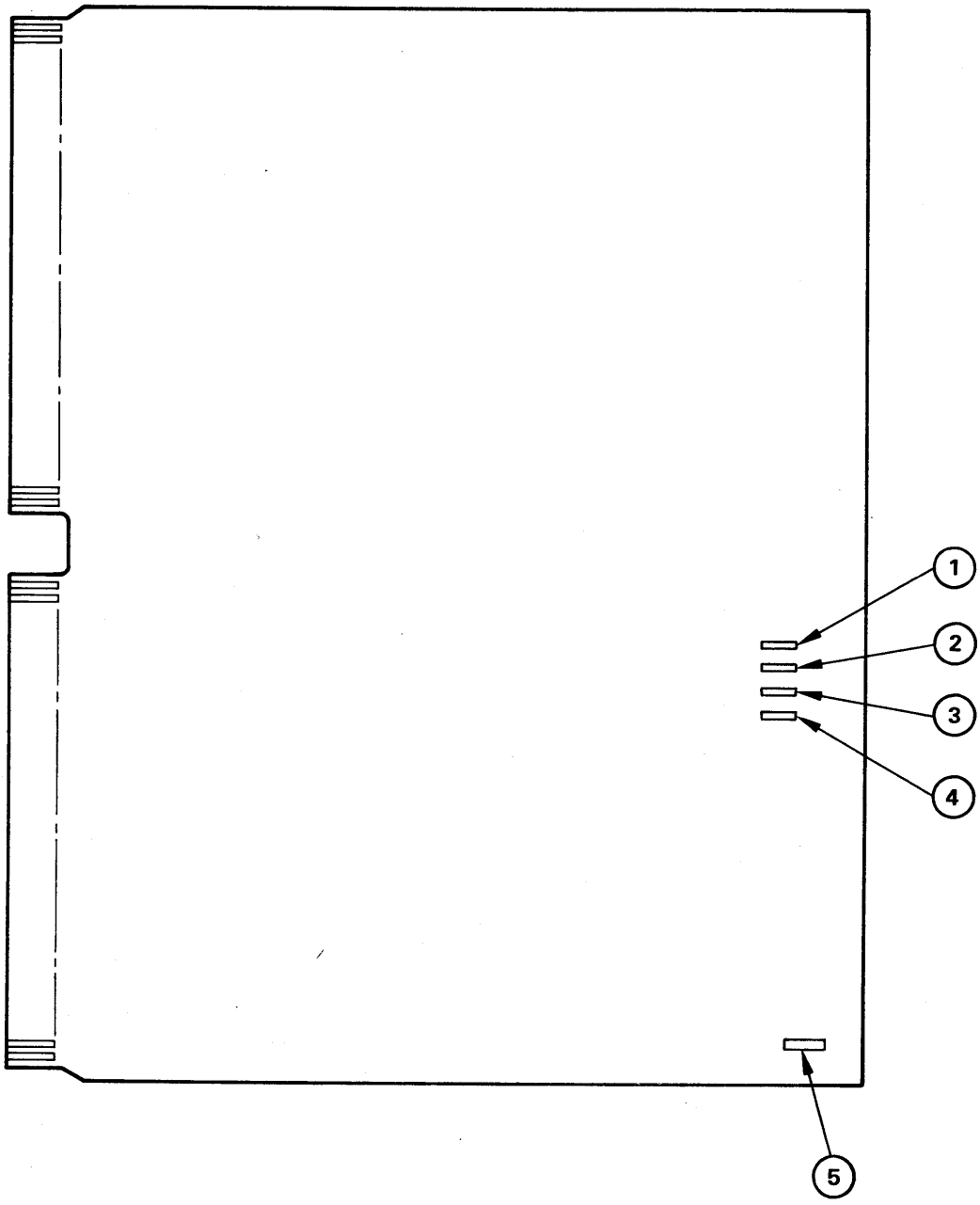


Figure 3-7. MLIA, Output Loop Interface Circuit Card - Indicators

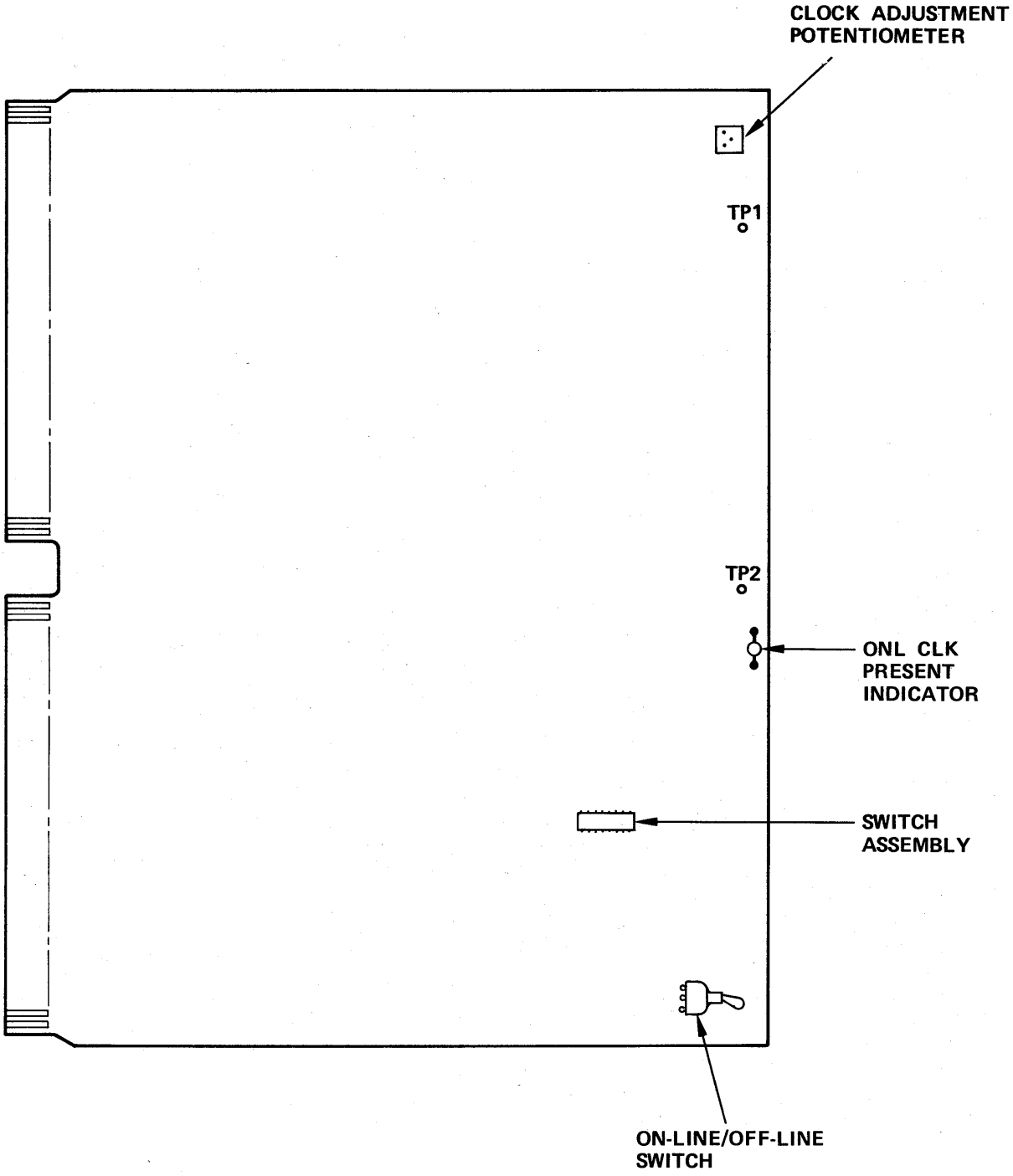


Figure 3-8. Communications Coupler, Host Interface Circuit Card - Controls and Indicators

TABLE 3-6. SWITCH SETTINGS - COMMUNICATIONS COUPLER HOST INTERFACE CIRCUIT CARD

1	2	3	Equip.† Code	4	5	6	7	8	Parity
On	On	On	0	Not Used	Not Used	Not Used	Not Used	On	Channel Parity Disabled (6000, CYBER 70)
Off	On	On	1						
On	Off	On	2						
Off	Off	On	3						
On	On	Off	4						
Off	On	Off	5					Off	Channel Parity Enabled (CYBER 170)
On	Off	Off	6						
Off	Off	Off	7						

†The standard equipment code assignment is seven (111).

MICROMEMORY

The micromemory circuit card is illustrated in figure 3-9. This circuit card contains a four-element switch assembly. Each element is a two-position switch. The switch is used for micromemory page selection as shown in table 3-7.

It should be noted that there are two configurations of the micromemory circuit card which are commonly referred to as Version 1 (V1) and Version 2 (V2). For V1, the switches are numbered in the sequence: 1, 2, 3, 4 (from top to bottom). For V2, the switches are numbered in reverse order: 4, 3, 2, 1 (from top to bottom). Numbering sequence depends on the vendor switch used. The terms top and bottom used above are applicable to the circuit card as normally installed, and as shown in figure 3-9.

TABLE 3-7. SWITCH SETTINGS - MICROMEMORY PAGE SELECT

Switch No.				Micromemory Pages Selected
(V1) 1 (V2) 4 Top	2 3	3 2	4 1 Bottom	
Off	Off	Off	N/A	0-3
Off	On	Off	N/A	4-7
On	Off	Off	N/A	8-11
On	On	Off	N/A	12-15
No other switch settings are used.				

NOTE

The switches are set at the time the HCP is installed, and are not reset under normal conditions. This information presented for reference only.

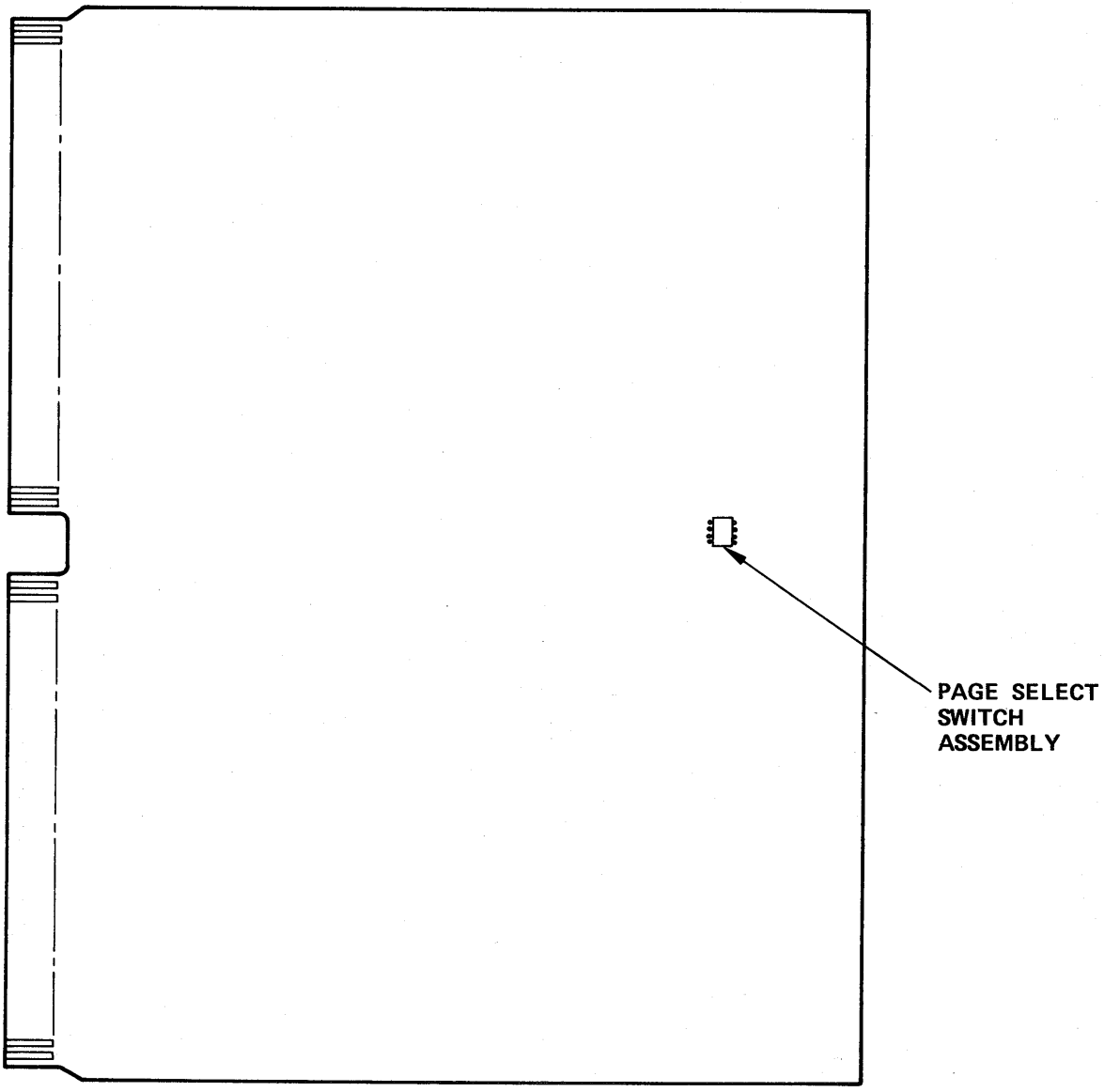


Figure 3-9. Micromemory Circuit Card - Controls

TAPE CASSETTE CONTROLLER

The tape cassette controller circuit card is illustrated in figure 3-10. This circuit card contains the following two switch assemblies:

1. EQUIPMENT SWITCH (S1). This switch assembly consists of four segments; each segment is a two-position switch. The switches are used to set the equipment code (address) for the tape cassette controller. The switches are set at the time the HCP is installed, and are not reset during normal operating conditions. Standard equipment address assignment is hex 7 (0111).

2. CONTROL FUNCTIONS SWITCH (S2). This switch assembly consists of four segments; each segment is a two-position switch. The control functions provided by the switches are listed in table 3-8. The switches are set at the time the HCP is installed, and are not reset during normal operating conditions.

   a. Auto-Rewind Enable Switch (Segment No. 1)

      When this switch is ON, the transport automatic-rewind feature is operative. Thus each time a cassette is inserted and the lid is closed, the tape is automatically rewound to the Beginning of Tape (BOT) hole. When this switch is OFF, the automatic-rewind feature is not operative. However, rewind function commands are always operative.

      In the DISABLED position, the switch facilitates editing operations, and permits sequential loads of diagnostic overlays from more than one deadstart device.

   b. Protect Switch (Segment No. 3)

      When this switch is in the UNPROTECTED position (switch is OFF), the controller ignores the state of the program protect line from the MP, and accepts all I/O instructions (function, status, and data transfer commands). An unprotected controller never rejects a function, status, or data transfer command.

      When PROTECTED (switch ON), the controller monitors the program protect line from the MP. The controller's response to the I/O instruction is then determined by the state of the program protect line as follows:

TABLE 3-8. CONTROL FUNCTIONS (S2) - TAPE CASSETTE CONTROLLER

Position	Segment No. 1	Segment No. 2	Segment No. 3	Segment No. 4
ON	Auto-Rewind Enabled	(not used)	Protected Mode	(not used)
OFF	Auto-Rewind Inhibited	(not used)	Unprotected Mode	(not used)

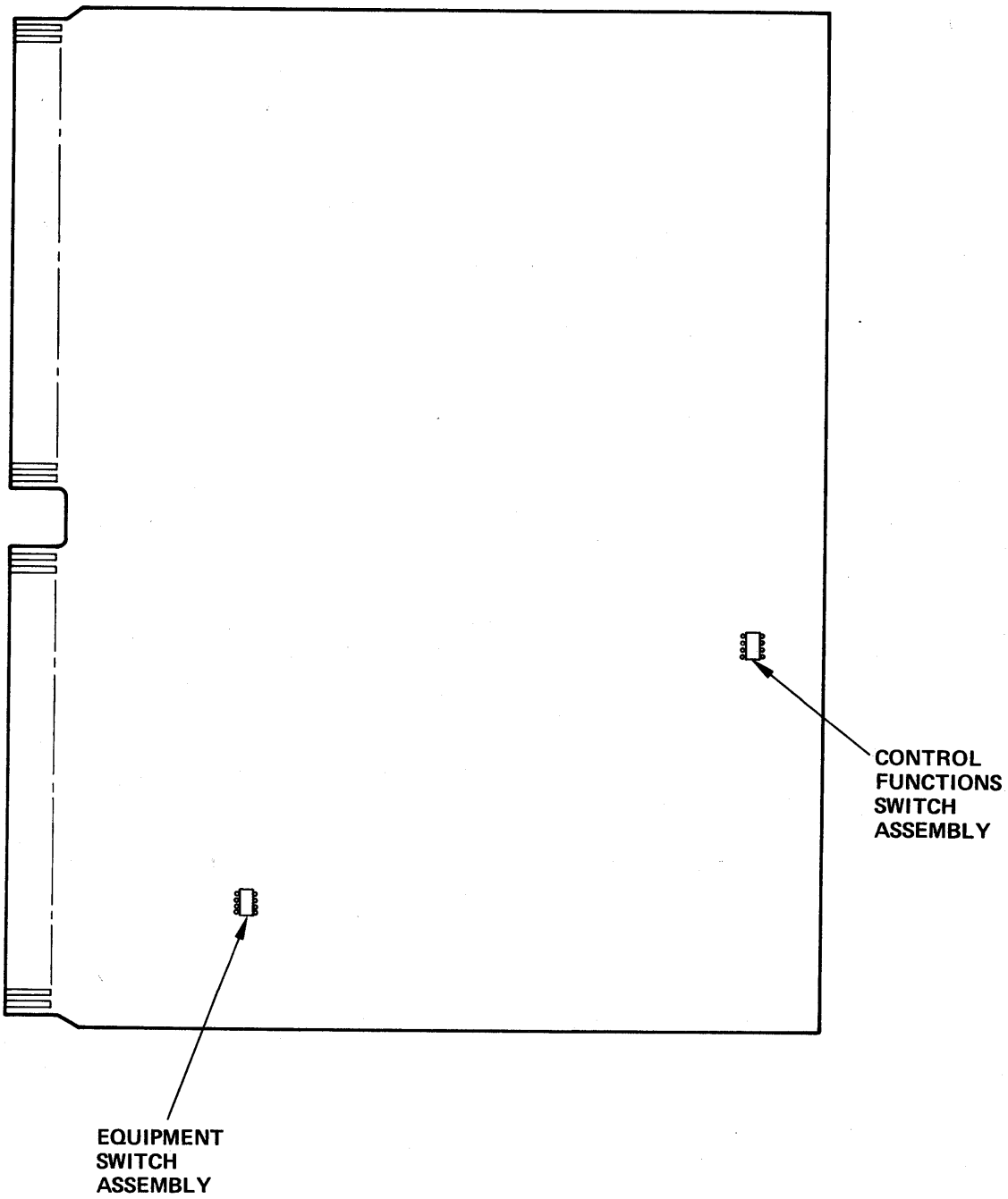


Figure 3-10. Tape Cassette Controller - Controls

(1) If the program protect line indicates that the I/O instruction is unprotected, then for each function and data transfer command, the controller generates a reject. It should be noted, however, that a status command is never rejected.

(2) If the program protect line indicates that the I/O instruction is protected, then for all commands (function, status, and data transfers) a reply response is enabled by the protect condition. The controller does not return a reply until all other conditions within the controller indicate that it is permissible to do so.

TAPE CASSETTE TRANSPORT

The tape cassette transport contains the following two controls:

1. Transport Lid Switch. This is an interlock switch which is depressed by the transport lid when closed. When the lid is open, all transport functions are disabled. Opening the lid is the primary means for making the cassette NOT READY (refer to dead-start operation).

2. Side A/B Switch. This switch senses a notch in the tape cassette to determine which side (A or B) is up. Position of this switch is reported as a status bit.

NOTE

There are no operator controls (in the usual sense) on the transport. All motion is directed by the controller.

ELAPSED TIME INDICATORS

An elapsed time indicator is located on the inner floor of the HCP. The indicator is accessible from the rear of the HCP as shown in figure 3-11.

The elapsed time indicator is a six-wheel decimal counter which displays cumulative operating time for the HCP. The indicator operates only when the HCP is connected to the appropriate source of electrical power, and the Main AC Power Circuit Breakers are set to the ON position. Time is displayed in hours and tenths-of-hours; the maximum possible reading of the indicator is 99,999.9 hours.

Readings are normally taken when circuit cards are changed, or when maintenance procedures are performed. Readings are normally entered in an operational or maintenance log.

CIRCUIT BREAKERS AND FUSES

Main power circuit breakers are located on the inner floor of the HCP as shown in figure 3-11. These three, ganged circuit breakers can also be used as main switches to control AC power to all circuits in the cabinet bay.

Additional circuit breakers and fuses are located on each of the power supplies shown in figure 3-11. The power supplies are mounted on hinges, and can be swung outward to provide ready access to these components.

The communications processor power supply contains the following components:

1. Fuse - 6.25 amp, Slo-Blo

2. Fuse - 2.5 amp, Slo-Blo

3. OV ADJ - potentiometer to adjust the $\pm$12 to $\pm$15 voltage output.

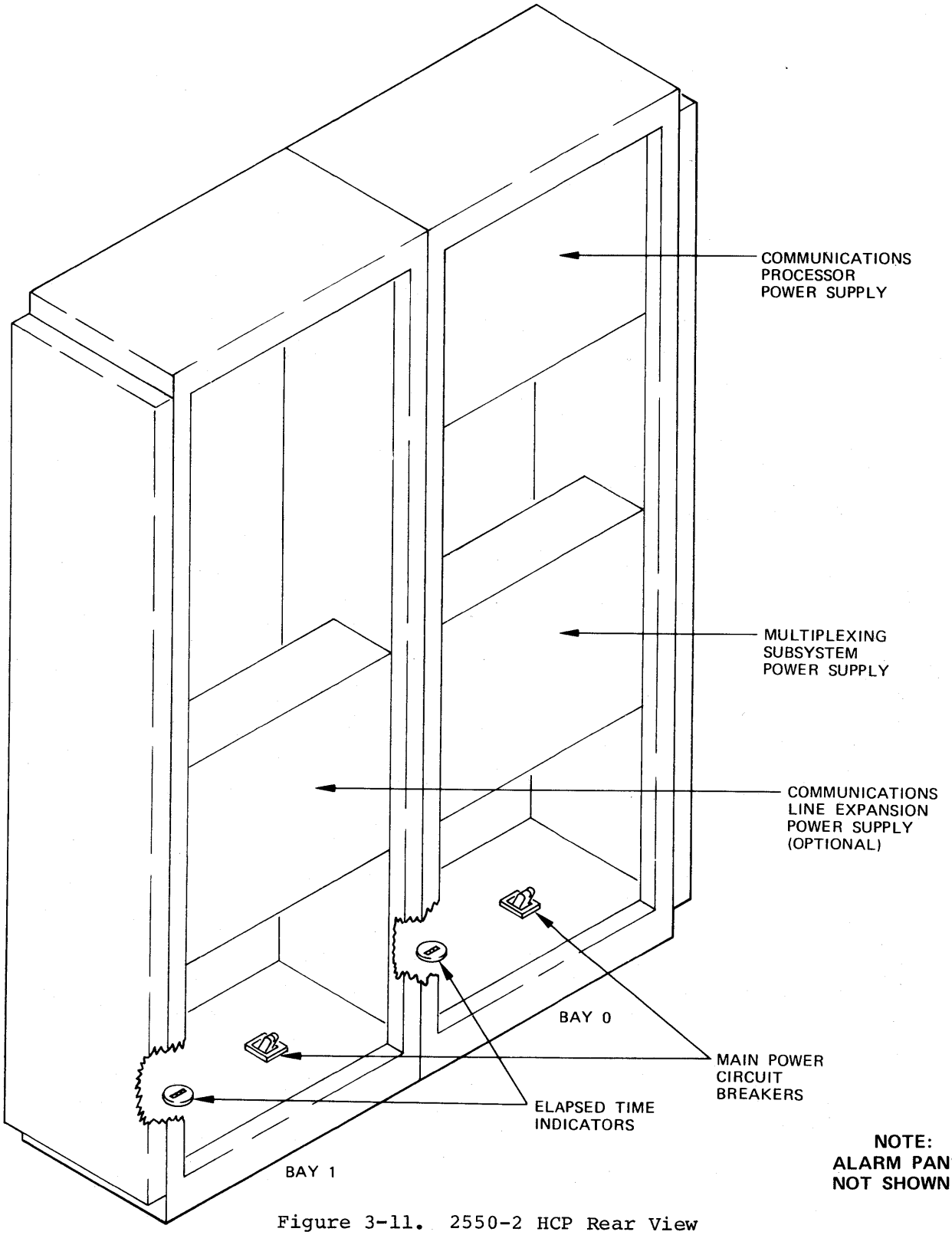


Figure 3-11. 2550-2 HCP Rear View

The multiplexing subsystem power supply contains the following components:

1. Fuse - 6.25 amp, Slo-Blo

2. Circuit Breaker - 5 volt, 111 amp.

NOTE

Circuit breakers are normally left ON, and are not used as operator controls.

START-UP PROCEDURE

NOTE

The HCP is normally powered-up, except for motor-driven peripherals which are turned on only as needed. However, the following procedure is to be used when the HCP has been completely shut off (for example, following an emergency shut-down).

1. Set the main AC power circuit breakers at the rear of the HCP to the ON position.

2. Set the circuit breaker on the loop multiplexer (LM) power supply to the ON position.

3. Set the PWR switches on all LM circuit cards to the ON position.

4. Turn on all peripherals and auxiliary power units.

5. Verify proper illumination of all LED indicators, on the CLA circuit cards, the LM circuit cards and the maintenance panel.

NOTE

Start-up sequence for circuit breakers is not critical.

OPERATING PROCEDURES

MASTER CLEAR

The HCP can be master-cleared by any of the following actions:

1. Depress the question mark (?) key on the communications console. If a TTY is to be used as the input device, first depress the ESCAPE key. On TTY units not having an ESCAPE key, this function can be generated by depressing SHIFT, CONTROL, and K simultaneously.

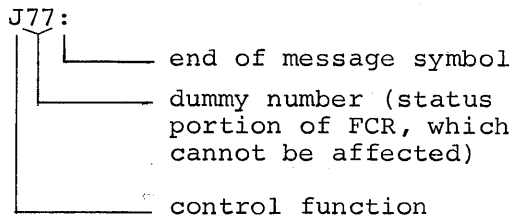
2. Depress the MASTER CLEAR switch on the maintenance panel.

3. Receipt of an external master clear signal from the host computer.

4. A power-on master clear.

DISPLAY FCR CONTENTS

To display the contents of the FCR at any time, enter:

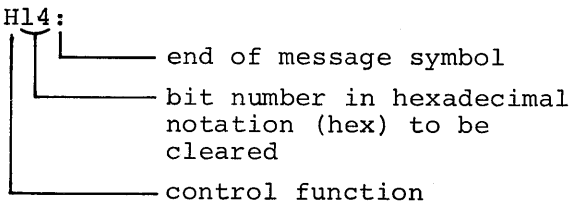


NOTE

Whenever a printout occurs, the initial symbol is the last Control Function entered. When the same Control Function is required, it need not be reentered. The stored value is changed by Master Clear (to 000) or Error (to 111).

CLEAR BIT IN FCR

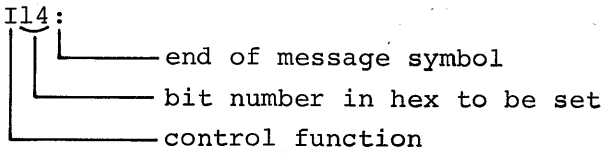
To clear a single bit position in the FCR, enter:



The updated FCR is displayed at the console.

SET BIT IN FCR

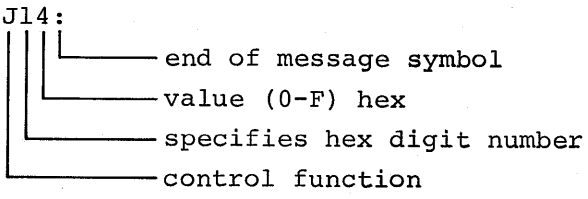
To set a single bit position in the FCR, enter:



The updated FCR is displayed at the console.

CHANGE FCR IN HEX DIGIT MODE

To change the contents of an FCR hex digit (0-5), enter:



The above entry specifies an FCR change is to be made in the digit mode, and updates FCR hex digit No. 1 to a value = 4 hex (0100). To determine the meaning of the value the user must refer to Table 3-2, where digit #1 = display 1, and a count code of 4 = "A" register. The response to the console is the updated FCR value (which is always displayed in 8 hex digits).

CHANGE FCR IN BIT MODE

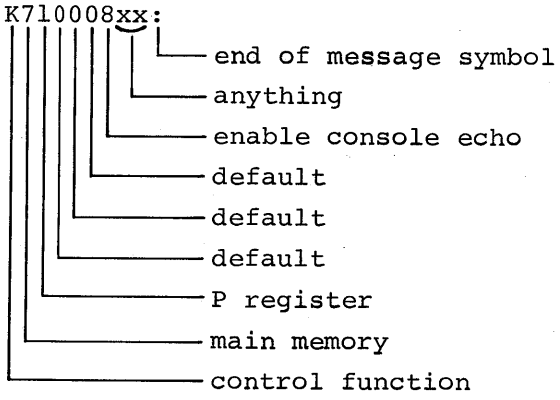
The operator cannot change the contents of FCR digits 6 and 7, as they are readouts of machine status. The remaining hex digits 0-5 can all be changed with the same command from the console, as follows:

a. ESCAPE key (or SHIFT/CTRL/K) (Panel Mode)

b. ? (Master Clear)

c. ESCAPE key (Panel Mode)

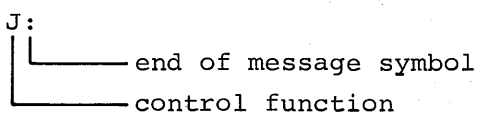
d. To change hex digits 0-5 together proceed entering digits per this example:



Default is the prior value of the FCR that is not to be changed (in this case zeros). The response to the console is the updated FCR.

TOGGLE UPPER INDICATOR

To toggle the UPPER indicator on the maintenance panel (alternately display the upper and lower 16 bits of the FCR), enter the following:

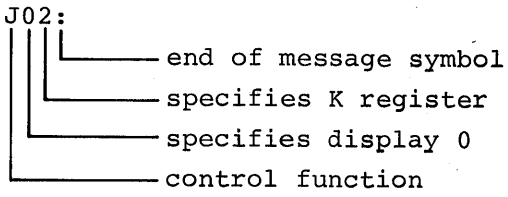


DISPLAY REGISTER DEFINED IN DISPLAY 0

The operator can display the contents of any of the registers defined in Display 0 (see table 3-2) by using the following procedure.

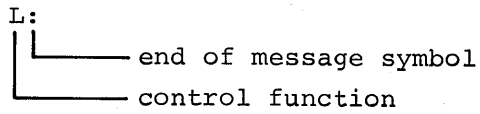
Example: Assume the operator wants to display the contents of the K register at the console.

1. The element to be displayed must first be defined in the FCR. The console input is:



The above input specifies FCR change is in the hex digit mode and defines the K register in display 0. The response to the console is the updated FCR.

2. The operator then types a command from the console to Display 0:



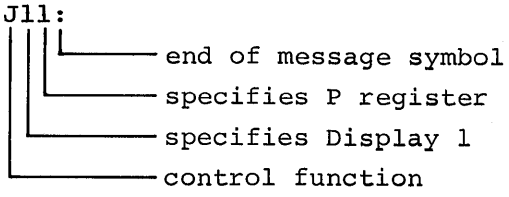
The response to the console is the present value of the K register.

DISPLAY REGISTER DEFINED IN DISPLAY 1

The operator can display the contents of any of the registers defined in Display 1 (see table 3-2) by using the following procedure.

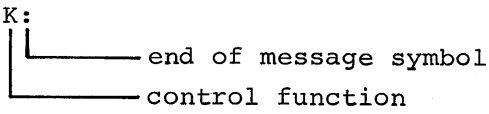
Example: Assume the operator wants to display the contents of the P register at the console.

1. The element to be displayed must first be defined in the FCR. The console input is:



The above input specifies FCR change is in the hex digit mode and defines the P register in Display 1. The response to the console is the updated FCR.

2. The operator then types a command from the console to Display 1:



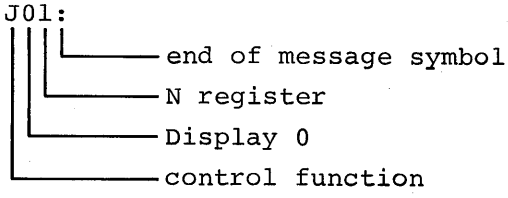
The response to the console is the present value of the P register.

LOAD REGISTER DEFINED IN DISPLAY 0

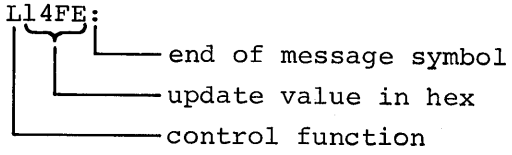
The operator can load a value into any register defined in Display 0 (see Table 3-2) by using the following procedure.

Example: Assume the operator wants to load $14FE_{16}$ into the N register.

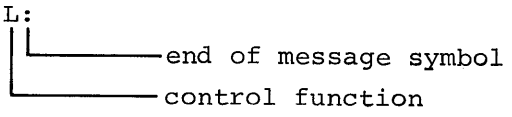
1. The element to be loaded must first be defined.



2. The update value is then entered.



3. Verify that the new value has been transferred into the desired register.

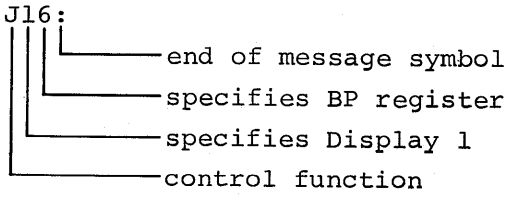


LOAD REGISTER DEFINED IN DISPLAY 1

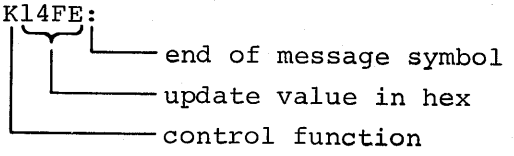
The operator can load a value into any register defined in Display 1 (see Table 3-2) by using the following procedure.

Example: Assume the operator wants to load $14FE_{16}$ into the breakpoint (BP) register.

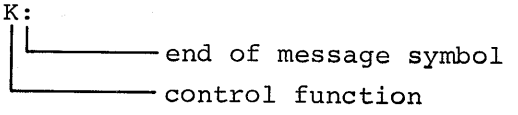
1. The element to be loaded must first be defined.



2. The update value is then entered.



3. Verify that the new value has been transferred into the desired register.



TAPE CASSETTE AUTOLOAD

1. Master clear the HCP using any of the methods described earlier in this section.

2. Insert tape cassette in transport. Ensure that the lid on the tape transport is closed securely; this enables transport operation by depressing the lid switch. Tape moves to Beginning of Tape (BOT).

3. Press the ESCAPE key on the communications console.

4. Press the DEADSTART switch on the maintenance panel. The tape loads and processor starts.

NOTE

When loading any sort of program, it is always good practice to observe the value displayed in the three Control Code LED indicators (Item 12, figure 3-1). Any time all three indicators are lit simultaneously an error is indicated. Frequently the final value of the Control Code indicators is the Error condition (all on) if an error was sensed anywhere in the load operation. This applies regardless of the type of autoload device used.

CARD READER AUTOLOAD

If system is equipped with 2571-1 Peripheral Controller and a card reader, the following procedure may be used:

1. Master clear the HCP using any of the methods described earlier in this section.

2. Place the card deck in the card reader. Make card reader READY.

3. Press the ESCAPE key on the communications console.

4. Press the DEADSTART switch on the maintenance panel. Cards feed in and the processor starts.

HOST COMPUTER AUTOLOAD

The host computer autoload is a software function which provides a DOWNLINE LOAD command to the HCP. In order for this command to be received, the ON LINE/OFF LINE switch located on the communications coupler, host interface circuit card, must be set to the ON LINE position.

START PROCESSOR

The HCP can be started from the console by entering the following command:

```
I:
||└──────end of message symbol
└└──────control function
```

If bit $0C_{16}$ (12_{10}) of the FCR is 0, both micro and macroprograms begin running. If bit $0C_{16}$ (12_{10}) of the FCR is 1, only microprograms begin running.

STOP PROCESSOR

The HCP can be stopped from the console by entering the following command:

```
H:
||└──────end of message symbol
└└──────control function
```

If FCR bit $0C_{16}$ (12_{10}) is 0, then a macro stop occurs.

If FCR bit $0C_{16}$ (12_{10}) is 1, then a micro stop occurs.

It is often good practice to halt the processor when inserting patches to programs being executed. Failure to observe this caution frequently results in patches not entering as expected.

PROCEDURE EXAMPLES

The following subsections contain typical procedural sequences which are used in operating the HCP.

Display Main Memory Location

The contents of a specific location within main memory can be interrogated and displayed by entering the following:

1. ESCAPE (Panel Mode)
2. ? (Master Clear)
3. ESCAPE (Panel Mode)
4. Load FCR to define P register and main memory

   K71000800:

   Verify FCR update.

5. Load P with desired memory location

   Kxxxx: (xxxx → P)

6. Display memory location

   L:

7. Response to console is memory contents "pointed to" by the P register; the P register is incremented.
8. Consecutive memory locations can be displayed each time the following is entered at the console:

   :
   response
   :
   response, etc.

Write Into Main Memory

Specific locations within main memory can be loaded by entering the following:

1. ESCAPE
2. ?
3. ESCAPE
4. K71000800:, set up FCR
5. Kxxxx:, (xxxx → P)
6. Lyyyy:, (yyyy → loc xxxx)

   The P register has automatically been incremented following the : from the console.

7. The next memory location (xxxx +1) may now be loaded with a new value by Lyyyy:, etc. without having to update P.

Display P Register In Repeat Mode

The P register can be displayed in the repeat mode by entering the following:

1. ESCAPE (Panel Mode)
2. ? (Master Clear)
3. ESCAPE (Panel Mode)
4. K71000400: (Select P register, repeat mode)
5. K: (Display P contents)

   Response is P register contents, repeatedly. This command is usually used with local (maintenance panel) operations rather than console mode when a TTY is used as the console device.

To display the P register locally (as to determine, for example, if the program is running), enter the following:

1. LOCAL/REMOTE switch to local
2. J11: (selects P register)
3. J54: (sets repeat mode, resets console echo)

Response is P register content, repeatedly. To exit this mode, enter:

J58: (set console echo, resets repeat mode)

Operation in Step Mode

The processor can be placed in the step mode, and programs can be executed in single steps (individual instructions) by entering the following:

1. ESCAPE (Panel Mode)
2. ? (Master Clear)
3. ESCAPE (Panel Mode)
4. K71100800: (load FCR)
5. K1000: (1000 → P register)
6. K: (display P contents)
7. Repeat step 6 to execute individual instructions sequentially through the program.

Load and Execute Macroprogram

A program can be loaded into main memory and executed by entering the following:

1. ESCAPE (Panel Mode)
2. ? (Master Clear)
3. ESCAPE (Panel Mode)
4. K71000800: (load FCR; select P register, memory; console echo)
5. K1000: (0B00 → Loc 1000)

6. L0B00: (0B00 → Loc 1000)

7. L0B00: (0B00 → Loc 1001)

8. L0B00: (0B00 → Loc 1002)

9. L0B00: (0B00 → Loc 1003)

10. L18FB (18FB → Loc 1004)

11. K1000: (1000 → P)

12. I:, execute program starting st location 1000

EMERGENCY–OFF PROCEDURE

1. Set the main AC power circuit breakers at the rear of the HCP to the OFF position.

2. Turn off all peripherals.

CHECKS AND ADJUSTMENTS

Operational checks are limited to observing the LED indicators on the circuit cards and the maintenance panel to ensure the presence of all required signals as described in Controls and Indicators.

SHUTDOWN PROCEDURE

NOTE

The HCP is normally powered-up, except for motor-driven peripherals which are turned on only as needed. However, the following procedure is to be used should the operator desire to completely shut down the HCP).

1. Turn off all peripherals and auxiliary power units.

2. Set the PWR switches on all LM circuit cards to the OFF position.

3. Set the circuit breaker on the LM power supply to the OFF position.

4. Set the main AC power circuit breakers at the rear of the HCP to the OFF position

INSTRUCTION DESCRIPTIONS 4

INTRODUCTION

This section contains formats, descriptions and timing information for the two types of instructions used in the HCP: macroinstructions and microinstructions.

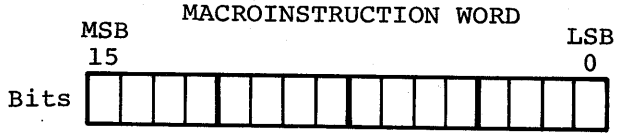
The macroinstruction set is stored in main memory and is compatible with the CONTROL DATA® 1700 instruction repertoire.

Microinstructions are stored in micromemory and are used to control HCP operations.

Refer to appendices A through J for additional information.

MACROINSTRUCTION FORMATS AND DESCRIPTIONS

The general format for macroinstructions consists of 16-bit words numbered right to left as 0 to 15 with the leftmost bit (15) as the most significant bit (MSB).



More specific formats are presented for individual instructions, or groups of instructions, in the following paragraphs.

A listing of macroinstructions, grouped by function, is included as appendix C of this manual; and a numeric listing is provided as appendix D.

Macroinstructions are divided into two groups: basic and enhanced.

BASIC MACROINSTRUCTIONS

Basic macroinstructions consist of the following subgroups:

1. Storage Reference Instructions
2. Register Reference Instructions
   a. Interregister Instructions
   b. Skip Instructions
   c. Shift Instructions

Storage Reference Instructions

The storage reference format shown in figure 4-1 contains 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 where the most significant bit is the sign bit, except where noted.

The following definitions apply to the description of addressing modes:

1. Instruction Address - The address of the instruction being executed; contents of P register.

2. Indirect Address - A storage location that contains an address rather than an operand.

3. 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.

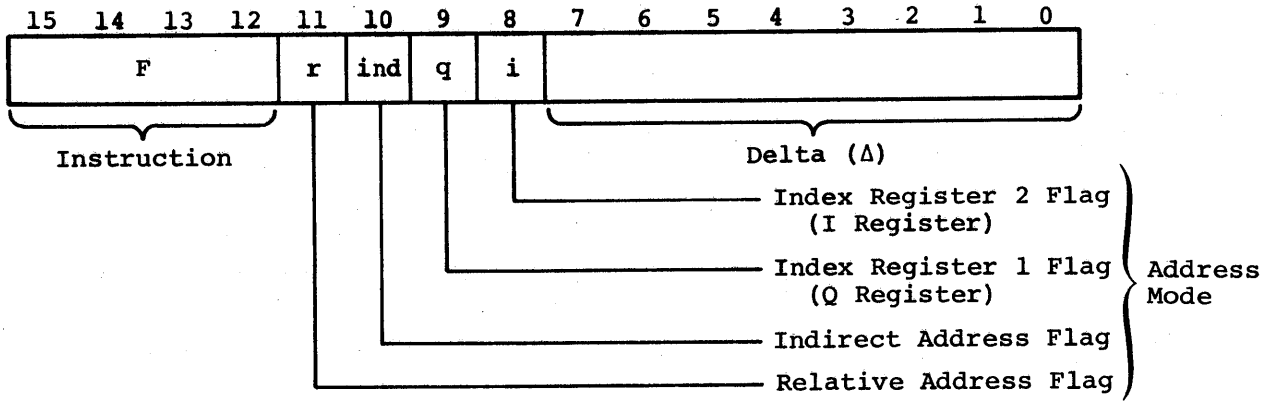


Figure 4-1. Storage Reference Instructions Format

4. Effective Address - The final address of the operand. At certain times the effective address equals the operand for read-operand type instructions.

5. Indexing - The microprocessor (MP) has two index registers. Index register number 1 is the Q register. Index register number 2 is storage location $00FF_{16}$ (the I register). The base address may be modified by either or both of the index registers. If the index register number 1 flag is set, the contents of the Q register are added to the base address to form the effective address. If the index register number 2 flag is set, the contents of storage location $00FF_{16}$ 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. Indexing occurs after completion of indirect addressing.

The MP uses the 16-bit one's complement adder during indexing operations. Consequently, the index register contents are treated as signed quantities (bit 15 = sign bit).

The storage reference instructions have eight different types of addressing modes:

1. Absolute
2. Constant
3. Indirect
4. Storage
5. Relative
6. 16-bit Relative
7. Relative Indirect
8. 16-bit Relative Indirect

1. Absolute (address mode bits = 0, 1, 2, or 3). Both relative and indirect flags equal 0 and delta $\neq 0$. 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.

2. Constant (address mode bits = 0, 1, 2, or 3). Both relative and indirect flags equal 0 and delta = 0. Where the address mode bits = 0, P + 1 is the effective address. Where the address mode bits = 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.

3. Indirect (address mode bits = 4, 5, 6, or 7). Relative address flag equals 0, indirect flag equals 1, and delta ≠ 0. The eight-bit value of delta is an indirect address. Delta is a magnitude quantity for this operation (no sign bit).

4. Storage (address mode bits = 4, 5, 6, or 7). Relative address flag equals 0, indirect flag equals 1, and delta = 0. 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.

5. Relative (address mode bits = 8, 9, A, or B). The relative flag equals 0, and delta ≠ 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. The address referenced by this mode is located forward or backward relative to the P in the program.

6. 16-bit Relative (address mode bits = 8, 9, A, or B). The relative address flag equals 1, the indirect address flag equals 0, and delta equals 0. If no indexing is specified, the instruction address P + 1, plus the contents of location P + 1, form the base address = 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.

7. Relative Indirect (address mode bits = C, D, E, or F). Both relative and indirect flag equal 1. If delta ≠ 0, the value of the instruction address, P, plus the value of delta with sign extending is an indirect address. When in the multilevel indirect mode (FCR bit 18 on), bit 15 of the contents of this indirect address is 0, and the contents of this indirect address is the base address. If bit 15 of the contents of the indirect address is set, the contents of the indirect address is another indirect address. Indirect addressing continues until bit 15 of the contents of the indirect address is 0. The contents of the index registers, when specified, are then added to the base address to form the effective address. When in the single-level indirect mode (FCR bit 18 off), all 16 bits of the indirect address form the base address, regardless of whether or not bit 15 is set.

8. 16-bit Relative Indirect (address mode bits = C, D, E, or F). Both relative and indirect flags equal 1. In single-level indirect mode, the contents of P + 1 + (P + 1)* is the base address. Then the contents of the index register(s), when specified, are added to the base address to form the effective address. In multilevel indirect mode, P + 1 + (P + 1) = base address if bit 15 of (P + 1) is 0; if 1, then P + 1 + (P + 1) form an indirect address and the indirection continues until bit 15 = 0.

Table 4-1 lists all the addressing possibilities for storage reference instructions that may be obtained through combinations of flag bits.

The following paragraphs describe the storage reference instructions.

1. Unconditional Jump JMP

   Effective address specifies the location of the next instruction.

*( ) denotes contents of.

TABLE 4-1. STORAGE REFERENCE INSTRUCTION ADDRESSING

Mode	Binary b_{11}-b_8	Hex	Delta	Effective Address	Next Instruction Address
Absolute Constant	0000	0	≠0 =0	Δ P+1	P+1 P+2
Absolute Constant	0001	1	≠0 =0	Δ+(00FF) (P+1)+(00FF)†	P+1 P+2
Absolute Constant	0010	2	≠0 =0	Δ+(Q) (P+1)+(Q)†	P+1 P+2
Absolute Constant	0011	3	≠0 =0	Δ+(Q)+(00FF) (P+1)+(Q)+(00FF)†	P+1 P+2
Indirect Storage	0100	4	≠0 =0	(Δ) (P+1)	P+1 P+2
Indirect Storage	0101	5	≠0 =0	(Δ)+(00FF) (P+1)+(00FF)	P+1 P+2
Indirect Storage	0110	6	≠0 =0	(Δ)+(Q) (P+1)+(Q)	P+1 P+2
Indirect Storage	0111	7	≠0 =0	(Δ)+(Q)+(00FF) (P+1)+(Q)+(00FF)	P+1 P+2
Relative 16-bit Relative	1000	8	≠0 =0	P+Δ P+1+(P+1)	P+1 P+2
Relative 16-bit Relative	1001	9	≠0 =0	P+Δ+(00FF) P+1+(P+1)+(00FF)	P+1 P+2
Relative 16-bit Relative	1010	A	≠0 =0	P+Δ+(Q) P+1+(P+1)+(Q)	P+1 P+2
Relative 16-bit Relative	1011	B	≠0 =0	P+Δ+(Q)+(00FF) P+1+(P+1)+(Q)+(00FF)	P+1 P+2
Relative Indirect 16-bit Relative Indirect	1100	C	≠0 =0	(P+Δ) (P+1+(P+1))	P+1 P+2
Relative Indirect 16-bit Relative Indirect	1101	D	≠0 =0	(P+Δ)+(00FF) (P+1+(P+1))+(00FF)	P+1 P+2
Relative Indirect 16-bit Relative Indirect	1110	E	≠0 =0	(P+Δ)+(Q) (P+1+(P+1))+(Q)	P+1 P+2
Relative Indirect 16-bit Relative Indirect	1111	F	≠0 =0	(P+Δ)+(Q)+(00FF) (P+1+(P+1))+(Q)+(00FF)	P+1 P+2

†Effective address = operand for read-operand type instructions.
††Multilevel only in multilevel indirect mode.

NOTE: Multilevel indirect mode normally not used in the 2550 product.

2. Multiply Integer (F = 2) — MUI

Multiply the contents of the storage location specified by the contents of the A register. The 32-bit product replaces the contents of Q and A with the most significant bits in the Q register. One's complement arithmetic is used.

3. 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 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.

4. 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.

5. 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 is P + 1 or P + 2 depending on the addressing mode of RTJ. The contents of P are then replaced with the effective address + 1.

6. 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.

7. 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 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.

8. 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. One's complement arithmetic is used. The overflow indicator is set if the magnitude of the sum is greater than the capacity of the A register. Once set, the overflow indicator remains set until a skip on overflow instruction is executed.

9. 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. One's complement arithmetic is used. Operation of overflow is the same as in ADD.

10. 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 contents of the A register.

11. 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.

12. 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.

13. Replace Add One in Storage (F = D) — RAO

    Add one to the contents of the storage location specified by the effective address. The contents of A and Q are not changed. One's complement arithmetic is used. Operation of overflow is the same as in ADD.

14. Load Q (F = E) — LDQ

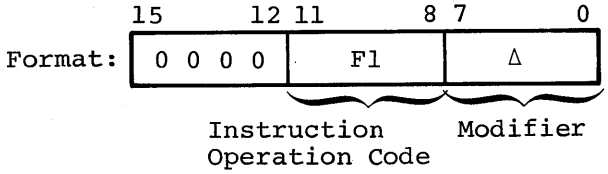
    Load the Q register with the contents of the storage location specified by the effective address. The contents of the storage location are not altered.

15. 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. One's complement arithmetic is used. Operation of overflow is the same as in ADD.

Register Reference Instructions

Register reference instructions use the address mode field for the operation code. Register reference instructions are identified by zeros in the upper four bits of an instruction.



The following paragraphs describe the register reference instructions:

1. Selective Stop (F1 = 0)Δ = 0 — SLS

    Stops the machine if this instruction is executed when the STOP switch is on. The instruction becomes a No Operation (NOP) when the switch is off.

2. 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 an external 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.

3. Output from A (F1 = 3) — OUT

    Deliver 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 an external 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.

4. Increase A (F1 = 9) — INA

    Replaces 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 of overflow is the same as in ADD.

5. Enter A (F1 = A) — ENA

Replaces the contents of the A register with the eight-bit delta, sign extended.

6. No Operation (F1 = B)Δ = 0 — NOP

7. Enter Q (F1 = C) — ENQ

Replaces the contents of Q with the eight-bit delta, sign extended.

8. Increase Q (F1 = D) — INQ

Replaces 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 of overflow is the same as in ADD.

The following instructions are legal only 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).

Switch ON: Selective Stop unless instruction is protected.

Switch OFF: Normal execution (no program protection).

9. Enable Interrupt (F1 = 4)Δ = 0 — EIN

Activates the interrupt system. The interrupt system must be active and the mask bit set for an interrupt to be recognized.

*This is a software switch which is activated by setting a bit in the Function Control Register (see table 3-1).

10. Inhibit Interrupt (F1 = 5)Δ = 0 — IIN

Deactivates interrupt system.

11. Set Program Protect (F1 = 6)Δ = 0 — SPB

Sets the program protect bit in the address specified by Q.

12. Clear Program Protect (F1 = 7)Δ = 0 — CPB

Clears the program protect bit in the address specified by Q.

13. Exit Interrupt State (F1 = E) — EXI

This instruction is used to exit from any interrupt state. Delta defines the interrupt state from which the exit is taken. This instruction reads the address containing the return address, resets the overflow indicator, if necessary, activates the interrupt system, and jumps to the return address.

INTERREGISTER INSTRUCTIONS

(F1 = 8) — Interregister

These instructions cause data from certain combinations of origin registers to be sent through the adder to some combination of destination registers. Various operations, selected by the adder control lines, are performed on the data as it passes through the adder.

If M is the destination register and the instruction is not protected and the program protect switch is set, this is a nonprotected Selective Stop instruction. The program protect fault bit is set and interrupt occurs if selected.

The interregister instructions format is shown in figure 4-2.

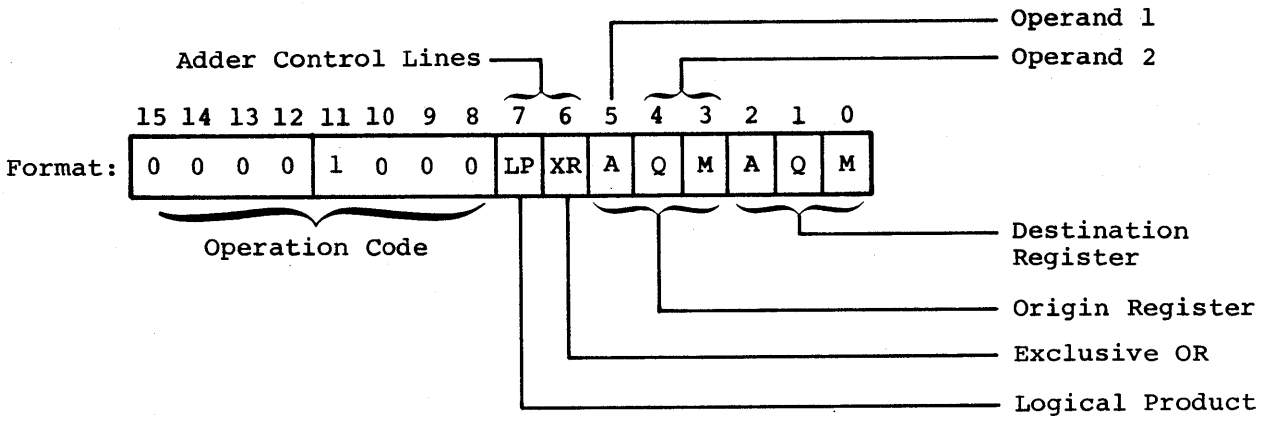


Figure 4-2. Interregister Instructions Format

The contents of origin registers are considered as operands. There are two kinds defined as follows:

a. Operand 1 may be one of the following:

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

   (2) The contents of A (bit 5 = 1)

b. Operand 2 may be one of the following:

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

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

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

   (4) 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 4-2 for examples of all possible 4-bit operands).

a. 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.

b. LP = 1 and XR = 0 - The data placed in the destination register(s) is the logical product-bit by bit, of operand 1 and operand 2.

c. LP = 0 and XR = 1 - The data placed in the destination register(s) is the exclusive OR, bit by bit, or operand 1 and operand 2.

d. LP = 1 and XR = 1 - The data in the destination register(s) is the complement of the logical product, bit by bit, of operand 1 and operand 2.

TABLE 4-2. INTERREGISTER 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 (Exclusive OR)	1 (Logical Product)	0 (Complement Logical Product)	

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. Magnitude comparisons without destroying either operand 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 either operand 1 or operand 2 equal to $FFFF_{16}$.

The Interregister instructions for the MP are as follows:

Mnemonic	Description	Bit 76543
SET	Set to Ones	10000
CLR	Clear to Zero	01000
TRA	Transfer A	00100
TRQ	Transfer Q	00010
TRM	Transfer M	00001
TRB	Transfer Q+M	00011
TCA	Transfer Complement A	01100
TCM	Transfer Complement M	01001
TCQ	Transfer Complement Q	01010
TCB	Transfer Complement Q+M	01011
AAM	Transfer Arithmetic Sum A,M	00101
AAB	Transfer Arithmetic Sum A,Q+M	00111
AAQ	Transfer Arithmetic Sum A,Q	00110
EAM	Transfer Exclusive OR A,M	01101
EAQ	Transfer Exclusive OR A,Q	01110
EAB	Transfer Exclusive OR A, Q+M	01111
LAM	Transfer Logical Product A,M	10101
LAQ	Transfer Logical Product A,Q	10110
LAB	Transfer Logical Product A,Q+M	10111
CAM	Transfer Complement Logical Product A,M	11101
CAQ	Transfer Complement Logical Product A,Q	11110
CAB	Transfer Complement Logical Product A,Q+M	11111

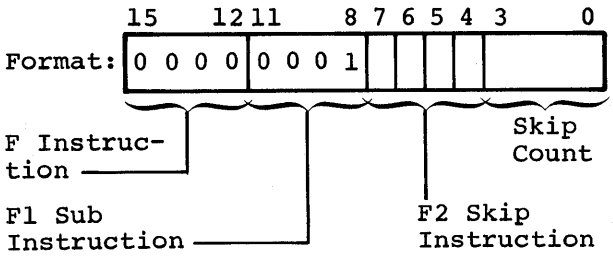
NOTE

The use of + implies an OR.

SKIP INSTRUCTIONS

SKIPS

(F = 1)



Skip instructions are identified when the address mode field is 1 and the instruction mode field is 0. When the skip condition is met, the contents of the skip count +1 are added to P to obtain the address of the next instruction (e.g., when the skip count is zero, 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 for the MP are as follows:

SAZ Skip if A = +0 (F2 = 0) all bits zero

SAN Skip if A ≠ +0 (F2 = 1) not all bits zero

SAP Skip if A = positive (F2 = 2)

SAM Skip if A = negative (F2 = 3)

SQZ Skip if Q = +0 (F2 = 4) all bits zero

SQN Skip if Q ≠ +0 (F2 = 5) not all bits zero

SQP Skip if Q = positive (F2 = 6)

SQM Skip if Q = negative (F2 = 7)

SWS Skip if SELECTIVE SKIP switch* set (F2 = 8)

SWN Skip if SELECTIVE SKIP switch* not set (F2 = 9)

*This is a software switch which is activated by setting a bit in the Function Control Register (see table 3-1).

SOV Skip on overflow (F2 = A)

SNO Skip on no overflow (F2 = B)

SPE Skip on storage parity error (F2 = C)

SNP Skip on no storage parity error (F2 = D)

SPF Skip on Program Protect Fault (F2 = E)

SNF Skip on no program protect Fault (F2 = F)

The skip-on-overflow (SOV) instruction causes a skip if an overflow condition occurs. It clears the overflow indicator.

The skip-on-storage-parity-error (SPE) instruction causes a skip if a storage parity error occurs. It clears the storage parity error signal and indicator.

The program protect fault is set by:

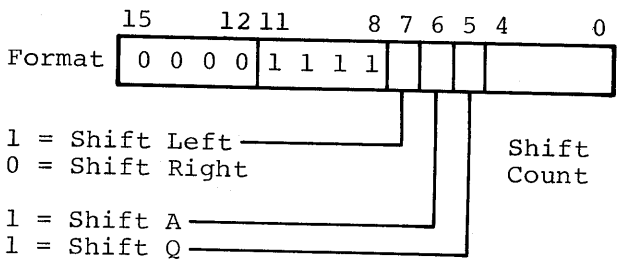
1. A nonprotected instruction attempting to write into a protected address.
2. A protected instruction executed immediately following a nonprotected instruction, except when an interrupt has caused the instruction sequence.
3. Execution of any nonprotected instruction which attempts to alter the interrupt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

SHIFT INSTRUCTIONS

SHIFTS

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.



The following are the shift instructions for the MP:

Shift Instructions

Mnemonic	Instruction Name
ARS	A Right Shift
QRS	Q Right Shift
LRS	Long Right Shift (QA)
ALS	A Left Shift
LLS	Long Left Shift

MEMORY PAGING

Memory paging information required to use the memory paging capability is as follows:

1. Total memory capacity comprises 256K 18-bit words (16 data bits, 1 parity bit, and 1 protect bit).

2. Memory paging file consists of 64 9-bit words, divided into two 32 9-bit word banks. Selection of either bank is by a special microcommand. The bank address is specified by the 5 most significant bits of the 16-bit address presented by the CPU. Seven of the 9 bits in the file word addressed are concatenated with the remaining 11 bits presented by the CPU to form the 18-bit physical address.

3. The 11-bit portion of the address specifies one of 2048 words in each page.

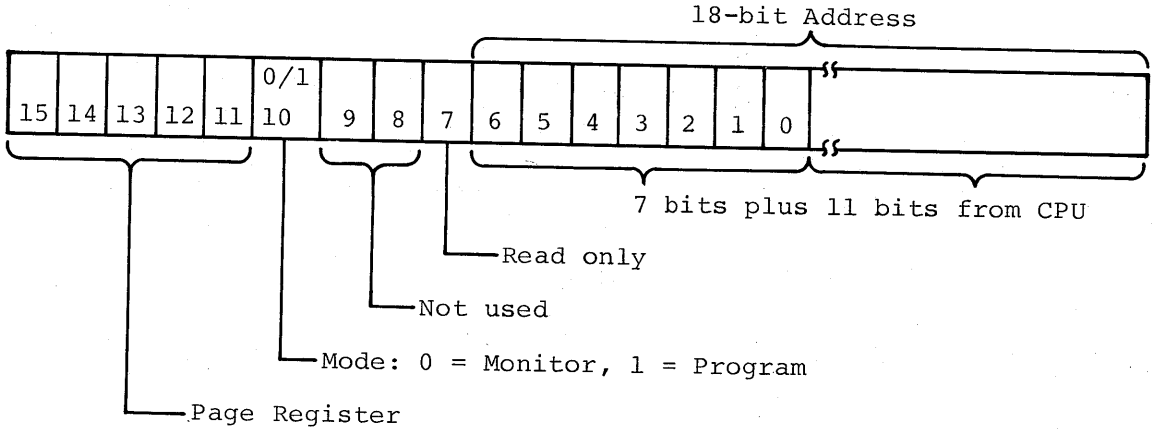
4. The remaining 2 bits in each file word are used as follows:

   a. Nonprotected Page - The protect bit in all 2048 words of the page is ignored and treated as a 0.

   b. Read Only Page - Any write operation to the page is blocked.

MP17 INSTRUCTIONS FOR 256K MOS MEMORY

Format for MP17 instructions used with the 256K MOS memory is as follows:



APM ($0B0B)
Absolute page mode is specified. Only the first 64K (K=1024) of memory can be addressed.

PM0 ($0B0C)
Page mode zero is specified. Segments of 2048 words (totaling 64K) can be addressed via the 32-page mode 0 registers.

PM1 ($0B0D)
Page mode one is specified. Segments of 2048 words (totaling 64K) can be addressed via the 32-page mode 1 registers.

WPR R ($0B23, $0B43,...$0BE3)
Write the contents of register R, bits 0-8, into the page register specified by bits 10-15 of register R. Bit 10 is either zero or one for a page mode 0 or 1 register, respectively. Bits 11-15 specify one of the 32-page registers. Bit 9 is unused and should be zero.

RPR R ($0B24, $0B44,...$0BE4)
Read the contents of the page register, specified by bits 10-15 of register R, into the A register. Bit 10 of register R is either zero or one for a page mode 0 or 1 register, respectively. Bits 11-15 specify one of the 32-page registers. Bits 0-9 are unused and should be zero.

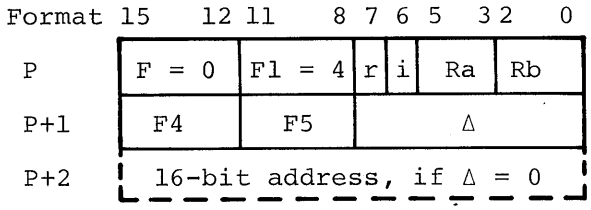
ECC R ($0B25, $0B45,...$0BE5)
Input the ECC status of the memory word, specified by the address in register R, into the A register.

ENHANCED MACROINSTRUCTIONS

Enhanced macroinstructions are divided into the following subgroups:

1. Type 2 Storage Reference Instructions
2. Field Reference Instructions
3. Type 2 Skip Instructions
4. Decrement and Repeat Instructions
5. Type 2 Interregister Instructions
6. Miscellaneous Instructions
   a. Load Micromemory
   b. Load Registers
   c. Store Registers
   d. Set/Sample Output or Input
   e. Sample Position/Status
   f. Define Microinterrupt
   g. Clear Breakpoint Interrupt
   h. Generate Character Parity Even
   i. Generate Character Parity Odd
   j. Scale Accumulator
   k. Load Upper Unprotected Bounds
   l. Load Lower Unprotected Bounds
   m. Execute Microsequence
7. Auto-Data Transfer Instructions

Type 2 Storage Reference Instructions



The type 2 storage reference instruction is identified by a $\Delta \neq 0$ of the enable interrupt instruction (F = 0, F1 = 4). A type 2 instruction is made up of 2 (and sometimes 3) words. The r, i, Ra, and Δ subsections provide addressing modes; Rb specifies an operand.

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

It has the following definition:

F5 = 0 - word processing, register destination

F5 = 1 - word processing, memory destination

F5 = 2 - character processing, register destination

F5 = 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 to determine the addressing:

1. Delta determines 8- or 16-bit addressing. If Δ = 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 type 1 storage instructions. Type 1 allows indexing by one or two registers (I and Q); while type 2 allows indexing by any one of seven registers (1, 2, 3, 4, Q, A, or I). Table 4-3 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 character instructions). Operand B is always the A register for character addressing.

NOTE

For character addressing, a 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.

1. Instruction Address: The address of the instruction being executed, also called P.

2. Indirect Address: A storage address that contains an address rather than an operand. Note that there is no multilevel indirect addressing for type 2 storage reference instructions.

3. 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.

4. Effective Address: The final address of the operand.

5. Indexing: If specified, the contents of register Ra is added to the base address to form the effective address. Indexing occurs after completion of addressing.

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

6. 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

Type 2 storage reference instructions (table 4-3) have the following eight different types of addressing modes:

1. Eight-Bit Absolute - (r is zero, i is zero, and delta is not zero.) The base address equals delta. 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.

2. Eight-Bit Absolute Indirect - (r is zero, i is one, and delta is not zero.) The 8-bit value of delta is an indirect address. The sign bit of delta is not extended. The contents of this address in low core (addresses 0001 to 00FF) is the base address. The contents of index register Ra, when specified, are added to the base address to form the effective address.

3. Eight-Bit Relative - (r is one, i is zero, and delta is not zero.) 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 8-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.

4. Eight-Bit Relative Indirect - (r is one, i is one, and delta is not zero.) The address of the second word of the instruction, P + 1, plus the value of delta with sign extended is an indirect address. The contents of this address is the base address. The contents of index register Ra, when specified, is added to the base address to form the effective address.

5. Absolute Constant - (r is zero, i is zero, and delta is zero.) 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 type 1 storage reference addressing.

6. Sixteen-Bit Absolute - (Storage) (r is zero, i is one, and delta is zero.) 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.

7. Sixteen-Bit Relative - (r is one, i is zero, and delta is zero.) 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.

8. Sixteen-Bit Relative Indirect - (r is one, i is one, and delta is zero.) 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 contents 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 following paragraphs describe the Type 2 storage reference instructions. A detailed listing of these instructions is included as Appendix E of this manual.

TABLE 4-3. TYPE 2 STORAGE ADDRESSING RELATIONSHIPS

Addressing Mode	Delta	r	i	Ra	Effective Address (EA)	Address of Next Instruction
8-bit Absolute	Δ ≠ 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	Δ ≠ 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†	Δ ≠ 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†	Δ ≠ 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
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

TABLE 4-3. TYPE 2 STORAGE ADDRESSING RELATIONSHIPS (Contd)

Addressing Mode	Delta	r	i	Ra	Effective Address (EA)	Address of Next Instruction
16-bit Absolute (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

†For these addressing modes, delta is sign extended.
NOTE: ( ) denotes contents of.

1. Subroutine/Jump Exit (F4 = 5, F5 = 0, Rb = 0) **SJE**

The contents of P are replaced with the effective address. This instruction can be used as a jump or subroutine exit. For example, if delta equals one and Ra has been set up by a previous subroutine jump (see below), control is 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 non-zero) rather than three.

For example, the following program makes a subroutine jump at location 1000. Register A will contain 1002 upon entry to the subroutine SUB. Upon completion, the SUB exits to location 1003.

```
1000    0446          SJA+SUB
1001    5000
1002    2000
1003                    :
2000    ....   SUB    ... ...
                        :
2020    0430          SJE-1,A
2021    5001
```

NOTE

Since Rb = 0, a selection of absolute (r = 0), no indirect (i = 0), and no index register (Ra = 0) results in an EIN instruction.

2. Subroutine Jump — SJr
(F4 = 5, F5 = 0, Rb = 1, 2, 3, 4, 5, 6, or 7)
(r = 1, 2, 3, 4, Q, A, or I)

Load register r with the address of the last word of this instruction (i.e., P + 1 for delta not zero; P + 2 for delta zero). The contents of P are then replaced with the effective address.

3. Add Register — ARr
(F4 = 8, F5 = 0, Rb = 1, 2, 3, 4, 5, 6, or 7)
(r = 1, 2, 3, 4, A, Q, or I)

Add (using one's 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.

4. Subtract Register — SBr
(F4 = 9, F5 = 0, Rb = 1, 2, 3, 4, 5, 6, or 7)
(r = 1, 2, 3, 4, Q, A, or I)

Subtract (using one's 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.

5. AND Register — ANr
(F4 = A, F5 = 0, Rb = 1, 2, 3, 4, 5, 6, or 7)
(r = 1, 2, 3, 4, Q, A, or I)

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.

6. AND Memory — AMr
(F4 = A, F5 = 1, Rb = 1, 2, 3, 4, 5, 6, or 7)
(r = 1, 2, 3, 4, Q, A, or I)

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 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 CPs via memory.

7. Load Register — LRr
(F4 = C, F5 = 0, Rb = 1, 2, 3, 4, 5, 6, or 7)
(r = 1, 2, 3, 4, Q, A, or I)

Load register r with the contents of the storage location specified by the effective address. The contents of storage are not altered.

8. Store Register — SRr
(F4 = C, F5 = 1, Rb = 1, 2, 3, 4, 5, 6, or 7)
(r = 1, 2, 3, 4, Q, A, or I)

Store the contents of register r in the storage location specified by the effective address. The contents of register r are not altered.

9. Load Character to A — LCA
(F4 = C, F5 = 2)

Load bits 0-7 of register A with a character from the storage location specified by the sum of the effective address and bits 1-15 of register Rb. Register Rb bit 0 = 0 specifies the left

character (bits 8-15) of the storage location; bit 0 = 1 specifies the right character (bits 0-7). Bits 8-15 of register A are not altered.

10. Store Character from A (F4 = C, F5 = 3) SCA

    Store the contents of bits 0-7 of register A into a character of the storage location specified by the sum of the effective address and bits 1-15 of register Rb. Bit 0 = 0 of register Rb specifies the left character (bits 8-15) of the storage location; bit 0 = 1 specifies the right character (bits 0-7). The contents of register A and other storage characters are not altered.

11. 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.

12. 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 (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 CPs via memory.

13. 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

    If the contents of register r and the contents of the storage location specified by the effective address are equal, bit by bit, skip one location; otherwise execute next instruction. The contents of register r and storage are not altered.

14. Compare Character Equal (F4 = E, F5 = 2) CCE

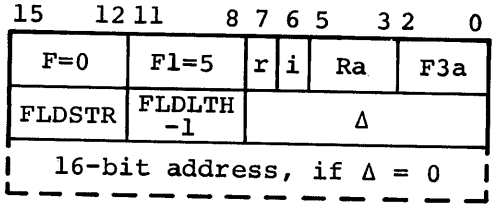
    If the contents of bits 0-7 of register A and the character of the storage location specified by the sum of the effective address and bits 1-15 of register Rb are equal, bit by bit, skip one location; otherwise execute the next instruction. Bit 0 = 0 of register Rb specifies the left character (bits 8-15) of the storage location; bit 0 = 1 specifies the right character (bits 0-7). The contents of register A and storage are not altered.

NOTE

Each compare instruction assumes that a one word instruction follows it.

Field Reference Instructions

Format:



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

Field reference instructions contain four parts: operation field (F3a), addressing mode fields (delta, r, i, and Ra), FLDSTR, and FLDLTH-1 fields. The F3a field determines the operation (e.g., load, store, etc.). The addressing mode fields are defined exactly as the type 2 storage reference instructions.

FLDSTR defines the starting bit of the field: FLDSTR = 0 means the field starts at bit 0; FLDSTR = 15 means the field starts at bit 15. FLDLTH-1 defines the length of the field minus one: FLDLTH-1 = 0 means the field is one bit long; FLDLTH-1 = 15 means the field is 16 bits long. Note that if FLDLTH-1 = 0, the field reference instructions become bit reference instructions.

A field starts at the bit specified by FLDSTR and includes FLDLTH contiguous 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 zero); e.g., 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.

The following paragraphs describe the field reference instructions:

1. Skip if Field Zero (F3a = 2) **SFZ**

   If the contents of the specified field of the storage location identified in the effective address are zero (all bits are zero), skip one location; if the contents are not zero execute the next instruction.

2. Skip if Field Not Zero (F3a = 3) **SFN**

   If the contents of the specified field of the storage location field identified in the effective address are non-zero (not all bits are zero), skip one location; if zero, execute the next instruction.

   NOTE

   Each skip field instruction assumes that a one-word instruction follows it.

3. 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 zero. The contents of storage are not altered.

4. 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.

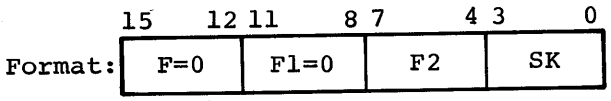
5. Clear Field (F3a = 6) — CLF

   Clear the specified field of the storage location specified by the effective address to all zeros. All other storage bits are unchanged. Memory is locked until completion of the instruction.

6. Set Field (F3a = 7) — SEF

   Set the specified field of the storage location identified in the effective address to all ones. All other storage bits are unchanged. Memory is locked until completion of the instruction.

Type 2 Skip Instructions



Type 2 skip instructions are identified when the F and F1 fields are both zero, and the F2 and SK fields are not both zero. (If these fields are both zero, the instruction is an SLS.)

Type 2 skip instructions (similar to type 1 skips, e.g., 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, non-zero, 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.

Note from above that if F2 = 0 (S4Z), the skip count cannot be zero because the instruction would be an SLS.

The following paragraphs describe the type 2 skip instructions

1. 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 zero (all bits are zero).

2. Skip if Register Non-Zero (F2 = 1, 5, 9, or D) (r = 4, 1, 2, or 3) — SrN SK

   Skip if register r is not a positive zero (not all bits are zero).

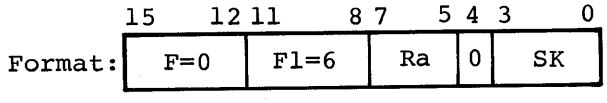
3. Skip if Register Positive (F2 = 2, 6, A, or E) (r = 4, 1, 2, or 3) — SrP SK

   Skip if register r is positive (bit 15 is zero).

4. 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 one).

Decrement and Repeat Instructions



Decrement and repeat instructions are specified when the F field is zero, the F1 field is equal to six, bit 4 is zero, and the Ra and SK fields are

both not zero. (If these fields are both zero, 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.

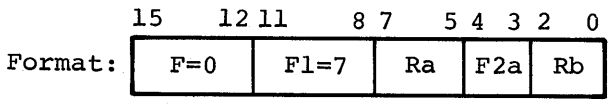
The decrement and repeat instruction is as follows:

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

DrP SK

Decrement by one the contents of register r. 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 zero); if otherwise, execute the next instruction.

Type 2 Interregister Instructions



Type 2 interregister instructions are identified when the F field is zero, the F1 field is equal to seven, and the F2a, Ra, and Rb fields are not all zero. (If these fields are all zero, the instruction is CPB.)

Type 2 inter-register instruction (similar to type 1 interregister, e.g., TRA Q) contains 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 type 2 interregister instruction is as follows:

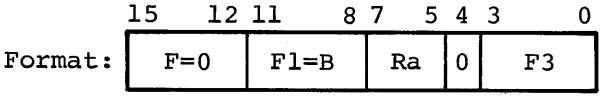
Transfer Register
(F2a = 0, Ra = 1,2,3, r,5,6, or 7)
(r = 1,2,3,4,Q,A, or I)

XFr R

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.

Miscellaneous Instructions



Miscellaneous instructions are identified when the F field is zero, the F1 field is equal to a decimal 11 (hex B) and bit 4 is zero, and the Ra and F3 fields are not both zero. (If these fields are both zero, the instruction is a NOP.) All of the miscellaneous instructions are privileged instructions, i.e., they cannot be executed by an unprotected program and causes a program protect violation instead.

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

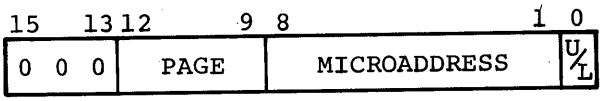
If Ra is non-zero, the F3 operation field can select up to 16 miscellaneous instructions with register Ra used to specify an operand. If Ra is zero, the F3 operation field can select up to 15 more miscellaneous instructions without any explicit operand specified.

The miscellaneous instructions are described in the following paragraphs:

1. Load Micromemory (F3 = 1, Ra = 0) LMM

Load a block of 32-bit micromemory words into read/write micromemory from 16-bit CP main memory.

Initially, the Q register contains the number of 32-bit micromemory words to be transferred (if Q = 0, no words will be transferred). Register 1 contains the starting address of micromemory.



a. Bits 13-15 must be zero.
b. Bits 9-12 specify the micropage.
c. Bits 1-8 specify the micromemory address.
d. Bit 0 specifies the upper (0) or lower (1) microinstruction.

Register 2 contains the starting address of CP main memory.

This instruction is interruptible after storing each 32-bit micromemory word, and when registers 1, 2, and Q are incremented/decremented to allow the restarting of the instruction after any interruption. Therefore, upon completion of the instruction, these registers will not contain their original values, but they will contain the following:

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

where i denotes the initial value before execution.

2. 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 next location, P + 1 as follows:

```
(((P+1))+1) → 1
(((P+1))+2) → 2
(((P+1))+3) → 3
(((P+1))+4) → 4
(((P+1))+5) → Q
(((P+1))+6) → A
(((P+1))+7) → I
(((P+1))+8) → M
bit 15 of (((P+1))+9) → OVERFLOW
```

The contents of the storage location specified by the contents of the next location are decremented by a decimal 10, that is:

```
((P+1))-10 → (P+1)
```

Any data stored in location ((P + 1)) or bits 0-14 of location (((P + 1)) + 9) may be extracted before the execution of the LRG instruction via the address (specified by the contents of the next location).

The contents of the nine storage locations are not altered and the next instruction is executed at location P + 2 (i.e., the LRG instruction is a two-word instruction).

3. Store Registers (F3 = 3, Ra = 0) SRG

The contents of the storage location specified by the contents of the next location, P + 1, is first incremented by a decimal 10, that is:

((P+1))+10 → (P+1)

Then the contents of registers 1, 2, 3, 4, Q, A, I, and M, and the OVERFLOW indicator are stored in nine storage locations, beginning at a storage location specified by the contents of the next location).

(1) → ((P+1))+1
(2) → ((P+1))+2
(3) → ((P+1))+3
(4) → ((P+1))+4
(Q) → ((P+1))+5
(A) → ((P+1))+6
(I) → ((P+1))+7
(M) → ((P+1))+8
(OVERFLOW) bit 15 of ((P+1))+9

After the storing is completed, the OVERFLOW indicator will be cleared.

Note that location ((P + 1)) and bits 0-14 of location ((P + 1)) + 9 can be used to store a program address, priority level, parameter address, or other data after the execution of the SRG instruction via the address (specified by the contents of the next location).

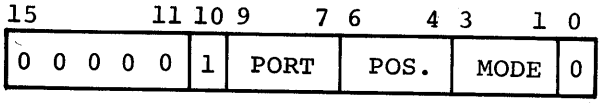
The contents of the registers will not be altered and the next instruction will be executed at location P + 2 (i.e., the SRG instruction is a two-word instruction).

4. Set/Sample Output or Input (F3 = 4, Ra = 0) SIO

For output, one word from register A is set (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.

This instruction permits transmission to/from MO5 peripheral devices. The Q register is defined as follows:



a. Bits 11-15 and bit 0 must be zero.

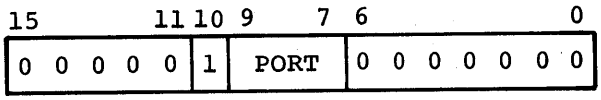
b. Bits 7-10 contain the port number of the device, with bit 10 always a one. Port numbers are analogous to the AQ I/O equipment numbers and thus cannot conflict with them.

c. Bits 4-6 contain the device's position on the port. These bits may also be mode bits (see below) if some or all of the position bits are not required.

d. Bits 1-3 contain the mode in which the device is to operate. Bit 3 is always the set/sample condition bit; if one, one data word will be set (output); if zero, one data word is sampled (input).

5. Sample Position/Status (F3 = 5, Ra = 0) SPS

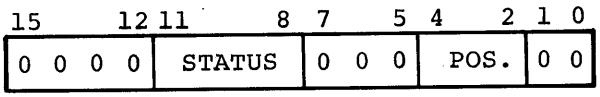
Sample (input) to the A register the position and the status of a MO5 device, which has caused a macrointerrupt. The word in the Q register selects the device. This instruction also provides for the clearing of the MO5-generated CP macrointerrupt. The Q register is defined as follows:



a. Bits 0-6 and 11-15 must be zero.

b. Bits 7-10 contain the port number of the MO5 device, with bit 10 always a one. Port numbers are analogous to the AQ I/O equipment numbers and thus cannot conflict with them.

Upon completion of the instruction, the A register contains the following:



a. Bits 0-1, 5-7, and 12-15 are zero.

b. Bits 2-4 contain the device's position.

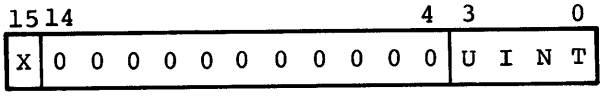
c. Bits 8-11 contain the least significant four bits of the data input lines. If these four bits of status information are insufficient, the device's controller may provide for the transfer of additional status bits via the SIO instruction.

6. Define Microinterrupt (F3 = 6, Ra = 0) DMI

Defines the use of one of the 12 available microinterrupts. (The use of microinterrupts 12 through 15 are restricted for internal use.)

This instruction allows the enabling/disabling of a microinterrupt and defining it for Auto-Data Transfer (ADT) or special usage.

The Q register selects and enables/disables the microinterrupt, which is defined as follows:

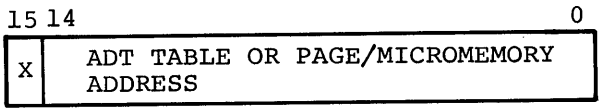


a. Bit 15 enables/disables the microinterrupt; if one, the microinterrupt is enabled; if zero, the microinterrupt is disabled, the A register is not utilized. Initially, all 12 microinterrupts are disabled.

b. Bits 4-14 must be zero.

c. Bits 0-3 contain the microinterrupt number (0-11). Note that 12 through 15 are not available for use and, if used, the instruction will be treated as a NOP.

The A register defines the microinterrupt for ADT or special usage, and is defined as follows:



a. If bit 15 is a zero, ADT is selected and bits 0-14 contain the ADT Table Address. It must be within the lower 32K of main memory.

Four types of ADT Tables are possible for a particular microinterrupt:

(1) Single AQ device
(2) Multi AQ devices
(3) Clock device
(4) Single or Multi MO5 devices

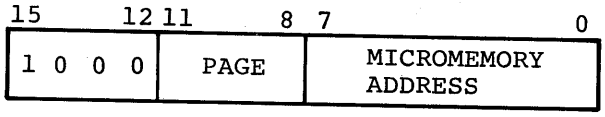
b. If bit 15 is a one, the special usage is selected and a jump is made to the upper microinstruction of the page/micromemory in bits 0-14. A

section of micromemory is assumed to have been previously loaded, and it must process the microinterrupt properly and return control to the current macroinstruction address (P) by jumping to the lower microinstruction of micromemory address $3E_{16}$ in micropage zero. Registers P, A, Q, and all file 2 should not be altered, and return must be within 12.5 microseconds.

NOTE

The CP microfunction SUB- must not be used.

Bits 0-15 of the A register are defined as follows for this special usage:



a. Bit 15 must be one.
b. Bit 12 must be zero.
c. Bits 8-11 specify the micropage.
d. Bits 0-7 specify the micromemory address.

NOTE

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

7. Clear Breakpoint Interrupt (F3 = 7, Ra = 0) CBP

Clears the CP macrobreakpoint interrupt. This interrupt occurs when the following conditions are true.

a. Macrobreakpoint (reference and/or storage) is externally selected.

b. Macrobreakpoint interrupt option is externally selected.

c. The MP recognizes a breakpoint condition and generates a CP macrobreakpoint interrupt because of b.

The macro programmer then has the responsibility of clearing (CBP instruction) and processing the interrupt.

8. Generate Character Parity Even (F3 = 8, Ra = 0) GPE

Set or clear bit 7 of the A register to cause the parity of bits 0-7 to be even. The rest of the contents of the A register are not altered.

9. Generate Character Parity Odd (F3 = 9, Ra = 0) GPO

Set or clear bit 7 of the A register to cause the parity of bits 0-7 to be odd. The rest of the contents of the A register are not altered.

10. Scale Accumulator (F3 = A, Ra = 0) ASC

The A register is shifted left (end-around) until bits 14 and 15 of the A register are different. Upon completion of the instruction, register 1 contains the number of places that the A register was shifted. (This number may range from zero through 14_{10}.) If the A register is $\pm$ zero (0000 or FFFF), no shifting is done and register 1 will contain minus zero (FFFF).

11. 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.

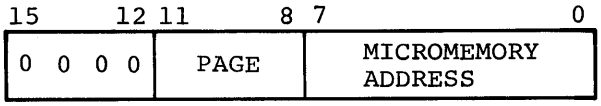
12. Load Lower Unprotected Bounds **LLB R**
(F3 = 1, Ra = 1,2,3,4,5,6, or 7)
(R = 1,2,3,4,Q,A, or I)

Load the lower unprotected bounds from the contents of register R.

13. Execute Microsequence **EMS R**
(F3 = 2, Ra = 1,2,3,4, 5,6, or 7)
(R = 1,2,3,4,Q,A, or I)

Transfer machine control to the upper microinstruction of the page/micromemory address in bits 0-15 of register R. A section of micromemory is assumed to have been previously loaded and it should return control to the next macroinstruction address (P + 1) by jumping to the lower micro-instruction of micromemory addre address $3E_{16}$ in micropage 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). Bits 0-15 of register R are defined as follows:



a. Bits 12-15 must be zero.
b. Bits 8-11 specify the micropage.
c. Bits 0-7 specify the micromemory address.

NOTE

Extreme caution should be exercised in using this instruction, since it provides escape from 1700 emulation.

Auto-Data Transfer Instructions

Auto-Data Transfer (ADT) provides for pseudo-direct memory transfers of data blocks to or from a device. At the macrolevel, the transfer appears as direct memory/storage access (DMA/DSA) transfer; however, at the microlevel, the 1700 emulator processes each data interrupt, and inputs or outputs the next datum 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 ADT for a particular device, perform the following:

1. The device and its controller must adhere to the Auto-Data Transfer specifications.

2. The macroprogrammer 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.

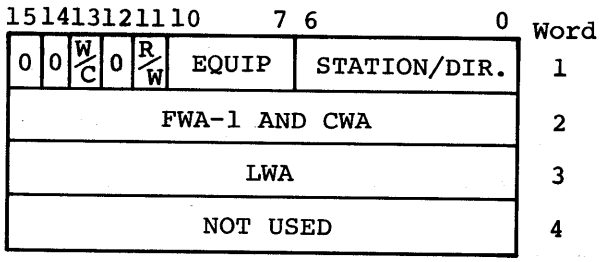
3. The ADT operation is then initiated by an INP, OUT, or SIO instruction as specified by the particular device.

While the ADT operation is in progress, the emulator is executing instructions. After each instruction is executed, interrupts are checked. When the particular ADT microinterrupt becomes the highest active interrupt, the next datum 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 microinterrupt or initiate another ADT operation to or from the device.

The following defines the three types of ADT tables specified by DMI instructions:

1. ADT Table for a Single AQ Device



The ADT table for this type consists of four words:

a. Word 1, bits 12, 14, and 15 must be zero.

b. Word 1, bit 13 equals zero, if a word operation; equals one, if a character operation. For word operation, data is transferred one word at a time. Normally, a total of (CWA - FWA + 1) words are transferred.

   For character operation, data is transferred one character (8 bits) at a time. On input, the first character is stored in the most significant half (bits 8-15) of the current word address; the second character in the least significant half (bits 0-7). Subsequent pairs of characters are input in the same fashion. For output, the first character is output from the most significant half (bits 8-15) of the current word address; the second character from the last significant half (bits 0-7). Likewise, subsequent pairs of characters will be output in the same way. Normally, a total of 2 x (CWA - FWA + 1) characters are transferred.

c. Word 1, bit 11 equals zero, if a read ADT operation; equals one, if a write ADT operation.

d. Word 1, bits 7-10 contain the equipment number of the device.

e. Word 1, bits 0-6 contain the station/director bits of the device to execute the ADT operation. The director bits should specify a data (not a status/function) transfer.

f. Word 2 is initially set to the first word address -1 (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 process and points to the last data word read or stored. Each time a word (or two characters, if character operation) 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 the data has been transferred).

g. Word 3 contains the last word address (LWA) of the data block to be transferred.

h. Word 4 is not currently used and should be zero. It is reserved for future use.

2. ADT Table for Multiple AQ Devices

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Word
0	1	0	0	0	EQUIP				MAX. STATION					1	0	1
T_{15}	T_{14}	T_{13}	T_{12}	T_{11}	T_{10}	T_9	T_8	T_7	T_6	T_5	T_4	T_3	T_2	T_1	T_0	2
T_{31}	T_{30}	T_{29}	T_{28}	T_{27}	T_{26}	T_{25}	T_{24}	T_{23}	T_{22}	T_{21}	T_{20}	T_{19}	T_{18}	T_{17}	T_{16}	3
NOT USED																4
0	0	W/C	0	R/W	EQUIP				STATION/DIR.							5
FWA-1 AND CWA																6
LWA																7
NOT USED																8
•																•
•																•
•																•
0	0	W/C	0	R/W	EQUIP				STATION/DIR.							I*4+1
FWA-1 AND CWA																I*4+2
LWA																I*4+3
NOT USED																I*4+4

The ADT table for this type consists of I*4+4 words, where I is the number of multiple AQ devices, up to 32, on one microinterrupt:

a. Word 1, bit 15, bits 11-13, and bit 0 must be zero.

b. Word 1, bits 14 and 1 must be one.

c. Word 1, bits 7-10 contain the equipment number of the device.

d. Word 1, bits 2-6 contain the maximum station (or channel) number. This is equivalent to the number of multiple AQ devices, minus one, on a microinterrupt. Note that the station numbers must be contiguous (i.e., 0 to I-1).

e. Words 2 and 3 contain termination bits for the 32 devices. Initially, they must be all zero. When a macrointerrupt occurs, one or more of these bits is set to one to indicate that one or more ADT operations have terminated. Thus T_7 = 1 indicates that the seventh device has terminated its ADT operation. Note that after receipt,

the bit should be cleared via an instruction that locks memory (e.g., a CLF instruction).

f. Word 4 is not currently used and should be zero. It is reserved for future use.

g. Words 5, 6, 7, and 8 (and also words I*4+1, I*4+2, I*4+3, and I*4+4, where $2 \leq I \leq 32$) are defined the same as a single AQ device, except that bit 14 of the first word I*4+1 must be a one. (Refer to ADT Table for a Single AQ Device.)

3. ADT Table for the Clock

15	14	13	12	11	10	7	6	0	Word
1	0	0	0	0	EQUIP		STATION/DIR.		1
CLOCK COUNTER									2
CLOCK LIMIT									3
NOT USED									4

The ADT table for this type consists of four words:

a. Word 1, bit 15 must be one.

b. Word 1, bits 11-14 must be zero.

c. Word 1, bits 7-10 contain the equipment of the clock, which is always equal to one.

d. Word 1, bits 0-6 contain the station/director bits of the clock, which is always equal to 70_{16}. (Thus, word 1 should equal $80F0_{16}$.)

e. Word 2 is initially set to zero. Whenever the clock has been enabled, the clock counter is incremented every 3-1/3 milliseconds.

f. Word 3 contains the clock limit, which is interpreted as a multiple of 3-1/3 milliseconds. When the clock counter equals the clock limit and the macroclock interrupt is enabled, the macroclock interrupt occurs. 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 zero, or the limit counter should be incremented by its original value (e.g., five). In the later method, the clock counter can function as an elapsed time counter. Note that if the macroclock interrupt is not answered, the clock limit continues to be incremented.

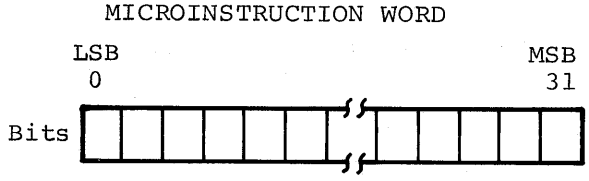
g. Word 4 is not currently used and should be zero. It is reserved for future use.

MACROINSTRUCTION TIMING

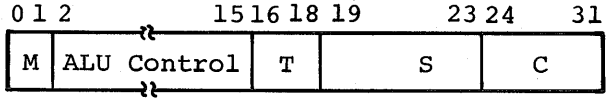
Refer to table 5-4 for information on macroinstruction execution times.

MICROINSTRUCTION FORMATS

The microinstruction format consists of 32-bit instruction words which are numbered left to right as bits 0 to 31. The rightmost bit (31) is the most significant bit (MSB).



Microinstruction formats are divided into five main sections as follows:



Bits	Function
0,1	Mode (M) field specifies format of S and C field and sequencing mode to obtain next microinstruction pair
2-15	ALU control field specifies ALU operation, sources of operands, and destination of result of operation
16-18	Test (T) field specifies method of selecting which microinstruction of next microinstruction pair to execute
19-23	Special (S) field specifies subformat selection and special operation
24-31	Constant (C) or suboperation field specifies constants, or other codes

The M field specifies one of four addressing modes to be used to obtain the next microinstruction pair from micromemory and specifies the format to be used in interpreting bits 19 to 31 of the microinstruction as follows:

M	Addressing Mode	Format for bits 19 to 31
00	Return	Format 1
01	Sequential	Format 1
10	Jump	Format 2
11	Sequential	Format 3

Figure 4-3 shows all microinstruction formats. Note that format 1 and format 2 microinstructions have two subformats, which are selected by the value of bit 19 (SF).

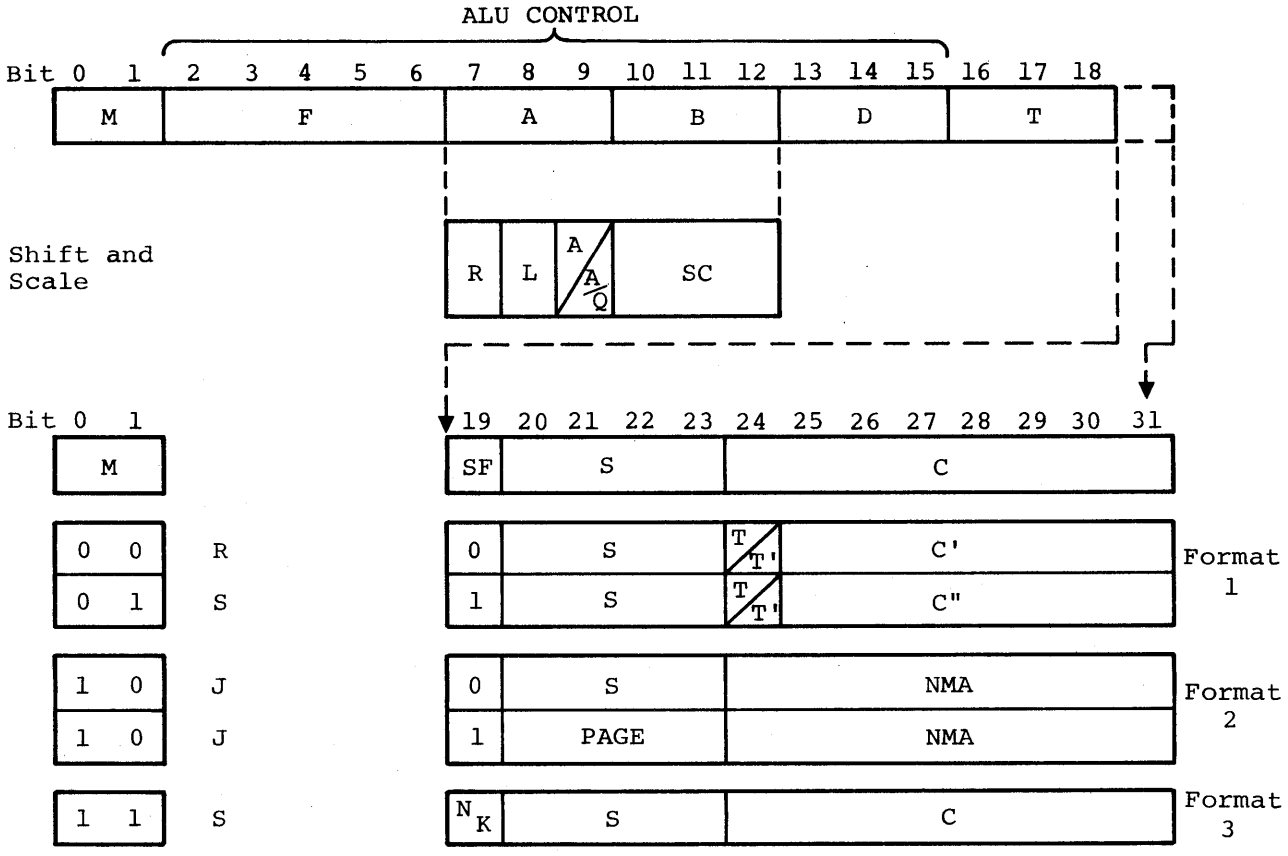
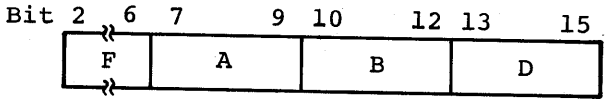
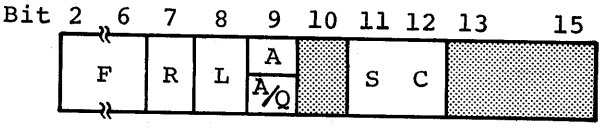


Figure 4-3. Microinstruction Formats

The ALU control fields specify the sources of two operands on which an arithmetic, logical, shift, or scale operation is to be performed, and specify the destination of the result of the operation. For arithmetic and logical operations, the ALU control fields consist of the ALU function (F), source (A), source (B), and destination (D) fields, shown as follows:



For shift and scale operations, these fields are interpreted as follows:

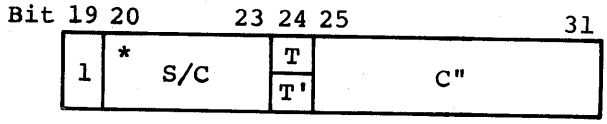


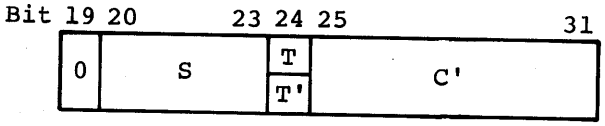
The F field specifies shift or scale operation. Bits 7 and 8 specify right or left shifting. Bit 9 specifies whether the A register alone or the A and Q registers together are to be shifted or scaled. Bit 10 is not used, and bits 11 and 12 specify the shift control code. The D field contains a no-operation code for shift and scale operations.

The T field is the conditional branch of the microinstruction and specifies which microinstruction, upper or lower, of the next microinstruction pair to execute. The test can be based on the result of the ALU operation of the current microinstruction or on some other condition.

The codings in the S and C fields depend upon the contents of the M field. The S and C fields are coded in three formats. Format 1 is specified when the M field contains 00 (return mode) or 01 (sequential mode) as follows:

Format 1

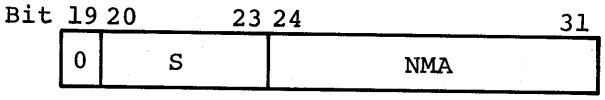


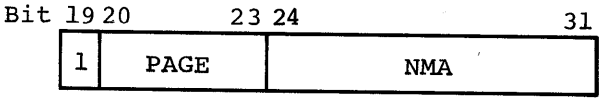


The S field specifies operations such as main memory read or write operations; alternate codings to be used in the A, B, and D fields, etc. The T/T' bit specifies that the code in the T field is to be interpreted as the normal T code (T/T'=0) or as the alternate T' code (T/T'=1). The subformat select bit, bit 19, determines whether bits 25 through 31 are to be interpreted as C' codes or as C" codes. The C' code can contain a constant for driving the bit generator, additional information to control the main memory read or write, or other operations. The "C" codes are associated with transforms.

Format 2 is specified when the M field contains 10 (jump mode) as follows:

Format 2





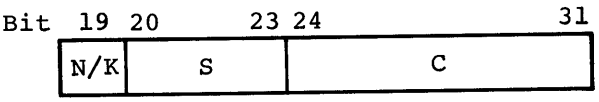
If a jump is specified to a microinstruction pair within the same microinstruction page, the subformat select bit is 0; bits 20 to 23

*If SM207 is set to a 1, the normal S field commands are disabled with C" codes TMA/j, TMAK/j, GITMAK/j, GITMAK/xt, and TK/j.

contain a special operation code as in format 1. Bits 24 through 31 contain the micromemory address of the next microinstruction pair. The subformat select bit is 1 when a jump is specified to a different micromemory page; bits 20 through 31 contain the complete micromemory address of the next microinstruction pair.

Format 3 is specified when the M field contains 11 (sequential mode), as follows:

Format 3



This format allows one special operation to be performed as specified by the S code and also causes the eight bits of the C field to be transferred to the N register (bit 19 =1) or to the K register (bit 19 =0).

MICROINSTRUCTION DESCRIPTIONS

The following paragraphs describe the microinstructions used in the MP.

M FIELD OPERATIONS

The M field (bits 0 and 1) defines the major operation taking place in the microinstruction and also specifies the type of sequencing to be used to obtain the next instruction pair. The operations specified in the M field are listed in table 4-4.

F FIELD OPERATIONS

The F field (bits 2 through 6) specifies the logical or arithmetic operation to be performed by the ALU, or the shift or scale operations performed with the A and Q registers. The split adder option is described in the following paragraph.

Split Adder Option

The split adder option allows the main ALU to be split into two independent adders. This split is activated by setting the adder-split flag in the SM register. The split blocks the carry between the two portions of the adder. The upper portion of the adder always functions as a two's complement adder; the lower portion can function as a one's complement or as a two's complement adder, depending upon the state of the one's complement SM register flag (32-bit processor only only). In one's complement mode the carryout of the lower portion is used as the end-around carry bit. In two's complement mode, both portions of the adder act as independent two's complement adders. The split adder has no effect on logical operations because no carry is involved in these operations.

Logical Operations

The logical operations perform bit-by-bit combinations of the A input and the B input for delivery to the destination. The logical operations are described in table 4-5.

Arithmetic Operations

The arithmetic operations (table 4-6) operate on single-precision operands (using the main ALU).

Overflow Capture Option

This option allows capture of the overflow condition in the SM register (indicated by a T in the instruction mnemonic). If this is indicated, the overflow test is performed by checking the sign of the two inputs to the ALU and by setting a status mode bit if the result if inconsistent. The SM overflow bit is set to 1 when the overflow occurs; it must be set to 0 by a microinstruction which sets that SM bit to 0. Two other

TABLE 4-4. M FIELD OPERATIONS

M Code	Mnemonic	Operation
00	R	Select next microinstruction pair from address contained in RTJ register. Use format 1 for special operations.
01	S	Select next microinstruction pair in current page from address contained in MAC (normally next sequential pair, unless suppressed by T field coding). Use format 1 for special operations.
10	J	A jump or page jump. Select next microinstruction pair from address specified by bits 24 to 31 of this microinstruction. Address is in current MM page if bit 19 is 0, or from page specified in bits 20 through 23 if bit 19 is 1. Use format 2 for special operations.
11	S	Transfer bits 24 through 31 of this microinstruction to N or K register as specified by bit 19 of this microinstruction. N register is specified if bit 19 is 1, and K register is specified if bit 19 is 0. Select next microinstruction pair in current page from address contained in MAC (normally next sequential microinstruction pair). Use format 3 for special operations.

TABLE 4-5. LOGICAL OPERATIONS

F Code	Mnemonic	A input = 0 0 1 1 B input = 0 1 0 1
		Bit Result
01100	ZERO	0 0 0 0
01110	A·B	0 0 0 1
01101	A·-B	0 0 1 0
01111	A	0 0 1 1
01000	-A·B	0 1 0 0
01010	B	0 1 0 1
01001	EOR	0 1 1 0
01011	A+B	0 1 1 1
00100	-A·-B	1 0 0 0
00110	-EOR	1 0 0 1
00101	-B	1 0 1 0
00111	A+-B	1 0 1 1
00000	-A	1 1 0 0
00010	-A+B	1 1 0 1
00001	$-\overline{A}$+-B	1 1 1 0
00011	ONE	1 1 1 1

TABLE 4-6. ARITHMETIC OPERATIONS

F Code	Applicable Notes	Mnemonic	Operation
10110		SUB	Subtract B input from A input.
11000		ADD	Add A and B inputs.
10101		SUBT	Subtract with an overflow test.
11001		ADDT	Add with an overflow test.
10110	1,3	SUB-	Perform A-B-1. See applicable notes.
	2,3	SUB-C	Perform A-B with force carry in. See applic. notes.
11010	3	ADD+	Perform A + B with force carry in. See applicable notes.
10111	1,3	SUB-T	Perform SUB- with an overflow test. See applicable notes.
	2,3	SUB-TC	Perform SUB-C with an overflow test. See applicable notes.
11011	3	ADD+T	Perform ADD+ with an overflow test. See applicable notes.

NOTES:

1. When this F code is specified and when the system is in two's complement mode, this is the resultant arithmetic operation.
2. When this F code is specified and when the system is in one's complement mode, this is the resultant arithmetic operation.
3. If split adder mode is selected, this operation is performed on the upper adder without a force carry.

optional SM bits allow checking for binary or decimal arithmetic overflow.

Shift Operations

The shift operations in the F field specify a shift of the A register or the A/Q register of the main MP organization. The ALU is not used to perform the shift, but performs some operation based on its decoding of the F field (which should be considered as unknown). The destination receives this meaningless output unless a NOP is chosen for the destination of the D field.

The type of shift is determined by the coding in bits 7 to 12 of the microinstruction, and the amount of the shift is determined by the number contained in the N register. The operation examines the register, and, if it is zero, the shift operation is terminated and the next microinstruction is executed. The T field codes that can be used with a shift are U, L, KZU, INTU, OVFL, K7L, and BTU. Other T codes cannot be used.

If the N register is not zero, a shift of one bit position is taken as specified, N is decremented by one, and the test for zero is repeated as above.

The shift conditions are:

1. Shift A - Shift the A register only.

2. Shift AQ - Shift the combined A and Q register. The Q register is considered to be the least significant bits of the combined A/Q register.

3. Shift external (transform)

4. Shift left or right.

5. Enter 0 - Enter a 0 in the vacated bit position at the end of the register.

6. Enter 1 - Enter a 1 in the vacated bit position at the end of the register.

7. Extend sign - Extend the sign (for a right shift only).

8. End-around carry - Enter the bit coming off the end of the register into the vacated position at the other end.

All shifts are performed with an F code of 11110. The type of shift is determined by bits 7 through 12 of the microinstruction. The shifts are defined in table 4-7.

Scale Operations

The scale operations are similar to the shift operations but the stopping of the shift is conditioned on bits 0 and 1 of A not being equal. (The scale point is normally between bits 0 and 1 of the A register. A design option allows the scale point to be specified between different bits in the A register if necessary for efficient floating-point emulation.) The maximum number of bits to scale is contained in the N register, and, on completion of the scale, N is decremented by the number of shifts which were necessary to scale the number.

The scale operation is performed as follows:

1. Examine N; if it is zero, exit the microinstruction.

2. Examine bits 0 and 1 of the A register; if they differ, exit the microinstruction.

3. Shift the A or A/Q register left by one bit position as specified in the instruction.

4. Decrement the N register by one count and go to step 2.

NOTE

If the number that is being scaled is all zeroes or all ones (i.e., the number cannot be scaled), then when N=$FE (after-passing through N=0) the scale operation is terminated. In order to avoid exiting the micro-instruction before the scale operation is actually completed, (N) should be at least equal to the number of bits in the word to be scaled.

The scale operation is coded the same in bits 7 through 12 of the micro-instruction and allows the same left shift options as the shift command. The same comments on exiting the shift and the usable T field codes apply to the scale operation.

All scale operations are performed with an F code of 11111. The type of shift for the scale is determined by bits 7 through 12 of the instruction. The scales are given in table 4-8.

A INPUT OPERATIONS

The A field (bits 7 through 9) specifies the input to S1 and thus to the A side of the ALU. The eight A codes are expanded by eight A' codes by placing 1010 or 0111 in the S field.

TABLE 4-7. SHIFT OPERATIONS

Bit Code		Mnemonic	Operation
7 8 9	11 12		
1 0 0	0 0	AROE	A is right-shifted (N) bits, with 0 entered at most significant bit.
1 0 0	0 1	ARSE	A is right-shifted (N) bits, with sign extension.
1 0 0	1 0	AREA	A is right-shifted (N) bits, with end-around carry.
0 1 0	0 0	ALOE	A is left-shifted (N) bits, with 0 entered as least significant bit.
0 1 0	0 1	ALlE	A is left-shifted (N) bits, with 1 entered as least significant bit.
0 1 0	1 0	ALEA	A is left-shifted (N) bits, with end-around carry.
1 0 1	0 0	AQROE	A/Q is right-shifted (N) bits, with 0 entered as most significant bit in A.
1 0 1	0 1	AQRSE	A/Q is right-shifted (N) bits, with sign extension.
1 0 1	1 0	AQREA	A/Q is right-shifted (N) bits, with end-around carry.
0 1 1	0 0	AQLOE	A/Q is left-shifted (N) bits, with 0 entered at least significant bit in Q.
0 1 1	1 0	AQLEA	A/Q is left-shifted (N) bits, with end-around carry.

TABLE 4-8. SCALE OPERATIONS

Bit Code		Mnemonic	Operation
7 8 9	11 12		
0 1 0	0 0	SLOE	A is scaled left, with 0 entered as least significant bit.
0 1 0	0 1	SL1E	A is scaled left, with 1 entered as least significant bit.
0 1 0	1 0	SLEA	A is scaled left, with end-around carry.
0 1 1	0 0	SDLOE	A/Q is scaled left, with 0 entered as least significant bit in Q.
0 1 1	1 0	SDLEA	A/Q is scaled left, with end-around carry.

The eight A inputs to S1 and to the A side of the ALU are indicated when the S field is not 0111 or 1010; the eight A codes specify inputs from the files, the registers, or main memory, according to table 4-9.

A' INPUT OPERATIONS

The eight A' inputs to S1 and to the A side of the ALU are indicated if the S field is 0111 or 1010. The A' codes specify input from the SM registers or mask registers. The A' codes are given in table 4-10.

B CODES

The B field (bits 10 through 12) specifies the input to S2 and thus to the B side of the ALU. The 11 possible B codes are expanded by the seven B' codes when the S field contains 1000. The N and K codes are controlled by bits 28 and 29 from the C field.

The eleven B inputs to S2 and thus to the B side of the ALU are indicated if the S field is not 1000. Code 001 of the B field is expanded by the use of bits 28 and 29 of the microinstruction for enabling the N or K register to S2. This use of bits 28 and 29 is independent of the other use of the C field, and thus, by judicious use of commands in the C field, the input for the N and K may be used in conjunction with commands or constants in the C field. The codes for B inputs are given in table 4-11.

B' CODES

The B' inputs are specified in the B field when the S field contains 1000. The codes and actions are given in table 4-12.

D CODE TRANSFERS

The D field (bits 13 through 15) specifies the destination of information from the main organization of the MP. There are four sources of this information as follows:

1. An optionally shifted ALU output. This shifting occurs in the S3 shift network that connects the ALU output to the P, A, F, X, or Q registers
2. The output of the ALU
3. The A source
4. The B source

All D destinations except the I register are optionally shiftable by S3 when specified by a code in the C field or by the L8EA command in the S field, if the alternate codings, D' or DD", are not specified. The I destination differs from the others in that the output of the A source is the input to the I register. The codes and their operations are given in table 4-13.

D' CODE TRANSFERS

The D' destinations are specified by the D field if the S field is set to 1001 or 1010. The codes and action are given in table 4-14.

D" CODE TRANSFERS

The D" destinations are specified by the D field if the S field is set to 1011. These destinations transfer data to the double-precision logic from S1. The codes and actions are given in table 4-15.

DD" CODES

The DD" option allows the performance of an operation on A, X, or F; this changes the register, but keeps a copy of the original register in a double-precision register. This is another way of getting data to the double-precision registers. The DD" option is specified when the S field contains 0001. Table 4-16 lists the DD" codes and their operations.

TABLE 4-9. A INPUT OPERATIONS

A Code	Mnemonic	Operation
000†	F2	Use contents of file 2 register as A source input. Current value of N register is used to address register file 2. If value of N is changed in current microinstruction, its initial value is used to reference file register.
001	P	Use contents of P register as A source.
010	I	Use contents of I register as A source.
011	X	Use contents of X register as A source.
100	A	Use contents of A register as A source.
101	F	Use contents of F register as A source.
110†	F1	Use contents of file 1 register (when it is furnished) or external source as A source. Current value of K register is used to address register file 1. If value of K is changed in current microinstruction, initial value of K is used to reference file register. Note that file 1 is available only as an option.
111	MEM	Obtain data read from main memory and use it as A source. If main memory READ command was not given in the preceding microinstruction, all ones are input to A source if B' source is not selected. Restriction: This command is restricted to a microinstruction with type A, B, or C execution time.

†RESTRICTION: Value of addressing register (N or K) cannot have been modified by a C' increment or decrement command in preceding microinstruction. In addition, whichever file is read during the current microinstruction, this same file cannot have been written in the preceding microinstruction.

TABLE 4-10. A' INPUT OPERATIONS

A' Code	Mnemonic	Operation
000	SM1	Use contents of SM register 1 as A source.
001	M1	Use contents of interrupt mask register 1 as A source.
010	SM2	Use contents of SM register 2 as A source.
011	M2	Use contents of interrupt mask register 2 as A source input.
100	A*R8	Use contents of double-precision A* register, shifted right eight bits with end-around carry, as A source. A* register remains unshifted.
101	A*	Use contents of double-precision A* register as A source.
110	X*	Use contents of double-precision X* register as A source.
111	Q*	Use contents of double-precision Q* register as A source.

NOTE: Within the 2550 product, the Cyclic Encoder replaces the double precision logic. However, for uniformity of definition, these starred references remain unchanged. For further information, refer to the descriptive material on the Cyclic Encoder.

TABLE 4-11. B CODES

B Code	28 29	Mnemonic	Operation
000 ††		F2	Use contents of file 2 register as B source. Value of N register, before instruction is executed, is used to address register file 2. If value of N is changed in current microinstruction its initial value is used to reference file register.
001	1 1	Zero	B source is all zeros.
001 †	1 0	N	Use contents of N register as B source. Since N is an 8-bit register, this source uses N as upper eight bits and zeros as lower bits.
001 †	0 1	K	Use contents of K register as B source. Since K is an 8-bit register, upper bits are zeros and K serves as lower eight bits.
001 †	0 0	N, K	Use contents of N and K registers as B source. These registers are combined, with N register as upper eight bits of source and K as lower eight bits.
010		BG	Use contents of BG register as B source. This register has only one bit set to 1, and position of bit in BG register is specified either on value in the N register or by number in C field, depending on state of controlling SM register bit.
011		X	Use contents of X register as B source.
100		Q	Use contents of Q register as B source.
101		F	Use contents of F register as B source.
110 ††		F1	Similar to F2, but uses the contents of file 1 register (when it is furnished) addressed by (K) as B source. Note that file 1 is available only as an option.
111		MEM	Obtains data read from main memory and uses it as B source. If main memory READ command was not given in the preceding microinstruction all one's are input to B source if A' source not selected. Restriction: This command (MEM) is restricted to a microinstruction with type A, B, or C execution time.

† In a 32-bit processor, the 16 most significant bits are always zeros. It is the 16 least significant bits of the 32-bit B source that are controlled with these codes.

†† RESTRICTION: Value of addressing register (N or K) cannot have been modified by a C' increment or decrement command in preceding microinstruction. In addition, whichever file is read during the current microinstruction, this same file can not have been written in the preceding microinstruction.

TABLE 4-12. B' CODES

B' Code	Mnemonic	Operation
000	OPEN	
001	CRTJ	Transfer the complement of the RTJ register to the twelve LSB's of S2. Transfer 1's to the four most significant bits of S2.
010	INRD	Input data/status from I/O channel.
011	INRS	Input to S2 I/O response signals.
100	MMU	Transfer upper 16 bits of data from micromemory to X register in 16-bit MP; transfer total 32 bits in 32-bit MP. F field must make a reference to B source. Address is specified by transform or NK. D field must be an NOP.
101†	MML	Same as above, but takes lower 16 bits of data in a 16-bit MP.
110 or 111	INTA	Use contents of interrupt address encoder as B source. The output of this encoder represents the complement of the interrupt address of the highest priority interrupt line active having its corresponding mask bit set. Restrictions: An INTU test command must be given in the preceding microinstruction.

† This code is non-operative in a 32-bit processor.

TABLE 4-13. D CODE TRANSFERS

D Code	Mnemonic	Operation
000	NOP	Do not transfer data to any destination.
001	P	†Transfer output of S3 to P, AB†††
010	I	Transfer output of S1 to I, AB†††
011	Q	Transfer output of S3 to Q, AB†††
100	F1	††Transfer output of S3 to F register, AB, and write this data in file 1 (when it is furnished) at address specified by K at completion of this instruction.***

TABLE 4-13. D CODE TRANSFERS (Contd)

D Code	Mnemonic	Operation
101	A	Transfer output of S3 to A, AB
110	X	Transfer output of S3 to X, AB
111	F	Transfer output of S3 to F, AB

†If a D field command to load the AB is issued in the next microinstruction following the microinstruction with this command, the transfer AB is inhibited.

††The writing of data into the file 1 register takes place during the first part of the next microinstruction and takes advantage of the updated value of K from this microinstruction. Also, the next microinstruction must not specify a read of file 1.

†††AB is the main memory address buffer register.

TABLE 4-14. D' CODE TRANSFERS

D' Code	Mnemonic	Operation
000	IOD	Transfer output of S3 to I/O data register.
001	IOA	Transfer output of S3 via the I/O data register to I/O address register. Destroys contents of I/O data register.
010	MMU	Transfer output of S2 to upper 16 bits of micro-memory in 16-bit MP, or transfer output of S2 to 32-bit word in micromemory in 32-bit MP.
011	MML	Transfer output of S2 to lower 16 bits of micro-memory location in 16 bit MP, or transfer output of S2 to 32-bit word in micromemory in 32-bit MP.
100	M1	Transfer output of ALU to mask register 1.
101	SM1	Transfer output of ALU to SM register 1.
110	M2	Transfer output of ALU to mask register 2.
111	SM2	Transfer output of ALU to SM register 2.

NOTE: Outputs to the mask and SM registers are direct from the ALU and are not shiftable.

TABLE 4-15. D" CODE TRANSFERS

D" Code	Mnemonic	Operation
000	NOP	Do not transfer data to any destination.
001	A*LHW	Transfer output of S1 to A* register, shifted left one-half word, with end-around carry.
010	X*LHW	Transfer output of S1 to X* register, shifted left one-half word, with end-around carry.
011	Q*LHW	Transfer output of S1 to Q* register, shifted left one-half word, with end-around carry.
100	NOP	Do not transfer data to any destination.
101	A*	Transfer output of S1 to A* register.
110	X*	Transfer output of S1 to X* register.
111	Q*	Transfer output of S1 to Q* register.

TABLE 4-16. DD" CODES

DD" Code	Mnemonic	Operation
101	AA*	Transfer output of S3 to A register, and transfer output of S1 to A* register.
110	XX*	Transfer output of S3 to X register, and transfer output of S1 to X* register.
111	FQ*	Transfer output of S3 to F register, and transfer output of S1 to Q* register.

T AND T' ADDRESSING MODES

T field (bits 16 through 18) - The purpose of the T field is to select the upper or lower microinstruction of the next microinstruction pair to execute. The selection of the next microinstruction may be a conditional selection or an unconditional selection, depending on the T field code. This field is available to all addressing modes in addition to I/O operations. A conditional selection may test the ALU output, the value of certain registers, certain internal conditions such as interrupts, and particular bits wired to the transform board. The only exception to these uses of the T field is when micromemory data is being read or written; in these cases, the T field is used as part of the micromemory data reference address and the upper instruction in the next sequential microinstruction pair is always selected.

The T field codes consist of two groups, T codes, and T' codes. Similarly to the A, B, and D fields that are extended by using the S field, the T field is always extended in the following sense. Bit 24 of format 1 microinstructions is either 0 or 1 if T or T', respectively, is specified. The T' codes are available only for the return and sequential addressing modes (format 1). The T/T' codes select the upper or lower portion of the next microinstruction pair as the next microinstruction to execute. The T codes are listed in table 4-17; the T' codes are listed in table 4-18.

SUBFORMAT SELECT BIT

The subformat select bit (SF), bit 19, is used to select either variations in format 1 and format 2 decoding or the choice of addressing the N or K register in format 3.

S FIELD CODES

The S field (bits 20 through 23) of the microinstruction is used to specify a special command (including alternate codings in the A, B, and D fields), in addition to page or constant information (as required by the code in the C field). The S codes specify actions which take place at the same time as the ALU operation specified in the F, A, B, and D fields. The codes and operations are given in table 4-19.

C CODE OPERATIONS

The C field (bits 24 through 31) is used to specify an additional special operation, an address for a jump, or a constant for setting the K or N register. Bit 24 in format 1 specifies the T field interpretation. The codes for the C field are listed in table 4-20.

TABLE 4-17. T ADDRESSING MODES

T Code	Mnemonic	Operation
000†	L	Execute lower microinstruction of this microinstruction pair as next microinstruction. This operation overrides M field addressing mode.
001	U	Execute upper microinstruction of next microinstruction pair.
010	L	Execute lower microinstruction of next microinstruction pair.
011†	KZU	If initial contents of K register are zero, execute upper microinstruction of next microinstruction pair; otherwise, execute lower microinstruction of next microinstruction pair. If decrement K command is included in same microinstruction, K will contain all ones on satisfying zero test.
100†	NZU	If initial contents of N register are zero, execute upper microinstruction of next microinstruction pair; otherwise, execute lower microinstruction of the next microinstruction pair. If decrement N command is included in same microinstruction, N will contain all ones on satisfying zero test.
101	INTU	If there is an interrupt and its corresponding interrupt mask bit is set, execute upper microinstruction of next microinstruction pair; otherwise, execute lower microinstruction of next microinstruction pair.
110	NU	If sign bit of ALU output is negative on completion of this microinstruction, execute upper microinstruction of next microinstruction pair; otherwise, execute lower microinstruction of next microinstruction pair.
111	ZL	If output of ALU is zero on completion of this instruction, execute lower microinstruction of next microinstruction pair; otherwise, execute upper microinstruction of next microinstruction pair.

†NOTE: Restriction - These T-field commands cannot be used in microinstruction with a C-field INCK or INCN commands respectively.

TABLE 4-18. T' ADDRESSING MODES

T' Code	Mnemonic	Operation
000	*L	Execute lower microinstruction of this microinstruction pair. This operation overrides M field addressing mode.
001	LQL	If, at start of this microinstruction, least significant bit of Q is 1, execute lower microinstruction of next microinstruction pair. Otherwise, execute upper microinstruction of next microinstruction pair.
010	K7L	If the LSB of K register is set, execute lower of next microinstruction pair. If clear, execute upper microinstruction of next microinstruction pair.
011	OVFL	If overflow exists, execute lower microinstruction of next microinstruction pair; if not, execute upper microinstruction of next microinstruction pair.
100	BTU	Bit test. Lower order five bits in C field of this microinstruction specify a setting of bit test selector. If bit at this position is 1, execute upper microinstruction of next microinstruction pair; otherwise, execute lower microinstruction of next microinstruction pair. Bit test is general-purpose testing facility that allows wiring any bit of organization available on machine's backpanel to bit test selector. This wiring is defined on transform board.
101	LQ*L	If, at start of this microinstruction, least significant bit of Q* is 1, execute lower microinstruction of next microinstruction pair; otherwise, execute upper microinstruction of next microinstruction pair.
110	COL	Carry-out lower. If as result of arithmetic operation, there is carry-out of ALU, execute lower microinstruction of next microinstruction pair. Otherwise, execute upper microinstruction of next microinstruction pair. This instruction allows a test for carry-out of ALU during multiple-precision arithmetic.
111	Z*L	Same as ZL (Table 4-17) except ALU* tested.

TABLE 4-19. S FIELD CODES

S Code	Mnemonic	Operation
0000	NOP	No operation for S field.
0001	DD	Alternate D field coding, DD".
0010	RPT	If N register is not equal to zero, selection of next microinstruction **pair is inhibited and current microinstruction pair is next microinstruction** pair. N register is decremented by one. Normal T field selection applies. If N register is equal to zero, normal next microinstruction pair is used.
0011	READ	Read main memory command. Read word from main memory at address contained in the address buffer (AB). Instruction execution is delayed until a resume is received from memory acknowledging command. Restrictions: AB must be loaded in a previous micro-instruction (D field command). If D field command reloading AB is issued in same microinstruction as READ command, the new data will not be loaded into AB until completion of read portion of cycle. A READ command can only be given in a microinstruction with type A, B, C, or D execution time. Memory data must be input to system in following micro-instruction with type A, B, or C execution time only.
0100	WRITE	Write main memory. Transmit output of S3 to main memory as data to be written at address contained in address buffer (AB). Instruction execution is delayed until a resume is received from memory acknowledging command. Data is stored in memory at completion of this instruction. Restrictions: AB must be loaded in a previous microinstruction (D field command). If D field command reloading AB is issued in same micro-instruction as WRITE command, the new data will not be loaded into AB until completion of WRITE portion of cycle.
0101	L8EA	Output of ALU via S3 is shifted left eight bits, end-around.
0110	F2WR	Write data contained in F register into file 2 at address specified by contents of N register at beginning of current microinstruction. Actual writing takes place during first part of instruction.
0111	AP	Alternate A field coding, A'.
1000	BP	Alternate B field coding, B'.

TABLE 4-19. S FIELD (Contd)

S Code	Mnemonic	Operation
1001	DP	Alternate D field coding, D'.
1010	APDP	Alternate A and D field coding, A'D'.
1011	DPP	Alternate D field coding, D".
1100	GATEI	Gate output of S1 to I register.
1101	HALT	If halt bit of SM register is 1, stop operation of MP on completion of this microinstruction. When start signal is received, continue with next microinstruction specified by addressing mode and T field. If halt bit is 0, continue with microinstruction sequencing.
1110	RTJ	Transfer address of next sequential microinstruction pair to RTJ register. This is done regardless of actual addressing mode used in this instruction.
1111	CLRNP	Clear N register and page register.

TABLE 4-20. C CODE OPERATIONS

C' Code	Mnemonic	Operation
		The following special operations, for format 1, have a zero in bit 19 (SF) to specify C' codes (bit 25 through 31). Bit 24 specifies the T field interpretation.
00xxxxx		xxxxx is a constant for use in driving the bit generator or in any other commands using lower five bits of instruction for control.
0100000 0100001 0100010† 0100011†	WRCH/0 WRCH/1 WRCH/2 WRCH/3	Write 8-bit character specified from output of S3 at memory address specified by output of AB. See WRITE command in S field for details of operation and restrictions. Character 0 is bits 0 through 7 (MSBs); character 1 is bits 8 through 15, etc. Remainder of word in memory is unchanged. Character is not repositioned in WRITE command.
0100100	RMW	Read modify write - Perform a read of information from main memory in same manner as READ instruction in S field. Memory system will perform a read cycle and lock-up before performing write cycle. Memory must be forced to complete write cycle by issuing WRITE instruction, using same address, before it will respond to any other read operations.

TABLE 4-20. C CODE OPERATIONS (Contd)

C' Code	Mnemonic	Operation
0100101	WRHW0	Write bits 0 through 15 (MSBs) from output of S3 at memory address specified by output of AB. Bits 16 through 31 in memory are unchanged. See WRITE command in S field for details of operation and restrictions.
0100111†	WRHW1	Write bits 16 through 31 (LSBs) from output of S3 at memory address specified by output of AB. Bits 0 through 15 in memory are unchanged. See WRITE command in S field for details of operation and restrictions.
0101000	WRPB	Write protect bit - See protect system discussion (Appendix A).
011xxxx	GATEIXT	General purpose strobe at T4 time. In CPU, this signal is used to gate "IXT" on 1700 transform module.
1000101	INCK	Increment number contained in K register by one.
1001101	INCN	Increment number contained in N register by one.
1000100	DECK	Decrement number contained in K register by one.
1001100	DECN	Decrement number contained in N register by one.
1000000	CLRK	Clear K register.
1001000	CLRN	Clear N register.
101xxxx	SETF/j	xxxx is value of j, from 0 to 15. Set SM register flag j to 1.
110xxxx	CLRF/j	xxxx is value of j, from 0 to 15. Clear SM register flag j to 0.
1110000†† or 1110001††	RQLXN	Destination register (P, A, F, or X) and Q register are considered as one double-length register with Q register as lower order bits. Combined register is shifted left one bit position with complement of ALU sign bit entered into lowest bit position of Q register.
1110011††	RQR1E	Shift combined destination and Q register right one bit, and enter 1 in sign position of destination register. This command is used in multiply iteration.
1110010††	RQR0E	Shift combined destination and Q register right one bit, and enter 0 in sign position of destination register. This command is used in multiply iteration.

TABLE 4-20. C CODE OPERATIONS (Contd)

C' Code	Mnemonic	Operation
1110100††	RL0E	Shift destination register left one bit, entering 0 in lowest bit position of the register. This operation cannot be performed when Q is destination register.
1110101††	RL1E	Shift destination register left one bit, entering 1 in lowest bit position. This operation cannot be performed when Q is destination register.
1110110††	RR0E	Shift destination register right one bit, entering 0 in sign position of register. This operation cannot be performed when Q is destination register.
1110111††	RR1E	Shift destination register right one bit, entering 1 in sign position. This operation cannot be performed when Q is destination register.
C" Code	Mnemonic	Operation
The following special operations, for format 1, have a one in bit 19 (SF) to specify the C" codes (bits 25 through 31). Bit 24 specifies the T field interpretation.		
000xxxx†††	TMA/j	xxxx = j, with values from 0 to 15. Obtain next microinstruction pair from address specified by MA transform selector setting j.
001xxxx†††	TMAK/j	xxxx = j, from 0 to 15. Obtain next instruction pair from address specified by MA transform selector setting j. Also, set K register to value specified by K transform selector setting j.
0100xxx†††	GITMAK/j	Gate output of main memory to IXT register (on transform module) and perform TMAK/j operation. Note that j = xxx with values of 0 to 7. Restrictions: This command must be executed in the microinstruction following a READ command.
0101xxx†††	GITMAK/xt	Gate output of main memory to IXT Register (on transform module) and perform a transform of the upper 16 bits of MIR and select one of eight transforms of MA from the decoded and encoded microinstruction loaded into IXT. Restrictions: This command must be executed in the microinstruction following a READ command.

TABLE 4-20. C CODE OPERATIONS (Contd)

C" Code	Mnemonic	Operation
011xxxx†††	TK/j	xxxx = j, with values from 0 to 15. Set K register to value specified by K transform selector setting j.
100xxxx	TN/j	xxxx = j, from 0 to 15. Set N register to value specified by N transform selector setting j.
101xxx	SUB	Upper bounds - transfer output of S3 to upper bounds register in main memory interface.
110xxxx	SLB	Set lower bounds - transfer output of S3 to lower bounds register in main memory interface.
C Code	Mnemonic	Operation
The following format 3 codings are for setting the K and N register; this format has the M field bits 0 and 1 set to 11, while bit 19 selects the register.		
Value	K = value	When microinstruction bit 19 = 0, transfer C field value (bits 24 to 31) to K register and execute next sequential microinstruction pair.
Value	N = value	When microinstruction bit 19 = 1, transfer C field value (bits 24 to 31) to N register and execute next sequential microinstruction pair.
The following format 2 codings in the C field are used to perform a jump, specified if the M field (bits 0 and 1) is 10.		
Number	Number	Number is address of next instruction pair. If page jump is required (bit 19 = 1), S field contains page setting instead of S field code.

†NOTE: Commands applicable to 32-bit processor only.

††RESTRICTION: This operation cannot be performed when S field command (L8EA) is selected.

†††RESTRICTION: If SM 207 is set to "1", S field commands are disabled.

MICROINSTRUCTION TIMING

The basic CPU microinstruction execution time is 168 nanoseconds. Some microinstructions have longer execution times to allow certain operations to be completed. The microinstructions have been grouped according to execution times as A, B, C, D, E, F and G as shown in table 4-21.

The classification of microinstructions is shown in figure 4-2. This figure provides execution time by types for all legal combinations of

microcommands which extend the basic cycle time.

Exceptions to the execution times as listed in figure 4-4 are:

1. The combination of one's complement arithmetic with an ADD+ or ADD+T arithmetic operation is classified as a type B rather than a type C.

2. The MMU and MML B' commands are included under read/write micromemory operand commands; therefore, they are a type F rather than a type B.

3. A type G microinstruction followed by a microinstruction with a GITMAK command (C field) has an additional execution time of 185 nanoseconds (typical); thus the total execution time becomes 625 nanoseconds.

Analysis of a microprogram for execution time starts by classifying each of the microinstructions as type A, B, C, D, E, F, or G. This is done by using the microinstruction classification table or by examining the assembler output listing.

Each microinstruction with a main memory read or write command has a 440 nanosecond (typical) execution time regardless of the type of cycle (assuming no DMA activity on main memory interface). However, type E and F microinstructions cannot contain a main memory reference command. An additional execution time of 185 nanoseconds (typical) is required on all read commands followed by a microinstruction containing a GITMAK command.

The total execution time required is then calculated from main memory command to main memory command. If the total time, starting with a microinstruction with a read or write command and including all microinstructions up to the following read or write command is less than the memory cycle time (600 nanoseconds), take 600 nanoseconds as the execution time for that sequence. If the total time is greater than 600 nanoseconds, the execution time for that sequence is the calculated time. The following is an example of how the execution time for a microprogram sequence would be calculated (figure 4-5).

TABLE 4-21. MICROINSTRUCTION EXECUTION TIMES

Microinstruction Type	Execution Time in Nanoseconds†
A	168
B	224
C	280
D	336
E (shift or scale)	280 + 56n (where n = number of shifts)
F (micromemory read/ write operand)	504
G (read/write main memory)	440 (typical)

†Execution times may vary ±4% over temperature range (0°-70°C) and voltage ±5%.

READ/WRITE MAIN MEMORY	READ/WRITE MICROMEMORY OPERAND	SHIFT OR SCALE	ONE'S COMPLEMENT ARITHMETIC	ADD OR SUBTRACT	TMA, TMAK, GITMAK	A', B', NU, ZL, COL, Z*L	INSTRUCTION TIME
0	0	0	0	0	0	0	A
0	0	0	0	0	0	1	B
0	0	0	0	0	1	0	C
0	0	0	0	0	1	1	C
0	0	0	0	1	0	0	B
0	0	0	0	1	0	1	C
0	0	0	0	1	1	0	C
0	0	0	0	1	1	1	C
0	0	0	1	0	0	0	A
0	0	0	1	0	0	1	B
0	0	0	1	0	1	0	C
0	0	0	1	0	1	1	C
0	0	0	1	1	0	1	D
0	0	0	1	1	1	0	C
0	0	0	1	1	1	1	D
0	0	1	0	0	0	0	E
0	0	1	0	0	1	0	E
0	1	0	0	0	0	0	F
1	0	0	X	X	X	X	G

X = 0 OR 1 (DON'T CARE CONDITION)

Figure 4-4. Microinstruction Classification

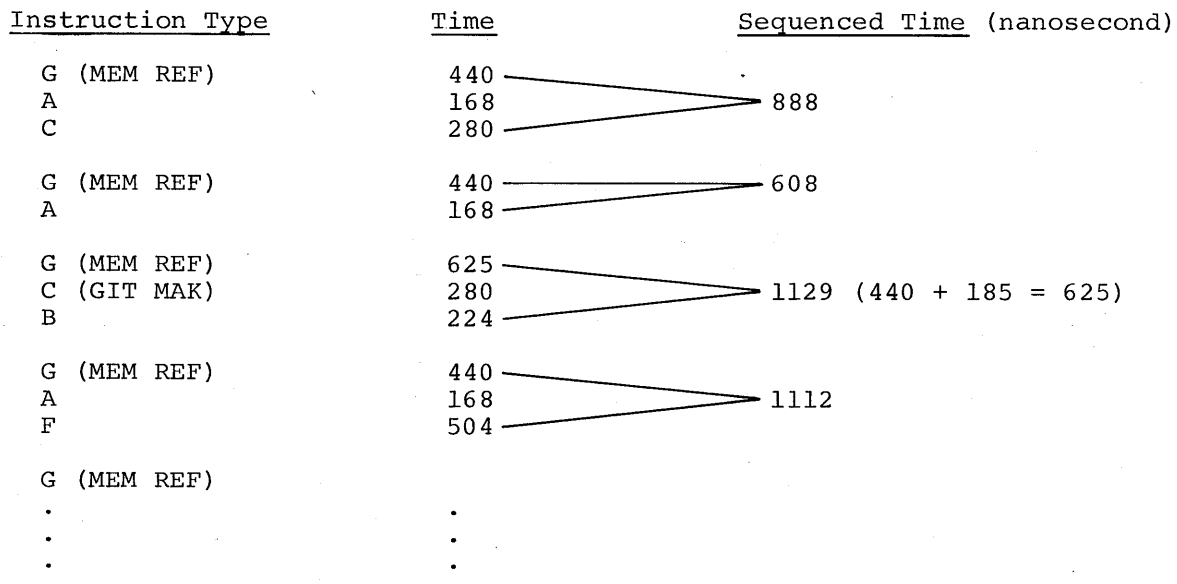


Figure 4-5. Calculation of Microprogram Sequence Execution Time

PROGRAMMING INFORMATION 5

This section describes special programming information for the 2550-2 HCP subsystems and major functional units in the order as follows:

1. Communications Processor (CP)
2. Communications Coupler
3. Multiplex Loop Interface Adapter (MLIA)
4. Loop Multiplexer (LM)
5. Tape Cassette Controller
6. Peripheral Controller (Card Reader/Line Printer)

COMMUNICATIONS PROCESSOR

I/O PROGRAMMING REQUIREMENTS

The pivots of input/output are the A and Q registers of the computer. The Q register designates the equipment to be used; the A register holds function codes, accepts status bits, or serves to transfer data either in or out of the CP in a nonbuffered mode of operation. The use of the Automatic Data Transfer (ADT) mode enables data transfer to and from memory, independent of the internal operation of the CP; the operation is still initiated with the two registers. The programmer must remember that the A and Q registers perform a multitude of operations. The Q register serves as one of the index registers, is used in arithmetic operations, and in transfers between registers, in addition to holding the address of the device during input/output. The A register is the principal arithmetic register. During input/output the A register transmits data and functions and receives status. Either a 16-bit word or 8-bit character can be transmitted to or from the A register. The Q register transmits addresses and control signals.

COMMUNICATIONS CONSOLE

The communications console, which can be either a teletypewriter (TTY) or a cathode ray tube (CRT) device, is connected internally to the A and Q registers. Thus this console operates as if it were a peripheral on the AQ data channel. Any other peripherals, however, use separate controllers plugged into I/O slots in the CP card cage.

Control Signals

READ

The Read signal signifies the request for an input operation. If data is available at the time the Read signal rises, a Reply is returned within four microseconds; if data is not available at the time the Read signal rises, a Reject signal is returned within four microseconds.

WRITE

The Write signal signifies the request for an output operation. If the data can be used at the time the Write signal rises, a Reply is returned within four microseconds.

REPLY

Reply to Write

If the peripheral equipment can accept data when the Write signal rises, the following sequence of events occurs:

1. The CP transfers data to the appropriate register in the peripheral equipment.

2. The peripheral equipment sends a Reply to the channel a minimum of 200 nanoseconds and a maximum of 4 microseconds later.

3. The channel drops the Write signal when it receives the Reply.

4. Absence of a Write signal for 100 nanoseconds drops the Reply.

5. The data lines drop when the Reply drops.

Reply to Read

If data is available when the Read signal rises, the following sequence of events occurs:

1. The data available is gated to the data cable.

2. The Reply is returned a minimum of 200 nanoseconds and a maximum of four microseconds later.

3. The Reply causes the Read line to drop.

4. Absence of a Read signal for 100 nanoseconds causes the Reply to drop.

5. The data lines drop when the Reply drops.

REJECT

If the specified operation cannot or should not be performed at the time a Read or Write signal appears, a Reject will be returned within four microseconds.

PROGRAM PROTECT

The Program Protect signal is present if the I/O instruction requires access to a protected device. If the signal is not present, the protected device returns a Reject signal.

CHARACTER INPUT

This signal is generated by the peripheral device if the data transfer is an 8-bit character or less in the low-order bit positions. Devices which never exceed an 8-bit transfer may have this line up continuously while reading.

Addressing

The Q register is used to send addressing codes to peripheral equipment. The format of the Q register is shown in figure 5-1. Each level of peripheral equipment, except a unit, is addressed by a unique section of the Q register.

CONVERTER

Because it is desirable to have peripheral devices operate interchangeably on the buffered and non-buffered channels, address bits 11 through 15 (W) are reserved for addressing the Buffered Data Channel or similar converter. The (W) field must be zero for lower level peripheral devices and standard peripheral controllers. The 2550 does not use a converter, and as such this field must always be zero.

EQUIPMENT

Address bits 7 through 10 (E) contain the equipment number of the peripheral equipment on the channel (0 through F_{16}). Each device responds when the Equipment Number switch setting matches the code in bits 7 through 10.

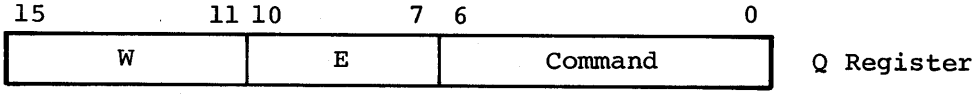


Figure 5-1. Q Register - Address Format

COMMAND CODE

Bits 0 through 6 of the Q register (figure 5-2) are not specifically used by the channel and are therefore available to meet specific requirements of the station and unit. These bits control and direct information on the data cable in the following ways:

1. Specify the data transfer
2. Direct the control functions and function level
3. Direct the status and status level
4. Address the data cable to specific stations under one equipment having multiplexing capabilities

The Command code is divided into two sections: S contains the Station code and D contains the Director. The Station code is located in bit 6 and adjacent lower order bits as required. The Director is located in bit 0 and adjacent higher order bits as required. They cannot overlap and all bits in the Command code are not necessarily used.

If the controller does not contain any stations, the Station code may be used to indicate the type of functions without the need for the A register to be loaded (see figure 5-3).

Units are controlled by a higher-level controller and respond only to the controller. Units on the controller are selected by a function code which directs the data cable (A) to select the unit.

I/O Operations

All input/output operations for the 2550-2 HCP are initiated by the instructions input to A and output from A. The type of transfer during an input or output operation is determined by the Director (bits 0 and upward of the Q register). Bit 0 of the Director determines whether the content of A is data, a function code, or status. The use of the remainder of the Director bits (if any) is detailed in the reference information for each device.

When referencing the communications console (TTY or CRT), the Q register should contain either 0090_{16} or 0091_{16} according to figure 5-4.

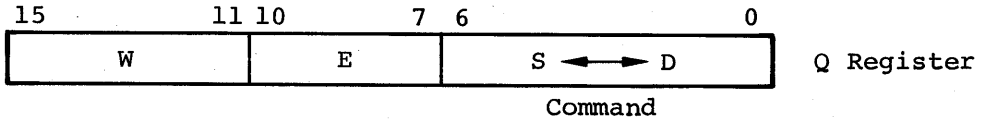


Figure 5-2. Q Register - Command Format

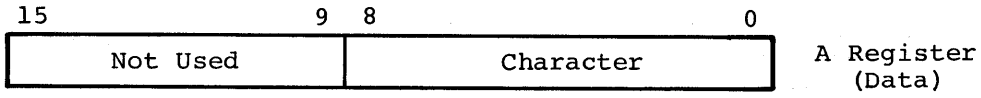
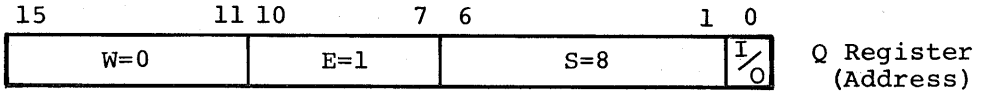


Figure 5-3. A Register - Data Format



Q Register	Output	Input
0090	Write Data	Read Data
0091	Director Function	Director Status

Figure 5-4. Q Register - I/O Address Format

DATA TRANSFER

To transfer data, the Director must equal 0. An Input to A instruction initiates a Read operation and an Output from A instruction initiates a Write operation.

If the peripheral equipment can receive or send data to/from the channel, it sends a Reply. If the peripheral is unable to receive or send data to/from the channel, it sends a Reject. A Read or Write signal will always be rejected if the device is Not Ready.

DIRECTOR FUNCTION

When bit 0 of the address code in the Q register is a 1, all station bits (if any) are 0, and a Write signal is present, the data lines (A) are directed to control the functions of the equipment, including the selection of a unit on a nonmultiplexing device. When bit 0 is set on a multiplexing device, and both the Station code and Write signal are present, the data lines are directed to control the functions of the station within the equipment. Additional bits of Q can be used to direct function levels.

Bit A Reg	Function	Operation
00	Clear Controller	Clear all interrupt requests. Clear Busy, Interrupt, Data, Alarm, Manual Interrupt conditions. Select Read mode. Connect printer. Any interrupt request bit shall take precedence over this function. Clear controller also is used in conjunction with bits 11, 12, 14, and 15.
01	Clear Interrupt	Clears all interrupt requests and the manual interrupt. Any interrupt request bit shall take precedence over this function.
02	Data Interrupt Request	Conditions the controller to send an interrupt signal whenever a Data Status is active.
03	End of Operation (EOP) Interrupt Request	Conditions the controller to send an interrupt signal when the controller is not busy. When in the EOP state, the controller will accept a change in its mode.
04	Alarm Interrupt Request	Conditions the controller to send an interrupt signal when the Lost Data Status is active.
05	Not used	
06	ADT Mode	Conditions the controller for an Auto-Data Transfer (ADT) operation.
07	Not used	
08	Select Write Mode	Conditions the controller for an output operation. Does not clear alarm status.
09	Select Read Mode	Conditions the controller for an input operation.
10 thru 15	Not used	Always zero.

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 instruction is accepted (Reply) but those functions which should be rejected are not executed. When none of the functions can be executed, the output instruction is rejected.

Bit A Reg	Status	Description
00	Ready	Unit is ready.
01	Busy	Read mode - The controller is in the process of receiving a character or is holding data for transfer to the computer. The Busy status will drop upon completion of the data transfer. Write mode - The controller is in the process of transferring data. Busy will drop upon completion of the transfer.
02	Interrupt	An interrupt condition exists in the controller.
03	Data	Read mode - The controller is holding one character of data for transfer to the CP. The data status will drop upon completion of the transfer. Write mode - The controller is ready to accept another character from the CP.
04	Available	Always the inverse of Busy.
05	Alarm	Parity error, lost data or field error (no stop bit when expected) occurred.
06	Lost Data	The buffer contained data for transfer to the CP, and the TTY/CRT 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 I/F and is returned.
09	Read Mode	The controller is conditioned for input operation.
10	Reserved	This bit indicates that the TTY or CRT is currently assigned to the panel I/F and is unavailable to the TTY controller.
11	Manual Interrupt	A manual interrupt has occurred.
12 thru 15	Not used	Always zero.

REAL-TIME CLOCK

The real-time clock which is an integral part of the I/O module is designed to appear as a peripheral to the macrolevel software. Two functions are available to the macrolevel program: Enable/Disable Limit Interrupt and Enable/Disable Clock. Two status bits are also

available to the macrolevel program: Limit Interrupt and Lost Count.

The Enable Clock and Limit Interrupt functions are selected by performing a Write function to the real-time clock (W=0, E=1, and S=7) and the two least significant bits of the Q register equal to one (Q=$00F3). 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 zero (Q=$00F0). Either Write function will clear an existing limit interrupt and clear the status of the real-time clock.

Status of the real-time clock 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 being received by a macrolevel program is dependent on the appropriate M register bit being set and the macrointerrupt enabled as with all other peripherals.

The emulator is capable of receiving a microinterrupt from the real-time clock every 3-1/3 milliseconds (based on a crystal oscillator). This interrupt is enabled any time the define microinterrupt (DMI) instruction has been cleared and defines the microinterrupt address of the clock, and the clock has been enabled as indicated above.

The determination of when the Emulator generates the macrolevel interrupt is dependent on the value stored for the Clock Limit in the ADT table 5-1 for the clock.

If the Emulator becomes overloaded with higher priority microinterrupts and the microinterrupt 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.

The ADT table for this type consists of four words:

1. Word 1, bit 15 must be one.
2. Word 1, bits 11-14 must be zero.
3. Word 1, bits 7-10 contain the equipment of the clock, which is always equal to one.
4. Word 1, bits 0-6 contains the station/director bits of the clock, which is always equal to 70_{16}. (Thus, word 1 should equal $80F0_{16}$.
5. Word 2 is initially set to zero. Whenever the clock has been enabled, the clock counter will be incremented every 3-1/3 milliseconds.

TABLE 5-1. ADT TABLE FOR THE CLOCK

Word	15	14	13	12	11	10	7	6	0
1	1	0	0	0	0	EQUIP		STATION/DIR.	
2	CLOCK COUNTER								
3	CLOCK LIMIT								
4	NOT USED								

6. Word 3 contains the clock limit, which is interpreted as a multiple of 3-1/3 milliseconds. When the clock counter equals the clock limit and the macroclock interrupt is enabled, the macroclock interrupt will occur. Thus, if the clock limit is five, the clock interrupt interval is 16-2/3 milliseconds or 60 times a second. Note that to continue the process, the clock counter should be reset to zero, or the limit counter incremented by its original value (e.g., five). In this later method, the clock counter can function as an elapsed time counter.

7. Word 4 is not currently used and should be zero. It is reserved for future use.

INTERRUPT SYSTEM

The purpose of the interrupt system is to test whether or not certain conditions exist, without having these tests in the main program. Examples of these conditions are faults (internal) and end of operation (in an external equipment). After executing each main program instruction, a test is made for these conditions. If one of these conditions exists and the conditions for interrupting are present, execution of the main program halts. The contents of the Program Address register, P, are stored at a fixed address, and an interrupt routine is initiated. This interrupt routine takes the necessary action for the condition and then returns control to the next unexecuted instruction in the main program.

For each condition that can cause an interrupt, the program has two alternatives. It may select an interruptible condition so that interrupt occurs when that condition arises, or it may choose to have the interrupt system ignore the condition. The program also has the choice of whether the interrupt system is to be used. The EIN and IIN instructions activate and deactivate the interrupt system.

The interrupt system gives the program the ability to establish priority of interrupts so that an interrupt of high priority can interrupt the machine while processing an interrupt of a lower priority. The return path to the lower priority interrupt routine(s) and then to the main program is clearly established and saved.

If all conditions for interrupting have been met, the main program is interrupted just before the next storage reference. Consequently:

1. If conditions for interrupting occur while the CP is reading up an instruction which references storage, the main program is interrupted before that instruction is executed.

2. If conditions for interrupting occur while the CP is reading up an instruction which does not reference storage (e.g., inter-register instruction), interrupt does not occur until after the CP has executed that instruction.

3. If conditions for interrupting occur while the CP is reading up an indirect address and bit 15 is set, the interrupt occurs before that instruction is executed.

In all three preceding cases, the value of P stored at the fixed interrupt trap location enables return to the next unexecuted instruction in the main program after interrupt processing.

Logical Description

The interrupt system consists of fixed interrupt trap locations and the interrupt Mask register.

BASIC AND OPTIONAL INTERRUPTS

The 2550-2 HCP has 16 interrupts, one internal (storage parity error, power failure, or program protect fault, interrupt state 00) and 15 external. Each of these interrupts has its respective bit in the interrupt mask register and its respective address to which control is transferred upon recognizing the interrupt.

INTERRUPT TRAP LOCATIONS

Interrupt trap locations are established for each interrupt line. They are in the range of addresses 0100 through $013C_{16}$. The assignment for each interrupt state or line is shown in table 5-2. The first column is the interrupt state. The second column is the value of delta to be used in the Exit Interrupt instruction to exit from that state. The third column is the address where the contents of the Program Address register are stored when an interrupt occurs. The fourth column is the address of the first instruction to be executed following an interrupt. These addresses are reserved exclusively for interrupts unless that particular interrupt is not being used.

MASK REGISTER

The 16-bit Mask register is the enable for each interrupt state or line. Bit 00 of the Mask register corresponds to interrupt line 0, bit 01 to line 1, etc.

TABLE 5-2. INTERRUPT STATE DEFINITIONS

Interrupt State$_{10}$	Delta Used in Exit State$_{16}$	Location of Return Address$_{16}$	Location of First Instruction After Interrupt Occurs$_{16}$
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

Interrupt System Programming

If an interrupt is desired when one or more specific conditions arise, a number of preparatory steps must first be accomplished by the programmer. These steps are:

1. The interrupt system must be activated.

2. Internal and external conditions to be tested must be selected with various masks.

3. Interrupt routines must be programmed to determine the cause of interrupt and to process and clear the interrupt.

When the CP is processing a particular interrupt, it is defined as being in that interrupt state (00 through 15). Thus, the interrupts and their respective bits in the interrupt Mask register are numbered 00 through 15 (e.g., bit 7 corresponds to interrupt state 7).

Before the CP can recognize any interrupt, the mask bit for that interrupt must be set, and the interrupt system must be activated. The Mask register can be set by an Interregister instruction, and the interrupt system is activated by an Enable Interrupt instruction.

Upon recognizing an interrupt, the CP automatically stores the return address in the storage location reserved for that interrupt state. Bit 15 of the storage location is set or cleared to record the current state of the OVERFLOW indication, providing 32K mode has been selected. If the CP is in 65K mode, 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 later restoration of the OVERFLOW indicator. In both 32K and 65K modes, the interrupt system is deactivated and control is transferred when the interrupt occurs. Also, at this time (32K mode) overflow is cleared; in 65K mode overflow is not cleared until the SOV or SNO instruction is 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 while in this state. The 1s in the mask denote interrupts that have higher priority than the interrupt being processed. The mask should not have a 1 in the position being processed. If an interrupt is allowed into the same state which is being processed, the return link is lost. The program then activates the interrupt system and processes the interrupt.

The CP exits from an interrupt state as follows. The program inhibits interrupt and restores the registers, including the Mask register. If the CP is in 65K mode, the program must restore the overflow condition that existed when the interrupt occurred by clearing any overflow condition with an SOV or SNO instruction, then forcing an overflow condition if one existed when the interrupt occurred. After loading the registers, the program executes the exit interrupt instruction with delta equal to the lower 8 bits of the base address of the interrupt state. This instruction reads the storage location where the return address is stored. The OVERFLOW indicator is automatically set or cleared in accordance with bit 16 if the CP is in 32K mode. The interrupt system is reactivated, and control transfers to the return address.

INTERRUPT PRIORITY

The priority of interrupts is under program control. The program assigns priority by establishing an interrupt mask for each interrupt state which enables all higher priority interrupts and disables all lower priority interrupts. When an interrupt state is entered, the mask for that state is placed in the Mask register. There may be up to 16 levels of priority. It is possible to change

priority during execution of a program.

If two or more interrupts have equal priority and occur at the same time, the computer recognizes the lowest interrupt line.

Table 5-3 and sample program steps apply if there are five different possible interrupts and the programmer wants three levels of priority so that interrupt 01 has high priority, interrupts 02 and 05 have next priority, and interrupt 03 and 04 have low priority. Interrupt 00 has highest priority, but this example does not consider interrupt 00. This example assumes the CP is in 65K mode.

TABLE 5-3. INTERRUPT PRIORITY LEVELS

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

```
Set Mask register to Mask 1
Enable interrupt
----
----
----
----
----
----
```

State 01 Program

```
Store registers
†Check overflow with SOV or SNO
Set Mask to Mask 04
Enable interrupt
----
----
Inhibit interrupt
†Reset overflow condition
Exit interrupt 01
```

State 02 Program

```
Store registers
†Check overflow with SOV or SNO
Set Mask to Mask 03
Enable interrupt
----
----
Inhibit interrupt
†Reset overflow condition
Repleace registers
Exit interrupt 02
```

State 03 Program

```
Store registers
†Check overflow with SOV or SNO
Set Mask to Mask 2
Enable interrupt
----
----
Inhibit interrupt
†Reset overflow condition
Replace registers
Exit interrupt 03
```

State 04 Program

```
Store registers
Check overflow with SOV or SNO
Set Mask to Mask 02
Enable interrupt
----
----
Inhibit interrupt
†Reset overflow condition
Replace registers
Exit interrupt 04
```

State 05 Program

```
Store registers
†Check overflow with SOV or SNO
Set Mask to Mask 03
Enable interrupt
----
----
Inhibit interrupt
†Reset overflow condition
Replace registers
Exit interrupt 05
```

†If 32K mode is selected, disregard this step as it applies to 65K mode only.

SHARING SUBROUTINES BETWEEN INTERRUPT LEVELS

Properly programmed, programs in different interrupt states can reference the same subroutine. The first instruction in the subroutine must be an IIN, and the last two instructions must be EIN and JMP.

Example:

```
Main Program

Interrupt state 1
    -
    -
    -
RTJ α
    -
Interrupt state 2
    -
    -
    -
RTJ α
    -
    -
    -
α  [return link]
α + 1 IIN - inhibit interrupt
                -
                -
                -
                -
                -
EIN - enable interrupt
JMP (Indirect α)
```

Interrupts occurring after the execution of the RTJ are blocked because the IIN is executed. These interrupts are not recognized until after the jump is executed, because one instruction must be executed after an EIN before the interrupt system is active.

PROGRAM PROTECT

The CP has a program protect system which makes it possible to protect a program in the CP from being accessed or executed by any other nonprotected program also in the CP or by an external Direct Memory Access. The system is built around a program protect bit contained in each word of storage. If the bit is set, it means that word is an operand or an instruction of the protected program and may be accessed only by that program. Any other attempted access constitutes a protect violation.

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 bit set. External devices wishing to access protected storage must be set to Protected mode, otherwise a protect violation occurs.

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

Another form of program protection makes use of the Upper and Lower Bounds register in main memory. These are loaded under program control. Any attempted program access to a location less than the contents of the Lower Bounds register, or greater than the contents of the Upper Bounds register, also constitutes a protect violation.

Program Protect Violations

These are the program protect violations.

1. A nonprotected instruction attempts to write in a protected storage location. The contents of the storage location are not changed.

2. 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.

3. An attempt is made to execute a protected instruction following the 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.

4. 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 (OBXX). Those instructions become a nonprotected Selective Stop instruction under these circumstances.

Program protect is enabled by a two-position switch (bit 08 of the function control register). If the switch is not enabling program protect, none of the above violations are recognized, with the exception of external storage access violation.

Set/Clear Program Protect Bit

The program protect bit can be set or cleared in each word of storage only by the program protect instructions (SPB or CPB).

Programming Requirements

For this program protect system to work, the following program requirements must be met:

1. There must be completely checked-out program package which handles all interrupts for the nonprotected program. This program must also be part of the protected program.

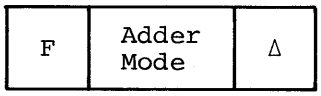
2. The protected program must be a completely checked-out program.

PERIPHERAL EQUIPMENT PROTECTION

All peripheral equipment which is essential to the operation of the protected program must have an operating control which designates whether it is a protected device. If the control is on, the peripheral device responds with a reject to all nonprotected commands (except status request) addressed to it. All protected commands are responded to in the normal manner. If the control is off, the peripheral device responds in the normal manner to protected and nonprotected commands.

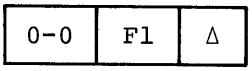
INSTRUCTION SUMMARY

Figures 5-5 through 5-11 illustrate macroinstructions grouped by function: storage reference, register reference, skip, shift, interregister, field reference, and miscellaneous. Table 5-4 contains an alphabetical listing of macroinstructions.



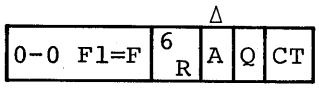
F = 1	JMP
F = 2	MUI
F = 3	DVI
F = 4	STQ
F = 5	RTJ
F = 6	STA
F = 7	SPA
F = 8	ADD
F = 9	SUB
F = A	AND
F = B	EOR
F = C	LDA
F = D	RAO
F = E	LDQ
F = F	ADQ

Figure 5-5. Storage Reference



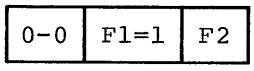

```
F1 = 0   SLS (Δ=0)
F1 = 1   SKIP (see figure 5-7)
F1 = 2   INP
F1 = 3   OUT
F1 = 4   EIN (Δ=0)
F1 = 5   IIN (Δ=0)
F1 = 6   SPB (Δ=0)
F1 = 7   CPB (Δ=0)
F1 = 8   Interregister (see
           figure 5-9)
F1 = 9   INA
F1 = A   ENA
F1 = B   NOP (Δ=0)
F1 = C   ENQ
F1 = D   INQ
F1 = E   EXI
F1 = F   SHIFT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

Figure 5-6. Register Reference



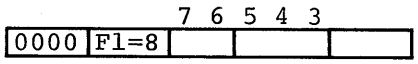

```
010xxxxx  ARS
001xxxxx  QRS
011xxxxx  LRS
110xxxxx  ALS
111xxxxx  LLS
```

Figure 5-8. SHIFT 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
```

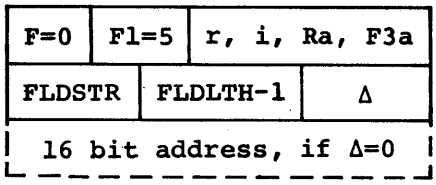
Figure 5-7. SKIP Format



Adder CTL — Orig. Dest.

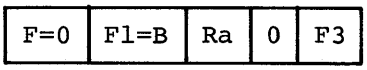
76543	
10000	SET
01000	CLP
10100	TRA
10010	TRQ
10011	TRB
01100	TCA
01001	TCM
01010	TCQ
01011	TCB
00101	AAM
00111	AAB
00110	AAQ
01101	EAM
01110	EAQ
01111	EAB
10101	LAM
10110	LAQ
10111	LAB
11101	CAM
11110	CAQ
11111	CAB

Figure 5-9. Interregister Format



F3a = 2 SFZ
F3a = 3 SFN
F3a = 4 LFA
F3a = 5 SFA
F3a = 6 CLF
F3a = 7 SEF

Figure 5-10. Field Reference



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 = B Ra = 0 APM
F3 = C Ra = 0 PM0
F3 = D Ra = 0 PM1
F3 = 0 Ra = r LUB
F3 = 1 Ra = r LLB
F3 = 2 Ra = r EMS

Figure 5-11. Miscellaneous

TABLE 5-4. MACRO INSTRUCTION MNEMONIC SUMMARY AND EXECUTION TIMES

Mnemonic	Definition	Execution (μsec)†	Op Code			
AAB	Transfer Arithmetic Sum A, Q + M	1.2 - 2.2	0	8	3	8-F
AAM	Transfer Arithmetic Sum A, M	1.2 - 2.2	0	8	2	8-F
AAQ	Transfer Arithmetic Sum A, Q	1.1 - 1.5	0	8	3	0-7
ADD	ADD A	1.6	8	*	Δ	Δ
ADQ	ADD Q	1.6	F	*	Δ	Δ
ALS	A Left Shift	1.6+0.06N	0	F	C/D	0-F
AMr	AND Memory		0	4	*	*
			A	1	Δ	Δ
AND	AND with A	1.5	A	*	Δ	Δ
ANr	AND Register		0	4	*	*
			A	0	Δ	Δ
ARr	Add Register		0	4	*	*
			8	0	Δ	Δ
ARS	A Right Shift	1.6+0.06N	0	F	4/5	0-F
ASC	Accumulator Scale		0	B	0	A
CAB	Transfer Complement Logical Product A, Q + M	1.7 - 2.2	0	8	F	8-F
CAM	Transfer Complement Logical Product A, M	1.7 - 2.2	0	8	E	8-F
CAQ	Transfer Complement Logical Product A, Q	1.1 - 1.5	0	8	F	0-7
CBP	Clear Breakpoint Interrupt		0	B	0	7
CCE	Compare Character Equal		0	4	*	*
			E	2	Δ	Δ
CLF	Clear Field		0	5	+	6
			#	#	Δ	Δ
CLR	Clear to Zero	1.1 - 1.5	0	8	4	0-7
CPB	Clear Program Protect	1.6	0	7	0	0
CrE	Compare Register Equal		0	4	*	*
			E	0	Δ	Δ
DMI	Define Microinterrupt	1.3	0	B	0	6
DrP	Decrement and Repeat		0	6	2,4,6,8 A,C,S,E	
DVI	Divide Integer	9.4- 10.6	3	*	Δ	Δ
EAB	Transfer Exclusive OR A, Q, M	1.7 - 2.2	0	8	7	8-F
EAM	Transfer Exclusive OR A, M	1.7 - 2.2	0	8	6	8-F
EAQ	Transfer Exclusive OR A, Q	1.7 - 2.2	0	8	7	0-7

†See Note

TABLE 5-4. MACRO INSTRUCTION MNEMONIC SUMMARY AND EXECUTION TIMES (contd)

Mnemonic	Definition	Execution (μsec) †	Op Code			
EIN	Enable Interrupt	1.4	0	4	0	0
EMS	Execute Microsequence		0	B	r,0	2
ENA	Enter A	0.9	0	A		
ENQ	Enter Q	0.9	0	C		
EOR	Exclusive OR with A	1.5	B	*		
EXI	Exit Interrupt State	1.9	0	E		
GPE	Generate Character Parity Even		0	B	0	8
GPO	Generate Character Parity Odd		0	B	0	9
IIN	Inhibit Interrupt	1.4	0	5	0	0
INA	Increase A	0.9	0	9	Δ	Δ
INP	Input to A	2.8 - 9.9	0	2	Δ	Δ
INQ	Increase Q	0.9	0	D	Δ	Δ
JMP	Jump	1.1	1	*	Δ	Δ
LAB	Transfer Logical Product A, Q + M	1.7 - 2.2	0	8	B	8-F
LAM	Transfer Logical Product A, M	1.7 - 2.2	0	8	A	8-F
LAQ	Transfer Logical Product A, Q	1.1 - 1.5	0	8	B	0-7
LCA	Load Character to A		0	4	*	*
			C	2	Δ	Δ
LDA	Load A	1.5	C	*	Δ	Δ
LDQ	Load Q	1.5	E	*	Δ	Δ
LFA	Load Field		0	5	*	4/0
			*	*	Δ	Δ
LLB	Load Lower Unprotected Bounds		0	B	r,0	0-F
LLS	Long Left Shift	2.2+0.06N	0	F	E/F	0-4
LMM	Load Micromemory		0	B	0	1
LRG	Load Register		0	B	0	2
LRr	Load Register		0	4	*	*
			C	0	Δ	Δ
LRS	Load Right Shift	2.2+0.06N	0	F	6/7	0-F
LUB	Load Upper Unprotected Bounds		0	B	r,0	0
MUI	Multiply Integer	5.4 - 7.4	2	*	Δ	Δ
NOP	No Operation	1.1	0	F	0/1	0-F
OMr	OR Memory		0	4	*	*
			D	1	Δ	Δ

†See Note

TABLE 5-4. MACRO INSTRUCTION MNEMONIC SUMMARY AND EXECUTION TIMES (contd)

Mnemonic	Definition	Execution (μsec)†	Op Code			
ORr	OR Register		0	4	*	*
			D	1	Δ	Δ
OUT	Output from A	2.4 - 9.5	0	3	Δ	Δ
QLS	Q Left Shift	1.9+0.06N	0	F	A/B	0-F
QRS	Q Right Shift	1.9+0.06N	0	F	2/3	0-F
RAO	Replace Add 1 in Storage	1.9	D	*	Δ	Δ
RTJ	Return Jump	1.6	5	*	Δ	Δ
SAM	Skip if A = -	1.1 / 1.4*	0	1	3	S
SAN	Skip if A ≠ +0	1.1 / 1.4*	0	1	1	S
SAP	Skip if A = +	1.1 / 1.4*	0	1	2	S
SAZ	Skip if A = +0	1.1 / 1.4*	0	1	0	S
SBr	Subtract Register		0	4	*	*
			9	0	Δ	Δ
SCA	Store Character from A		0	4	*	*
			C	3	Δ	Δ
SEF	Set Field		0	5	*	7/F
			*	*	Δ	Δ
SET	Set to '1's	1.1 - 1.5	0	8	8	0-7
SFA	Store Field		0	5	*	5/D
			*	*	Δ	Δ
SFN	Skip if Field Not Zero	1.1 / 1.4*	0	5	*	3/B
			*	*	Δ	Δ
SFZ	Skip if Field Zero	1.1 / 1.4*	0	5	*	2/A
			*	*	Δ	Δ
SIO	Set/Sample Output or Input		0	B	0	4
SJE	Subroutine Jump Exit		0	4	*	*
			5	0	Δ	Δ
SJr	Subroutine Jump		0	4	*	*
			5	0	Δ	Δ
SLS	Select Stop	1.3 / 1.5**	0	0	0	0
SNF	Skip on No Program Protect Fault	1.1 / 1.4*	0	1	B	S

†See Note
*If skip is taken
**If selective stop set

TABLE 5-4. MACRO INSTRUCTION MNEMONIC SUMMARY AND EXECUTION TIMES (contd)

Mnemonic	Definition	Execution (μsec)†	Op Code			
SNO	Skip on No Overflow	{1.1 / 1.4*}	0	1	F	S
SNP	Skip on No Storage Parity Error	{1.1 / 1.4*}	0	1	D	S
SOV	Skip on Overflow	{1.1 / 1.4*}	0	1	A	S
SPA	Store A, Parity to A	2.1	7	*	Δ	Δ
SPB	Set Program Protect	1.6	0	6	0	0
SPE	Skip on Storage Parity Error	{1.1 / 1.4*}	0	1	C	S
SPF	Skip on Program Protect Fault	{1.1 / 1.4*}	0	1	E	S
SPS	Sample Position Status		0	B	0	5
SQM	Skip if Q = -	{1.1 / 1.4*}	0	1	7	S
SQN	Skip if Q ≠ +0	{1.1 / 1.4*}	0	1	5	S
SQP	Skip if Q = +	{1.1 / 1.4*}	0	1	6	S
SQZ	Skip if Q = +0	{1.1 / 1.4*}	0	1	4	S
SRG	Store Registers		0	B	0	3
SrM	Skip if Register Negative	{1.1 / 1.4*}	0	0	3,7 B,F	S
SrN	Skip if Register Non Zero	{1.1 / 1.4*}	0	0	1,5 9,D	S
SrP	Skip if Register Positive	{1.1 / 1.4*}	0	0	2,6 A,E	S
SRr	Store Registers		0	4	*	*
			C	1	Δ	Δ
SrZ	Skip if Register Zero	{1.1 / 1.4*}	0	0	0,4 8,C	S
STA	Store A	1.6	6	*	Δ	Δ
STQ	Store Q	1.6	4	*	Δ	Δ
SUB	Subtract	1.6	9	*	Δ	Δ
SWN	Skip if Switch Not Set	{1.1 / 1.4*}	0	1	9	S

†See Note
*If skip is taken

TABLE 5-4. MACRO INSTRUCTION MNEMONIC SUMMARY AND EXECUTION TIMES (contd)

Mnemonic	Definition	Execution (μsec)†	Op Code			
SWS	Skip if Switch Set	{1.1 / 1.4*}	0	1	8	S
TCA	Transfer Complement A	1.1 - 1.5	0	8	6	0-7
TCB	Transfer Complement Q + M	1.2 - 1.7	0	8	5	8-F
TCM	Transfer Complement M	1.2 - 1.7	0	8	4	8-F
TCQ	Transfer Complement Q	1.1 - 1.5	0	8	5	0-7
TRA	Transfer A	1.1 - 1.5	0	8	A	0-7
TRB	Transfer Q + M	1.2 - 1.7	0	8	9	8-F
TRM	Transfer M	1.2 - 1.7	0	8	8	8-F
TRQ	Transfer Q	1.1 - 1.5	0	8	9	0-7
XFr	Transfer Register		0	7	1-7,0,1-7	

†See Note
*If skip is taken

NOTE: Add 550 ns (approx.) per memory reference for each instruction:

CPU = 750 ns Read	Exp = 1145 ns Read	CPU/Exp Avg = 425 ns
850 ns Write	1305 ns Write	Setup +155 ns
Avg = 800 ns	Avg = 1225 ns	580 ns

CYCLIC ENCODER

Introduction

The cyclic encoder, operating under microinstruction control, is used to compute cyclic redundancy checksum (CRC) and longitudinal redundancy checksum (LRC) characters. CRCs and LRCs are used by some protocols on both inbound and outbound message characters of from 1 to 8 bits, using as a generator any 16th degree or less polynomial (see table 5-5).

The cyclic encoder operates in a serial mode, computing a partial checksum for each character it receives, one bit at a time. The generator polynomials, the number of bits in a message character associated with a generator polynomial, and a shift control bit which determines the direction (right or left) that a message character is shifted out of the X* register into the serial computation logic are stored in programmable read only memory (PROM) on the cyclic encoder card.

After computing a checksum on a message character (inbound or outbound) the results of the computation are accessible by the program until another character is issued to the cyclic encoder; after this time the new checksum is available and the previous one is either destroyed (inbound), or stored on the cyclic encoder card (outbound), but is no longer available to the program.

Characteristics

PERFORMANCE

The cyclic encoder computes a checksum on an 8-bit message

TABLE 5-5. GENERATOR POLYNOMIAL ADDRESSES AND FORMAT

	X* Reg. Bits 12	11	10	9	8	Generator Polynomial	Character Length	Shift Direction
	0	0	0	0	0	$X^{16}+X^{15}+X^2+1$	8 bits	Right
	0	0	0	0	1	$X^{16}+X^{12}+X^5+1$	8 bits	↓
	0	0	0	1	0	$X^{12}+X^{11}+X^3+X^2+X+1$	6 bits	
	0	0	0	1	1	X^6+X^5+1	6 bits	
	0	0	1	0	0	(X^8+1)†	8 bits	
	0	0	1	0	1	$X^8+X^7+X^6+X+1$	6 bits	
	0	0	1	1	0	$X^8+X^7+X^6+X+1$	4 bits	Right
	0	0	1	1	1	Not Assigned	------	-----
	0	1	0	0	0	Not Assigned	------	-----
	0	1	0	0	1	Not Assigned	------	-----
	0	1	0	1	0	$X^{12}+X^{11}+X^3+X^2+X+1$	6 bits	Left
	0	1	0	1	1	X^6+X^5+1	6 bits	Left
	0	1	1	0	0	Not Assigned	------	----
	0	1	1	0	1	↓	------	----
	0	1	1	1	0		------	----
	0	1	1	1	1	Not Assigned	------	----
††	1	0	0	0	0	$X^{16}+X^{14}+X^{12}+X^{10}+X^8+X^6+X^4+X^2+1$	1 bit	Right
↓	1	0	0	0	1	↓	2 bits	↓
	1	0	0	1	0		3 bits	
	1	0	0	1	1		4 bits	
	1	0	1	0	0		5 bits	
	1	0	1	0	1	↓	6 bits	
	1	0	1	1	0	$X^{16}+X^{14}+X^{12}+X^{10}+X^8+X^6+X^4+X^2+1$	7 bits	
↓	1	0	1	1	1	$X^{16}+1$	8 bits	Right
††	1	1	0	0	0	$X^{16}+X^{15}+X^{13}+X^{11}+X^9+X^7+X^5+X^3+X$	8 bits	Left
	1	1	0	0	1	Not Assigned	------	----
	1	1	0	1	0	↓	------	----
	1	1	0	1	1		------	----
	1	1	1	0	0		------	----
	1	1	1	0	1		------	----
	1	1	1	1	0	↓	------	----
	1	1	1	1	1	Not Assigned	------	----

†Also used to generate LRC polynomials

††Used for diagnostics

character in less than 760 nanoseconds following execution of a load X* microinstruction. Computation time decreases by 61 nanoseconds per bit for message characters having less than eight bits.

OPERATING

Information is transferred to the cyclic encoder A* and X* registers from selector S1 under microinstruction control. The A* and X* registers are loadable only and cannot be used as utility registers since their outputs are not available from the card.

A* REGISTER

A 16-bit register used to store the previous partial checksum or initial value for inbound messages, and the initial value to be used with the first character of an outbound message.

The A* word format for a 16th degree generator polymonial is shown in figure 5-12.

Bit 8 corresponds to the highest order coefficient of the CRC polynomial for any length generator polynomial. If a generator polynomial of less than 16th degree is used for CRC computation, the unused bits of the CRC word must be set to zero.

For example, if a generator polynomial of 12th degree is used, bit 8 corresponds to the X^{11} coefficient of the CRC polynomial and bits 7-4 must be set to zero when A* is loaded.

The A* word format for a 12th degree generator polynomial is shown in figure 5-13.

A* must be loaded for each inbound message character and for the first character of an outbound message. The initial value, usually zero, is dependent on the CRC protocol which in some cases can be other than zero.

A* is loaded when the microinstruction S1 field equals 1011 and D^{11} field equals 101 (transfer output of selector S1 to A* register) or if the S field equals 0001 and DD^{11} field equals 101 (transfer output of selector S3 to A register and transfer output of selector S1 to A* register).

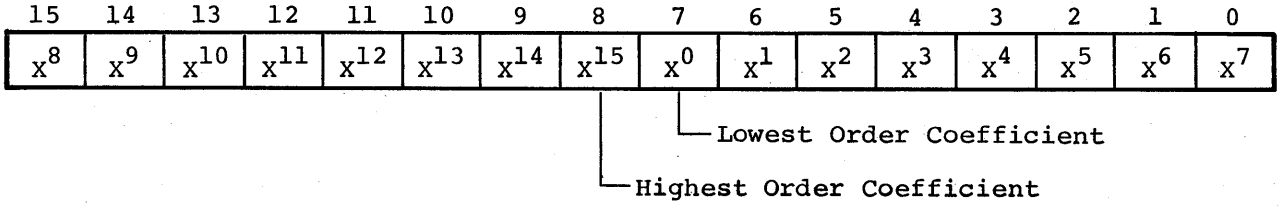


Figure 5-12. A* Register Word Format (16th Degree Generator Polynomial)

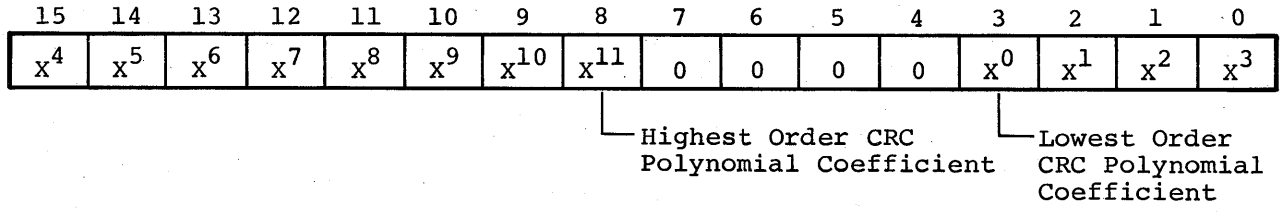


Figure 5-13. A* Register Word Format (12th Degree Generator Polynomial)

X* REGISTER

A 16-bit register used to store the message character for which a partial sum is to be computed, to store the ROM address of the generator polynomial, number of bits in the polynomial, number of bits in the character to be used and shift direction out of X*, and to indicate whether the character is an inbound or outbound character.

The X* word format is shown in figure 5-14.

X* is loaded when the microinstruction S1 field equals 1011 and D^{11} code equals 110 (transfer output of selector S1 to X* register), or if S field equals 0001 and DD^{11} code equals 110 (transfer output of selector S3 to X register, and transfer output of selector S1 to X* register).

CRC FORMAT

The results of a checksum computation are gated to the input bus to selector when the microinstruction S1 field equals 0111 or 1010 (A^1 code) and the A^1 code equals 100, 101, or 110. Only the results of the last CRC computation are available regardless of the A^1 code specified; shift operations and double precision register selection on output from the card cannot be specified by the program.

The CRC word format for a 16th degree generator polynomial is shown in figure 5-15.

Bit 8 corresponds to the highest order bit of the CRC polynomial. If a generator polynomial of less than 16th degree has been used for the CRC computation, then only the bits of the CRC word corresponding to the degree of the generator polynomial minus one are valid. The remaining bits will be set to zero. For example, if a 12th degree generator polynomial has been used, the CRC is contained in bits 8 through 15 and bits 0-3. Bits 4-7 will be set to 0. The CRC format for a 12th degree generator polynomial is shown in figure 5-16.

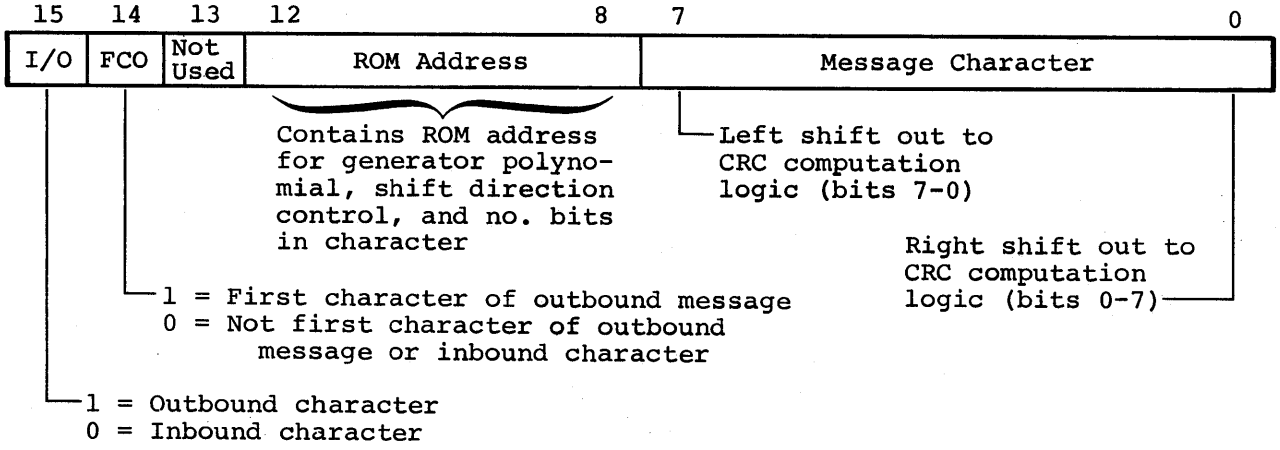


Figure 5-14. X* Register Word Format

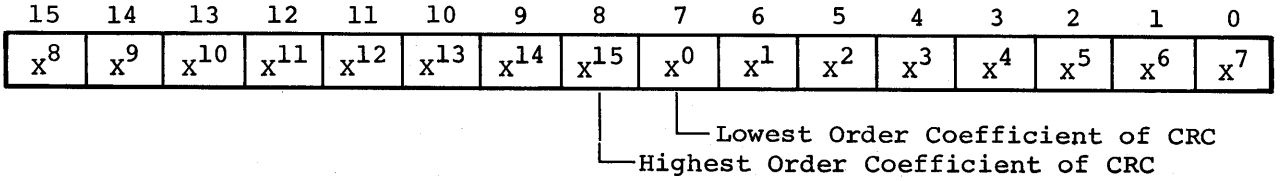


Figure 5-15. CRC Word Format (16th Degree Generator Polynomial)

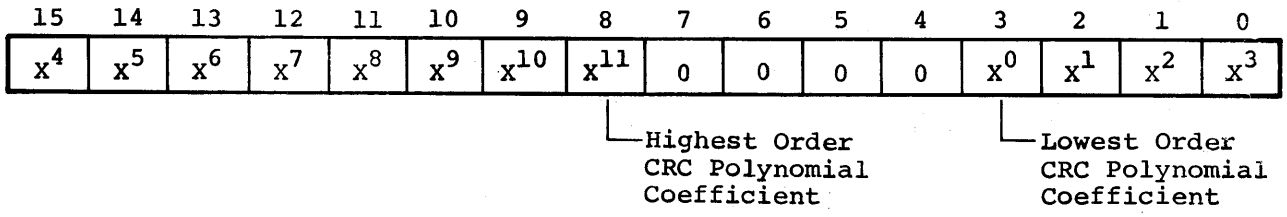


Figure 5-16. CRC Word Format (12th Degree Generator Polynomial)

SEQUENCE OF OPERATIONS

Checksum computation starts when the X* register is loaded. For all inbound message characters and for the first character of an outbound message, A* must be loaded before the X* register.

When a load X* microinstruction is executed, the generator polynomial to be used, the number of bits in the character, and the shift direction control bit are output from the PROM location specified by X* bits 8-12. Bit 15 is then examined to determine the source of information for the checksum calculation. If bit 15=0, indicating an inbound message character, the contents of A* register are used. If bit 15=1, then bit 14 is examined. If bit 14=1, then the contents of A* register are used. If bit 14=0, the contents of S0 register are used for the computation.

Once a serial checksum computation has started, bits 0-7 are shifted out of the X* register (starting with bit 0 if right shift or bit 7 if left shift specified) through the serial encoder logic. The number of bits shifted out is controlled by the number of bits/character from the PROM.

At the completion of a checksum computation for each outbound message character, the partial sum is stored in register S0 to be used for the next outbound message character. The contents of S0 are not accessible by the program. The results of a partial sum calculation, if required, must be taken before the next microinstruction to load X* is executed.

COMMUNICATIONS COUPLER

This part contains information on PPU functions, CP set/sample commands, CP interrupts and interface characteristics for the communications coupler.

PPU FUNCTIONS

All PPU/coupler communications begin with an output function instruction (FAN or FNC), see figure 5-17.

The 12-bit function code of the instruction specifies which operation the coupler will perform. The coupler responds immediately to an output function instruction (by returning an Inactive signal to the data channel) if all of the following conditions exist:

1. Bits 9, 10 and 11 of the function code match the setting of the equipment code switches on the CYBER interface card.
2. The ON-LINE switch is ON.
3. The ENABLE PARITY switch is OFF (6000, CYBER 70) or there is no parity error (CYBER 170).

NOTE

PPU function codes are expressed in octal notation in the descriptions which follow. The coupler equipment code is noted as X (standard assignment is seven).

11 10 9	8 7 6	5	4	3	2 1 0	
Equip Code (=7)	1 0 0	0	0	X	X X X	Clear Coupler
	0 1 0	0		X	X X X	Master Clear
	0 0 1	0		X	X X X	Stop CP
	0 0 0	1		X	X X X	Start CP
	X X X	X		0	0 0 0	Input Memory Address Zero†
	X X X	X		0	0 0 1	Input Memory Address One†
	X X X	X		0	0 1 0	
	X X X	X		0	0 1 1	Input Data
	X X X	X		0	1 0 0	Input CP Status
	X X X	X		0	1 0 1	Input Coupler Status
	X X X	X		0	1 1 0	Input Order Word†
	X X X	X		0	1 1 1	Input Program
	X X X	X		1	0 0 0	Output Memory Address Zero (upper byte)
	X X X	X		1	0 0 1	Output Memory Address One (lower byte)
	X X X	X		1	0 1 0	
	X X X	X		1	0 1 1	
	X X X	X		1	1 0 0	Output Data
	X X X	X		1	1 0 1	Output Program
	X X X	X		1	1 1 0	Output Order Word
	X X X	X		1	1 1 1	

†Hardware Maintenance feature

Figure 5-17. PPU Function Code

Clear Coupler (X400)/PPU Master Clear

This function can reset all coupler control logic. Any operation in process terminates. All error conditions and CP interrupts are cleared. Memory Address Registers Zero and One are cleared; other registers are not affected. Except for bit 2, all coupler status bits are cleared (refer to table 5-6).

Master Clear (X200)

This function directs the coupler to send a momentary external master clear signal to the CP. Both CP micro and macroinstruction execution halt, and the micromemory address register is cleared. Refer to table 5-6.

Stop CP (X100)

This function directs the coupler to send a momentary external stop signal to the CP. CP macroinstruction execution halts.

Start CP (X040)

This function directs the coupler to send a momentary external start signal to the CP. CP microinstruction execution begins at the present contents of the micromemory address register. CP macroinstruction execution begins at the program step indicated by contents of the P register (typically zero).

NOTE

The coupler requires additional PPU I/O instructions to execute the remaining PPU functions.

Input Memory Address Zero (X000)

This function directs the coupler to send the most significant nine bits of the memory address adder output as the lower nine bits of the next channel word. This value is the most significant nine bits of the present 17-bit CP memory address. Bit 2^{11} is Last Character Compare (PCDR = LCDR).

After an input or output data transfer terminates, the present 17-bit address is the FWA of the last buffer operated on. After an input or output program transfer terminates, the 17-bit value is the address of the last word transferred plus one. Note that since the 2550-2 is limited to 65,536 words of main memory, the MSB is always zero. This function is provided only as a hardware maintenance feature.

Input Memory Address One (X001)

This function directs the coupler to send the least significant 12 bits of the memory address adder output as the next channel word. This value is the least significant 12 bits of the present 17-bit CP memory address. This function is provided only as a hardware maintenance feature.

Input Data (X003)

This function directs the coupler to access memory via the DMA channel and to send the 8-bit characters it retrieves as the lower eight bits of following channel words.

The data characters are read from one or more buffers in CP main memory. Memory Address Registers Zero and One contain the starting address of the first buffer. Control words placed in each buffer specify the first and last data character positions within the buffer, and the FWA of the next buffer, if buffer chaining occurs. The CP data buffer format is specified in figure 5-18.

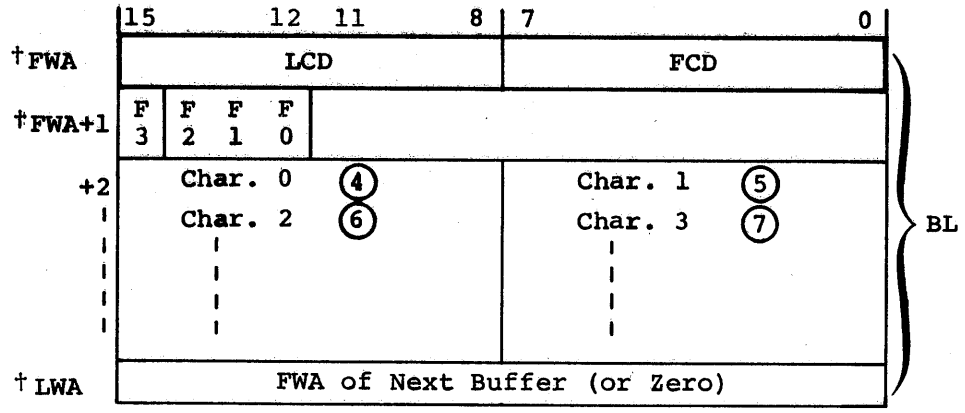
Although the coupler responds to the PPU Function signal, data transfers from main memory only if the Memory Address Register Loaded flag is set. If the flag is not set when the PPU activates the data channel, the coupler waits for the CP to output a memory address before it accesses main memory to start the transfer.

The first time that the data channel is active and the Memory Address Register Loaded flag is set, the coupler performs a DMA read at the FWA. The first character displacement (FCD) control character is loaded into both the FCD and Present Character Displacement (PCD) Registers. The last character displacement (LCD) control character is loaded into the LCD Register.

The coupler then performs a DMA read at the FWA+1 and stores the four MSBs of this word in the Flag Register. If bit 15 is set, the coupler automatically terminates transfer after it empties the buffer. Statuses of the four flag bits are always sent to the PPU with the last character of each buffer.

Data characters are read from main memory starting at the location specified by the PCD Register. Each DMA read retrieves two 8-bit characters. If the FCD is even, both characters retrieved during the first read are sent to the CYBER data channel. If the FCD is odd, only the lower eight bits of the first main memory location are sent.

Characters are transferred to the CYBER data channel in sequential order of thier associated character displacements until the character read from the LCD position is accepted.



FWA = First Word Address of Buffer (must be multiple of BL)

LWA = Last Word Address of Buffer (LWA = FWA + (BL-1))

LCD = Last Character Displacement (relative to FWA)

FCD = First Character Displacement (relative to FWA)

F3 = Last Buffer Flag

F2-F0 = Last Character Flag (place succeeding character in next buffer(s))

BL = Buffer Length (words) $BL=2^N$, $2 \leq N$ (integer) ≤ 7

CD = Character Displacement ($4 \leq CD \leq 253$)

†Control words placed in core before data I/O

NOTE: LCD≥FCD. If LCD<FCD, buffer is skipped and operation points to the next buffer chain.

Figure 5-18. CP Data Buffer Format

If the Flag Register MSB is reset the coupler then performs a DMA read at the LWA. If the LWA contains a non-zero value, this value is loaded into the least significant 16 bit positions of Memory Address Registers Zero and One. The coupler then chains to that address to begin emptying another data buffer.

If the contents of the LWA is zero, the coupler sets the Chain Address Zero flag and generates a CP interrupt. Main memory transfers are now suspended until the CP outputs a new memory address. The coupler then chains to that address to begin emptying another data buffer.

Normal termination of the input data operation occurs when the coupler finds the Last Buffer Flag set after the character specified by the LCD position is accepted by the CYBER data channel.

After the last word is accepted the coupler sends an Inactive signal and generates a CP interrupt. Coupler status after a normal termination of input data operation is octal 0040/ hex 0020.

If the PPU disconnects the data channel before the last data character is accepted, the coupler sets the Transfer Terminated by PPU flag and generates a CP interrupt.

If the CP outputs the terminate transfer command at any time during input data operation, the coupler sets the Transfer Terminated by CP flag, sends an Inactive signal to the CYBER data channel, and generates a CP interrupt.

Input CP Status (X004)

This function directs the coupler to send the 12-bit contents of the CP Status Register as the next channel word(s). After each word is accepted by the data channel, the CP Status Accepted flag is set, the lower eight bits of the CP Status Register are cleared, bit 2 of the coupler status word is cleared, and a CP interrupt is generated.

This function enables the PPU to obtain supervisory information from the CP since the CP Status Register is loaded by the CP.

Single or multiple words are transferred depending on whether the coupler receives an Empty or an Inactive signal after any word is sent and whether or not the CP Status Register is reloaded by the CP. If the CP Status Register is not loaded, the coupler does not return a Full signal (and data) until the CP loads the register.

NOTE

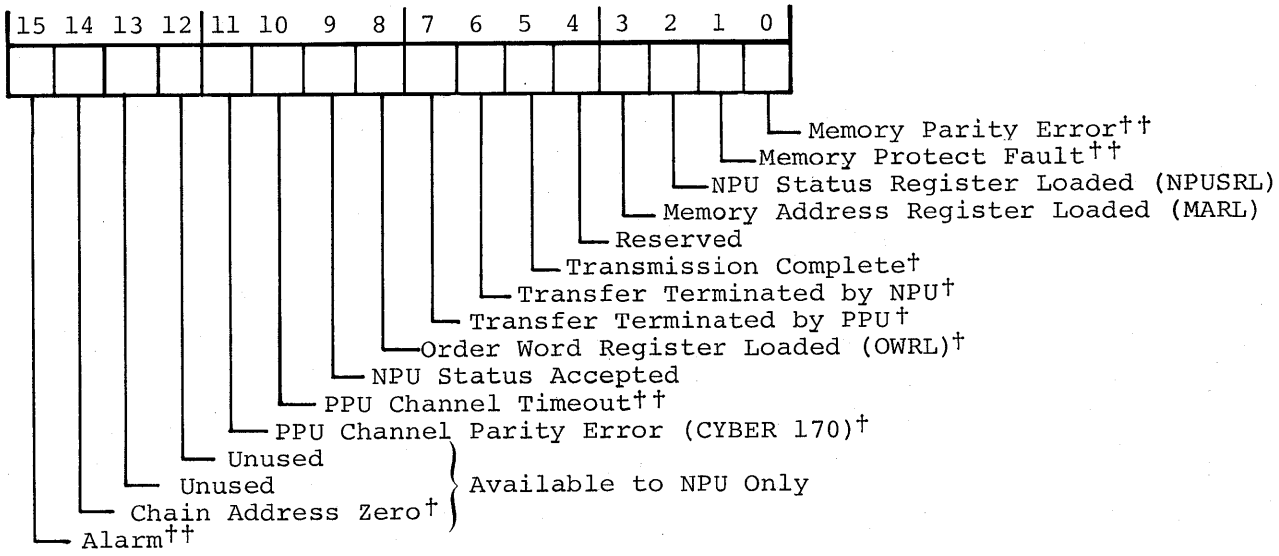
Occurrence of this event can cause a "hang" condition.

The PPU can determine if the CP Status Register is loaded by inputting coupler status and testing bit 2 (see next paragraph).

Input Coupler Status (X005)

This function directs the coupler to send the state of various flags in the coupler as the next channel word. See figure 5-19 and table 5-6. Note that only status bits 0-11 are available to the PPU.

This function clears all coupler bits (except bit 2) and associated interrupts.



††Alarm Condition (all alarms generate NPU Interrupt)
†NPU Interrupt Condition (NPU-Network Processing Unit)

Figure 5-19. Coupler Status Format

TABLE 5-6. COUPLER STATUS SET/CLEAR CONDITION

Bit	Description	Set Condition	Clear Condition
0	Memory Parity Error	Parity Error detected by hardware	Coupler Status read by NPU, CC
1	Memory Protect Fault	Protect violation detected by NPU hardware	Coupler Status read by NPU, CC
2	NPU Status Register Loaded	NPU Status Register loaded by NPU	NPU Status Register read by PPU
3	Memory Address Register Loaded	Memory Address Register loaded by PPU or NPU	Host channel Master Clear input or output operation begun using the Memory Address Register (i.e., any DMA cycle); CC
4	Reserved	---	---
5	Transmission Complete	Input or output operation	Coupler Status read by NPU, CC
6	Transfer Terminated by NPU	NPU sends 079C command	Coupler Status read by NPU; CC
7	Transfer Terminated by NPU	PPU sends Inactive before character having bit 11 set is transferred in either direction	Coupler status read by NPU; CC
8	Order Word Register	Order Word Register loaded by PPU	Order Word Register read by NPU; CC
9	NPU Status Accepted	PPU channel returned empty after Input NPU Status	Coupler Status read by NPU; CC
10	PPU Channel Timeout	Inactive sent to PPU by coupler after being Active 3 seconds (approximately)	Coupler Status read by NPU, CC
11	PPU Channel Parity Error (CYBER 170)	12-bit word plus parity from data channel not odd parity	Coupler Status read by NPU, CC
12	---	---	---
13	---	---	---
14	Chain Address Zero	Coupler finds all zeros in last word of buffer	Coupler Status read by NPU, CC
15	Alarm	Any alarm condition present	Coupler Status read by NPU, CC

NOTE: In the above, "CC" means Clear Coupler function (PPU) or command (NPU). It is also generated either by the Clear CP function (PPU) or the Master Clear command (NPU), but not by the host channel Master Clear signal.

Input Order Word (X006)

This function directs the coupler to send the 12-bit contents of the Order Word Register as the next channel word.

This function is provided only as a hardware maintenance feature, and does not clear any CP interrupt conditions.

Input Program (X007)

This function directs the coupler to access CP main memory via the DMA channel and to send the 8-bit characters it retrieves as the lower eight bits of following channel words.

The coupler does not use main memory control words, and does not buffer-chain in this operation.

Program character-pairs are read from sequential main memory locations starting at the 17-bit address stored in Memory Address Registers Zero and One plus one. The first program character sent to the data channel is from the upper eight bit positions of the FWA.

Termination occurs when the PPU disconnects the data channel after accepting the desired number of characters. Coupler status after normal termination is octal/hex 0000.

Output Memory Address Zero (X101)

This function directs the coupler to load the lower nine bits of the following channel word into Memory Address Register Zero. This value is the most significant nine bits of the 17-bit CP buffer starting address (FWA).

Output Memory Address One (X011)

This function directs the coupler to load the lower eight bits of the following channel word into Memory Address Register One.

The value loaded into Memory Address Register One with this function is the least significant eight bits of the 17-bit CP buffer starting address (FWA).

PPU functions X010 and X011 are normally used to establish the starting address of an output program or input program transfer.

Output Data (X014)

This function directs the coupler to transfer the lower eight bits of following channel words to CP main memory via the DMA channel. The data characters are written into one or more buffers in CP main memory. Memory Address Registers Zero and One contain the starting address of the first buffer.

Control words placed in each buffer specify the first and last data character positions within the buffer, and the FWA of the next buffer if buffer chaining occurs. The CP data buffer format is specified in figure 5-1.

Although the coupler responds to the PPU Function signal, data transfers to main memory only if the Memory Address Register Loaded flag is set. If the flag is not set when the PPU activates the data channel and outputs a word, the coupler waits for the CP to output a memory address before it returns an Empty signal to the data channel.

The first time that the data channel is active and the Memory Address Register Loaded flag is set, the coupler performs a DMA read at the FWA. The FCD control character is loaded into both the FCD and PCD Registers. The LCD control character is loaded into the LCD Register.

If the FCD is odd, the first character from the CYBER data channel is written into main memory with the upper eight bits of the location filled with zeros. If the FCD is even, the first character from the data channel is held in the coupler

until the next data channel character is received. The character-pair is then written into main memory.

After each character is accepted from the data channel, the PCD Register is incremented and points to the next character position in the buffer to be filled.

Data channel characters are written into CP main memory in pairs until the coupler receives the character to be stored at the LCD position. If the LCD is even, the character is written into main memory with the lower eight bits of the location filled with zeros.

After a character is stored at the LCD position, the coupler performs a DMA read at the LWA. If the LWA contains a non-zero value, this value is loaded into the least significant 16 bits of Memory Address Registers Zero and One. The coupler then chains to that address to begin filling another data buffer.

If the contents of the LWA is zero, the coupler sets the Chain Address Zero flag and generates a CP interrupt.

Main memory transfers are now suspended until the CP outputs a new memory address. The coupler then chains to that address to begin filling another data buffer.

Buffer chaining can also be initiated from the PPU. Each time a word is accepted from the PPU, bits 8, 9, 10 and 11 are stored in the Flag Register. Anytime bit 8, 9, 10 or 11 is non-zero, the lower eight bits of the PPU word are stored, the coupler then stores the LCD and FCD at FWA and stores the four flag bits at FWA+1.

If PPU bit 11 was not set (but bit 8, 9, or 10 was set), the coupler next examines the contents of LWA as described above.

If PPU bit 11 was set, the coupler waits for an Inactive signal, then sets the Transfer Terminated by PPU flag, and interrupts the CP.

Coupler status after normal termination of output data operation is octal 0240/hex 00A0.

If the CP outputs the terminate transfer command, the coupler sets the Transfer Terminated by CP flag, generates a CP interrupt, and sends an Inactive signal to the data channel.

Output Program (X015)

This function sets the coupler Program Protect flag and directs the coupler to transfer the lower eight bits of following channel words to CP main memory via the DMA channel.

The coupler does not use main memory control words, and does not buffer-chain in this operation.

Data channel character-pairs are written into sequential main memory locations starting at the 17-bit address stored in Memory Address Register Zero and One. The first data channel character is placed in the upper eight bit positions of the FWA; the second character, in the lower eight bit positions.

Termination occurs when the PPU disconnects the data channel after the desired number of characters have been transferred. Bit 2^{11}, of the channel word with the last character, must not be set. Coupler status after normal termination is octal/hex 0000.

Output Order Word (X016)

This function directs the coupler to load the following channel word(s) into the Order Word Register and to then set the Order Word Register Loaded flag and generate a CP interrupt.

This function enables the PPU to pass supervisory information to the CP since the CP can sample the Order Word Register by the input order word command. Multiple word transfers can occur provided the CP inputs the Order Word Register each time it is loaded by the PPU. If the PPU outputs an order word and the Order Word Register is already loaded, the coupler does not return an Empty signal until the CP inputs the previous orderword in the Order Word Register.

CP SET/SAMPLE COMMANDS

Unlike the peripheral devices previously described (e.g., console, real-time clock) which respond to 1700-A/Q-type commands, the coupler (like the MLIA, discussed later) communicates using NCR-compatible MO5 commands. Thus the SIO command is used instead of a Read or Write instruction. Bit 03 determines direction: Q03=0 corresponds to Sample (read), Q03=1 corresponds to Set (write) (see figure 5-20).

Port addresses are determined by CP backplane wiring. The standard coupler port address is four (E='C'). The optional coupler port address is five (E='D').

NOTE

In the descriptions which follow CP set/sample instruction codes are expressed in hexadecimal notation and a coupler port address of four is assumed.

Input Memory Address Zero (0600)†

This command directs the coupler to gate the most significant nine bits of the memory address adder output to the least significant nine bits of the input data lines. Bit 2^{11} is Last Character Compare (PCDR = LCDR).

Input Memory Address (0610)

This command directs the coupler to gate the least significant 16 bits of the memory address adder output to the input data lines. After an input or output data transfer terminates, the present CP 17-bit address is the FWA of the last buffer operated on. The MSB of the present CP 17-bit memory address is available in the CP status.

Input First/Present Character Displacement (0630)

This command directs the coupler to gate the contents of the FCD Register to the lower eight bits of the input data lines and the contents of the PCD Register to the upper eight bits.

Input CP Status (0640)*

This command directs the coupler to gate the contents of the CP Status Register to the lower 12 bits of the input data lines. Bit 2^8 is the MSB of Memory Address Register 0.

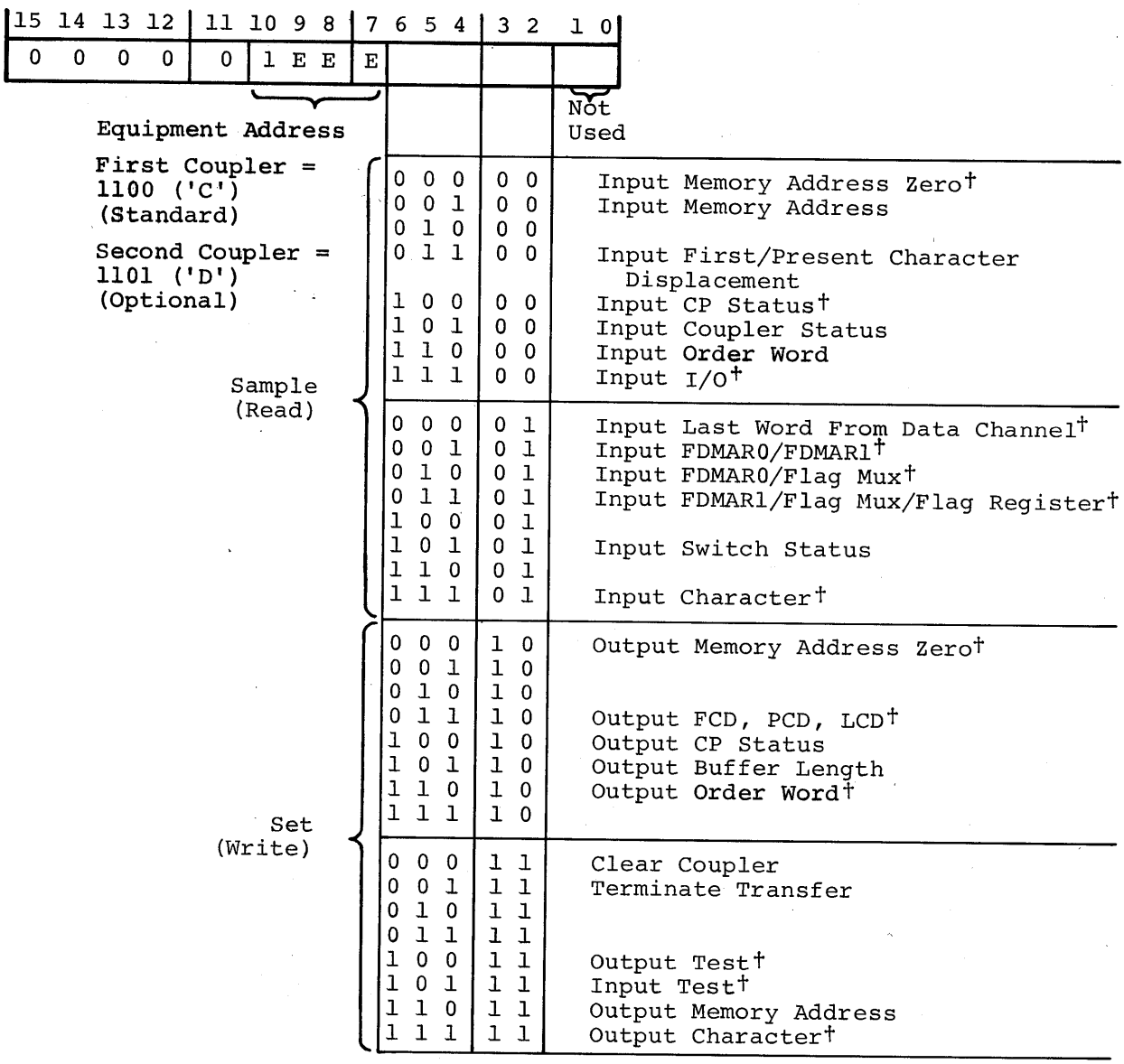
Input Coupler Status (0650)

This command directs the coupler to gate the state of various flags in the coupler to the input data lines. This command clears all non-alarm CP interrupt conditions except OWRL. Coupler status bits are specified in figure 5-19 and table 5-6.

Input Order Word (0660)

This command directs the coupler to gate the contents of the Order Word Register to the lower 12 bits of the input data lines.

†This command is a hardware maintenance feature.



†Hardware maintenance feature

Figure 5-20. CP Set/Sample Instruction Format

This command enables the CP to obtain supervisory information from the PPU, since the Order Word Register is loaded by the PPU. This command clears the Orderword Loaded flag and interrupt, and allows multiple words to be transferred from the PPU to the CP program.

Input I/O (0670)†

This command directs the coupler to gate the lower 12 bits of the output data lines to the lower 12 bits of the input data lines.

Input Last Word from Data Channel (0604)†

This command directs the coupler to gate the contents of the From Channel Buffer to the lower 12 bits of the input data lines.

Input FDMAR0/FDMAR1 (0614)†

This command directs the coupler to gate the contents of the From DMA Channel buffer to the input data lines.

Input FDMAR0/Flag Mux (0624)†

This command directs the coupler to gate the lower eight bits of the From DMA Channel buffer to the lower eight bits of the input data lines. The output of the Flag Mux is gated to the next significant four bits.

Input FDMAR0/Flag Mux/Flag Register (0634)†

This command directs the coupler to gate the lower eight bits of the From DMA Channel buffer to the lower eight bits of the input data lines, the Flag Mux output to the next most significant four bits, and the contents of the Flag buffer to the upper four bits of the input data lines.

Input Switch Status (0654)

This command directs the coupler to gate the setting of all six switches plus two status flags to the upper eight bits of the input data lines.

Status bit configuration is specified in figure 5-21.

Bit 15 of this status is Character Request. Since the sample and set commands have no reply/reject capability, Character Request can be statused to determined whether or not the coupler has a character available during input test operation, or needs a character during output test operation.

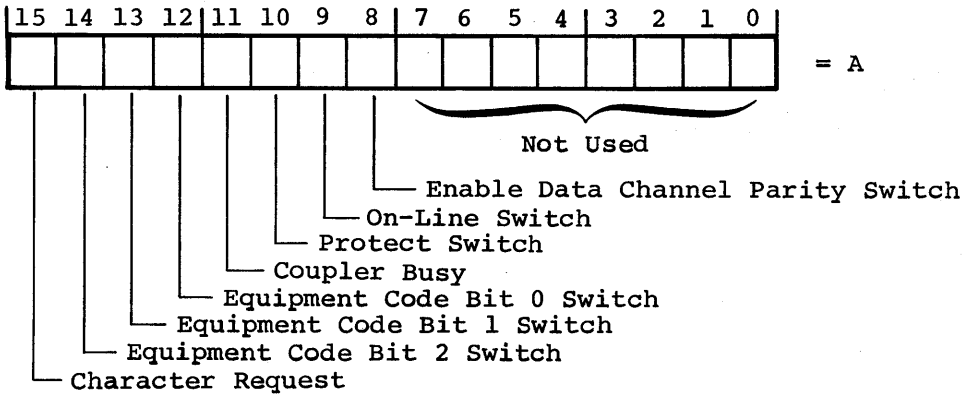


Figure 5-21. Status Bit Configuration

†This command is a hardware maintenance feature.

Bit 11 of this status is coupler busy. It is true whenever the channel is active or Input Test or Output Test operations are in progress. It clears when these conditions are no longer present.

Input Character (0674)†

This command directs the coupler to gate a data character (and flag bits) to the lower 12 bits of the input data lines. This command is issued only during input test operation, and the 12 bits gated would be sent to the CYBER data channel during input data operation.

Output Memory Address Zero (0608)†

This command directs the coupler to load the least significant nine bits of the output data lines into Memory Address Register Zero (upper nine bits of the Memory Address Register).

Output FCD/PCD/LCD (0638)†

This command directs the coupler to: (a) load the least significant eight bits of the output data lines into both the FCD and PCD Registers, and to (b) load the most significant eight bits of the output data lines into the LCD Register.

Output CP Status (0648)

This command directs the coupler to load the lower 12 bits of the output data lines into the CP Status Register and to set the CPSRL flag. Bit 2^8 is also loaded into the MSB position Memory Address Register Zero (MSB of the 17-bit present CP memory address).

This command enables the CP to pass supervisory information to the PPU since the PPU can sample the CP Status Register by the input CP status function. This command allows multiple words to be transferred to the PPU from the CP program.

Output Buffer Length (0658)

This command directs the coupler to load the lower eight bits of the output data lines into the Buffer Length Register. This value is the CP buffer word length, including three control words, minus one. The buffer chain address is always located at the FWA + BL-1.

Output Order Word (0668)†

This function directs the coupler to load the lower 12 bits of the output data lines into the Order Word Register, and to then set the Order Word Register Loaded flag and then generate a CP interrupt.

Clear Coupler (060C)/CP Master Clear

Either this command or a CP Master Clear resets all error conditions and interrupts. CP Status Register, Memory Address Registers Zero and One, the DMA Channel Register, Order Word Register and PCD Register are cleared; other registers are not affected. The CP Status Register Loaded flag is cleared, but not the CP Status Register content. An input test or output test operation in process terminates. Note that the CP Clear has a disastrous affect on operations initiated by the PPU due to the clearing of the registers listed above.

Terminate Transfer (061C)

This command directs the coupler to immediately terminate transfers and to set the Transfer Terminated by CP flag. If the data channel is active and the coupler is connected, an Inactive signal is sent.

---

†This command is a hardware maintenance feature.

The values loaded into Memory Address Registers Zero and One with this command are the least significant 16 bits of the 17-bit CP buffer starting address (FWA).

This command is normally used to establish the starting address of an input or output data transfer. Memory Address Register Loaded must be set before the coupler will access CP main memory during test operations. If the command is issued while the coupler is idle during a chain address zero condition, transfers resume. The program protect status of this instruction is loaded into the coupler Program Protect flag buffer.

Output Test (064C)

This command directs the coupler to interpret the lower 12 bits of the output data lines during following output character commands as data words from the PPU.

Operation is identical to output data operation except the source of the data is the lower 12 bits of the output data lines instead of the CYBER data channel. During output test operation no signals are sent to the CYBER data channel.

Input Test (065C)

This command directs the coupler to access CP main memory via the DMA channel and to gate the 8-bit characters it retrieves to the lower bits of the input data lines during following input character commands. Operation is identical to input data operation except the destination of the data is the lower 12 bits of the input data lines instead of the CYBER data channel. During input test operation no signals are sent to the CYBER data channel.

Output Memory Address (066C)

This command directs the coupler to load the output data lines into the least significant 16 bits of Memory Address Registers Zero and One and to set the Memory Address Register Loaded flag.

Output Character (067C)†

This command directs the coupler to accept the lower 12 bits of the output data lines as a data character or checkword. This command is issued only during output test operation, and the 12 bits accepted would come from the CYBER data channel during output data operation.

CP INTERRUPTS

The coupler interrupts the CP via one macrolevel interrupt. The standard coupler interrupt assignment is six; the optional coupler interrupt assignment is five. Each interrupt condition is identified by a coupler status bit. Bits A00, A01, A05 through A15 cause an interrupt when set.

TAPE CASSETTE CONTROLLER

This part contains programming information for the tape cassette controller status and control functions.

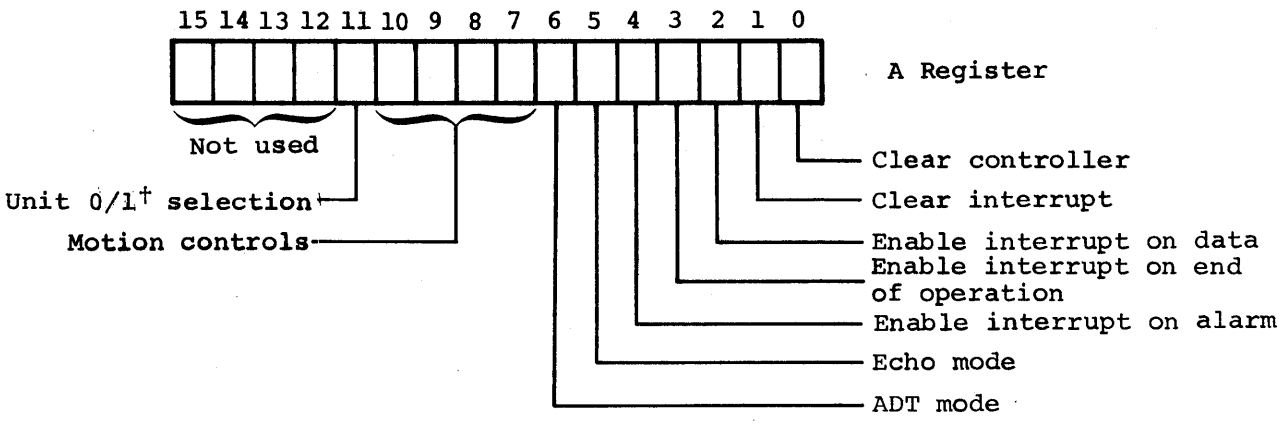
CONTROL FUNCTIONS

When Q00=1, and it is accompanied by both an equipment address and an Output From A instruction, the control function specified by the bits in the A register will be performed. These control functions which can be specified by the A register are shown in figure 5-22.

NOTE

The tape cassette equipment address is normally hexadecimal seven (0111).

†This command is a hardware maintenance feature.



†A manual switch on the controller card can override this function bit and force Unit 1 to always be selected. Standard 2550 systems always have this switch on.

Figure 5-22. Control Functions Specified by A Register

It should be noted that motion control commands (table 5-7) can only be accepted when the tape transport is Ready, and the controller is Not Busy.

The following are brief descriptions of the 15 control functions provided by the tape cassette controller.

Clear Controller

This command functions as a momentary Master Clear and clears all interrupt conditions (Status bits A03-A06, A08, A09, and A14), all interrupt enables (set with Function bits A02-A04), and the interrupt responses (Macro and Micro) between the controller and MP interrupt hardware. In addition, this command clears motion requests, BOT Status, Tape Mark Status, ADT Mode, and Echo Mode.

Clear Interrupt

This command clears the interrupt responses between the controller and the MP by clearing all interrupt enables. However, the command does not clear the interrupt conditions, which are still available as Status bits (A03-A06, A08, A09, and A14).

TABLE 5-7. TAPE CASSETTE CONTROLLER - MOTION CONTROLS

A Register Bits				Function
10	9	8	7	
1	0	0	0	Search Tape Mark (Reverse)
1	0	0	1	Search Tape Mark (Forward)
1	0	1	0	Write Tape Mark
1	1	0	0	Backspace One Record (or Tape Mark)
1	1	0	1	Rewind
1	1	1	0	Erase
1	1	1	1	Read One Record

Enable Interrupt on Data

This command allows an interrupt to be generated whenever the controller has data available (Read) or requests data (Write). Data interrupts are normally cleared by accepting or providing the data. They may also be cleared by Master Clear, Clear Controller, or Clear Interrupt commands.

Enable Interrupt on End of Operation

This command allows an interrupt to be generated when the controller has completed a motion command. The interrupt occurs when the controller becomes Not Busy. EOP interrupts are normally cleared when a new motion command is received. They may also be cleared by Master Clear, Clear Controller, or Clear Interrupt commands.

Enable Interrupt on Alarm

This command allows an interrupt to be generated when an alarm condition occurs. The alarm interrupt (when enabled) is generated by any of the following conditions:

1. End of Tape (Status bit A09=1)
2. Transport Not Ready (Status bit A00=0)
3. Overflow (Lost Data during Read) (Status bit A06=1)
4. Underflow (Lost Data during Write) (Status bit A06=1)
5. CRC Error (Status bit A08=1)
6. Format Error (Status bit A08=1)

Echo Mode

This command causes the controller to internally loop-back the data path. Data normally sent to the cassette, passes through the controller write and read circuitry and becomes data from the cassette. In this way, the integrity of the controller write and read logic can be tested independently of the transport. This is primarily a maintenance aid.

ADT Mode

This command directs the controller to operate in auto-data transfer (ADT) mode. To operate in ADT mode, an End of Operation interrupt must be enabled (A02=1), in conjunction with either a read motion or write motion command or else the function will be rejected.

Search Tape Mark (Reverse)

This command directs the controller to read the tape at 7.5 in. (19cm) per second in the reverse direction. When a tape mark is found, the tape transport stops. No data is transferred to the MP. Note that if this command is issued and no tape marks are encountered before BOT is detected, the tape ultimately stops at load point and the BOT status bit is set.

Search Tape Mark (Forward)

This command directs the controller to read the tape in the forward direction. When a tape mark record is found, the tape transport stops. No data is transferred to the MP. Note that if this command is issued and no tape marks are encountered before EOT is detected, tape motion ceases with the EOT status bit set.

Write Tape Mark

This command initiates the write tape mark operation. For this motion command, the MP does not go to data transfer mode (Q00=0), but handles all activities automatically without external influence. The sequence of events as tape moves forward is as follows:

1. IRG
2. Preamble
3. CRC (all Zeros)
4. Postamble
5. IRG

Write One Record

This command initiates write motion. The MP goes into transfer mode so that the data can be converted from 8-bit parallel to serial form and properly recorded on the tape (Manchester code) with IRG, preamble, data, Cyclic Redundancy Checksum (CRC), postamble, and IRG. Data records are usually one to 256 characters in length (ECMA standard), but longer records can be written if desired.

Backspace

This command moves the tape backward (Reverse) one record; either a data record or a tape mark record. If BOT status is encountered before a record is detected, tape motion ceases and BOT status is set.

Rewind

This command causes the tape to rewind to the beginning-of-tape (BOT) position. The BOT status bit will then be on. Due to the nature of cassette BOT/EOT/leader/trailer characteristics, a rewind command will always result in tape motion continuing to the physical beginning of the tape (transparent leader) followed by a forward movement to BOT. Note that BOT status is a logically derived signal, reset by Clear Controller/Master Clear.

Erase

Upon receipt of this command, the tape will move forward and be erased for approximately 3-4 in. (8-10 cm). The resultant tape will appear as a long IRG.

Read One Record

This command initiates read motion in the forward direction; data records cannot be read in the reverse direction. Preamble, CRC and postamble are stripped off; only the originally recorded data characters are sent to the MP. Records may be of any length.

STATUS

When A00=1 and is accompanied by both an equipment number (address) and an Input to A instruction, cassette controller status is transferred to the A register. The controller always responds to an MP request. The Status responses are shown in figure 5-23.

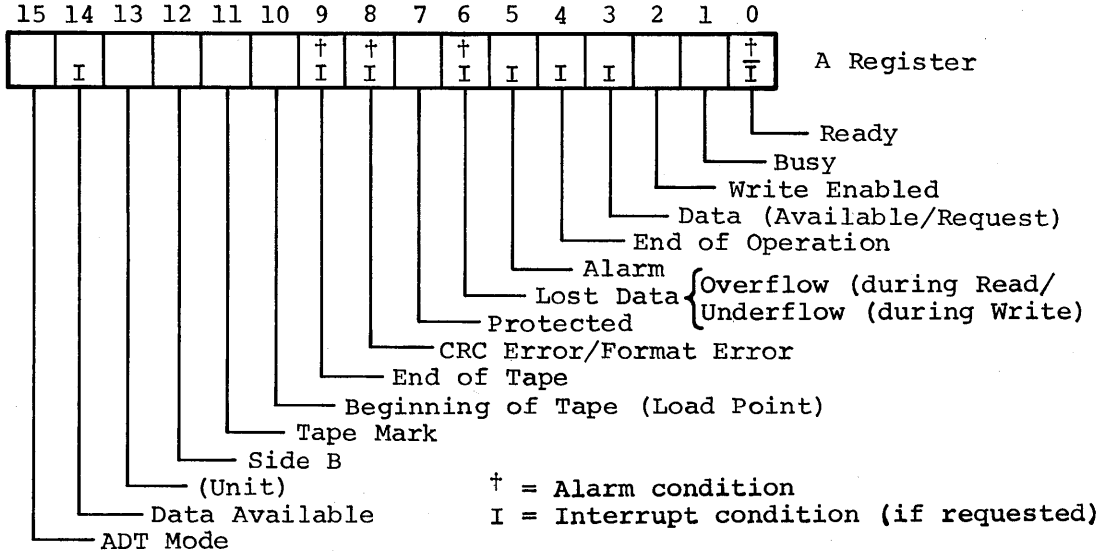


Figure 5-23. Status Responses

The following are brief descriptions of the 16 status responses provided by the tape cassette controller.

Ready

This bit indicates that the tape transport is operational (+5V applied, cassette loaded, lid closed). This condition must exist before the transport can move tape in either direction. If this bit is not on, operator intervention is usually required. Note that in Echo Mode this bit will be off (and an alarm present) while the test is in progress.

Busy

This bit indicates that the tape transport is busy during any motion cycle. Controller can accept motion commands only in Not Busy condition. Non-motion (NOP) commands or status requests are honored at all times. This bit clears automatically whenever the specified tape operation has been completed and all tape motion has ceased.

Write Enabled

This bit indicates the presence or absence of the Write Plug at the cassette case and is defined as follows:

A02=1, Plug is present - Write operations permitted

A02=0, Plug is missing - Write operations NOT permitted.

Data (Available/Request)

This bit indicates that the controller has data available from a read or is requesting data for a write (either from a true read or write operation or from an echo mode operation). If data interrupts are enabled, setting of this status will cause an interrupt. Normally this status is reset (and, therefore, the interrupt cleared) when data is taken or provided by the CP. It is also cleared by Master Clear and the Clear Controller command. The Clear Interrupt command does NOT clear this status bit.

End of Operation

This bit indicates that the cassette controller has completed a tape motion cycle and all tape motion has ceased. If end of operation interrupt is enabled, setting of this status will cause an interrupt. It is normally cleared when a new motion command is received. It is also cleared by Master Clear and the Clear Controller command. It is NOT cleared by the Clear Interrupt command. Note that during tape runaway situations this bit sets whenever tape motion ceases; but in this case it does not mean the tape motion command that led to the tape runaway has completed the cycle normally.

Alarm

This bit indicates that one (or more) of the alarm conditions are active. These are Not Ready, Lost Data (Overflow during Read/Underflow during Write), CRC Error, Format Error, and End of Tape. If alarm interrupts have been enabled, occurrence of this bit will cause an interrupt. It is cleared whenever the conditions causing the alarm have been cleared.

Lost Data

OVERFLOW (DURING READ)

This bit indicates that a data character was available to the MP as part of a read operation, but was not accepted prior to the availability of the succeeding character; part of the data block was therefore lost. If alarm interrupts are enabled, setting of this bit will cause an interrupt.

UNDERFLOW (DURING WRITE)

This bit can also indicate that a data underflow condition developed during one of the following write operations:

A/Q Mode - When a data word was not furnished by the MP (data underrun), the Record Termination (shutdown) sequence was initiated, and a Write Data from the MP occurs during the shutdown sequence. (The presence of the "late" Write Data distinguishes host Data from the normal end of data block condition.)

ADT Mode - When, due to a lack of a data word (data underrun), the Record Termination (shutdown) sequence was initiated but the MP ADT logic had not signalled end of the data block.

Whenever a Tape Mark is accidentally written by starting to write a record but no data is provided in sufficient time to become the first character after the preamble (approximately 120 milliseconds).

Protected

This bit indicates that the Program Protect switch is in the Protected position.

Cyclic Redundancy Checksum (CRC) Error/Format Error

This bit indicates that a CRC error has occurred during either a data read operation, or during the read-after-write portion of a write operation, or that a Format error (unexpected bit sequence) has occurred during a data read operation. If alarm interrupts are enabled, setting of this bit causes an interrupt.

End of Tape

This bit indicates that the EOT warning hole was sensed during a forward motion command (Write Motion, Read Motion, Write Tape Mark, Search Tape Mark, Erase). The EOT hole (like the BOT hole) is located between the two recording tracks and thus sensing of the EOT in no way invalidates the operation currently underway.

Beginning of Tape (Load Point)

This bit indicates that the cassette tape is currently at the BOT point. As noted under the discussion of the rewind motion command this term is logically derived since the transport light detector circuitry is unable to distinguish the BOT hole, the EOT hole and the transparent leader and trailer. Note that Clear Controller/Master Clear clears this status bit.

Tape Mark

This bit indicates that a tape mark has been detected either as part of a read operation, read-after-write portion of a write operation, or a Search Tape Mark command. Tape Mark is defined as a tape block (record) consisting of preamble, no data, CRC (all zeros), and a postamble. Tape mark status is cleared upon receipt of any new motion command. It is also cleared by Master Clear and the Clear Controller command.

Side B

This bit indicates which track of the cassette tape is in position under the Read/Write head (the cassette has two sides, A and B with either A or B clearly visible on the label). A sensor in the transport detects the presence/absence of an offset slot in the cassette, used to define sides A/B of the cassette.

Unit 1

This bit indicates that transport unit one is selected. It is always ON in the 2550-2 HCP.

Data Available

This bit indicates that the controller has data available, either from a read operation or from Echo Mode

operation. If data interrupts are enabled, setting of this status will cause an interrupt. This status is reset (and the interrupt cleared) when data is taken by the MP. It is also cleared by Master Clear and the Clear Controller command. This bit is used to resolve ambiguities of the data (available/request) bit during Echo Mode operations.

Auto Data Transfer Mode

This bit indicates that the controller is operating in the ADT mode. It is set whenever a function code is received in which function bit A06 (ADT) is set. It is normally cleared whenever a Terminate signal is received from the MP during the normal termination of an ADT operation. Master Clear and Clear Controller also clear this bit.

PERIPHERAL CONTROLLER

The peripheral controller is programmed to control operation of the 2572-1 or -2 Card Reader and/or the 2570-1 or -2 Line Printer as described in the following paragraphs.

CARD READER CONTROLLER OPERATING CHARACTERISTICS

Addressing

The E portion of the Q register (Q07-Q10) defines the card reader equipment code. This code is set up in the controller by four equipment select jumpers. Standard assignment for the card reader controller is E='B'. The W portion of the Q register (Q11-Q15) must be all zeroes.

Bits 0 and 1 (Q00 and Q01) of the Q register define a command code which, when accompanied by an equipment code and either a Read or Write signal, defines the operation to be performed by the card reader (see table 5-8). Only Q00 and Q01 are sampled in the check for a command code; Q02 through Q05 are ignored. Q06 must be zero. All illegal commands will be rejected (external reject) by the card reader controller.

TABLE 5-8. Q REGISTER COMMAND CODE

Q01	Q00	Read	Write
0	0	Data Transfer	Illegal
0	1	Director Status 1	Director Function
1	0	Illegal	Test Mode
1	1	Director Status 2	Illegal

Program Protect

When the card reader is operating in the protect mode, I/O instructions not having the protect bit set will cause a protect violation and will be rejected (external reject). Note that a director status request is never rejected regardless of the protect condition.

Director Functions

When the bits in the Q register define a director function, and they are accompanied by both an equipment number and an A/Q Write command, the director function specified by the bits in the A register will be performed. The director functions are defined in figure 5-24.

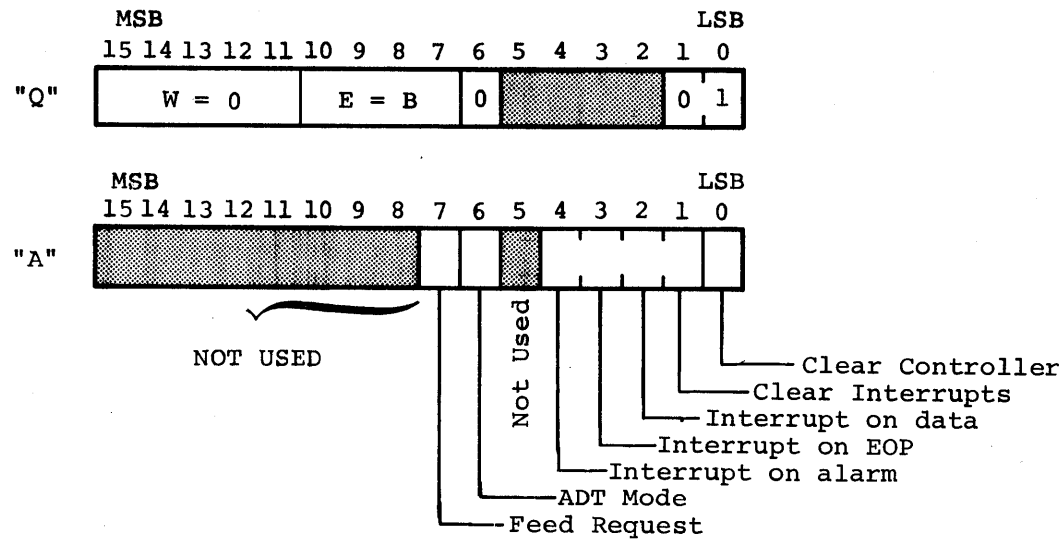


Figure 5-24. Card Reader Director Functions

A00=1, CLEAR CONTROLLER

This command clears all control, interrupt requests, interrupt responses and alarm conditions. It is subordinate to all other function bits.

A01-1, CLEAR INTERRUPTS

This command clears all interrupt requests and interrupt responses. It is subordinate to all interrupt request bits (A02, A03, A04), so it cannot prevent the setting of new interrupts.

A02=1, INTERRUPT ON DATA

This command enables an interrupt when the controller has data ready to be input by the CP. The interrupt request and response is cleared by a master clear or by either A00=1 or A01=1 with A02=0. When the response is cleared by A00=1 or A01=1, the request may be re-enabled in one operation if A02=1.

Before a data transfer from the card reader (either before or after a feed command), this interrupt may be requested and the response will signal the CP that reader data is ready. Without re-enabling or clearing this interrupt response, the data transfer can take place. During the data transfer, the interrupt response will be removed until the card reader again has data ready.

A03=1, INTERRUPT ON END OF OPERATION (EOP)

This command enables an interrupt upon completion of an operation. It is also used to notify the CP that some condition existed at the end of the last data transfer which will prevent any further data transfers. An interrupt response will not occur for an operation which was completed before the selection was made. The interrupt request and response may be cleared by a Master Clear or by either A00=1 or A01=1 with A03=0. When the response is cleared by A00=1 or A01=1, the request may be re-enabled in one operation if A03=1.

Note that a reader operation is defined as the time between acceptance of a feed command and the completion of a card (81st column time) or the time lost data occurred.

A04=1, INTERRUPT ON ALARM

This command enables an interrupt when an alarm condition exists. An alarm condition that exists at the time this interrupt request is made will immediately provide a response. If the alarm condition does not exist at the time of the interrupt request, the interrupt response will be provided as soon as an alarm condition is detected. The interrupt request and response may be cleared by a Master Clear or by either A00=1 or A A01=1 with A04=0. When the response is cleared by A00=1 or A01=1, the request may be re-enabled in one operation if A04=1.

A06=1, ADT MODE

This command directs the card reader controller to operate in the auto data transfer (ADT) mode. ADT mode operation may be cleared by a Master Clear or by a director function with A06=0.

A07=1, FEED REQUEST

This command directs the card reader to initiate a one-card feed cycle.

If a second feed request is issued before EOP is sensed for the first read cycle, data transfer is stopped and a new card is fed to the read station. No EOP condition is generated for the first card.

Note that the director functions may be stacked; that is, two or more functions may be issued at the same time.

The controller will accept and execute a clear controller function if it is not ready, provided that no other director bit, except A01, is selected at the same time (bits A02, A03, A04, A06 and A07 must be zero). Care should be exercised in using the clear controller function while the card reader is busy.

Director Status 1

When the bits in the Q register define director status 1, and they are accompanied by both an equipment number and an A/Q Read command, the card reader status will be transferred to the CP "A" register. The controller always replies to a status 1 request. The status responses are described in figure 3-25.

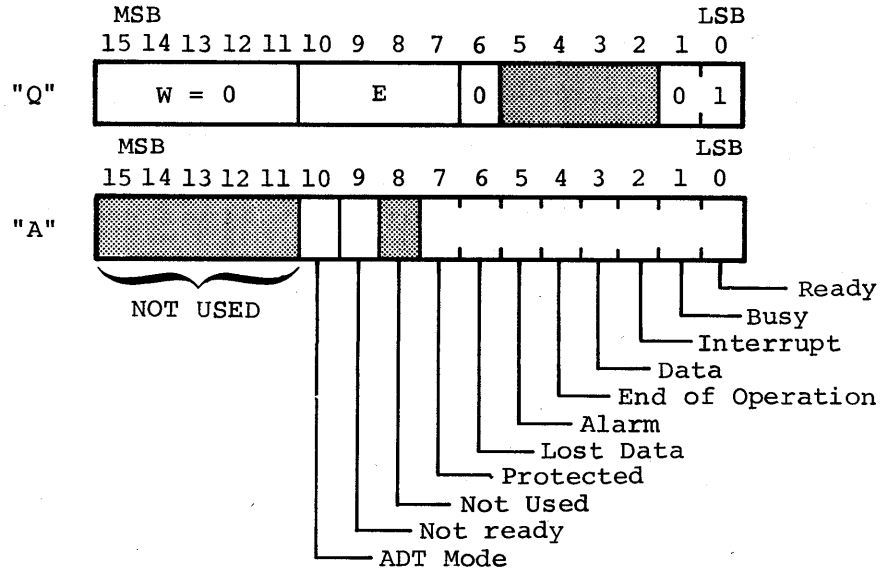


Figure 5-25. Director Status 1 Responses

A00=1, READY

This bit indicates that the card reader is operational. The ready condition must exist before the reader can operate. The requirements are as follows:

1. All power supplies are operating
2. The input hopper is loaded
3. The output stacker is not full
4. There is no card jam condition
5. There is no read alert condition

A01=1, BUSY

This bit indicates that the card reader is busy during a read cycle that was initiated and is not yet completed.

A02=1, INTERRUPT

This bit indicates that an interrupt response (macro) has been generated by the card reader. Status bits A03, A04 and A05 define the interrupt which occurred.

A03=1, DATA

This bit indicates that the controller data buffer contains information ready for transfer to the CP. If interrupt on data has been selected, this status also indicates this interrupt has occurred. The status and interrupt will drop upon completion of the data transfer.

A04=1, END OF OPERATION (EOP)

This bit indicates that the card reader has completed a read cycle (81st column time), or that an alarm condition existed at the end of the last data transfer which prevents any further data transfer. If EOP interrupt has been selected, this status also indicates this interrupt has occurred. This status is cleared when the next read cycle is initiated, or by Master Clear or a Clear Controller Director Function.

A05=1, ALARM

This bit indicates that an alarm condition exists. The alarm may be either a lost data condition (status bit A06=1) or a card reader malfunction. This status bit is cleared by a Master Clear, Clear Controller Director Function, or a Feed Director Function provided that the cause for the alarm has been corrected.

A06=1, LOST DATA

This bit indicates that data was not transferred to the CP before the next column of data was ready for input. All subsequent columns of data on this card will be lost since the controller will reject any further data transfer commands until the end of operation (EOP) status comes true. The lost data status is cleared by a master clear, clear controller director function, or a feed director function.

A07=1, PROTECTED

This bit indicates that the controller protect jumper is in (Protect switch in "Protected" position), and it is therefore operating in a protected mode.

A09=1, NOT READY

This bit is always the logical complement of the Ready status bit (A00).

A10=1, ADT MODE

This bit indicates that the controller is currently operating in the ADT mode.

Director Status 2

When the bits in the Q register define director status 2, and they are accompanied by both an equipment number and an A/Q Read command, the

card reader error status will be transferred to the CP A register. The controller always replies to a status 2 request. The status responses are described in figure 5-26.

A00=1, HOPPER EMPTY

This bit indicates that input hopper is empty and last card has been read.

A01=1, STACKER FULL

This bit indicates that the output stacker is full and that the last card fed has been read.

A02=1, FAIL TO FEED

This bit indicates that the reader was not able to feed the next card from the deck after two feed attempts.

A05=1, STACKER AREA JAM

This bit indicates a jam in the card path from the read station to the stacker.

Data Transfers

When the bits in the Q register define the data transfer command, and they are accompanied by an equipment number and an A/Q Read command, then data from the last card column will be transferred to bits 0-11 of the computer A register. See figure 5-27.

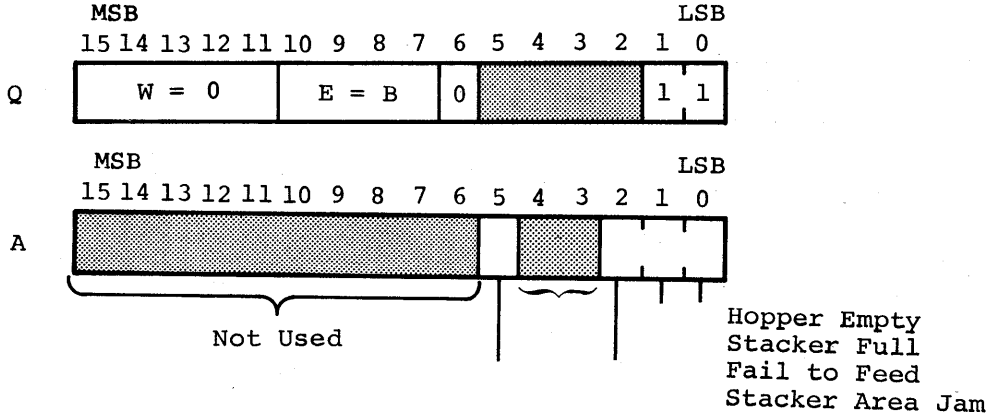


Figure 5-26. Director Status 2 Responses

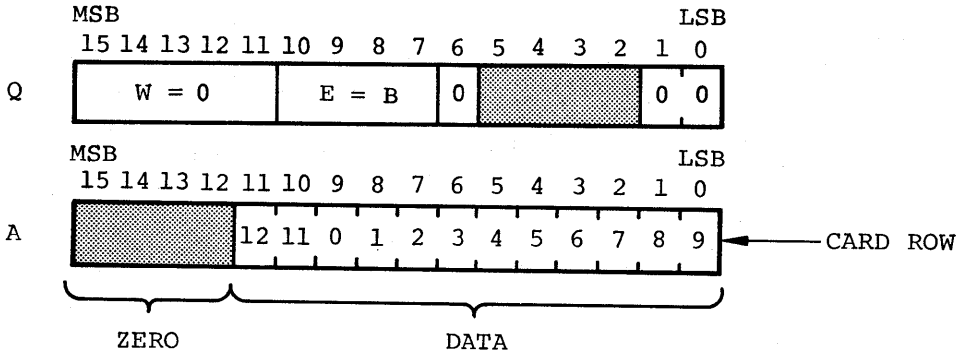


Figure 5-27. Data Transfer Command

The numbers in the A register bit positions indicate the card row in which a punched hole was detected.

Data transfer commands will be accepted only when the following conditions are true:

1. The card reader is ready
2. Data status (bit A03) is true
3. There is no program protect violation
4. Alarm status (bit A05) is false

The data transfer clears the data status and the data interrupt response if it was selected.

TEST MODE

When the bits in the Q register define test mode, and they are accompanied by both an equipment number and an A/Q Write command, if A00=1, both the card reader controller and the printer controller will go into a self test mode. This command is accepted only if both controllers are not busy. (See figure 5-28.)

Test mode operation may be cleared by either a master clear, or a test mode function with bit A00=0.

Test mode provides internal switching of data and control signals in such a manner that each controller may be used to test the other. With proper programming, all I/O signals and most internal controller functions and status may be tested independent of the end peripheral devices.

Note that although test mode is selected by addressing the card reader controller, this mode puts both the line printer and card reader controllers into a test configuration. One controller cannot be tested without the other in this mode.

INTERRUPTS

There are five interrupts available from the card reader controller.

1. Data Interrupt
2. EOP Interrupt
3. Alarm Interrupt
4. Common Interrupt
5. ADT Interrupt

The common interrupt is active whenever any of the data, EOP or alarm interrupts are active (logical OR).

The ADT interrupt is active only when the card reader controller is operating in the ADT mode and the data interrupt condition is true and the EOP and alarm interrupts are false. The ADT interrupt is cleared by a Master Clear, a Clear Interrupt or Clear Controller Director Function or a Data Transfer.

The five card reader interrupts are brought to an interrupt selection matrix where any two of the five may be jumpered to either of two CPU interrupt lines. Standard interrupt

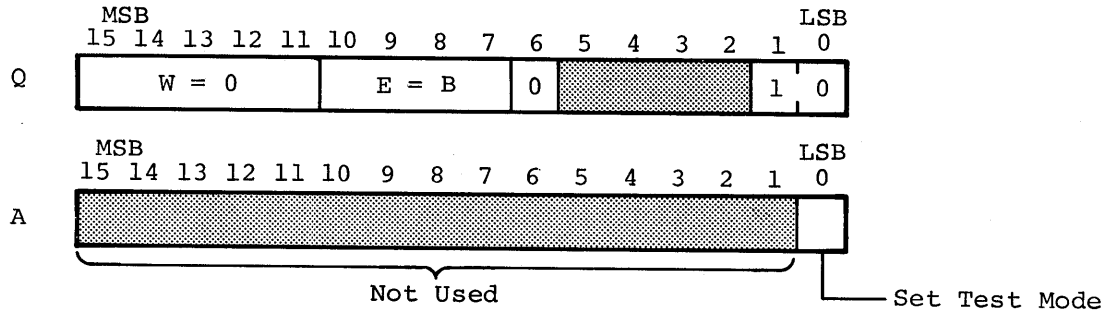


Figure 5-28. Test Mode Command

assignment has the macro interrupt line driven by the common interrupt and the micro interrupt driven by the ADT interrupt.

LINE PRINTER CONTROLLER OPERATING CHARACTERISTICS

Refer to table 5-9 for printer options.

Addressing

The E portion of the Q register (Q07-Q10) defines the printer equipment code. This code is set up by the four equipment select jumpers. Standard assignment for the line printer controller is E='4'. The W portion of the Q register (Q11-Q15) must be all zeros.

Bit 0 (Q00) of the Q register defines a command code which, when accompanied by an equipment code and either a Read or Write signal, defines the operation to be performed by the printer. Only Q00 is sampled in the check for a command code; Q01 through Q05 are ignored, and Q06 must be zero.

Q00	Read	Write
0	Illegal	Data Transfer
1	Director Status	Director Function

The illegal command code will be rejected (external reject) by the printer controller.

Program Protect

When the printer is operating in the protect mode, I/O instructions not having the protect bit set will cause a protect violation and will be rejected (external reject). Note that a director status request is never rejected regardless of the protect condition.

Director Function

When Q00=1, and it is accompanied by both an equipment code and an A/Q Write command, the director function specified by the bits in the A register will be performed. The director functions are described in figure 5-29.

A00=1, CLEAR PRINTER

This command clears all control, interrupt requests, interrupt responses and alarm conditions. It is subordinate to all other commands except the print directive. The printer must be Ready and Not Busy to execute this function.

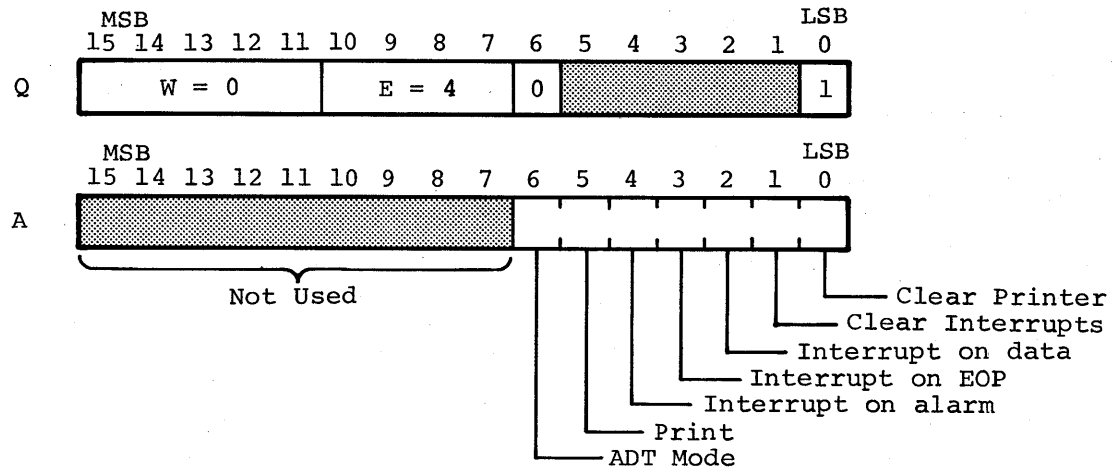


Figure 5-29. Line Printer Director Functions

TABLE 5-9. PRINTER OPTIONS

OPTION		2570-1	2570-2
Differential Transmitter/Receivers		Required	Required
Character Sets	⚠1	Optional	Optional
Line Counter Options	⚠1	Optional	Optional
VFU Options	⚠1	Optional	Optional
Pre-print Paper Motion		Required	Required
Optional Data Interchange		No	No
Transmission Parity Option		Optional	Optional
Number of Data Bits:	7	Optional	Optional
	8	N/A	Optional
	9	N/A	No
Paper Out Status		Optional	Optional
Error Status		Optional	Optional
Buffer Overflow Status		Optional	Optional
Data Load Sequence		No	No

DEFINITIONS:

Required - This option is required for correct operation of the printer with the 2571-1 controller.
Optional - This option may or may not be included.
No - This option must not be included for correct operation of the printer with the controller
N/A - This option is not available with this printer.
⚠1 - Some of these options are not supported by standard software.

A01=1, CLEAR INTERRUPTS

This command clears all interrupt requests and interrupt responses. It is subordinate to all interrupt request bits (A02, A03, A04) so it cannot prevent the setting of new interrupts.

A02=1, INTERRUPT ON DATA

This command enables an interrupt when the printer can receive data. The interrupt request and response is cleared by a Master Clear or by either A00=1 or A01=1 with A02=0. When the response is cleared by A00=1 or A01=1, the request may be re-enabled in one operation if A02=1.

Before a data transfer to the printer, this interrupt may be requested and the response will signal the CP that the printer is ready to receive data. Without re-enabling or clearing this interrupt response, the data transfer can take place or

a print command may be issued. During the data transfer or print cycle, the interrupt response will be removed until the printer is again ready to receive data.

A03=1, INTERRUPT ON END OF OPERATION (EOP)

This command enables an interrupt upon completion of an operation. The interrupt may be selected before or during an operation. An interrupt response will not occur for an operation which has completed before the selection was made. The interrupt request and response may be cleared by a Master Clear or by either A00=1 or A01=1 with A03=0. When the response is cleared by A00=1 or A01=1, the request may be re-enabled in one operation if A03=1.

A04=1, INTERRUPT ALARM

This command enables an interrupt when an alarm condition exists. An alarm condition that exists at the time this interrupt request is made will immediately provide a response. If the alarm condition does not exist at the time of the interrupt request, the interrupt response will be provided as soon as an alarm condition is detected. The interrupt request and response may be cleared by a Master Clear or by either A00=1 or A01=1, the request may be re-enabled in one operation if A04=1.

A05=1, PRINT

This command directs the printer to initiate a print cycle. A print cycle is the time between acceptance of a print command and completion of a line of print. During a print cycle, the data status (A03) will be 0.

A06=1, ADT MODE

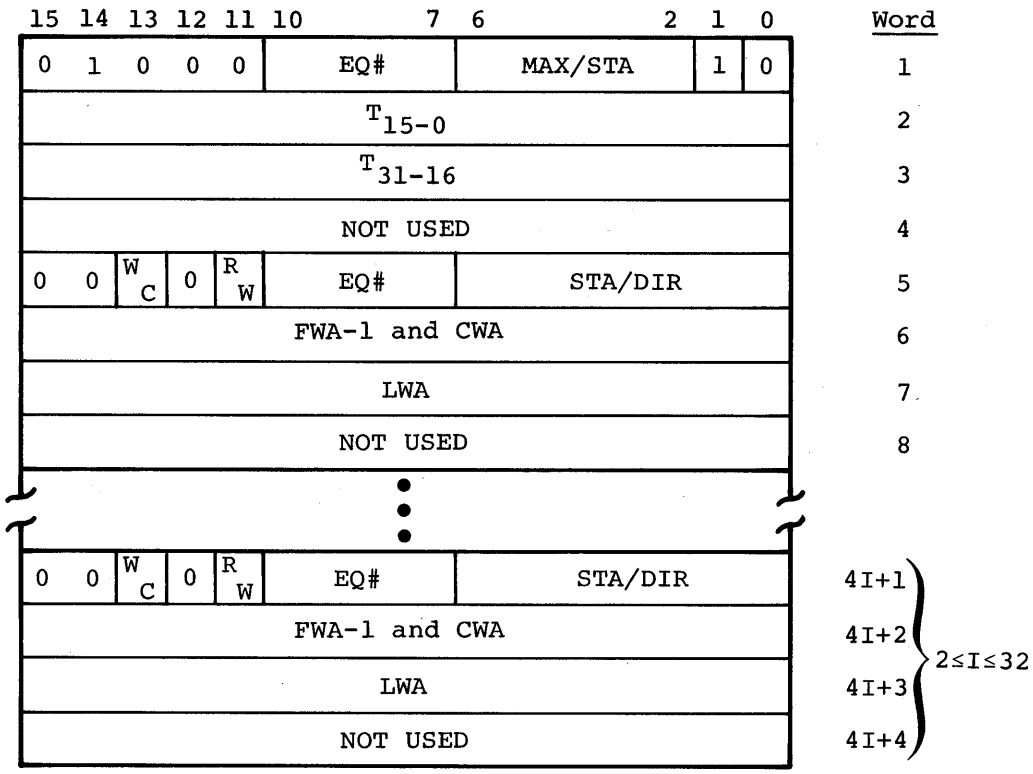
This command directs the printer controller to operate in the auto data transfer mode (figure 5-30). ADT mode operation may be cleared by a Master Clear or by a director function with A06=0.

SINGLE AQ DEVICE

15	13	12	11	10 7	6 0	Word
0 0	W C	0	R W	EQ#	STA/DIR	1
FWA-1 and CWA						2
LWA						3
NOT USED						4

Figure 5-30. Auto-Data Transfer Mode Format

MULTIPLE AQ DEVICES



ADT TABLE FOR CLOCK

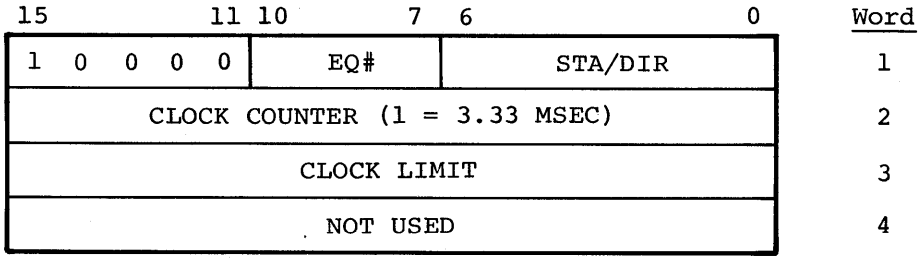


Figure 5-30. Auto-Data Transfer Mode Format (Contd)

SINGLE OR MULTIPLE SET/SAMPLE DEVICE

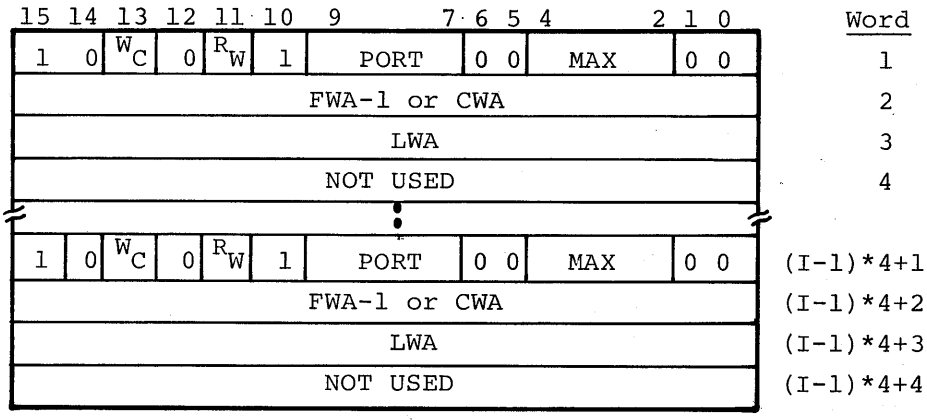


Figure 5-30. Auto-Data Transfer Mode Format (Contd)

Note that director functions may be stacked; that is, two or more functions may be issued at the same time. It should be noted that when A00=1 and/or A01=1 are used with stacking, clear function will take place first, followed by any stacked functions.

A clear controller function requires 5 microseconds, so it should not be immediately followed by a Data Transfer or Print Director Function.

Director Status

When Q00=1, and it is accompanied by both an equipment code and an A/Q read command, the printer status will be transferred to the CP A register. The controller always replies to a status request. The status responses are described in figure 5-31.

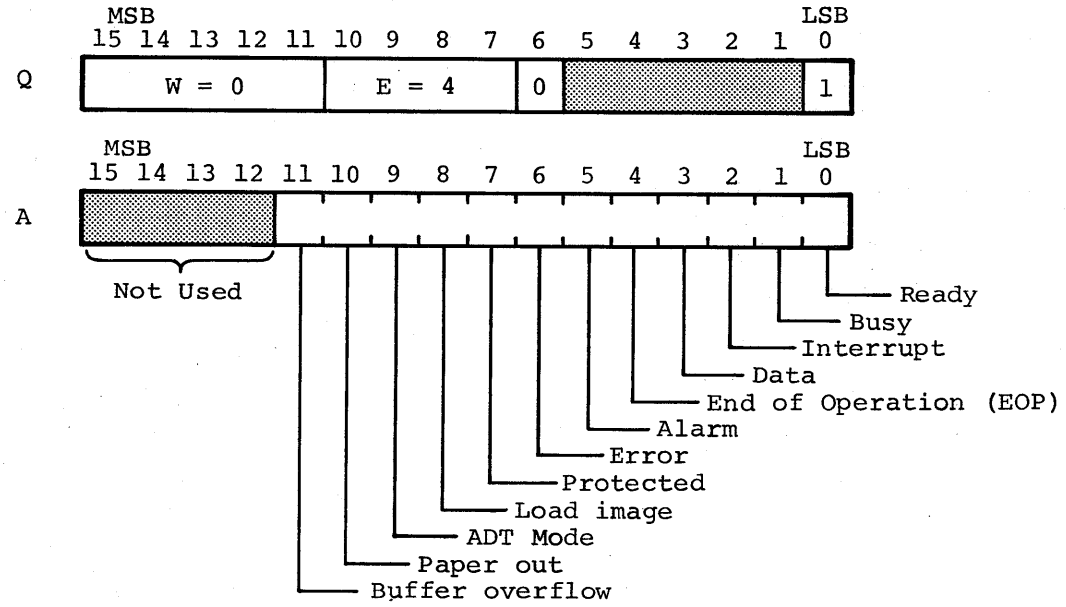


Figure 5-31. Printer Status Responses Format

A00=1, READY

This bit indicates that the printer is operational and on-line. The ready condition must exist before the printer can operate. The requirements are as follows:

1. All power supplies are operating
2. Hammer fuses are intact
3. Paper is present
4. The drum gate or train array gate is latched
5. No paper motion fault exists
6. Image memory is loaded or the load image indicator is illuminated (2570-2 only)

A01=1, BUSY

This bit indicates that the printer is busy during a print cycle or paper motion operation.

A02=1, INTERRUPT

This bit indicates that an interrupt response has been generated by the printer. Status bits A03, A04 and A05 define the interrupt which occurred.

A03=1, DATA

This bit indicates that the printer is ready to receive data. If interrupt on data has been selected, this status also indicates this interrupt has occurred.

A04=1, END OF OPERATION (EOP)

This bit indicates that the printer has completed an operation. If EOP interrupt has been selected, this status also indicates this interrupt has occurred. The EOP status bit is cleared by a Master Clear, Clear Printer Director Function, Data Transfer or a Print Director Function.

A05=1, ALARM

This bit indicates that an alarm condition exists. The alarm may be either an error (status bit A06=1) or a printer malfunction. This status bit is cleared by a Master Clear, Clear Printer Director Function, Data Transfer or a Print Director Function provided that the cause for the alarm has been corrected.

A06=1, ERROR

2570-1 Printer - This bit indicates that a data transfer parity error occurred. The erroneous character will be printed as a blank if it was a data character, or ignored if it was a control character. The printer will remain ready when this condition exists. This status bit is cleared by a Master Clear, Clear Printer Director Function, Data Transfer or a Print Director Function.

2570-2 Printer - This bit indicates that any combination of Parity Error, synchronization error or compare error occurred. The printer will remain ready when any of these conditions exist. This status bit is cleared by a Master Clear, Clear Printer Director Function, Data Transfer or a Print Director Function.

A07=1, PROTECTED

This bit indicates that the controller protect jumper is installed (Protect switch in Protected Position) and it is therefore operating in a protected mode.

A08=1, LOAD IMAGE (2570-2 PRINTER ONLY)

This bit indicates that the start pushbutton has been depressed, if no fault conditions exist, after the fill image pushbutton has been depressed on the printer. This

status indicates that the printer will place the next 288 data characters in the image memory. If a parity error occurs, no further transmission will take place until a Master Clear or Clear Printer Director Function is issued. This status bit is cleared upon successful transfer of the 288th data character.

A09=1, ADT MODE

This bit indicates that the controller is currently operating in the ADT mode.

A10=1, PAPER OUT

If the printer is equipped with the paper-out status option, then this bit indicates that an out-of-paper condition has been detected at the lower tractor switches. Approximately three inches of paper normally remains between the bottom edge of the last form and the print station. When this condition occurs, the printer will become not ready.

ALL=1, BUFFER OVERFLOW

If the printer is equipped with the buffer overflow status option, then this bit indicates that more than the maximum number of characters (per line) have been received. This status will become true during the transfer of the first extra data byte. This status bit is cleared by a Master Clear, a Clear Printer Director Function, or a Print Director Function. All overflow characters will be responded to, but not stored or printed.

Data Transfers

When Q00=0, and it is accompanied by both an equipment code and an A/Q Write command, then bits 0-7 of the **A** register will be transmitted to the printer as data (see figure 5-32). The first character of a line is identified as a control character and will not be printed.

Data transfer commands will be accepted only when the following conditions are true:

1. The printer is ready
2. The printer is not busy
3. Data status (bit A03) is true
4. There is no program protect violation

Control Characters

The first character in each line is defined as a control character for vertical paper motion and will not be printed, except in the case of loading the image buffer. The printer must be equipped with the pre-print paper motion option. This option causes the paper motion cycle

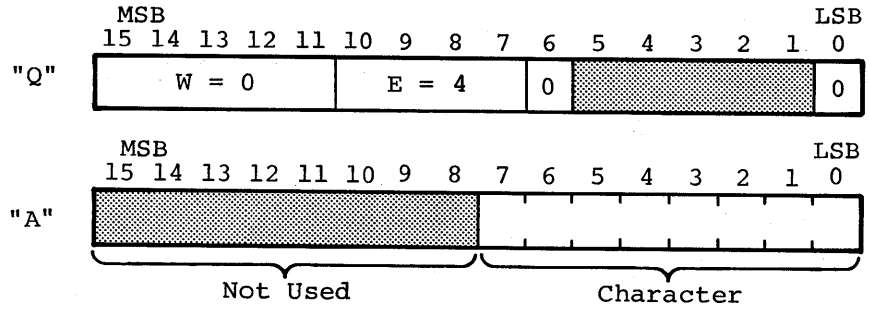


Figure 5-32. Printer Data Transfer Command Format

to occur before the print cycle. Paper motion is initiated upon reception of the format command (first transmitted character), and the printer does not set the busy status. The remainder of the line is then loaded while the paper advance cycle is in process and the print cycle occurs after the advance cycle is complete.

Note that if a control character is issued which performs a vertical paper motion and a Clear Printer Director Function is issued immediately thereafter, the busy status will be set until paper motion stops.

The control characters are listed below:

Spacing control characters for the Line Printer Controller are shown in figure 5-33.

INTERRUPTS

There are five interrupts available from the line printer controller:

1. Data Interrupt
2. EOP Interrupt
3. Alarm Interrupt
4. Common Interrupt
5. ADT Interrupt

15 8	7 6 5 4	3 2 1 0	
Not used			
	X X 0 X	X X 0 0	Suppress spacing
	X X 0 X	X X 0 1	Single space before printing
	X X 0 X	X X 1 0	Double space before printing
	X X 0 X	X X 1 1	Triple space before printing
	X X 1 X	0 0 0 0	Move to VFC Channel 1 (TOF) before printing
	X X 1 X	0 0 0 1	Move to VFC Channel 2 (BOF) before printing
	X X 1 X	0 0 1 0	Move to VFC Channel 3 before printing
	X X 1 X	0 0 1 1	Move to VFC Channel 4 before printing
	X X 1 X	0 1 0 0	Move to VFC Channel 5 before printing
	X X 1 X	0 1 0 1	Move to VFC Channel 6 before printing
	X X 1 X	0 1 1 0	Move to VFC Channel 7 before printing
	X X 1 X	0 1 1 1	Move to VFC Channel 8 before printing
	X X 1 X	1 0 0 0	Move to VFC Channel 9 before printing
	X X 1 X	1 0 0 1	Move to VFC Channel 10 before printing
	X X 1 X	1 0 1 0	Move to VFC Channel 11 before printing
	X X 1 X	1 0 1 1	Move to VFC Channel 12 before printing
	X X 1 X	1 1 0 0	Illegal as Vertical Format Control (VFC) character (decoded as if bit 5=0)
	X X 1 X	1 1 0 1	
	X X 1 X	1 1 1 0	
	X X 1 X	1 1 1 1	

(X = don't care)

Figure 5-33. Line Printer Controller Spacing Control Characters

The common interrupt is active whenever any of the data, EOP, or alarm interrupts are active (logical OR).

The ADT interrupt is active only when the printer controller is operating in the ADT mode and the data interrupt conditions are true and the EOP and alarm interrupts are false. The ADT interrupt is cleared by a Master Clear, a Clear Interrupt or Clear Controller Director Function, a Data Transfer or a Print Director Function.

The five line printer interrupts are brought to an interrupt selection matrix where any two of the five may be jumpered to either of two CPU interrupt lines. Standard interrupt assignment has the macrointerrupt line driven by the common interrupt and the microinterrupt line driven by the ADT interrupt.

MULTIPLEX SUBSYSTEM

The Loop Multiplexer (LM) and associated communications line adapters (CLAs) are under complete control of the Multiplex Loop Interface Adapter (MLIA). The format of the loop cells utilized to communicate between the MLIA and the LM is shown in figure 5-34.

LOOP CELL ARRANGEMENT

The loop cells are arranged in line frames on both the input and output loop. Each frame must start with an address loop cell and end with a CRCS loop cell. Each frame must not contain more than 16 loop cells, including address and CRCS cells. Empty cells must not appear within line frames, but may appear between frames. Null cells may appear within or between frames.

LOOP BATCHES

Line frames are always grouped in loop batches. Each loop batch must be followed by a loop end cell. Loop end cells to be utilized on the input loop, for response by the Loop Multiplexer, must be followed immediately by at least one empty cell. A loop batch may contain any number of line frames but is limited by the number of empties. A typical loop batch is shown in figure 5-35.

CLA

Each CLA, connected to LMs which interface a given loop, must have an address different from the address of any other CLA in that loop. If the LM is connected to two loops for redundant operation, each CLA in both loops must have an address different from the address of any other CLA in both loops.

LOOP MULTIPLEXER

The LM and all CLAs are reset and cleared when clock signals are not received for 20 microseconds on both input and output loops. The input section of the LM is reset and cleared when the clock is not received on the input loop for 20 microseconds; and the output section of the LM is reset and cleared when the clock is not received for 20 microseconds on the output loop. When two LM cards are present in a redundant configuration, the CLAs will be reset when the clock is absent for 20 microseconds on both loops. With a bit rate of 20 MHz, generation of approximately 34 loop cells requires a period in excess of 20 microseconds.

Input Loop

1. When the MLIA can accept data/supervision from the CLAs in the loop, it initiates an input loop batch by generating a loop end followed by a block of null cells (to mark the end of the available loop cells). If the MLIA can accept more than one frame, it places several empty cells on the loop. If it cannot

CELL MARKER
CELL IDENTIFICATION
CELL INFORMATION

TYPE	FORMAT NUMBER	M	ID			INFORMATION								CELL DESIGNATION
		B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	
LOOP MANAGEMENT FORMAT	0	1	0	0	0	0	0	0	0	0	0	0	0	EMPTY
	0	1	0	0	0	1	0	1	0	1	0	1	0	NULL
	1	1	0	0	1	F3	F2	1	1	1	0	0	0	LOOP END
	1	1	0	0	1	F3	F2	0	0	0	1	1	1	LOOP END RESTART
RESERVED FORMATS	2	1	0	1	0	R	R	R	R	R	R	R	R	UNDEFINED
	3	1	0	1	1	R	R	R	R	R	R	R	R	UNDEFINED
LOOP USAGE FORMATS	6/7	1	1	1	F1	A1	A2	A3	A4	A5	A6	A7	A8	CLA ADDRESS
	4	1	1	0	0	D1	D2	D3	D4	D5	D6	D7	D8	DATA
	5	1	1	0	1	S1	S2	S3	S4	S5	S6	S7	S8	SUPERVISION
	6	1	1	1	0	C1	C2	C3	C4	C5	C6	C7	C8	CRCS

INPUT LOOP CODING

F1 = 0 NO OUTPUT DATA DEMAND
F1 = 1 OUTPUT DATA DEMAND
F2 = X UNDEFINED
F3 = 0 PRIMARY DECODES LOOP END; SECONDARY IGNORES LOOP END
F3 = 1 PRIMARY IGNORES LOOP END; SECONDARY DECODES LOOP END

OUTPUT LOOP CODING

F1 = 1
F2 = X UNDEFINED
F3 = X UNDEFINED

CODING ON BOTH LOOPS

A1 TO A8 = BINARY CODE OF CLA ADDRESS; A1 IS MOST SIGNIFICANT BIT
D = DATA
C = CYCLIC REDUNDANCY CHECK CHARACTER (CRCS), C1 IS MOST SIGNIFICANT BIT
R = X RESERVED AND UNDEFINED

Figure 5-34. Format of Loop Cells

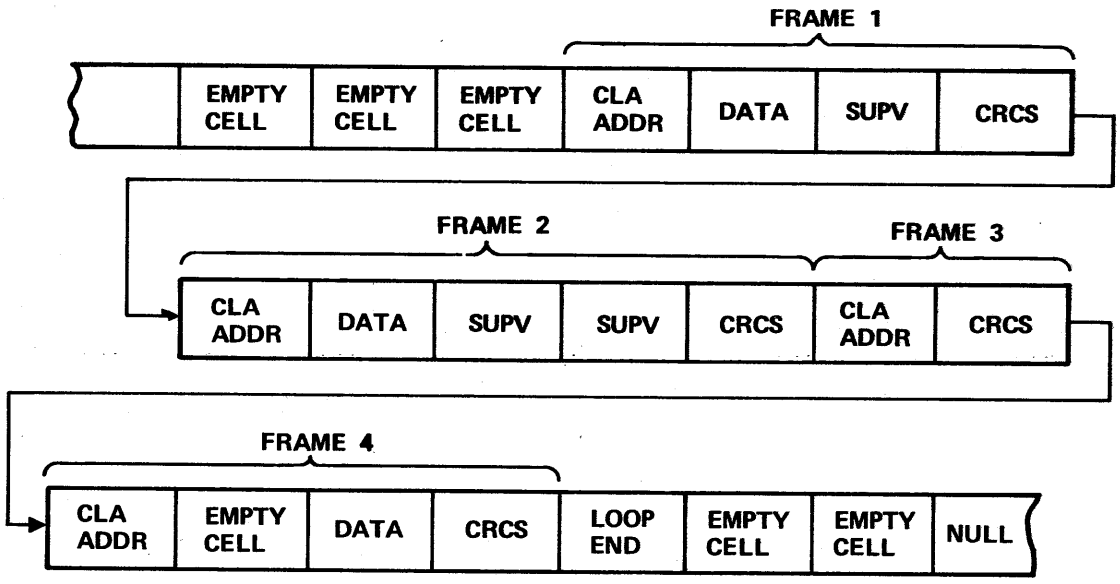


Figure 5-35. Input Loop Batch and Line Frame Structure

accept more data supervision from the loop, the adapter halts generation of the loop batches. The number of empties and nulls to be sent are programmable parameters.

2. When the LM receives the loop end and empty cell combination and has information for the loop, it deletes the loop end and replaces it with an empty cell. It will immediately place contiguous line frames on the loop in cell synchronization with the received empties. The frames will have a structure similar to those illustrated in figure 5-35. If there are no CLAs with information for the loop, the LM will not disturb the passing loop cells.

3. If a null cell is encountered by the LM immediately prior to placement of a CRCS character on the loop, it will place a loop end between the CRCS cell and the null cell.

4. When all CLAs requesting access have placed one frame on the loop, the LM replaces the next cell received with a loop end cell.

5. Information regarding the cells within each line frame, relating to each CLA, is contained within the manual for that CLA type (see Preface).

Input Loop Restart

1. When the MLIA wishes to inform the CLAs of an input loop batch error, it places a restart loop end followed by an empty cell on the loop. Each LM responds to the restart loop end by informing those CLAs that utilized the last loop batch of the loop error. CLAs using the last loop batch will then generate error status.

2. The LM will not utilize the restart loop end for placement of frames on the loop.

Output Loop

1. At least one empty cell must precede each loop batch to be passed on to the CLAs on the output loop. There must be at least one empty cell between each loop batch. The frames will always have the internal structure of figure 5-35.

2. The LM passes all output loop cells to the next LM or to the MLIA with a delay not to exceed two bit times.

3. Information regarding the cells within each loop frame, relating to each CLA, is contained within the manual for that CLA type (see Preface).

Redundant Operation

1. When two loops are to service a group of CLAs in a given LM cage, the LM card position and the presence or absence of a tag bit in the loop end determines which loop communicates with the CLAs.

2. An LM card in the primary card slot will process a loop end-empty cell combination if the secondary LM Flag (B4)=0.

3. A LM card in the secondary card slot will process loop end-empty cell combinations if the secondary LM Flag (B4)=1.

4. The MLIA, in conjunction with system hardware and software, will prevent both loops from accessing the CLAs in a single LM cage at the same time.

5. The input section of a given LM card determines the connection of the output section to the CLA

bus. If the input section is active in a redundant system, the output section will also be active. If the input section is disconnected from the CLA bus, the output section will also be disconnected from the CLA bus.

LOOP MULTIPLEXER TO CLA LOGIC SIGNAL DEFINITIONS (INPUT)

1. IAV1 to IAV32 - Input Available

   A single line from each CLA to the LM indicating:

   Logic 1 = CLA has input information
   Logic 0 = CLA has no input information

2. INS1 to INS32 - Input Select

   A single line from the LM to each CLA indicating:

   Logic 1 = CLA selected
   Logic 0 = CLA not selected

3. IF1, IF2, IF3 - Input Format

   Three bused lines from the CLAs to the LM identifying the contents of the information bits. Only the selected CLA has access to these lines. The line codes are:

IF1	IF2	IF3	Definition
0	0	0	Not used
0	0	1	Not used
0	1	0	Spare
0	1	1	Spare
1	0	0	Data present
1	0	1	Supervision present
1	1	0	Address without output data demand
1	1	1	Address with output data demand

4. II1 to II8 - Information Input

   Eight bused signals from the CLAs to the LM containing information characters. Only the selected CLA has access to the bus. Each bit is coded:

   Logic 1 = Information bit is 1
   Logic 0 = Information bit is 0

5. IEN - Input End

   A single bused signal from the CLAs to the LM indicating:

   Logic 1 = Selected CLA is presenting the last word to LM
   Logic 0 = Selected CLA is not presenting the last word to LM

6. IST - Input Strobe

   A single bused signal from the LM to the CLAs. The LM controls the information flow between the selected CLA and the LM via this signal. The LM has accepted the bused information on the logic 1 to logic 0 transition of the input strobe.

7. IER - Input Error

   A single bused signal from the LM to the CLAs. When the LM is informed of an input batch error, this line will be set to a logic 1. Each CLA that used the last batch will be selected and will receive a strobe signal. The CLAs will not place information on the bus, when selected, with IER at a logic 1.

8. MCL - Master Clear

   A single bused signal from the LM to all CLAs. When clock signals have not been received on the input and output loops for 20 microseconds, this line will be set to a logic 1. At all other times it will be set to a logic 0.

LOOP MULTIPLEXER TO CLA LOGIC SIGNAL DEFINITIONS (OUTPUT)

All output section signals are bused and directed from the LM to the CLAs.

OSL - Output Select

A logic 1 on this line informs all CLAs that a CLA address is presented on the information bus for examination. A logic 0 indicates CLA selection is not in progress.

OSC - Output Select Clear

A logic 1 on this line indicates that the LM has no further information for the selected CLA. A logic 0 on this line indicates that selection and information transfer may continue.

OF1, OF2, OF3 - Output Format

These three signals identify the contents of the information bits. The line codes are:

OF1	OF2	OF3	Definition
0	0	0	Not used
0	0	1	Not used
0	1	0	Spare
0	1	1	Spare
1	0	0	Data
1	0	1	Supervision
1	1	0	CRCS
1	1	1	Address

IO1 to IO8 - Information Output

These eight lines contain the data, commands, or other information required by the CLAs. A logic 1 on a line indicates the information bit is 1. A logic 0 on a line indicates the information bit is 0.

OST - Output Strobe

The LM controls the signal flow between the selected CLA and the LM via this signal. The CLA has accepted information from the bus on the logic 1 to logic 0 transition.

OER - Output Error

When a CRCS error is detected by the LM, this line is set to a logic 1 prior to CLA deselection. When no CRCS error is detected, this line remains at a logic 0.

MULTIPLEX LOOP INTERFACE ADAPTER

The MLIA maintains a logical interface with both input and output Mux Loops, a logical interface with both IDC and DMA channels of the CP, plus the physical interface of each of the MLIA cards to the CP and the other MLIA cards.

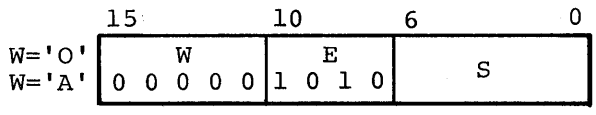
CHANNEL ADDRESSING (Q REGISTER)

The MLIA is addressed on the CP in MO5 (NCR) address format.

Since there are no devices, station, or other subdivisions within the MLIA, six bits are available to distinguish between commands uniquely. Thus the contents of the address (Q) register alone define the operation. The data (A) register is only required when parameters are being transferred. The need to load only one register instead of two results in greater efficiency and permits greater throughput with no loss of versatility. (Figure 5-36.)

CDC 1700-AQ CHANNEL

CDC 1700 addressing divides the addressing into three major fields:



Since the MLIA is not operated on the buffer data channel, the W parameter

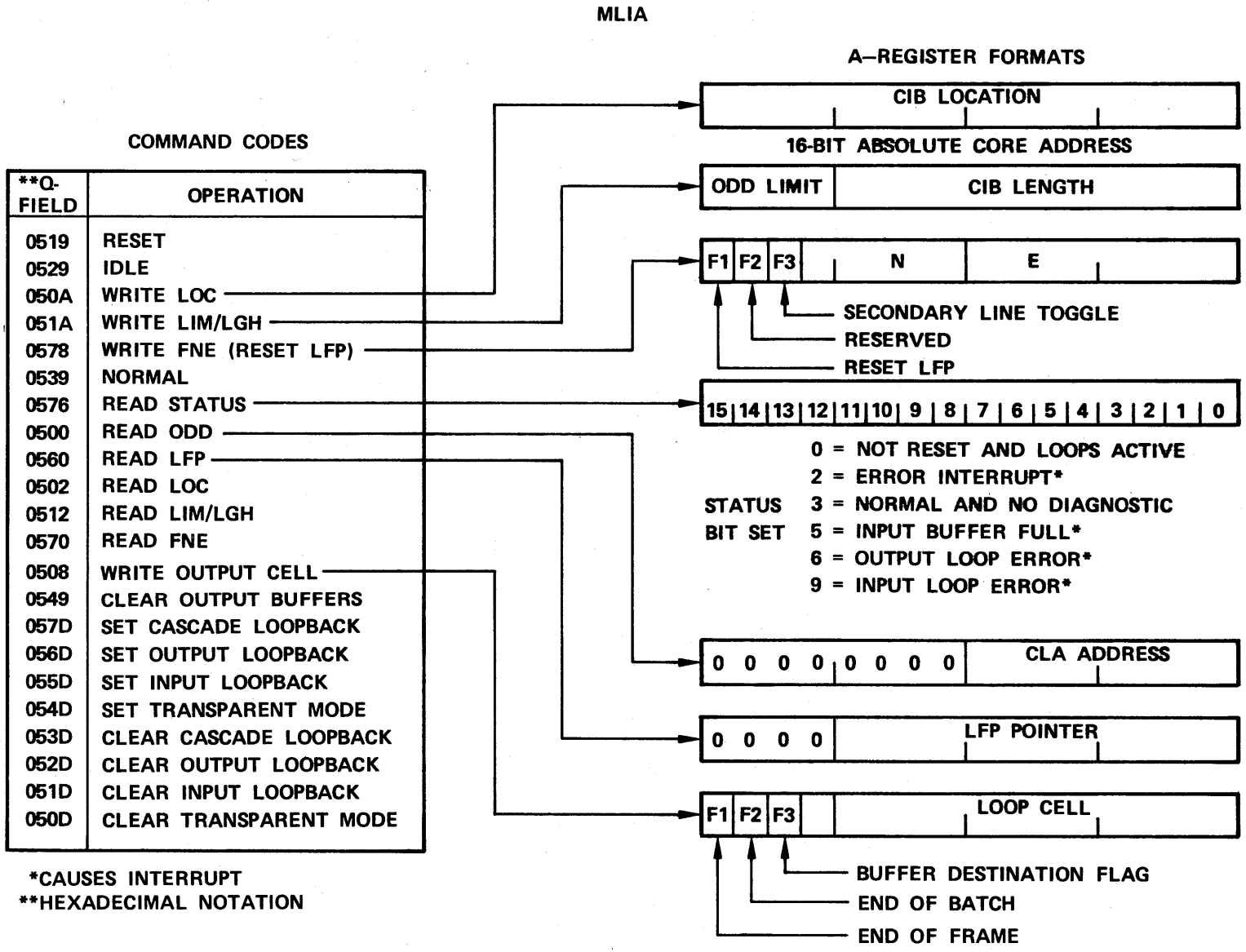


Figure 5-36. MLIA Commands

is always zero. There is only one MLIA in the HCP and it is assigned a fixed address determined by CP wiring. Tentatively, the equipment address of 'A' has been assigned.

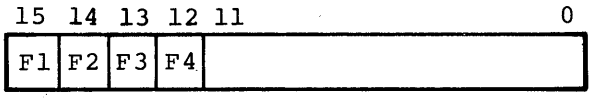
AQ Channel Output Operations

In the following sections, reference is made to A and Q registers.

OUTPUT COMMAND (Q=0508)

Upon receipt of this code, the contents of the 16 output data lines (A register) are transferred to the MLIA and stored in an output data holding buffer. Additional words are similarly transferred and stored. Whenever a word is transferred in where the end-of-loop-batch flag is set, the buffer empties its contents to form a loop batch, a CRCS is added, and line frames are transferred on to the LMs. The process continues until a word in the output buffer is reached containing both end-of-loop-batch and end-of-line frame. This terminates the output operation.

OUTPUT DATA (08_{16})



F1 = End of Line Frame (EOF)
F2 = End of Loop Batch (EOB)
F3 = ---
F4 = ---

RESET COMMAND (MODE) (Q=0519)

Upon receipt of this command, all operations are terminated, all buffers are reset, stored status is set to 0268 and interrupts are reset, all timing is reset, and all parameters are set to zero. Loop clock and data lines are inactive. Sync lock is lost. All diagnostic commands are reset.

IDLE COMMAND (MODE) (Q=0529)

Upon receipt of this command, all output operations resume as in the NORMAL mode, but input loop batches are not sent and continuous empty cells are sent. There are no loop ends or nulls. Immediately following a change from RESET mode to IDLE mode, Restart Loop End may or may not be sent, depending on random timing factors.

All error sensors are enabled. Note that upon entering this mode, interrupt status indicates that the unit was out of sync and now is (or soon will be) back in sync.

Input loop batch E and N parameter, all flags, input buffer location and length, etc., must all be properly initialized prior to entering NORMAL mode since operation starts immediately.

In IDLE mode the unit can output if desired, but all input is inhibited. This mode is used following a RESET command in order to load parameters; another use is to stop all input (risking CLA overrun) if the processor circular input buffer approaches capacity. (The E parameter controls the number of empty cells sent in each loop batch.)

NORMAL (MODE) (Q=0539)

Upon receipt of this command, the unit begins performing both input and output functions. As soon as one input loop batch returns, another is initiated. Output data is transmitted upon CP command (as in the IDLE mode). The MLIA remains in sync when switched from IDLE to NORMAL or from NORMAL to IDLE mode.

NOTE

Use extreme caution in issuing commands and/or parameters to this unit while running in NORMAL mode since the new changes are implemented immediately and the resulting state may not be a desirable condition.

CLEAR OUTPUT BUFFER (Q=0549)

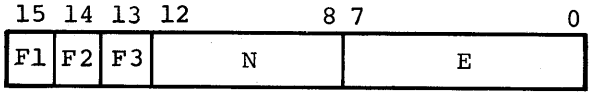
Clears both output buffers. The output data buffers read out only upon receipt of an end-of-loop-batch flag, and prior processing may inadvertently leave residuals in the FIFOs. This command initializes the buffer to the cleared condition prior to beginning a new operation. (Note that this command need not be sent if the RESET command was just used. This command should not be issued while the MLIA is still outputting in response to the EOB bit.)

WRITE FNE PARAMETERS (Q=0578)

Upon receipt of this code, the contents of the 16 output data lines (A register) are sampled and the resulting word transferred to the loop parameter register.

Data format is shown below. F1 is an embedded command (rather than a parameter per se) and results in the pointer for writing data into the CP circular input buffer (CIB) being set to zero (i.e., beginning of buffer). The remaining two flags are transmitted in the loop end cell.

SET LOOP PARAMETERS (78_{16})



F1 = Reset LFP (Last Frame Position) pointer (immediate command)
F2 = - (part of Loop End Cell)
F3 = Input from secondary lines (part of Loop End Cell)
N = Number of null cells generated for input loop batch
E = Number of empty cells generated for input loop batch

The F1 bit in the A field momentarily resets both the Next Word Position Pointer (NWP) and the Last Frame Position Pointer (LFP). The act of issuing the command with F1 set is what generates the reset, not the presence of the F1 bit itself. Hence, the F1 bit remains set but its effect is only momentary.

Presently, F2 is a spare flag. F3, however, is the controlling flag in dual systems. In this mode there are two MLIA units, each driven by a separate CP. Thus each LM cage connects to two sets of loops, each one through a separate LM card. In normal operation (flag F3 not set), each loop services half the traffic (its primary lines). Should either loop become inoperative for any reason, the other loop then alternately services the primary and secondary lines when loop-end-cell flag F3 is set.

Although flag F3 is set and stored, only alternate loop cells have the corresponding bit set. This is necessary to service high-priority lines in the same sequence as the original service. This assures proper handling of high-speed lines on both loops.

Care must be exercised in establishing values for N and E since too conservative an approach could slow down input loop response time unnecessarily. Generation of input loop batches is suspended any time either the ODD buffer or the input FIFO cannot accommodate the worst-case data pattern. Worst case for the ODD buffer occurs when only ODDs are received back (i.e., a CLA address containing an ODD, followed by the CRCS). Worst case for the input FIFO occurs when a maximum length line frame is begun by the last empty cell in the loop batch (the last empty cell prior to the string of nulls). Specific ground rules are as follows:

1. The N parameter does not specify the exact number of nulls in the generated input loop batch. Rather, the number of nulls is two greater than the value of the N parameter. Thus, the MLIA always generates at least two nulls, even if N is set to zero.

2. If a Loop End cell is received by MLIA on the input loop while the input loop batch is still being generated, the generation sequence is terminated and a new input loop batch is started immediately. Hence, the number of nulls and empties actually present on the input loop can be considerably less than the settings of the N and E parameters if no cells were inserted by a connected LM.

3. The MLIA tests only the main FIFO for worst-case data pattern; it does not test the ODD FIFO. The only space-available test on the ODD FIFO is controlled by the ODD limit parameter or programmed by the Write LIM/LGH command (Q=051) (A=051A). Hence, the maximum permissible value of the E parameter is determined by the size of the ODD FIFO.

NOTE

This operation is also affected by the ODD threshold parameter.

SET BUFFER LOCATION (Q=050A)

Upon receipt of this code, the contents of the 16 data lines (A register) are sampled and the resulting word transferred to the CIB location register. The lower four bits are not presently used.

SET BUFFER LOCATION ($0A_{16}$)

15	3	0
Circular Input Buffer Address	0 0 0 0	

Contents of the buffer location register determine the beginning of the area in CP memory into which the input FIFO is unloaded. Storage is sequential, continuing until the end of buffer is reached (determined by the buffer length parameter). At that time, the internal working pointer is reset to zero and writing resumes at the beginning of the buffer. Firmware must assure that unused information is not overwritten and thus lost.

SET BUFFER LENGTH/ODD LIMIT (Q=051A)

Upon receipt of this code, the contents of the 16 output data lines (A register) are sampled. The middle eight bits are transferred to the CIB length register; the lower four bits are not used at this time.

The length parameter is a full 12-bit field specifying the main memory address of the final word of the CIB. The address is relative to the location parameter programmed by the Write Location command (Q=050A). Thus, the absolute main memory address of the last word in the CIB is the sum of the length and location parameters, and the total number of words spanned by the buffer is one greater than the value of the length parameter. (See below for use of this parameter.)

SET BUFFER LENGTH/ODD LIMIT ($1A_{16}$)

15	11	3	0
ODD Limit	Circular Input Buffer Length	0 0 0 0	

The upper four bits are transferred to the ODD threshold register. Whenever the number of entries in the ODD FIFO register exceeds this threshold value, initiation of input loop batches is suspended until the number of stored ODDs again goes below this threshold.

SET TRANSPARENT MODE (Q=054D)

Upon receipt of this diagnostic mode command, normal editing functions performed on input loop batches received from the loop are inhibited. Thus all bits received are forwarded to the CP for subsequent analysis.

Several cautions should be noted in conjunction with this command. End-of-line-frame and end-of-loop-batch flags are not set. CRCS errors or sequence errors are not tested (although now the CRCS character is included). Normally the characters on the line outside loop batches are not sent, although this is dependent

upon the ability of the MLIA to detect the loop end character. Should this not be recognized for some reason, characters would continue to fill the buffer until the loop end timeout was reached. Thus the buffer could be overrun.

CLEAR TRANSPARENT MODE (Q=050D)

Upon receipt of this diagnostic mode command, the effect of the above ($4D_{16}$) command is cancelled.

SET INPUT LOOPBACK (Q=055D)

Upon receipt of this diagnostic mode command, input loop signals are shunted across an internal path rather than going through the normal external loop. Specifically, the input to the loop transmitter is connected as the output of the loop receiver (in lieu of the normal connections). Thus no traffic (data or clock) appears on the external loop.

This command can be used alone or in conjunction with the other diagnostic commands.

CLEAR INPUT LOOPBACK (Q=051D)

Upon receipt of this diagnostic mode command, the effect of the above ($5D_{16}$) command is cancelled.

SET OUTPUT LOOPBACK (Q=056D)

Upon receipt of this diagnostic mode command, output loop signals are shunted across an internal path rather than going through the normal external loop. Specifically, the input to the loop transmitter is connected as the output of the loop receiver (in lieu of the normal connections). Thus no traffic (data or clock) appears on the external loop.

This command can be used alone or in conjunction with the other diagnostic commands.

CLEAR OUTPUT LOOPBACK (Q=052D)

Upon receipt of this diagnostic mode command, the effect of the above ($6D_{16}$) command is cancelled.

SET CASCADE LOOPBACK (Q=057D)

Upon receipt of this diagnostic mode command, the two loops, whether normal external or internal loopback (or any combination), are linked together in a serial chain. Thus the output data passes through the output loop, then through the input loop, and finally back to the CIB as input data.

Virtually any data format can be sent over the chain, although the effects of the stream on the various LMs and CLAs must be considered. To provide total flexibility, the entire loop batch is issued by the CP. **End-of-line-frame** flags are ignored. **End-of-loop-batch** flags only trigger unloading of the buffer. In addition to the array of line frames to be sent (each consisting of CLA address, data/or supervision, and the CRCS character), the buffer must contain a loop end cell followed by at least one null (to define the end of loop batch). At all other times, empty cells (between loop batches) **are sent.**

CLEAR CASCADE LOOPBACK (Q=053D)

Upon receipt of this diagnostic mode command, the effect of the above ($7D_{16}$) command is cancelled.

CLEAR INTERRUPT

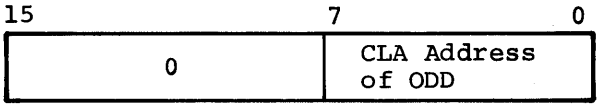
This code is reserved for the stated function for 1700 I/O compatibility, but is not implemented.

AQ Channel Input Operations

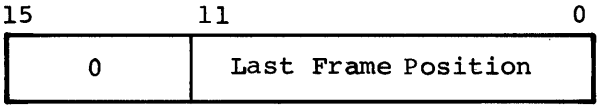
READ ODD (Q=0500)

Upon receipt of this code, the oldest entry in the ODD FIFO buffer is placed on the data lines and transferred into the low-order eight bits of the A register. This 8-bit field contains the address of the CLA demanding output data. The upper eight bits are zero. If ODD FIFO is empty, the value returned is always 00FF (since the MO5 channel is incapable of generating a REJECT). Data lines can assume any value and in general are not specified as to content. **Future ODDs are not lost.**

READ ODD (00_{16})



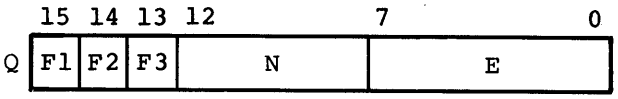
Read Last Frame Position Pointer (Q=0560)



Upon receipt of this code, the current value of the **last-frame-position** pointer is placed on the data lines and transferred into the low-order 12 bits of the A register. This field contains the displacement from the buffer origin of the last cell of the last complete line frame. The upper four bits are zero.

This 12-bit field contains the value of the CIB write pointer (measured as a displacement from the buffer origin) at which the last cell of the last complete line frame was written, plus one. It thus points to the location in which the first cell of the next line frame will be written. The value of this pointer is not updated until the next complete line frame has been written. Thus data in the next few buffer locations may have already been overwritten (and thus lost, if not already read). Note that whenever the value of the pointer equals the buffer length, it resets to zero and begins incrementing from that point (in the manner of a true CIB). Issuance of this code also resets the input data available interrupt.

READ FNE PARAMETERS (Q=0570)



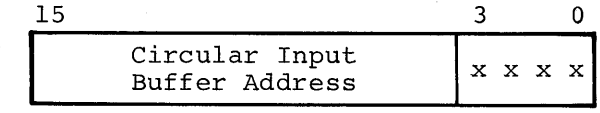
F1 = - (Reserved)
F2 = - (Part of loop end cell)
F3 = Input from secondary lines (part of loop end cell)
N = Number of null cells generated for input loop batch
E = Number of empty cells generated for input loop batch

Upon receipt of this code, the current value of the loop parameter register is placed on the data lines and transferred into the A register.

Note that through the use of this Read command and the corresponding set command (78_{16}), 16-bit words can be transferred bidirectionally across the channel, thus serving as a vehicle for testing data integrity across the PIO channel while in a diagnostic **(nonoperational) mode.**

As described above, the F1 bit of the A-field retains the state to which it was programmed by the last Write FNE command (Q=0578). It does not automatically clear itself after its function has been performed.

READ BUFFER LOCATION (Q=0502)



Upon receipt of this code, the current value of the CIB location register is placed on the data lines and transferred into the A

register. The value is the location into which one loop cell is written when the pointer is zero (reset). All 16 bits are programmed by the WRITE LOC command (Q=050A).

READ BUFFER LENGTH/ODD LIMIT (Q=0512)

15 — 11	3 — 0
ODD Limit	Circular Input Buffer Length: x x x x

Upon receipt of this code, the current value of the ODD threshold register is placed on the data lines and transferred into the upper four bits of the A register.

Concurrently, the current value of the CIB length register is placed on the data lines and transferred into the lower 12 bits of the A register; the lower four bits are not used at this time. All 16 bits are programmed by the WRITE LIM/LGH (Q=051A) command.

READ STATUS (Q=0576)

Upon receipt of this code, the current value of the MLIA status register is placed on the data lines and transferred into the A register. Issuance of this code also resets the normal MLIA external interrupt.

As shown in figure 5-36, several of the status bits represent stored conditions. These are reset after their state is reported to the CP.

Many of the bits, when on, represent error conditions, as indicated in figure 5-37. Bits not so marked are normally on.

Several of the bits, when on, cause the normal MLIA external interrupt line to be enabled. Note that the MLIA uses two additional, special-purpose interrupts: ODD present and input data available.

If MLIA is in RESET mode, the value returned in A register is 0268. Upon entering IDLE mode

STATUS (76_{16})

15															0
S_{15}	S_{14}	S_{13}	S_{12}	S_{11}	S_{10}	S_9	S_8	S_7	S_6	S_5	S_4	S_3	S_2	S_1	S_0

Bit	Function	Error	Interrupt	Reset
S0	Ready	E	I	
S1	(Busy)			
S2	Error interrupt		I	
S3	Normal and no diagnostic			
S4	(End of operation)			
S5	Input buffer full	E	I	R
S6	Output loop error	E	I	R
S7	Protected			
S8	Parity error	(E)	(I)	(R)
S9	Input loop error	E	I	R
S10				
S11				
S12				
S13	(Reserved)			
S14	(Reserved)			
S15	(Reserved)			

Figure 5-37. Read Status Format

from RESET, first status command yields 0265. Subsequent status commands yield 0001 unless an abnormal or diagnostic condition is present. Upon entering NORMAL mode, status should be 0009, if no diagnostics or errors are present.

The function of each status bit is as follows:

S0 Ready - Begin ready implies that the static operational requirements have been met: cards are plugged in, power applied, cables connected, etc. If this bit is not true for some reason, operator (and/or maintenance personnel) intervention is required. Activity sensors connected to the returning clock and data lines of each loop are conditions required to be ready.

Since the NCR M05 channel does not provide the ability to reject a command, this bit must be tested before becoming operational and during maintenance.

Bit S0 is true if the MLIA is not RESET and the input and output loops are active. Since the loops become inactive when loopback conditions are set, this status bit is false if either the the input loopback or output loopback or both are set.

Removal of a circuit board causes the MLIA to reset immediately and remain reset after the card has been reinstalled. If all MLIA boards are in place and operating properly, issuance of an IDLE or NORMAL command clears the RESET condition. Thus, bit S0 is false if a card is missing or was temporarily removed.

S1 Busy - Being busy usually implies a condition that will clear with time. Normally, it is sensed as one reason for a channel reject. However, this reject capability is not available on the NCR M05 channel. Since there is no time when the MLIA is busy and yet still capable of reading status, this bit is not implemented and is permanently off.

S2 Error Interrupt - When true, this bit indicates that one or more causes of the normal MLIA external interrupt are present. The exact cause may be found by examination of other status bits. It represents the logical OR of these bits. Upon responding to the READ STATUS code, these interrupt conditions (and therefore the external interrupt line) are reset.

S3 NORMAL Mode - Unlike some systems, this bit being true does not signal the presence of an interrupt requesting data. The ODD in the MLIA is a separate interrupt line. Furthermore, it does not represent a data interrupt in the context of the NCR ADT feature (which is not implemented in the MLIA). Rather, NORMAL is a status bit indicating that data transfers can occur. It signifies the "on-line" (non-diagnostic mode) condition.

S4 End of Operation - In most systems, this bit being true signifies that a block of data has been written, read, etc., over a DMA channel. Within the MLIA, this function, signifying that a line frame has been written into CP memory, is performed by a unique interrupt. This bit is currently not implemented within the MLIA.

S5 Input Buffer Full - When true, causes an interrupt. Set whenever the generation of a new input loop batch was delayed owing to lack of room in either the input FIFO or ODD FIFO buffer. Occurrence of this bit suggests that the number of empty/null cells parameter should be modified (or all loop operations halted). Reset by status read.

S6 Output Loop Error - When true, causes an interrupt. Set whenever output loop sync loss, output buffer overrun, or output loop end timeout is detected. (Note that, unlike the input loop, multiple loop batches can exist on the output loop, thus allowing the possibility of losing some loop end timeouts. However, the effect is sensed and reported by the CLA units.) Reset by status read.

Bit S6 does not respond to output buffer overrun. The MLIA does not detect overrun of the output buffers since writing of cells into the buffers is under direct control of the firmware/software.

S7 Protected - Normally true; indicates that the MLIA is capable of writing into protected memory. At present, no means are provided for inhibiting this bit.

S8 Parity Error - Not used in the MLIA since the DMA read capability is not implemented.

S9 Input Loop Error - When true, causes an interrupt. Set whenever input loop CRCS error, illegal format, sync loss, or loop timeout is sensed. Reset by status read.

In addition to the conditions listed above, bit S9 is also set by malfunction of the up/down counters for the main FIFO and ODD FIFO (i.e., invalid counter state). Such a malfunction normally does not occur, but can be induced under certain diagnostic conditions.

S10 (Unassigned)
thru
S12

S13 (Reserved)
thru
S15

NOTE

Bits S1, S4, S7, S8, and S10 through S15 are not implemented in the MLIA and are always zero.

DMA/DEVICE INTERFACE

Each DMA device connected to the MP DMA bus must have the logic to interface to the following signal lines:

16 Data Lines into Memory $\overline{\text{DTM01-16}}$

16 Data Lines from Memory $\overline{\text{DFM01-16}}$

16 Address Lines $\overline{\text{MAB01}}$ thru $\overline{\text{MAB16}}$

3 Control Lines from Memory

a) Memory Acknowledge ($\overline{\text{MCA}}$)

b) Data Strobe ($\overline{\text{MDS}}$)

c) Memory Cycle Timing ($\overline{\text{MCT}}$)

1 Control Line to Memory

a) Memory Cycle Request ($\overline{\text{MCR}}$)

2 Error Signals from Memory

a) Parity Error ($\overline{\text{MPE}}$)

b) DMA Out Error ($\overline{\text{MAE}}$)

The DMA signal line usage is shown in table 5-10.

The DMA signal timing listed in table 5-10 usage is defined below:

1. $\overline{\text{MCR}}$ (Memory Request) must be held low until it is reset (goes high) by $\overline{\text{MCA}}$-$\overline{\text{MCT}}$ or $\overline{\text{MCA}}$-$\overline{\text{MDS}}$. Address, data and write lines must be stable†† within 50 ns after $\overline{\text{MCR}}$ is lowered. $\overline{\text{MCR}}$ must be high within 300 ns after the leading edge of $\overline{\text{MDS}}$ unless an additional memory cycle is requested. $\overline{\text{MCR}}$ may be held continuously low for consecutive memory cycles; however data, address, and write lines must be stable†† within 700 ns

††stable at the input of the open collector gates used to drive common bus into memory

TABLE 5-10. DMA SIGNAL LINE USAGE

FUNCTION	NAME	LOGIC LEVEL	OPEN COLLECTOR	COMMON SIGNAL
Address lines	$\overline{\text{MAB01-16}}$	Low	Yes	Yes
Data Lines to Memory	$\overline{\text{DTM01-16}}$	Low	Yes	Yes
Data Lines from Memory	$\overline{\text{DFM01-16}}$	High	Yes	Yes
Memory Write	MW	High	Yes	Yes
Memory Request	$\overline{\text{MCR}}$†	Low	No	No
Memory Acknowledge	$\overline{\text{MCA}}$†	Low	No	No
Memory Timing	$\overline{\text{MCT}}$	Low	No	Yes
Data Strobe	$\overline{\text{MDS}}$	Low	No	Yes
Parity Error	$\overline{\text{MPE}}$	Low	No	Yes
Address Error	$\overline{\text{MAE}}$	Low	Yes	Yes

†Each DMA device is assigned a unique pair of these signal lines.

after the trailing edge of the previous $\overline{\text{MDS}}$. MCR must drive two TTL loads. MCR must be high at power initiation time. A way to guarantee that MCR is a high is to use $\overline{\text{MR}}$ to set (or reset) the latch that creates $\overline{\text{MCR}}$.

2. $\overline{\text{MCA}}$ (Memory Acknowledge) is used to gate data and address lines and the write line to a common bus connected to the memory interface logic. Address, data, and write lines must not change when $\overline{\text{MCA}}$ is how. Loading on $\overline{\text{MCA}}$ must not exceed eight TTL loads.

3. $\overline{\text{DTM01 -16}}$ are data lines from the CP or DMA to memory. Each data line must be driven with an open collector gate.

4. MW is the memory write line. MW must be driven with an open collector gate.

5. $\overline{\text{MAB01-16}}$ are address lines with MAB01 the least significant line. Address lines are driven with an open collector gate.

6. $\overline{\text{MCT}}$ (Memory Timing) is a logic signal which occurs 160 ns before data strobe. This signal is used by a DMA interface; loading should not exceed one TTL load. A common $\overline{\text{MCT}}$ signal is sent to all DMAs.

7. $\overline{\text{MDS}}$ (Data Strobe) is used to strobe data out of memory. Data **out of memory is valid for** 30 ns before the leading edge of $\overline{\text{MDS}}$ and **remains valid until the** start of the next memory cycle (800 ns minimum). Loading on $\overline{\text{MDS}}$ must not exceed eight TTL loads.

8. DFM01-16 are data lines from memory. Loading must not exceed two TTL loads on each data line.

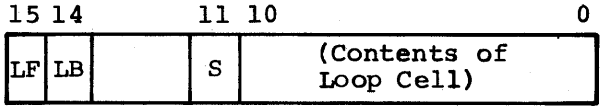
9. $\overline{\text{DMA-MPE}}$ (Parity Error) is sent when a parity error occurs in memory. $\overline{\text{DMA-MPE}}$ is a common signal sent to all DMAs. MPE occurs shortly after data strobe but before MCA goes high. Loading on $\overline{\text{DMA-MPE}}$ should not exceed one TTL load.

10. $\overline{\text{DMA-MAE}}$ is signal sent on a common signal line to all DMAs. It indicates that the DSA scanner control is in the Out position and access to CP memory cannot be achieved.

DMA Channel Input Operations

The MLIA uses the DMA channel to store input line frames directly in the CP memory (CIB). Except in the transparent mode, CRCS characters are deleted. Line frames containing only ODD and CRCS are deleted even when they are in the last frame in the loop batch; under these conditions, the end-of-loop-batch flag is dropped and does not appear in the CP. Note that in the CIB, mode six and seven have no distinction in terms of meaning and should be equated by the user.

INPUT DATA - This operation moves information from the main input FIFO to the CIB in CP memory. Format is as follows:



LF = End-of-line-frame flag
LB = End-of-loop-batch flag
S = Sync bit of loop cell

Loop cell contents are forwarded right-justified exactly as received from the loop. Bit 11 is the sync bit, bits 10-8 are the format bits, and bits 7-0 contain the specific information (CLA address, data, supervision, etc.).

End-of-line-frame flag is set in the last word stored in a line frame. It is associated with the loop cell immediately preceding the CRCS and typically contains the data (or supervision) character. It is set in the CLA address character only when the CLA address contains an ODD and is the last line frame of a loop batch. This flag is not set when in the transparent mode.

End-of-loop-batch is set along with end-of-line-frame in the last word stored in a loop batch. It cannot occur without an accompanying end-of-line-frame and thus meets all of the preceding requirements. This flag is set when in the transparent mode.

At the conclusion of each line frame transfer into the CIB, a special interrupt is turned on (if not already on). It alerts the firmware that the CIB read and write pointers are unequal owing to the storage of added information. The interrupt resets whenever the LFP pointer value is read.

DMA Channel Output Operations

The low-performance MLIA does not use the DMA output capabilities.

INTERRUPTS

Operation of the MLIA interrupt system differs markedly from more conventional users owing to the need for high processing efficiency plus the unique capabilities of the CP hardware and firmware.

CP Interrupt System

Conventional processors, upon receipt of an external interrupt, halt processing at the conclusion of the present instruction and execute special interrupt code before resuming normal operation. In the 2550-2 HCP, however, interrupt lines are merely external data sources for loading a special register. Under firmware control, these register inputs are strobed and thus available for any use by reading through a programmable mask register.

Two approaches are used depending on whether, at the time of interrupt occurrence, control is held by the emulation firmware or the special

system firmware. The latter approach is used by the multiplex subsystem portion of the communications processing package, while the former approach is used both by higher level (macro) code and all diagnostics.

Standard emulation samples the interrupt register as part of the Read Next Instruction (RNI) phase of each software instruction execution. Should either an internal (special system) or external (standard macro) interrupt be detected, control is then passed to the special firmware or the emulation package (respectively).

Special multiplex system firmware samples the interrupt register on a time basis. Under this plan, the register is sampled a minimum of once every 20 microseconds. During the interval between samples, numerous macroinstructions can be processed. Further, this sampling can be delayed until any level of critical processing is completed without incurring time penalties in extensive interrupt lockouts. It is also possible to assign some of these interrupts as internal and others as external.

Interrupt Implementation

Within the NCR MO5 channel specification, two interrupts are specified: one program and one data interrupt. The MLIA uses the program interrupt as the normal external interrupt and uses the bits of the status word to identify the specific cause. The data interrupt is not implemented within the MLIA since it is used in conjunction with the ADT feature. However, two additional, unique interrupts are provided as described below.

Program Interrupts

Main use of this interrupt is to report MLIA error conditions. The MLIA protects itself against data loss, forcing the effect to show up on a per-line basis as overruns and/or underruns. Using the bit of the status word to identify error causes, error statistics are accumulated and recorded. When the number of errors exceeds some programmable threshold, software/firmware marks the unit down as in need of maintenance.

This interrupt is not cleared by setting bit one in the A register and then issuing a function code. To preserve operating efficiency, the A register is not used. Instead, a unique code (76_{16}) in the address (Q register) requests that MLIA status be read. As a part of this operation, stored conditions are reset and, since it is the logical OR of these bits that drives the interrupt, it becomes reset.

Output Data Demand Interrupt

Whenever the ODD FIFO contains any entry (or entries), this line is true. Its presence (or absence is controlled entirely by the contents, of this ODD FIFO. Whenever, through reading out the FIFO contents, the internal contents counter goes to zero, the interrupt automatically resets.

Input Line Frame Interrupt

Each time a complete line frame has been stored in CP memory, this interrupt is raised. Since input data is a low-priority task, several frames may accumulate before the firmware uses this interrupt. Its main purpose is to eliminate the need for firmware comparing the CIB read and write pointers in order to detect recent activity.

Whenever the current value of the last-frame-position (LFP) pointer is sensed (60_{16}), this interrupt is automatically reset.

NOTE

Each of the three interrupts uses DC-type (rather than pulse-type) signalling and thus

does not depend on CP internal storage. Depending upon system requirements, any or all of these interrupts can be wired as both internal and external simultaneously, thus providing ready access for operational software, firmware, and diagnostics.

COMMUNICATIONS LINE ADAPTERS

Interface to external modems and/or locally attached terminals is via a limited number of communications line adapters (CLAs). Different types of CLA units are required for major electrical differences (e.g., synchronous versus asynchronous, RS-232/V-24 versus differential/V-35 versus coaxial).

Other parameters such as speed, sync code, character length, and parity are controlled by software/firmware through the use of parameter registers. Supervisory commands are sent to the CLA units to set up these parameter registers; correspondingly, status can be received back from the CLA units, either upon command or upon the occurrence of a specified event.

For inbound traffic, CLA units strip off all communications-oriented distinctive characteristics which can be removed without knowing the meaning of the bits (e.g., code set, line protocol). At the interface to the LM, each CLA presents characters one at a time (character serial - at a rate determined by the external communications channel speed), bit paralleled, right-justified, properly framed, with housekeeping characters (start/stop bit, parity, etc.) removed. At this interface all CLA units present an identical interface. Outbound, this process is reversed.

As each inbound character is received, it is converted to bit-parallel form and transferred to a one-word buffer where it is held until it can be placed on the loop. Similarly, outbound characters leave a one-word buffer as they become serialized and are sent out. Each time this outbound buffer is empty (and the CLA is active), an ODD is sent to the CP for the next character. Failure to provide or accept characters in a timely manner results in an overrun/underrun condition.

In summary, the CLA accepts supervision (i.e., control word) and output data from the loop and yields status, input data and ODDs to the loop.

Specific programming information for any specific CLA can be obtained from the corresponding CLA Hardware Reference/Maintenance Manual (see Preface).

APPENDIX A
MNEMONICS LISTING

MNEMONIC AND DESCRIPTION

ALU
Arithmetic and Logical Unit

A/Q
Type of IDC Interface (CDC)

BG
Bit Generator

BP
Breakpoint

CCITT
Consultive Committee, International Telephone & Telegraph

CDC
Co[illegible]ata Corporation

CE
Cyclic Encoder

CIB
Circular Input Buffer

CLA
Communications Line Adapter

CLE
Communications Line Expansion (Unit)

CLR
Clear

CLRF
Clear Flags

CP
Communications Processor

CRC
Cyclic Redundancy Checksum (created by Cyclic Encoder)

CRCS
Cyclic Redundancy Checksum (created by Multiplex Loop Interface Adapter)

CRT
Cathode Ray Tube

CTS
Clear to Send

DMA
Direct Memory Access

DSA
Direct Storage Access

DSR
Data Set Ready

DTO
Data Transfer Overrun

DTR
Data Terminal Ready

DTU
Data Transfer Underrun

ECHO
Echoplex Mode

EIA
Electronic Industries Association

EOB
End of Loop Batch

EOF
End of Line Frame

EXGO
External Go

EXMC
External Master Clear

EXSTOP
External Stop

FCD
First Character Displacement

FCO
First Character Outbound

FCR
Function Control Register

FES
Framing Error Status

FIFO
First In, First Out

FNE
Flag, Null, Empty

HCP
Host Communications Processor

IAV
Input Available

IC
Integrated Circuit

IDC
Internal Data Channel

IEN
Input End

IER
Input Error

I/F
Interface

ILE
Input Loop Error

INS
Input Select

INT
Interrupt

INTU
Interrupt Test

I/O
Input/Output

ION
Input Section On

ISON
Input Supervisor On

ISR
Input Supervision Report

IST
Input Strobe

LB
End of Loop Batch Flag

LCD
Last Character Displacement

LED
Light Emitting Diode

LF
End of Line Frame Flag

LFP
Last Frame Position

LGH
Length

LIM
Limit

LIT
Loop Input Test

LM
Loop Multiplexer

LOC
Location

LRC
Longitudinal Redundancy Checksum

LSB
Least Significant Bit

LSI
Large Scale Integration

MA
Micromemory Address

MAC
Memory Address Counter

MAE
Memory Address Error

MCA
Memory Acknowledge

MCL
Master Clear

MCR
Memory Cycle Request

MCT
Memory Cycle Timing

MDS
Memory Data Strobe

MIR
Microinstruction Register

MLIA
Multiplex Loop Interface Adapter

MM
Micromemory

MML
Micromemory Lower

MMU
Micromemory Upper

Modem
Modulator/Demodulator

M05
Type of IDC Interface (NCR)

MOS
Metal Oxide Semiconductor

MP
Microprocessor

MPE
Memory Parity Error

MPP
Microprogrammed Processor

MR
Master Reset Line

MSB
Most Significant Bit

MSI
Medium Scale Integration

MW
Memory Write

NCNA
Next Character Not Available

NSYN
New Synchronization

NWP
Next Word Position

ODD
Output Data Demand

OER
Output Error

OLE
Output Loop Error

OM
Originate Mode

OON
Output Section On

OSC
Output Select Clear

OSL
Output Select

OST
Output Strobe

PAD
Process Additional Data

PC
Program Controlled

PCB
Printed Circuit Board

PCD
Present Character Displacement

PES
Parity Error Status

PG
Protective Ground

PI
Parity Inhibit

PIO
Program Input/Output

P/MA
Page/Micromemory Address

PPU
Peripheral Processing Unit

PROM
Programmable Read Only Memory

PS
Page Storage

PSET
Parity Set

RAM
Random Access Memory

RD
Receive Data

RDATA
Receive Data (Modem)

RETL
Return Address Location

RI
Ring Indicator

RLSD
Receive Line Signal Detector

RNI
Read Next Instruction

ROM
Read Only Memory

RSD
Restraint Detector

RSYN
Resynchronize

RTJ
Return Jump

RTS
Request to Send

R/W
Read/Write

SB
Stop Bit

SCL
Serial Computation Logic

SCR
Serial Block Receive

SCT
Serial Clock Transmit

SD
Send Data

SETF
Set Flags

SM
Status Mode

SMI
Status Mode Interrupt

SO
Storage Register Outbound

SQD
Signal Quality Detector

SRLSD
Secondary Receive Line Signal Detector

SRTS
Secondary Request to Send

SSEE
Test Section and Error Number

SSI
Small Scale Integration

SSTB
Strobe Line

STERM
Terminate

STJP
Stop Jump

TB
Terminal Busy

TBD
To Be Determined (or defined)

TDATA
Transmit Data

TMA
Transmit Memory Address

TTL
Transistor/Transistor Logic

TTY
Teletypewriter

U/L
Upper/Lower

XT/MA
Transform/Memory Address

APPENDIX B
HEXADECIMAL/DECIMAL CONVERSION

TO CONVERT DECIMAL TO HEXADECIMAL

1. Find decimal number in body of table (Example: 157).
2. Scan horizontally to the left to find the first hexadecimal digit (in this case, 9).
3. Scan vertically up (from 157 in table) to find second hexadecimal digit (in this case, D).
4. Thus decimal number 157 = hexadecimal number 9D.

TO CONVERT HEXADECIMAL TO DECIMAL

1. Find first hexadecimal digit in left-hand column (Example: D).
2. Find second hexadecimal digit in top row (Example: 9).
3. Simultaneously scan horizontally to right (from D) and scan vertically downward (from 9) to find point of intersection in body of table (in this case, 217).
4. Thus hexadecimal number D9 = decimal number 217.

TABLE B-1. HEXADECIMAL/DECIMAL CONVERSION

FIRST HEXADECIMAL DIGIT	SECOND HEXADECIMAL DIGIT															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
4	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
5	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
6	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
7	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
8	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
9	144	145	146	147	148	149	150	151	152	153	154	155	156	157	518	159
A	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
B	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
C	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
D	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
E	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
F	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

APPENDIX C
FUNCTIONAL LISTING OF MACROINSTRUCTIONS

Function	Mnemonic	Code	Instruction Description
Transfers	LDA	Cxxx	Load A (Storage Reference)
	STA	6xxx	Store A (Storage Reference)
	LDQ	Exxx	Load Q (Storage Reference)
	STQ	4xxx	Store Q (Storage Reference)
	SPA	7xxx	Store A, Parity to A (Storage Reference)
	ENA	0Axx	Enter A (Register Reference)
	ENQ	0Cxx	Enter Q (Register Reference)
	TRA	08Ax	Transfer A (Inter Register)
	TRQ	089x	Transfer Q (Inter Register)
	TRB	089x	Transfer Q or M (Inter Register)
	LRr	04xxC0xx	Load Register r (Enhanced Storage Reference)
	SRr	04xxC1xx	Store Register r (Enhanced Storage Reference)
	LCA	04xxC2xx	Load Character to A (Enhanced Storage Ref.)
	SCA	04xxC3xx	Store Character From A (Enhanced Stor. Ref.)
	LFA	05xxxxxx	Load Field (Field Reference)
	SFA	05xxxxxx	Store Field (Field Reference)
	XFr R	07xx	Transfer Register (Enhanced Inter Register)
	LMM	0B01	Load Micromemory (Enhanced Register Ref.)
	LRG	0B02	Load Registers (Enhanced Register Reference)
	SRG	0B03	Store Registers (Enhanced Reg. Reference)
	LUB R	0Bx0	Load Upper Unprotected Bounds (Enhanced Register Reference)
	LLB R	0Bx1	Load Lower Unprotected Bounds (Enhanced Register Reference)
Arithmetic	ADD	8xxx	Add to A (Storage Reference)
	SUB	9xxx	Subtract From A (Storage Reference)
	ADQ	Fxxx	Add to Q (Storage Reference)
	RAO	Dxxx	Replace Add One in Storage (Storage Ref.)
	MUI	2xxx	Multiply Integer (Storage Reference)
	DVI	3xxx	Divide Integer (Storage Reference)
	INA	09xx	Increase A (Register Reference)
	INQ	0Dxx	Increase Q (Register Reference)
	SET	088x	Set to Ones (Inter Register)
	CLR	084x	Clear to Zero (Inter Register)
	AAM	082x	Transfer Arithmetic Sum A, M (Inter Register)
	AAQ	083x	Transfer Arithmetic Sum A, Q (Inter Register)
	AAB	083x	Transfer Arithmetic Sum A, Q + M (Inter Reg.)
	ARr	04xx80xx	Add Register r (Enhanced Storage Reference)
	SBr	04xx90xx	Subtract Register r (Enhanced Storage Ref.)
	SEF	05xxxxxx	Set Field (Field Reference)
	CLF	05xxxxxx	Clear Field (Field Reference)
Logical	AND	Axxx	AND with A (Storage Reference)
	EOR	Bxxx	Exclusive OR with A (Storage Reference)
	TCA	086x	Transfer Complement A (Inter Register)
	TCM	084x	Transfer Complement M (Inter Register)
	TCQ	085x	Transfer Complement Q (Inter Register)
	TCB	085x	Transfer Complement Q + M (Inter Register)

Function	Mnemonic	Code	Instruction Description
Logical (cont'd)	EAM	086x	Transfer Exclusive OR A, M (Inter Register)
	EAQ	087x	Transfer Exclusive OR A, Q (Inter Register)
	EAB	087x	Transfer Exclusive OR A, Q + M (Inter Reg.)
	LAM	08Ax	Transfer Logical Product A, M (Inter Reg.)
	LAQ	08Bx	Transfer Logical Product A, Q (Inter Reg.)
	LAB	08Bx	Transfer Logical Prod. A, Q + M (Inter Reg.)
	CAM	08Ex	Transfer Comp. Logical Prod. A, M(Inter Reg.)
	CAQ	08Fx	Transfer Comp. Logical Prod. A, Q(Inter Reg.)
	CAB	08Fx	Transf. Comp. Logical Prod. A, Q + M (Inter Register)
	ANr	04xxA0xx	AND Register (Enhanced Storage Reference)
	AMr	04xxA1xx	AND Memory, r (Enhanced Storage Reference)
	ORr	04xxD0xx	OR Register r (Enhanced Storage Reference)
	OMr	04xxD1xx	OR Memory, r (Enhanced Storage Reference)
	GPE	0B08	Generate Character Parity Even (Enhanced Register Reference)
	GPO	0B09	Generate Character Parity Odd (Enhanced Register Reference)
Jumps and Stops	JMP	1xxx	Jump (Storage Reference)
	RTJ	5xxx	Return Jump (Storage Reference)
	SLS	0100	Selective Stop (Register Reference)
	NOP	0B00	No Operation (Register Reference)
	SJE	04xx50xx	Subroutine/Jump Exit (Enhanced Storage Ref.)
	SJr	04xx50xx	Subroutine r Jump (Enhanced Storage Ref.)
	EMS R	0Bx2	Execute Micro Sequence (Enhanced Reg. Ref.)
Decisions	SAZ	010x	Skip if A = +0 (Register Reference)
	SAN	011x	Skip if A ≠ +0 (Register Reference)
	SAP	012x	Skip if A = + (Register Reference)
	SAM	013x	Skip if A = - (Register Reference)
	SQZ	014x	Skip if Q = +0 (Register Reference)
	SQN	015x	Skip if Q ≠ +0 (Register Reference)
	SQP	016x	Skip if Q = + (Register Reference)
	SQM	017x	Skip if Q = - (Register Reference)
	SWS	018x	Skip if Skip Switch Set (FCR)(Register Ref.)
	SWN	019x	Skip if Skip Switch not set (FCR)(Reg. Ref.)
	SOV	01Ax	Skip on Overflow (FCR)(Register Reference)
	SNO	01Bx	Skip on No Overflow (FCR)(Register Ref.)
	SPE	01Cx	Skip on Storage Parity Error (FCR)(Reg. Ref.)
	SNP	01Dx	Skip on No Storage Parity Error (FCR) (Register Reference)
	SPF	01Ex	Skip on Program Protect Fault (FCR) (Register Reference)
	SNF	01Fx	Skip on No Program Protect Fault (FCR) (Register Reference)
	CrE	04xxE0xx	Compare Register r Equal (Enhanced Storage Reference)
	CCE	04xxE2xx	Compare Character Equal (Enhanced Storage Reference)

Function	Mnemonic	Code	Instruction Description
Decisions (cont'd)	SFZ	05xxxxxx	Skip if Field Zero (Field Reference)
	SFN	05xxxxxx	Skip if Field Not Zero (Field Reference)
	SrZ SK	00xx	Skip if Register r Zero (Enhanced Reg. Ref.)
	SrN SK	00xx	Skip if Register r Non-zero (Enhanced Register Reference)
	SrP SK	00xx	Skip if Register r Positive (Enhanced Register Reference)
	SrM SK	00xx	Skip if Register r Negative (Enhanced Register Reference)
	DrP SK	06xx	Decrement and Repeat if r Positive (Enhanced Register Reference)
Shifts	ARS	0Fxx	A Right Shift (sign extended)(Register Ref.)
	QRS	0Fxx	Q Right Shift (sign extended)(Register Ref.)
	LRS	0Fxx	Long (QA) Right Shift (sign extended) (Register Reference)
	ALS	0Fxx	A Left Shift (end-around) (Register Ref.)
	QLS	0Fxx	Q Left Shift (end-around) (Register Ref.)
	LLS	0Fxx	Long (QA) Left Shift (end-around) (Register Reference)
Input/ Output	INP	02xx	Input to A (Register Reference)
	OUT	03xx	Output from A (Register Reference)
	SIO	0B04	Set/Sample Output or Input (Enhanced Register Reference)
	SPS	0B05	Sample Position/Status (Enhanced Reg. Ref.)
Interrupt	EIN	0400	Enable Interrupt (Register Reference)
	IIN	0500	Inhibit Interrupt (Register Reference)
	EXI	0Exx	Exit Interrupt State (Register Reference)
	DMI	0B06	Define Micro-Interrupt (Enhanced Reg. Ref.)
	CBP	0B07	Clear Breakpoint Interrupt (Enhanced Register Reference)
Program Protect	SPB	0600	Set Program Protect (Register Reference)
	CPB	0700	Clear Program Protect (Register Reference)
Extended Memory Page Control	PM0	0B0B	Set absolute page mode (0 → AMR) (Enhanced Register Reference)
	PM1	0B0C	Set page mode 1, "monitor" (2 → AMR) (Enhanced Register Reference)
	PM2	0B0D	Set page mode 2, "program" (3 → AMR) (Enhanced Register Reference)
	RPR R	0Bx3	Read page register (Enhanced Register Ref.)
	WPR R	0Bx4	Write page register (Enhanced Register Ref.)

NOTE: The term "r" can assume values of 1, 2, 3, 4, Q, A, or I. These result in machine code values of one thorugh seven, respectively.

APPENDIX D
NUMERIC LISTING OF MACROINSTRUCTIONS

Hex			Δ (Binary)		Mnemonic	Instruction
F	Fl	Δ	7 6 5 4	3 2 1 0		
0	0	Δ	0 0 0 0	0 0 0 0	SLS	Selective Stop
			0 0 0 0		S4Z SK	Skip if Register 4 Zero
			0 0 0 1	Skip	S4N SK	Skip if Register 4 Non-zero
			0 0 1 0	Count	S4P SK	Skip if Register 4 Positive
			0 0 1 1		S4M SK	Skip if Register 4 Negative
			0 1 0 0		S1Z SK	Skip if Register 1 Zero
			0 1 0 1		S1N SK	Skip if Register 1 Non-zero
			0 1 1 0		S1P SK	Skip if Register 1 Positive
			0 1 1 1		S1M SK	Skip if Register 1 Negative
			1 0 0 0		S2Z SK	Skip if Register 2 Zero
			1 0 0 1		S2N SK	Skip if Register 2 Non-zero
			1 0 1 0		S2P SK	Skip if Register 2 Positive
			1 0 1 1		S2M SK	Skip if Register 2 Negative
			1 1 0 0		S3Z SK	Skip if Register 3 Zero
			1 1 0 1		S3N SK	Skip if Register 3 Non-zero
			1 1 1 0		S3P SK	Skip if Register 3 Positive
0	0	Δ	1 1 1 1	SK	S3M SK	Skip if Register 3 Negative
0	1	Δ	0 0 0 0	Skip	SAZ	Skip if A = +0
			0 0 0 1	Count	SAN	Skip if A ≠ +0
			0 0 1 0		SAP	Skip if A = +
			0 0 1 1		SAM	Skip if A = -
			0 1 0 0		SQZ	Skip if Q = +0
			0 1 0 1		SQN	Skip if Q ≠ +0
			0 1 1 0		SQP	Skip if Q = +
			0 1 1 1		SQM	Skip if Q = -
			1 0 0 0		SWS	Skip if switch set
			1 0 0 1		SWN	Skip if switch not set
			1 0 1 0		SOV	Skip on overflow
			1 0 1 1		SNO	Skip on no overflow
			1 1 0 0		SPE	Skip on storage parity error
			1 1 0 1		SNP	Skip on no storage parity error
			1 1 1 0		SPF	Skip on Program Protect Fault
			1 1 1 1		SNF	Skip on no Program Protect Fault
0	2	XX			INP	Input to A
0	3	XX			OUT	Output from A
0	4	Δ	0 0 0 0	0 0 0 0	EIN	Enable Interrupts
			≠0			(see Note)

NOTE: When Δ ≠ 0 for 04Δ, a type 2 storage instruction is indicated. This is a 2 (or 3) word instruction in which the second word determines the operation (add, subtract, etc.). See special list at the end of this table.

Hex			Δ (Binary)								Mnemonic	Instruction
F	F1	Δ	7	6	5	4	3	2	1	0		
0	5	Δ	0	0	0	0	0	0	0	0	IIN	Inhibit Interrupts
								0	1	0	SFZ	Skip if Field Zero
					≠0			0	1	1	SFN	Skip if Field Not Zero
								1	0	0	LFA	Load Field
								1	0	1	SFA	Store Field
								1	1	0	CLF	Clear Field
								1	1	1	SEF	Set Field

NOTE: Special format double word instruction.

0	6	Δ	0	0	0	0	0	0	0	0	SPB	Set Program Protect
			0	0	1	0					D1P SK	Decrement and repeat if 1 positive
			0	1	0	0	Skip				D2P SK	Decrement and repeat if 2 positive
			0	1	1	0	Count				D3P SK	Decrement and repeat if 3 positive
			1	0	0	0					D4P SK	Decrement and repeat if 4 positive
			1	0	1	0					DQP SK	Decrement and repeat if Q positive
			1	1	0	0					DAP SK	Decrement and repeat if A positive
0	6	Δ	1	1	1	0	SK				DIP SK	Decrement and repeat if I positive
0	7	Δ	0	0	0	0	0	0	0	0	CPB	Clear Program Protect
			0	0	1	0	0	0	0	1	XF1 1	Transfer Register
			0	1	0	0	0	0	0	1	XF2 1	Transfer Register
			0	1	1	0	0	0	0	1	XF3 1	Transfer Register
			1	0	0	0	0	0	0	1	XF4 1	Transfer Register
			1	0	1	0	0	0	0	1	XFQ 1	Transfer Register
			1	1	0	0	0	0	0	1	XFA 1	Transfer Register
			1	1	1	0	0	0	0	1	XFI 1	Transfer Register
			0	0	1	0	0	0	1	0	XF1 2	Transfer Register
			0	1	0	0	0	0	1	0	XF2 2	Transfer Register
			0	1	1	0	0	0	1	0	XF3 2	Transfer Register
			1	0	0							
						0	0	1	1	1		
			1	0	1	0	0	1	1	1	XFQ I	Transfer Register
			1	1	0	0	0	1	1	1	XFA I	Transfer Register
			1	1	1	0	0	1	1	1	XFI I	Transfer Register
0	8	Δ	0	0	1	0	1				AAM	Transfer Arithmetic Sum A, M
			0	0	1	1	0	Dest.			AAQ	Transfer Arithmetic Sum A, Q
			0	0	1	1	1	Reg.			AAB	Transfer Arithmetic Sum A, Q + M
			0	1	0	0	0				CLP	Clear to Zero
			0	1	0	0	1	Same			TCM	Transfer Complement M
			0	1	0	1	0	code			TCQ	Transfer Complement Q
			0	1	0	1	1	as			TCB	Transfer Complement Q + M
			0	1	1	0	0	Orig.			TCA	Transfer Complement A
			0	1	1	0	1	Reg.			EAM	Transfer Exclusive OR A, M
			0	1	1	1	0				EAQ	Transfer Exclusive OR A, Q
			0	1	1	1	1				EAB	Transfer Exclusive OR A, Q + M
			1	0	0	0	0				SET	Set to One
			1	0	0	1	0				TRQ	Transfer Q
			1	0	0	1	1				TRB	Transfer Q or M
			1	0	1	0	0				TRA	Transfer A

Hex			Δ (Binary)								Mnemonic	Instruction
F	F1	Δ	7	6	5	4	3	2	1	0		
0	8	Δ (cont'd)	1	0	1	0	1				LAM	Transfer Logical Product A, M
			1	0	1	1	0				LAQ	Transfer Logical Product A, Q
			1	0	1	1	1				LAB	Transfer Logical Product A, Q + M
			1	1	1	0	1				CAM	Transfer Comp. Logical Prod. A, M
			1	1	1	1	0				CAQ	Transfer Comp. Logical Prod. A, Q
			1	1	1	1	1				CAB	Transfer Comp. Logical Prod. A, Q + M
0	9	XX									INA	Increase A
0	A	XX									ENA	Enter A
0	B	Δ	0	0	0	0	0	0	0	0	NOP	No Operation
			0	0	0	0	0	0	0	1	LMM	Load Micromemory
			0	0	0	0	0	0	1	0	LRG	Load Registers
			0	0	0	0	0	0	1	1	SRG	Store Registers
			0	0	0	0	0	1	0	0	SIO	Set/Sample Output or Input
			0	0	0	0	0	1	0	1	SPS	Sample Position/Status
			0	0	0	0	0	1	1	0	DMI	Define Micro Interrupt
			0	0	0	0	0	1	1	1	CBP	Clear Breakpoint Interrupt
			0	0	0	0	1	0	0	0	GPE	Generate Character Parity Even
			0	0	0	0	1	0	0	1	GPO	Generate Character Parity Odd
			0	0	0	0	1	0	1	1	APM	Set absolute page mode (>64K MOS)
			0	0	0	0	1	1	0	0	PM0	Set page mode 0 (monitor) (>64K MOS)
			0	0	0	0	1	1	0	1	PM1	Set page mode 1 (program) (>64K MOS)
0	B	Δ	0	0	1	0	0	0	0	0	LUB 1	Load Upper Unprotected Bounds
			0	1	0	0	0	0	0	0	LUB 2	Load Upper Unprotected Bounds
			0	1	1	0	0	0	0	0	LUB 3	Load Upper Unprotected Bounds
			1	0	0	0	0	0	0	0	LUB 4	Load Upper Unprotected Bounds
			1	0	1	0	0	0	0	0	LUB Q	Load Upper Unprotected Bounds
			1	1	0	0	0	0	0	0	LUB A	Load Upper Unprotected Bounds
			1	1	1	0	0	0	0	0	LUB I	Load Upper Unprotected Bounds
			0	0	1	0	0	0	0	1	LLB 1	Load Lower Unprotected Bounds
			0	1	0	0	0	0	0	1	LLB 2	Load Lower Unprotected Bounds
			0	1	1	0	0	0	0	1	LLB 3	Load Lower Unprotected Bounds
			1	0	0	0	0	0	0	1	LLB 4	Load Lower Unprotected Bounds
			1	0	1	0	0	0	0	1	LLB Q	Load Lower Unprotected Bounds
			1	1	0	0	0	0	0	1	LLB A	Load Lower Unprotected Bounds
			1	1	1	0	0	0	0	1	LLB I	Load Lower Unprotected Bounds
0	B	Δ	0	0	1	0	0	0	1	0	EMS 1	Execute Micro Sequence
			0	1	0	0	0	0	1	0	EMS 2	Execute Micro Sequence
			0	1	1	0	0	0	1	0	EMS 3	Execute Micro Sequence
			1	0	0	0	0	0	1	0	EMS 4	Execute Micro Sequence
			1	0	1	0	0	0	1	0	EMS Q	Execute Micro Sequence
			1	1	0	0	0	0	1	0	EMS A	Execute Micro Sequence
			1	1	1	0	0	0	1	0	EMS I	Execute Micro Sequence

Hex			Δ (Binary)								Mnemonic	Instruction
F	Fl	Δ	7	6	5	4	3	2	1	0		
0	B	Δ	0	0	1	0	0	0	1	1	RPR 1	Read Page Register (>64K MOS)
(cont'd)			0	1	0	0	0	0	1	1	RPR 2	Read Page Register (>64K MOS)
			0	1	1	0	0	0	1	1	RPR 3	Read Page Register (>64K MOS)
			1	0	0	0	0	0	1	1	RPR 4	Read Page Register (>64K MOS)
			1	0	1	0	0	0	1	1	RPR Q	Read Page Register (>64K MOS)
			1	1	0	0	0	0	1	1	RPR A	Read Page Register (>64K MOS)
			1	1	1	0	0	0	1	1	RPR I	Read Page Register (>64K MOS)
			0	0	1	0	0	1	0	0	WPR 1	Write Page Register (>64K MOS)
			0	1	0	0	0	1	0	0	WPR 2	Write Page Register (>64K MOS)
			0	1	1	0	0	1	0	0	WPR 3	Write Page Register (>64K MOS)
			1	0	0	0	0	1	0	0	WPR 4	Write Page Register (>64K MOS)
			1	0	1	0	0	1	0	0	WPR Q	Write Page Register (>64K MOS)
			1	1	0	0	0	1	0	0	WPR A	Write Page Register (>64K MOS)
			1	1	1	0	0	1	0	0	WPR I	Write Page Register (>64K MOS)
0	C	XX									ENQ	Enter Q
0	D	XX									INQ	Increase Q
0	E	XX									EXI	Exit Interrupt State
0	F	Δ	0	0	0	Shift						
			0	0	1	Count					QRS	Q Right Shift
			0	1	0						ARS	A Right Shift
			0	1	1						LRS	Long Right Shift (QA)
			1	0	0							
			1	0	1						QLS	Q Left Shift
			1	1	0						ALS	A Left Shift
			1	1	1						LLS	Long Left Shift (QA)
												NOTE: Right shift is sign-extended
												NOTE: Left shift is end-around
1	X	X	X								JMP	Conditional jump
2	X	X	X								MUI	Multiply Integer
3	X	X	X								DVI	Divide Integer
4	X	X	X								STQ	Store Q
5	X	X	X								RTJ	Return Jump
6	X	X	X								STA	Store A
7	X	X	X								SPA	Store A, Parity to A
8	X	X	X								ADD	Add to A
9	X	X	X								SUB	Subtract from A
A	X	X	X								AND	AND with A
B	X	X	X								EOR	Exclusive OR
C	X	X	X								LDA	Load A
D	X	X	X								RAO	Replace Add One in Storage
E	X	X	X								LDQ	Load Q
F	X	X	X								ADQ	Add to Q

APPENDIX E
TYPE 2 STORAGE REFERENCE INSTRUCTIONS

Hex First Word	Δ 7 6 5 4 3 2 1 0	Hex Second Word	Mnemonic	Description
0 4 0 0	0 0 0 0 0 0 0 0	X X X X	EIN	Enable Interrupts (Standard Instruction)
0 4 Δ	0 0 0	5 0 Δ	SJE	Subroutine/Jump Exit
	0 0 1		SJ1	Subroutine Jump
	0 1 0		SJ2	Subroutine Jump
	0 1 1		SJ3	Subroutine Jump
	1 0 0		SJ4	Subroutine Jump
	1 0 1		SJQ	Subroutine Jump
	1 1 0		SJA	Subroutine Jump
	1 1 1		SJI	Subroutine Jump
0 4 Δ	0 0 1	8 0 Δ	AR1	Add Register 1
	0 1 0		AR2	Add Register 2
	0 1 1		AR3	Add Register 3
	1 0 0		AR4	Add Register 4
	1 0 1		ARQ	Add Register Q
	1 1 0		ARA	Add Register A
	1 1 1		ARI	Add Register I
0 4 Δ	0 0 1	9 0 Δ	SB1	Subtract Register 1
	0 1 0		SB2	Subtract Register 2
	0 1 1		SB3	Subtract Register 3
	1 0 0		SB4	Subtract Register 4
	1 0 1		SBQ	Subtract Register Q
	1 1 0		SBA	Subtract Register A
	1 1 1		SBI	Subtract Register I
0 4 Δ	0 0 1	A 0 Δ	AN1	AND Register 1
	0 1 0		AN2	AND Register 2
	0 1 1		AN3	AND Register 3
	1 0 0		AN4	AND Register 4
	1 0 1		ANQ	AND Register Q
	1 1 0		ANA	AND Register A
	1 1 1		ANI	AND Register I
0 4 Δ	0 0 1	A 1 Δ	AM1	AND Memory
	0 1 0		AM2	AND Memory
	0 1 1		AM3	AND Memory
	1 0 0		AM4	AND Memory
	1 0 1		AMQ	AND Memory
	1 1 0		AMA	AND Memory
	1 1 1		AMI	AND Memory
0 4 Δ	0 0 1	C 0 Δ	LR1	Load Register 1
	0 1 0		LR2	Load Register 2
	0 1 1		LR3	Load Register 3
	1 0 0		LR4	Load Register 4
	1 0 1		LRQ	Load Register Q
	1 1 0		LRA	Load Register A
	1 1 1		LRI	Load Register I

Hex First Word	Δ 7 6 5 4 3 2 1 0	Hex Second Word	Mnemonic	Description
0 4 0 0	0 0 0 0 0 0 0 0	X X X X		
0 4 Δ	0 0 1	C 1 Δ	SR1	Store Register 1
	0 1 0		SR2	Store Register 2
	0 1 1		SR3	Store Register 3
	1 0 0		SR4	Store Register 4
	1 0 1		SRQ	Store Register Q
	1 1 0		SRA	Store Register A
	1 1 1		SRI	Store Register I
0 4 Δ		C 2 Δ	LCA	Load Character to A
0 4 Δ		C 3 Δ	SCA	Store Character from A
0 4 Δ	0 0 1	D 0 Δ	OR1	OR Register 1
	0 1 0		OR2	OR Register 2
	0 1 1		OR3	OR Register 3
	1 0 0		OR4	OR Register 4
	1 0 1		ORQ	OR Register Q
	1 1 0		ORA	OR Register A
	1 1 1		ORI	OR Register I
0 4 Δ	0 0 1	D 1 Δ	OM1	OR Memory
	0 1 0		OM2	OR Memory
	0 1 1		OM3	OR Memory
	1 0 0		OM4	OR Memory
	1 0 1		OMQ	OR Memory
	1 1 0		OMA	OR Memory
	1 1 1		OMI	OR Memory
0 4 Δ	0 0 1	E 0 Δ	C1E	Compare Register 1 Equal
	0 1 0		C2E	Compare Register 2 Equal
	0 1 1		C3E	Compare Register 3 Equal
	1 0 0		C4E	Compare Register 4 Equal
	1 0 1		CQE	Compare Register Q Equal
	1 1 0		CAE	Compare Register A Equal
	1 1 1		CIE	Compare Register I Equal
0 4 Δ		E 2 Δ	CCE	Compare Character Equal

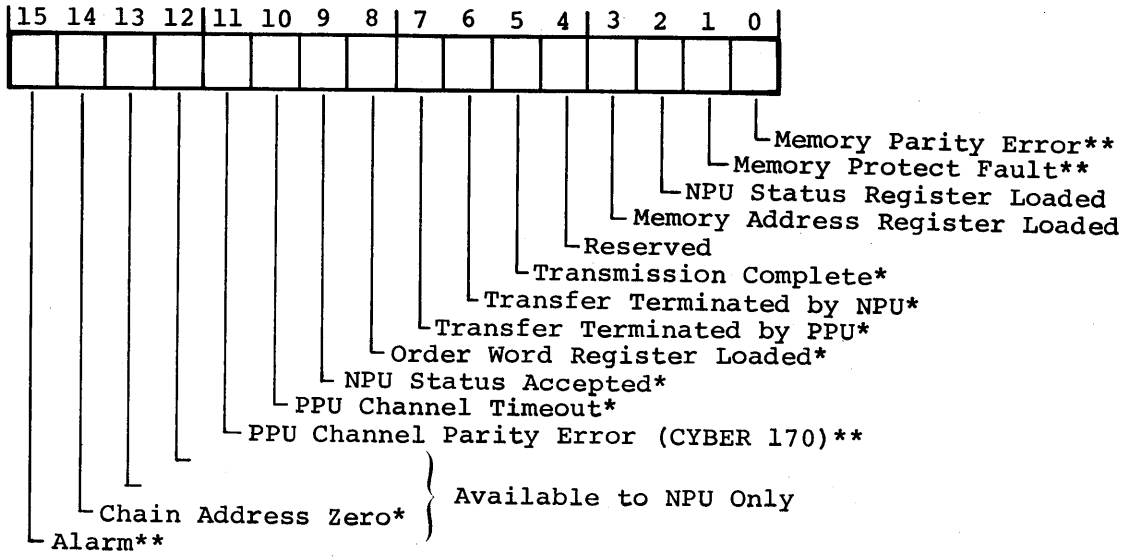
APPENDIX F
I/O PROGRAMMING FORMATS

COMMUNICATIONS COUPLER PPU FUNCTION CODES

11 10 9	8 7 6	5	4	3	2 1 0	
Equip Code (=7)	1 0 0	0	0	X	X X X	Clear Coupler
	0 1 0	0		X	X X X	Clear CP
	0 0 1	0		X	X X X	Stop CP
	0 0 0	1		X	X X X	Start CP
	X X X	X		0	0 0 0	Input Memory Address Zero*
	X X X	X		0	0 0 1	Input Memory Address One*
	X X X	X		0	0 1 0	
	X X X	X		0	0 1 1	Input Data
	X X X	X		0	1 0 0	Input CP Status
	X X X	X		0	1 0 1	Input Coupler Status
	X X X	X		0	1 1 0	Input Order Word*
	X X X	X		0	1 1 1	Input Program
	X X X	X		1	0 0 0	Output Memory Address Zero (upper byte)
	X X X	X		1	0 0 1	Output Memory Address One (lower byte)
	X X X	X		1	0 1 0	
	X X X	X		1	0 1 1	
	X X X	X		1	1 0 0	Output Data
	X X X	X		1	1 0 1	Output Program
	X X X	X		1	1 1 0	Output Order Word
	X X X	X		1	1 1 1	

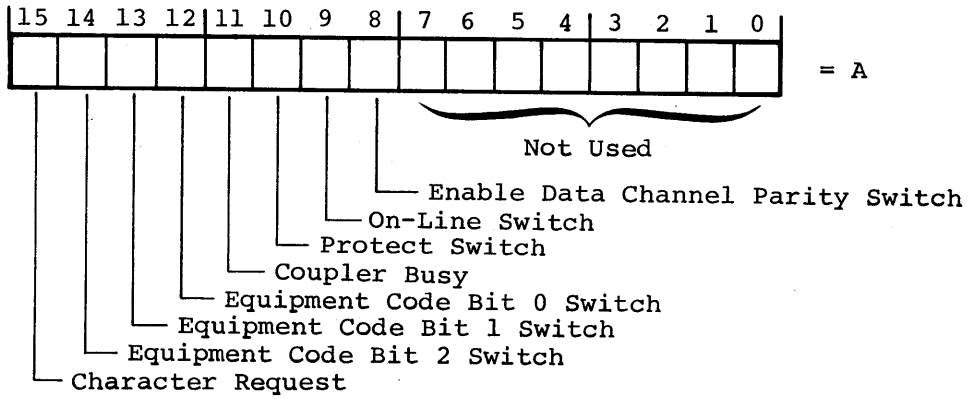
*Hardware Maintenance feature

COMMUNICATIONS COUPLER STATUS FORMAT

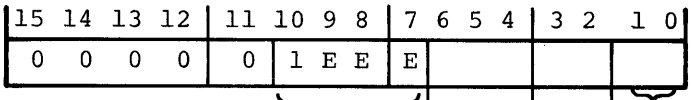


**Alarm Condition (all alarms generate NPU Interrupt)
*NPU Interrupt Condition

COMMUNICATIONS COUPLER INPUT SWITCH STATUS FORMAT



COMMUNICATIONS COUPLER - CP SET/SAMPLE INSTRUCTION FORMAT



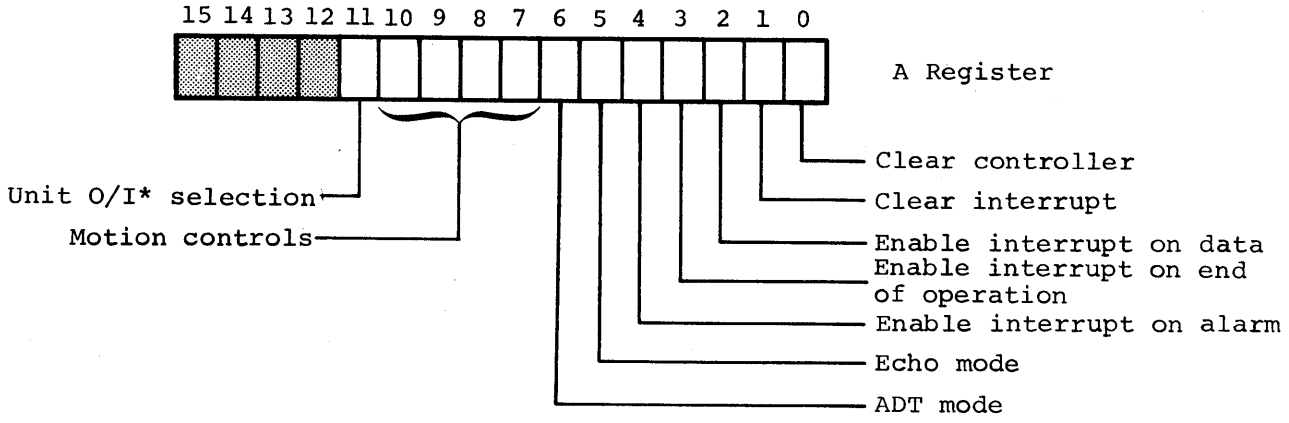
Equipment Address

Not Used

Sample (Read)	First Coupler = 1100 ('C') (Standard) Second Coupler = 1101 ('D') (Optional)	0 0 0	0 0	Input Memory Address Zero*
		0 0 1	0 0	Input Memory Address
		0 1 0	0 0	
		0 1 1	0 0	Input First/Present Character Displacement
		1 0 0	0 0	Input CP Status*
		1 0 1	0 0	Input Coupler Status
		1 1 0	0 0	Input Order Word
		1 1 1	0 0	Input IO*
		0 0 0	0 1	Input Last Word From Data Channel*
		0 0 1	0 1	Input FDMAR0/FDMAR1*
		0 1 0	0 1	Input FDMAR0/Flag Mux*
		0 1 1	0 1	Input FDMAR1/Flag Mux/Flag Register*
		1 0 0	0 1	
		1 0 1	0 1	Input Switch Status
		1 1 0	0 1	
		1 1 1	0 1	Input Character*
Set (Write)		0 0 0	1 0	Output Memory Address Zero*
		0 0 1	1 0	
		0 1 0	1 0	
		0 1 1	1 0	Output FCD, PCD, LCD*
		1 0 0	1 0	Output CP Status
		1 0 1	1 0	Output Buffer Length
		1 1 0	1 0	Output Order Word*
		1 1 1	1 0	
		0 0 0	1 1	Clear Coupler
		0 0 1	1 1	Terminate Transfer
		0 1 0	1 1	
		0 1 1	1 1	
		1 0 0	1 1	Output Test*
		1 0 1	1 1	Input Test*
		1 1 0	1 1	Output Memory Address
		1 1 1	1 1	Output Character*

*Hardware maintenance feature

TAPE CASSETTE CONTROLLER - CONTROL FUNCTIONS FORMAT

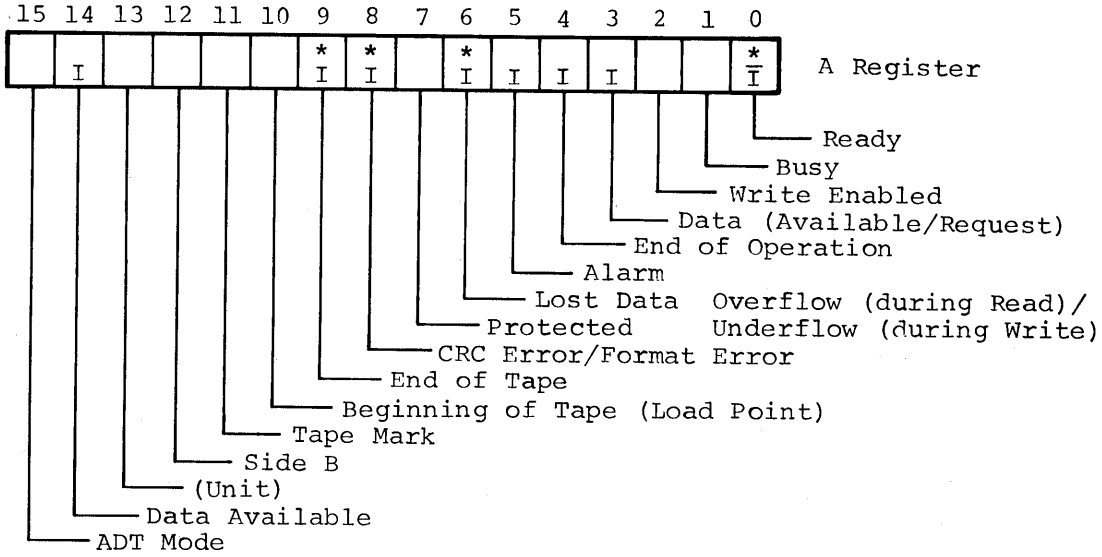


NOTE: *A manual switch on the controller card can override this function bit and force Unit 1 to always be selected. Standard 2550 systems always have this switch on.

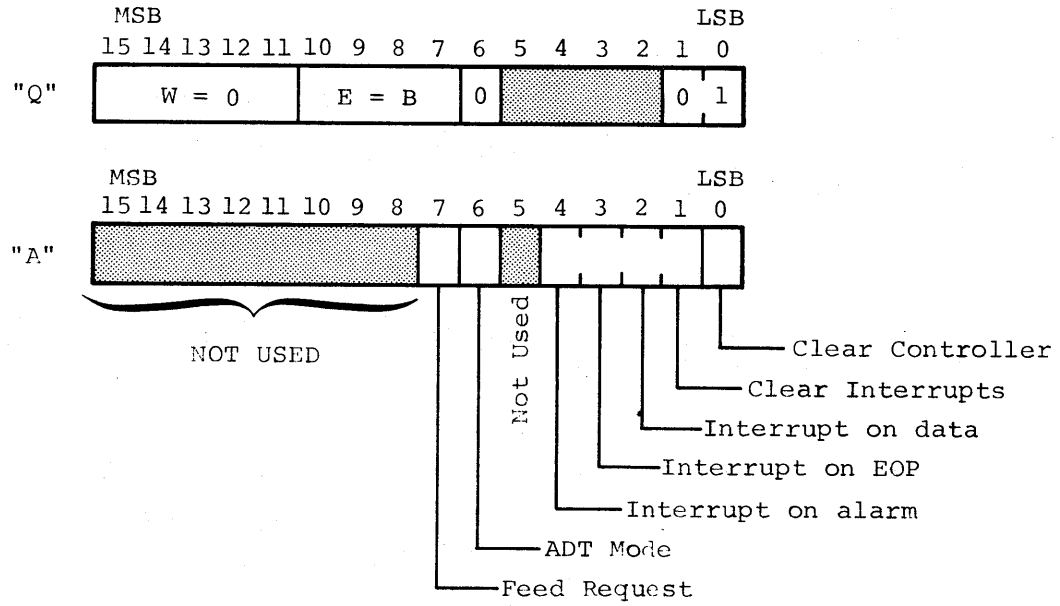
TAPE CASSETTE CONTROLLER - MOTION CONTROLS

A Register Bits				
10	9	8	7	Function
1	0	0	0	Search Tape Mark (Reverse)
1	0	0	1	Search Tape Mark (Forward)
1	0	1	0	Write Tape Mark
1	0	1	1	Write One Record
1	1	0	0	Backspace One Record (or Tape Mark)
1	1	0	1	Rewind
1	1	1	0	Erase
1	1	1	1	Read One Record

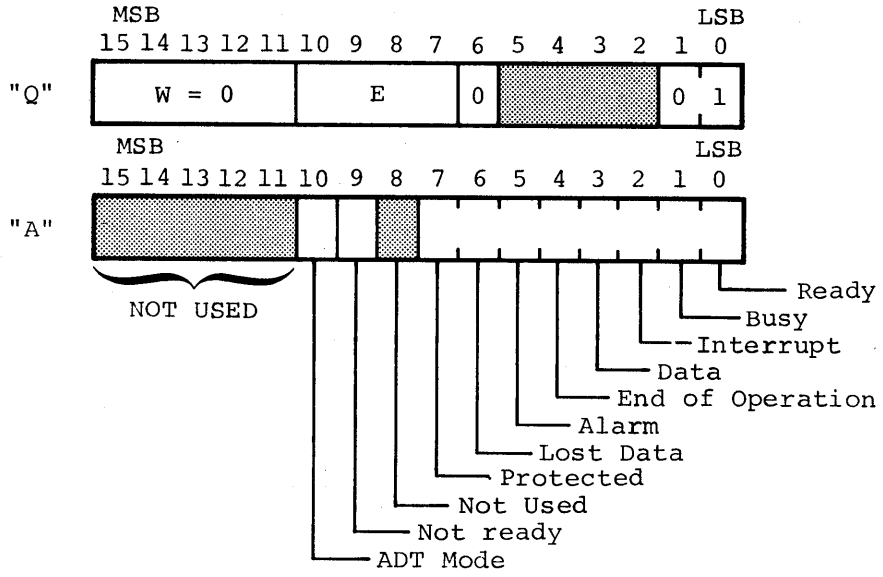
TAPE CASSETTE CONTROLLER - STATUS RESPONSE FORMAT



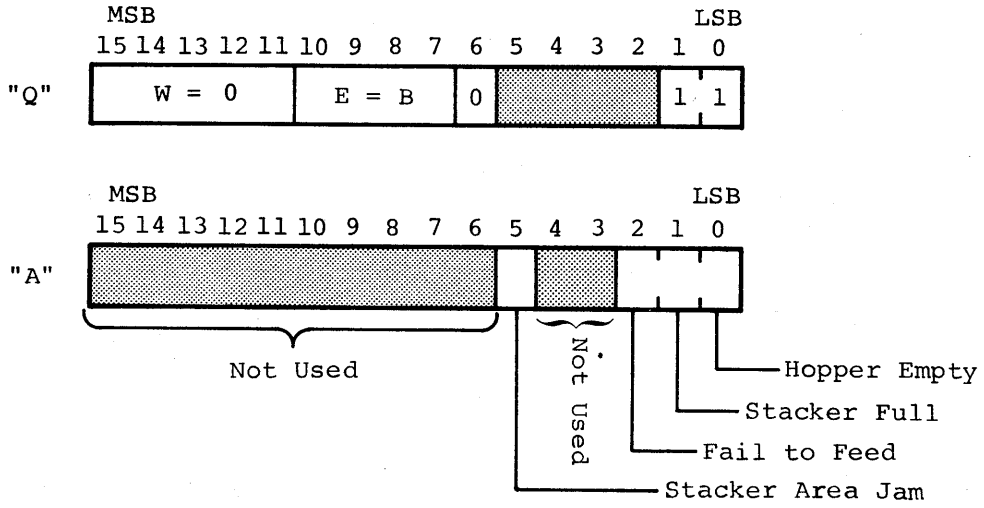
PERIPHERAL (CARD READER) CONTROLLER - DIRECTOR FUNCTION FORMAT



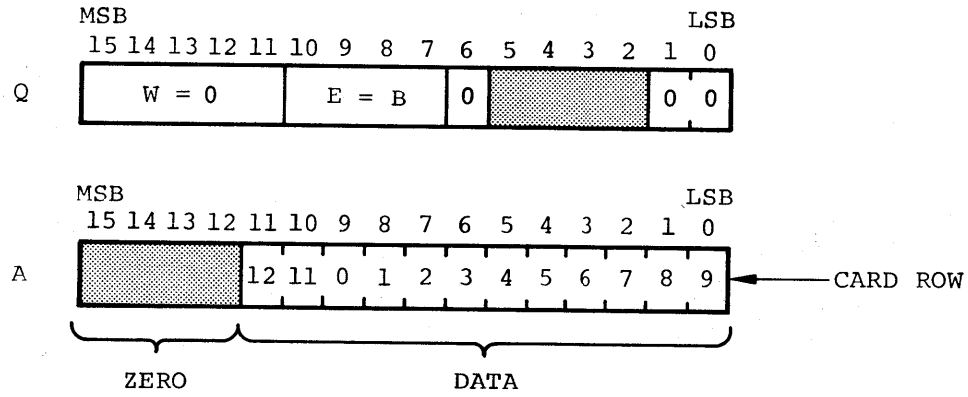
PERIPHERAL (CARD READER) CONTROLLER - DIRECTOR STATUS 1 FORMAT



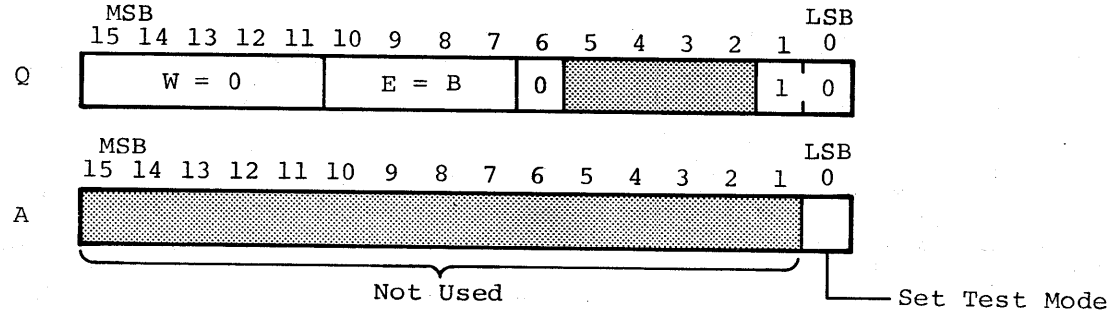
PERIPHERAL (CARD READER) CONTROLLER - DIRECTOR STATUS 2 FORMAT



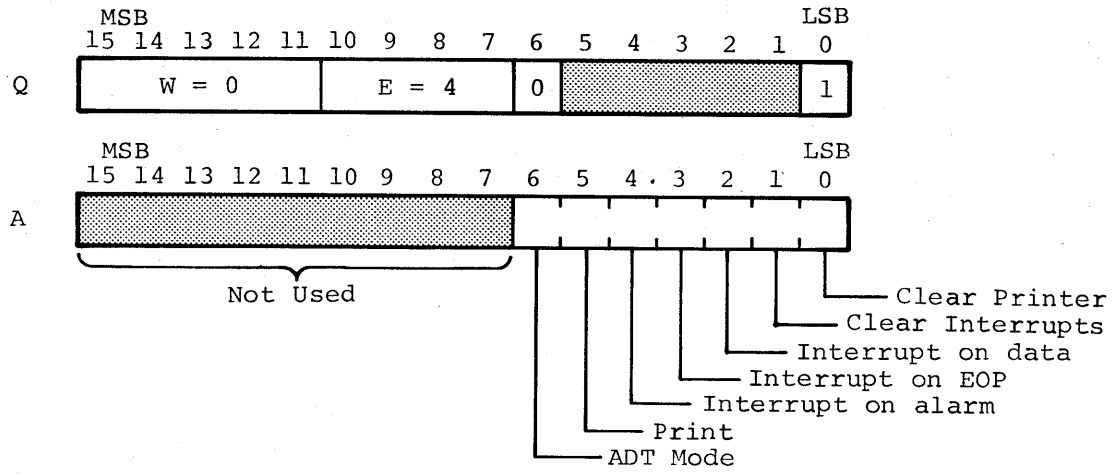
PERIPHERAL (CARD READER) CONTROLLER - DATA TRANSFER COMMAND FORMAT



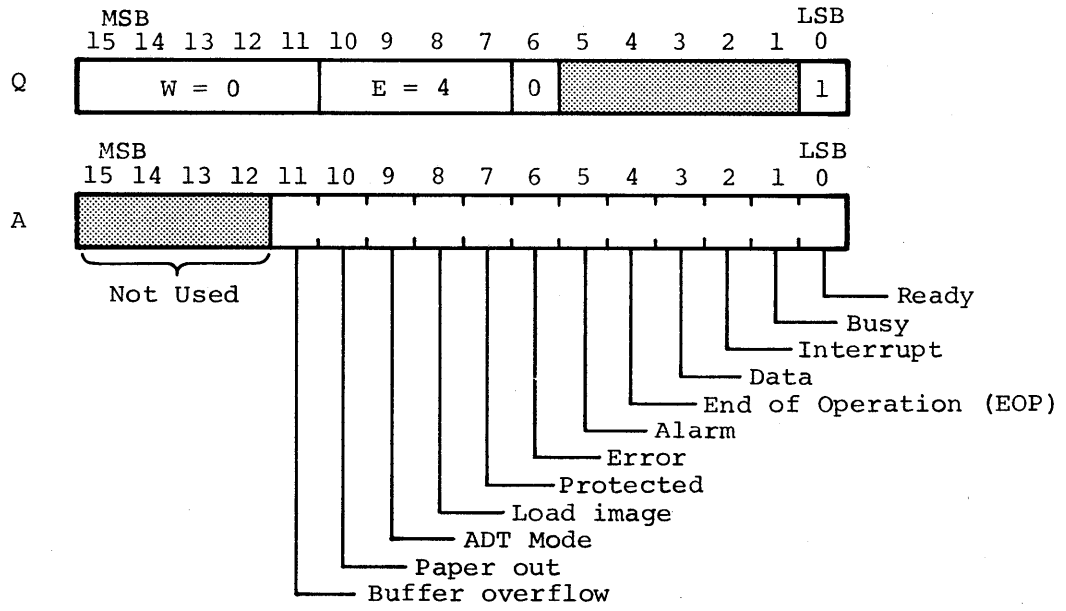
PERIPHERAL CONTROLLER - SELF-TEST COMMAND FORMAT



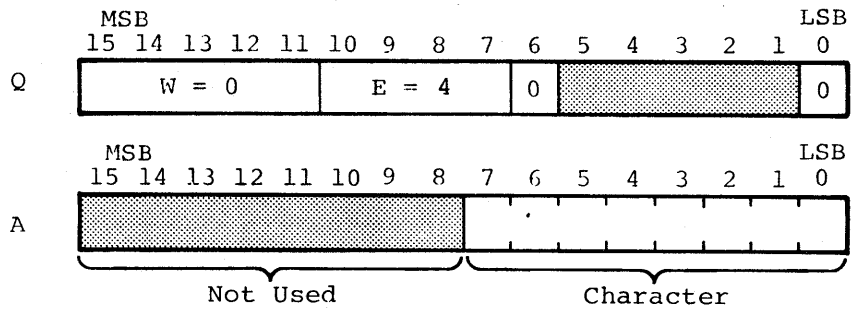
PERIPHERAL (LINE PRINTER) CONTROLLER - DIRECTOR FUNCTION FORMAT



PERIPHERAL (LINE PRINTER) CONTROLLER - DIRECTOR STATUS RESPONSE FORMAT



PERIPHERAL (LINE PRINTER) CONTROLLER - DATA TRANSFER COMMAND FORMAT

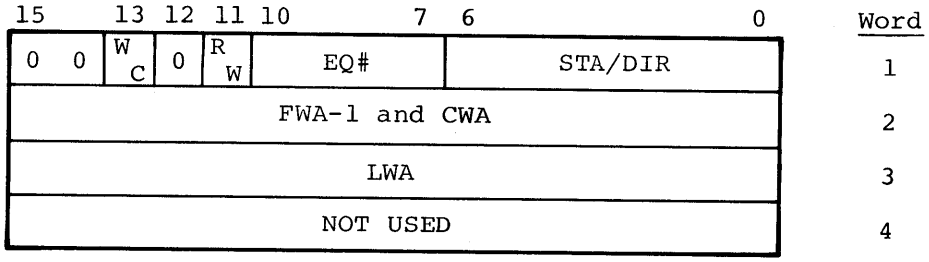


MULTIPLEX LOOP INTERFACE ADAPTER - READ STATUS BITS

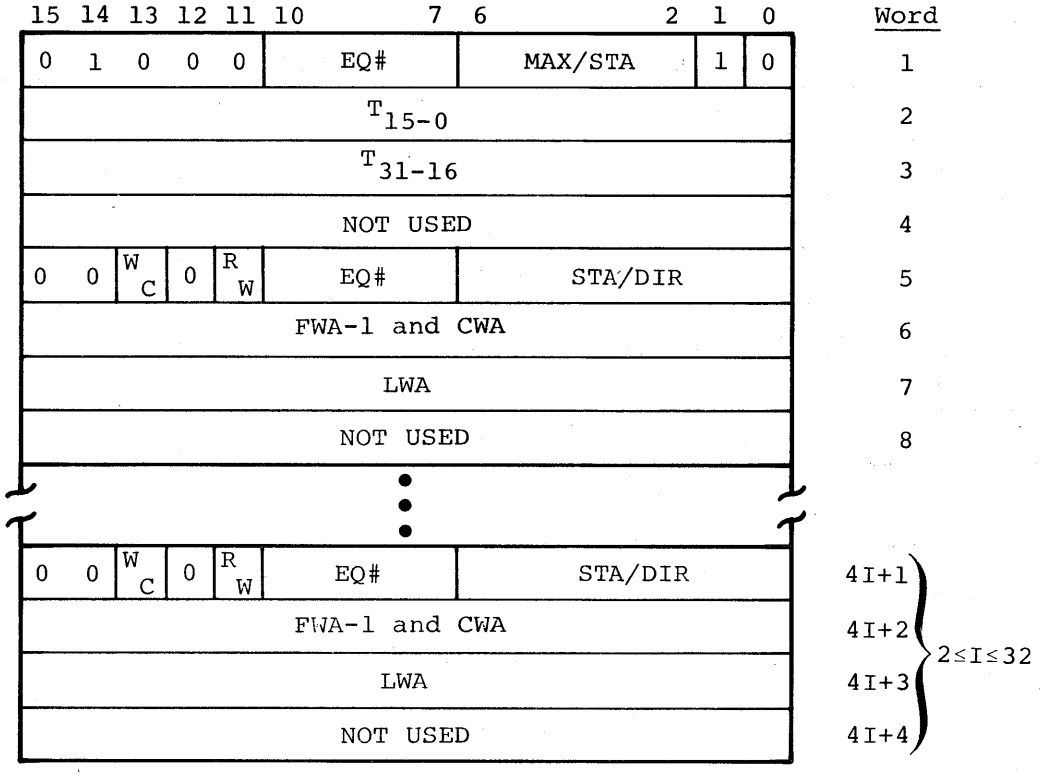
Bit	Function	Error	Interrupt	Reset
S0	Ready	E	I	
S1	(Busy)			
S2	Error Interrupt		I	
S3	Normal and No Diagnostic			
S4	(End of Operation)			
S5	Input Buffer Full	E	I	R
S6	Output Loop Error	E	I	R
S7	Protected			
S8	Parity Error	(E)	(I)	(R)
S9	Input Loop Error	E	I	R
S10				
S11				
S12				
S13	(Reserved)			
S14	(Reserved)			
S15	(Reserved)			

APPENDIX G
AUTO-DATA TRANSFER (ADT) MODE FORMATS

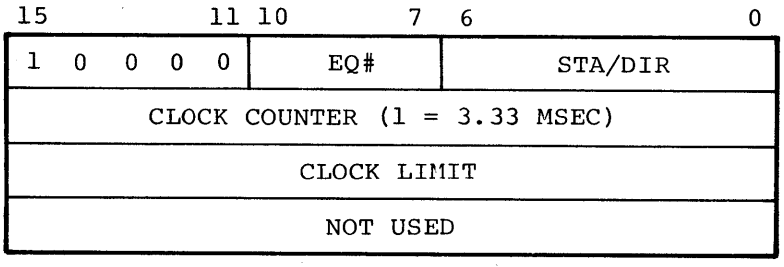
Single AQ Device



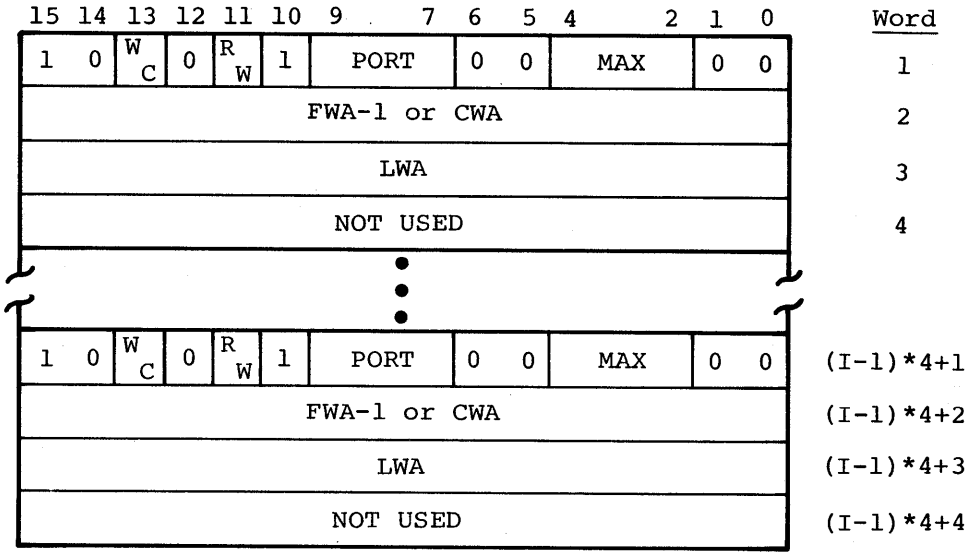
Multiple AQ Devices



ADT Table for Clock



SINGLE OR MULTIPLE SET/SAMPLE DEVICE



APPENDIX H
STATUS MODE BIT ASSIGNMENTS

Bit	Function
SM100	Not used (double precision)
SM101	One's Complement Arithmetic
SM102	Enable Bit Generator Input from N Register
SM103	Not used (split adder)
SM104	Macro Breakpoint Occurred (use with INT31/)
SM105	1700 Protect Fault Occurred
SM106	Enable Macro Interrupt System
SM107	Not used (enable decimal arithmetic)
SM108	Main Memory Parity Error Occurred
SM109	Enable Micromemory Halt
SM110	Overflow
SM111	Enable File 1 (Read/Write)
SM112	Not used (enable binary overflow)
SM113	Not used (enable R/W MM via transform)
SM114	Delay Enabling Macro Interrupts
SM115	MLIA Busy
SM200	Enable ADT Mode
SM201	Strobe or Read Data from MO5 Device
SM202	Write Data to AQ Device
SM203	Terminate I/O Transfer
SM204	Enable Autoload
SM205	Enable MUX Subsystem Priority 3 Interrupt
SM206	Not used
SM207	Enable page selection with transform
SM208	Not used
SM209	Enable the unprotected instruction followed by a Protected Instruction Check
SM210	Write Data to Panel Device
SM211	Read Data to Panel Device
SM212	Write Data to FCR Register
SM213	Enable 1700 Enhanced Transform
SM214	Enable Console Control
SM215	Enable Macro Instruction Run

APPENDIX I
INTERRUPT BIT ASSIGNMENTS

Bit	Function
I100	MLIA (Output Data Demand)
I101	Communications Console (TTY or CRT)
I102	MLIA (Line Frame)
I103	(unassigned)
I104	2571/2570 Line Printer Controller
I105	(unassigned)
I106	(unassigned)
I107	Tape Cassette Controller
I108	Real-Time Clock
I109	(unassigned)
I110	(unassigned)
I111	2571-2572 Card Reader Controller
I112	1700 Emulator
I113	1700 Emulator
I114	1700 Emulator
I115	1700 Emulator
I200	Power Fail/Memory Parity
I201	Communications Console (TTY or CRT)
I202	MLIA (Error)
I203	Multiplex Subsystem Priority 3 (source is SM205)
I204	2571/2570 Line Printer Controller
I205	CYBER Coupler 2 (2558-1)
I206	CYBER Coupler
I207	Tape Cassette Controller
I208	Real-Time Clock
I209	1732/608-609 Mag Tape Controller (QSE)
I210	1740-1742 Line Printer Controller (QSE)
I211	2571-2572 Card Reader Controller
I212	MLIA (Same as I100)
I213	MLIA (Same as I102)
I214	(unassigned)
I215	Macro Breakpoint

APPENDIX J
INTERRUPT AND EQUIPMENT CODE ASSIGNMENTS

Device	CP Card Cage Slot	Macro Interrupt	Micro Interrupt	Equipment Code (Hex)
Primary Communications Coupler*	B,C,D	6	-	C
Multiplex Loop Interface Adapter*	E,F,G	2,12,13	0,2	A
Tape Cassette Controller	H	7	7	7
2571 Peripheral Controller (2572 Card Reader)	J	11	11	B
2571 Peripheral Controller (2570 Line Printer)	J	4	4	4
Real Time Clock	-	8	8	1
I/O TTY Interface	K	1	1	1
Status Mode Interrupt	L	3	—	—
Power Fail/Memory Parity	-	0	-	-
2558-1 Expansion Communications Coupler*	A,AA,AB	5	-	D
1732/608-609 Mag Tape Controller**	External	9	-	9
1740-1742 Line Printer Controller**	External	10	-	8
1700 Emulator	-	-	12,13,14, 15	-
Unassigned	-	14,15	3,4,6,9,10	0,2,3,5, 6,E,F

*MO5 Interface; all others are A/Q interface

**QSE - nonstandard controller.

INDEX

COMMENT SHEET

MANUAL TITLE 2550-2 (MOS) Host Communications Processor Hardware Reference Manual

PUBLICATION NO. 74375500 REVISION A

FROM: NAME: _____

BUSINESS ADDRESS: _____

_____

COMMENTS:

CUT ALONG DOTTED LINE

FOLD ON DOTTED LINES AND STAPLE

STAPLE

STAPLE

FOLD

FOLD

PLACE
STAMP
HERE

CUT ALONG LINE

CONTROL DATA CORPORATION

Technical Publications Department
3519 W. Warner Avenue
P. O. Box 5007
Santa Ana, California 92704

FOLD

FOLD