# CONTROL DATA® SC-17 system controller

CONTROL DATA
CORPORATION

Reference Manual

*Preliminary Edition*

# CONTROL DATA® SC-17 system controller

Information presented in this edition is of a preliminary nature and is subject to change. Any corrections necessitated by design changes and/or product improvement will be handled by standard manual revision procedures. Errors and suggestions should be communicated to Technical Publications Department, 1700/3000 Development Division, ARHOPS.

CONTROL DATA
CORPORATION

Reference Manual

*Preliminary Edition*

# RECORD of REVISIONS

| REVISION | NOTES |
|---|---|
| 01 | Preliminary edition. |
| (10-1-69) | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |

Address comments concerning this
manual to:

Control Data Corporation
Technical Publications Department
4201 North Lexington Avenue
St. Paul, Minnesota 55112

or use Comment Sheet in the back of
this manual.

Pub No. 60270600

Printed in United States of America

# CONTENTS

## FIGURES

## TABLES

# BASIC SYSTEM DESCRIPTION 1

## INTRODUCTION

The CONTROL DATA® System Controller is a stored program, digital computer. It is physically small due to integrated logic. It is designed for high computation and input/output (I/O) speed. The program protection features and high reliability under a wide range of environmental conditions make the System Controller suitable for real-time, on-line, or control applications. The design of the System Controller is modular so that a large variety of configurations are available. The various modular units are listed below.

1. Compute Module: Includes 16-bit 1's complement arithmetic, multiply/divide, breakpoint, auto load, two hardware index registers, two interrupts and program protect. Space provided within module to accommodate 32K memory drive, A/Q receiver/ transmitters, DSA receiver/transmitters, console drive, drive for 1 each TTYI or TTYII and character handling.

2. Magnetic Core Memory Module: 4096 18-bit words, including Parity checking, memory protection and read/write control; expandable in 4096-word increments to 16, 384 words. Connects directly to Compute Module. Two units required to expand to 32K.

3. A/Q Interrupt Data Channel: Provides 14 additional interrupts and implements direct data transfers through the A/Q registers. Connects directly to Compute Module.

4. Direct Storage Access Channel: Provides a direct storage access channel to memory for buffer channels. Connects directly to Compute Module.

5. Storage Increments: Provides 4096 words of magnetic core memory increments to expand Magnetic Core Memory Module.

6. Console: Provides manual register entry and display, breakpoint, auto load, and selective condition switches. Connects directly to Compute Module

7. TTY 1: 35 KSR 100 wpm with control including keyboard, printer, and connections to Compute Module. ASCII '68 "64 Character" Version.

8. TTY 2: 35 ASR 100 wpm with control and remote mode selection including keyboard, printer, paper tape reader, paper tape punch and connections to Compute Module. ASCII '68 "64 Character" Version.

9. Character Handling: Provides character addressing of memory on load and store from the lower character position of the "A" register. Connects directly to Compute Module.

10. Cabinetry: Provides cabinet and cooling blower for one Compute Module and/or one or two Magnetic Core Memory Modules.

11. Power: Provides sufficient power for one Compute Module and Magnetic Core Memory Modules expanded to 32, 768 words.

## SYSTEM CHARACTERISTICS

Figure 1-1 shows a typical system configuration. The input/output for this system is handled through a teletypewriter and a paper tape station. The teletypewriter connects directly to the System Controller; however, the A/Q Interrupt Data Channel is required for the paper tape station and all other peripheral devices. Refer to Table 1-1 for further characteristics of this typical system configuration.

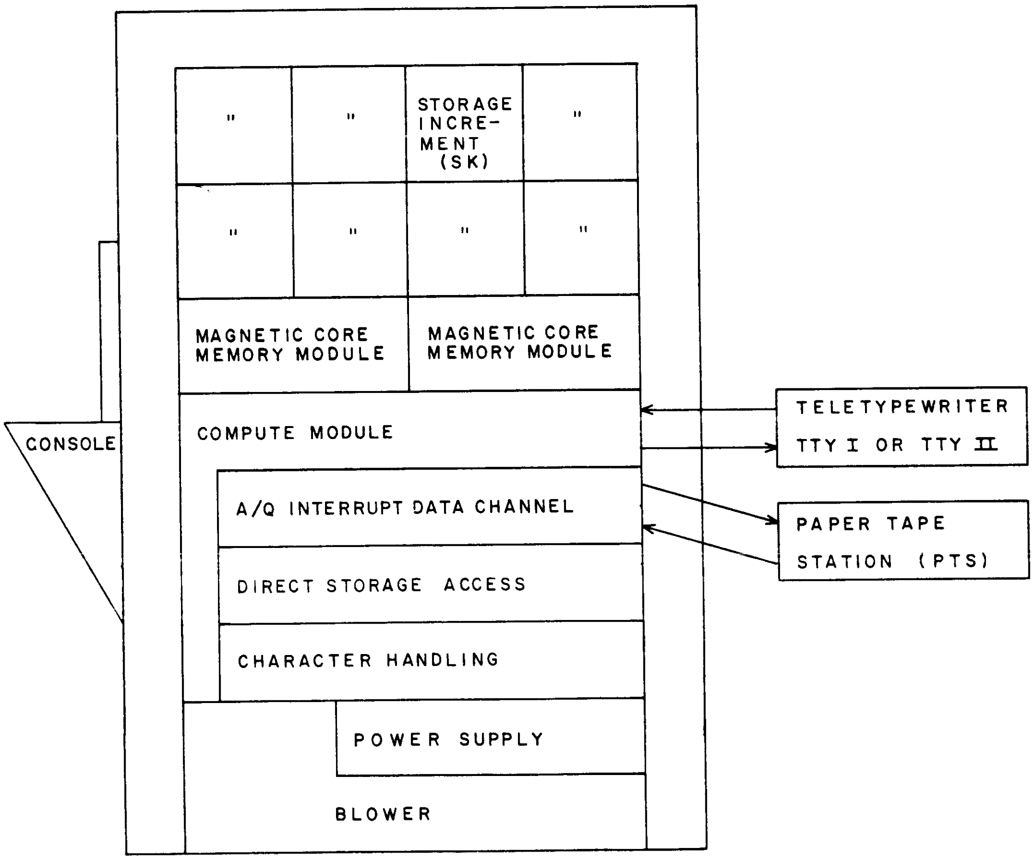


Figure 1-1. Typical System Controller Configuration

TABLE 1-1. SYSTEM CONTROLLER CHARACTERISTICS

Stored Program, Digital Computer

Integrated Logic

Parallel mode of operation

18-bit storage word
- 16 data bits
- 1 parity bit
- 1 program protect bit

16-bit instruction word

Two 16-bit index registers

Multilevel indirect addressing

Character Addressing

Magnetic core storage
- 4,096 18-bit words expandable to 32,768 words.

Input/Output transmission of 16-bit words or 8-bit characters.

Auto-restart

Breakpoint

Auto Load

Variable test mode

Environment:
- 40° F to 120° F
- Relative humidity 0% to 80%

Cooling:
- Forced air

System interrupt

Flexible repertoire of inst.
- Arithmetic operations
- Logical and masking operations
- Interregister transfers

Number System: Hexadecimal (base 16)

Intercomputer communications:
- S.C. to S.C.
- Satellite operations

Console includes:
- Register contents display
- Operating switches and indicators

Memory cycle time: 1.5 μsec.

## EXPANDABLE INPUT/OUTPUT CAPABILITIES

With the addition of the AQ Interrupt Data Channel and the Direct Storage Access, there is a wide range of peripheral equipment which can be added to the System Controller. See publication number 60182700, Standard Peripheral Reference Manual, for descriptions of peripherals available to the System Controller.

# STORAGE

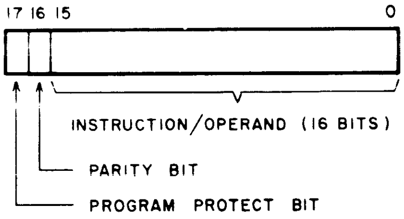
The Magnetic Core Memory Module provides high-speed, random-access magnetic core storage for 4, 096 18-bit words. The storage capacity may be expanded from 4K to 32K in standard 4K increments.

Storage cycle time is 1.5 microseconds. This is defined as the shortest possible time between successive Read/Write operations in storage.

## STORAGE WORD

A storage word may be a 16-bit instruction, a 16-bit operand, two 8-bit characters, * or one-half of a 32-bit operand (multiply or divide). A parity bit and a program protect bit are appended to each 16-bit storage word; thus, a storage word is 18 bits long.

Format:



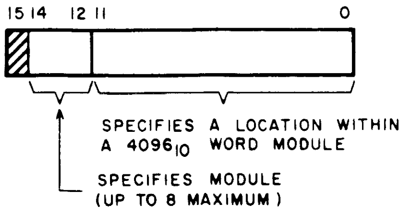
Bit 16 is the parity bit. It takes on a value so that the total number of "1" bits is odd (total number of bits includes the program protect bit). For example, if all 16 data bits are "1's" and the program protect bit is "0", the parity bit is a "1".

Bit 17 is the program protect bit. If it is a "1", it indicates the word is part of a protected program.

## STORAGE ADDRESSING

The location of each word in storage is identified by an assigned number (address). An address consists of 15 bits. During Character Addressing (if this option is installed), bit 0 of the I register is also considered part of the address since it designates which 8-bit character of the memory location is to be referenced.

Format:


15 14 12 11
0
SPECIFIES A LOCATION WITHIN A $4096_{10}$ WORD MODULE
SPECIFIES MODULE (UP TO 8 MAXIMUM)

Bits 00 through 11 specify a location within a 4, $096_{10}$-word storage increment (module). Bits 12 through 14 specify one of up to eight storage increments (modules).

Sometimes a condition arises in which the program references an address in a nonexistent storage module. In this case, the computer references the address specified by bits 00 through 11 in some existing storage module (storage addressing wrap-around). Table 2-1 lists the actual 1700 storage size, the storage module addressed, and the effective module addressed.

For example, if the computer has 16K ($16,384_{10}$) words of storage, the highest permissible address is $3FFF_{16}$. If the program attempts to address location $5040_{16}$ (located in a nonexistent storage module 5), it actually references location $1040_{16}$ in module 1.

TABLE 2-1. STORAGE MODULE ADDRESSING RELATIONSHIPS

| | STORAGE MODULE ADDRESSED | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Storage Size | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 4K | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8K | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| 12K | 0 | 1 | 2 | 2 | 0 | 1 | 2 | 2 |
| 16K | 0 | 1 | 2 | 3 | 0 | 1 | 2 | 3 |
| 20K | 0 | 1 | 2 | 3 | 4 | 4 | 4 | 4 |
| 24K | 0 | 1 | 2 | 3 | 4 | 5 | 4 | 5 |
| 28K | | 1 | 2 | 3 | 4 | 5 | 6 | 6 |
| 32K | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

Effective Module Addressed

## STORAGE REGISTERS

### Z Register

The Z register is the 18-bit data storage register. It transfers data between the computer, external storage access, and magnetic core storage. Each time a word is read from storage, it is transferred to the Z register, and parity is checked. Word parity is also generated when the data is in the Z register prior to writing the word into storage. See Section 4 for a detailed description of storage parity errors.

### S/Bank Select Register

This register is the 15-bit address storage register. It holds the address specifying where a word is to be read from or written into storage.

## Storage Accesses

There are two accesses to the computer storage. One is internal from the computer itself and is included in the basic System Controller. The computer uses the internal access for all computation operations which reference storage.

The second access to memory is the external access. The external access is included in the Interrupt Data and the Direct Storage Access Channel combination. This access is via the terminated twisted-pair, transmission line technique used in the I/O of the 3000 Series computers.* The cables, connectors, signal levels, and termination are identical to the 3000 Series I/O.**

*Maximum total cable length from the external storage access is 80 feet.

**The 3000 Series controllers, synchronizers, etc. are not compatible with the System Controller.

# CENTRAL COMPUTER 3

The System Controller performs calculations and processes data in a parallel, binary mode through the step-by-step execution of individual instructions. The instructions and data are stored in the magnetic core storage of the computer system.

Functionally, the computer may be divided into an arithmetic section and a control section.

## ARITHMETIC SECTION

The arithmetic section performs the arithmetic and logical operations necessary for executing instructions. It consists primarily of several operational registers. In all discussions of registers, the rightmost bit is the least significant and is defined as bit 00. Figure 3-1 is a block diagram of the computer.

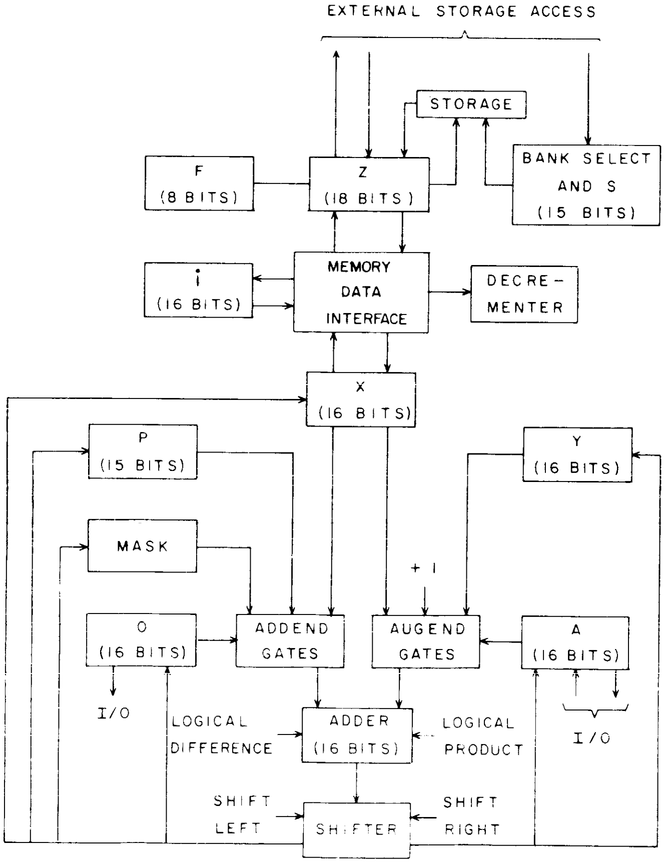


Figure 3-1. Block Diagram: System Controller

### A Register

The A register is the principal arithmetic register; it contains 16 bits (15 through 00) of which bit 15 is the sign bit. The A register also serves as the data interface during I/O operations.

### Decrementer

The 5-bit decrementer is a counter which keeps track of the number of shifts to be made during iterative shift operations (DIV, MVI, SHIFT).

### F Register

The F register is a function register containing 8 bits (07 through 00). This register holds the instruction identification and/or addressing mode bits during the execution of an instruction.

### i Register

The i register is an indexing register*. It is used for instructions requiring indexing. Bit 0 of this register is also used during character addressing as the character designation bit (CDB).

### M Register

The M register is the Mask register. It is used in the interrupt system and is described in detail in Section 4.

### P Register

The 15-bit P register functions as a program address counter. The P register holds the address of each instruction. After executing the instruction at address P, P is advanced to the address of the next instruction. The amount by which P is advanced is determined by the type of instruction being executed.

### Q Register

The Q register is an auxiliary arithmetic register containing 16 bits (15 through 00). This register is also used as an index register for instructions requiring indexing. The Q register also holds the address of a peripheral device during I/O operations. During Set/Clear protect bit instructions, the Q register holds the affected address.

### X Register

The X register is an exchange register containing 16 bits (15 through 00). This register holds data going to or from storage. It also holds one of the parameters in most arithmetic operations.

### Y Register

The Y register is an address register containing 16 bits (15 through 00). In this register, storage addresses are formed and held for transfer during a storage reference.

## CONTROL SECTION

The control section of the computer directs the operations required to execute instructions and establishes the timing relationships needed to perform these operations in the proper sequence. It also controls interrupt processing, program protection, and operations involving I/O and storage.

The control section acquires an instruction from storage, interprets it, and sends the necessary commands to other sections. The program address counter, P, provides program continuity by generating in sequence the storage addresses which contain the individual instructions. Usually, at the completion of each instruction, the count in P is advanced by one to specify the address of the next instruction in the program.

## INSTRUCTION FORMATS

There are three types of instructions in the System Controller: storage reference, register reference, and skip.

- Storage reference instructions reference storage for operands.
- Register reference instructions operate on the computer registers or control logic.
- Skip instructions sense the existence of specific conditions within the computer.

* Access to the i register is obtained by addressing memory location OOFF which is reserved for the i register.

Hexadecimal notation is used in this computer for ease in expressing the 4-bit groups which occur in the instruction format. Hexadecimal is base 16 and requires the additional characters A, B, C, D, E, and F. The relationships between binary, decimal, and hexadecimal are shown in Table 3-1.

TABLE 3-1. BINARY-DECIMAL-HEXADECIMAL RELATIONSHIPS

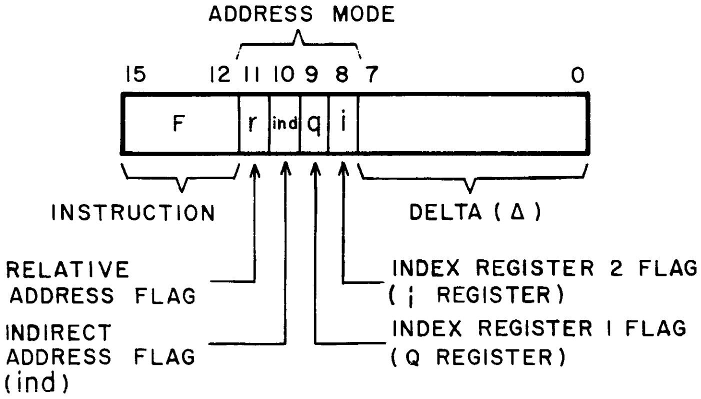
| DECIMAL | HEXADECIMAL | BINARY | DECIMAL | HEXADECIMAL | BINARY |
|---|---|---|---|---|---|
| 0 | 0 | 0000 | 8 | 8 | 1000 |
| 1 | 1 | 0001 | 9 | 9 | 1001 |
| 2 | 2 | 0010 | 10 | A | 1010 |
| 3 | 3 | 0011 | 11 | B | 1011 |
| 4 | 4 | 0100 | 12 | C | 1100 |
| 5 | 5 | 0101 | 13 | D | 1101 |
| 6 | 6 | 0110 | 14 | E | 1110 |
| 7 | 7 | 0111 | 15 | F | 1111 |

## Storage Reference Instructions

The storage reference instructions contain three fields: instruction, address mode, and delta.

The instruction field contains the 4-bit operation code, F. The address mode contains 4 flags for indexing, indirect addressing, and relative addressing. The sign bit of delta is extended in all cases except those noted.

Format:


ADDRESS MODE
15
12
11
10
9
8
7
0
F
r
ind
q
i
INSTRUCTION
DELTA ( Δ )
RELATIVE ADDRESS FLAG
INDIRECT ADDRESS FLAG (ind)
INDEX REGISTER 2 FLAG ( i REGISTER)
INDEX REGISTER 1 FLAG (Q REGISTER)

The following definitions apply to the description of addressing modes.

- Instruction Address: the address of the instruction being executed, also called P.
- Indirect Address: a storage address which contains an address rather than an operand.
- Base Address: the operand address after all indirect addressing but before modification by index registers. The base address is the effective address if no indexing is specified.
- Effective Address: the final address of the operand. In certain cases, the effective address equals the operand for read-operand-type instructions. These cases are noted in Table 3-2.
- Indexing: The computer has two index registers. Index register number 1 is the Q register, and index register number 2 is the i register. The base address may be modified by either one 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 the i register are added to the base address to form the effective address. If both index register flags are set, the contents of Q and the contents of i are added to the base address to form the effective address. Indexing occurs after indirect addressing has been completed.

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

Storage reference instructions have eight different types of addressing modes. They are:

1. Absolute (address mode bits equal 0, 1, 2, or 3). Relative and indirect flags are both "0" and delta does not equal zero. The base address equals delta. The sign bit of delta is not extended. The contents of the index registers, when specified, are added to the base address to form the effective address.

   If no indexing takes place, the addresses which can be referenced in the Absolute mode are restricted to the low core area. Delta can be only two hexadecimal characters and thus the computer references a location between 0000 and 00FF (address 00FF is the address of the i register).

   Example: Load A register with the contents of the absolute address specified by delta. C010 loads the contents of address 0010 into A. If an index register is used, the contents of either the i or the Q register are added to the absolute address to specify the effective address.

   i register or Address 00FF = 0005<br>
   0015 = 1234

   The command C110 adds the contents of i to delta and from location 0015 loads the A register with 1234.

2. Constant (address mode bits equal 0, 1, 2, or 3). Relative and indirect flags are both "0" and delta is zero.

   a. When the address mode bits are zero, P + 1 is the effective address.

b. When the address mode bits equal 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 6 bit operand for read-operand-type instructions.

Example: P = C000 P + 1 = 03E8

The computer loads the A register with the contents of P + 1 (i.e., 03E8 is in the register). If indexing is specified, the index register is added to form the constant.

P = C200 P + 1 = 03E8 Q = 0001

The A register is loaded with 03E9 because the Q register contained 0001.

3. Indirect (address mode bits equal 4, 5, 6, or 7). Relative address flag is "0", indirect flag is "1", and delta does not equal zero. The 8-bit value of delta is an indirect address. Delta is a magnitude quantity for this operation (no sign bit).

The computer goes to the base address in low core (addresses 0000 through 00FF) and treats the contents of this address as the effective address of the operand unless indexing is specified. Indexing takes place after indirect addressing is completed. If the sign bit of an indirect address (bit 15) is set, the address is another indirect address.

Indirect addressing continues until the sign bit of a word is not set.

Example: P = C4FE 00FE = 0500 0500 = 1234

The computer examines the contents of location 00FE and finds that the sign bit is not set. It then loads the contents of location 0500 into the A register. If indexing had been specified, the indexing quantity would have been added to the contents of address 00FE to reach the effective address. The program continues at P + 1. If address 00FE had contained 8500, the computer would have accepted 1234 in location 0500 as another indirect address and loaded the A register with the contents of address 1234.

Both Absolute and Indirect modes of addressing reference the low core locations, 0000 through 00FF. The difference is that absolute loads the A register with the contents of the address in low core and indirect uses this location as the address of the operand.

4. Storage (address mode bits equal 4, 5, 6, or 7). Relative address flag is "0", indirect flag is "1", and delta is zero. The content 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.

This mode of addressing is very similar to indirect but instead of low core references, the indirect address is found in P + 1. Again, indirect addressing continues as long as the sign bit is set.

Example: P = C400 P + 1 = 8500 0500 = 1234

The computer examines P + 1 and finds that the upper bit is set. It accepts the contents of location 0500 as another indirect address and loads A with the contents of address 1234.

5. Relative (address mode bits equal 8, 9, A, or B). The relative flag is "1", indirect address flag is "0", and delta does not equal zero. 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. The delta portion of the instruction is evaluated with a sign extension which makes it possible to subtract from the current address.

   Example: P = C805 P + 5 = 1234

   The instruction address, P, is augmented by 0005, and the computer loads A with 1234.

   P(0100) = C883

   In this case, the quantity delta is negative (bit 7 is set) so that P (0100) plus FF83 is 0084. The A register is then loaded with the contents of location 0084.

   One or both index registers can be added to the base address to form the effective address.

6. 16-Bit Relative (address mode bits equal 8, 9, A, or B). The relative address flag is "1", the indirect address flag is "0", and delta is zero. If no indexing is specified, the instruction address, P + 1, plus the contents of location P + 1 forms the base address which in this case equals the 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.

   This mode is very similar to Relative except the base address is formed from P + 1 + (P + 1) instead of P plus delta.

   Example: P (010E) = C800 P + 1 (010F) = 0011
   Effective Address (0120) = 1234

   The contents of P + 1 are added to P + 1 resulting in an effective address of 0120, and 1234 is loaded into the A register. Since the address referenced is relative to P + 1, a program can be relocated easily.

7. Relative Indirect (address mode bits equal C, D, E, or F). Both relative and indirect flags are "1".

   a. Delta does not equal zero. The value of the instruction address, P, plus the value of delta with sign extended is an indirect address. If bit 15 of the contents of this indirect address is "0", the content of this indirect address is the base address. If bit 15 of the contents of the indirect address is a "1", the content 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.

Example: 0100 = CCAB 00AB = 8103

0101 = 00AB 00AC = 8102

0102 = 00AB

The Load A instruction forms P plus delta (0100 + FFAB) which is 00AC. The content of address 00AC is 8102. Since the sign bit is set, the next indirect address referenced is 0102. The effective address at location 0102 is 00AC. The computer loads 8103 from this location into the A register. The program continues at 0101.

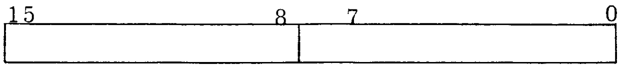
b. Delta equals zero. If bit 15 of [P + 1 + (P + 1)] is a "1", [ P + 1 + (P + 1)] is an indirect address. Indirect addressing continues until bit 15 of the contents of an indirect address is "0". Then the contents of the index registers, when specified, are added to the base address to form the effective address.

Example: 0100 = CC00 0101 + FFFE 4C00 = 0101

The Load A instruction adds P + 1 (0101) to the contents of this address, FFFE, and obtains 0100. This address is referenced; the CC00 indicates that another indirect addressing cycle is necessary (bit 15 is set). Indirect addressing is repeated at 4C00 and produces an effective address of 0101. The A register is loaded with FFFE from this location. The binary bits of C are 1100 and the computer disregards the sign bit to determine the indirect address. For this reason, 4C00 (binary 4 is 0100) is referenced and not location CC00.

8. Character Character addressing gives the System Controller the ability to address one half of a memory location. It is used with Load A and Store A instructions only. A character address is 16 bits long: 15 bits to reference 32K memory locations and the 16th bit to divide each location into two 8-bit characters. The 15 bits represent the standard effective address obtained by using any of the above addressing modes. The 16th bit is bit zero of the i register; the logical state of this bit determines whether the upper or lower half of the memory location is to be used. If bit zero is a "1", character 1 (lower 8 bits of the memory location) is referenced. If bit zero is a "0", character 0 (upper 8 bits of the memory location) is referenced. Since bit zero of the register indicates which character is addressed, it is called the Character Designation Bit (CDB).

Contents of memory location XXXX or A register



Character 0
selected by i Register
bit zero being a "0".

Character 1
selected by i Register
bit zero being a "1".

The execution of two instructions, Enable Character Addressing (ECA) and Disable Character Addressing (DCA), will determine when the Character Addressing mode is used. Once a particular character of a particular address is selected, the Load A and Store A instructions function as follows:

1. STORE A (F=6) (STA)

   a. If the CDB is a "0", character 1 of the A register is stored in the character 0 position of the memory word specified by the 15-bit word address. Character 1 of the memory word is unchanged and new parity is generated. The contents of the A register are unchanged.

   b. If the CDB is a "1", character 1 of the A register is stored in the character 1 position of the memory word specified by the 15-bit word address. Character 0 of the memory word is unchanged and new parity is generated. The contents of the A register are unchanged.

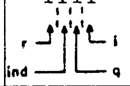
2. LOAD A (F=C) (LDA)

   a. If the CDB is a "0", character 0 of the memory word specified by the 15-bit word address is loaded into the character 1 position of the A register. Character 0 in the A register is cleared. The 18-bit memory word remains unchanged.

   b. If the CDB is a "1", character 1 of the memory word specified by the 15-bit word address is loaded into the character 1 position of the A register. Character 0 in the A register is cleared. The 18-bit memory word remains unchanged.

## TABLE 3-2. STORAGE ADDRESSING RELATIONSHIPS

NOTE
( ) and [ ] denote "contents of."

| Mode | Address Mode Bits<br>Binary | Hex | Delta | Effective Address | Address of<br>Next Instruction |
|---|---|---|---|---|---|
| Absolute | 0000 | 0 | Δ ≠ 0 | Δ | P + 1 |
| Constant | | | Δ = 0 | P + 1 | P + 2 |
| Absolute | 0001 | 1 | Δ ≠ 0 | Δ + (00FF) | P + 1 |
| Constant | | | Δ = 0 | (P + 1) + (00FF)* | P + 2 |
| Absolute | 0010 | 2 | Δ ≠ 0 | Δ + (Q) | P + 1 |
| Constant | | | Δ = 0 | (P + 1) + (Q)* | P + 2 |
| Absolute | 0011 | 3 | Δ ≠ 0 | Δ + (Q) + (00FF) | P + 1 |
| Constant | | | Δ = 0 | (P + 1) + (Q) + (00FF)* | P + 2 |
| Indirect | 0100 | 4 | Δ ≠ 0 | (Δ) | P + 1 |
| Storage | | | Δ = 0 | (P + 1) | P + 2 |
| Indirect | 0101 | 5 | Δ ≠ 0 | (Δ) + (00FF) | P + 1 |
| Storage | | | Δ = 0 | (P + 1) + (00FF) | P + 2 |
| Indirect | 0110 | 6 | Δ ≠ 0 | (Δ) + (Q) | P + 1 |
| Storage | | | Δ = 0 | (P + 1) + (Q) | P + 2 |
| Indirect | 0111 | 7 | Δ ≠ 0 | (Δ) + (Q) + (00FF) | P + 1 |
| Storage | | | Δ = 0 | (P + 1) + (Q) + (00FF) | P + 2 |
| Relative | 1000 | 8 | Δ ≠ 0 | P + Δ | P + 1 |
| 16-Bit Relative | | | Δ = 0 | P + 1 + (P + 1) | P + 2 |
| Relative | 1001 | 9 | Δ ≠ 0 | P + Δ + (00FF) | P + 1 |
| 16-Bit Relative | | | Δ = 0 | P + 1 + (P + 1) + (00FF) | P + 2 |
| Relative | 1010 | A | Δ ≠ 0 | P + Δ + (Q) | P + 1 |
| 16-Bit Relative | | | Δ = 0 | P + 1 + (P + 1) + (Q) | P + 2 |
| Relative | 1011 | B | Δ ≠ 0 | P + Δ + (Q) + (00FF) | P + 1 |
| 16-Bit Relative | | | Δ = 0 | P + 1 + (P + 1) + (Q) + (00FF) | P + 2 |
| Relative Indirect | 1100 | C | Δ ≠ 0 | (P + Δ) | P + 1 |
| Relative Indirect | | | Δ = 0 | [ P + 1 + (P + 1)] | P + 2 |
| Relative Indirect | 1101 | D | Δ ≠ 0 | (P + Δ) + (OOFF) | P + 1 |
| Relative Indirect | | | Δ = 0 | [ P + 1 + (P + 1)] + (00FF) | P + 2 |
| Relative Indirect | 1110 | E | Δ ≠ 0 | (P + Δ) + (Q) | P + 1 |
| Relative Indirect | | | Δ = 0 | [ P + 1 + (P + 1)] + (Q) | P + 2 |
| Relative Indirect | 1111 | F | Δ ≠ 0 | ( P + Δ) + (Q) + (00FF) | P + 1 |
| Relative Indirect |  | | Δ = 0 | [ P + 1 + (P + 1)] + (Q) + (00FF) | P + 2 |
| Character** | Any of the above modes may be used to form the effective address. The CDB (bit 0 of i) will select upper or lower character of the effective address. | | | | |

*Effective address is the 16-bit operand for read-operand-type instructions
**Optional character addressing

## Data Transmission

### STQ (F = 4)

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

### STA (F = 6)

Store A   Store the contents of the A register in the storage location specified by the effective address. The contents of A are not altered. If Character Addressing is enabled, this instruction will operate as described on page 3-7.

### SPA (F = 7)

Store A, Parity to A   Store the contents of the A register in the storage location specified by the effective address. Clear A if the number of "1" bits in A is odd. Set A equal to $0001_{16}$ if the number of "1" bits in A is even. The contents of A are not altered if the write into storage is aborted because of parity error or protect fault.

### LDA (F = C)

Load A   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. If Character Addressing is enabled, this instruction will operate as described on page 3-7.

### LDQ (F = E)

Load Q   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.

## Arithmetic

All the following arithmetic operations use one's complement arithmetic.

### MUI (F = 2)

Multiply Integer   Multiply the contents of the storage location specified by the effective address by the contents of the A register. The 32-bit product replaces the contents of Q and A, the most significant bits of the product in the Q register.

### DVI (F = 3)

Divide Integer   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 dividing. The quotient is in the A register and the remainder in the Q register at the end of the Divide operation. If a 16-bit dividend is loaded into A, the sign of A must be set (the sign of the dividend A must be extended throughout Q).

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.

### ADD (F = 8)

Add to A   Add the contents of the storage location specified by the effective address to the contents of the A register.

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.

### SUB (F = 9)

Subtract From A   Subtract the contents of the storage location specified by the effective address from the contents of the A register. Operation on overflow is the same as for an Add to A instruction.

### RAO (F = D)

Replace Add One in Storage   Add one to the contents of the storage location specified by the effective address. The contents of A are not altered. Operation on overflow is the same as for an Add to A instruction.

### ADQ (F = F)

Add to Q   Add the contents of the storage location specified by the effective address to the contents of the Q register. Operation on overflow is the same as for an Add to A instruction.

## Logical

The AND (AND With A) instruction achieves its result by forming a logical product. A logical product is a bit-by-bit multiplication of two binary numbers according to the following rules:

| | |
|---|---|
| 0 x 0 = 0 | 1 x 0 = 0 |
| 0 x 1 = 0 | 1 x 1 = 1 |

```
Example: 0011 Operand A
         0101 Operand B
         ----
         0001 Logical Product
```

A logical product is used, in many cases, to select only specific portions of an operand for use in some operation. For example, if only a specific portion of an operand in storage is to be entered into the A register, the operand is subjected to a mask in A. This mask is composed of a predetermined pattern of "0's" and "1's". Executing the AND instruction causes the operand to retain its original contents only in those bits which have "1's" in the mask in A.

The EOR (Exclusive OR With A) instruction achieves its result by forming an exclusive OR. Executing the EOR instruction causes the operand to complement its original contents only in those bits which have "1's" in the mask in A. An exclusive OR is a bit-by-bit logical subtraction of two binary numbers according to the following rules:

Exclusive OR

| A | B | A ⊻ B |
|---|---|---|
| 1 | 1 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 0 | 0 | 0 |

```
Example:  0011  Operand A
          0101  Operand B
          ----
          0110  Exclusive OR
```

AND (F = A)

AND With A  Form the logical product, bit by bit, of the contents of the storage location specified by the effective address and the contents of the A register. The result replaces the contents of A. The contents of storage are not altered.

EOR (F = B)

Exclusive OR With A  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 result replaces the contents of A. The contents of storage are not altered.

## Jumps

A Jump (JMP) instruction cases a current program sequence to terminate and initiates a new sequence at a different location in storage. The Program Address register, P, provides continuity between program instructions and always contains the storage location of the current instruction in the program.

When a Jump instruction occurs, P is cleared and a new address is entered.* In the Jump instruction, the effective address specifies the beginning address of the new program sequence. The word at the effective address is read from storage and interpreted as the first instruction of the new sequence.

A Return Jump (RTJ) instruction enables the computer to leave the main program, jump to some subprogram, execute the subprogram, and return to the main program via another instruction. The Return Jump provides the computer with the necessary information to enable returning to the main program. Figure 3-2 shows how a Return Jump instruction can be used.

*Jumps or return jumps from unprotected to protected storage cause a fault, but the address that is saved in the trap location is the destination address (i.e., the address of the next sequential main program instruction). See Programming Requirements in Section 4.

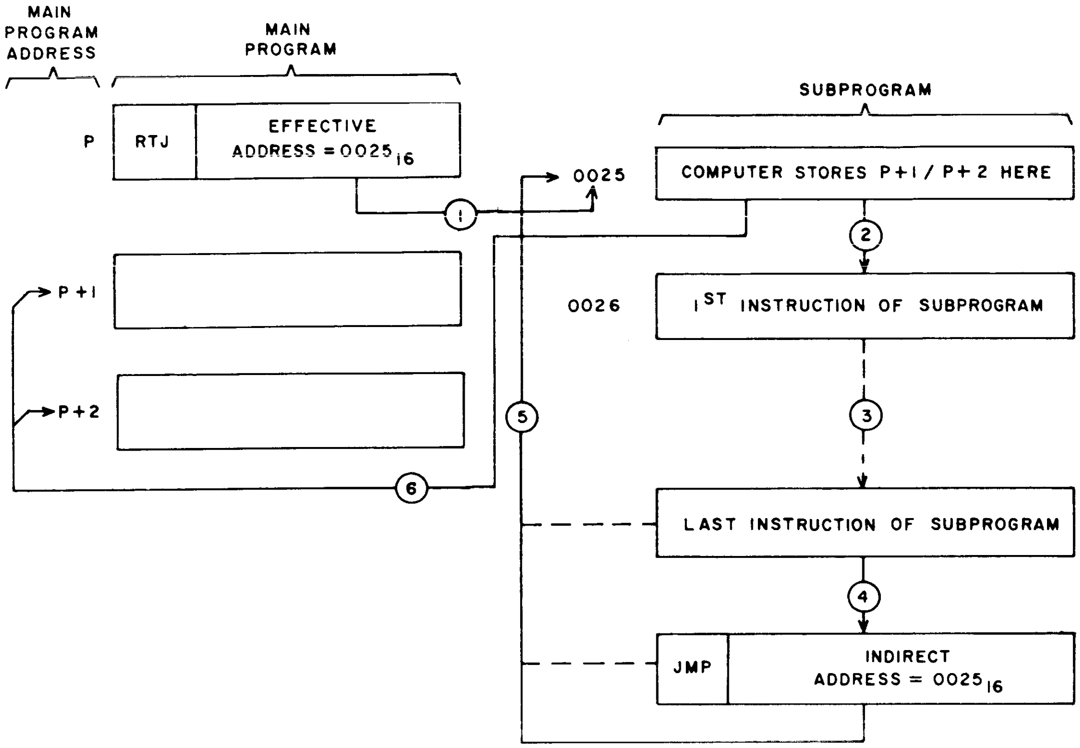

MAIN PROGRAM ADDRESS
MAIN PROGRAM
SUBPROGRAM
P
RTJ
EFFECTIVE ADDRESS = 0025₁₆
0025
COMPUTER STORES P+1 / P+2 HERE
1
2
P+1
P+2
0026
1ST INSTRUCTION OF SUBPROGRAM
3
5
6
LAST INSTRUCTION OF SUBPROGRAM
4
JMP
INDIRECT ADDRESS = 0025₁₆

Figure 3-2. Program Using Return Jump Instruction

A Return Jump instruction is executed at main program address P. The computer jumps to effective address $0025_{16}$ and stores P + 1 or P + 2 (depending on the address mode of RTJ) at this location. Then the program address counter, P, is set to $0026_{16}$ and the computer starts executing the subprogram. At the end of the subprogram, the computer executes a Jump instruction (JMP) with indirect addressing. This causes the computer to jump to the address specified by the subprogram address $0025_{16}$ which is P + 1 or P + 2 of the main program. Now main program execution continues at P + 1 or P + 2.

JMP (F = 1)

Jump Jump to the address specified by the effective address. This effectively replaces the contents of the program address counter, P, with the effective address specified in the Jump instruction.

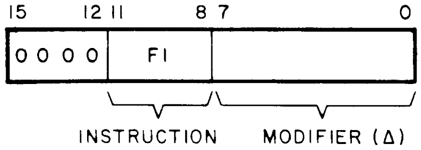
RTJ (F = 5)

Return Jump 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 plus one.

## Register Reference Instructions

Register reference instructions use the address mode field for the operation code. Register reference instructions are identified when the upper 4 bits (15 through 12) of an instruction are "0's"

Format:



### SLS (F1 = 0)

Selective Stop   Stops the computer if this instruction is executed when the SELECTIVE STOP switch is on. This becomes a Pass instruction when the switch is off. On restart, the computer executes P + 1.

### INP (F1 = 2)

Input to A   Reads one word from an external device into the A register. The word in the Q register selects the sending device. If the device sends a Reply, the next instruction comes from P + 1. If the device sends a Reject, the next instruction comes from P + 1 plus delta, when delta is an 8-bit signed number. If an internal Reject occurs, the next instruction comes from P plus delta. Refer to Section 5, Input/Output.

### OUT (F1 = 3)

Output from A   Outputs one word from the A register to an external device. The word in the Q register selects the rcceiving device. If the device sends a Reply, the next instruction comes from P + 1. If the device sends a Reject, the next instruction comes from P + 1 plus delta, where delta is an 8-bit signed number. If an internal Reject occurs, the next instruction comes from P plus delta. Refer to Section 5, Input/Output.

### INA (F1 = 9)

Increase A   Replaces the contents of A with the sum of the initial contents of A and delta, where delta is treated as a signed number with the sign extended into the upper 8 bits. Operation on overflow is the same as for an Add to A instruction.

### ENA (F1 = A)

Enter A   Replaces the contents of the A register with the 8-bit delta, sign extended.

### NOP (F1 = B)

No Operation   Compare with Selective Stop Pass instruction.

ENQ (F1 = C)

Enter Q Replaces the contents of the Q register with the 8-bit delta, sign extended.

INQ (F1 = D)

Increase Q Replaces the contents of Q with the sum of the initial contents of Q and delta, where delta is treated as a signed number with the sign extended into the upper 8 bits. Operation on overflow is the same as for an Add to A instruction.

ECA (F1 = 5) Δ = B0

Enable Character Addressing Activates the character addressing feature. After this instruction has been executed, all Load A and Store A instructions will use the character addressing mode.

DCA (F1 = 5) Δ = C0

Disable Character Addressing Deactivates the character addressing feature.

The following instructions (F1 equals 4, 5, 6, 7, with Δ = 0,or E) are legal only if the PROGRAM PROTECT switch is off or if the instructions themselves are protected (refer to Section 4). If an instruction is illegal, it becomes a selective stop, and an interrupt on program protect fault is possible (if selected).

- Switch on: Pass unless instruction is protected (program protect bit set).
- Switch off: Normal instruction execution (no program protection).

EIN (F1 = 4) Δ = 0

Enable Interrupt Activates the interrupt system after one instruction following EIN has been executed. The interrupt system must be active and the appropriate mask bit set for an interrupt to be recognized.

IIN (F1 = 5) Δ = 0

Inhibit Interrupt Deactivates the interrupt system. If interrupt occurs during execution of this instruction, the interrupt is not recognized until one instruction after the next EIN instruction is executed.

SPB (F1 = 6) Δ = 0

Set Program Protect Bit Sets the program protect bit in the address specified by Q.

CPB (F1 = 7) Δ = 0

Clear Program Protect Bit Clears the program protect bit in the address specified by Q.

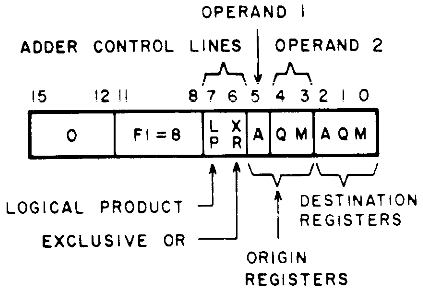
EXI (F1 = E)

Exit Interrupt State This instruction must be used to exit from any interrupt state. Delta defines the interrupt state from which the exit is taken (see Table 4-1). This instruction automatically reads the address containing the return address, resets the OVERFLOW indicator according to bit 15, activates the interrupt system, and jumps to the return address.

## Interregister

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

Format:



If bit 0 of an Interregister instruction is set (M is the destination register) and the instruction is not protected, this is a nonprotected Selective Stop instruction. The program protect fault bit is set and interrupt occurs if selected. See Section 4 for additional information.

The origin registers are considered as operands of which there are two kinds defined as follows:

Operand 1 may be:

- FFFF (bit 5 is "0") or
- the contents of A (bit 5 is "1")

Operand 2 may be:

- FFFF (bit 4 is "0" and bit 3 is "0") or
- the contents of M (bit 4 is "0" and bit 3 is "1") or
- the contents of Q (bit 4 is "1" and bit 3 is "0") or
- the inclusive OR, bit by bit, of the contents of Q and M (bit 4 is "1" and bit 3 is "1")

Operations possible are (see Table 3-3):

- Exclusive OR (LP = "0" and XR = "1"). The data placed in the destination register(s) is the exclusive OR, bit by bit, of operand 1 and operand 2.
- Logical Product (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.
- Complement Logical Product (LP = "1" and XR = "1"). The data placed in the destination register(s) is the complement of the logical product, bit by bit, of operand 1 and operand 2.
- Arithmetic Sum (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 for an Add to A instruction.

TABLE 3-3. INTERREGISTER INSTRUCTION TRUTH TABLE

| Operand 1 | Operand 2 | Exclusive OR LP = 0 XR = 1 | Logical Product LP = 1 XR = 0 | Complement Logical Product LP = 1 XR = 1 | LP = 0 XR = 0 |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | Arithmetic Sum |
| 0 | 1 | 1 | 0 | 1 | |
| 1 | 0 | 1 | 0 | 1 | |
| 1 | 1 | 0 | 1 | 0 | |

Notes:

a. Register transfers can be accomplished by setting LP and XR to "0" and either operand 1 or operand 2 to $FFFF_{16}$.

b. Magnitude comparisons without destroying either operand can be done by setting LP and XR to "0", selecting no destination register, and then testing the OVERFLOW indicator. The overflow condition that results from adding two operands together without altering either operand is obtained when LP and XR equal "0" and no destination register is selected. Assume we wish to test a set of operands, N, to exceed the value of one operand, M. The test value to be used is that number, P, which when added to M, produces a sum which exceeds the register capacity by one bit, causing overflow. If the sum of N and P cause overflow:

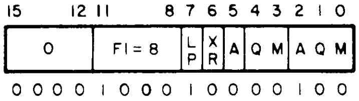
$$N \geq M \text{ if } M > 0,$$

$$\text{and } N \leq M \text{ if } M < 0.$$

c. Complementing registers can be done by setting LP to "0", XR to "1", and either operand 1 or operand 2 to $FFFF_{16}$.

Destination register bits must be determined and the whole instruction written out.

Example: Set A to "1's" is 0884 since bits 7 and 2 are "1" and all others from 0 through 7 are "0".



Interregister Mnemonics:

| | |
|---|---|
| SET | (F1 = 8, bits 7 through 3 = 10000) Set To "1's" |
| CLR | (F1 = 8, bits 7 through 3 = 01000) Clear To "0" |
| TRA | (F1 = 8, bits 7 through 3 = 10100) Transfer A* |
| TRM | (F1 = 8, bits 7 through 3 = 10001) Transfer M* |
| TRQ | (F1 = 8, bits 7 through 3 = 10010) Transfer Q* |

Note: "+" symbol implies an OR.

*The use of bit 7 is optional; it may be a "1" or a "0". The assembler uses bit 7 = "0".

TRB (F1 = 8, bits 7 through 3 = 10011) Transfer Q + M*

TCA (F1 = 8, bits 7 through 3 = 01100) Transfer Complement A*

TCM (F1 = 8, bits 7 through 3 = 01001) Transfer Complement M*

TCQ (F1 = 8, bits 7 through 3 = 01010) Transfer Complement Q*

TCB (F1 = 8, bits 7 through 3 = 01011) Transfer Complement Q + M*

AAM (F1 = 8, bits 7 through 3 = 00101) Transfer Arithmetic Sum A, M

AAQ (F1 = 8, bits 7 through 3 = 00110) Transfer Arithmetic Sum A, Q

AAB (F1 = 8, bits 7 through 3 = 00111) Transfer Arithmetic Sum A, Q + M

EAM (F1 = 8, bits 7 through 3 = 01101) Transfer Exclusive OR A, M

EAQ (F1 = 8, bits 7 through 3 = 01110) Transfer Exclusive OR A, Q

EAB (F1 = 8, bits 7 through 3 = 01111) Transfer Exclusive OR A, Q + M

LAM (F1 = 8, bits 7 through 3 = 10101) Transfer Logical Product A, M

LAQ (F1 = 8, bits 7 through 3 = 10110) Transfer Logical Product A, Q

LAB (F1 = 8, bits 7 through 3 = 10111) Transfer Logical Product A, Q + M

CAM (F1 = 8, bits 7 through 3 = 11101) Transfer Complement Logical Product A, M

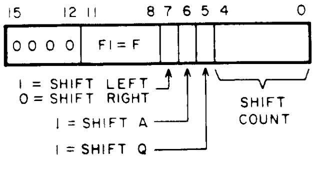
CAQ (F1 = 8, bits 7 through 3 = 11110) Transfer Complement Logical Product A, Q

CAB (F1 = 8, bits 7 through 3 = 11111) Transfer Complement Logical Product A, Q + M

## Shifts

These Shift instructions shift A, Q, or QA left or right the number of places specified by the 5-bit shift count. Right shifts are end-off with sign extension in the upper bits. Left shifts are end-around. The maximum long-right or long-left shift is $31_{10}$ places.

Format:

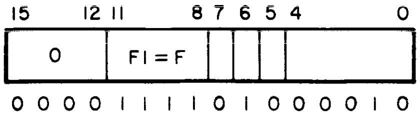


Note: "+" symbol implies an OR.

*The use of bit 7 is optional; it may be a "1" or a "0". The assembler uses bit 7 = "0".

Bit configurations must be determined for each instruction.

Example: Shift A right two places is 0F42.



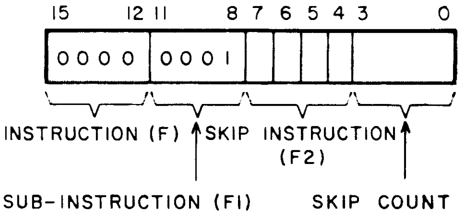
Shift Mnemonics:

| | |
|---|---|
| ARS | (F1 = F) A Right Shift |
| QRS | (F1 = F) Q Right Shift |
| LRS | (F1 = F) Long Right Shift (QA) |
| ALS | (F1 = F) A Left Shift |
| QLS | (F1 = F) Q Left Shift |
| LLS | (F1 = F) Long Left Shift (QA) |

## Skip Instructions

Skip instructions are identified when the instruction mode field is zero and the subinstruction mode field is one.

Format:



When the Skip condition is met, the skip count plus one is 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.

| | |
|---|---|
| SAZ | (F2 = 0) Skip if A is positive zero (all bits are "0") |
| SAN | (F2 = 1) Skip if A is not positive zero (not all bits are "0") |
| SAP | (F2 = 2) Skip if A is positive (bit 15 is "0") |
| SAM | (F2 = 3) Skip if A is negative (bit 15 is "1") |
| SQZ | (F2 = 4) Skip if Q is positive zero (all bits are "0") |
| SQN | (F2 = 5) Skip if Q is not positive zero (all bits are not "0") |

SQP (F2 = 6) Skip if Q is positive (bit 15 is "0")

SQM (F2 = 7) Skip if Q is negative (bit 15 is "1")

SWS (F2 = 8) Skip if SELECTIVE SKIP switch is set

SWN (F2 = 9) Skip if SELECTIVE SKIP switch is not set

SOV (F2 = A)

Skip on Overflow This instruction skips if an Overflow condition occurred; this skip clears the OVERFLOW indicator.

SNO (F2 = B) Skip on No Overflow

SPE (F2 = C)

Skip on Storage Parity Error This instruction skips if a Storage parity error occurred; it clears the Storage Parity Error Interrupt signal and the STORAGE PARITY FAULT indicator.

SNP (F2 = D) Skip on No Storage Parity Error

SPF (F2 = E)

Skip on Program Protect Fault Program protect fault is set by:

- A nonprotected instruction attempting to write into an address which is protected.
- An attempt to execute a protected instruction immediately following a nonprotected instruction unless an interrupt caused the instruction sequence.
- Execution of any nonprotected instruction affecting interrupt mask or enables.

The program protect fault is cleared when it is sensed by the SPF instruction. The program protect fault cannot be set if the program protect system is disabled. (Refer to Program Protection in Section 4.)

SNF (F2 = F) Skip on No Program Protect Fault

## NEGATIVE ZERO

Negative zero ($FFFF_{16}$) can be caused because of two characteristics of the computer.

1. The computer has a one's complement subtractive adder.
2. Multiply and divide are done with positive numbers only. Therefore, a sign correction occurs, if required, before and after the multiply or divide.

Arithmetic operations which produce a negative zero result in the computer are:

| | |
|---|---|
| ● Addition | (-0) + (-0) = (-0) |
| ● Subtraction | (-0) - (+0) = (-0) |
| ● Multiplication | (+0) x (-N) = (-0)<br>(-N) x (+0) = (-0)<br>(-0) x (+N) = (-0)<br>(+N) x (-0) = (-0) |
| ● Division | (+0) / (-N) = (-0), R = (+0)<br>(-0) / (+N) = (-0), R = (-0)<br>(+2N) / (-N) = (-2), R = (+0) |

# INTERRUPT SYSTEM AND PROGRAM PROTECTION 4

## INTERRUPT SYSTEM

The computer interrupt system provides for testing 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:

- If conditions for interrupting occur while the computer is reading up an instruction, the main program is interrupted before that instruction is executed.

- If conditions for interrupting occur while the computer is reading up an indirect address and bit 15 is set, the interrupt occurs before that instruction is executed.

In the 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 of Interrupt System

The interrupt system consists of fixed interrupt trap locations and the interrupt Mask register.

## Basic and Optional Interrupts

The basic computer has two interrupts, one internal (storage parity error, power failure, or program protect fault, interrupt state 00) and one external (interrupt state 01). The AQ interrupt data channel option may be added which gives an additional 14 external interrupt lines. Thus, the computer may have up to 16 different interrupts. The discussions that follow assume the computer has 16 interrupts.

## 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 4-1. 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.

TABLE 4-1. 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 |

*Interrupts in basic computer

**Interrupts added by 1705 Interrupt Data Channel option

### 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. To enable an interrupt line, its corresponding bit in the Mask register must be set. The Mask register is set by the Interregister instruction.

## Programming and Operation of the Interrupt System

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:

- The interrupt system must be activated.
- Internal and external conditions to be tested must be selected with various masks.
- Interrupt routines must be programmed to determine the cause of interrupt and to process and clear the interrupt.

The computer can distinguish up to 16 different interrupts. 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.

When the computer 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 arc numbered 00 through 15 (e.g., bit 7 corresponds to interrupt state 7).

Before the computer can recognize any interrupt, the mask bit for that interrupt must be set, and the interrupt system must be activated. The Mask register can be set by an Interregister instruction, and the interrupt system is activated by an Enable Interrupt instruction.

Upon recognizing an interrupt, the computer automatically stores the return address in the lower 15 bits of the storage location reserved for that interrupt state. Bit 16 of the storage location is set or cleared to record the current state of the OVERFLOW indicator. The OVERFLOW indicator itself is then cleared. Also, the state of the Character Addressing feature for each interrupt state is recorded and deactivated. The computer then deactivates the interrupt system and transfers control to another address also specified by the interrupt state. 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 "1's" in the mask denote interrupts that have higher priority than the interrupt being processed. The mask should not have a "1" in the position of the interrupt 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 computer exits from an interrupt state as follows. The program inhibits interrupt and restores the registers, including the Mask register. 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 set or cleared in accordance with bit 16, the interrupt system is activated, the state of the Character Addressing feature for each interrupt state is reinstated, and control transfers to the return address.

## Interrupt Priority

The priority of interrupts is under control of the computer program. 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.

The following table 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 interrupts 03 and 04 have low priority. Interrupt 00 has highest priority, but this example does not consider interrupt 00.

| 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
Set mask to Mask 4
Enable interrupt
       -
       -
Inhibit interrupt
Replace registers
Exit interrupt 01
```

### State 02 Program

```
Store registers
Set mask to Mask 3
Enable interrupt
       -
       -
Inhibit interrupt
Replace registers
Exit interrupt 02
```

### State 03 Program

```
Store registers
Set mask to Mask 2
Enable interrupt
       -
Inhibit interrupt
Replace registers
Exit interrupt 03
```

```
State 04 Program                 State 05 Program

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

### 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                   α    return link
Interrupt state 1                α + 1 IIN - inhibit interrupt
      -                                  -
      -                                  -
      -                                  -
RTJ   α                                  -
      -                                  -
Interrupt state 2                EIN - enable interrupt
      -                          JMP (Indirect α )
      -
      -
RTJ   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.

## Internal Interrupts

Certain internal interrupts are generated by conditions arising within the computer. If such a condition occurs, it generates interrupt 00 (corresponding interrupt mask bit is 00). Normally, internal interrupts are assigned the highest priority. These interrupts are:

- Storage parity error
- Program protect fault
- Power failure

### Storage Parity Error

A storage parity bit is generated and entered with every word written into storage. Storage parity is checked (and consequently an error may occur) in two cases:

1) When instructions/data are read from storage, parity is checked.
2) When a word is written into storage, the existing word at the address is first read from storage and its parity is checked.

A 00 interrupt state occurs if conditions for interrupting (mask bit 00 set and interrupt active) are present and a storage parity error occurs in either of the two preceding operations.

Once a storage parity error occurs and the PROGRAM PROTECT switch is set (whether or not interrupt is selected), no instruction can write into storage unless the instruction is protected. The operation that stores P or P + 1 when interrupt occurs is the protected operation.

The Storage Parity Error Interrupt signal and indicator are cleared when the computer executes a Skip On Storage Parity Error instruction.

### Program Protect Fault

Refer to Program Protection in this section for some special cases of storage parity errors as related to program protection and for a discussion of the program protect fault interrupt.

### Power Failure

The internal interrupt (00) sends a response to the computer if the mask bit is set when a power failure occurs. To determine that power failure caused the interrupt, the Skip on Parity Error and Skip on Program Protect Fault instructions should be used. A negative response to these instructions indicates that power failure caused the interrupt. The programmer can use the interrupt to store the contents of significant data registers so that the program can be continued upon power restoration. A Master Clear follows a minimum of 0.5 millisecond after the interrupt, providing at least 0.5 milliseconds of program execution time.

## PROGRAM PROTECTION

The computer has a program protect system which makes it possible to protect a program in the computer from any other nonprotected program also in the computer. The system is built around a program protect bit (bit 17) contained in each word of storage. If the bit is set, that word is an operand or an instruction of the protected program. All operands and instructions in the protected program must have the program protect bit set. None of the instructions or operands of the nonprotected program may have the program protect bit set.

### Clearing/Setting the Program Protect Bit

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

### Program Protect Switch

Program protect is manually enabled by a two-position switch on the computer console. If the switch is not enabling program protect, no program protect violations are recognized.

### Program Protect Violations

Whenever a violation of the program protect system is detected, the program protect fault is set and an internal interrupt is enabled. A 00 interrupt occurs if mask bit 00 is set and the interrupt system is active. A violation indicates that the nonprotected program has attempted an operation which could harm the protected program. The four program protect violations are:

1. An attempt is made by nonprotected instruction to write into a storage location containing a protected instruction/operand. The content of the storage location is not altered.

2. An attempt is made to execute a protected instruction following the execution of a nonprotected instruction. The protected instruction is executed as a nonprotected Selective Stop instruction. It is not a violation, however, if an interrupt caused this sequence of instructions.

3. An attempt is made to execute the following instructions when they are not protected: Interregister instruction with bit 0 a "1", instruction EIN, IIN, EXI, SPB, or CPB. These instructions become nonprotected Selective Stop instructions under these circumstances.

## Storage Parity Errors as Related to Program Protection

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

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

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

## Peripheral Equipment Protection

All peripheral equipments essential to operation of the protected program have a PROGRAM PROTECT switch. If the switch is on, the peripheral device responds with a Reject to all nonprotected commands (except status requests) addressed to it. The peripheral device responds to all protected commands in the normal manner. If the switch is off, the peripheral device responds in the normal manner to both protected and nonprotected commands.

## Direct Storage Access Memory Protect Fault

This fault occurs when an attempt is made to write into a protected storage location via the external storage access when a nonprotected instruction was the ultimate source of the attempt. The content of the storage location is not altered.

## Programming Requirements

In order for the program protect system to work, the following program requirements must be met:

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

Interrupt conditions are examined by the computer after each instruction is read from memory. If an interrupt condition is present at that time, the interrupt occurs at the end of the memory cycle. Thus, instructions which require one memory cycle are executed before the interrupt. Instructions which require more than one memory cycle are interrupted before they are completed and, in effect, are not executed before the interrupt.

Program protect violations can be detected by the computer at two different times, depending on the type of violation. Protect violations 1 and 3* are examined after the interrupt condition is detected. Thus, the program protect interrupt is found when the next instruction is read from memory, and the interrupt occurs just after that memory cycle. Protect violation 2 is examined just before the interrupt condition is detected. Thus, the program protect interrupt is found during the same memory cycle, and the interrupt occurs just after that memory cycle.

*See pages 4-6 and 4-7.

# INPUT/OUTPUT

Section 5 describes the input/output of the System Controller in a general manner and then describes the 1711/1712/1713 Teletypewriter located in the System Controller. For detailed codes and operating information for the other peripheral equipment, refer to the 1700 Computer System, Standard Peripheral Reference Manual, Pub. No. 60182700.

## GENERAL INFORMATION

The pivot of input/output is 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 in and out of the computer in a nonbuffered mode of operation. The addition of the buffered data channel enables data transfer to and from memory, independent of the internal operation of the computer; 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, besides 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 on the data cable. Either a 16-bit word or 8-bit character can be transmitted to or from the A register on the data cable. The Q register transmits addresses and control signals on the address cable.

The System Controller provides two ways to attach peripheral equipment: the AQ channel and the buffered data channel. The AQ channel can handle approximately 50,000 words per second, and the buffered data channel can handle approximately 570,000 words per second in a buffered mode. The characteristics of the peripheral equipment and its use in the system determine the data path used. An AQ Interrupt Data Channel is required to implement the AQ channel. The buffer data channel requires both an AQ Interrupt Data Channel and a Direct Storage Access. Then the 1706 Buffered Data Channel can be connected to the System Controller. The buffered data channel may use the AQ channel to transfer one word at a time into storage.

### Basic Peripheral Equipment

The basic peripheral equipment, Teletypewriter I or Teletypewriter II, is attached internally to the A and Q registers. Thus a System Controller with the basic peripheral operates as if the peripheral is on the AQ data channel. Any other peripherals require the addition of the AQ Interrupt Data Channel.

### Storage of Data

The AQ channel relies on the A register for access to storage. There are two commands in the System Controller used to reference storage for the A register: Load A (LDA) loads the A register and Store A (STA) stores the contents of the A register in memory. If the Character Addressing feature is present, LDA and STA operate differently.

The buffered I/O uses the A register to send the first word address minus one (FWA-1). The converter receives the contents of this address from memory, which is the last word address plus one (LWA+1). A comparison of the address currently being accessed and the LWA+1 is made in the converter to determine when to terminate the buffer operation.

## CONTROL SIGNALS

### Read

The Read signal signifies the request for an input operation. If data is available at the time of the Read signal, a Reply is returned within 4 microseconds; if data is not available, a Reject signal is returned within 4 microseconds.

### Write

The Write signal signifies the request for an output operation. If the data can be used at the time of the Write signal, a Reply is returned within 4 microseconds; if data cannot be used, a Reject signal will be returned within 4 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 computer channel performs the transfer 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 drops the Reply.
5. The data lines drop not less than 0.1 μsec after Write 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 4 microseconds later.
3. The Reply causes the Read line to drop.
4. Absence of a Read signal 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 4 microseconds. If a Reply or Reject is not returned within 4 microseconds, the computer will generate an internal Reject.

**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.

**Continue Bit**

Bit 15 of Q is a Continue bit and is used to speed the operation devices which require continuous random addressing. Such a device operates as follows:

1) Address the device with Q15 = 0 and the remainder of Q set to select this device. The device is not connected.

2) All succeeding addresses with Q15 = 1 will be recognized by this device. Thus 15 bits of address are available to this device.

3) The next address with Q15 = 0 will disconnect this device unless Q is the address of this device.

## PERIPHERAL EQUIPMENT LEVELS

Figure 5-1 shows the relationship of the AQ interrupt data channel and the direct storage access to the System Controller. Figure 5-2 shows the hierarchy of peripheral equipment in the System Controller. The equipment contains logic shared by the stations.

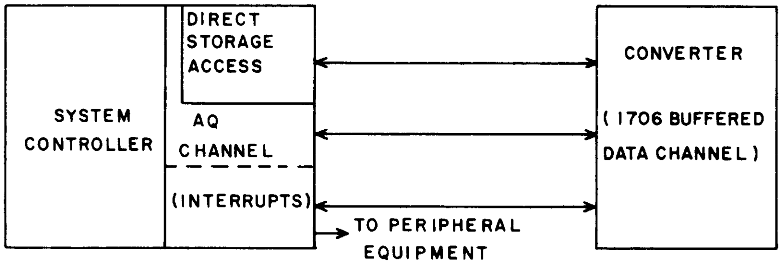


Figure 5-1. Generalized Block Diagram

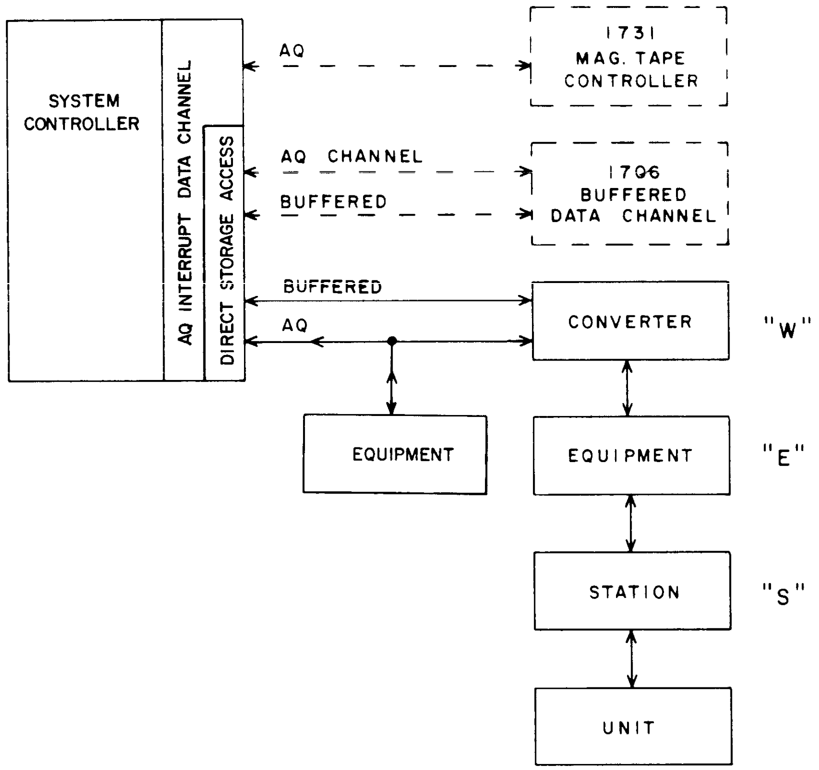


Figure 5-2. Peripheral Equipment Levels

## ADDRESSING

The Q register in the System Controller is used to send addressing codes to peripheral equipments. The format of the Q register is shown below. Each level of peripheral equipment (Figure 5-2) except a unit is addressed by a unique section of the Q register.

| 15 11 | 10 7 | 6 0 | |
|---|---|---|---|
| W | E | Command | 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 1706 Buffered Data Channel or similar converter. The (W) field must be zero for lower level peripheral devices and standard peripheral controllers.

### Equipment

Address bits 7 through 10 (E) contain the equipment number of the peripheral equipments on the channel (0 through $F_{16}$). Each device responds when the Equipment Number switch setting and the code in bits 7 through 10 match.

### Command Code

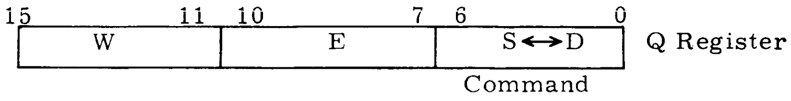
Bits 0 through 6 of the Q register are not specifically used by the channel and are therefore available to meet specific requirements of the station and unit. These bits control and direct information on the data cable in the following ways:

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 is zero.

## Unit

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 in the System Controller are initiated by the instructions Input to A and Output from A. The contents of the data cable 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 contents 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.

TABLE 5-1. USE OF D

| DIRECTOR BIT 0 | SYSTEM CONTROLLER INSTRUCTION | PERIPHERAL OPERATION |
|---|---|---|
| 0 | Output from A | Write Data |
| 0 | Input to A | Read Data |
| 1 | Output from A | Function code sent to peripheral |
| 1 | Input to A | Status of peripheral sent to the computer |

## Data Transfer

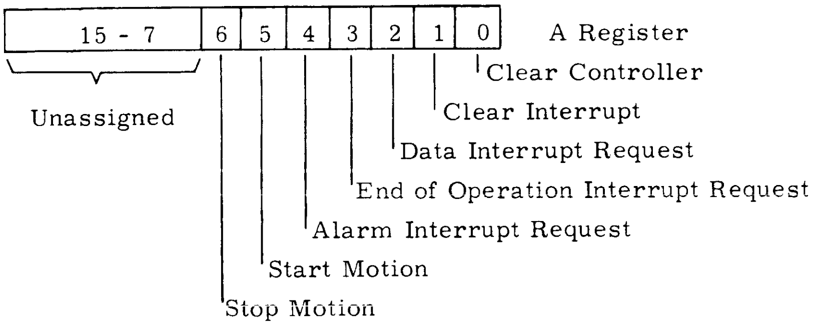
To transfer data, the Director must equal "0". An Input to A instruction initiates a Read operation and an Output from A instruction initiates a Write operation.

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.

## Function Bit Definitions



### Clear Controller

Bit 0 clears all interrupt requests and responses, motion requests, errors, and other logic. A function code in which bit 0 is set and any of bits 2 through 7 are set will first clear all previous functions and then immediately set the function conditions indicated by bits 2 thorough 7.

### Clear Interrupt

Bit 1 clears all interrupt requests and responses. A function code in which bit 1 is set and any of bits 2 through 7 are set will first clear all previous interrupt functions and then immediately set the function conditions indicated by bits 2 through 7.

### Data Interrupt Request

Bit 2 sets a Data Interrupt Request. An interrupt response is generated when a data transfer is possible. The interrupt response is cleared by the Reply to data transfer. Both the interrupt request and response are cleared by either the Clear Controller, Clear Interrupt, or Master Clear signals.

### End of Operation Interrupt Request

Bit 3 selects the End of Operation Interrupt Request. An End of Operation results any time the continuous data transfer is interrupted, e.g., End of Record. Both the interrupt request and response are cleared by either the Clear Controller, Clear Interrupt, or Master Clear signals.

### Alarm Interrupt Request

Bit 4 selects the Alarm Interrupt Request. An alarm may indicate a change of status (e.g., Ready to Not Ready) or it may be an indication of an error (e.g., Lost Data) or a warning (e.g., End of Tape). Each equipment must specify the manner in which the alarm is used and must provide a status indication for each condition causing an alarm. Both the interrupt request and response are cleared by either the Clear Controller, Clear Interrupt, or Master Clear signals.

### Start Motion

Bit 5 directs the device to start motion in its storage medium. If Start Motion does not apply to the particular device, the bit may be optionally used in another manner.

Stop Motion

Bit 6 halts the operation started by Start Motion. Stop Motion takes precedence over Start Motion.

Unassigned

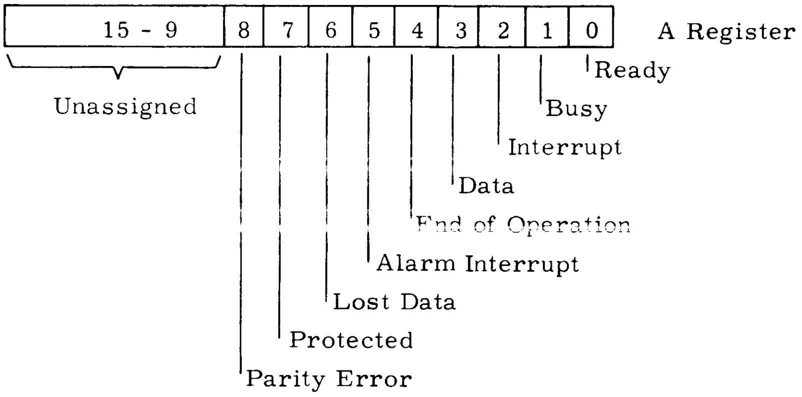
Bits 7 through 15 are unassigned and are used for the specific requirements of particular peripherals.

## STATUS

### Director Status

When bit 0 of the address code in the Q register is a "1", all station bits (if any) are "0", and a Read signal is present, the data lines (A) are directed to transfer the status of the equipment to the computer. When this bit is set on a multiplexing device, and both the station select code and Read signal are present, the data lines are directed to transfer status of the station to the computer. Additional bits of Q may be used to select status levels, e.g., interrupt conditions or addresses.

### Status Bit Definitions



Ready

Bit 0 indicates that an equipment is Ready and an operation can be performed when requested by a Start request. Once Ready, an equipment remains so until operation is no longer possible. An equipment cannot become Not Ready while information transfer is actually in progress. Those equipments which require manual intervention must be made Ready manually.

Busy

Bit 1 indicates that an equipment is Busy, or in operation. The equipment becomes Busy immediately upon initiation of the Start operation if the operation can be performed. Normally, an equipment remains Busy until it has finished all activity and is able to perform another operation.

Interrupt

Bit 2 indicates an interrupt response has been sent from this controller. Other bits must be monitored to determine the cause of the interrupt.

Data

Bit 3 indicates that the controller is ready to perform a data transfer. If a Data Interrupt had been selected, this bit also indicates the type of interrupt which has occurred.

End of Operation

Bit 4 indicates an End of Operation which means continuous transfers of data can no longer occur. It may also indicate the source of the interrupt response if the request had been selected. Each equipment specifies the particular conditions which constitute an End of Operation.

Alarm Interrupt

Bit 5 indicates an alarm which may be any one of several conditions. See reference information for each equipment for the specific conditions.

Lost Data

Bit 6 indicates that data may have been lost. This occurs when the computer does not service the controller within the prescribed time for the device. This loss should be detected and displayed as Lost Data. This may be a condition for an Alarm interrupt.

Protected

Bit 7 indicates that the Program Protect switch for an equipment has manually been placed in the Protected position.

Parity Error

Bit 8 indicates that a Parity Error has occurred in those storage devices that do incorporate parity as part of their format.

Unassigned

Bits 9-15 are unassigned and may be used at the discretion of the designer. Where more practical, it may be desirable to assign another status level in the address and repeat use of the lower bit transmitters.

## INTERRUPTS

### Interrupt Signals

Interrupt on Data

Director function codes set and clear this interrupt request. On a Read operation, the interrupt occurs when data has been loaded into the Data Hold register and is ready for transfer to the computer. The interrupt response is cleared by the reply to data transfer. On a Write operation, the interrupt occurs when data can be loaded into the Data Hold register of the output device. The interrupt response is cleared by the reply to data transfer. A status bit indicates the condition of the interrupt.

### Interrupt on End of Operation

A director function sets this interrupt request. Another director function clears the interrupt request and response. The operation may or may not be in progress at the time of the selection. The interrupt cannot occur from an operation which has ended before the selection was made. An operation and an End of Operation must be defined for each peripheral device. A status bit indicates the condition of the interrupt.

### Interrupt on Alarm

A director function code sets this interrupt request. Another director function code clears the interrupt request and response. An alarm condition that exists at the time of the interrupt request immediately provides a response. The alarm conditions must be defined for each peripheral device. A status bit should indicate the state of each alarm condition.

## INPUT/OUTPUT ON THE BASIC PERIPHERAL DEVICES

Basic peripherals for the System Controller are defined as those which do not require a AQ Interrupt Data Channel for hookup to the AQ channel. There are two basic peripherals available to the System Controller, the Teletypewriter I and II. The basic peripherals are equipment number one and operate without an I/O channel by using internal signals provided by the System Controller. Each station is individually selectable and all may be in operation at one time.

### Standard and Basic Peripherals

In addition to the basic peripherals there are standard peripherals which require a AQ Interrupt Data Channel and possibly a Direct Storage Access for hookup to the System Controller. The 1706 Buffered Data Channel is a converter and connects to the AQ Interrupt Data Channel and the Direct Storage Access. It performs buffered operations through the DSA and single word transfers through the AQ channel.

The 1731 Magnetic Tape Controller is shown twice in Figure 5-3 to show that it can be used either in conjunction with the AQ channel or the 1706. If the magnetic tape controller is attached to the AQ channel because the 1706 is not present, data can be transferred from the magnetic tape recorded at 200 or 556 bpi.

Input/output requires that the address of the peripheral device be placed in the Q register. The A register either contains a function code, receives status bits, or serves to transfer and receive data.

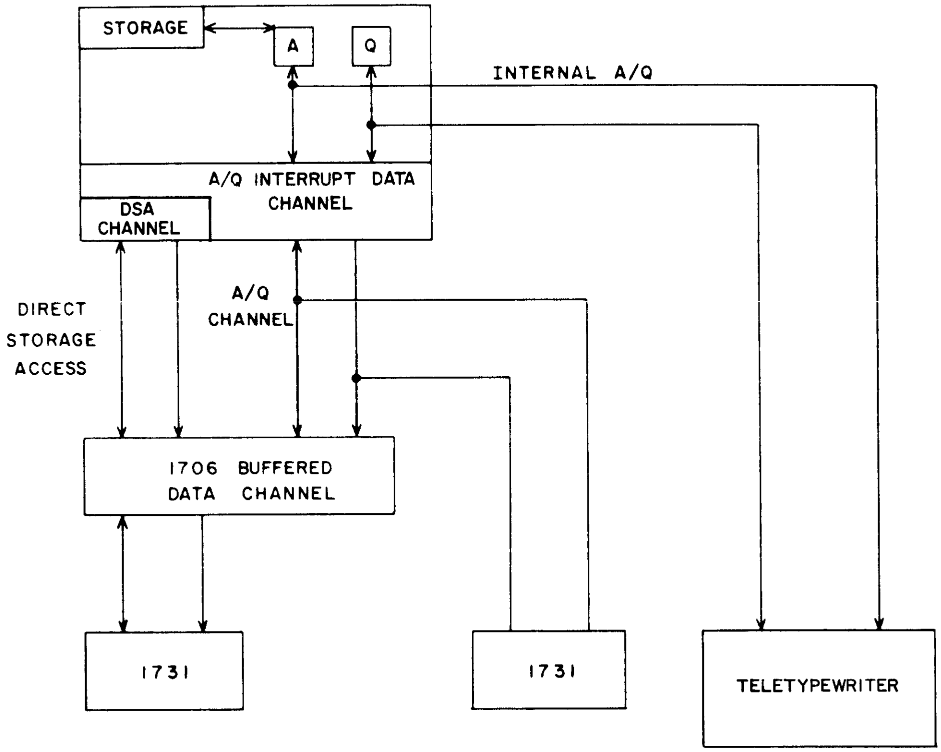


Figure 5-3. Graphic Presentation of Input/Output Devices

The codes for addressing the teletypewriter are 0090 and 0091.

The sequence of commands, as an illustration, would proceed as follows:

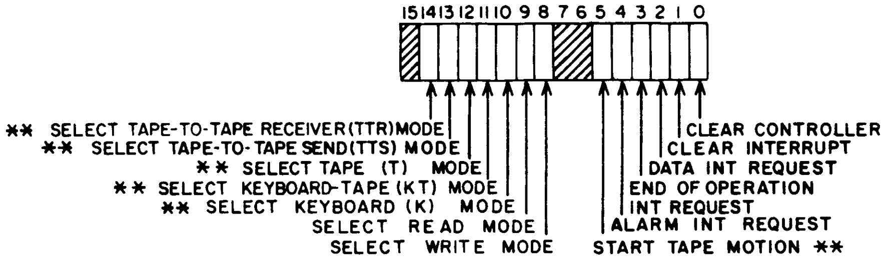
- 0091 in the Q register followed by an OUT instruction results in recognition of the function code set in the A register.
- 0091 in the Q register followed by an INP instruction results in status bits being set in the A register.
- 0090 in the Q register followed by an INP instruction results in the input of 8 data bits from the Data Hold register of the teletypewriter to the A register of the computer.

## Teletypewriter I and II

The teletypewriter is the basic peripheral for the System Controller. It comes in two varieties: TTY I and TTY II. The TTY I consists of a keyboard and a printer. The TTY II consists of a keyboard, printer, paper tape reader, paper tape punch, and remote mode selection features.

Director Function

When bit 0 of the Q register is a "1" (e.g., 0091) and the computer executes an OUT instruction, bits in the A register control the functions of the teletypewriter as follows:



The following functions are accepted unconditionally:

Clear Controller (A0 is "1"): Clears all interrupt requests, motion requests, errors, and other logic which may be cleared.

Clear Interrupts (A1 is "1"): All interrupt requests and responses are cleared. Any interrupt request bit takes precedence.

Data Interrupt Request (A2 is "1"): An interrupt is generated when an information transfer can occur. The interrupt is cleared by reply to a data transfer.

Select Interrupt on End of Operation (A3 is "1"): This function conditions the controller to send an interrupt signal when the TTY is not busy.

Select Interrupt on Alarm (A4 is "1"): This interrupt notifies the computer that the teletypewriter is Not Ready, or has lost data.

The following functions are accepted only if the TTY is not busy:

*Start Tape Motion (A5 is "1"): This causes the paper tape reader to advance by one character and transmit it to the controller. Paper motion stops after one character.

Select Write Mode (A8 is "1"): The controller is conditioned for an Output operation on the teletypewriter. This function is rejected if the controller is Busy. The controller accepts a word of data from the computer every 100 milliseconds.

Select Read Mode (A9 is "1"): The controller is conditioned for an Input operation. The function is rejected if the controller is Busy. Eight bits of data can be provided by the controller to the computer at a maximum rate of once every 100 milliseconds, depending on the operation of the teletypewriter. The Clear Controller function puts the controller in a Read mode.

*These Director Functions are used on the TTY II only.

*Select Keyboard (K) Mode (A10 is "1"): Places the TTY II in the Keyboard Mode of operation.

*Select Keyboard-Tape (KT) Mode (A11 is "1"): Places the TTY II in the Keyboard and Tape Mode of operation.

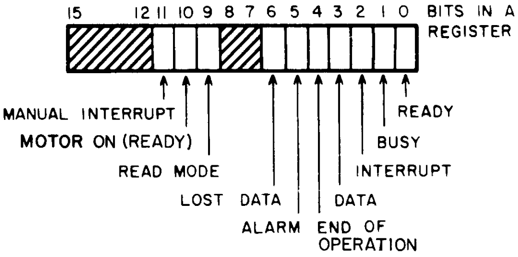
* Select Tape (T) Mode (A12 is "1"): Places the TTY II in the Tape Mode of operation.

* Select Tape-to-Tape Send (TTS) Mode (A13 is "1"): Places the TTY II in the Tape-to-Tape Send Mode of operation.

* Select Tape-to-Tape Receive (TTR) Mode (A14 is "1"): Places the TTY II in the Tape-to-Tape Receive Mode of operation.

Director Status

When bit 0 of the Q register is a "1" (e.g., 0091) and the computer executes an INP instruction, bits in the A register show the status of the teletypewriter as follows:



Ready (A0 is "1"): If this bit is set in the A register, the Power switch on the console of the teletypewriter is in the ON-LINE position identical to A10. See A10 for further description.

Busy (A1 is "1"): If this bit is set, one or more of the following conditions exist:

a. The controller is in Read mode and is in the process of receiving a character from the teletypewriter or the Data Hold register contains data for transfer to the computer. The Busy status drops upon completion of the transfer to the computer if data has not been lost. If data has been lost, the controller requires 200 milliseconds to stop the teletypewriter and remains Busy all this time.

b. Write mode and the Data Hold register contains data and is in the process of transferring it to the teletypewriter. Busy drops upon completion of the transfer.

Interrupt (A2 is "1"): An interrupt condition exists. Other bits must be monitored to determine the condition causing this interrupt.

Data (A3 is "1"): An interrupt is generated and this status bit is a "1" under the following conditions:

a. Read mode and the Data Hold register contains data for transfer to the computer. The status drops upon completion of a Read.

b. Write mode and the controller is ready to accept another Write from the computer. The status drops upon completion of the Write.

*These Director Functions are used only on the TTY II.

End of Operation (A4 is "1"): This status is equivalent to a Not Busy status.

Alarm (A5 is "1"): The teletypewriter is not in a Ready state or has lost data.

Lost Data (A6 is "1"): The controller was not serviced by the computer before a new character was sent by the teletypewriter. The keyboard and tape transmitter are locked out. The status bit indicates a Lost Data condition, and a Busy status indicates that the process of stopping the teletypewriter is in progress. Data held in the Data Hold register is not disturbed, but the incoming data is ignored. The lost data status can be cleared by a Clear Controller or a Select Write Mode command. These two functions are rejected while the controller is stopping the teletypewriter. The Select Write Mode command must be preceded by a Read operation to clear the Data Hold register. After the teletypewriter has stopped, the computer may do an Output operation to notify the controller of the Error condition.

Read Mode (A9 is "1"): If this bit is a "1", the controller is conditioned for an Input operation from the teletypewriter.

Ready (A10 is "1"): The teletypewriter is in the on-line mode.

Manual Interrupt (A11 is "1"): A manual interrupt has occurred at the teletypewriter. When the Manual Interrupt switch on the teletypewriter console is operated, an interrupt occurs and the status bit is set. The manual interrupt is not a selectable condition and depends entirely on the interrupt mask (M register) in the computer for recognition. The condition can be cleared with the Clear Controller or Clear Interrupts function.

Data Transfer

When the Director code is a "0" (e.g., 0090) and the computer performs either an Input (INP) or Output (OUT) instruction, the teletypewriter sends or receives 8 bits of data. The mode of the teletypewriter must be selected with a function code but a Clear Controller function automatically puts it in Read mode.

Switches

LOCAL/OFF/ON-LINE Switch: This three-position rotary switch is located to the right of the keyboard. It is the Power switch; however, the TTY can not be operated unless the System Controller has power to its logic chassis.

| Position | Function |
|---|---|
| OFF | No power is applied to the teletypewriter. A Not Ready signal is sent to the controller. |
| LOCAL | This position allows the teletypewriter to be used as an off-line device, similar to an electric typewriter. A Not Ready signal is sent to the controller and no data can be transferred to or from the controller. In the local position the TTY is in the K mode and paper tape operations cannot be performed. |
| ON-LINE | The teletypewriter is capable of communicating with the computer. A Ready signal is sent to the controller. |

Mode Switch/Indicators: These five pushbutton switch/indicators are located to the left of the keyboard. They control the combination of devices (printer, keyboard, reader, punch) which is capable of communicating with the computer. These switches are on the TTY II only.

Mode — Function

K — Keyboard mode

This is the equivalent of a TTY I; the keyboard and printer are on line. Data may be sent to the controller from the keyboard and is simultaneously printed on the printer. Data may be sent to the printer from the controller.

KT — Keyboard/Tape Mode

Data sent from the keyboard is punched on tape, printed by the printer, and transmitted to the controller simultaneously. Data sent from the paper tape reader is printed, punched on tape, and sent to the computer. Data sent from the computer is punched on tape and printed.

CAUTION

The tape reader and keyboard are both active. Striking a key while the reader is running may result in a garbled character.

T — Tape Mode

This position provides two independent devices. The keyboard and punch are provided as an off-line tape preparation device. No printed copy of the message is provided; however, the Character counter and red End of Line indicator just above the keyboard may be used for proper positioning of the message.

The reader and printer are provided as an on-line device for normal sending to and receiving from the controller. The transmission from the reader is also printed by the printer.

TTS — Tape-to-Tape Send Mode

This mode is provided to allow eight-level tapes punched in codes other than ASCII to be transmitted to the controller. The printer is inactive in this mode to prevent printing a garbled message and behaving in an erratic fashion.

The keyboard and punch are provided as an off-line tape preparation device.

TTR — Tape-to-Tape Receive Mode

This mode allows an eight-level tape to be punched from output from the controller in codes other than ASCII.. Only the punch is active in this mode.

Indicators

The printer is capable of printing 72 columns. An End of Line indicator and Character counter are provided to aid in preparing tape off-line for later transmission. Both are located above the keyboard. The End of Line indicator lights at character 65 and remains lighted to the end of the line.

## Signal Coding

The coding used on the TTY I and TTY II is the ASCII (1968) version. This coding consists of eight binary bits shown below:

Example: The bit representation for the character K, positioned in column 4, row 11 is:

| B8 | B7 | B6 | B5 | B4 | B3 | B2 | B1 |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |

The eighth bit is the parity; it either marks or spaces in order to provide an even number of marking pulses for each combination.

The following table shows all possible bit combinations and the TTY interpretation of them. It should be noted that not all characters are printed on the printer; only those in columns 2, 3, 4, and 5 are printed but all combinations of bits will be punched. Also, not all control signals shown are used by the TTY; those not used are indicated.

TABLE 5-2. ASCII (1968) TELETYPEWRITER CODES

| BITS | b7→<br>b6→<br>b5→ | 0<br>0<br>0 | 0<br>0<br>1 | 0<br>1<br>0 | 0<br>1<br>1 | 1<br>0<br>0 | 1<br>0<br>1 | 1<br>1<br>0 | 1<br>1<br>1 |
|---|---|---|---|---|---|---|---|---|---|
| b4 b3 b2 b1<br>↓ ↓ ↓ ↓ | COLUMN→<br>ROW↓ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 0 0 0 | 0 | NUL | DLE | SP | 0 | @ | P |  | P |
| 0 0 0 1 | 1 | SOH[1] | DC1/<br>XON[2,3] | ! | 1 | A | Q | a | g |
| 0 0 1 0 | 2 | STX[2] | DC2/<br>TAPE[2,3] | " | 2 | B | R | b | r |
| 0 0 1 1 | 3 | ETX[1] | DC3/<br>X OFF[2,3] | # | 3 | C | S | C | S |
| 0 1 0 0 | 4 | EOT | DC4/<br>~~TAPE~~[2,3] | $ | 4 | D | T | d | t |
| 0 1 0 1 | 5 | ENQ/<br>WRU[2,3] | NAK[1] | % | 5 | E | u | e | u |
| 0 1 1 0 | 6 | ACK[1] | SYN[1] | & | 6 | F | V | f | v |
| 0 1 1 1 | 7 | BELL | ETB[1] | ' | 7 | G | W | g | w |
| 1 0 0 0 | 8 | BS[1] | CAN[1] | ( | 8 | H | X | h | x |
| 1 0 0 1 | 9 | HT/TAB[3] | EM[1] | ) | 9 | I | Y | i | y |
| 1 0 1 0 | 10 | LF | SUB[1] | * | : | J | Z | j | z |
| 1 0 1 1 | 11 | VT | ESC[1] | + | ; | K | [ | k | { |
| 1 1 0 0 | 12 | FF[1] | FS[1] | , | < | L | \ | 1 | ¦ |
| 1 1 0 1 | 13 | CR | GS[1] | - | = | M | ] | M | } |
| 1 1 1 0 | 14 | SO[1] | RS[1] | . | > | N | ∧ | n | ~ |
| 1 1 1 1 | 15 | SI[1] | 1<br>DU[1] | / | ? | O | – | o | DEL/<br>RUB-<br>OUT[2,3] |

Only these characters will be printed by the printer

[1] This control signal does not affect printer but is punched on tape.

[2] This control signal may be generated by keyboard but will not affect printer when generated by System Controller; it will be punched on tape.

[3] Teletypewriter Corp.'s symbol for this signal.

TABLE 5-3. ASCII (1968) CONTROL SIGNALS

| COLUMN/LINE POSITION | SYMBOL | DEFINITION |
|---|---|---|
| 0/0 | NUL | Null (two successive nulls lock keyboard and stop tape reader) |
| 0/1 | SOH | 1. |
| 0/2 | STX | 1. |
| 0/3 | EXT | 1. |
| 0/4 | EOT | End of transmission (shuts off motors) |
| 0/5 | ENQ/WRU | 1. |
| 0/6 | ACK | 1. |
| 0/7 | BELL | Bell (audible or attention signal) |
| 0/8 | BS | 1. |
| 0/9 | HT/TAB | Horizontal tab |
| 0/10 | LF | Line Feed |
| 0/11 | VT | Vertical tab |
| 0/12 | FF | 1. |
| 0/13 | CR | Carriage Return (does not advance paper) |
| 0/14 | SO | 1. |
| 0/15 | SI | 1. |
| 1/0 | DLE | 1. |
| 1/1 | DCI/XON | 1. |
| 1/2 | DC2/TAPE | 1. |
| 1/3 | DC3/XOFF | 1. |
| 1/4 | DC4/~~TAPE~~ | 1. |
| 1/5 | NAK | 1. |
| 1/6 | SYN | 1. |
| 1/7 | ETB | 1. |
| 1/8 | CAN | 1. |
| 1/9 | EM | 1. |
| 1/10 | SUB | 1. |
| 1/11 | ESC | 1. |
| 1/12 | FS | 1. |
| 1/13 | GS | 1. |
| 1/14 | RS | 1. |
| 1/15 | US | 1. |
| 7/15 | DEL/RUB-OUT | Delete (Punches all levels on paper tape, no effect on printer) |

1. No control function on the TTY I or TTY II

The Break indicator is located to the right of the keyboard. It indicates that the keyboard has been locked and that the reader has been stopped under remote control. The keyboard and reader may be reactivated by pressing the red BRK REL (Break Release) key located on the lower-left portion of the keyboard.

Pressing the Break indicator/switch turns the motor on if the teletypewriter is in the On-Line condition and the controller is in Read mode.

## Programming Considerations

The simplest operation on the peripheral devices is an input or output without monitoring status or relying on interrupts (e.g., the bootstrap paper tape loading routine). The INP instruction loops on itself in case of a Reject from the paper tape reader and waits until another frame can be transferred. Input or output routines can be written to monitor status bits in the A register to observe the condition of the peripheral device. Input of status information changes the contents of the A register and thus packing of incoming frames into words must be done through storage. The programmer can take advantage of the interrupts to operate the peripheral equipment. The routine processing of the selected interrupts can determine from the status bits what caused the condition.

An interrupt from the basic peripheral device causes the computer to store the current address in core location 0104. It then reads the next instruction in location 0105. The interrupt system is deactivated and the programmer has the option of processing this particular interrupt to completion or reactivating the interrupt immediately after certain housekeeping functions have been performed. The housekeeping function should include setting the Mask register bit 01 to "0" to insure a successful return to the main program. The programming of the interrupt system is explained in detail in Section 4.

If the paper tape reader is operated by monitoring the data status bit, it becomes imperative to empty the Data Hold register promptly to insure continuous operation. If the programmer fails to transfer the 8 bits from the Data Hold register after the status bit is set, the reader comes to a halt before reading the next frame. This feature can be used to read one frame at a time. The lost data status does not occur because the reader stops before reading the next frame.

If the paper tape reader is operated through an interrupt when the Data Hold register is full, it is important to empty it promptly to avoid a Lost Data condition. In this mode of operation, the reader does not stop before reading the next frame and thus causes a Lost Data condition. Time between frames is 2.857 milliseconds.

The teletypewriter also can have a Lost Data condition which should be avoided by the programmer. Since paper tape does not move on the paper tape punch without having data in the Data Hold register, no lost data can occur.

The mechanical action of the carriage and paper on the teletypewriter requires that a fill character be sent out on form, vertical and horizontal tab, or carriage return. This can be accomplished by sending a rub out (all "1" bits). The teletypewriter, unlike the typewriter, does not move paper on a carriage return command. To space the paper, a carriage return must be followed by line feed. A fill character is not needed if the carriage return and line feed are issued together (line feed is equivalent to a fill character). The carriage return, if sent alone, requires a fill character similar to form, vertical or horizontal tab.

The Manual Interrupt on the teletypewriter is always selected and depends on the state of the M register for recognition. An Interrupt may occur if the Manual Interrupt switch is operated; the programmer should make provisions in the interrupt routine for handling it.

The teletypewriter controller responds to the computer even if no teletypewriter is attached or it is in the Local or Off condition. Visual verification of the condition of the teletypewriter by the operator is essential. Ready status tells if the teletypewriter is not on line, present, or the motor is on.

The teletypewriter has no Protect switch. It may be addressed by either a protected or a nonprotected instruction.

Loading a Paper Tape Bootstrap Routine into the Autoload Area of Memory:

1. Turn power on.
2. Operate CLEAR switch.
3. Press AUTO LOAD PROTECT switch.
4. Press P REGISTER SELECT switch.
5. Press CLEAR pushbutton to set P to 7FEO.
6. Set ENTER/SWEEP switch to ENTER.
7. Press X REGISTER SELECT switch.
8. Press CLEAR pushbutton to clear X register.
9. Enter first word of program (e.g., 68FE).
10. Set RUN/STEP switch to STEP one time.
11. Repeat steps 7 through 10 until the bootstrap routine is entered.
12. Return ENTER/SWEEP switch to center position.
13. Operate Master Clear switch.
14. Set SELECTIVE STOP switch and SELECTIVE SKIP switch if needed.
15. Insert paper tape to be read on the 1721 Paper Tape Reader.
16. Turn reader on.
17. Press READY switch on reader.
18. Press AUTO LOAD.

The program starts to read paper tape and after bootstrapping in the program starts execution at location 0000 in core storage.

```
7FEO   68FE              Address
7FE1   E000   Start      LDQ      =N$A1       Prepare for output of
                                              function
7FE2   00A1
7FE3   0A20              ENA      $20         Bit 5 set to Start Motion
7FE4   03EE              OUT      -1          Output, On Reject Loop
                                              here
7FE5   0DFE              INQ      -1          Increase Q by -1 to set
                                              Input
7FE6   02FE   Loadl      INP      -1          Input one frame
7FE7   0112              SAN      2           Check if this was blank
                                              Leader
7FE8   18FD              JMP*     Loadl       If blank Leader, Loop
7FE9   02FE   Load2      INP      -1          Not blank, input more
7FEA   0FC8              ALS      8           Shift Frame left
7FEB   02FE              INP      -1          Bring in second frame
                                              of word
7FEC   6CF2              STA*     (Addres)    Store in Core Storage
7FED   0103              SAZ      Exit-*-1    If word was zero, Exit
7FEE   D8FO              RAO*     Addres      Increase storage address
7FEF   18F9              JMP*     Load2       Loop, word was not zero
7FFO   1400   Exit       NUM      $0
7FF1   0000                                   Jump
```

# MANUAL CONTROLS AND INDICATORS 6

Section 6 describes the operation of the manual controls for the System Controller.

## POWER AND TEMPERATURE CONTROLS

These controls are not mounted on the control panel of the console.

### Power On

This switch, located on the power distribution panel, provides power to the System Controller.

### High Temp

This switch indicates that the temperature of the System Controller is above the safe operating temperature.

## CONSOLE CONTROLS

Switches and indicators mounted on the console are shown in Figure 6-1.

### Register Select (1)

The M, P, Y, X, A, and Q registers are available for manual entry of values via the Register Bit switches. This six-pushbutton switch/indicator selects the register for display and entry.

### Register Bit (2)

The contents of the register selected by the Register Select switch are displayed and may be changed by these switch/indicators. To enter a value into a register, select the register using the Register Select switch, press the Clear pushbutton to clear that register, and then set bits using the 16 indicator pushbuttons.

### Instruction Sequence/Remote Auto Load (3-7)

When an instruction is being stepped, this group of five indicators describes the meaning of the storage reference just completed. The data of the storage reference (read or write) is in the X register. The five indicators and description when lighted are:

INSTRUCTION*: The contents of the X register is an instruction.

•INDIRECT ADDRESS: The contents of the X register is the result of indirect addressing. The indirect address may also be another indirect address, hence, this indicator may remain lighted for several consecutive storage references.

•STORAGE INDEX: The contents of the X register is the value of the Storage Index register.

•OPERAND: The contents of the X register is the value of the operand either written into or read from storage.

•PROGRAM PROTECT: The program protect bit of the last word read from storage is set.

If more than one Instruction Sequence indicator is lighted, the computer is running. If only one indicator is lighted, the computer is not running or is in a rather unlikely program loop which does not use operands, the storage index, or indirect addressing.

*This is a switch/indicator; when pressed it operates parallel to the Auto-Load switch on a mass memory device.

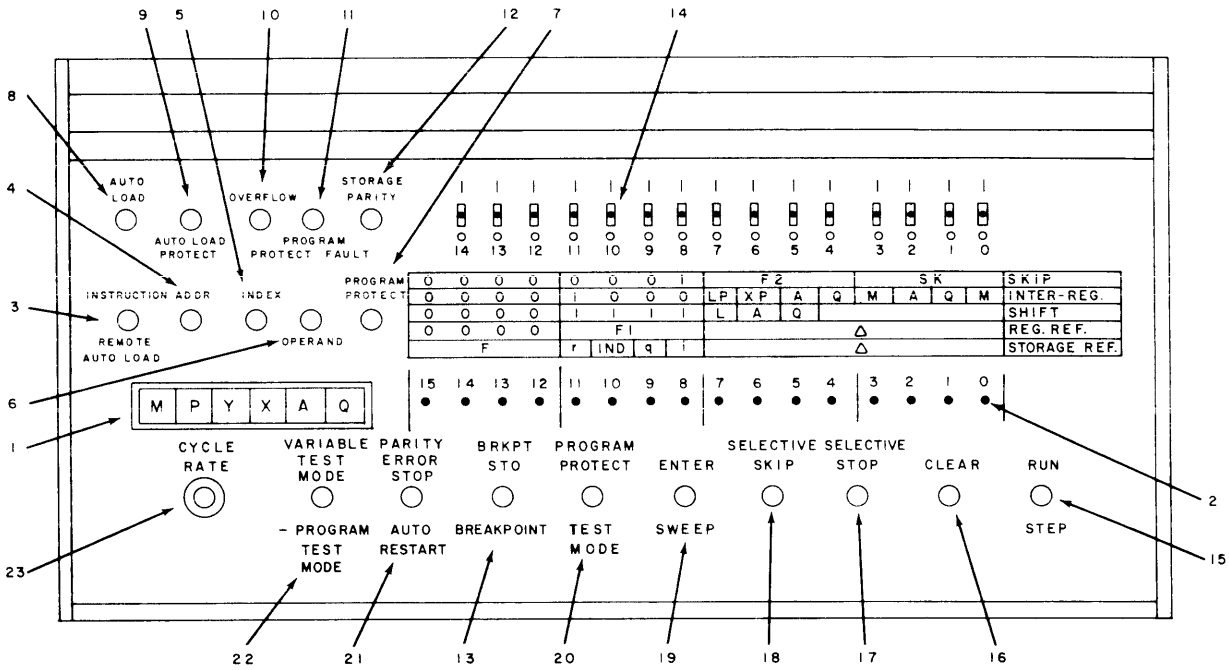


Figure 6-1. System Controller Console

**Auto Load (8)**

This momentary switch activates the auto load feature. When pressed, the sequence of events is as follows: the computer and all peripherals are Master Cleared, the program address registers are set to 7FE0, and the computer is in Run to RNI from that address.

Locations 7FE0 through 7FFE would be protected by the use of the program protect bit in storage and a console Auto load Protect switch.

**Auto Load Protect (9)**

This switch will deactivate the auto load protect system. The auto load protect is activated by an M/C when this switch is lit.

**Faults (10-12)**

There are three fault indicators; when lighted, the fault condition is present.

- Overflow: An arithmetic register overflow has occurred.
- Program Protect: A violation of the program protect system has been detected.
- Storage Parity: A parity error has been detected in an operand or instruction read from storage.

**Breakpoint (13)**

This switch activates the breakpoint feature of the System Controller. Breakpoint will allow the computer to stop at any predetermined address in memory. The address is selected by the Breakpoint Address switches. The up position of the Breakpoint Control switch will stop the computer on a store operand only, and the down position will stop on either read or store operand or RNI.

**Breakpoint Address (14)**

These fifteen switches allow selection of any address in 32K of memory as the breakpoint address.

**Run/Step (15)**

This is a three-position switch. When the switch is momentarily placed in the Run position, the computer begins program execution, starting with the instruction whose address is in the P register. The computer is stopped by momentarily placing the switch in the Step position.

If the switch is repeatedly placed in the Step position, the computer steps through the program, stopping after each storage reference. The significance of the storage reference just made is indicated by the Instruction Sequence indicators (INSTRUCTION, INDIRECT ADDRESS, etc. ).

**Master Clear (16)**

This is a three-position switch. Master Clear is executed whenever it is momentarily operated either up or down. A Master Clear returns the computer and peripheral devices to initial conditions.

**Selective Stop (17)**

This is a three-position switch. The computer stops when it executes a Selective Stop instruction if this switch is either in the up or down position. The up position is maintained; the down position is momentary.

**Selective Skip (18)**

This is a three-position switch. Two Selective Skip instructions (SWS and SWN) are conditioned by this switch. This switch is off in the center position; the up position is maintained; the down position is momentary.

**Enter/Sweep (19)**

This is a three-position switch maintained in all three positions. The center position is off.

ENTER

The Enter position selects the enter mode. In this mode, each step operation of the Run/Step switch stores the contents of the X register at the location specified by P+1 and then advances the P register by one. The first step after a Master Clear or Clear P stores the contents of the X register at the location specified by P.

To store a few instructions in unprotected storage, proceed as follows:

a. Power is on but computer is stopped.

b. Operate Master Clear switch.

c. Press P Register Select switch and Clear pushbutton in that order. Set desired address for instruction in P by use of indicator pushbuttons.

d. Set Enter/Sweep switch to Enter.

e. Press X Register Select switch.

f. Press Clear pushbutton, then enter word to be stored by use of indicator pushbuttons.

g. Move Run/Step switch to Step one time.

To store additional words in successive storage locations, repeat steps f and g until finished. To change to a new sequence of addresses, start at step two for the first one, then repeat steps f and g for each successive word.

A lighted indicator pushbutton indicates a "1" and a dark indicator pushbutton indicates a "0".

SWEEP

The Sweep position selects the Sweep mode. In this mode, each step operation of the Run/Step switch enters in the X register the contents of the storage location whose address is P+1. The P register is advanced by one after each Step operation. The first step after a Master Clear or Clear P displays the location specified by P. Instructions are not executed.

**Program Protect / Test Mode (20)**

This is a three-position switch maintained in all positions. The center position is off.

Program Protect

The Program Protect position selects program protection.

Test Mode

The Test Mode position selects Test mode. When in test mode, the computer executes the following sequence of events:

a. A 20-usec Master Clear. This clears the P register and all other operational registers (all bits are "0").

b. A 100-usec program run, starting from program address $0000_{16}$.

c. A return to step a and repeat.

**Parity Error Stop/ Auto Restart (21)**

This is a three-position switch with the center position off. If the switch is in the up position and the computer is in the sweep mode and running, the computer will stop at P+1 of those instructions, or instructions which reference locations with parity errors.

If the switch is in the down position, it will automatically recycle logic power, Master Clear, and place the system in Run to RNI from address 0000. This will occur when source power is restored after a power failure.

**Variable Test Mode/ Program Test Mode and Cycle Rate (22-23)**

The controls for the variable test modes feature consist of a three-position switch (Variable Test Mode/Program Test Mode) and a rotational-controlled potentiometer (Cycle Rate). In the up position the switch selects variable test mode. In this mode, the potentiometer controls the total cycle time (variable to 90 ms. maximum.) In the down position, the three-position switch selects the programmable mode. In this mode, the cycle length is determined by a Selective or Breakpoint stop. Variable test mode is an extension of test mode. The center position of the Variable Test Mode switch is off.

# APPENDIX SECTION

# SYSTEM CONTROLLER INSTRUCTION EXECUTION TIMES A

REGISTER REFERENCE

| MNEMONICS | INSTRUCTION | EXECUTION TIME (usec)* |
|---|---|---|
| LDA | Load A | 3.0 |
| STA | Store A | 3.0 |
| LDQ | Load Q | 3.0 |
| STQ | Store Q | 3.0 |
| ADD | Add A | 3.0 |
| SUB | Subtract | 3.0 |
| ADQ | Add Q | 3.0 |
| AND | AND with A | 3.0 |
| EOR | Exclusive OR with A | 3.0 |
| RAO | Replace Add One in Storage | 4.5 |
| MUI | Multiply Integer | 20.0 |
| JMP | Jump | 1.5 |
| RTJ | Return Jump | 3.5 |
| DVI | Divide Integer | 30.0 |
| SPA | Store A, Parity to A | 3.0 |

* Add 0.75 usec if Index Register is used.
Add 1.5 usec for each level of Indirect Addressing.
Add 0.5 usec for Q indexing.

## REGISTER REFERENCE

| MNEMONICS | INSTRUCTION | EXECUTION TIME (usec) |
|---|---|---|
| SLS | Selective Stop | 1.5 |
| INP | Input to A | 2.0 min., 10 max. |
| OUT | Output from A | |
| ENA | Enter A | 1.5 |
| ENQ | Enter Q | 1.5 |
| INA | Increase A | 1.5 |
| INQ | Increase Q | 1.5 |
| ARS | A Right Shift | 1.5 + shift count (0.5) |
| QRS | Q Right Shift | |
| ALS | A Left Shift | |
| QLS | Q Left Shift | |
| LRS | Long Right Shift | 1.5 + shift count (1.0) |
| LLS | Long Left Shift | |
| NOP | No Operation | 1.5 |
| EIN | Enable Interrupt | 1.5 |
| IIN | Inhibit Interrupt | 1.5 |
| EXI | Exit Interrupt State | 3.5 |
| SPB | Set Program Protect | 3.5 |
| CPB | Clear Program Protect | 3.5 |
| ECA | Enable Character Addressing | 1.5 |
| DCA | Disable Character Addressing | 1.5 |

## INTERREGISTER
## REGISTER REFERENCE

| MNEMONICS | INSTRUCTION | EXECUTION TIME (usec) |
|---|---|---|
| SET | Set to Ones | 1.5 |
| CLR | Clear to Zero | |
| TRA | Transfer A | |
| TRM | Transfer M | |
| TRQ | Transfer Q | |
| TRB | Transfer Q + M | |
| TCA | Transfer Complement A | |
| TCM | Transfer Complement M | |
| TCQ | Transfer Complement Q | |
| TCB | Transfer Complement Q + M | |
| AAM | Transfer Arithmetic Sum A, M | |
| AAQ | Transfer Arithmetic Sum A, Q | |
| AAB | Transfer Arithmetic Sum A, Q + M | |
| EAM | Transfer Exclusive OR of A, M | |
| EAQ | Transfer Exclusive OR of A, Q | |
| EAB | Transfer Exclusive OR of A, Q + M | |
| LAM | Transfer Logical Product A, M | |
| LAQ | Transfer Logical Product A, Q | |
| LAB | Transfer Logical Product A, Q + M | |
| CAM | Transfer Complement Logical Product A, M | |
| CAQ | Transfer Complement Logical Product A, Q | |
| CAB | Transfer Complement Logical Product A, Q + M | |

## SKIPS

| INSTRUCTION | | EXECUTION TIME (microseconds) |
|---|---|---|
| SAZ | Skip if A = +0 | 2.0 |
| SAN | Skip if A ≠ +0 | |
| SAP | Skip if A = + | |
| SAM | Skip if A = - | |
| SQZ | Skip if Q = +0 | |
| SQN | Skip if Q ≠ +0 | |
| SQP | Skip if Q = + | |
| SQM | Skip if Q = - | |
| SWS | Skip if Switch Set | |
| SWN | Skip if Switch Not Set | |
| SOV | Skip on Overflow | |
| SNO | Skip on No Overflow | |
| SPE | Skip on Storage Parity Error | |
| SNP | Skip on No Storage Parity Error | |
| SPF | Skip on Program Protect Fault | |
| SNF | Skip on No Program Protect Fault | |

## FUNCTION LISTING OF INSTRUCTIONS

| FUNCTION | MNEMONIC CODE | INSTRUCTION DESCRIPTION |
|---|---|---|
| Transfers | LDA | Load A (Storage Reference) |
| | STA | Store A (Storage Reference) |
| | LDQ | Load Q (Storage Reference) |
| | STQ | Store Q (Storage Reference) |
| | SPA | Store A, Parity to A (Storage Reference) |
| | ENA | Enter A (Register Reference) |
| | ENQ | Enter Q (Register Reference) |
| | TRA | Transfer A (Register Reference) |
| | TRM | Transfer M (Register Reference) |
| | TRQ | Transfer Q (Register Reference) |
| Arithmetic | ADD | Add A (Storage Reference) |
| | SUB | Subtract (Storage Reference) |
| | ADQ | Add Q (Storage Reference) |
| | RAO | Replace Add One in Storage (Storage Reference) |
| | MUI | Multiply Integer (Storage Reference) |
| | DVI | Divide Integer (Storage Reference) |
| | INA | Increase A (Register Reference) |
| | INQ | Increase Q (Register Reference) |
| | SET | Set to Ones (Register Reference) |
| | TRB | Transfer Q + M (Register Reference) |
| | AAM | Transfer Arithmetic Sum A, M (Register Reference) |
| | AAQ | Transfer Arithmetic Sum A, Q (Register Reference) |
| | AAB | Transfer Arithmetic Sum A; Q + M (Register Reference) |
| Logical | AND | AND with A (Storage Reference) |
| | EOR | Exclusive OR with A (Storage Reference) |
| | CLR | Clear to Zero (Register Reference) |
| | TCA | Transfer Complement A (Register Reference) |
| | TCM | Transfer Complement M (Register Reference) |
| | TCQ | Transfer Complement Q (Register Reference) |
| | TCB | Transfer Complement Q + M (Register Reference) |
| | EAM | Transfer Exclusive OR of A, M (Register Reference) |

FUNCTION LISTING OF INSTRUCTIONS (Cont'd)

| FUNCTION | MNEMONIC CODE | INSTRUCTION DESCRIPTION |
|---|---|---|
| Logical (Cont'd) | EAQ | Transfer Exclusive OR of A, Q (Register Reference) |
| | EAB | Transfer Exclusive OR of A, Q + M (Register Reference) |
| | LAM | Transfer Logical Product A, M (Register Reference) |
| | LAQ | Transfer Logical Product A, Q (Register Reference) |
| | LAB | Transfer Logical Product A, Q + M (Register Reference) |
| | CAM | Transfer Complement Logical Product A, M (Register Reference) |
| | CAQ | Transfer Complement Logical Product A, Q (Register Reference) |
| | CAB | Transfer Complement Logical Product A, Q + M (Register Reference) |
| Jumps & Stops | JMP | Jump (Storage Reference) |
| | RTJ | Return Jump (Storage Reference) |
| | SLS | Selective Stop (Register Reference) |
| | NOP | No Operation (Register Reference) |
| Decisions | SAZ | Skip if A = +0 (Register Reference) |
| | SAN | Skip if A = +0 (Register Reference) |
| | SAP | Skip if A = + (Register Reference) |
| | SAM | Skip if A = - (Register Reference) |
| | SQZ | Skip if Q = +0 (Register Reference) |
| | SQN | Skip if Q ≠ +0 (Register Reference) |
| | SQP | Skip if Q = + (Register Reference) |
| | SQM | Skip if Q = - (Register Reference) |
| | SWS | Skip if Switch set (Register Reference) |
| | SWN | Skip if Switch not set (Register Reference) |
| | SOU | Skip on overflow (Register Reference) |
| | SNO | Skip on no overflow (Register Reference) |
| | SPE | Skip on Storage Parity Error (Register Reference) |
| | SNP | Skip on no Storage Parity Error (Register Reference) |
| | SPF | Skip on Program Protect Fault (Register Reference) |
| | SNF | Skip on no Program Protect Fault (Register Reference) |

FUNCTION LISTING OF INSTRUCTIONS (Cont'd)

| FUNCTION | MNEMONIC CODE | INSTRUCTION DESCRIPTION |
|---|---|---|
| Shifts | ARS | A Right Shift (Register Reference) |
| | QRS | Q Right Shift (Register Reference) |
| | ALS | A Left Shift (Register Reference) |
| | QLS | Q Left Shift (Register Reference) |
| | LRS | Long Right Shift (Register Reference) |
| | LLS | Long Left Shift (Register Reference) |
| Input/Output | INP | Input to A (Register Reference) |
| | OUT | Output from A (Register Reference) |
| Interrupt | EIN | Enable Interrupt (Register Reference) |
| | IIN | Inhibit Interrupt (Register Reference) |
| | EXI | Exit Interrupt State (Register Reference) |
| Program Protect | SPB | Set Program Protect (Register Reference) |
| | CPB | Clear Program Protect (Register Reference) |
| Character Handling | ECA | Enables Character Addressing (Register Reference) |
| | DCA | Disables Character Addressing (Register Reference) |

# COMMENT SHEET

MANUAL TITLE CONTROL DATA SC-17 SYSTEM CONTROLLER

Reference Manual

PUBLICATION NO. 60270600 REVISION __________

**FROM:** NAME: __________

BUSINESS
ADDRESS: __________

## COMMENTS:

This form is not intended to be used as an order blank. Your evaluation of this manual will be welcomed by Control Data Corporation. Any errors, suggested additions or deletions, or general comments may be made below. Please include page number references and fill in publication revision level as shown by the last entry on the Record of Revision page at the front of the manual. Customer engineers are urged to use the TAR.

CUT ALONG LINE

PRINTED IN USA

NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

FOLD ON DOTTED LINES AND STAPLE

STAPLE

FOLD

FOLD

FIRST CLASS
PERMIT NO. 8241

MINNEAPOLIS, MINN.

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

POSTAGE WILL BE PAID BY

**CONTROL DATA CORPORATION**
**8100 34TH AVENUE SOUTH**
**MINNEAPOLIS, MINNESOTA 55440**

**ATTN: TECHNICAL PUBLICATIONS DEPT.**
**PLANT TWO**

CUT ALONG LINE

FOLD

FOLD

CONTROL DATA® SC-17 system controller Reference Manual Pub. No. 60270600