DAVID E. LEE

3300

COMPUTER SYSTEM

REFERENCE MANUAL

CONTROL DATA
CORPORATION

3300 CHARACTERISTICS

- Stored-program, solid-state, scientific and business data processing computer
- Time-sharing and multiprogramming features
- Parallel mode of operation
- Diode logic
- Character and word addressing (4 characters per word)
- Address modification (indexing)
- Indirect addressing
- 28-bit storage word (24 data bits and 4 parity bits)
- Nonvolatile magnetic core storage
- Complete cycle time: 1.25 microseconds
- Access time: 0.75 microsecond
- Storage sharing
- Selected storage protection
- Instruction repertoire compatible with the 3100, 3200, and 3500 Computers
- Business oriented Moves, Searches, Edit, Compare, Conversion, and BCD arithmetic instructions
- Logical and sensing operations
- Masked storage searches
- Block control operations
- Trapped instruction processing
- 24-bit accumulator register and auxiliary accumulator register
- Binary arithmetic: $2^{24}-1$ modulus, one's complement for all single precision (24-bit) operations and double precision (48-bit) addition and subtraction
- 64-word register file (0.5 microsecond cycle time)
- Complete interrupt system
- ASCII to BCD conversion (and vice versa) and 4-bit/6-bit packing
- Real-time clock (1.0 millisecond incrementation)
- Sit-down operator's console featuring: On-line typewriter and complete display and control system
- Upward compatability with 3100 and 3200 computer systems
- Standard 3000 Series type 12-bit bidirectional data channel
- Compatible I/O mediums include magnetic tape, disk file, punched cards, paper tape, and printed forms
- Options include:
  - Memory expansion to 262,144 words (over 1 million characters)
  - Additional 12-bit data channels or high-speed 24-bit data channels
  - Floating point and 48-bit precision multiply and divide hardware logic
  - Multiprogramming hardware module
  - Business Data Processor
  - Complete selection of advanced peripheral equipment

24·8·71

3300

COMPUTER SYSTEM

REFERENCE MANUAL

CONTROL DATA
CORPORATION

RECORD of REVISIONS

REVISION	NOTES
01 (11-16-65)	Original printing.
A (5-13-66)	Publications Change Order CA13641. Complete revision. All previous editions obsolete.
B	Publication Change Order 14387, no Product Designation change. The following pages were
(9-12-66)	revised or added: iii, v, vii, 1-3, 1-13, 1-14, 2-4, 2-5, 3-1 through 3-8, 4-3, 4-5, 4-8, 4-9,
	5-2, 5-7, 5-12, 5-13, 5-16, 5-18, 5-21, 5-22, 5-30, 5-41, 5-42, 5-68, 5-69, 5-73, 5-75, 5-76,
	5-79, 5-80, 5-82, 5-95.0, 5-95.1, 5-96.0, 5-96.1, 5-98 through 5-103, 5-105, 5-106, 5-114
	through 5-123, 5-129 through 5-137, 5-140 through 5-147, 5-149, 5-151, 5-153, 5-154, 5-155,
	C-7, E-10 through E-13, Instruction Tables 6, 7, 8, 11, 12, 13, 14, 16, 17, 18, 19, 22, 24, 25,
	and 30, Index-1, Index-2 and Index-3.
C	Publication Change Order 15865, no Product Designation change. The following pages were
(2-23-67)	revised or added: iv, v, 1-1, 1-10, 1-13, 1-14, 2-6, 3-3, 3-5, 3-6, 4-2, 4-3, 4-4, 4-5, 4-6,
	4-7, 4-8, 4-9, 4-10, 4-11, 5-26, 5-68, 5-69, 5-84, 5-87, 5-89, 5-91, 5-92, 5-94, 5-112, 5-116,
	5-117, 5-122, 5-144, 5-147, 5-149, 5-151, 5-155, Section 6, A-3, A-4, F-7, Index-1, Index-2,
	and Index-3.
D	Engineering/Publications Change Order 16076. The following pages were revised or added: v,
(6-14-67)	2-5, 2-6, 2-7, 2-8, 2-9 and 5-81.
E	Field Change Order 16164, new Product Designation 3312-A12. The following pages were revised
(6-14-67)	or added: 5-18, 5-21, 5-82, 5-82.0, D-10, Instruction Tables 8 and 22.
F	Publications Change Order 16626, no Product Designation change. The following pages were
(6-14-67)	revised or added: iii, v, 1-8, 2-1, 2-5, 2-6, 3-6, 3-7, 4-4, 4-11, 4-12, 5-3, 5-12, 5-13, 5-16,
	5-17, 5-20, 5-21, 5-22, 5-25, 5-28, 5-32, 5-33, 5-40, 5-112, 5-113, C-9, D-5, D-6, D-9, D-10,
	F-7, Instruction Tables 10, 11, 12, 13, 19, 21, 27, 28 and Index-3.
G (10-2-67)	Publication Change Order 17622. Page 5-155 revised and page 5-156 added.
H	Publication Change Order 19253, no Product Designation change. Pages iii, 1-13, 1-14, 1-15, 2-6,
(4-4-68)	4-6, 5-19, 5-127, 5-147, 5-155, C-11, C-12, C-13, Instruction Tables 2, 14, 16 and 26 revised.
J	Manual revised; includes Engineering Change Order 21959, publication change only. Pages iii,
(3-26-69)	4-12, 5-14, 5-52, 5-56, 5-68, 5-69, 5-92, 5-98, 5-100, 5-123, 5-125, 5-127 and 5-138 revised.
K	Manual revised; includes Engineering Change Order 24827, publication change only. Pages iii,
(3-2-70)	iv, v, vi, 1-3, 1-4, 1-13, 1-14, 1-15, 1-16, 4-6, 4-12, 5-12 through 5-18, 5-21, 5-60, 5-68
	through 5-168, Instruction Tables 1-22, Index 1, 2, 3, Comment Sheet, Quick Reference Instruction
	Index revised. Appendix D, E, F, instruction Tables 23-33 removed. Section 8 added.

now in

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. 60157000

Printed in United States of America

CONTENTS

FIGURES

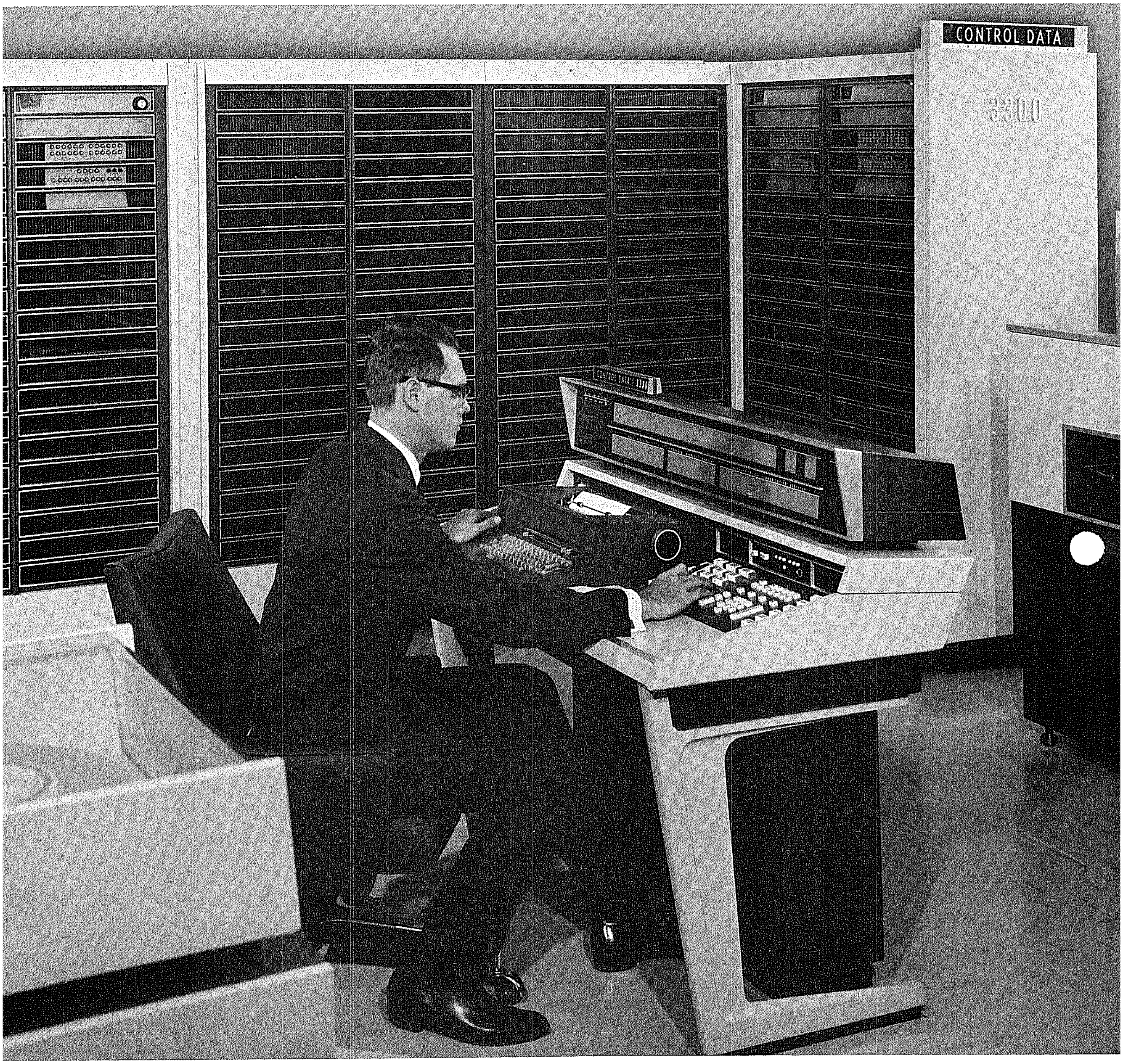
TABLES

FOREWORD

This manual provides information for the machine language use of the 3300 computer system. Its intention is to describe the capabilities and programming restraints of the hardware.

COMPASS mnemonics are used to abbreviate titles of instructions; however, no software systems are used in describing instructions. Brief descriptions of these software systems are included in Section 6. Detailed descriptions for those systems in operation are available in the appropriate software reference manuals.

Programming information for most available peripheral equipments is contained in the 3000 Series Peripheral Equipment Reference Manual, Pub. No. 60108800.



CONTROL DATA 3300 COMPUTER SYSTEM

1. GENERAL SYSTEMS DESCRIPTION

INTRODUCTION

The CONTROL DATA* 3300 is an advanced design general-purpose computing system providing high performance time-sharing with multiprogramming features to satisfy present and future needs of business and scientific users. Advanced design techniques are used throughout the 3300 to provide expedient solutions for scientific, real-time, and business data processing problems.

Time-sharing and multiprogramming features of the 3300 enable a user to enter many programs and receive processed results without the delays incurred in single-job batch processing systems. This feature not only reduces turnaround time but also provides a considerable saving in computer usage and personnel time. Multiprocessing of programs further enhances system performance when additional central processors are integrated into a total system.

Software systems for the 3300 take full advantage of the time-sharing and multiprogramming capabilities of the hardware and include the MASTER, Real-Time SCOPE, and MSOS operating systems, and the Mass Storage Input/Output (MSIO) system. A synopsis of each of these systems and other software is included in Section 6 of this manual.

All existing programs written for CONTROL DATA 3100 and 3200 systems can be processed by a 3300. I/O characteristics for the 3300 are identical to the 3100, 3200, 3400, 3500, 3600, and 3800 line of Control Data computers - a fact which facilitates incorporating the 3300 into a SATELLITE* configuration.

Included in the expanded repertoire of 3300 instructions is a complete list of business data processing instructions. These extend the flexibility of the 3300 by performing field searches, moves, compares, tests, conversions, arithmetic operations, and complete COBOL editing while utilizing the time-sharing feature of the 3300.

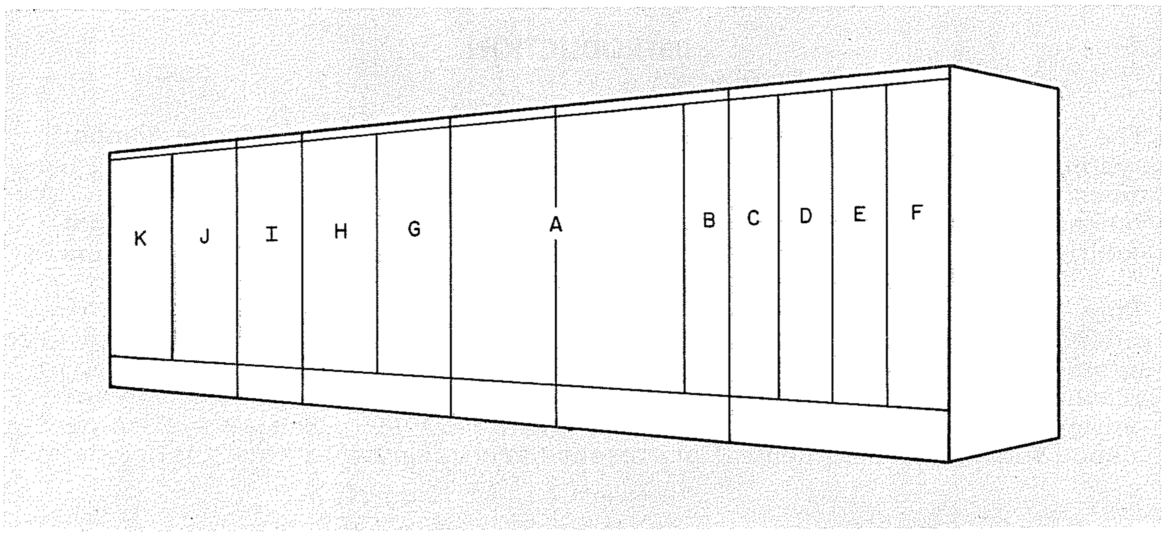
A wide selection of proven peripheral equipment is available for use in a 3300 system including many new and advanced equipments.

*Registered trademark of Control Data Corporation

This manual describes the various features of the 3300 and provides programming and operation information. Reference and supplementary information may be found in the Appendixes.

COMPUTER MODULARITY

A 3300 computer consists of various logic cabinet modules designed to perform specific operations. If additional storage, input/output channels, or arithmetic capabilities are desired for an existing installation, an appropriate module is integrated into the system. Figure 1-1 illustrates and describes the modules of a typical 3300 computer.



Ⓐ Central Processing Unit (CPU)

Ⓑ 2-3306 Input/Output (I/O) channels or 1-3307 channel and 1-3306 channel (Channels 0 and 1)

Ⓒ 2-3306 I/O channels or 1-3307 channel and 1-3306 channel

Ⓓ 2-3306 I/O channels or 1-3307 channel and 1-3306 channel

Ⓔ 2-3306 I/O channels or 1-3307 channel and 1-3306 channel

Ⓕ Power controls for I/O channels Ⓒ Ⓓ Ⓔ

Ⓖ 3310 Floating Point module

Ⓗ 3311 Multiprogramming module

Ⓘ Power Control Panel for Ⓐ Ⓑ

Ⓙ 3309-8K Storage Module

Ⓚ 3309-8K Storage Module

Figure 1-1. 3300 Modularity Example

NOTES

1. A minimum 3300 configuration consists of items Ⓐ , Ⓑ , Ⓘ , and Ⓙ .
2. A 3302 16K storage module may be substituted for items Ⓙ and Ⓚ .

(Cont'd on next page)

3. Additional storage modules are added to the left of item Ⓚ.

4. The 3312 BDP (not shown) is a "stand-alone" cable connected unit. Additional storage modules may also be stand-alone units to conform to installation space. Separate BDP cabinets similar to the 3312 are included as a part of 3304-2 and 3304-3 Business Data Processors.

5. 3307 I/O channels are always designated an even channel number, i.e., 0, 2, 4, or 6.

Central Processing Unit

The 3304 Central Processing Unit (CPU) performs the following functions:

- Controls and synchronizes most internal operations of the computer.
- Processes all 24-bit precision fixed point arithmetic.
- Processes 48-bit precision addition and subtraction.
- Executes Boolean instructions.
- Character and word loading and storing.
- Executes decision instructions.
- Controls standard search and move operations, external equipment and typewriter I/O, real-time clock referencing, and register file operations.
- Recognizes and processes all interrupts.

If the Business Data Processor (BDP) is present in a system, the CPU relinquishes control to it until the business oriented instructions(s) have been processed.

Business Data Processors

Two expanded central processors, the 3304-2 and 3304-3 Business Data Processors, are available. These processors provide a comprehensive set of variable field-length business data processing instructions in addition to all of the basic computing functions described for the 3304 CPU. The 3304-2 and 3304-3 provide somewhat different sets of business data processing instruction sets. Otherwise, the two processors are identical. The basic instruction set is the same as the 3304 CPU. Physically the 3304-2 and 3304-3 consist of two units: a basic central processor and a business data processing unit.

Optional Business Data Processing Unit

The 3312 Business Data Processing unit is an add-on device that can be connected by cables to the basic 3304 CPU. The composite 3304/3312 processor provides the same instruction set as the 3304-2 Business Data Processor.

Storage Modules

The magnetic core storage (MCS) available for 3300 systems ranges from a minimum of 8,192 (32,768 characters) to 262,144 (1,048,576 characters) words. An MCS system is expanded in 16,384 word increments after two initial 8,192 word storage modules are installed in the system. Up to 131,072 words of MCS may be included in a system without the multiprogramming option present.

The following optional storage modules are available:

Model 3309 - 8,192 word (32,768 characters) MCS memory module

Model 3302 - 16,384 word (65,536 characters) MCS memory module

The word "storage" is used synonymously with "memory" in this text and both refer to MCS unless otherwise stated. Additional information pertaining to the 3300 storage system may be found in Section 2.

Input/Output Modules

Two types of Input/Output (I/O) modules are available for use in 3300 systems. These are the 3306 and 3307 Communication Channels.

The 3306 is a bidirectional 12-bit parallel data channel and conforms to the standare I/O specifications for all CONTROL DATA 3000 Computers. A maximum of eight 3306 channels may be incorporated in a single system with up to eight peripheral controllers connected to each channel. Space is provided for mounting two 3306 channels per module. Figure 1-1 shows the placement of the channels in a maximum I/O channel configuration.

The 3307 is a bidirectional 24-bit parallel data channel and also conforms to the Control Data 3000 I/O specification. In each 3307 channel 12-to 24-bit assembly/ disassembly is included. A maximum of four 3307 channels in addition to four 3306 channels may be present in a single system.

Additional information pertaining to the 3306 and 3307 I/O channels may be found in Section 3.

Floating Point Module

The optional 3310 Floating Point Module permits a user to directly execute floating point addition, subtraction, multiplication, and division instructions utilizing 48-bit precision floating point operands. This option also permits direct execution of 48-bit precision multiplication and division instructions.

Multiprogramming Module

The optional 3311 Multiprogramming Module provides capability to relocate program instructions, data, and I/O in MCS. This option implements the 3300 memory page system and provides inherent memory protection as well as relocation and MCS extension to 262,144 words. If the 3311 is not present in a system, the maximum number of words is 131,072. Refer to Section 8 for additional information.

Operator's Console

The operators desk console includes:

- Octal register displays
- Built-in on-line typewriter
- Built-in entry keyboard and control switches
- Complete status monitoring system
- Operator's chair

A complete description of the console, examples of manual operations, and a picture of the console can be found in Section 7.

Power Control Panel

A power control panel is provided to control secondary logic power to the CPU, floating point module, and I/O channels 0 and 1. Other modules have their own power control panels. Primary power for the entire computer system is controlled by a group of switch boxes mounted on a nearby wall.

INTERNAL ORGANIZATION

Central Processing Unit

Computer Word Format

The standard 3300 computer word consists of 24 binary digits. Each word is divided into four 6-bit characters. In storage, an odd parity bit is generated and checked for each of the four characters, lengthening the storage word to 28 bits. Figure 1-2 illustrates the bit assignments of a computer word in storage.

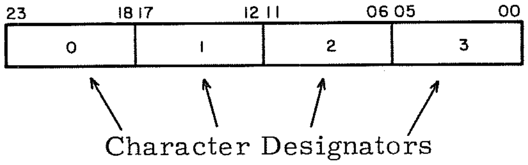


Figure 1-2. Computer Word Character Positions and Bit Assignments

Register Descriptions

A Register (Arithmetic): The A register (accumulator) is the principal arithmetic register. Some of the more important functions of this register are:

- Most arithmetic and logical operations use the A register in formulating a result. The A register is the only register with provisions for adding its contents to the contents of a storage location or another register.
- The A register may be shifted to the right or left separately or in conjunction with the Q register. Right shifting is end-off; the lowest bits are discarded and the sign is extended. Left shifting is end-around; the highest order bit appears in the lowest order stage after each shift; all other bits move one place to the left.
- The A register holds the word which conditions jump and search instructions.

Q Register (Arithmetic): The Q register is an auxiliary accumulator register and is generally used in conjunction with the A register.

The principal functions of Q are:

- Providing temporary storage for the contents of A while A is used for another arithmetic operation.
- Forming a double-length register, AQ.
- Shifting to the right or left, separately or in conjunction with A.
- Serving as a mask register for 06, 07, and 27 instructions.

E Register (Arithmetic): The optional arithmetic register E is present in a system whenever the 3311 Floating Point option is present in a system. During floating point/48-bit precision operations, the E register is divided into two parts, E_U* and E_L*, each composed of 24 bits. It is used as follows:

- 48-bit precision multiplication; holds the lower 48 bits of a 96-bit product.
- 48-bit precision division; initially holds the lower 48 bits of the dividend; upon completion, holds the remainder.
- Floating point multiplication; holds the residue of the coefficient of the 48-bit product.
- Floating point division; holds the remainder.

P Register (Main Control): The P register is the Program Address Counter. It provides program continuity by generating in sequence the storage addresses which contain the individual instructions. During a Normal Exit the count in P is incremented by 1 at the completion of each instruction to specify the address of the next instruction. These addresses are sent via the S (address) Bus to the specified storage module where the instruction is read. A Skip Exit advances the count in P by 2, bypassing the next sequential instruction and executing the following one. For a Jump Exit, the execution address portion of the jump instruction is entered into P and used to specify the starting address of a new sequence of instructions.

B^b Registers (Main Control): The three index registers, B^1, B^2, and B^3, are used in a variety of ways, depending on the instruction. In a majority of the instructions they hold quantities to be added to the execution address, $M=m+B^b$. The index registers may be incremented or decremented.

C Register (Main Control): Quantities to be entered into the A, Q, B, or P registers or into storage from the entry keyboard are temporarily held in the Communication (C) register until the TRANSFER switch is pushed. If an error is made while entering data into the Communication register, the KEYBOARD CLEAR switch may be used to clear this register.

The C register holds words read from storage during a Sweep or Read Storage operation. The contents of C are displayed on the console whenever the keyboard is active.

*E_U signifies E_{Upper}; E_L signifies E_{Lower} .

F Register (Main Control): The program control register F holds an instruction during the time it is being executed. During execution, the program may modify the instruction in one of three ways:

- Indexing (Address Modification) - A quantity in one of the index registers (B^b) is added to the lower 15 bits of F for word-addressed instructions, or to the lower 17 bits of F for character-addressed instructions. The signs of B^b and F are extended for the addition process.
- Indirect Addressing - The lower 18 bits of F are replaced by new 'a', 'b', and 'm' designators from the original address M (modified if necessary, $M = m + B^b$).
- Indirect Addressing (load and store index instructions) - Bits 00 - 14 and 17 of F are replaced by new 'a' and 'm' designators from the original address M (no modification possible).

After executing an instruction a Normal Exit, Skip Exit, or Jump Exit is performed. F is displayed on the console whenever the keyboard is inactive and the computer is not in the GO mode.

Instruction State Register (Main Control): Instruction State register is a 3-bit register that is referenced under certain conditions when the computer is operating in Executive mode. The (ISR)* are appended to the (P) in the process of referencing different program address groups. Refer to Appendix E for the different conditions when this register may be used.

Operand State Register (Main Control): Operand State register is also a 3-bit register that is referenced under certain conditions when the computer is operating in Executive mode. Appendix E describes the conditions when the OSR is referenced with regard to the operational state of the CPU.

Channel Index Register (Main Control): The Channel Index register (CIR) is a 3-bit register whose contents are logically OR'ed (inclusive OR function) with the channel designator 'ch' for the following instructions:

●	73-76	I/O instructions
●	77.0	Connect
●	77.1	Select Function
●	77.2	Sense External Status
●	77.2	Copy External Status
●	77.3	Sense Internal Status
●	77.3	Copy Internal Status
●	77.4	Sense Interrupt

This permits instructions to be written for channel 0 and allows the monitor program to assign the proper channel by altering the (CIR). The (CIR) can be transferred by instruction to the lower 3 bits of the A register and vice versa. A momentary switch is provided on the console for displaying (CIR) in the lowest digit position of the Index register display area.

*The parentheses, as they are used in this case, are an accepted method for expressing the words "the content(s) of" (in this case "the contents of" the ISR register).

Condition Register (Main Control) : Bits in the Condition register (CR) are used as flags to initiate computer action and to record current operating conditions during Executive mode. With the exception of bit 04, the register is not used during non-Executive mode operations.

All register bits can be set or cleared with the ACR (77. 634) instruction; selected bits are set or cleared by individual instructions and conditions. Refer to Section 4 for special considerations involving the register during interrupt processing. The register bit assignments are listed below:

Bit 00	-	Boundary Jump
Bit 01	-	Destructive Load A
Bit 02	-	Operands Relocated Using OSR
Bit 03	-	Program State Jump
Bit 04	-	Interrupt System Enabled
Bit 05	-	Program State

Data Bus Register (DBR - Main Control): A 24-bit Data Bus register is used to temporarily hold the data received from storage, Communication register, and other logic areas. It is a nondisplayed and nonaddressable register.

During character-addressed or I/O operations, data entering the DBR may be shifted one, two, or three character positions during the transfer to reach the correct character position within the DBR.

Interrupt Mask Register (Main Control): The 12-bit Interrupt Mask register allows a user to honor or ignore a group of various interrupts by setting the register bits to "1's" or "0's". Each register bit corresponds to a particular interrupt condition. The mask bits may be set or cleared by executing the SSIM and SCIM instructions, respectively. The specific mask register bit assignments are described in Section 4.

S Register (Storage): The S register holds the address of the storage word currently being referenced.

Z Register (Storage): The Z register is the storage restoration and Modification register. Data stored or being transferred to or from the address specified by the S register must pass through Z. The entire storage word including the four parity bits is represented by the Z register and is displayed on the Storage Module control panel.

Bus Systems

The Data Bus provides a common path over which data must flow to the storage, arithmetic, console typewriter, and I/O sections of the computer. These sections are connected in parallel to the Data Bus. During the execution of each instruction, Main Control determines which data transfer path is activated.

An odd parity bit is generated for the lower byte of each word as it leaves the DBR during I/O operations. In the case of a 3307 I/O Channel, parity for the upper byte of data is generated in the channel itself rather than in the Data Bus.

The S or Address Bus is a data link between Main Control and storage for transmitting storage addresses. Inputs to the S Bus are from the P register, F register, Block Control, and the Breakpoint circuits.

Executive Mode

The CPU can operate in either the non-Executive mode or Executive mode. In non-Executive mode the 3300 operates identically to the 3200.

Depressing the EXECUTIVE MODE switch on the operator's console causes the 3300 to function in the Monitor State of Executive mode. All 3300 instructions may be executed in the Monitor State provided the necessary hardware is present in the system.

After executing a Set Boundary Jump (SBJP) instruction, the next jump instruction causes the 3300 to advance to the Program State of Executive mode. In Program State, the CPU performs at its highest efficiency by restricting itself to actual computations by not executing I/O or Block Control instructions. If a Halt (00.0) instruction, any of the 71-77 instructions (except SBCD 77.72 and SFPF 77.71), or an inter-register transfer affecting registers 00 through 37 of the register file is attempted while in Program State, the 3300 reverts to the Monitor State of Executive mode. Additional information can be found in Appendix E.

Block Control

Block Control is an auxiliary control section within a 3300 processor. In conjunction with the register file and program control, it directs the following operations:

- External equipment I/O
- Search/Move
- Real-Time clock
- Console typewriter I/O
- High-speed temporary storage

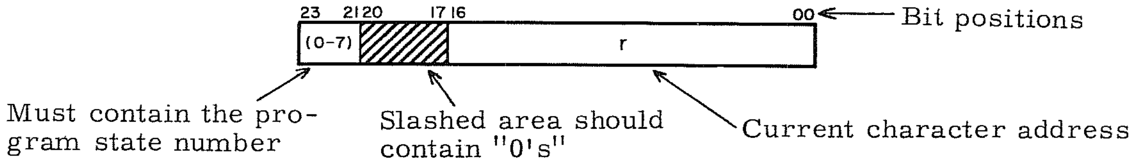
Register File: The register file is a 64-word (24 bits per word) rapid access memory with a cycle time of 0.5 usec. Although the programmer has access to all registers in the file with the interregister transfer (53) instruction, certain registers are reserved for specific purposes (see Table 1-1). All reserved registers may be used for temporary storage if their use will not disrupt other operations that are in progress.

The contents of any register in the file may be viewed by selecting the register number with the Breakpoint switch and pressing the READ STO switch on the keyboard. The contents may be altered by setting the Breakpoint switch, pressing the WRITE STO switch, and entering a new word from the keyboard.

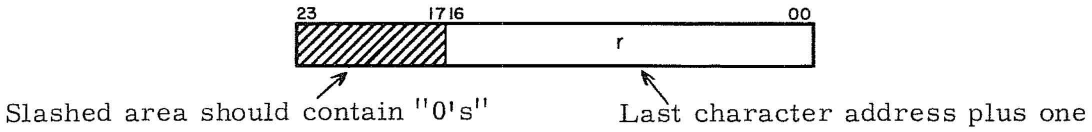
TABLE 1-1. REGISTER FILE ASSIGNMENTS

Register Numbers	Register Functions
00-07	Modified I/O instruction word containing the current character address (channel 0-7 control)
10-17	Modified I/O instruction word containing the last address ± 1, depending on the instruction (channel 0-7 control)
20	Search instruction word containing the current character address (search control)
21	Move instruction word containing the source character address (move control)
22	Real-time clock, current time
23	Current character address (typewriter control)*
24-27	Temporary storage
30	Instruction word containing the last character address + 1 (search control)
31	Instruction word containing the destination character address (move control)
32	Real-time clock, interrupt mask
33	Last character address +1 (typewriter control)**
34-77	Temporary storage

*The contents of register 23 should have the following format:


23
21 20
17 16
00
Bit positions
(0-7)
r
Must contain the program state number
Slashed area should contain "0's"
Current character address

**The contents of register 33 should have the following format:


23
17 16
00
r
Slashed area should contain "0's"
Last character address plus one

Block Control Priority: Access to Block Control circuits is shared between the computer's main program control and block control buffered functions. Functions within Block Control are divided into three groups (Refer to Table 1-2.) The five scanners that provide the priority access network for the system are the Program/Buffer scanner, the Group scanner, and the three inner group scanners. Figure 1-3 illustrates the scanning pattern of the priority network.

The Program/Buffer scanner alternately checks for Block Control requests from Program Control and any Group requests. Group requests have priority over Program requests and as long as Group requests are present, they will be serviced before a Program request. When all Group requests have been serviced, a Program request can be recognized.

Another free running scanner checks the three groups for an active Block Control request. After a request from one group has been processed, the scanner moves to the next group, rotating through the groups in a 3, 2, 1, 3 order.

Each group has a four-position scanner. These scanners search from top to bottom of their respective groups looking for active Block Control requests. After they find a request and it has been processed, the scanners return to the top of their group before resuming their search.

TABLE 1-2. BUFFER GROUPS

Group 1	Group 2	Group 3
Channel 0 Control	Channel 4 Control	Real-time clock control
1	5	Console typewriter control
2	6	Register File Display
3	7	Search/Move Control

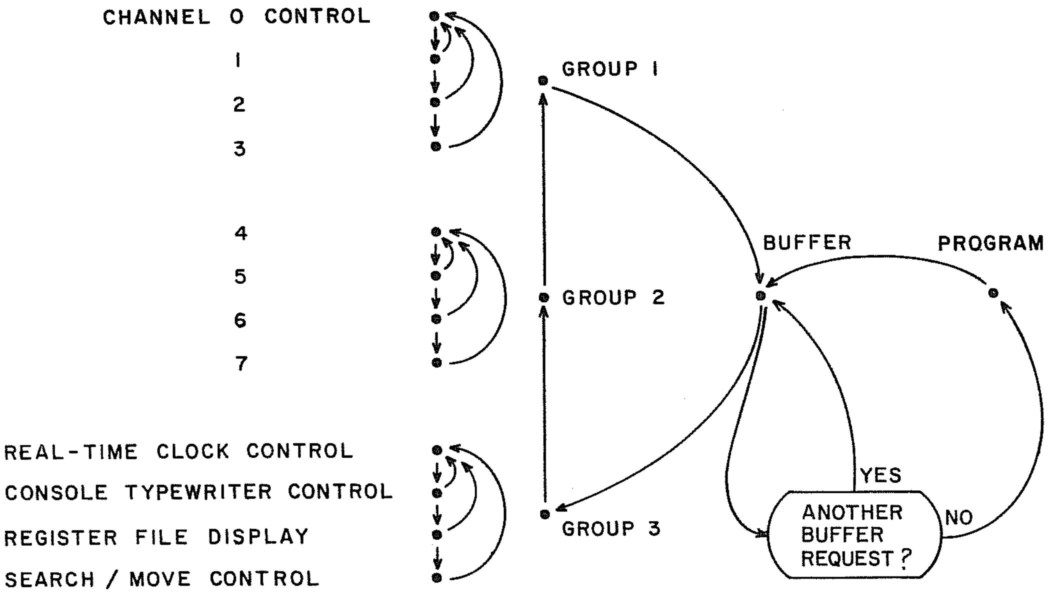


Figure 1-3. Block Control Scanning Pattern

Real-Time Clock

The real-time clock is a 24-bit counter that is incremented each millisecond to a maximum period of 16, 777, 216* milliseconds. After reaching its maximum count the clock returns to zero and the cycle is repeated continuously. The clock, which is controlled by a 1 kilocycle signal, starts as soon as power is applied to the computer. The current time is stored in register 22 of the Register File. It is removed from storage, updated, and compared with the contents of register 32 once each millisecond. When the clock time equals the time specified by the clock mask, an interrupt is set. When necessary, the real-time clock may be reset to any 24-bit quantity including zero by loading A and then transferring (A) into register 22. Performing a Master Clear does not affect the clock count.

For a special case involving the real-time clock, refer to the Priority Pause (PRP) instruction in Section 5.

Parity

Parity bits are generated and checked in 3300 systems for the following two conditions:

1. Whenever a data word is read from or written into storage.
2. When a data word is transferred via an I/O channel.

Storage Parity: A parity bit is generated and checked for each 6-bit character of a storage word. Refer to Figure 1-4.

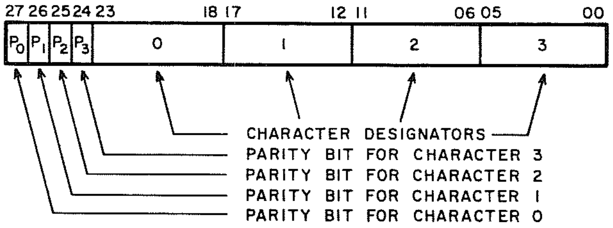

27 26 25 24 23
18 17
12 11
06 05
00
P0
P1
P2
P3
0
1
2
3
CHARACTER DESIGNATORS
PARITY BIT FOR CHARACTER 3
PARITY BIT FOR CHARACTER 2
PARITY BIT FOR CHARACTER 1
PARITY BIT FOR CHARACTER 0

Figure 1-4. Parity Bit Assignments

During each Write cycle, a parity bit is stored along with each character. When part or all of a word is read from storage, parity is checked for a loss or gain of bits. Failure to produce the correct parity during read operations causes the PARITY FAULT indicators on the storage module control panel and internal status lights to glow. As soon as a parity error is recognized and the PARITY STOP switch on console is active, program execution is halted. Master clearing the computer clears the fault condition.

If the PARITY INTERRUPT switch is active and an interrupt is recognized, the computer enters a special interrupt routine (see Section 4).

*16, 777, 216 milliseconds equals approximately 4 hours and 40 minutes.

The total number of "1's" in a character, plus the parity bit, is always an odd number in the odd parity system used in the 3300.

I/O Parity: The I/O communication channels provide parity lines in addition to the other signals that interface with external equipment. Parity is checked in the I/O channels to detect parity errors during data transmission to the external equipment and errors when data is received from external equipment. I/O parity errors can be detected by a sensing instruction; however, the parity error indicator is not activated. A complete description of I/O parity generation and checking may be found in Section 3 of this manual.

Business Data Processing Units

The business data processing instructions provided by the 3304-2 and 3304-3 Business Data Processors and the optional 3312 BDP add-on unit perform operations on variable length fields of 6-bit characters. Some typical operations are:

1. Move a block of 6-bit characters from one region in memory to another.
2. Add two fields of BCD digits.
3. Search a field of BCD characters for a specific value.
4. Compare to fields of characters for equality.
5. Convert a field of BCD digits to a binary number.

The major characteristics of the BDP instructions are summarized below.

MOVES AND EDITS

The following capabilities are features of this instruction category:

- Ability to transfer variable length data fields from one area of storage to another.
- Both fields may specify any 6-bit character location in storage as the beginning address
- Both fields may be independently indexed
- Up to 4095 characters may be moved
- Operations may be terminated by specifying lengths of fields or delimiting characters.
- Data moved from a source field to a receiving field may be manipulated and/or modified as follows:
  - ► Single character or block of characters transferred without modification
  - ► Move with blanks inserted in any unfilled character positions in the receiving field
  - ► Move with zeros inserted in any unfilled character positions in the receiving field
  - ► Move with leading zeros replaced with blanks and zone (sign) bits stripped during the transfer

    - ► Move with edit functions performed: insertion of commas, decimal point with suppression of leading zeros, or complete formatted edit with insertion of character set as defined in DOD COBOL-61 Extended specification

Instructions in this group are particularly useful in data processing applications involving character manipulation, formatting for printing of integer quantities, point alignment problems, etc. Editing functions are accomplished by hardware rather than a complex subroutine, resulting in fast processing times.

SEARCHES

The following capabilities are features of this category of instructions:

- Any 6-bit character location in storage may be specified as the location of the first character of a field to be searched.
- Up to 4095 characters may be examined
- Indexing may be accomplished on the search field
- Search key (character) specified by programmer and contained in instruction word
- Search may be terminated by:
    - ► Locating object character
    - ► Examining a specified number of characters without locating object character
    - ► Encountering delimiter character specified in a Search instruction.
- At conclusion of search operation, an index register holds number of characters searched to aid in determining location of character meeting search condition.
- Program control at search termination branches to either of two points, depending on result of search
- Searches may be of the following types:
    - ► Search successive character locations (either left to right or right to left) in a field for an object character equal to the search key
    - ► Search successive character locations (either left to right or right to left) in a field for an object character unequal to the search key
    - ► Search successive character locations (from left to right) in a field for an object character equal to the search key and jump; jump is to normal termination point plus the number of characters searched

CODE CONVERSION FEATURES

The following conversion operations can be performed on fields of 6-bit characters.

- Convert BCD to binary
- Convert binary to BCD
- Translate to ASCII
- Translate from ASCII

  Available in 3312 optional BDP unit and 3304-2 BDP only

- Pack (convert numeric 6-bit BCD characters into 4-bit characters)
- Unpack (convert numeric 4-bit BCD characters into 6-bit characters)

ARITHMETIC FUNCTIONS

The following capabilities are features of this category of instructions:

- Add or subtract two fields of 6-bit numeric BCD characters
- Both fields may specify any 6-bit character location in storage as the beginning address
- Both fields may be independently indexed
- Algebraic sign control
- Arithmetic overflow fault indicator provided
- Compare two fields of numeric characters to determine which field contains the largest number.
- Compare two fields of alpha-numeric characters to determine which field ranks highest in a collating sequence.
- Test instructions examine field for: greater than zero, zero, or less than zero. The result of the test sets a BCD condition register to +, 0, or -
- Jump instructions in the CPU may be used to examine arithmetic result flags in the BDP

Detailed information on the BDP instructions is included near the end of Section 5.

PERIPHERAL EQUIPMENT

A wide variety of peripheral equipment is available for use with the 3300 computer. All peripheral equipment available for 3100, 3200, 3300, 3400, 3600, and 3800 systems may be attached to a 3306 communication channel. For programming instructions, as well as a list of function codes and status response codes, refer to the Control Data 3000 Series Computer Systems Peripheral Equipment Reference Manual (Pub. No. 60108800).

2. STORAGE SYSTEM

GENERAL INFORMATION

The 3300 Magnetic Core Storage (MCS) system receives and transmits storage words to the CPU (and BDP if it is in the system). Each storage module provides parity checking and visual address and data displays. Each storage (or memory) reference requires 1.25 usec within the storage module referenced.

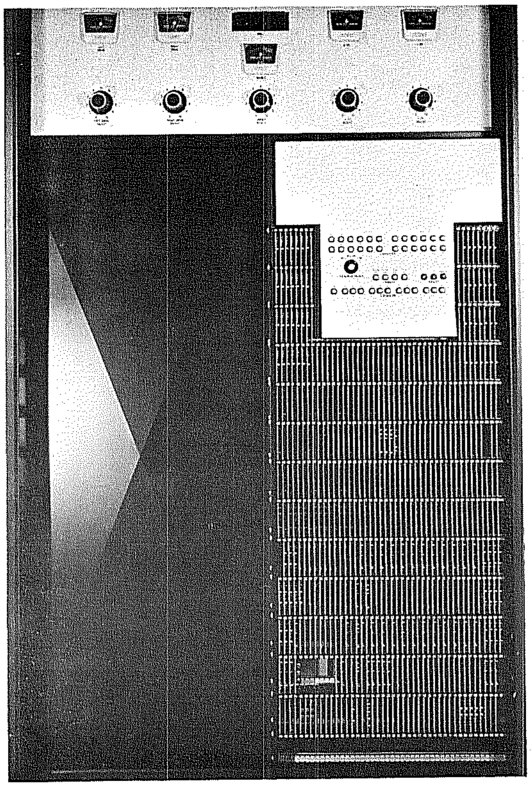
STORAGE MODULES

A minimum storage configuration consists of one 3309 8,192 word Storage Module. An additional 3309 Storage module brings the total storage capacity to 16,384 words. Further storage expansion is provided by adding model 3302 16,384 word Storage modules. If the 3300 is equipped with a 3311 Multiprogramming module, 3302 Storage Modules may be added to bring the total MCS capacity to 262,144 words. If the 3311 is not in the system, the maximum MCS is 131,072 words. The 3309 and 3302 Storage modules are shown in Figure 2-1 along with an enlarged view of their control panels.

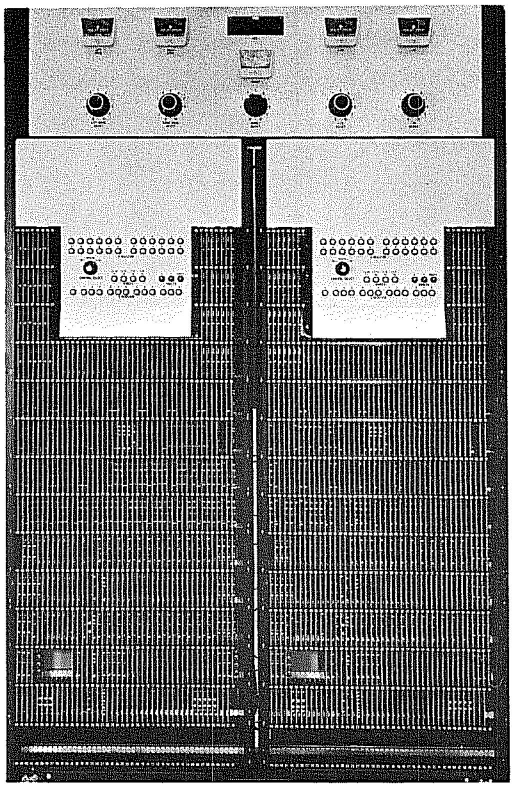
STORAGE REGISTERS

S Register - The S register receives and holds the storage address, enabling address translation for the word currently being referenced. The register consists of 13 bits and 14 bits, respectively, in the 3309 and 3302 storage modules.

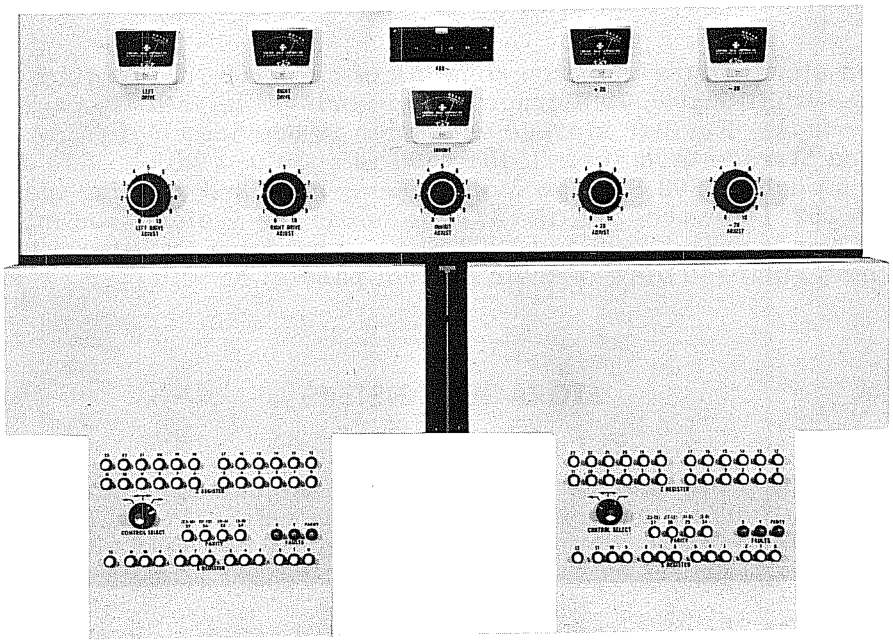
Z Register - The 28-bit Z register is the storage restoration and modification register. All data that is transferred to or from the storage module passes through Z.



3309 8K Storage Module

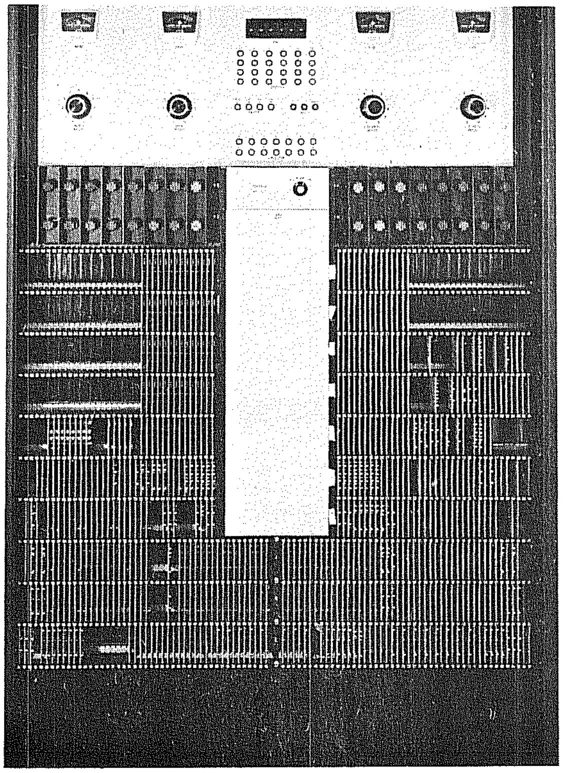


Dual 3309 16K Storage Module



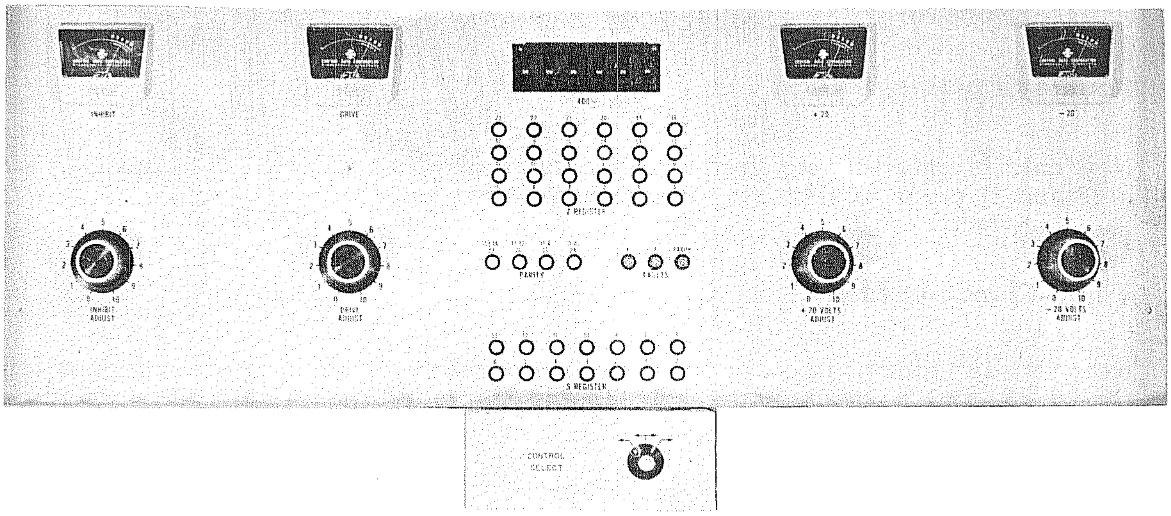
Dual 3309 Storage Module Control Panel

Figure 2-1. 3300 Storage Modules



1734

3302 Storage Module



1734

3302 Storage Module Control Panel

Figure 2-1. 3300 Storage Modules (Cont'd)

STORAGE WORD

A storage word is 28 bits in length of which four bits are used for parity checking the remaining 24 bits. The 24 bits, labeled 00 through 23 from right to left, may be a single 24-bit instruction, part of a two or three word instruction, a zero to 24-bit operand, or part of a larger operand. The storage corresponds to the standard computer word and its format as described in Section 1.

CHARACTER MODES

During a read storage operation, all bits of a word referenced by (S) are read out of core storage into the Z register (in parallel) and are restored without modification at the same address. For a write storage operation, five basic modes exist for modifying (Z) prior to restoration. Any characters not modified are restored unchanged. Write Character Designators from the computer or other access devices specify the type of write operation to be performed.

Single-Character Mode

New data is entered into any one of the four characters prior to restoring the word in core.

Double-Character Mode

New data is entered into any two adjacent characters (character 0 and 1, 1 and 2, or 2 and 3) prior to restoring the word in core.

Triple-Character Mode

New data is entered into either of the two possible three-character groups (characters 0, 1, and 2, or characters 1, 2, and 3) prior to restoring the word in core.

Full-Word Mode

New data is entered into characters 0-3 prior to restoring the word in core.

Address Mode

New data is entered into the lower 15 bits (word address) or the lower 17 bits (character address) prior to restoring the word in core.

ADDRESSING

The S bus, as described under Bus Systems in Section 1, carries the address of the memory location being referenced to the proper storage module. During Executive mode, the (ISR)* or (OSR)* are appended to a 15-bit basic address (as displayed in the P register) to form an 18-bit address. The upper 3 bits of address are forced to zero during non-Executive mode to limit storage addressing to 32,768 words.

If a storage reference is made for an address contained in a non-existent memory module, a high priority interrupt may be entered. Refer to the Storage Parity Error-No Response Interrupt in Section 4 for details.

MULTIPROGRAMMING AND RELOCATION

The 3311 Multiprogramming Module permits the instructions of many programs to be sequentially executed and relocated in MCS under the control of a monitor program. The available MCS in a 3300 system is grouped into "memory pages" consisting of 2,048 absolute memory locations. By using a Page Index File and advanced logic circuits, the 3311 makes optimum use of memory pages as they become available during program execution.

Appendix E includes detailed information on multiprogramming and relocation concepts as applied to the 3300.

STORAGE PROTECTION

It is often desirable to protect the contents of certain storage addresses against alteration during the execution of a program. There are four categories of addresses: those that are always protected, those that are protected at the option of the programmer, those that are protected by the multiprogramming and relocation features, and those that are never protected during special sequences.

If any attempt is made to write at a protected address during non-Executive mode, the illegally addressed location remains unaltered (Write is changed to a Read), the console Illegal Write indicator lights, and program execution continues. The illegal write condition is recorded by setting bit 05 of the internal status sensing network. The condition is cleared by a Master Clear, an Internal Clear, or by sensing.

* Only the numbers 0, 1, 2 and 3 in the ISR or OSR can be used in the Multiprogramming option is not in the system.

During Executive mode, a protected address remains unaltered (Write is changed to a Read) during all write operations, except those occuring in Monitor State and during Block Control operations. The condition is not recorded on the status line. Refer to the Illegal Write interrupt discussion in Section 4 for additional information.

Permanent Protection

The upper 32 memory locations of the existing MCS are reserved for Auto Load and Auto Dump programs when operating in the non-Executive mode. These addresses are always protected against alteration by a special storage protection circuit. The actual protected addresses depend upon the number of MCS locations in a system but always utilize the upper 32 locations in any system.

Logic circuits sense the total storage capacity of the system and check each storage address as it appears on the S (address) Bus to see if it is among the protected addresses. If it is one of those to be protected, reading, but no writing, is allowed at that address. The only time that this protection is disabled is when an operator presses the ENTER AUTO PROGRAM switch on the console to enter a new Auto Load or Auto Dump program.

When operating in Executive mode, the Auto Load and Auto Dump storage areas encompass addresses 003700_8 through 003777_8 and are protected when referenced through Page Index Zero. Refer to Section 3 for additi onal information on the Auto Load and Auto Dump features.

Selective Protection

3304-A Central Processor

Two different selective protect schemes are available with the 3304-A; one being standard and the other available by option. In the standard protect scheme, 15 three-position toggle switches, corresponding to the basic 15-bit storage address, are set to selectively protect individual addresses or a block of addresses. The switches are located on the power control panel as shown in Section 7.

Table 2-1 describes the three switch positions and Table 2-2 lists examples of switch settings. The switches are automatically disabled during execution of the BDP instructions (64-70). In Executive mode, the switches apply to an address range of which the upper 3 address bits (ISR) or (OSR) are equal to zero.

TABLE 2-1. STORAGE PROTECTION SWITCH DESCRIPTIONS

Output	Switch Position	Description
"1"	Up	Each address protected will have a "1" in this bit position.
"N"	Center	Each address protected may have either a "1" or a "0" in this position. For example, when all switches are set to the neutral position, all storage is protected, provided that the protect feature is enabled.
"0"	Down	Each address protected will have a "0" in this bit position.

TABLE 2-2. STORAGE PROTECTION SWITCH SETTINGS

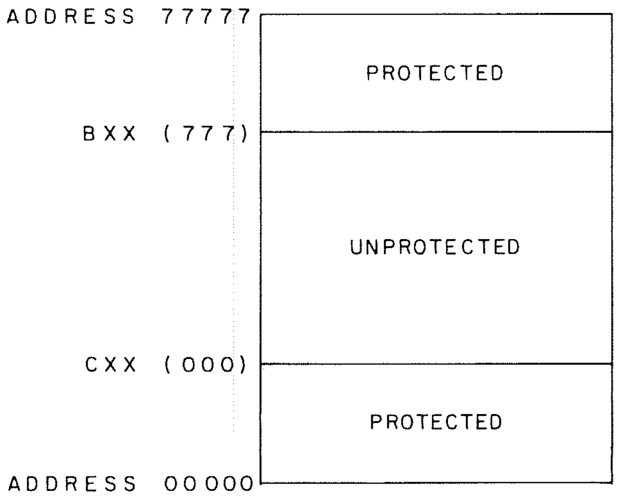
Description of Protected Addresses	Examples:	
	Settings-Storage Protection Switches	Addresses Protected (octal)
Single storage address	000 000 000 001 111	00017
Two nonsequential addresses of a group of 10_8.*	000 000 000 010 0N0 000 000 000 010 N10	00020 & 00022 00022 & 00026
Four nonsequential addresses of a group of 10_8.*	000 000 000 010 N0N 000 000 000 010 NN1	00020, 00021, 00024, & 00025 00021, 00023, 00025, & 00027
Four address block - may be the upper or lower half of a group of 10_8.*	000 000 000 100 0NN 000 000 000 100 1NN	00040-00043 00044-00047
10_8 address block	000 000 000 010 NNN	00020-00027
20_8 address block	000 000 001 00N NNN 000 000 001 11N NNN	00100-00117 00160-00177
40_8 address block - may be the upper or lower half of a group of 100_8.*	100 000 000 0NN NNN 100 000 000 1NN NNN	40000-40037 40040-40077
Numerous other groups and combinations of the above groups may also be protected.	000 000 000 NNN 110 NNN NNN NNN NNN 111 NNN NNN 001 NNN NNN	00006, 00016, 00026 ... 00076 All XXXX7 addresses All XX1XX addresses (00100-00177, 01100-01177, etc.)

*The first address of all groups of 10_8, 20_8, 40_8, 100_8 etc., must have a lower octal digit of zero. Blocks of 100_8, 200_8, 400_8, 1000_8, 2000_8, 4000_8, etc., may be protected in the same manner as blocks of 10_8, 20_8, & 40_8.

The optional protect scheme allows two independent blocks of locations within a designated 32K of storage to be protected during non-Executive or Executive mode. With this feature, protection can be given to the resident monitor program and to another program that may be operating.

The area increasing in address from address 00000 may be protected in multiples of 512_{10} locations. The area decreasing from address 77777 can similarly be protected. The number of locations protected in an area is determined by setting the six toggle switches associated with that area; each of the 77_8 possible settings represents one multiple of 512 locations. The six switches labeled 9 through 14 select the lower protected area; those labeled 0 through 5 select the upper protected area (refer to Figure 7-11). Figure 2-2 illustrates the protection scheme. Table 2-3 gives examples of switch settings and their corresponding protected areas.

Switch settings for both schemes are disabled by pressing the DISABLE STO PROTECT switch on the console.



Bxx = 6 switches to select upper area address boundary, lower 9 bits of which are always "1's"

Cxx = 6 switches to select lower area address boundary, lower 9 bits of which are always "0's"

Figure 2-2. Optional Protect Scheme

TABLE 2-3. OPTIONAL STORAGE PROTECTION EXAMPLES

Bxx Setting	Locations Protected (Upper and Lower areas)	Cxx Setting
76	01000_8 = 512	01
75	02000_8 = 1,024	02
74	03000_8 = 1,536	03
67	10000_8 = 4,096	10
57	20000_8 = 8,192	20
40	37000_8 = 15,872	37
37	40000_8 = 16,384	40
36	41000_8 = 16,896	41

3304-B Central Processor

The basic 3304-B Central Processor contains no standard storage protection. Storage protection is available by option and operates the same as the optional protection which is available on the 3304-A processor.

Program Protection

When the 3300 is operating in the Program State of Executive mode, the relocation features of the 3311 Multiprogramming module are used by the monitor program to protect certain addresses from being altered.

If the exclusion bit of a particular Page Index is a "1" and PL,† PA,† or PP† is a quantity other than zero, PA defines a memory area where only reading is permitted. If the exclusion bit is "1" and PL, PA,and PP are all equal to zero, neither reading nor writing is permitted.

The monitor program controls the relocation process and uses the paging system to provide efficient use of memory while processing various programs. Appendix E explains in detail the 3300 paging and relocation processes.

No Protection

Addresses 00002 through 00005, 00010, 00011, 00014, 00015, 00020, and 00021, which are used by the interrupt system, are never protected during the interrupt sequence.

STORAGE SHARING

Two 3300 computers may share the memory of a storage module. A switch on each storage module control panel allows the operator to give exclusive control to the right or left computer. A middle position on this switch actuates a two-position priority scanner. Storage control honors the requests in the order they are received. Neither computer has priority over the other and the computer involved in the current storage cycle relinquishes control to the requesting computer at the end of its cycle. Either computer can therefore be delayed a maximum of one storage cycle. A similar program delay may occur within either computer when an internal scanner determines whether Main Control or Block Control has access to the storage module.

Direct access to 3300 type storage modules is available for certain installations. The normal I/O channel route is bypassed and the customer's special equipment interfaces directly with the storage logic.

†Refer to Appendix E for designator descriptions.

3. INPUT/OUTPUT SYSTEM

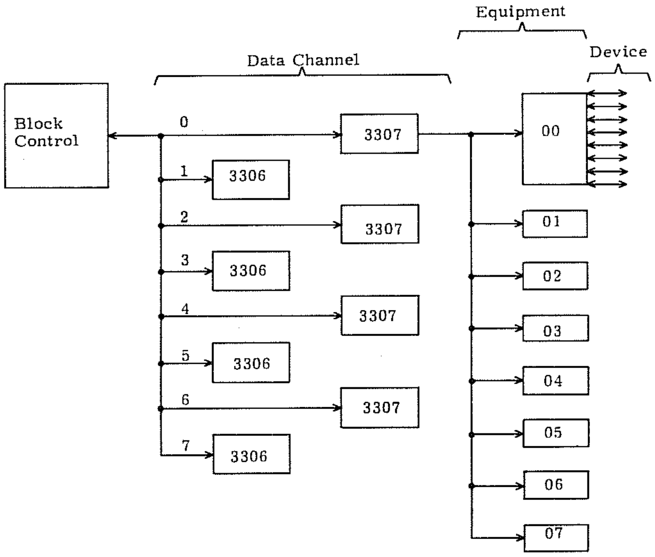
GENERAL INFORMATION

Data is transferred between the 3300 Central Processor and its associated peripheral equipment via a 3306 or a 3307 Communication Channel. The 3306 utilizes a 12-bit parallel-transfer byte and the 3307 provides a 24-bit byte. A maximum of eight 3306's or four 3307's and four 3306's may be linked to a single system. Both the 3306 and the 3307 are bidirectional and each channel may communicate with a maximum of eight peripheral controllers. A data channel can communicate with only one device at a given time, however. Each peripheral controller in turn may be attached to a number of peripheral devices. Figure 3-1 is a simplified block diagram of a 3300 Communication System.

For programming purposes, the eight possible I/O channels are designated by numbers 0 through 7. A 3307 channel will always be an even channel. The total number of channels must always be even. Depending upon the user's needs, any combination of 3306's and 3307's may be present provided all the forenamed rules are followed.

A basic 3300 system includes two 3306 Communication Channels or one 3307 channel and one 3306 channel. Figure 1-1 indicates the location of these channels in a fully expanded system.

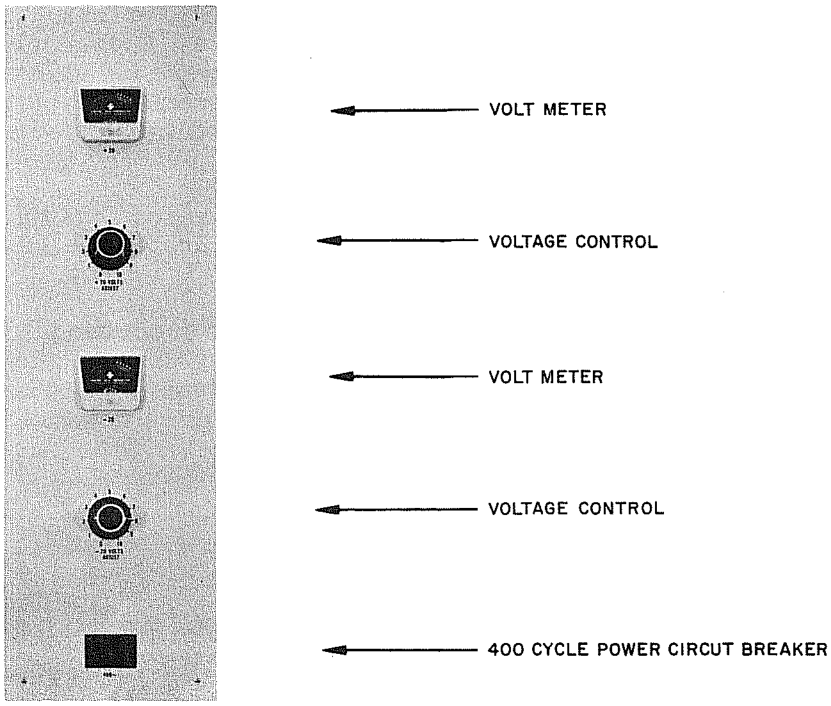
Channels 0 and 1 derive their operating power from the CPU. Power for all other channels is controlled through the I/O Channel Power Panel shown in Figure 3-2 and as F in Figure 1-1. The two voltage controls should be adjusted to produce 0% reading on the meters when the 400 cycle power circuit breaker is turned ON.



A maximum configuration of data channels. A smaller configuration may be obtained by removing the channels in pairs of odd and even. Any 3307 may be replaced by a 3306, but not vice versa.

Each channel may connect to a maximum of eight equipments. The number of devices connected to an equipment depends upon the equipment.

Figure 3-1. 3300 I/O System



1724

Figure 3-2. I/O Channel Power Panel

INTERFACE SIGNALS

Figure 3-3 shows the interface signals between a data channel and its external equipment. The twelve status lines are active only between the channel and the controller to which it has been connected by a CON (77.0) instruction. Since a Connect instruction causes all controllers on the specified channel to disconnect except the one to which it is directed, only one controller may be connected to a channel at one time. Thus to check status the program must first Connect the device.

There are eight interrupt lines, one to each controller. A controller need not be connected to return an interrupt signal to the data channel. These lines are designated as 0-7 and match the Equipment Number switch setting on each controller. For a complete description of the I/O interface signals as well as an I/O timing chart, refer to the 3000 Series Input/Output Specifications Manual, Pub. No. 60048800.

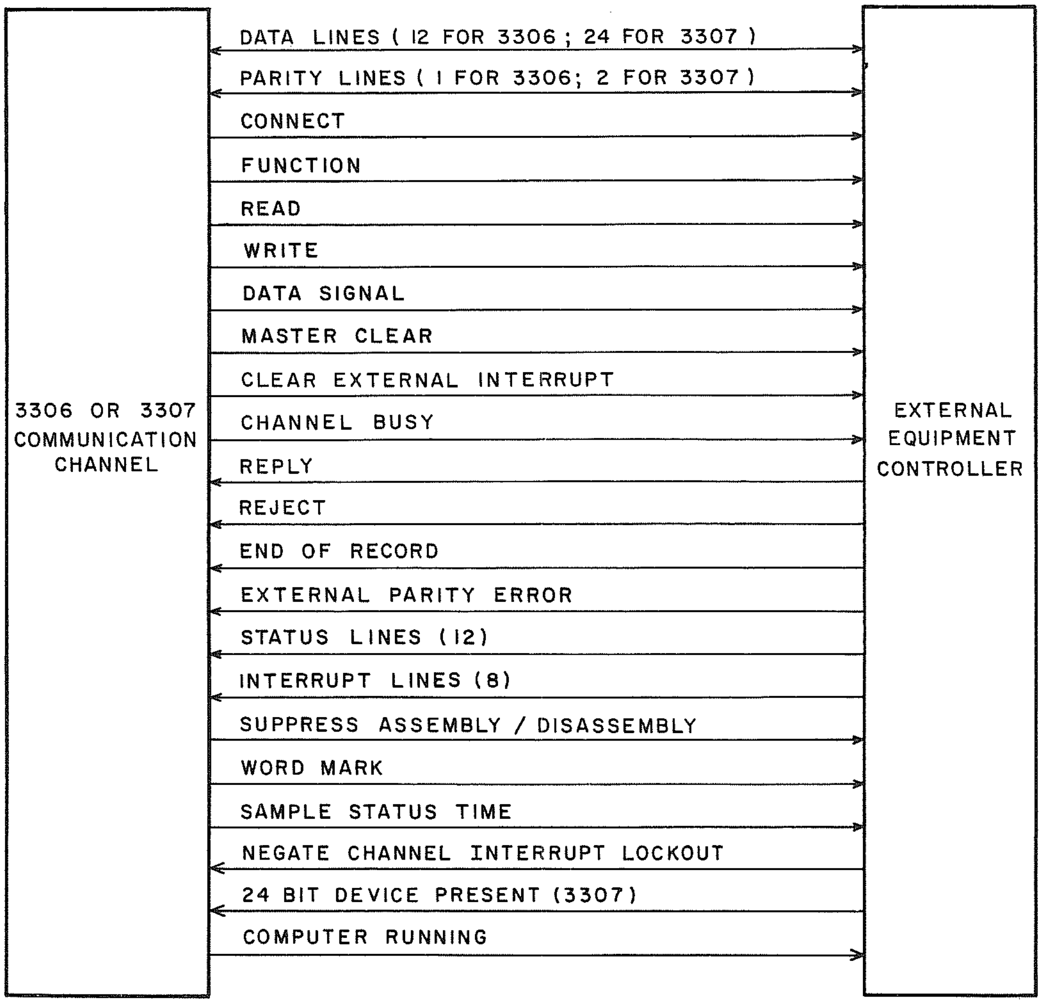


Figure 3-3. Principal Signals Between I/O Channel and External Equipment

3306 AND 3307 COMMUNICATION CHANNELS

Communication channels provide a buffer between the computation section and various peripheral controllers, thus preventing a tie-up of the computation section while awaiting a response from an external equipment. Since an I/O section contains no manual controls or indicators, all operations must be initiated by program via the computation section of the computer. Prior to actual data exchange the program must execute several instructions which connect the equipment to the channel, specify operating conditions, check status conditions, and initiate the Read orWrite operation. After the Central Processor initiates the Input or Output operation, a communication channel can exchange data between the peripheral device and core storage independent of the Central Processor.

All assembly and disassembly for the 3306 12-bit channel is done by block control, not the 3306. Two memory references are necessary to store or transmit a 24-bit word when doing a word addressed I/O instruction with 12- to 24-bit assembly. In contrast, the 3307 contains its own assembly/disassembly feature. The assembly feature allows the channel to receive two 12-bit bytes from an external equipment and assemble them into a 24-bit word before storing in memory. The disassembly feature permits the channel to accept a 24-bit word from storage and transmit it to an external equipment in 12-bit bytes.

The 3307 also facilitates a convenient interface with a 24-bit I/O device. The 24-bit transfers between memory and the 3307 reduce to one the number of memory references necessary to execute a word addressed I/O instruction. Thus the 3307 is adapted for use with high-speed 12- and 24-bit I/O devices. When doing character addressed instructions, it acts as a 3306.

I/O PARITY

Parity Checking With the 3306

The computer checks parity by one method for Connect, Function, and Write operations and by a second method for Read operations. External equipment responds differently to parity errors for a Connect than for a Function, Read, or Write. For details on external equipment responses to parity errors see 3000 Series Peripheral Equipment Reference Manual, Pub. No. 60108800.

Connect, Function, and Write

During the Connect, Function, and Write operations the Data Bus circuit of the computation section generates a parity bit and sends it to the external equipment with each 12-bit byte via the I/O channel. The external equipment generates a parity bit and compares it with the parity bit from the computer.

Connect: If a parity error exists in a Connect instruction, the external equipments:

- do not connect
- disconnect if already connected

- do not return an External Parity Error signal
- generally light a Parity Error indicator on the external equipment, and
- return neither a Reply nor a Reject signal.

After 100 usec the computer issues an Internal Reject.

Function and Write: If a parity error exists in a Function or Write instruction, the connected external equipment sends an External Parity Error signal back to the I/O channel. This signal causes the logic within the channel to provide a "1" on sense line zero. This logic is cleared every time an attempt is made to execute a Connect, Function, Read, or Write operation on this channel; however, these operations do not necessarily clear the logic in the external controller that transmits the External Parity Error signal. Thus to guarantee clearing this sense line the external equipment must also be cleared. Both the I/O channel and the external equipment may be channel cleared by the program or master cleared by the operator. If a transmission parity error is received from a controller, the controller remains inactive until both the external equipment and the I/O channel are cleared. A new I/O sequence must be initiated to continue or repeat the I/O operation.

Read

During a Read operation, the external equipment generates a parity bit and sends it to the I/O channel along with each 12-bit byte of data. The I/O channel holds the parity bit while the data is forwarded to the computation section. The Data Bus circuit of the computation section generates a second parity bit and sends it back to the I/O channel. The channel compares this second signal with the Parity signal which was generated by the external equipment. If an error exists, certain channel logic is set by an enable from the computation section. This logic provides a "1" on sense line zero. The channel parity logic is cleared every time an attempt is made to execute a Connect, Function, Read, or Write operation with this channel. It may also be channel cleared by the program or master cleared by the operator. If a transmission parity error is channel generated, it must be sensed by the INS instruction. If the error is not sensed, the next channel operation clears the error indication.

Parity Checking With the 3307

The computer checks parity with a 3307 in a slightly different manner than with a 3306.

Connect, Function, and Write

During the Connect, Function, and Write operations the Data Bus circuit in the computation section generates one parity bit for the lower 12-bit byte of data and one parity bit for the upper 12-bit byte. Both parity bits are sent to the external equipment via the I/O channel. The external equipment generates parity bits and compares them with the parity bits from the computer. The remainder of the parity checking is identical to that of the 3306 for Connect and for Function and Write.

Read

During a Read operation, the external equipment generates two parity bits per data word (one for each 12-bit byte) and sends them to the 3307 with the word. The 3307 holds the parity bits as the data is forwarded to the Data Bus circuit of the computer. Parity is generated in the Data Bus circuit and is sent back to the I/O channel where a comparison is made with the parity bits received from the external equipment.

If a parity error exists, the channel parity logic is set by an enable from the computation section, thus providing a "1" on sense line zero. Clearing the logic also occurs the same way as it does in the 3306. If a transmission parity error is channel generated, it must be sensed by the INS instruction. If the error is not sensed, the next channel operation clears the error indication.

TRANSMISSION RATES

The rate of transmitting each 12-bit word of I/O information depends upon the number of channels active, interregister transfers, the use of pause instructions to block out main control or the real-time clock, the length of connecting cables, and the use of multiprogramming. The 3000 Series Input/Output Specifications Manual, Pub. No. 60048800, describes in detail the measurement of these transfer rates using a variable-speed channel execiser. The exerciser measures the transfer rate by indicating a Lost Data condition when its speed exceeds that of the data channel. Word addressed I/O instructions with 12- to 24-bit Assembly/Disassembly were used.

Assuming a safe maximum transfer rate to the 10 percent slower than the average of the rates at which a Lost Data condition occurred, the following cases serve as examples of realizable transfer rates.

	Maximum Transfer Rate (12-bit word)
Without multiprogramming:	
1. Using a 3307 on channel 4, doing I/O only, blocking main control and the real-time clock with a Priority Pause instruction.	2.0 usec
2. Standard rate, no restrictions on program, channel 0 and 1 active, channel 0 is a 3307, channel 1 is a 3306.	8.0 usec (channel 0) 20.0 usec (channel 1)

With multiprogramming

1.	Using a 3306 on channel 0, doing I/O only, blocking main control and the real-time clock with a Priority Pause.	4.2 usec
2.	Standard rate, no restrictions on program, channels 0 and 4 active, both are 3306's.	16.0 usec (channel 0) 16.0 usec (channel 4)

The measured transfer rates when doing relocation were 0.2-.3 usec greater.

INPUT/OUTPUT RELOCATION

Data may be transmitted to or from several block locations in storage by using relocation. When an I/O instruction is encountered while executing a program in Executive Mode, Program State, an Executive Interrupt returns the computer to Monitor State. When a 3311 is present in the system the relocation of I/O information now occurs in the same manner as the relocation of a program. The monitor recognizes and assigns the appropriate I/O channels and devices. Whether or not relocation occurs, the largest block of data which may be transferred by a single I/O instruction is 32K 24-bit words.

AUTO LOAD/AUTO DUMP

The Auto Load and Auto Dump feature of the computer allows the programmer two groups of continuous storage locations for storing frequently used subroutines. These subroutines may be used whenever it is desirable to call in a particular tape unit or some other function that initiates an operation.

By depressing the AUTO LOAD console switch when the computer is stopped and in the non-Executive mode, the computer automatically jumps to address 77740 and executes the instruction stored there. The Auto Load routine is allotted sixteen addresses, 77740 through 77757.

Depressing the AUTO DUMP switch under the same conditions as Auto Load causes the computer to jump to address 77760 and execute the instruction stored there. Sixteen addresses, 77760 through 77777, may be used for the Auto Dump routine.

Although these storage areas may be used for any routine, the Auto Load area is generally used to bring in a program from a magnetic tape unit or other peripheral device. The last instruction in this routine should be a jump to the first address of the program just called in.

The Auto Dump area is most often used to output a block of data to a magnetic tape unit or other peripheral equipment and the last instruction in this routine can be a jump to any storage area within the confines of the system.

When the computer is operating in Executive mode, the Auto Load routine is stored in thirty-two locations encompassing addresses 003700 through 003737. The Auto Dump likewise has thirty-two locations ranging from address 003740 through 003777. The PA and PP designators of the page index associated with Page Index File zero are always zero thus providing a definitive area of storage (page zero) where the Auto Load and Auto Dump routines may be stored. The Auto Load and Auto Dump addresses are always protected in Non-Executive mode.

Examples of entering programs into the Auto Load and Auto Dump storage areas are given in Section 7.

4. INTERRUPT SYSTEM

GENERAL INFORMATION

The Interrupt System of a 3300 Computer can sense for the presence of certain internal and external conditions without having these tests in the main program. Examples of these conditions are internal faults and external equipment end-of-operation. Near the end of each RNI cycle, a test is made for interruptible conditions. If one of these conditions exists, and the interrupt system is enabled; execution of the main program halts, the contents of the Program Address register are stored, and an interrupt routine is initiated. This interrupt routine previously stored in memory, performs the necessary functions for the existing condition and then jumps back to the last unexecuted step in the main program. The instruction being read when the interrupt is recognized is executed when the main program is resumed.

There are seven categories of interrupts in the 3300 Computer: Internal Condition interrupts, I/O interrupts, Executive interrupt, Parity Error interrupt, Illegal Write interrupt, Trapped Instruction interrupts, and Power Failure interrupt. The store operations required for all types of interrupts occur regardless of the settings of the storage protection switches described in Section 2.

An additional programming feature is the MANUAL INTERRUPT switch on the operator's console. This interrupt is not masked since this switch is activated only when it is desirable to interrupt the computer, however, the interrupt system must be first enabled. The manual interrupt condition is automatically cleared after the interrupt is recognized.

When the 3300 is operating in the Program State of Executive mode, any interrupt that is recognized causes the processor to revert to the Monitor State. An Executive interrupt (described later in this section) also causes the processor to revert to the Monitor State if an attempt is made to execute one of several particular instructions.

INTERRUPT CONDITIONS

Internal Condition Interrupts

Any one of six internal conditions may cause an interrupt during the execution of a program. These conditions and their descriptions follow.

Arithmetic Overflow Fault

The Arithmetic Overflow fault is set when the capacity of the adder is exceeded. Its capacity, including sign, is 24 or 48 bits for 24-bit precision and 48-bit precision, respectively.

Divide Fault

The Divide fault sets if a quotient, including sign, exceeds 24 or 48 bits for 24-bit precision and 48-bit precision, respectively. Therefore, attempts to divide by too small a number, including positive and negative zero, result in a Divide fault. A Divide fault also occurs when a floating point divisor is either equal to zero or not in floating point format. The results in the A, Q, and E registers are insignificant if a fault occurs. A Divide fault can be correctly sensed only after the current instruction has been executed.

Exponent Overflow/Underflow Fault

During all floating point arithmetic operations, exponential overflow occurs if the exponent exceeds $+1777_8$ or is less than -1777_8. The fault is also set if the SFPF (77.71) instruction is executed.

BCD Fault

The BCD fault is generated by the BDP module if:

1. The lower 4 bits of any character in field A (except the sign character) exceed 11_8 during a numeric character operation.
2. The lower 4 bits of the sign character in field A exceed 12_8 during a numeric character operation.
3. The upper 2 bits of any character in field A (except the sign character) do not equal 00 during a numeric character operation.
4. An arithmetic carry out of the highest order character of field C occurs during an ADM or SBM instruction.
5. Field length $\S_1 > \S_2$ for an ADM or SBM instruction.
6. Field length $\S_1 \neq \S_2$ for a FRMT instruction, including provision for insertion characters.
7. A carry occurs out of the 14th character position during a CVBD instruction.
8. A field ($\S_1$) of more than 14 BCD characters is specified during a CVDB instruction.
9. Bits 05 and 06 of an ASCII character are both "1's" or both "0's" during the execution of an ATD instruction.

The BCD Fault may also be set by executing the SBCD (77.72) instruction.

Search/Move Interrupt

The Search/Move control may be programmed to generate an interrupt during a 71 or 72 instruction for either of the following conditions:

1. Completion or satisfaction of an equality or inequality search instruction (SRCE or SRCN).
2. Completion of a block move (MOVE instruction).

Real-Time Clock Interrupt

The Real-Time Clock interrupt is generated when the clock reaches a time previously stored in register 32 of the Register File.

Input/Output Interrupts

I/O Channel Interrupts

Any of the eight possible I/O channels may be programmed to generate an interrupt for either of the following conditions:

1. Reaching the end of an input or output block.
2. Receiving an End of Record (Disconnect) signal from an external device.

I/O Equipment Interrupt

The I/O equipment interrupt is set when an interrupt signal is received from any of eight peripheral equipment controllers connected to any of the eight possible I/O channels (there may be a total of 64 interrupt lines).

Associated Processor Interrupt

In a system of two or more processors (computers), each processor may interrupt, or be interrupted by, one other processor by executing an IAPR (77.57) instruction. This interrupt is not masked and becomes cleared as soon as it is recognized.

Executive Interrupt

The Executive Interrupt can only occur when the computer is operating in the Program State of Executive mode. An attempt to execute one of the following instructions then generates an Executive interrupt.

1. Halt instruction (00.0)
2. Inter-register transfer instructions with the Register File locations 00 through 37, [53.(4-7) (1-3) (XX00-XX37)]

3. Instructions with octal codes 71 through 77 except the 77.71 SFPF and 77.72 SBCD instructions

This interrupt is not masked and has priority over all of the internal condition interrupts. When the Executive interrupt has been recognized and the computer has reverted to the Monitor State, any of the instructions in the three categories above can be executed.

Storage Parity Error—No Response Interrupt

A Storage Parity Error interrupt has the highest priority of all interrupts and can occur if either a storage parity error is detected or if a storage module does not respond when referenced. The interrupt condition is recognized during the RNI and RADR sequence for an instruction.

The PARITY INTERRUPT switch on the console must be active for the interrupt to occur. If the PARITY STOP switch is active, the computer stops when a parity error or no-response condition is detected. The two switches cannot be simultaneously active, and pressing the PARITY STOP switch overrides the Parity Interrupt condition.

If Block Control has storage priority at the time of interrupt, the address of the next instruction to be executed is stored in the lower 15 bits of location 00020. The appropriate register file location contains the approximate address where the error occurred. An interrupt during Main Control priority causes the address of the current instruction to be stored in location 00020. If the error condition is detected during any of the RNI's for the BDP instructions, (P) is always stored at location 00020. Detecting the condition during either RNI for the 71 - 76 instructions results in either (P), (P + 1), or (P + 2) being stored.

A code representing conditions within the processor at the time of interrupt is automatically stored in the lower 12 bits of location 00021. A RNI is then performed at location 00021. The stored address and code enable the interrupt routine to isolate the storage area where the error occurred and aid in program recovery. Table 4-0 lists the various codes and their interpretations.

The instruction in progress when the interrupt is detected may be executed although the results are not necessarily correct. Once the parity error or no-response condition is detected, additional errors are not recognized until a DINT (77.73) instruction is executed.

TABLE 4-0. PARITY ERROR INTERRUPT CODES

Reason for Interrupt	Type of Operation or Sequence in Progress	Code
No-Response	Block Control - (73-76)	00X0 X = ch
Parity Error	Block Control - (73-76)	00X2 X = ch
No-Response	Block Control - 71, 72, or typewriter I/O	01X0 (X=0, Srch), (X=1, Move), (X=3, TWR)
Parity Error	Block Control - 71, 72, or typewriter I/O	01X2 (X=0, Srch), (X=1, Move), (X=3, TWR)
No-Response	Main Control - RNI or RADR	00X1 (X=0, RNI) (X=2, RADR)
Parity Error	Main Control - RNI or RADR	00X3 (X=0, RNI) (X=2, RADR)
No-Response	Main Control - ROP or STO	0005
Parity Error	Main Control - ROP or STO	0007

Illegal Write Interrupt

This interrupt has priority over all interrupts except the Storage Parity Error interrupt. The interrupt condition may result during a RNI, RADR, ROP, or STO sequence; however it is recognized only during RNI or RADR. When the condition is recognized, the interrupt system is disabled, (P) are automatically stored at address 00014, and an RNI is performed at address 00015.

The system must be in Program State of Executive mode to recognize the interrupt. The interrupt is disabled during Monitor State and during Search/Move and I/O cycles. The conditions for the interrupt are listed below. (Conditions 3 through 6 apply only if the 3311 Multiprogramming Module is present in the system.)

1. A Write operation into an area protected by the Storage Protect switches (Program State 0).
2. A Keyboard Write operation into the Executive Auto Load/Auto Dump area (addresses 03700 through 03777).
3. A Read or Write operation when bits 9 and 10 of the original address specify a quarter page equal to or greater than PL, when PL ≠ 0.
4. A Read or Write operation if the 'E' designator for any referenced index equals "1" and PA, PL, and PP are equal to zero.
5. A Write operation if the 'E' designator for any referenced index equals "1" and PA, PL, or PP is not equal to zero.
6. A double precision instruction if the first operand is to be read from the last available memory location specified by PL, or if from the last memory location when PL specifies a full page and the next index to be used contains 4000.

Bit 05 of the internal status sensing network is set on an Illegal Write interrupt only if the condition occurred during a RNI or RADR sequence. If the condition occurred during a ROP or STO sequence, the interrupt is generated but bit 05 is not set. If one of the 66.0 - 66.5 instructions is interrupted by an Illegal Write, the instruction always restarts at the beginning when the main program resumes. Other BDP instructions restart from the point of interrupt.

Trapped Instruction Interrupts

If an attempt is made to execute one of the instructions listed in Table 4-1 and the system is not equipped with a 3310 Floating Point module or 3312 BDP, the instruction becomes trapped. Only those instructions preceded by an asterisk (*) are trapped if the 3312 BDP is not present in the system and the 3310 Floating Point module is present.

TABLE 4-1. TRAPPED INSTRUCTIONS FOR NON-EXECUTIVE MODE WITHOUT A 3310 OR 3312 MODULE IN SYSTEM (MNEMONIC LISTING)

ELQ	*MVE	*SCAN, LR, NE, DC	*UPAK
EUA	*MVE, DC	*SCAN, RL, EQ	*ADM
EAQ	*MVBF	*SCAN, RL, EQ, DC	*SBM
QEL	*MVZF	*SCAN, RL, NE	*CMP
AEU	*MVZS	*SCAN, RL, NE, DC	*CMP, DC
AQE	*MVZS, DC	*CVDB	*TST
MUAQ	*ZADM	*CVBD	*TSTN
DVAQ	*FRMT	*DTA	*JMP, HI
FAD	*EDIT	*DTA, DC	*JMP, LOW
FSB	*SCAN, LR, EQ	*ATD	*JMP, ZRO
FMU	*SCAN, LR, EQ, DC	*ATD, DC	*SBR
FDV	*SCAN, LR, NE	*PAK	*LBR

NOTE

DTA, DTA dc, ATD and ATD dc instructions are available in 3312 and 3304-2 only.

Each instruction listed in Table 4-2 is processed as a no-Operation instruction, (refer to Section 5) if an attempt is made to execute one of them while operating in the non-Executive mode.

TABLE 4-2. NO-OPERATION INSTRUCTIONS FOR NON-EXECUTIVE MODE (MNEMONIC LISTING)

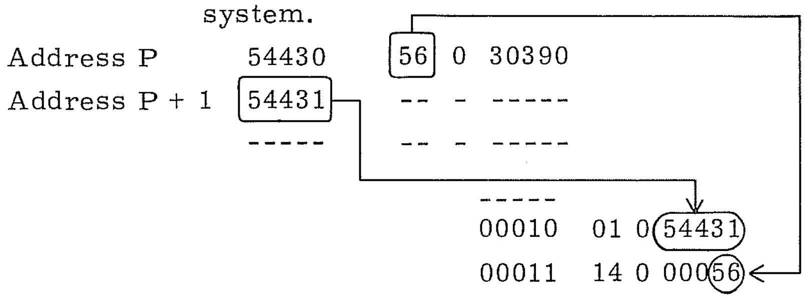
ACI	ISA	ROS
AIS	JAA	SBJP
AOS	OSA	CRA SDL
APF	PFA	~~SRA~~
CIA	~~RCR~~ ACR	TMAV
	RIS	

Although they are not true interrupts, trapped instructions are processed like interrupts once they have been detected. A conventional interrupt always takes priority over a trapped sequence.

The following operations take place when a trapped instruction is recognized:

1. The address of the next sequential program step, P + 1, is stored in the lower 15 bits of address 00010.
2. The upper 6 bits of the instruction in the F register are stored in the lower 6 bits of the operand stored at address 00011. The upper 18 bits of this operand remain unchanged.
3. Program execution commences at address 00011.

EXAMPLE: MUAQ (56) Instruction execution attempt without the Floating Point/Double Precision hardware option in the system.



```
Address P        54430   56 0 30390
Address P + 1    54431   -- - -----
                 -----   -- - -----

                         -----
                         00010   01 0 54431
                         00011   14 0 00056
```

At this point the MUAQ operation may be simulated by software and re-entry to the main program is possible by a jump to the contents of address 00010.

Power Failure Interrupt

If source power to the computer system fails, the power failure is detected and the computer program is interrupted. This interrupt is necessary to prepare a controlled shutdown and prevent the loss of data. The operation requires 16 ms for detection, and up to 4 ms for processing the special Power Failure interrupt routine.

The Power Failure interrupt overrides any other interrupt except the Illegal Write and Storage Parity Error interrupts, regardless of the state of interrupt control. Since this interrupt overrides all others, the address where the present contents of P are stored and the address to which program control is transferred must be different from that for a normal interrupt. When a Power Failure interrupt occurs, the machine stores the contents of P in the lower 15 bits of address 00002 and transfers program control to address 00003.

The normal interrupt system is disabled during a power failure sequence; i. e., the hardware simulates the execution of a DINT (77. 73) instruction.

INTERRUPT CONTROL

Through the use of certain instructions, a program can recognize, sense, and clear interrupts, and enable or disable the interrupt system.

Enabling or Disabling Interrupt Control

Instruction EINT (77.74) enables the interrupt system and the DINT instruction (77.73) disables it. After recognizing an interrupt and entering the interrupt sequence, other interrupts are disabled automatically. When leaving the interrupt subroutine, the interrupt system must again be enabled by the EINT instruction if interrupts that are waiting or subsequent interrupts are to be recognized by the system. Refer to the EINT (77.74) instruction in Section 5 for special conditions regarding the actual interruption of the CPU.

Interrupt Priority

An order of priority exists between the various interrupt conditions. As soon as an interrupt becomes active, the computer scans the priority list until it reaches an interrupt that is active (not necessarily the interrupt that initiated the scanning). The computer processes this interrupt and the scanner returns to the top of the list where it waits for another active interrupt to appear. Table 4-3 lists the order of priority.

TABLE 4-3. INTERRUPT PRIORITY

PRIORITY	TYPE OF INTERRUPT	PRIORITY	TYPE OF INTERRUPT
1	Storage Parity Error	8	BCD Fault
2	Illegal Write	9-72	I/O Equipment (External)*
3	Power Failure	73-80	I/O Channel**
4	Executive	81	Search/Move
5	Arithmetic Overflow	82	Real-time Clock
6	Divide Fault	83	Manual
7	Exponent Overflow/ Underflow	84	Associated Processor
		85	Trapped Instruction

* There are eight interrupt lines on each of the eight possible I/O channels, or 64 lines in all. On any given channel, a lower numbered line has priority over a higher numbered line. Likewise, a lower numbered channel has priority over a higher numbered channel. Example: line 0 of channel 0 has highest priority of all external I/O interrupts, line 0 of channel 1 has second highest, and line 7 of channel 7 has the lowest.

** A lower numbered I/O channel interrupt has priority over a higher numbered I/O channel interrupt.

Sensing Interrupts

The programmer may selectively sense interrupts by using the INTS (77. 4) instruction. Sensing the presence of internal faults automatically clears them. Interrupt lines representing channels not present in the system are sensed as being active. The interrupt system need not be enabled for sensing.

Clearing Interrupts

Internal condition interrupts are cleared by:

- Sensing with an INTS (77. 4) or INS (77. 3), after which interrupts are automatically cleared,
- Executing an INCL (77. 50) instruction
- Executing an IOCL (77. 51) instruction - clears only Search/Move interrupt, or
- Pressing the MC or INTERNAL CLEAR buttons.

I/O channel interrupts are cleared by:

- Executing an INCL (77. 50), IOCL (77. 51), or CLCA (77. 512) instruction, or
- Pressing the MC or EXTERNAL CLEAR buttons.

I/O equipment interrupts are cleared by:

- Executing an IOCL (77. 51) instruction,
- Reselecting or releasing the interrupt with a SEL (77. 1) instruction, or,
- Pressing the MC or EXTERNAL CLEAR buttons.

The manual and associated processor interrupts are automatically cleared upon recognition by the computer.

INTERRUPT PROCESSING

Four conditions must be met before an Internal Condition, Executive, or I/O interrupt can be processed:

1. A bit representing the interrupt condition must be set to "1" in the Interrupt Mask register (except for Manual, Associated Processor, and Executive interrupts).
2. The interrupt system must have been enabled (except for Executive Interrupt).
3. An interrupt-causing condition must exist.

4. The interrupt scanning logic (Refer to Table 4-3) must reach the level of the active interrupt on the priority list.

When an active interrupt has met the above conditions, the following takes place:

1. The instruction in progress proceeds until the point is reached in the RNI or RADR cycle where an interrupt can be recognized. At this time the count in P has not been advanced nor has any operation been initiated. When an interrupt is recognized, the address of the current unexecuted instruction in P is stored in address 00004.
2. A number representing the interrupt-causing condition is stored in the lower 12 bits of address 00005 without modifying the upper bits. Table 4-4 lists the octal codes which are stored for each interrupt condition.
3. Program control is transferred to address 00005 and an RNI cycle is executed.

TABLE 4-4. REPRESENTATIVE INTERRUPT CODES

Conditions	Codes
External interrupt	*00LCh
I/O channel interrupt	010Ch
Real-Time Clock interrupt	0110
Arithmetic overflow fault	0111
Divide fault	0112
Exponent overflow fault	0113
BCD fault	0114
Search/move interrupt	0115
Manual interrupt	0116
Associated processor interrupt	0117
Executive Interrupt	0120

*L = line 0-7 and Ch = channel designator, 0-7

INTERRUPT MASK REGISTER

The programmer can choose to honor or ignore an interrupt by means of the Interrupt Mask register. All but three of the normal interrupt conditions are represented by the 12 Interrupt Mask register bits. The Manual, Associated Processor and Executive interrupts are not masked. The mask is selectively set with the SSIM (77. 52) instruction and selectively cleared by the SCIM (77. 53) instruction. See Table 4-5 for Interrupt Mask register bit assignments.

The contents of the Interrupt Mask register may be transferred to the upper 12 bits of the A register for programming purposes with the COPY (77. 2) or CINS (77. 3) instructions.

TABLE 4-5. INTERRUPT MASK REGISTER BIT ASSIGNMENTS

Mask Bits	Mask Codes	Interrupt Conditions Represented
00	0001	I/O Channel 0 (Includes interrupts generated within the channel and external equipment interrupts.)
01	0002	1
02	0004	2
03	0010	3
04	0020	4
05	0040	5
06	0100	6
07	0200	7
08	0400	Real-time clock
09	1000	Exponent overflow/underflow and BCD faults
10	2000	Arithmetic overflow and divide faults
11	4000	Search/Move completion

INTERRUPTS DURING EXECUTIVE MODE

Although all interrupts can be recognized during Executive mode, special consideration must be given to handling these interrupts. During Executive mode, the Condition register records current operating information that must temporarily be stored in the event of interrupt to enable proper recovery. Table 4-6 lists the Condition register bit assignments.

TABLE 4-6. CONDITION REGISTER BIT ASSIGNMENTS

Bit	Condition Represented	
00	Boundary Jump -	Set by SBJP (77. 62) instruction. Cleared by next jump instruction.
01	Destructive Load A -	Set by SDL (77. 624) instruction. Cleared by next LDA instruction.
02	Operands Relocated Using OSR -	Set by ROS (55. 4) instruction. Cleared by RIS (55. 0) instruction.
03	Program State Jump -	Set by any jump during Program State. Cleared when jumping to Program State.
04	Interrupt System Enabled -	Set by EINT (77. 74) instruction. Cleared by DINT (77. 73) instruction.
05	Program State -	Set when jumping to Program State.

To insure the processing of stacked interrupts, it is necessary to transfer these conditions to the A register at the start of the interrupt routine by executing a CRA (77. 63) instruction. At the completion of the interrupt routine, these conditions must be restored by executing a ACR (77. 634) instruction.

Upon interrupt recognition, the interrupt system is automatically disabled and the Central Processor enters the Monitor State. The Condition register and all interrupts, except Trapped Instruction interrupts, are disabled during the intervals between interrupt recognition and CRA instruction execution, and between execution of the ACR instruction and the jump instruction normally used to exit from an interrupt routine.

The Condition register is cleared as the transfer to A is completed; the interrupt system remains disabled until a EINT (77.74) or ACR instruction is executed.

INTERRUPTS DURING BDP INSTRUCTIONS

Interrupts are recognized near the end of the first RNI of all instructions. However, after the first RNI of BDP instructions, Main Control continually tests for active interrupt conditions. If a selected interrupt (or Abnormal interrupt) condition becomes active, an Interrupt Stop signal is sent to the BDP section. The BDP relinquishes control after the current character operation is completed. The interrupt is actually recognized as Main Control rereads the instruction at P, or at the address of the next instruction if the current instruction was completed.

The BDP records interrupt recovery conditions (refer to the LBR instruction), and enables operating information to the B^3 register. If recovery from interrupts is desired, the interrupt routine used must contain a SBR instruction to store the recorded interrupt recovery conditions, and a LBR instruction to return the recovery conditions to the BDP once the interrupt processing is completed. These conditions normally enable a restart to be made from the point of interrupt. Exceptions to the recovery start are: the 66.0 and 66.1 instructions always restart from the beginning if interrupted, and if the interrupt is because of an Illegal Write, the instructions 66.2 through 66.5 (3312 and 3304-2) and 66.4 and 66.5 (3304-3) also restart from the beginning.

The (B^3) register has the following significance when a BDP instruction is interrupted:

Bits 00 - 11, record the count of the Field C characters processed prior to interrupt.

Bit 12 = "1", if a second pass (complementing operation) was in progress.

Bit 13 = "1", if an arithmetic carry was generated on a ADM or SBM instruction.

Bit 14 = "1", if a BCD fault occurred.

5. INSTRUCTIONS

GENERAL INFORMATION

A 3300 machine coded instruction word is 24 bits in length and may require up to three sequential words for a particular function. Although there are 24 distinct instruction formats, as illustrated in Appendix D, there are several that are used more frequently than others. These formats (word oriented, character oriented, and business oriented) are shown in the following pages along with their appropriate instruction parameters.

Instruction Parameters

The following parameters are used in the 3300 instruction list. A capitalized letter generally indicates a modified parameter, however, this is not always the case and the specific instruction should be consulted. Some parameters are general in nature, i.e., specifying a character address, but in some instances may indicate a high order address and in others a low order address. The parameter descriptions listed below each instruction format should be checked for the explicit memory of the parameters for that particular instruction.

If an octal number appears in the format of a specific instruction, only that number must be placed in the exact position as indicated. In cases where only a single binary position is involved, a "1" or "0" is used depending upon the instruction. The following parameters are used throughout the 3300 instruction list:

A = (1) variable length field of characters designated field A in BDP instructions; usually the transmitting field.

(2) A is used in the descriptions for instructions (other than those for BDP) to indicate the A register.

a = addressing mode (a = "0" for direct addressing, a = "1" for indirect addressing)

B = "1" for backward storage

b = index register designator 1, 2, or 3.

B_m = index register flag for a field in certain BDP instructions. The flag indicates which index register will have its contents added to the unmodified address 'm'. $M = m + [B_m]$ for these instructions only.

B_n = index register flag for a field in certain BDP instructions where both fields are specified by word addresses. The flag indicates which index register will have its contents added to the unmodified address 'n'.

B_r = index register flag for field A in most BDP instructions (refer to individual instruction descriptions). Initial character address of field A is defined: $R = r + [B_r]$. If $B_r = 1$ or 3, use (B^1); if $B_r = 2$, use (B^2); if $B_r = 0$, no indexing is performed and $R = r$.

B_s = index register flag for field C in most BDP instructions (refer to individual instruction descriptions). Initial character address of field C is defined: $S = s + [B_s]$. If $B_s = 1$ or 3 use (B^1); if $B_s = 2$, use (B^2); if $B_s = 0$, no indexing is performed and $S = s$.

C = variable length field of characters designated field C in BDP instructions. Usually the receiving field.

ch = denotes I/O channel (0 through 7).

DC = indicates delimiting character position within the instruction word or mnemonic. Generally, a delimiting character of 6 or 8 bits is specified in an instruction and if a character is recognized, during the particular operation, that equals the delimiting character, the operation is terminated.

G = "1" for word count control with the INPC and INPW instructions.

H = indicates special Assembly/Disassembly operation in certain character oriented I/O instructions.

INT = "1" for interrupt upon completion in certain I/O instructions.

I = assembly language designator indicating indirect addressing.

i = internal parameter (decrement or increment).

j = Jump designator.

k = (1) unmodified shift count for SHA, SHQ, and SHAQ instructions.
(2) scale factor for SCAQ instruction.

K = (1) modified shift count, $K = k + (B^b)$ for SHA, SHQ, and SHAQ instructions.
(2) residue quantity for SCAQ instruction.

$\mathcal{S}$ = field length of data block for MOVE instruction.

$\mathcal{S}_1$ = number of characters in BDP field A (character count).

$\mathcal{S}_2$ = number of characters in BDP field C (character count).

m = unmodified 15-bit storage word address.

M = modified 15-bit storage address. $M = m + (B^b)$.

n = same as 'm', but the word address of the second operand for certain I/O instructions.

N = indicates special Assembly/Disassembly operation in certain word oriented I/O instructions.

r = unmodified 17-bit character address.

R = modified 17-bit character address:
$R = r + (B_r)$ for BDP instructions. (Refer also to 'B_r'.)
$R = r + (B^b)$ for all other instructions.

s = same as 'r', but the character address of the second operand for certain I/O instructions.

S = (1) modified 17-bit character address of field C for BDP instructions only. $S = s + (B_s)$. (Refer also to 'B_s'.)

(2) also used to denote sign extension for certain instructions.

SC = 6-bit comparison scan character used in search instructions. May be used with DC.

v = a specific register number (00-77) within the Register File.

w = 7-bit Page Index File address. (Refer to APF and PFA instructions and Appendix E for additional information.)

x = connect code or interrupt mask.

y = 15-bit operand

z = 17-bit operand

////// = slashing indicates a particular area of an instruction that should be loaded with zeros although the particular area is not used for the instruction.

In addition to the instruction parameters, various abbreviations are used in the instruction descriptions that refer to various registers and operations. These abbreviations and their literal meanings are listed here:

ISR	=	Instruction State register
OSR	=	Operand State register
CIR	=	Channel Index register
CR	=	Condition register
BCR	=	BDP Condition register
PIF	=	Page Index File

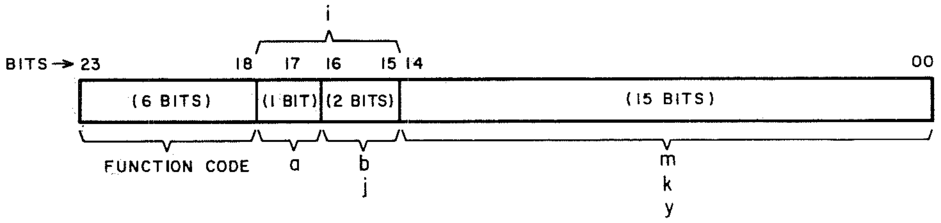
(A_{00-02}) = contents of the lower 3 bits (00, 01, and 02) of the A register

$[B_r]$ = contents of the index register as defined by the value of the B_r flag.

$m<n$	=	m "less than" n
$m>n$	=	m "greater than" n
$m\geq n$	=	m "greater than or equal to" n
$m\bullet n$	=	logical product of m and n
$(m)\rightarrow n$	=	contents of m, transferred to n
$\vee$	=	Exclusive OR function
$\wedge$	=	AND function

Instruction Word Formats

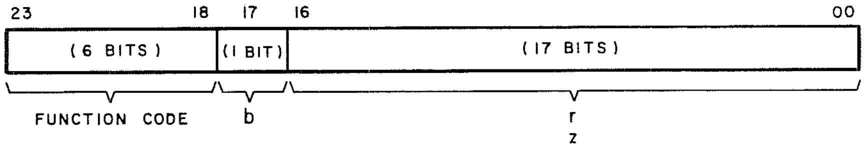
Word oriented instructions are the most common of the instruction formats. Fifteen bits are allocated for an unmodified storage address, operand, or shift count. Indirect addressing is usually available. Figure 5-1 illustrates a word oriented instruction and the significance of the first 15 bits when they represent an unmodified word address 'm'.



Symbol designators
(See Symbol Definitions)

Figure 5-1. Word-Addressed Instruction Format

Character oriented instructions allocate 17 bits for unmodified character addresses or extended operands. Indirect addressing is not available for these instructions; however, address modification is permissible by referencing a specific index register. Figure 5-2 illustrates the format of a character oriented instruction word and the significance of the first 17 bits when they represent an unmodified character address 'r'.



Characters in a data word are always specified in the following manner:

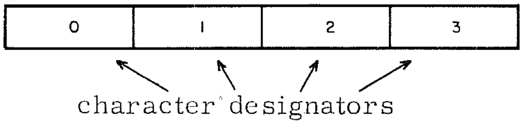


Figure 5-2. Character-Addressed Instruction Format

Word Addressing/Character Addressing

It is often desirable to convert a word address and character position to its corresponding character address or vice versa. The following procedure is a technique used for this purpose.

To convert a word address to a character address:

- Octally multiply the word address by four. (During program execution, this operation is simulated by a left shift of two binary places.)
- Add the character position to the product.

The sum is the character address.

EXAMPLE:

Given: Word address 12442, character position 2

Find: Corresponding character address

```
1.       12442
            x4
         -----
         52210
2.          +2
         -----
         52212 = character address
```

To convert a character address to a word address:

- Octally divide the character address by four

The quotient is the word address and the remainder is the character position. No remainder indicates character zero.

EXAMPLE:

Given: Character address 03442

Find: Word address and character position

```
         00710
       4|03442
          34
          --
           4
           4
           -
            2 = remainder = character position 2
```

NOTE

Octal multiplication and division tables may be found in Appendix C of this manual.

Instruction word formats that differ from word and character orientation are described in the instruction listing.

Business oriented instructions require three instruction words to completely define an operation. These instructions are executed only by the BDP. These subinstruction words are always located at consecutive memory locations, nominally designated P, P+1, and P+2.

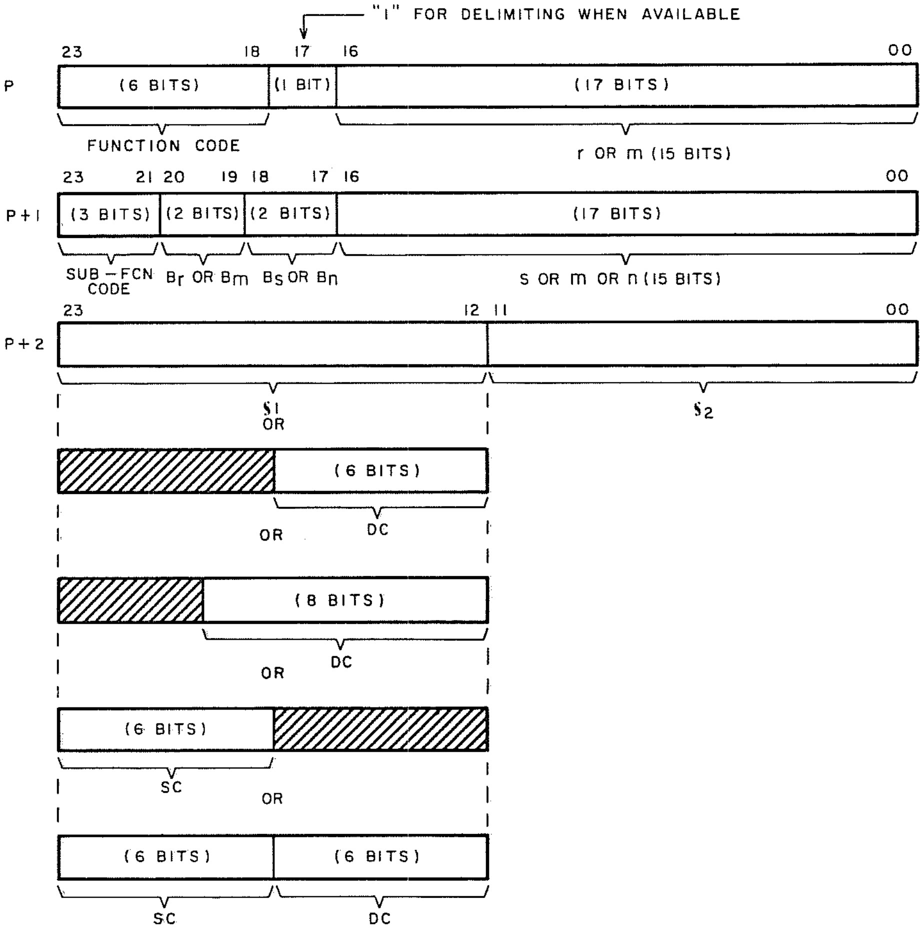


Figure 5-3. Business Oriented Instruction Format

Indexing and Address Modification

In some instructions, the execution address 'm' or 'r', or the shift count 'k' may be modified by adding to them the contents of an index register, B^b. The 2-bit designator 'b' specifies which of the three index registers is to be used. Symbols representing the respective modified quantities are M, R, and K.

$M = m + (B^b)$
$R = r + (B^b)$ the sign of B^b is extended to bit 16 $(2^{17}-1)$
$K = k + (B^b)$

In each case, if b = 0, then M = m, R = r and K = k.

Special index considerations apply to BDP instructions where an index register flag is present. A flag defines which index register is used for indexing:

EXAMPLE:

> If $s = 00413$, $B_s = 2$, and $(B^2 = 00364$, then $S = s + [B_s]$ or "the modified address 'S' equals the unmodified address 's' added to the contents of the index register as defined by B_s".

Thus:

$$S = s + [B_s]$$

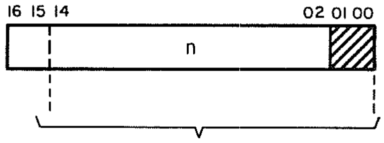
$$S = 00413 + (B^2)$$

$$S = 00413 + 00364$$

$$S = 00777$$

Some BDP instructions, i.e., PAK, CVBD, DTA, etc. utilize both word and character addresses in their formats. Although the first two bits preceding the address are unused and not part of the word address, the lower 15 bits of this word are added to the contents of the specified index register. The lower two bits of the specified index register must be set to "1's" to allow for an end-around carry during the index addition.

EXAMPLE:



$[B_n]$ are added to the first 15 bits

Addressing Modes

Three modes of addressing are used in the computer: No Address, Direct Address, and Indirect Address.

No Address

This mode is used when an operand 'y' or a shift count 'k' is placed directly into the lower portion of an instruction word. Symbols 'a' and 'b' are not used as addressing mode and index designators with any of the no address instructions.

Direct Address

The direct addressing mode is used in any instruction in which an operand address 'm' is stored in the lower portion of the initial instruction word. This mode is specified by making 'a' equal to 0. In many instructions, address 'm' may be modified (indexed) by adding to it the contents of register B^b, $M = m + (B^b)$.

Indirect Address

It is possible to use indirect addressing only with instructions that require an execution address 'm'. For applicable instructions, indirect addressing is specified by making 'a' equal to 1. Several levels (or steps) of indirect addressing may be used to reach the execution address; however, execution time is delayed in direct proportion to the number of steps. The search for a final execution address continues until 'a' equals 0. It is important to note that direct or indirect addressing and address modification are two distinct and independent steps. In any particular instruction, one may be specified without the other. Figure 5-4 shows the indirect addressing routine.

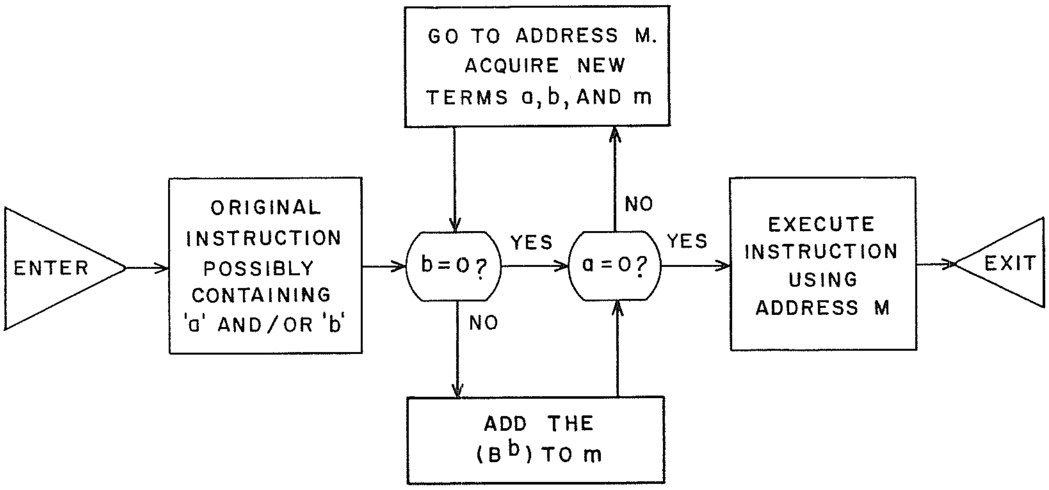

GO TO ADDRESS M. ACQUIRE NEW TERMS a, b, AND m
ENTER
ORIGINAL INSTRUCTION POSSIBLY CONTAINING 'a' AND/OR 'b'
b = 0?
YES
NO
a = 0?
NO
YES
EXECUTE INSTRUCTION USING ADDRESS M
EXIT
ADD THE (B^b) TO m

Figure 5-4. Indexing and Indirect Addressing Routine Flow Chart

NOTE

Unless it is otherwise stated, indirect addressing follows the above routine throughout the list of instructions.

Indexing and Indirect Addressing Examples

The following examples utilize the LDA (20) instruction; however, the process applies to any of the instructions with an 'a' and/or 'b' designator.

EXAMPLE 1:

(ADDRESS MODIFICATION - (indexing) ONLY)

```
P = 00000     20 2 54430                                        (B2) = 13342
                 ↑                                  ↓
          Indicates Direct Address          Add this address to (B2)
          mode and address modification             ↓
          by B2                                   54430
                                                 +13342
              20 2 67772←This address is replaced←67772
                         temporarily in the
                         original instruction
```

LDA with the 24-bit quantity stored at address 67772

```
          67771      --------
P =       67772      77700000→This quantity is loaded into the A register
          67773      --------
```

EXAMPLE 2:

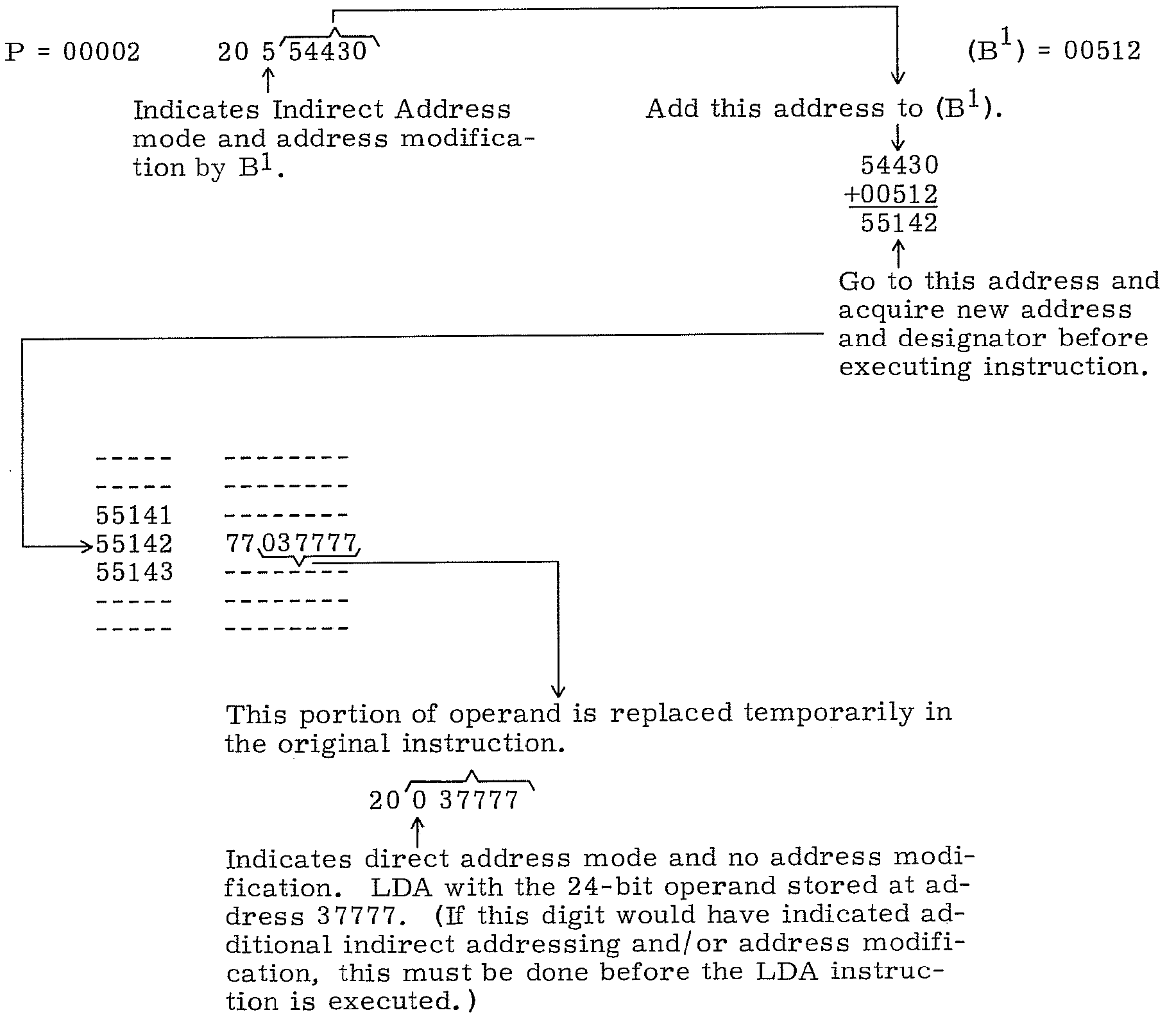
(INDIRECT ADDRESSING ONLY)

```
P = 00001     20 4 54430
                 ↑                                  ↓
         Indicates Indirect Addressing      Go to this address and acquire
         mode but no address modification   new address and designator be-
         (indexing).                        fore executing instruction.
     -----    --------
     -----    --------
     54427    --------
    →54430    310 77111
     54431    --------       ↓
     -----    --------   This portion of operand is replaced temporarily
     -----    --------   in the original instruction.
                             ↓
                  20 0 77111
                     ↑
```

Indicates Direct Address mode and no address modification. LDA with the 24-bit operand stored at address 77111. (If this digit would have indicated additional indirect addressing and/or address modification this must be done before the LDA instruction is executed.)

EXAMPLE 3:

(INDIRECT ADDRESSING AND ADDRESS MODIFICATION)



P = 00002   20 5 54430 (B^1) = 00512

Indicates Indirect Address mode and address modification by B^1.

Add this address to (B^1).

```
 54430
+00512
 55142
```

Go to this address and acquire new address and designator before executing instruction.

```
-----     --------
-----     --------
55141     --------
55142     77 0 37777
55143     --------
-----     --------
-----     --------
```

This portion of operand is replaced temporarily in the original instruction.

20 0 37777

Indicates direct address mode and no address modification. LDA with the 24-bit operand stored at address 37777. (If this digit would have indicated additional indirect addressing and/or address modification, this must be done before the LDA instruction is executed.)

Trapped Instructions

Certain instructions are trapped if the optional hardware module, necessary for their direct execution, is not present in the system. These instructions, and the routine followed in processing trapped instructions, are described in Section 4.

INSTRUCTION LIST

Each group of instructions is introduced with an index, and whenever necessary, a group description. Individual instructions are all presented in the same basic format:

- Heading, which includes the assembly language mnemonic and instruction name or function.
- Machine code instruction format.
- Parameters and their descriptions.
- Instruction description.
- Comments (when necessary).

The instructions are grouped into general functions, i.e., Loads, Stores, Arithmetic, etc., and the groups are arranged according to the complexity of their functions. This arrangement permits a programmer, unfamiliar with the instruction list, to progress through this Section with relative ease.

The abbreviation, RNI, is used throughout the list of instructions to indicate the Read Next Instruction sequence. This is a sequence of steps taken by the control section to advance the computer to its next program step. For an extensive description of this sequence, consult the 3300 Customer Engineering Manual.

Table 5-1 identifies the instructions by mnemonic and indicates on which page detailed instruction descriptions may be found.* Table 5-2 is a summary of the instruction execution times. In addition to these tables, additional tables are provided at the end of this manual for cross reference of the instruction list.

*The letter "I" after a mnemonic code in Table 5-1 as well as in each group of instructions, and in the mnemonic and octal listings in the last section of this manual, indicates that indirect addressing may be used for that instruction.

TABLE 5-1. INSTRUCTION SYNOPSIS AND INDEX

Mnemonic	Instruction	Page No.
ACI	Transmit (A) to CIR	5-38
ACR	Transmit (A) to CR	5-40
ADA, I	Add to A	5-60
ADAQ, I	Add to AQ	5-61
ADM	Add field A to field C	5-154
AEU	Transmit (A) to EU	5-36
AIA	Transmit (A) + (B^b) to A	5-33
AIS	Transmit (A) to ISR	5-37
ANA	Logical product (AND) of y and (A)	5-71
ANA, S	Logical product (AND) of y and (A), sign extended	5-71
ANI	Logical product (AND) of y and (B^b)	5-71
ANQ	Logical product (AND) of y and (Q)	5-72
ANQ, S	Logical product (AND) of y and (Q), sign extended	5-72
AOS	Transmit (A) to OSR	5-37
APF	Transmit (A_{00-11}) to PIF	5-39
AQA	Transmit (A) + (Q) to A	5-32
AQE	Transmit (AQ) to E	5-36
AQJ, EQ	jump if (A) = Q	
AQJ, NE	jump if (A) ≠ Q	
	Compare A with Q	5-46
AQJ, GE	jump if (A) ≥ Q	
AQJ, LT	jump if (A) < Q	
ASE	Skip next instruction, if (A) = y	5-29
ASE, S	Skip next instruction if (A) = y, sign extended	5-29
ASG	Skip next instruction if (A) ≥ y	5-30
ASG, S	Skip next instruction if (A) ≥ y, sign extended	5-30
ATD*	Convert ASCII to BCD	5-150
ATD, DC*	Convert ASCII to BCD delimiting character possibility	5-151
AZJ, EQ	jump if (A) = 0	
AZJ, NE	jump it (A) ≠ 0	
	Compare A with zero	5-45
AZJ, GE	jump if (A) ≥ 0	
AZJ, LT	jump if (A) < 0	

*Available in 3312 and 3304-2

TABLE 5-1. INSTRUCTION SYNOPSIS AND INDEX (Cont'd)

Mnemonic	Instruction	Page No.
CIA	Transmit (CIR) to A	5-38
CILO	Channel interrupt lockout	5-86
CINS	Copy internal status	5-81
CLCA	Clear channel activity	5-89
CMP*	Compare field A with field C, exit if ≠	5-158
CMP, DC**	Compare field A with field C, exit if ≠, delimiting character possibility	5-162
CMP**	Collating compare of field A with field C	5-159
CMP**	Numeric compare of field A with field C	5-160
CON	Connect	5-90
COPY	Copy external status	5-78
CPR, I	Within limits test	5-75
CRA	Transmit (CR) to A	5-40
CTI	Set console typewriter input	5-94
CTO	Set console typewriter output	5-94
CVBD	Convert binary to BCD	5-147
CVDB	Convert BCD to binary	5-146
DINT	Disable interrupt control	5-84
DTA*	Convert BCD to ASCII	5-148
DTA, DC*	Convert BCD to ASCII, delimiting character possibility	5-149
DVA, I	Divide AQ (48 by 24)	5-62
DVAQ, I	Divide AQE (96 by 48)	5-63
EAQ	Transmit (E_U) to A and (E_L) to Q	5-36
ECHA	Enter A with 17 bit character address	5-26
ECHA, S	Enter A with 17 bit character address, sign extended	5-26
EDIT	Edit field A, move to field C	5-132
EINT	Enable interrupt control	5-84
ELQ	Transmit (E_L) to Q	5-36
ENA	Enter A	5-25
ENA, S	Enter A, sign extended	5-25
ENI	Enter index	5-25
ENQ	Enter Q	5-25
ENQ, S	Enter Q, sign extended	5-25
EUA	Transmit (E_U) to A	5-36
EXS	Sense external status	5-78

*Available in 3312 and 3304-2 only.

**Available in the 3304-3 only.

TABLE 5-1. INSTRUCTION SYNOPSIS AND INDEX (Cont'd)

Mnemonic	Instruction	Page No.
FAD, I	Floating add to AQ	5-65
FDV, I	Floating divide AQ	5-66
FMU, I	Floating multiply AQ	5-66
FRMT	Formatted edit of field A, move to field C	5-130
FSB, I	Floating subtract from AQ	5-65
HLT	Unconditional halt; read next instruction from location m	5-24
IAI	Transmit $(B^b) + (A)$ to B^b	5-33
IAPR	Interrupt associated processor	5-110
IJD	Index jump; decrement index	5-44
IJI	Index jump; increment index	5-43
INA	Increase A	5-27
INA, S	Increase A, sign extended	5-27
INAC, INT	Character-addressed input to A	5-103
INAW, INT	Word-addressed input to A	5-104
INCL	Clear interrupt	5-84
INI	Increase index	5-27
INPC, INT,B,H,A	Character-addressed input to storage	5-95
INPW, INT,B,N,A	Word-addressed input to storage	5-97
INQ	Increase Q	5-27
INQ, S	Increase Q, sign extended	5-27
INS	Sense internal status	5-80
INTS	Sense interrupt	5-79
IOCL	Clear I/O, typewriter, and S/M	5-89
ISA	Transmit (ISR) to A	5-37
ISD	Index skip; decrement index	5-31
ISE	Skip next instruction if $y = 0$	5-28
ISE	Skip next instruction if $(B^b) = y$	5-28
ISG	Skip next instruction if $y \geq 0$	5-30
ISG	Skip next instruction if $(B^b) \geq y$	5-30
ISI	Index skip; increment index	5-31

TABLE 5-1. INSTRUCTION SYNOPSIS AND INDEX (Cont'd)

Mnemonic	Instruction	Page No.
JAA	Transmit jump address to A	5-40
JMP, HI	Jump if BDP register > 0 or +	5-42
JMP, ZRO	Jump if BDP register = 0	
JMP, LOW	Jump if BDP register < 0 or -	
LACH	Load A character	5-49
LBR	Load BDP Condition register	5-167
LCA, I	Load A complement	5-50
LCAQ, I	Load AQ complement (double precision)	5-51
LDA, I	Load A	5-49
LDAQ, I	Load AQ (double precision)	5-50
LDI, I	Load index	5-52
LDL, I	Load logical	5-50
LDQ, I	Load Q	5-51
LPA, I	Logical product with A	5-71
LQCH	Load Q character	5-52
MEQ	Masked equality search	5-73
MOVE,INT	Move (S) characters from r to s	5-115
MTH	Masked threshold search	5-74
MUA, I	Multiply A	5-62
MUAQ, I	Multiply AQ	5-63
MVBF	Move and blank fill	5-125
MVE	Move	5-123
MVE, DC	Move, delimiting character possibility	5-124
MVZF	Move and zero fill	5-126
MVZS	Move and zero suppress	5-127
MVZS, DC	Move and zero suppress, delimiting character possibility	5-128
OSA	Transmit (OSR) to A	5-37
OTAC, INT	Character-addressed output from A	5-106
OTAW, INT	Word-addressed output from A	5-107
OUTC, INT, B, H	Character-addressed output from storage	5-99
OUTW, INT, B,N	Word-addressed output from storage	5-101
PAK	Pack 6 bit BCD characters into 4 bit BCD characters	5-152

TABLE 5-1. INSTRUCTION SYNOPSIS AND INDEX (Cont'd)

Mnemonic	Instruction	Page No.
PAUS	Pause	5-82
PFA	Transmit (PFI) to A	5-39
PRP	Priority pause	5-83
QEL	Transmit (Q) to E_L	5-36
QSE	Skip next instruction if (Q) = y	5-29
QSE, S	Skip next instruction if (Q) = y, sign extended	5-29
QSG	Skip next instruction if (Q) $\geq$ y	5-30
QSG, S	Skip next instruction if (Q) $\geq$ y, sign extended	5-30
RAD, I	Replace add	5-60
RIS	Relocate to instruction state	5-109
ROS	Relocate to operand state	5-109
RTJ	Return jump	5-47
SACH	Store character from A	5-54
SBA, I	Subtract from A	5-61
SBAQ, I	Subtract from AQ	5-61
SBCD	Set BCD fault	5-86
SBJP	Set boundary jump	5-109
SBM	Subtract field A from field C	5-156
SBR	Store BDP Condition register	5-168
SCA, I	Selectively complement A	5-70
SCAN, LR, EQ, DC	left to right, stop on =	5-139
SCAN, LR, NE, DC	stop on ≠	5-141
	Scan — Delimiting character possibility	
SCAN, RL, EQ, DC	right to left, stop on =	5-143
SCAN, RL, NE, DC	stop on ≠	5-145
SCAN, LR, EQ	left to right, stop on =	5-138
SCAN, LR, NE	stop on ≠	5-140
	Scan	
SCAN, RL, EQ	right to left, stop on =	5-142
SCAN, RL, NE	stop on ≠	5-144
SCAQ	Scale AQ	5-59

TABLE 5-1. INSTRUCTION SYNOPSIS AND INDEX (Cont'd)

Mnemonic	Instruction	Page No.
SCHA, I	Store 17-bit character address from A	5-56
SCIM	Selectively clear interrupt mask	5-85
SDL	Set destructive load	5-110
SEL	Select function	5-92
SFPF	Set floating point fault	5-86
SHA	Shift A	5-57
SHAQ	Shift AQ	5-59
SHQ	Shift Q	5-59
SJ1	Jump if key 1 is set	5-41
SJ2	Jump if key 2 is set	5-41
SJ3	Jump if key 3 is set	5-41
SJ4	Jump if key 4 is set	5-41
SJ5	Jump if key 5 is set	5-41
SJ6	Jump if key 6 is set	5-41
SLS	Selective stop	5-24
SQCH	Store character from Q	5-55
SRCE, INT	Search character equality	5-111
SRCN, INT	Search character inequality	5-113
SSA,.I	Selectively set A	5-70
SSH	Storage shift	5-57
SSIM	Selectively set interrupt mask	5-85
STA, I	Store A	5-53
STAQ, I	Store AQ	5-54
STI, I	Store index	5-56
STQ, I	Store Q	5-55
SWA, I	Store 15-bit word address from A	5-56
TAI	Transmit (A) to B^b	5-33
TAM	Transmit (A) to high speed memory	5-34
TIA	Transmit (B^b) to A	5-33
TIM	Transmit (B^b) to high speed memory	5-35
TMA	Transmit (high speed memory) to A	5-34

TABLE 5-1. INSTRUCTION SYNOPSIS AND INDEX (Cont'd)

Mnemonic	Instruction	Page No.
TMAV	Test memory availability	5-77
TMI	Transmit (high speed memory) to B^b	5-35
TMQ	Transmit (high speed memory) to Q	5-34
TQM	Transmit (Q) to high speed memory	5-34
TST	Test field A for +, -, 0	5-165
TSTN	Test field A for numeric	5-166
UCS	Unconditional Stop	5-24
UJP, I	Unconditional Jump	5-41
UPAK	Unpack 4 bit BCD characters into 6 bit BCD characters	5-153
XOA	Exclusive OR y and (A)	
XOA, S	Exclusive OR y and (A) , sign extended	
XOI	Exclusive OR y and (B^b)	5-69
XOQ	Exclusive OR y and (Q)	
XOQ, S	Exclusive OR y and (Q) , sign extended	
ZADM	Zero and add	5-129

No-Operation Instructions

When an attempt is made to execute one of the following instructions at the current execution address, P, the computer recognizes them as No-Operation (NO-OP) instructions and advances to the next execution address, P + 1. In mnemonics a No-Operation instruction is written as: NOP.

NO-OPERATION
OCTAL CODES

02 0
14 0
15 0
16 0
17 0

During non-Executive mode operation each of the following instructions are recognized as No-Operation instructions if an attempt is made to execute one of them. Also refer to Trapped Instruction processing, Section 4.

ACI
AIS
AOS
APF
CIA

ISA
JAA
OSA
PFA
~~RCR~~
RIS ACR

ROS
SBJP
SDL
~~SRA~~ CRA
TMAV

Instruction Execution Times

Except for the 64.0 through 77.1 instructions, an actual instruction execution time consists of the base execution time listed plus the time for an RNI cycle. If indexing or indirect addressing is used, their execution times must be added to base instruction time.

Relocation time is added only if the RNI or RADR cycle preceding the RNI or RADR currently in progress was in a different memory page or if the ROP or STO cycle preceding the ROP or STO in progress was in a different memory page. That is, by programming RNI's and RADR's in the same page and ROP's and STO's in the same page, relocation processing time can be minimized.

Table 5-2 is an octal list of all 3300 instructions with their base execution times. During certain multiple cycle instructions, it is possible to execute other instructions concurrently, thus no additional execution time is required.

TABLE 5-2. SUMMARY OF INSTRUCTION EXECUTION TIMES, USEC

Instruction Processing Operation	Time Added to Base Execution Time (usec)
RNI cycle	1.375
Indexing (address modification)	.375
Relocation	.250*
Indirect addressing	1.375

*The time added to base execution time is 0.150 usec if the Multiprogramming Module is present and no relocation is performed.

TABLE 5-2 (Cont'd)

Basic Octal Code	Mnemonic Code	Execution Time (usec)	Basic Octal Code	Mnemonic Code	Execution Time (usec)
00	HLT	0.000	17	ANI	0.000
00	SJ1-6	0.000	17	ANA	0.000
00	RTJ	1.375	17	ANQ	0.000
01	UJP	0.000	20	LDA	1.375
02	IJI	0.000	21	LDQ	1.375
02	IJD	0.000	22	LACH	1.375
03	AZJ	0.625	23	LQCH	1.375
03	AQJ	0.625	24	LCA	1.375
04	ISE	0.625	25	LDAQ	2.625
04	ASE	0.625	26	LCAQ	2.625
04	QSE	0.625	27	LDL	1.375
05	ISG	0.625	30	ADA	1.375
05	ASG	0.625	31	SBA	1.375
05	QSG	0.625	32	ADAQ	2.625
06	MEQ	2.375 + 2.5n	33	SBAQ	2.625
07	MTH	2.375 + 2.5n	34	RAD	2.625
10	SSH	2.625	35	SSA	1.375
10	ISI	0.625	36	SCA	1.375
10	ISD	0.625	37	LPA	1.375
11	ECHA	0.000	40	STA	1.375
12	SHA	0.000 to 1.375	41	STQ	1.375
12	SHQ	0.000 to 1.375	42	SACH	1.375
13	SHAQ	0.000 to 1.375	43	SQCH	1.375
13	SCAQ	0.875 to 2.250	44	SWA	1.375
14	ENI	0.000	45	STAQ	2.625
14	ENA	0.000	46	SCHA	1.375
14	ENQ	0.000	47	STI	1.375
15	INI	0.000	50	MUA	6.875 to 9.875
15	INA	0.000	51	DVA	10.250
15	INQ	0.000	52	CPR	1.375 to 2.625
16	XOI	0.000	53	TIA	0.000
16	XOA	0.000	53	TAI	0.000
16	XOQ	0.000	53	TMQ	0.625

TABLE 5-2 (Cont'd)

Basic Octal Code	Mnemonic Code	Execution Time (usec)	Basic Octal Code	Mnemonic Code	Execution Time (usec)
53	TQM	0.625	67	ADM & SBM	③ 9+3.1♉+ 0.75 $\S_2$
53	TMA	0.625			⑤ 13+3.1♉+ 1.65 $\S_2$
53	TAM	0.625	67	ZADM	10.7 + 0.9$\S_2$
53	TMI	0.625	67 †	CMP	10.7 + 0.9T
53	TIM	0.625	67 †	CMP	16.4 + .9n
53	AQA	0.000	67 ††	CMP, N	10.7 + .9T
53	AIA	0.000	67	TST	⑥ 9.4
53	IAI	0.000			⑦ 8.5 + 0.9 $\S_1$
54	LDI	1.375	67	TSTN	⑦ 8.5 +0.9 $\S_1$
55	RIS	0.000	70	LBR	4.9
56	MUAQ	16.0 to 19.0	70	JMP	1.44
57	DVAQ	25.5	70	SBR	4.9
60	FAD	4.850 to 6.250	71	SRCE	*
61	FSB	4.850 to 6.250	71	SRCN	*
62	FMU	16.0	72	MOVE	*
63	FDV	19.0	73	INPC	*
64	MVE	10.7 + 0.9 S	73	INAC	*
64	MVBF	10.7 + 0.9$\S_2$	74	INPW	*
64	MVZF	12.9 + 0.9$\S_2$	74	INAW	*
64	MVZS	10.7 + 0.9$\S_2$	75	OUTC	*
64	FRMT	10.7 + 0.9$\S_2$	75	OTAC	*
64	EDIT	① 9+3.1♉+ 0.75 $\S_2$	76	OUTW	*
		② 13+3.1♉+1.65 $\S_2$	76	OTAW	*
65	SCAN	8.5 + 0.9 $\S_2$	77	CON	*
66	CVBD	17.9 + 0.92 [α (1+M) + $\beta M_1 + 8N_1 + 20N_2 + 28N_3$]	77	SEL	*
66	CVDB	③ 13.7 + 0.92 ($3N_1$ + $6N_2$ + $8N_3$)	77	EXS	0.000
		④ 13.7 + 0.92 ($4N_1$ + $10N_2$ + $14N_3$ - 1)	77	COPY	0.000
66 †	DTA	10.7 + 1.1 S	77	INS	0.000
66 †	ATD	10.7 + 1.1 S	77	CINS	0.000
66	PAK	10.7 + 0.9 $\S_1$	77	INTS	0.000
66	'UPAK	10.7 + 0.9$\S_1$	77	INCL	0.000
			77	IOCL	0.000
			77	CILO	0.000
			77	CLCA	0.000
			77	SSIM	0.000
			77	SCIM	0.000

†3312 and 3304-2
††3304-3 only

TABLE 5-2 (Cont'd)

Basic Octal Code	Mnemonic Code	Execution Time (usec)	Basic Octal Code	Mnemonic Code	Execution Time (usec)
77	ACI	0.000	77	AOS	0.000
77	CIA	0.000	77	AIS	0.000
77	JAA	0.000	77	OSA	0.000
77	IAPR	0.000	77	ISA	0.000
77	PAUS	0.0 to 40 ms	77	SLS	0.000
77	PRP	0.0 to 40 ms	77	SFPF	0.000
77	TMAV	1.375 or 6.375 us	77	SBCD	0.000
77	SBJP	0.000	77	DINT	0.000
77	SDL	0.000	77	EINT	0.000
77	CRA	0.000	77	CTI	*
77	ACR	0.000	77	CTO	*
77	APF	0.000	77	UCS	0.000
77	PFA	0.000			

n = number of characters searched

$\S_1$ = number of characters in source field (A)

$\S_2$ = number of characters in result field (C)

M = number of most significant 4-bit binary groups (in the lower 24 bits to be converted) which have a zero value

M_1 = number of most significant 4-bit binary groups (in the upper 24 bits to be converted) which have a zero value (see example)

N_1 = number of characters up to and including three in number (For the CVBD instruction, the term character defines a binary group of 4 bits to the right of any consecutive lead groups which are all zero. See example.)

N_2 = number of characters from three, up to and including seven in number. (See example.)

N_3 = number of characters greater than seven in number (For the CVBD instruction, the term character defines a binary group of 4 bits to the right of any consecutive lead groups which are all zero. See example.)

S = number of characters in the smaller of fields $\S_1$ and $\S_2$.

α = If all upper 24 bits to be converted are zero, $\alpha = 1$; if not $\alpha = 0$.

β = If one or more of the upper 24 bits to be converted is a "1", $\beta = 1$, if not $\beta = 0$.

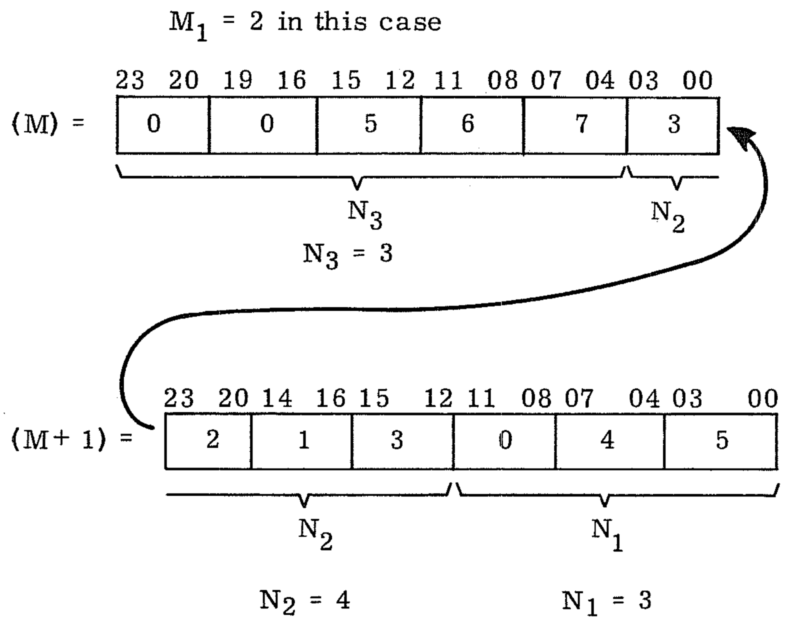
T = number of characters in the longer of fields $\S_1$ and $\S_2$.

* = Dependent upon a variable signal response time from an external equipment or internal source; i. e., Block Control.

γ = number of 4 character groups in field $\S_2$.

① = best case (no second pass)
② = worst case (second pass required)
③ = best case (no carry propagation)
④ = worst case (maximum carry propagation)
⑤ = worst case (maximum carry propagation and second pass required)
⑥ = best case (field ≠ zero)
⑦ = worst case (entire field = zero)

SPECIAL PARAMETER EXAMPLES

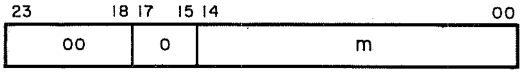


Data words are divided into six 4-bit groups internally by the CPU and are shown here only to illustrate the parameters necessary for determining the instruction execution time.

Halt and Stop Instructions

Operation Field	Address Field	Interpretation
HLT 00 SLS 77 UCS 77	m	Halt Selective stop Unconditional stop

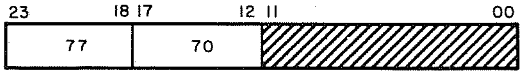
HLT
Halt



Instruction Description: Unconditionally halt at this instruction. Upon restarting, RNI from address m.

Comments: Indirect addressing and address modification may not be used.

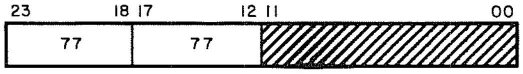
SLS
Selective Stop



Instruction Description: Program execution halts if the SELECT STOP switch on the console is set. RNI from address P + 1 upon restarting.

Comments: Bits 00 through 11 should be loaded with zeros.

UCS
Unconditional Stop



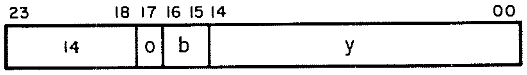
Instruction Description: This instruction unconditionally stops the execution of the current program. RNI from address P + 1 upon restarting.

Comments: Bits 00 through 11 should be loaded with zeros.

Enter Instructions

Operation Field		Address Field	Interpretation
ENI	14	y, b	Enter index b with y
ENA	14	y	Enter A with y
ENA, S	14	y	Enter A with y and extend sign of y
ENQ	14	y	Enter Q with y
ENQ, S	14	y	Enter Q with y and extend sign of y
ECHA	11	z	Enter A with z
ECHA, S	11	z	Enter A with z and extend sign of z

ENI
Enter Index with y

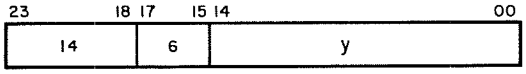


b = index register designator

Instruction Description: Clear index register B^b and enter y directly into it.

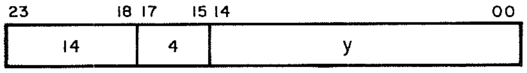
Comments: If b = 0, this is a no-operation instruction.

ENA
Enter A with y



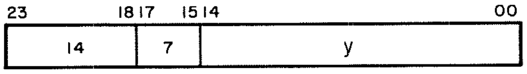
Instruction Description: Clear the A register and enter y directly into A.

ENA, S
Enter A with y, Sign Extended



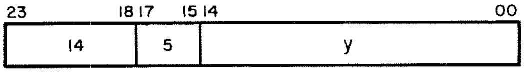
Instruction Description: Same as ENA except the sign of y is extended.

ENQ
Enter Q with y



Instruction Description: Clear the Q register and enter y directly into Q.

ENQ, S
Enter Q with y, Sign Extended



Instruction Description: Same as ENQ except the sign of y is extended.

ECHA
Enter Character Address into A

23 18	17	16 00
11	0	z

Instruction Description: Clear A; then enter a 17-bit operand z (usually a character address) into A.

ECHA, S
Enter Character Address into A, Sign Extended

23 18	17	16 00
11	1	z

Instruction Description: Clear A; then enter a 24-bit operand (17-bit z plus 7 bits of sign extension) into A.

NOTE

> If is often desirable to perform operations with the A or Q registers using a negative operand. By using the sign extension feature of certain instructions, 14-bit negative operands become available. This feature eliminates the need, in many instances, to reference prestored operands.

The following examples illustrate the use of sign extension in some instructions:

EXAMPLE A: To enter negative zero into Q, execute a 14 5 77777.

EXAMPLE B: To increase (A) by -17, execute a 15477760 instruction.

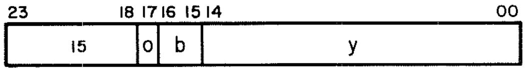
```
                      (A) = 00066667  (arbitrary value)
                            77777760
                            00066647
   (end around carry)              1
                            00066650 =(A) after instruction execution
```

In all cases of sign extension, bit 14 for 15-bit y operands and bit 16 for 17-bit z operands determines the sign of the quantity.

Increase Instructions

Operation Field		Address Field	Interpretation
INI	15	y, b	Increase index by y
INA	15	y	Increase A by y
INA, S	15	y	Increase A by y, sign extended
INQ	15	y	Increase Q by y
INQ, S	15	y	Increase Q by y, sign extended

INI
Increase Index by y

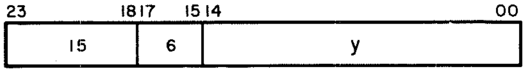


b = index register designator

Instruction Description: Add y to (B^b).

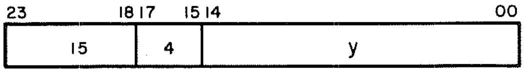
Comments: If b = 0, this is a no-operation instruction. Signs of y and B^b are extended.

INA
Increase A by y



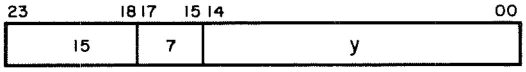
Instruction Description: Add y to (A).

INA, S
Increase A by y, Sign Extended



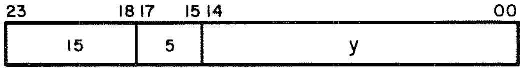
Instruction Description: Same as INA except the sign of y is extended.

INQ
Increase Q by y



Instruction Description: Add y to (Q).

INQ, S
Increase Q by y, Sign Extended



Instruction Description: Same as INQ except the sign of y is extended.

Skip Instructions

Operation Field		Address Field	Interpretation
ISE	04	y, b	Skip next instruction if $(B^b) = y$
ASE	04	y	Skip next instruction if (A) = y
ASE, S	04	y	Skip next instruction if (A) = y. Sign of y is extended.
QSE	04	y	Skip next instruction if (Q) = y
QSE, S	04	y	Skip next instruction if (Q) = y. Sign of y is extended.
ISG	05	y, b	Skip next instruction if $(B^b) \geq y$
ASG	05	y	Skip next instruction if $(A) \geq y$
ASG, S	05	y	Skip next instruction if $(A) \geq y$. Sign of y is extended.
QSG	05	y	Skip next instruction if $(Q) \geq y$
QSG, S	05	y	Skip next instruction if $(Q) \geq y$. Sign of y is extended.
ISI	10	y, b	Index skip, incremental
ISD	10	y, b	Index skip, decremental

NOTE

The SSH (10. 0) instruction, which also uses a Skip exit, is described in the Shift and Scale Instruction group.

ISE
Skip Next
Instruction if $(B^b) = y$

23 18	17	16 15	14 00
04	0	b	y

b = index register designator

Instruction Description: If $(B^b) = y$, skip to address P+2; if not, RNI from address P+1.

Comments: If b = 0, y is compared to zero.

ASE
Skip Next
Instruction if (A) = y

23 18	17 15	14 00
04	6	y

Instruction Description: If (A) = y, skip to address P+2; if not, RNI from address P+1.

Comments: Only the lower 15 bits of A are used for this instruction.

ASE, S
Skip Next
Instruction if (A) = y, Sign Extended

23 18	17 15	14 00
04	4	y

Instruction Description: Same as ASE except the sign of y is extended. All 24 bits of A are recognized.

QSE
Skip Next
Instruction if (Q) = y

23 18	17 15	14 00
04	7	y

Instruction Description: If (Q) = y, skip to address P+2; if not, RNI from address P+1.

Comments: Only the lower 15 bits of Q are used for this instruction.

QSE, S
Skip Next
Instruction if (Q) = y, Sign Extended

23 18	17 15	14 00
04	5	y

Instruction Description: Same as QSE except the sign of y is extended. All 24 bits of Q are recognized.

ISG
Skip Next
Instruction if (B^b) ≧ y

23 18	17	16 15	14 00
05	0	b	y

b = index register designator

Instruction Description: If (B^b) are equal to or greater than y, skip to address P+2; if not, RNI from address P+1. (B^b) and y are 15-bit positive numbers.

Comments: If b = 0, y is compared to zero.

ASG
Skip Next
Instruction if (A) ≧ y

23 18	17 15	14 00
05	6	y

Instruction Description: If (A) are equal to or greater than y, skip to address P+2; if not, RNI from address P+1. Only the lower 15 bits of A are used.

Comments: (A_{L15}) and y are considered 15-bit positive numbers.

ASG, S
Skip Next
Instruction if (A) ≧ y, Sign Extended

23 18	17 15	14 00
05	4	y

Instruction Description: Same as ASG except the sign of y is extended. All 24 bits of A are recognized. Positive zero (00000000) is recognized as greater than negative zero (77777777).

QSG
Skip Next
Instruction if (Q) ≧ y

23 18	17 15	14 00
05	7	y

Instruction Description: If (Q) are equal to or greater than y, skip to address P+2; if not, RNI from address P+1. Only the lower 15 bits of Q are used.

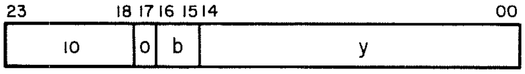
Comments: (Q_{L15}) and y are considered 15-bit positive numbers.

QSG, S
Skip Next
Instruction if (Q) ≧ y, Sign Extended

23 18	17 15	14 00
05	5	y

Instruction Description: Same as QSG except the sign of y is extended. All 24 bits of Q are recognized. Positive zero (00000000) is recognized as greater than negative zero (77777777).

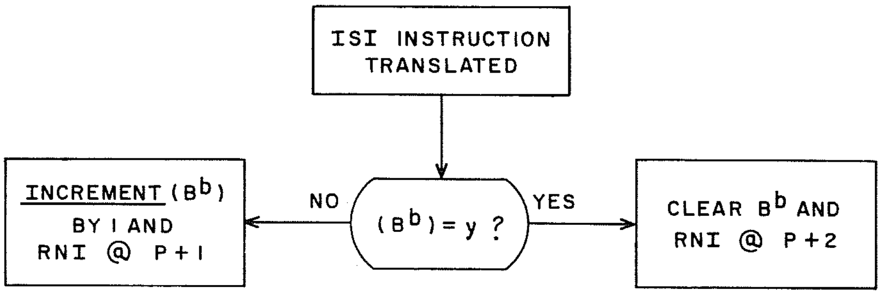
ISI
Index Skip, Incremental



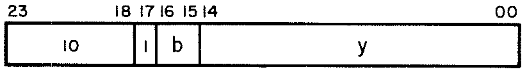
b = index register designator

Instruction Description: If (B^b) = y, clear B^b and skip to address P+2; if not, add one to (B^b) and RNI from address P+1.

Comments: The 10.0 instruction is a SSH (storage-shift) instruction, described later in this section.


ISI INSTRUCTION TRANSLATED
$(B^b) = y$?
NO
YES
INCREMENT (B^b) BY 1 AND RNI @ P+1
CLEAR B^b AND RNI @ P+2

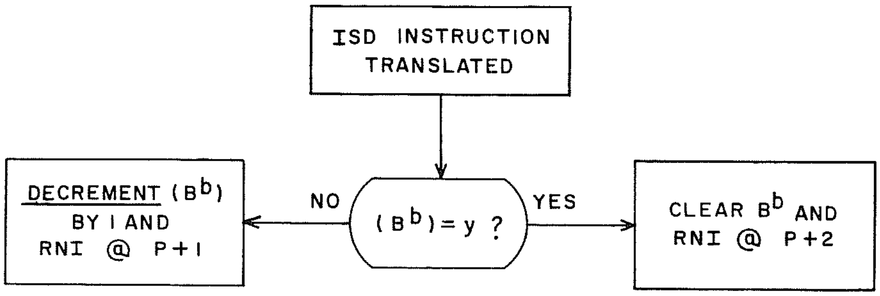
ISD
Index Skip, Decremental



b = index register designator

Instruction Description: If (B^b) = y, clear B^b and skip to address P+2; if not, subtract one from (B^b) and RNI from address P+1.

Comments: When b = 0, RNI from P+1 if $y \neq 0$; RNI from P+2 if y = 0.

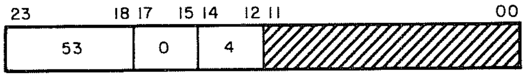

ISD INSTRUCTION TRANSLATED
$(B^b) = y$?
NO
YES
DECREMENT (B^b) BY 1 AND RNI @ P+1
CLEAR B^b AND RNI @ P+2

Inter-Register Transfer Instructions

Operation Field		Address Field	Interpretation
AQA	53		Transfer (A) + (Q) to A
AIA	53	b	Transfer (A) + (B^b) to A
IAI	53	b	Transfer (B^b) + (A) to B^b
TIA	53	b	Transfer (B^b) to A
TAI	53	b	Transfer (A) to B^b
TMQ	53	v	Transfer (Register v) to Q
TQM	53	v	Transfer (Q) to Register v
TMA	53	v	Transfer (Register v) to A
TAM	53	v	Transfer (A) to Register v
TMI	53	v, b	Transfer (Register v) to B^b
TIM	53	v, b	Transfer (B^b) to Register v
ELQ	55		Transfer (E_L) to Q
EUA	55		Transfer (E_U) to A
EAQ	55		Transfer (E_UE_L) to AQ
QEL	55		Transfer (Q) to E_L
AEU	55		Transfer (A) to E_U
AQE	55		Transfer (AQ) to E_UE_L
AIS	77		Transfer (A_{00-02}) to ISR
ISA	77		Transfer (ISR) to A_{00-02}
AOS	77		Transfer (A_{00-02}) to OSR
OSA	77		Transfer (OSR) to A_{00-02}
ACI	77		Transfer (A_{00-02}) to CIR
CIA	77		Transfer (CIR) to A_{00-02}
APF	77	w, b	Transfer (A_{00-11}) to PIF location 'w'
PFA	77	w, b	Transfer (PIF location 'w') to A_{00-11}
CRA	77		Transfer (CR) to A_{00-05}
ACR	77		Transfer (A_{00-05}) to CR
JAA	77		Transfer LJA to A_{00-14}

AQA

(A) + (Q) → A

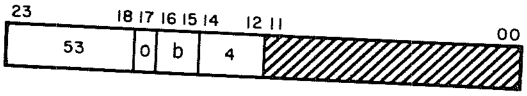


Instruction Description: Add the (A) to the (Q) and transfer the sum to A.

Comments: (Q) remain unchanged. Bits 00 through 11 should be loaded with zeros.

AIA

$(A)+(B^b)\rightarrow A$



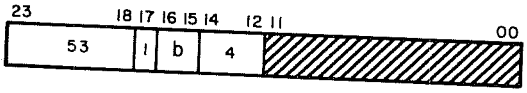
b = index register designator

Instruction Description: Add the (A) to the (B^b) and transfer the sum to A.

Comments: Bits 00 through 11 should be loaded with zeros. The sign of (B^b) is extended prior to the addition. (B^b) remain unchanged.

IAI

$(B^b)+(A)\rightarrow B^b$



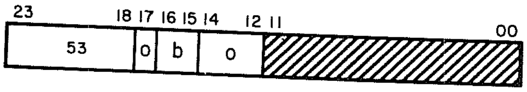
b = index register designator

Instruction Description: Add the (A) to the (B^b) and transfer the sum to B^b.

Comments: Bits 00 through 11 should be loaded with zeros. The sign of the original (B^b) is extended prior to the addition. The upper 9 bits of the sum are lost when the sum is transferred to the Index register. If b=0, this becomes a No-Op instruction.

TIA

$(B^b)\rightarrow A$



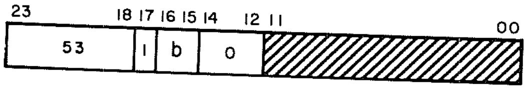
b = index register designator

Instruction Description: Transfer the (B^b) to A.

Comments: Bits 00 through 11 should.be loaded with zeros. No sign extension on B^b. Prior to the transfer, (A) are cleared. If b = 0, zeros are transferred to A.

TAI

$(A)\rightarrow B^b$

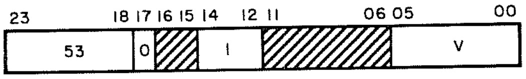


b = index register designator

Instruction Description: Transfer the (A) to B^b.

Comments: Bits 00 through 11 should be loaded with zeros. The (A) remain unchanged. If b = 0, this becomes a no-operation instruction.

TMQ
$(v) \rightarrow Q$

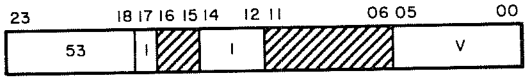


v = register file number, $00\text{-}77_8$

Instruction Description: Transfer the (v) to Q.

Comments: Bits 06 through 11, 15 and 16 should be loaded with zeros.

TQM
$(Q) \rightarrow v$

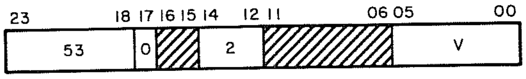


v = register file number, $00\text{-}77_8$

Instruction Description: Transfer the (Q) to v.

Comments: Bits 06 through 11, 15 and 16 should be loaded with zeros.

TMA
$(v) \rightarrow A$

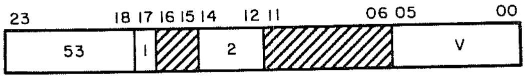


v = register file number, $00\text{-}77_8$

Instruction Description: Transfer the (v) to A.

Comments: Bits 06 through 11, 15 and 16 should be loaded with zeros.

TAM
$(A) \rightarrow v$



v = register file number, $00\text{-}77_8$

Instruction Description: Transfer the (A) to v.

Comments: Bits 06 through 11, 15 and 16 should be loaded with zeros.

TMI
$(v) \rightarrow B^b$

23 18	17	16 15	14 12	11 06	05 00
53	0	b	3	▨	v

b = index register designator
v = register file number, $00\text{-}77_8$

Instruction Description: Transfer the lower 15 bits of (v) to B^b.

Comments: Bits 06 through 11 should be loaded with zeros. If 'b' = 0, this becomes a no-operation instruction.

TIM
$(B^b) \rightarrow v$

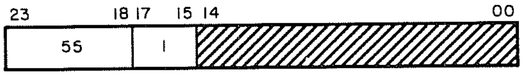
23 18	17	16 15	14 12	11 06	05 00
53	1	b	3	▨	v

b = index register designator
v = register file number, $00\text{-}77_8$

Instruction Description: Transfer (B^b) to v. The upper nine bits of 'v' are cleared.

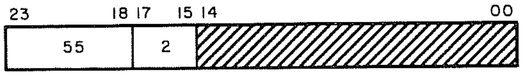
Comments: Bits 06 through 11 should be loaded with zeros. If 'b' = 0, all of (v) are cleared.

ELQ
$(E_L) \rightarrow Q$



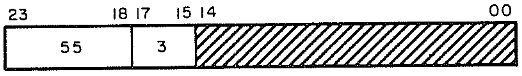
Instruction Description: Transfer the (E_L) to Q. Bits 00-14 should be loaded with zeros.

EUA
$(E_U) \rightarrow A$



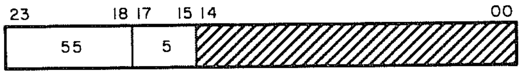
Instruction Description: Transfer the (E_U) to A. Bits 00 - 14 should be loaded with zeros.

EAQ
$(E) \rightarrow AQ$



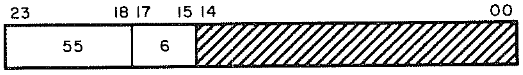
Instruction Description: Transfer the (E_UE_L) to AQ. Bits 00 - 14 should be loaded with zeros.

QEL
$(Q) \rightarrow E_L$



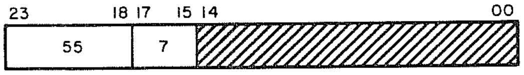
Instruction Description: Transfer the (Q) to E_L. Bits 00 - 14 should be loaded with zeros.

AEU
$(A) \rightarrow E_U$



Instruction Description: Transfer the (A) to E_U. Bits 00 - 14 should be loaded with zeros.

AQE
$(AQ) \rightarrow E$



Instruction Description: Transfer the (AQ) to E_UE_L. Bits 00 - 14 should be loaded with zeros.

AIS

$(A_{00\text{-}02}) \rightarrow ISR$

23 18	17 12	11 09	08 00
77	66	4	000

Instruction Description: Transfer $(A_{\text{bits 00, 01, and 02}})$ to the Instruction State register.

ISA

$(ISR) \rightarrow A_{00\text{-}02}$

23 18	17 12	11 09	08 00
77	67	4	0 0 0

Instruction Description: Transfer (ISR) to $A_{\text{(bits 00, 01, and 02)}}$.

AOS

$(A_{00\text{-}02}) \rightarrow OSR$

23 18	17 12	11 00
77	66	0000

Instruction Description: Transfer $(A_{\text{bits 00, 01, and 02}})$ to the Operand State register.

OSA

$(OSR) \rightarrow A_{00\text{-}02}$

23 18	17 12	11 00
77	67	0 0 0 0

Instruction Description: Transfer (OSR) to $A_{\text{(bits 00, 01, and 02)}}$.

ACI
$(A_{00\text{-}02}) \rightarrow CIR$

23 18	17 12	11 00
77	54	▨

Instruction Description: Transfer ($A_{\text{bits 00, 01, and 02}}$) to the Channel Index register.

Comments: Bits 00 through 11 should be loaded with zeros. This instruction is used to set the CIR to a value from 0 to 7. When referenced, (CIR) are logically OR'ed with the channel designator, ch, in the following instructions:

73 through 76	I/O instructions
77.0	Connect
77.1	Select Function
77.2	Sense External Status
77.2	Copy External Status
77.3	Sense Internal Status
77.3	Copy Internal Status
77.4	Sense Interrupt

CIA
$(CIR) \rightarrow A_{00\text{-}02}$

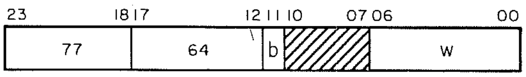
23 18	17 12	11 00
77	55	▨

Instruction Description: Transfer (CIR) to $A_{(\text{bits 00-02})}$.

Comments: Bits 00 through 11 should be loaded with zeros.

APF

$(A_{00\text{-}11}) \rightarrow PIF$



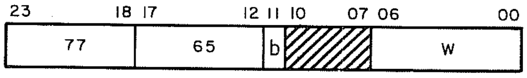
b = index designator, B^2 only

Instruction Description: Transfer ($A_{\text{bits 00 through 11}}$) to the 12-bit index at Page Index File address 'w'.

Comments: If bit 11 is a "1", (B^2) are used for address modification. Bits 07 through 10 should be loaded with zeros. This instruction is a no-operation instruction if the 3311 Multiprogramming option is not present in the system.

PFA

$(PIF) \rightarrow A_{00\text{-}11}$



b = index designator, B^2 only

Instruction Description: Transfer the 12-bit index at Page Index File address 'w' to $A_{\text{(bits 00 through 11)}}$.

Comments: If bit 11 is a "1", (B^2) are used for address modification. Bits 07 through 10 should be loaded with zeros. This instruction is a no-operation instruction if the 3311 Multiprogramming option is not present in the system.

CRA

(CR) → A_{00-05}

23 18	17 12	11 00
77	63	0000

Instruction Description: Transfer the (CR) to the lower 6 bits of A. The Condition register is cleared upon completion of the transfer.

Comments: This instruction should be used in interrupt routines during Executive Mode operations. All interrupts are disabled between interrupt recognition and the execution of the instruction. Program conditions represented by the Condition register are listed below:

Bit 00	-	Boundary Jump
Bit 01	-	Destructive Load A
Bit 02	-	Operand Relocation Using OSR
Bit 03	-	Program State Jump
Bit 04	-	Interrupt System Enabled
Bit 05	-	Program State

ACR

(A_{00-05}) → CR

23 18	17 12	11 09	08 00
77	63	4	000

Instruction Description: Transfer the (A_{00-05}) to the Condition register.

Comments: This instruction should be used at the end of interrupt routines during Executive mode to restore the Condition register to its original state. All interrupts are disabled between the execution of the ACR instruction and the jump instruction used to exit from interrupt routines. Refer to the CRA instruction for the conditions represented in the Condition register.

JAA

Jump Address → A

23 18	17 12	11 00
77	56	▨

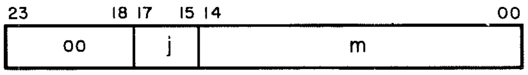
Instruction Description: Transfer the address, P, of the last jump type of instruction occurring in Program State, to $A_{(\text{bits 00 through 14})}$.

Comments: The LJA (Last Jump Address) can also be displayed on the console when the LJA switch is depressed and the computer is stopped (refer to Section 7). Bits 00 through 11 should be loaded with zeros.

Jump Instructions

Operation Field		Address Field	Interpretation
SJ1-6	00	m	Jump if appropriate key (1-6) is set
UJP, I	01	m, b	Unconditional jump
JMP, HI	70	m	Jump on Positive result
JMP, ZRO	70	m	Jump on Zero result
JMP, LOW	70	m	Jump on Negative result
IJI	02	m, b	Index jump; Incremental index
IJD	02	m, b	Index jump; Decremental index
AZJ	03	m	Compare A with zero for Jump condition
AQJ	03	m	Compare A with Q for Jump condition
RTJ	00	m	Return jump

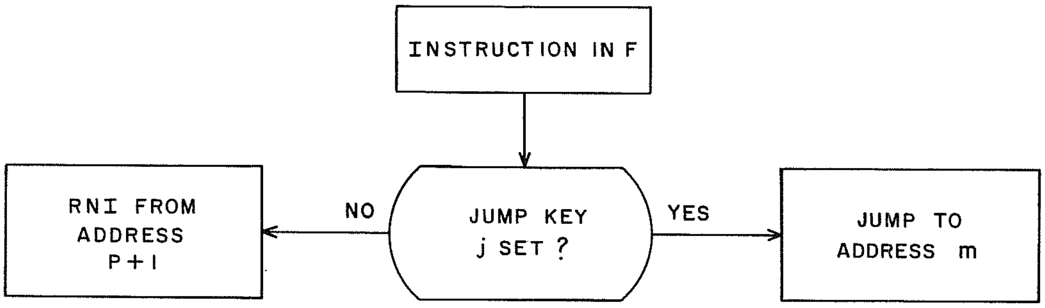
SJ1-6
Selective Jump



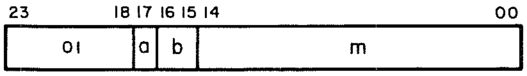
j = jump keys 1 to 6
m = jump address

Instruction Description: Jump to address m if Jump key j is set; otherwise, RNI from address P+1.

Comments: Indirect addressing and address modification may not be used.



UJP
Unconditional Jump



a = addressing mode designator
b = index register designator
m = storage address; $M = m + (B^b)$

Instruction Description: Unconditionally jump to address M.

Comments: Indirect addressing and indexing may be used.

JMP, ZRO
Jump if result = 0

23 18	17 15	14 00
70	1	m

m = jump address

Instruction Description: Sense the status of the BCR (BDP Condition Register). If the result from the preceding BDP operation was zero, jump to address m.

Comments: If the console BDP switch is not active or if the BDP is not present in the system, this instruction is trapped. (Refer to Section 4.)

JMP, HI
Jump if result is + or high

23 18	17 15	14 00
70	0	m

m = jump address

Instruction Description: Sense the status of the BCR (BDP Condition Register). If the result from the preceding BDP operation was positive or greater than zero, jump to address m.

Comments: If the console BDP switch is not active or if the BDP is not present in the system, this instruction is trapped. (Refer to Section 4.)

JMP, LOW
Jump if result is — or low

23 18	17 15	14 00
70	2	m

m = jump address

Instruction Description: Sense the status of the BCR (BDP Condition Register). If the result from the preceding BDP operation was negative or less than zero, jump to address m.

Comments: If the BDP is not in the system, or the console BDP switch is not active, this instruction is trapped. (Refer to Section 4.)

IJI
Index Jump, Incremental

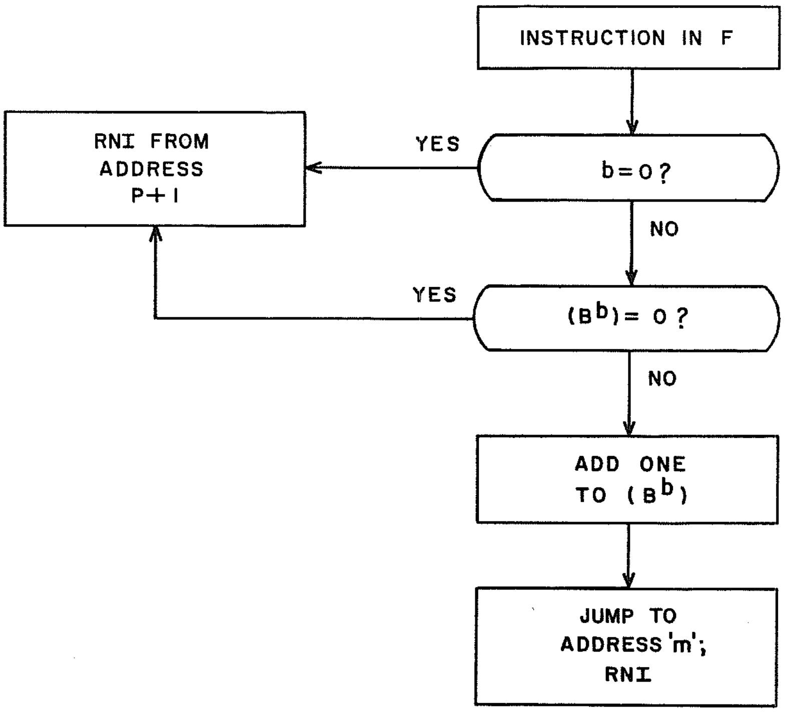
23 18	17	16 15	14 00
02	0	b	m

b = index register designator
m = jump address

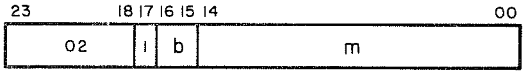
Instruction Description: If b = 1, 2, or 3, the respective Index register is examined:

1. If (B^b) = 00000, the jump test condition is not satisfied; RNI from address P+1.
2. If (B^b) ≠ 00000, the jump test condition is satisfied. One is added to (B^b); jump to address m and RNI.

Comments: If b = 0, this is a No-Operation instruction; RNI from address P+1. Indirect addressing and jump address modification may not be used. The counting operation is done in a one's complement additive accumulator. Negative zero (77777) is not generated because the count progresses from: 77775, 77776, to 00000 (positive zero) and stops. If negative zero is initially loaded into B^b, the count progresses: 77777, 00001, 00002, etc. In this case, the counter must increment through the entire range of numbers to each positive zero.



IJD
Index Jump, Decremental

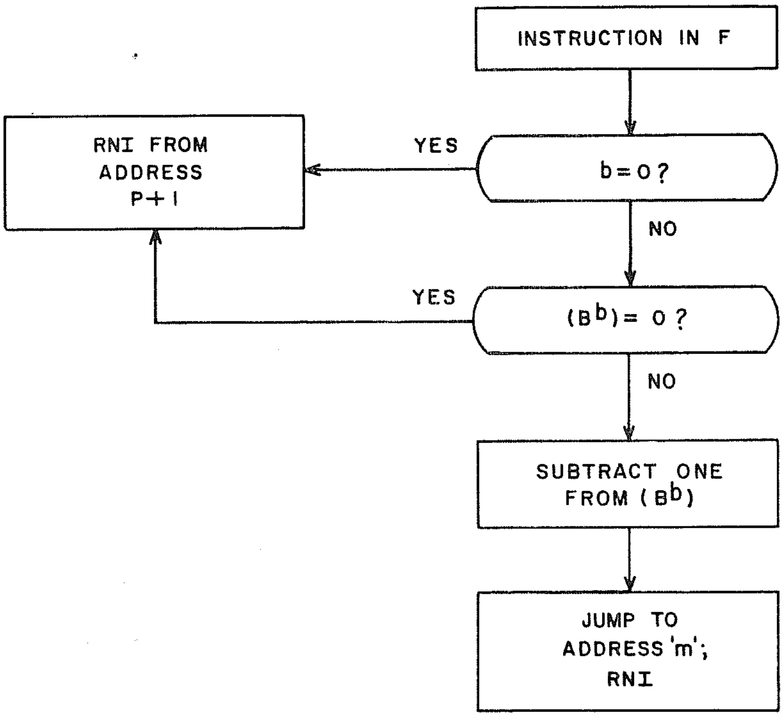


b = index register designator
m = jump address

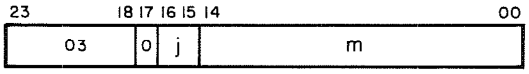
Instruction Description: If b = 1, 2,or 3, the respective Index register is examined:

1. If (B^b) = 00000, the Jump Test condition is not satisfied; RNI from address P+1.
2. If $(B^b) \neq$ 00000, the Jump Test condition is satisfied. One is subtracted from (B^b); jump to address m and RNI.

Comments: If b = 0, this is a no-Operation instruction; RNI from address P+1. Indirect addressing and jump address modification may not be used. If negative zero (77777) is initially loaded into B^b, the count decrements through the entire range of numbers to reach 00000 before the program will RNI from P+1.



AZJ
Condition Compare A with Zero, Jump

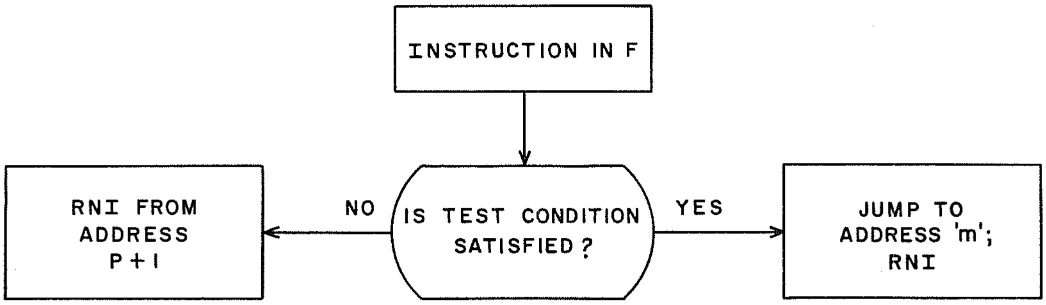


j = jump designator (0-3)
m = jump address

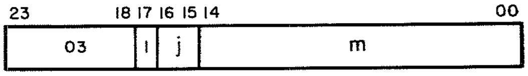
Instruction Description: The operand in A is algebraically compared with zero for an equality, inequality, greater-than or less-than condition (see table). If the test condition is satisfied, program execution jumps to address m. If the test condition is not satisfied, RNI from address P+1.

Comments: Positive zero (00000000) and negative zero (77777777) give identical results when j = 0 or 1. When j = 2 or 3, negative zero is recognized as less than positive zero. Indirect addressing and address modification may not be used.

Condition Mnemonic	Jump Designator j	Test Condition
EQ	0	(A) = 0
NE	1	(A) ≠ 0
GE	2	(A) ≥ 0
LT	3	(A) < 0



AQJ
Condition Compare A with Q, Jump

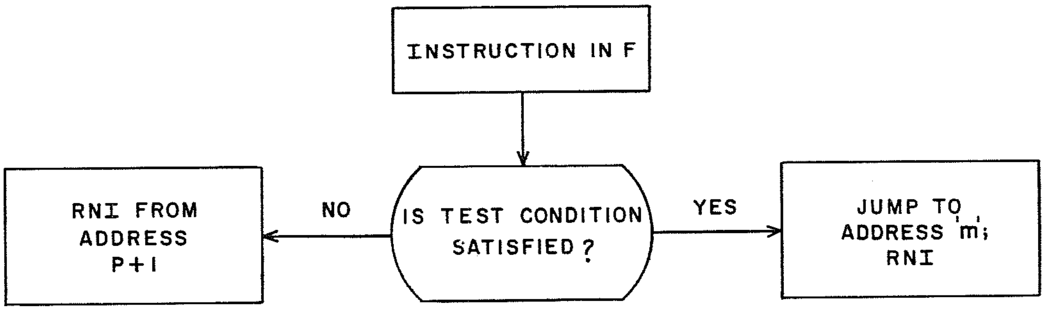


j = 0-3 jump designator (0-3)
m = jump address

Instruction Description: The quantity in A is algebraically compared with the quantity in Q for equality, inequality, greater-than or less-than condition (see table). If the test condition is satisfied, program execution jumps to address m. If the test condition is not satisfied, RNI from address P+1.

Comments: This instruction may be used to test (Q) by placing an arbitrary value in A for the comparison. Positive and negative zero give identical results in this test when j = 0 or 1. When j = 2 or 3, negative zero is recognized as less than positive zero. Indirect addressing and address modification may not be used.

Condition Mnemonic	Jump Designator j	Test Condition
EQ	0	(A) = (Q)
NE	1	(A) ≠ (Q)
GE	2	(A) ≥ (Q)
LT	3	(A) < (Q)



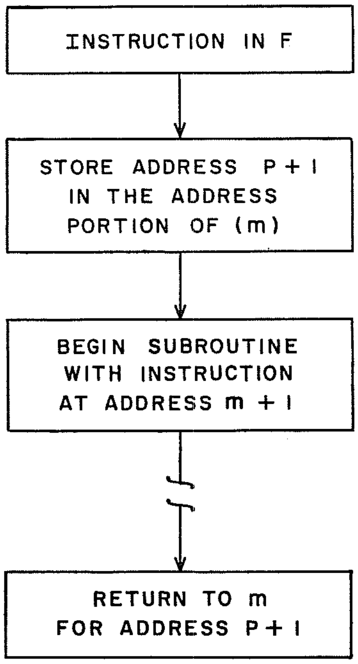
RTJ
Return Jump

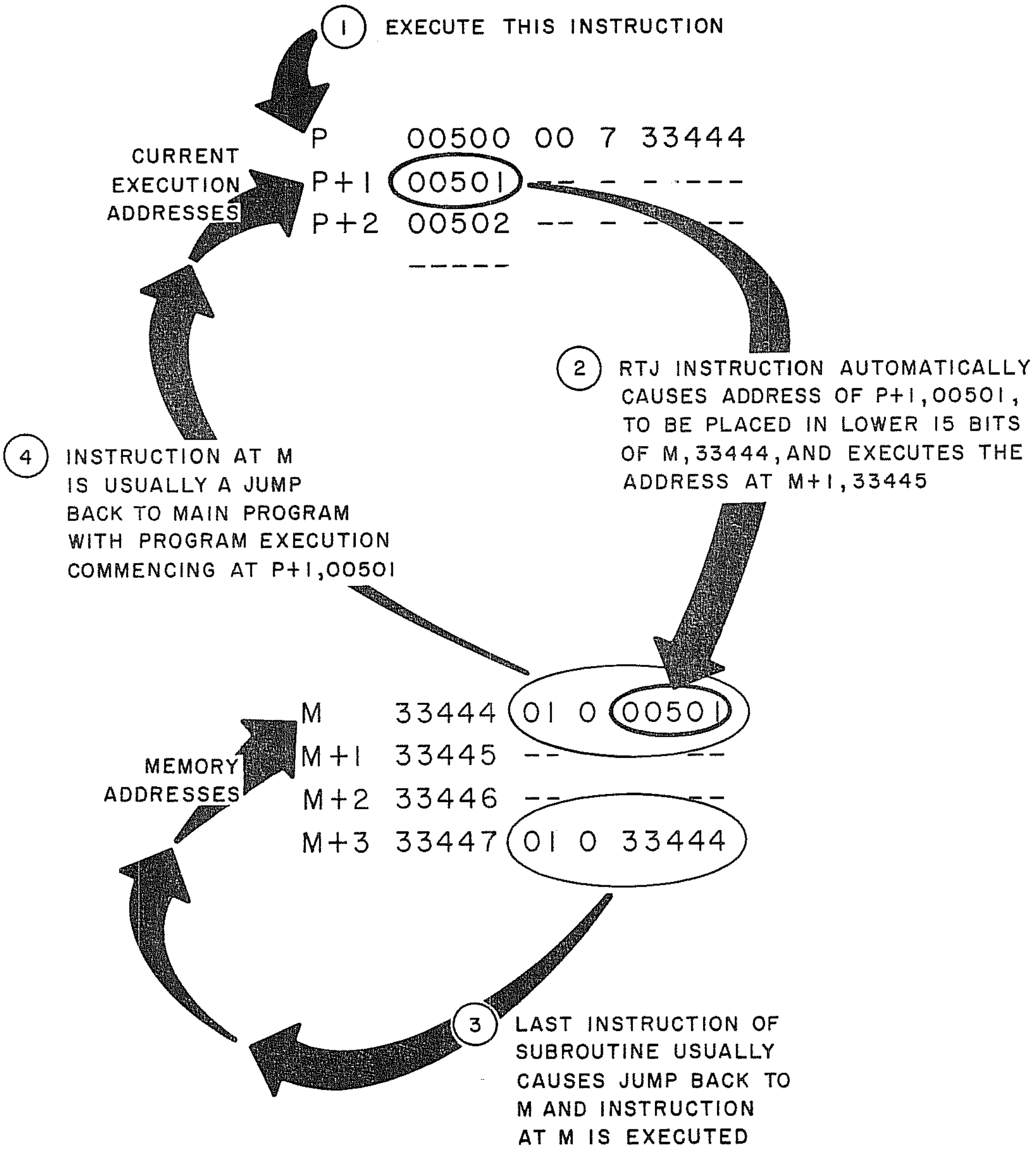
23 18	17 15	14 00
00	7	m

m = jump address

Instruction Description: The address portion of m is replaced with the return address, P+1. Jump to location m+1 and begin executing instructions at that location.

Comments: This instruction should not be used to transfer control from Monitor State to Program State. If an RTJ instruction is executed and the Boundary Jump flag is set (refer to SBJP instruction), the STO cycle is executed in Monitor State, i.e., address 'P' is stored at address 'm' of the monitor program. Indirect addressing and address modification may not be used. An example of an executed RTJ instruction is illustrated on the following page.


INSTRUCTION IN F
STORE ADDRESS P + I IN THE ADDRESS PORTION OF (m)
BEGIN SUBROUTINE WITH INSTRUCTION AT ADDRESS m + I
RETURN TO m FOR ADDRESS P + I

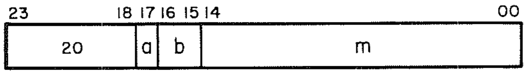

1 EXECUTE THIS INSTRUCTION
CURRENT EXECUTION ADDRESSES
P 00500 00 7 33444
P+1 00501 -- - - ----
P+2 00502 -- - - --
-----
2 RTJ INSTRUCTION AUTOMATICALLY CAUSES ADDRESS OF P+1,00501, TO BE PLACED IN LOWER 15 BITS OF M,33444,AND EXECUTES THE ADDRESS AT M+1,33445
4 INSTRUCTION AT M IS USUALLY A JUMP BACK TO MAIN PROGRAM WITH PROGRAM EXECUTION COMMENCING AT P+1,00501
MEMORY ADDRESSES
M 33444 01 0 00501
M+1 33445 -- --
M+2 33446 -- --
M+3 33447 01 0 33444
3 LAST INSTRUCTION OF SUBROUTINE USUALLY CAUSES JUMP BACK TO M AND INSTRUCTION AT M IS EXECUTED

RTJ EXECUTION EXAMPLE

Load Instructions

Operation Field		Address Field	Interpretation
LDA, I	20	m, b	Load A
LACH	22	r, B^1	Load A, Character
LCA, I	24	m, b	Load A, Complement
LDL, I	27	m, b	Load A, Logical
LDAQ, I	25	m, b	Load AQ
LCAQ, I	26	m, b	Load AQ, Complement
LDQ, I	21	m, b	Load Q
LQCH	23	r, B^2	Load Q, Character
LDI, I	54	m, b	Load Index

LDA
Load A

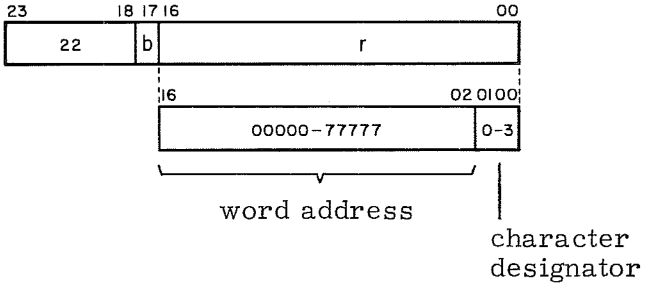


a = addressing mode designator
b = index register designator
m = storage address; $M = m + (B^b)$

Instruction Description: Load A with a 24-bit quantity from the storage address specified by M.

Comments: Indirect addressing and address modification may be used.

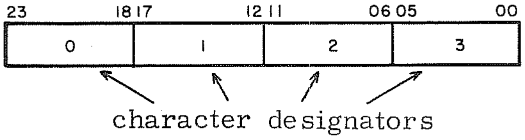
LACH
Load A, Character



If b = 1, r is modified by index register B^1; $R = r + (B^1)$.
If b = 0, r is not modified (r = R).

Instruction Description: Load bits 00 through 05 of A with the character from storage specified by character address R. The A register is cleared prior to the Load operation.

Comments: Indirect addressing may not be used. Characters are specified in storage as follows:



NOTE

Since the sign of B^b is extended during character address modification, it is possible to only reference within ± $16,383_{10}$ characters.

LCA
Load A, Complement

23 18	17	16 15	14 00
24	a	b	m

a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Load A with the ones complement of a 24-bit quantity from storage address M.

Comments: Indirect addressing and address modification may be used.

LDL
Load A, Logical

23 18	17	16 15	14 00
27	a	b	m

a = addressing mode designator
b = index register designator
m = storage address; m=m+(B^b)

Instruction Description: Load A with the logical product (the AND function) of (Q) and the 24-bit quantity from storage address M.

LDAQ
Load AQ

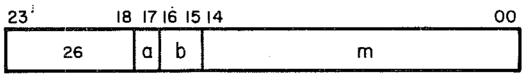
23 18	17	16 15	14 00
25	a	b	m

a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Load the A and Q registers with the 24-bit quantities from addresses M and M+1, respectively.

Comments: Addresses 77776 and 77777 should be used only if it is desirable to have M and M+1 as non-consecutive addresses, since one's complement arithmetic is used to form M+1.

LCAQ
Load AQ, Complement

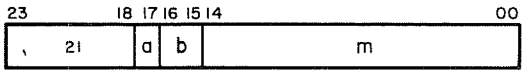


a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Load registers A and Q with the complement of the 24-bit quantities from addresses M and M+1, respectively.

Comments: Addresses 77776 and 77777 should be used only if it is desirable to have M and M+1 as non-consecutive addresses, since one's complement arithmetic is used to form M+1.

LDQ
Load Q

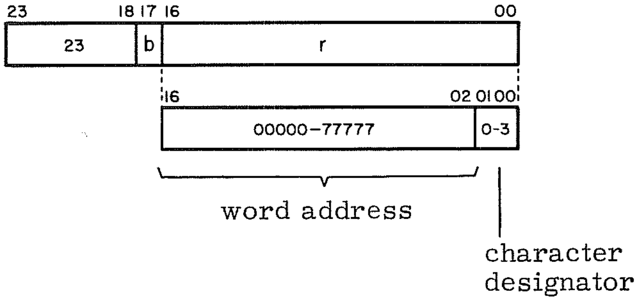


a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Load Q with a 24-bit quantity from storage address M.

Comments: Indirect addressing and address modification may be used.

LQCH
Load Q, Character



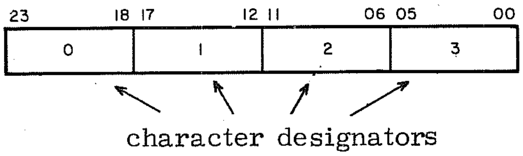
If b = 1, r is modified by
index register B^2; R = r + (B^2).
If b = 0, r is not modified (r = R).

NOTE

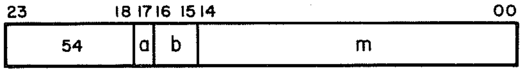
Since the sign of B^b is extended during character address modification, it is possible to only reference within ± $16,383_{10}$ characters.

Instruction Description: Load bits 00 through 05 of Q with the character from storage specified by character address R. The Q register is cleared prior to the load operation.

Comments: Indirect addressing may not be used. Characters are specified in storage as follows:



LDI
Load Index



a = addressing mode designator
b = index register designator
m = storage address (indexing not permitted)

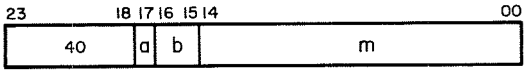
Instruction Description: Load the specified index register, B^b, with the lower 15 bits of the operand stored at address m.

Comments: Indirect addressing may be used, but address modification (indexing) is not possible anywhere within the indirect address. If indexing in the indirect address is specified, it is totally ignored.

Store Instructions

Operation Field		Address Field	Interpretation
STA, I	40	m, b	Store A
SACH	42	r, B^2	Store A, character
STAQ, I	45	m, b	Store AQ
STQ, I	41	m, b	Store Q
SQCH	43	r, B^1	Store Q, character
STI, I	47	m, b	Store index
SWA, I	44	m, b	Store 15-bit word address
SCHA	46	m, b	Store 17-bit character address

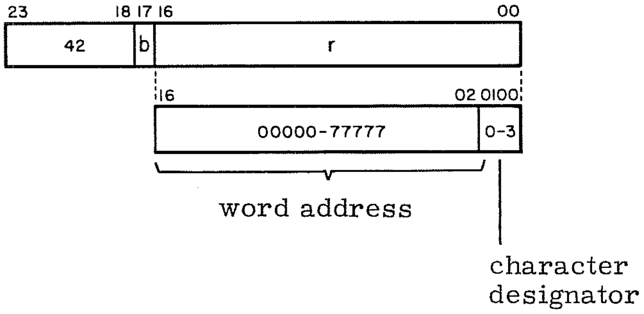
STA
Store A



a = addressing mode designator
b = index register designator
m = storage address; $M = m + (B^b)$

Instruction Description: Store (A) at the storage address specified by M. The (A) remain unchanged.

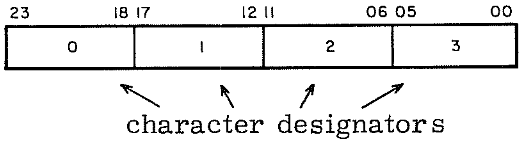
SACH
Store A, Character



If b = 1, r is modified by index register B^2; $R = r + (B^2)$.
If b = 0, r is not modified (r = R).

Instruction Description: Store the contents of bits 00 through 05 of the A register in the specified character address. All of (A) and the remaining three characters in storage remain unchanged.

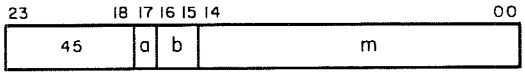
Comments: Indirect addressing may not be used. Characters are specified in storage as follows:



NOTE

Since the sign of B^b is extended during character address modification, it is possible to only reference within ± $16,383_{10}$ characters.

STAQ
Store AQ

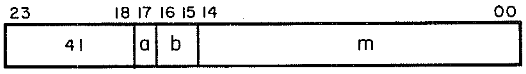


a = addressing mode designator
b = index register designator
m = storage address; $M = m + (B^b)$

Instruction Description: Store (A) and (Q) in the storage locations specified by address M and M+1, respectively. The (A) and (Q) remain unchanged.

Comments: Address 77776 and 77777 should be used only if it is desirable to have M and M+1 as non-consecutive addresses, since one's complement arithmetic is used to form M+1.

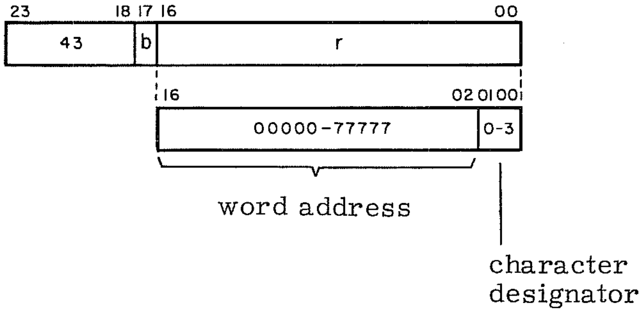
STQ
Store Q



a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Store (Q) at the storage address specified by M. The (Q) remain unchanged.

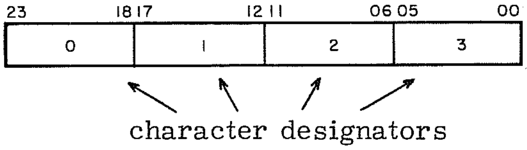
SQCH
Store Q Character



If b = 1, r is modified by
index register B^1; R = r + (B^1).
If b = 0, r is not modified. (r = R).

Instruction Description: Store the contents of bits 0 through 5 of the Q register in the specified character address. All of (Q) and the remaining three characters in storage remain unchanged.

Comments: Indirect addressing may not be used. Characters are specified in storage as follows:



NOTE

Since the sign of B^b is extended during character address modification, it is possible to reference only within ± $16,383_{10}$ characters.

STI
Store Index

23 18	17	16 15	14 00
47	a	b	m

a = addressing mode designator
b = index register designator
m = storage address (indexing not permitted)

Instruction Description: Store the contents of the specified index register, B^b, in the lower 15 bits of storage address m. The upper 9 bits of m and all of (B^b) remain unchanged.

Comments: Indirect addressing may be used, but address modification (indexing) is not possible anywhere within the indirect address. If indexing in the indirect address is specified, it is totally ignored.

SWA
Store Word Address

23 18	17	16 15	14 00
44	a	b	m

a = addressing mode designator
b = index register designator
m = storage address; $(M = m + (B^b)$

Instruction Description: Store the lower 15 bits of (A) in the designated address M. The upper 9 bits of M and all of (A) remain unchanged.

SCHA
Store
Character Address

23 18	17	16 15	14 00
46	a	b	m

a = addressing mode designator
b = index register designator
m = storage address; $M = m + (B^b)$

Instruction Description: Store the lower 17 bits of (A) in the address designated by M. The upper 7 bits of M and all of (A) remain unchanged.

Shift and Scale Instructions

Operation Field		Address Field	Interpretation
SSH	10	m	Storage shift
SHA	12	k, b	Shift A
SHQ	12	k, b	Shift Q
SHAQ	13	k, b	Shift AQ
SCAQ	13	k, b	Scale AQ

SSH
Storage Shift

23 18	17 15	14 00
10	0	m

m = storage address

Instruction Description: Sense bit 23 of the quantity stored at address m. If bit 23 = "0" (positive), RNI from P + 1; if negative ("1"), RNI from P + 2. Shift (m) one place left, end around, and replace it in this same storage location.

Comments: Address modification may not be used.

SHA
Shift A

23 18	17	16 15	14 00
12	0	b	k

b = index register designator
k = unmodified shift count; $K = k + (B^b)$

Instruction Description: (B^b) and k, with their signs extended, are added. If b = 0, the sign of k is still extended. The sign and magnitude of the 24-bit sum determine the direction and magnitude of the shift. The computer senses only bits 00-05 and 23 of the sum for this information. For left shifts, the shift magnitude is the lower 6 bits of K ; for right shifts, the complement of the lower 6 bits of K equals the shift magnitude.

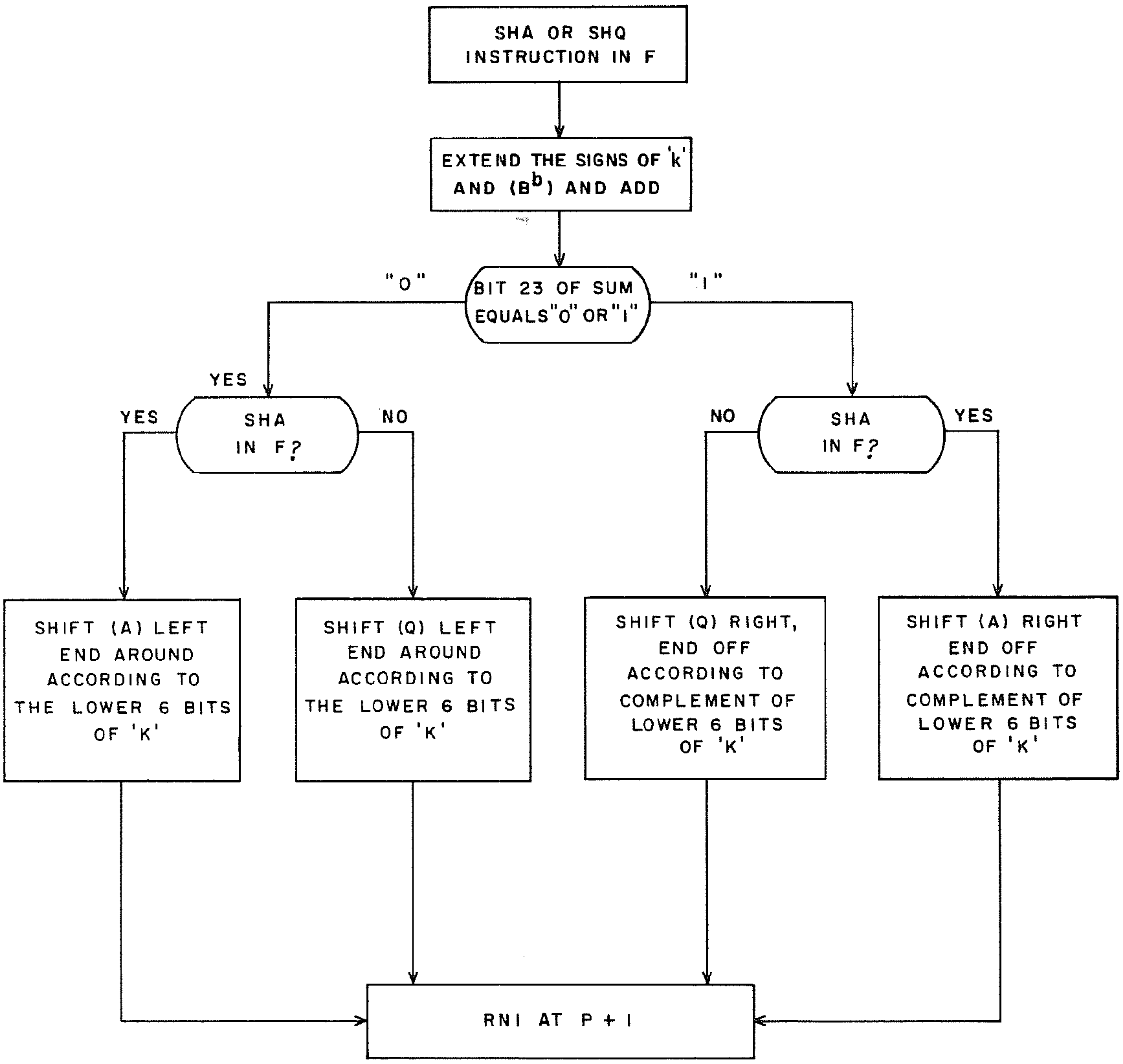
Examples:

Shift left six positions:	K = 00000006
Shift right six positions:	K = 77777771

Comments: During left shifts, bits reaching the upper bit position of the A (during SHA) or Q (during SHQ) registers are carried end-around. Therefore, a left shift of 24 places results in no change in (A) or (Q). A left shift that exceeds 24 places results in an effective shift of K-24 (or K-48) places.

During right shifts, the sign bit is extended and the bits are shifted end-off. A right shift of 23 or more places results in (A) or (Q) becoming all "0's" or all "1's", depending upon the original sign.

SHA/SHQ FLOW CHART


SHA OR SHQ INSTRUCTION IN F
EXTEND THE SIGNS OF 'k' AND (B^b) AND ADD
"0"
BIT 23 OF SUM EQUALS "0" OR "1"
"1"
YES
YES
SHA IN F?
NO
NO
SHA IN F?
YES
SHIFT (A) LEFT END AROUND ACCORDING TO THE LOWER 6 BITS OF 'K'
SHIFT (Q) LEFT END AROUND ACCORDING TO THE LOWER 6 BITS OF 'K'
SHIFT (Q) RIGHT, END OFF ACCORDING TO COMPLEMENT OF LOWER 6 BITS OF 'K'
SHIFT (A) RIGHT END OFF ACCORDING TO COMPLEMENT OF LOWER 6 BITS OF 'K'
RNI AT P + 1

SHQ
Shift Q

23 18	17	16 15	14 00
12	1	b	k

b = index register designator
k = shift count: K = k + (B^b)

Instruction Description: Shift (Q). Refer to SHA description.

SHAQ
Shift AQ

23 18	17	16 15	14 00
13	0	b	k

b = index register designator
k = unmodified shift count; K = k+(B^b)

Instruction Description: Shift (AQ). (B^b) and k, with their signs extended, are added. If b = 0, the sign of k is still extended. The sign and magnitude of the 24-bit sum determine the direction and magnitude of the shift. The computer senses only bits 00-05 and 23 of the sum for this information. For left shifts, the shift magnitude is the lower 6 bits of K; for right shifts, the complement of the lower 6 bits of K equals the shift magnitude.

Examples:

Shift left three places:	K = 00000003
Shift right three places:	K = 77777774

Comments: During left shifts, bits reaching the upper bit position of the A register are carried end-around to the lowest bit position of Q. Therefore, a left shift of 48 places results in no change in (AQ). A left shift exceeding 48 places results in an effective shift of K-48 places. During right shifts, the sign bit is extended and the bits are shifted end-off. A right shift of 47 or more places results in (AQ) becoming all "0's" or all "1's", depending upon the original sign.

SCAQ
Scale AQ

23 18	17	16 15	14 00
13	1	b	k

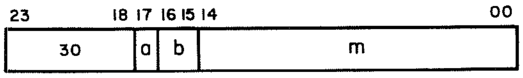
b = index register designator
k = scale factor
K = residue = k minus the number of shifts.
If b = 1,2, or 3, then K →B^b

Instruction Description: (AQ) are shifted left, end-around, until the 2 highest order bits (46 and 47) are unequal. If (AQ) should initially equal positive or negative zero, 48_{10} shifts are executed before the instruction terminates. During scaling, the computer counts the number of shifts. A quantity K, called the residue, is equal to the scale factor 'k' minus the number of shifts made. If b = 0, this quantity is discarded; if b = 1, 2, or 3, the residue is transferred to the designated index register.

Arithmetic Instructions

Operation Field		Address Field	Interpretation
ADA, I	30	m, b	Add to A
RAD, I	34	m, b	Replace add
SBA, I	31	m, b	Subtract from A
ADAQ, I	32	m, b	Add to AQ
SBAQ, I	33	m, b	Subtract from AQ
MUA, I	50	m, b	Multiply A
DVA, I	51	m, b	Divide A
MUAQ, I	56	m, b	Multiply AQ
DVAQ, I	57	m, b	Divide AQ
FAD, I	60	m, b	FP addition to AQ
FSB, I	61	m, b	FP subtraction from AQ
FMU, I	62	m, b	FP multiplication of AQ
FDV, I	63	m, b	FP division of AQ

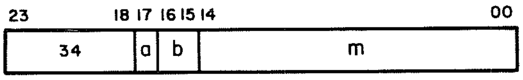
ADA
Add to A



a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Add the 24-bit operand located at address M to (A). The sum replaces the original (A).

RAD
Replace Add



a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Replace the 24-bit operand at address M with the sum of (M) and (A). The original (A) remain unchanged.

SBA
Subtract from A

23 18	17	16 15	14 00
31	a	b	m

a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Subtract the 24-bit operand located at address M from (A). The difference replaces the original (A).

ADAQ
Add to AQ

23 18	17	16 15	14 00
32	a	b	m

a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Add the 48-bit operand located in addresses M and M+1 to (AQ). The sum is displayed in AQ.

Comments: The upper 24 bits of the 48-bit operand in memory are contained at address M.

SBAQ
Subtract from AQ

23 18	17	16 15	14 00
33	a	b	m

a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Subtract the 48-bit operand located in addresses M and M+1 from (AQ). The difference is displayed in AQ.

MUA
Multiply A

23 18	17	16 15	14 00
50	a	b	m

a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Multiply (A) by the operand located at address M.

NOTE

The 48-bit product is displayed in QA. The higher order bits are in Q and the lower order bits are in A.

DVA
Divide A

23 18	17	16 15	14 00
51	a	b	m

a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Divide the 48-bit operand in AQ by the operand at storage address M. The quotient is displayed in A and the remainder with sign extended is displayed in Q. If a divide fault occurs, the operation halts and program execution advances to the next address. The final (A) and (Q) are meaningless if a divide fault occurs.

MUAQ
Multiply AQ

23 18	17	16 15	14 00
56	a	b	m

a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Multiply (AQ) by the 48-bit operand in addresses M and M+1. The 96-bit product is displayed in AQE.

Comments: Refer to Figure 5-5 for operand formats.

DVAQ
Divide AQ

23 18	17	16 15	14 00
57	a	b	m

a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Divide (AQE) by the 48-bit operand in addresses M and M+1. The quotient is displayed in AQ, and the remainder with its sign extended is displayed in E.

Comments: If a divide fault occurs, program execution advances to the next address. The final contents of AQ and E are meaningless if a divide fault occurs. Refer to Figure 5-5 for operand formats.

NOTE

Figure 5-5 illustrates operand format and bit allocations for all floating point instructions. Refer to the Floating Point section of Appendix B for additional floating point considerations and examples.

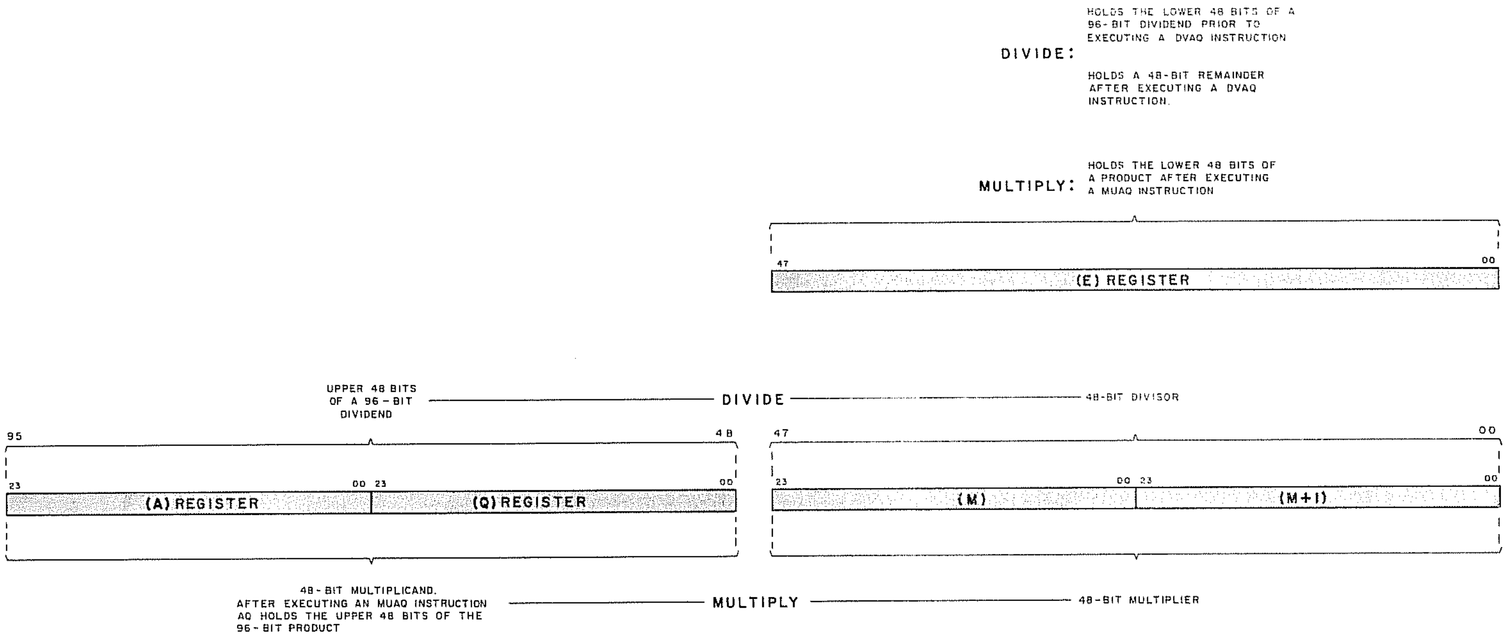
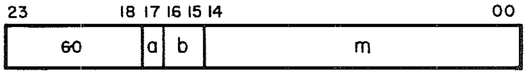

DIVIDE:
HOLDS THE LOWER 48 BITS OF A 96-BIT DIVIDEND PRIOR TO EXECUTING A DVAQ INSTRUCTION
HOLDS A 48-BIT REMAINDER AFTER EXECUTING A DVAQ INSTRUCTION.
MULTIPLY:
HOLDS THE LOWER 48 BITS OF A PRODUCT AFTER EXECUTING A MUAQ INSTRUCTION
47
00
(E) REGISTER
UPPER 48 BITS OF A 96-BIT DIVIDEND
DIVIDE
48-BIT DIVISOR
95
48
47
00
23
00 23
00
(A) REGISTER
(Q) REGISTER
23
00 23
00
(M)
(M+1)
48-BIT MULTIPLICAND. AFTER EXECUTING AN MUAQ INSTRUCTION AQ HOLDS THE UPPER 48 BITS OF THE 96-BIT PRODUCT
MULTIPLY
48-BIT MULTIPLIER

Figure 5-5. Operand Formats and Bit Allocations for MUAQ and DVAQ Instructions

FAD
FP Addition to AQ



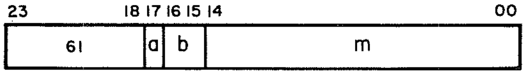
a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Add the 48-bit operand located in addresses M and M+1 to (AQ). The rounded and normalized sum is displayed in AQ.

Comments: The higher order bits of E hold the portion of the operand that was shifted out of AQ during exponent equalization.

Refer to Figure 5-6 for operand formats.

FSB
FP Subtraction from AQ

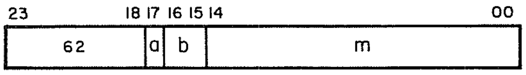


a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Subtract the 48-bit floating point operand located at storage addresses M and M+1 from the floating point operand in AQ. The rounded and normalized difference is displayed in AQ.

Comments: The upper order bits of E hold the portion of the operand that was shifted out of AQ during the equalization of exponents. Refer to Figure 5-6 for operand formats.

FMU
FP Multiplication of AQ

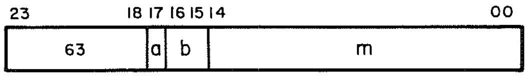


a = addressing mode designator
b = index register designator
m = storage address; $M = m + (B^b)$

Instruction Description: Multiply the 48-bit floating point operand in AQ by the floating point operand located at storage addresses M and M+1. The rounded and normalized product is displayed in AQ.

Comments: Bits 12-47 of E hold the lower 36 bits of the 72-bit unnormalized product. Refer to Figure 5-6 for operand formats.

FDV
FP Division of AQ



a = addressing mode designator
b = index register designator
m = storage address; $M = m + (B^b)$

Instruction Description: Divide the floating point operand in AQ by the 48-bit floating point operand located at storage addresses M and M+1. The rounded and normalized quotient is displayed in AQ. The remainder with sign extended appears in the E register.

Comments: The sign of the remainder is the same as that of the dividend. Refer to Figure 5-6 for operand formats.

NOTE

The divisor must be properly normalized or a divide fault will result. Refer to Interrupt conditions, Section 4.

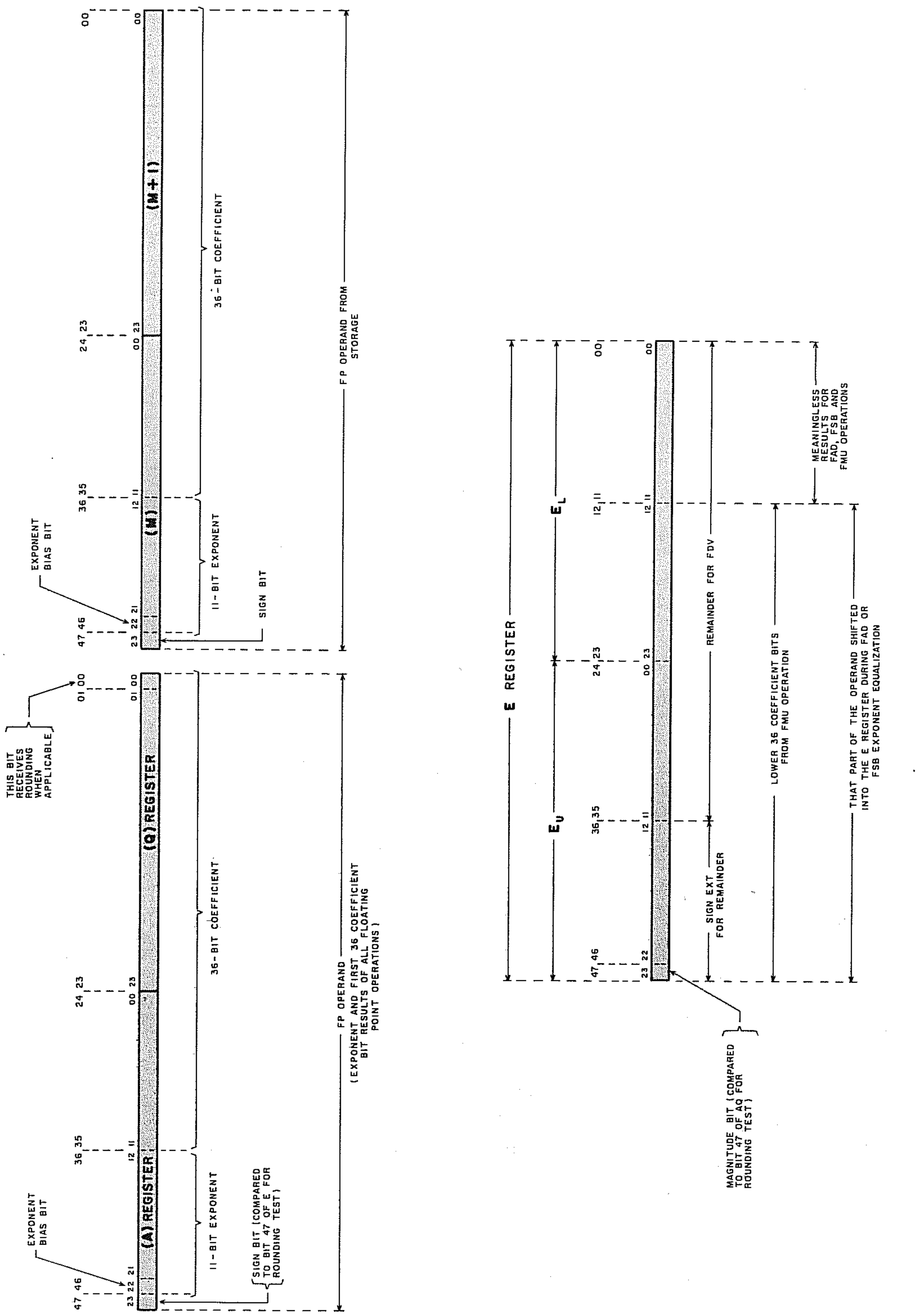

(A) REGISTER
(Q) REGISTER
(M)
(M+1)
E REGISTER
E_U
E_L
47 46
36 35
24 23
01 00
12 11
00 23
23 22 21
EXPONENT BIAS BIT
11-BIT EXPONENT
36-BIT COEFFICIENT
SIGN BIT (COMPARED TO BIT 47 OF E FOR ROUNDING TEST)
THIS BIT RECEIVES ROUNDING WHEN APPLICABLE
FP OPERAND (EXPONENT AND FIRST 36 COEFFICIENT BIT RESULTS OF ALL FLOATING POINT OPERATIONS)
SIGN BIT
FP OPERAND FROM STORAGE
MEANINGLESS RESULTS FOR FAD, FSB AND FMU OPERATIONS
REMAINDER FOR FDV
SIGN EXT FOR REMAINDER
LOWER 36 COEFFICIENT BITS FROM FMU OPERATION
THAT PART OF THE OPERAND SHIFTED INTO THE E REGISTER DURING FAD OR FSB EXPONENT EQUALIZATION
MAGNITUDE BIT (COMPARED TO BIT 47 OF AQ FOR ROUNDING TEST)

Figure 5-6. Operand Formats and Bit Allocations for Floating Point Arithmetic Instructions

Logical Instructions

Operation Field		Address Field	Interpretation
XOI	16	y, b	Exclusive OR of index and y
XOA	16	y	Exclusive OR of A and y
XOA, S	16	y	Exclusive OR of A and y, sign of y extended
XOQ	16	y	Exclusive OR of Q and y
XOQ, S	16	y	Exclusive OR of Q and y, sign of y extended
ANI	17	y, b	AND of index and y
ANA	17	y	AND of A and y
ANA, S	17	y	AND of A and y, sign of y extended
ANQ	17	y	AND of Q and y
ANQ, S	17	y	AND of Q and y, sign of y extended
SSA, I	35	m, b	Selectively set A
SCA, I	36	m, b	Selectively complement A
LPA, I	37	m, b	Logical product A

NOTE

The LDL (Load A, Logical) instruction may be found in the LOAD INSTRUCTIONS subsection.

The following two examples use logical instructions and illustrate the Exclusive OR and AND functions:

```
EXAMPLE A:                          (Binary Equivalents)

    (A) = 23456701        =    010 011 100 101 110 111 000 001
    Execute: 16 4 50321   =    111 111 111 101 000 011 010 001
                               -------------------------------
    (XOA, S)                   101 100 011 000 110 100 010 000

                Final (A) =     5   4   3   0   6   4   2   0

EXAMPLE B:

    (Q) = 23456701        =    010 011 100 101 110 111 000 001
    Execute: 17 7 77170   =    000 000 000 111 111 001 111 000
                               -------------------------------
    (ANQ)                      000 000 000 101 110 001 000 000

                Final (A) =     0   0   0   5   6   1   0   0
```

XOI
Exclusive OR of B^b and y

23 18	17	16 15	14 00
16	0	b	y

b = index register designator

Instruction Description: Enter the selective complement (the Exclusive OR function) of y and (B^b) back into the same index register.

Comments: If b = 0, this is a no-operation instruction.

XOA
Exclusive OR of A and y

23 18	17 15	14 00
16	6	y

Instruction Description: Enter the selective complement (the Exclusive OR function) of y and (A) back into the A register.

XOA, S
Exclusive OR of A and y Sign Extended

23 18	17 15	14 00
16	4	y

Instruction Description: Same as XOA except the sign of y is extended.

XOQ
Exclusive OR of Q and y

23 18	17 15	14 00
16	7	y

Instruction Description: Enter the selective complement (the Exclusive OR function) of y and (Q) back into the Q register.

XOQ, S
Exclusive OR of Q and y Sign Extended

23 18	17 15	14 00
16	5	y

Instruction Description: Same as XOQ except the sign of y is extended.

SSA
Selectively Set A

23 ... 18	17	16 15	14 ... 00
35	a	b	m

a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Selectively set the bits in the A register to "1's" for all corresponding "1's" in the quantity at address M. Initial "1's" in A remain unchanged.

```
EXAMPLE:  (A) = 23456710 = 010 011 100 101 110 111 001 000
          (M) = 76345242 = 111 110 011 100 101 010 100 010
                           _______________________________
                Final (A) = 111 111 111 101 111 111 101 010
                             7   7   7   5   7   7   5   2
```

SCA
Selectively Complement A

23 ... 18	17	16 15	14 ... 00
36	a	b	m

a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Selectively complement the bits in the A register that correspond to the "1" bits in the quantity at address M.

```
EXAMPLE:  (A) = 23456710 = 010 011 100 101 110 111 001 000
          (M) = 20341573 = 010 000 011 100 001 101 111 011
                           _______________________________
                Final (A) = 000 011 111 001 111 010 110 011
                             0   3   7   1   7   2   6   3
```

LPA
Logical Product A

23 18	17	16 15	14 00
37	a	b	m

a = addressing mode designator
b = index register designator
m = storage address; M = m + (B^b)

Instruction Description: Replace (A) with the logical product of (A) and (M).

```
EXAMPLE:  (A) = 23456710 = 010 011 100 101 110 111 001 000
          (M) = 45210376 = 100 101 010 001 000 011 111 110
                Final (A) = 000 001 000 001 000 011 001 000
                             0   1   0   1   0   3   1   0
```

ANI
AND of B^b and y

23 18	17	16 15	14 00
17	0	b	y

b = index register designator

Instruction Description: Enter the logical product (the AND function) of y and (B^b) back into the same index register.

Comments: If b = 0, this is a no-operation instruction.

ANA
AND of A and y

23 18	17 15	14 00
17	6	y

Instruction Description: Enter the logical product (the AND function) of y and (A) back into the A register.

ANA, S
AND of A and y, Sign Extended

23 18	17 15	14 00
17	4	y

Instruction Description: Same as ANA except the sign of y is extended.

ANQ
AND of Q and y

23 18	17 15	14 00
17	7	y

Instruction Description: Enter the logical product (the AND function) of y and (Q) back into the Q register.

ANQ, S
AND of Q and y, Sign Extended

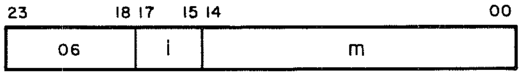
23 18	17 15	14 00
17	5	y

Instruction Description: Same as ANQ except the sign of y is extended.

Masked Searches and Compare Instructions

Operation Field		Address Field	Interpretation
MEQ	06	m, i	Masked equality search
MTH	07	m, i	Masked threshold search
CPR, I	52	m, b	Compare (within limits test)

MEQ Masked Equality Search



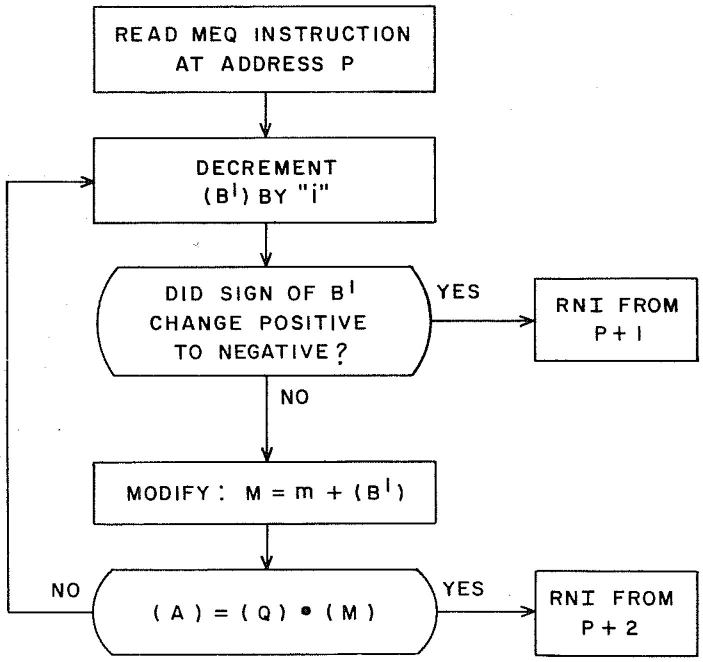
i = interval designator, 0 to 7
m = storage address

Instruction Description: (A) is compared with the logical product of (Q) and (M). This instruction uses index register B^1 exclusively. m is modified just prior to step 3 in the test below. Instruction sequence follows:

1. Decrement (B^1) by i. (Refer to table below.)
2. If (B^1) changed sign from positive to negative, RNI from P + 1; if not,
3. Test to see if (A) = (Q) ● (M). M = m + (B^1).
   If (A) = (Q) ● (M), RNI from P+2; if not,
4. Repeat the sequence.

Comments: i is represented by 3 bits, permitting a decrement interval selection from 1 to 8. Address modification always utilizes (B^1). Positive zero and negative zero are recognized as equal quantities.

DESIGNATOR i	DECREMENT INTERVAL
1	1
2	2
3	3
4	4
5	5
6	6
7	7
0	8



MTH
Masked
Threshold Search

23 18	17 15	14 00
07	i	m

i = interval designator, 0 to 7
m = storage address

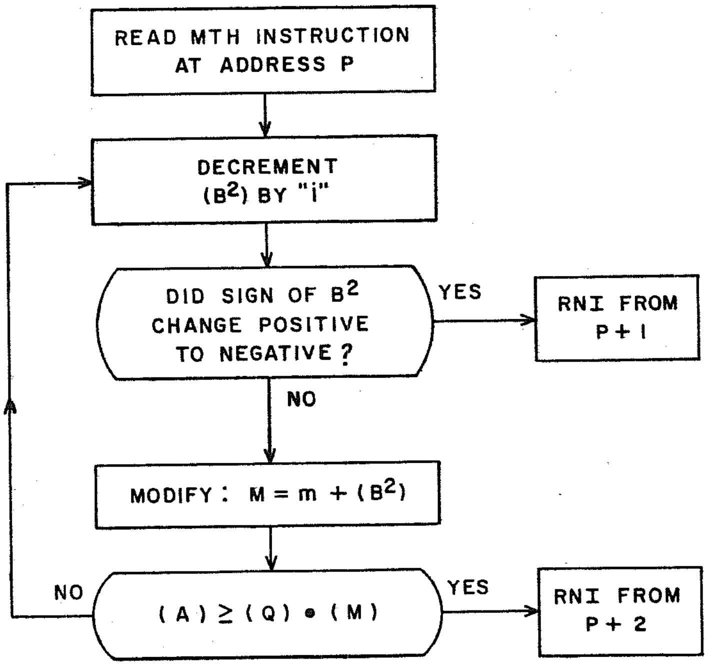
Instruction Description: (A) is compared with the logical product of (Q) and (M). This instruction uses index register B^2 exclusively. m is modified just prior to step 3 in the test below.

Instruction Sequence:

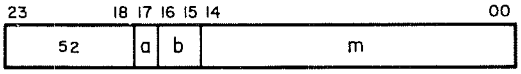
1. Decrement (B^2) by "i". (Refer to table below.)
2. If (B^2) changed sign from positive to negative, RNI from P + 1; if not,
3. Test to see if $(A) \geq (Q) \bullet (M)$. $M = m + (B^2)$.
   If $(A) \geq (Q) \bullet (M)$, RNI from P + 2; if not,
4. Repeat the sequence.

Comments: i is represented by 3 bits, permitting a decrement interval selection from 1 to 8. Address modification always utilizes (B^2). Positive zero and negative zero are recognized as equal quantities.

DESIGNATOR i	DECREMENT INTERVAL
1	1
2	2
3	3
4	4
5	5
6	6
7	7
0	8


READ MTH INSTRUCTION AT ADDRESS P
DECREMENT (B^2) BY "i"
DID SIGN OF B^2 CHANGE POSITIVE TO NEGATIVE ?
YES
RNI FROM P + 1
NO
MODIFY : $M = m + (B^2)$
NO
$(A) \geq (Q) \bullet (M)$
YES
RNI FROM P + 2

CPR
Compare
(Within Limits Test)



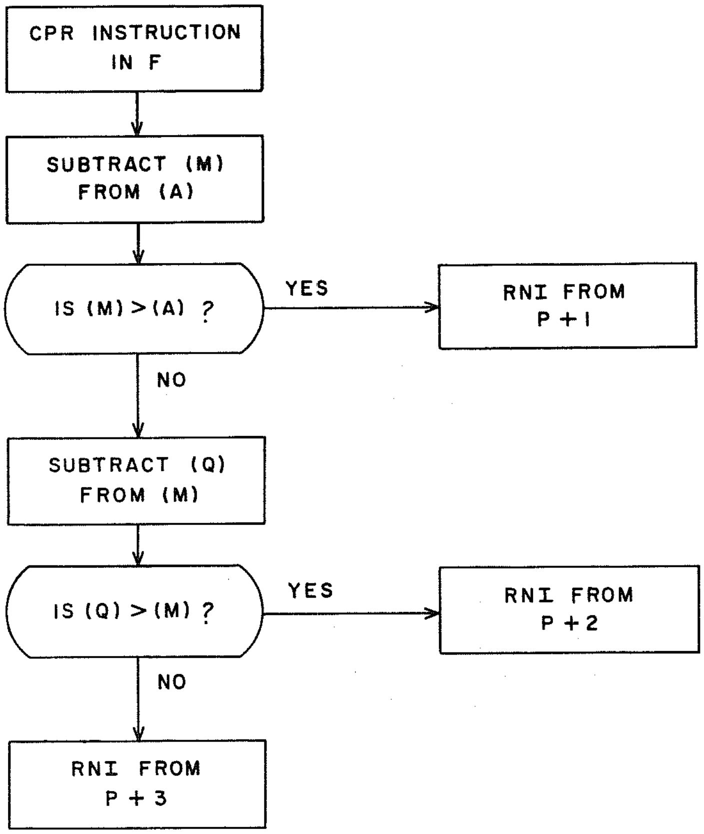
a = addressing mode designator
b = index register designator
m = storage address; $M=m+(B^b)$

Instruction Description: The quantity stored at address M is tested to see if it is within the upper limits specified by A and the lower limits specified by Q. The testing proceeds as follows:

1. Subtract (M) from (A). If (M) > (A), RNI from address P + 1; if not,
2. Subtract (Q) from (M). If (Q) > (M), RNI from P + 2; if not,
3. RNI from address P + 3.

Comments: The final state of the A and Q registers remains unchanged. (A) must be ≥ (Q) initially or the test cannot be satisfied. 77777777 is not sensed as negative zero. The following table is a synopsis of the CPR test:

Test Sequence	Jump Address if Test is Satisfied
(M) > (A)	P + 1
(Q) > (M)	P + 2
(A) ≥ (M) ≥ (Q)	P + 3



CPR FLOW CHART

Condition Test Instructions

Operation Field	Address Field	Interpretation
TMAV 77		Test memory availability

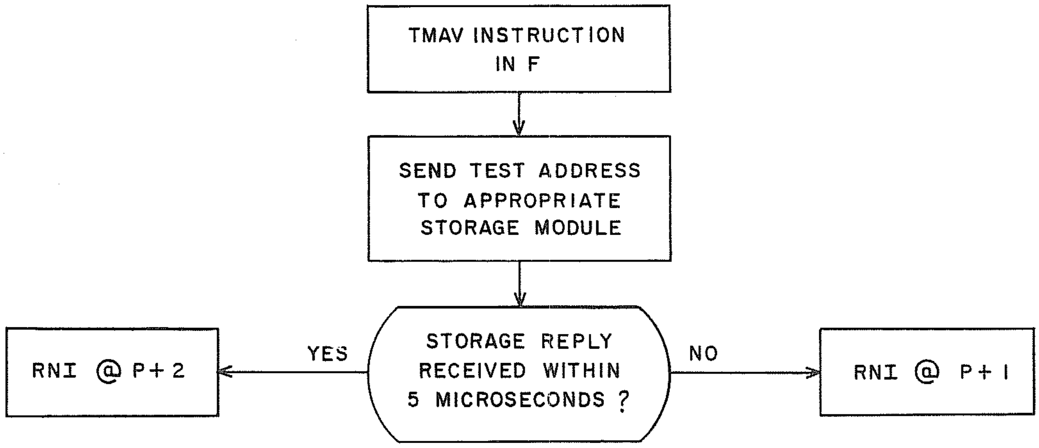
TMAV
Test
Memory Availability

23 18	17 12	11 00
77	61	0000

Instruction Description: This instruction is used to test core storage for the presence of a particular address.

Comments: Prior to executing this instruction, the lower 15 bits of the testing address must be formed in B^2. The upper 3 bits of the address will be zeros unless the OSR has been previously selected by the (ROS) 55.4 instruction.

If a storage reply is received within 5 microseconds after executing the instruction, the address does exist, and the next instruction is read from P + 2. If a reply does not occur within 5 microseconds, the address does not exist in the system and the next instruction is read from P + 1. The contents of the test address are not returned to the CPU and are of no consequence during the test.


TMAV INSTRUCTION IN F
SEND TEST ADDRESS TO APPROPRIATE STORAGE MODULE
STORAGE REPLY RECEIVED WITHIN 5 MICROSECONDS ?
YES
RNI @ P+2
NO
RNI @ P+1

Sensing Instructions

Operation Field		Address Field	Interpretation
EXS	77	x, ch;x≠0	Sense external status
COPY	77	x, ch;x=0	Copy external status
INTS	77	x, ch	Sense interrupt
INS	77	x, ch;x≠0	Sense internal status
CINS	77	x, ch;x=0	Copy internal status

GENERAL NOTES

Refer to the ACI instruction for special considerations regarding the 'ch' designator in these instructions. Refer to the SSIM instruction in the Interrupt group for a method of program testing for the presence of I/O channels in a system.

EXS
Sense
External Status

23 18	17 15	14 12	11 00
77	2	ch	x

ch = I/O channel designator, 0-7
x = external status sensing mask code (see Comments below)

Instruction Description: When a peripheral equipment controller is connected to an I/O channel by the CON (77.0) instruction, the EXS instruction can sense conditions within that controller. Twelve status lines run between each controller and its I/O channel. Each line may monitor one condition within the controller, and each controller has a unique set of line definitions. To sense a specific condition, a "1" is placed in the bit position of the status sensing mask that corresponds to the line number. When this instruction is recognized, RNI at address P + 1 if an external status line is active when its corresponding mask bit is "1". If no selected line is active, RNI at address P + 2.

Comments: Refer to the 3000 Series Computer Systems Peripheral Equipment Codes manual (Pub. No. 60113400) for a complete list of status response codes.

COPY
Copy
External Status

23 18	17 15	14 12	11 00
77	2	ch	0000

ch = I/O channel designator, 0-7

Instruction Description: This instruction performs the following functions:

1. The external status code from I/O channel ch is loaded into the lower 12 bits of A. (See EXS instruction.)
2. The contents of the Interrupt Mask register are loaded into the upper 12 bits of A. (See Table 5-3.)
3. RNI from address P + 1.

TABLE 5-3. INTERRUPT MASK REGISTER BIT ASSIGNMENTS

Masked Bit Positions	Mask Codes (x)	Interrupt Conditions Represented
00	0001	I/O Channel 0 (includes interrupt gener-
01	0002	1 ated within the channel
02	0004	2 and external equipment
03	0010	3 interrupts)
04	0020	4
05	0040	5
06	0100	6
07	0200	7
08	0400	Real-time clock
09	1000	Exponent overflow/underflow & BCD faults
10	2000	Arithmetic overflow & divide faults
11	4000	Search/Move completion

INTS
Sense Interrupt

23 18	17 15	14 12	11 00
77	4	ch	x

ch = I/O channel designator, 0-7
x = interrupt sensing mask code

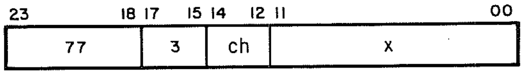
Instruction Description: Sense for the interrupt conditions listed in Table 5-4, RNI from P + 1 if an interrupt line is active and the corresponding sensing mask bit is a "1". If none of the selected lines are active, RNI from P + 2. Bits 08-11 represent conditions that may be sensed without regard to channel designation.

TABLE 5-4. BIT ASSIGNMENTS FOR INTERRUPT SENSING CONDITIONS

Mask Bit Positions	Mask Codes (x)	Interrupt Conditions Represented
00	0001	External equipment interrupt line 0 active
01	0002	1
02	0004	2
03	0010	3
04	0020	4
05	0040	5
06	0100	6
07	0200	7
08	0400	*Real-time clock
09	1000	*Exponent overflow/underflow & BCD faults
10	2000	*Arithmetic overflow & divide faults
11	4000	*Search/Move completion

* Internal faults are cleared as soon as they are sensed.

INS
Sense Internal Status



ch = I/O channel designator, 0-7
x = internal status sensing mask code

Instruction Description: Table 5-5 lists the bit definitions of the internal status sensing mask. Bits 00-04 and 06-07 represent conditions within I/O channel Bits 05 and 08-11, which represent internal faults, may be sensed without regard to channel designation.

To sense a specific condition, load a "1" into the bit position of the mask that corresponds to the condition. When this instruction is executed, RNI from address P + 1 if an internal status line is active and the corresponding mask bit is a "1". If none of the selected lines is active, RNI from address P + 2.

TABLE 5-5. INTERNAL STATUS SENSING MASK

Mask Bit Positions	Mask Codes (x)	Condition Represented
00	0001	Parity error on channel ch
01	0002	Channel ch busy reading
02	0004	Channel ch busy writing
03	0010	External reject active on channel ch
04	0020	No-response reject active on channel ch
05	0040	*Illegal write
06	0100	Channel ch preset by CON or SEL, but no reading or writing in progress
07	0200	Internal I/O channel interrupt on channel ch upon: 1) completion of read or write operation, or 2) end of record
08	0400	*Exponent overflow/underflow fault (floating point)
09	1000	*Arithmetic overflow fault (adder)
10	2000	*Divide fault
11	4000	*BCD fault

*Internal faults are cleared as soon as the condition is sensed.

CINS
Copy Internal Status

23 18	17 15	14 12	11 00
77	3	ch	0000

ch = I/O channel designator, 0-7

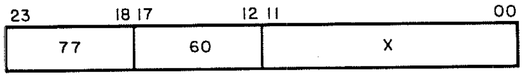
Instruction Description: This instruction performs the following functions:

1. The internal status code is loaded into the lower 12 bits of A. (See INS instruction.)
2. The contents of the Interrupt Mask register are loaded into the upper 12 bits of A. (See Table 5-3.)
3. RNI from address P + 1.

Pause Instructions

Operation Field		Address Field	Interpretation
PAUS	77	x	Pause on condition
PRP	77	x	Priority pause

PAUS
Pause



x = pause sensing mask code

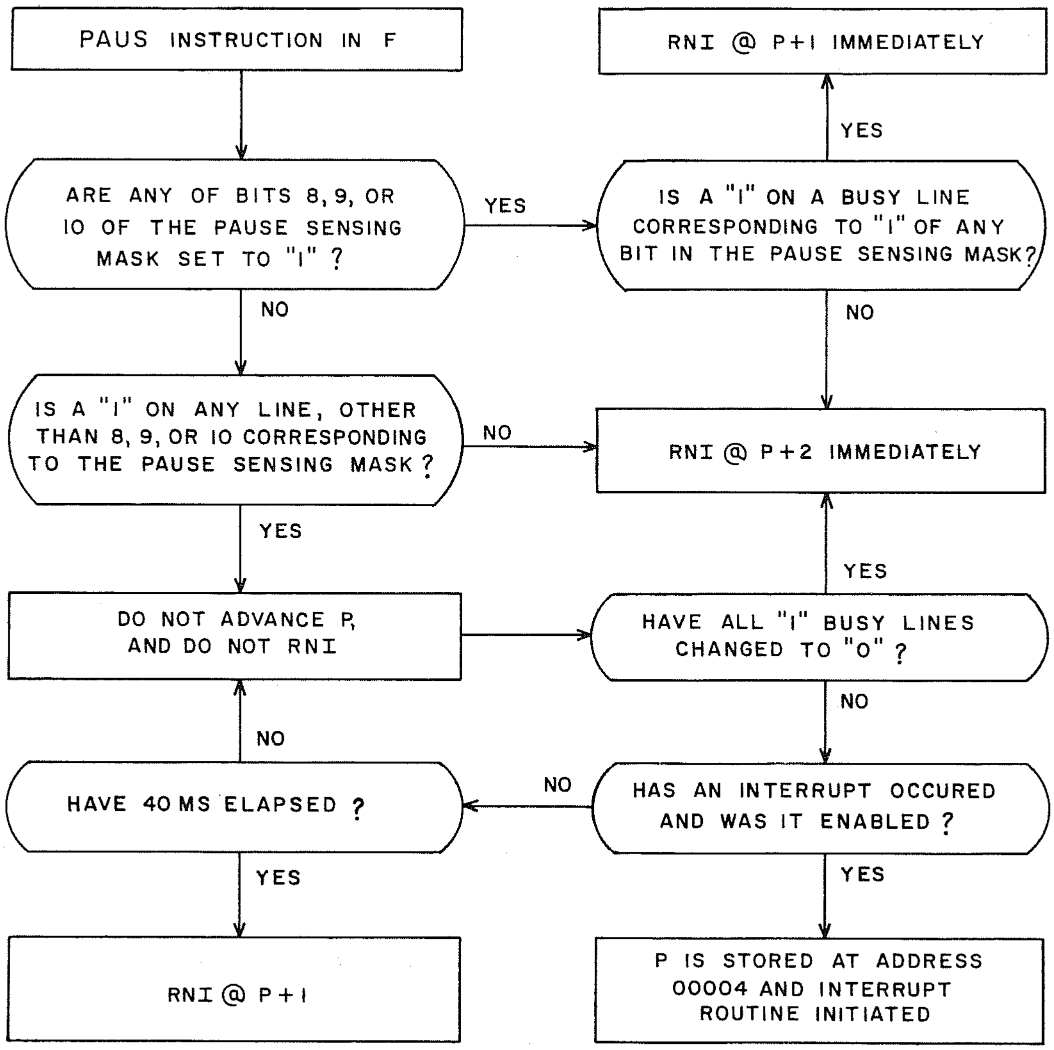
Instruction Description: This instruction allows the program to halt for a maximum of 40 ms if a condition (excluding typewriter - see note) defined by the pause sensing mask exists. (See Table 5-6.) If a "1" appears on a line that corresponds to a mask bit that is set, the count in P will not advance. If the advancement of P is delayed for more than 40 ms, the next instruction is read from address P + 1. If none of the lines being sensed are active,or if they become inactive during the pause, the program immediately skips to address P + 2. If an interrupt occurs and is enabled during a PAUS, the pause condition is terminated, the interrupt sequence is initiated and the address of the PAUS instruction is stored as the interrupted address.

NOTE

> If either bit 08, 09 or 10 (or any combination of these bits) is set and the sensed condition exists, a pause will not occur and the instruction at P + 1 is read up immediately. If these bit(s) are set but the condition(s) does not exist, the program immediately skips to P + 2. For all other bits, the normal PAUS routine is followed. TYPE FINISH and/or TYPE REPEAT are cleared if bit 9 and/or 10 are set and the sensed condition(s) does not exist.

TABLE 5-6. PAUSE SENSING MASK

Mask Bits	Mask Codes	Condition	Notes
00	0001	I/O channel 0 busy	Channel read or write operation in progress, the External MC logic within the channel is set, or a Reply or Reject from a previous operation is still present at the channel
01	0002	1	
02	0004	2	
03	0010	3	
04	0020	4	
05	0040	5	
06	0100	6	
07	0200	7	
08	0400	Typewriter busy	Typewriter input or output in progress
09	1000	Typewriter NOT finish	Finish logic not set
10	2000	Typewriter NOT repeat	Repeat logic not set
11	4000	Search/Move control busy	Search or Move operation in progress


PAUS INSTRUCTION IN F
ARE ANY OF BITS 8, 9, OR 10 OF THE PAUSE SENSING MASK SET TO "1" ?
YES
IS A "1" ON A BUSY LINE CORRESPONDING TO "1" OF ANY BIT IN THE PAUSE SENSING MASK?
YES
RNI @ P+1 IMMEDIATELY
NO
NO
IS A "1" ON ANY LINE, OTHER THAN 8, 9, OR 10 CORRESPONDING TO THE PAUSE SENSING MASK ?
NO
RNI @ P+2 IMMEDIATELY
YES
YES
DO NOT ADVANCE P, AND DO NOT RNI
HAVE ALL "1" BUSY LINES CHANGED TO "0" ?
NO
NO
HAVE 40 MS ELAPSED ?
NO
HAS AN INTERRUPT OCCURED AND WAS IT ENABLED ?
YES
YES
RNI @ P+1
P IS STORED AT ADDRESS 00004 AND INTERRUPT ROUTINE INITIATED

PRP
Priority Pause

23 18	17 12	11 00
77	61	x

x = pause sensing mask code

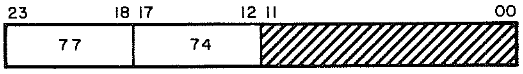
Instruction Description: This instruction performs the same operation as the preceding PAUS (77.6) instruction, however, the real-time clock is prevented from incrementing during the pause.

Comments: Preventing the real-time clock from incrementing enables block control to continue an I/O operation without being referenced by the clock. This provides a more efficient I/O transfer operation.

Interrupt Instructions

Operation Field	Address Field	Interpretation
EINT 77		Enable interrupt control
DINT 77		Disable interrupt control
INCL 77	x	Clear interrupt
SSIM 77	x	Selectively set interrupt mask
SCIM 77	x	Selectively clear interrupt mask
CILO 77	cm	Channel interrupt lockout
SFPF 77		Set floating point fault
SBCD 77		Set BCD fault

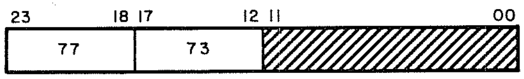
EINT
Enable
Interrupt Control



Instruction Description: This instruction enables the interrupt control system. One additional instruction at P + 1 is executed before the processor is interrupted, provided that the additional instruction requires no more than one Read Address (RADR) cycle. If the EINT instruction is executed at P, the earliest possible interrupt will occur at P + 2. For an instruction containing more than one RADR cycle, the earliest possible interrupt can occur during its second RADR cycle.

Comments: Bits 00 through 11 should be loaded with zeros.

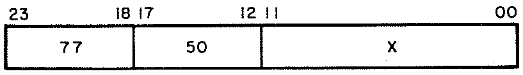
DINT
Disable
Interrupt Control



Instruction Description: This instruction disables the interrupt control system. The system remains disabled until an EINT instruction is executed. Selected interrupts may still be sensed.

Comments: Bits 00 through 11 should be loaded with zeros.

INCL
Clear Interrupt



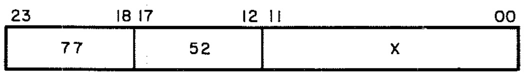
x = interrupt mask register codes

Instruction Description: This instruction clears the interrupt faults defined by the mask codes in Table 5-7. Internal I/O channel interrupts are cleared by this instruction and although the Interrupt Clear is sent to peripheral equipment, not all equipments drop their interrupt lines. Refer to the Peripheral Equipment Reference Manual, Pub. No. 60108800, for information of specific equipment.

TABLE 5-7. INTERRUPT MASK REGISTER BIT ASSIGNMENTS

Mask Bit Positions	Mask Codes (x)	Interrupt Conditions Represented
00	0001	I/O Channel 0 (includes interrupts gener-
01	0002	1 ated within the channel
02	0004	2 and external equipment
03	0010	3 interrupts)
04	0020	4
05	0040	5
06	0100	6
07	0200	7
08	0400	Real-time clock
09	1000	Exponent overflow/underflow & BCD faults
10	2000	Arithmetic overflow & divide faults
11	4000	Search/Move completion

SSIM
Selectively Set Interrupt Mask Register



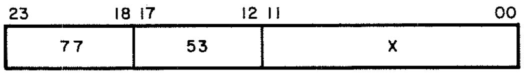
x = interrupt mask register codes

Instruction Description: This instruction selectiveiy sets the Interrupt Mask register according to the interrupt mask code x.* For each bit set to "1" in x, the corresponding bit position in the Interrupt Mask register is set to "1" (see Table 5-7). Bit positions representing non-existent I/O channels cannot be set.

Comments: A program test for the existence of I/O channels for a system is as follows:

1. Set the interrupt mask bits to all "1"s by executing a SSIM (77 5 27777) instruction.
2. Execute either a COPY or CINS instruction and examine the upper 12 bits of A.
3. As bits representing non-existent I/O channels cannot be set, a "0" in bits 00-07 of the Interrupt Mask register indicates a non-existent I/O channel.

SCIM
Selectively Clear Interrupt Mask Register

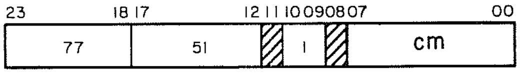


x = interrupt mask register codes

Instruction Description: This instruction selectively clears the Interrupt Mask register according to the interrupt mask code x.* For each bit set to "1" in x, the corresponding bit position in the Interrupt Mask register is set to "0" (see Table 5-7).

*The Interrupt Mask register must not be set or cleared while the interrupt system is enabled to prevent extraneous interrupts from occuring.

CILO
Channel Interrupt Lockout



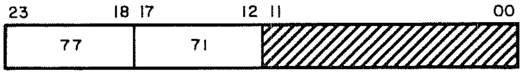
cm = channel mask

Bits 08 and 11 should be loaded with zeros.

Instruction Description: Disables all external interrupts on channel(s) cm while the channel(s) are busy. Termination of the I/O operation clears the disabling function.

Comments: Bit 00 corresponds to channel 0, bit 01 corresponds to channel 1, etc. More than one channel may be set to "1" for multiple channel interrupt lockout. The mask is cleared by termination of the I/O operation, by clearing the channel(s), and by a Negate Channel Interrupt Lockout signal from certain peripherals.

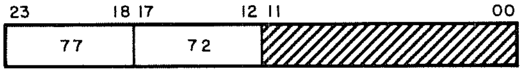
SFPF
Set Floating Point Fault



Instruction Description: The floating point fault logic sets when a floating point fault occurs. This instruction is used when the optional floating point arithmetic logic is not present in a system. An interpretive software routine should recognize any conditions which would have caused a fault if the operation had been executed by the optional hardware.

Comments: Bits 00 through 11 should be loaded with zeros.

SBCD
Set BCD Fault



Instruction Description: The BCD fault logic sets when a BCD fault occurs. This instruction is used when the optional BCD arithmetic is not present in a system. An interpretive software routine should recognize any condition which would have caused a fault if the operation had been executed by the optional hardware.

Comments: Bits 00 through 11 should be loaded with zeros.

Input/Output Instructions

Operation Field		Address Field	Interpretation
CLCA	77	cm	Clear I/O channel(s)
IOCL	77	x	Clear I/O channel(s) and equipment
CON	77	x, ch	Connect to external equipment
SEL	77	x, ch	Select function
CTI	77		Set console typewriter input
CTO	77		Set console typewriter output
INPC, INT, B,H,, A	73	ch, r, s	Character-Addressed Input to storage
INAC, INT	73	ch	Character-Addressed Input to A
INPW, INT, B,N, A	74	ch, m, n	Word-Addressed Input to storage
INAW, INT	74	ch	Word-Addressed Input to A
OUTC, INT, B,H	75	ch, r, s	Character-Addressed Output from storage
OTAC, INT	75	ch	Character-Addressed Output from A
OUTW, INT, B,N	76	ch, m, n	Word-Addressed Output from storage
OTAW, INT	76	ch	Word-Addressed Output from A

Unlike I/O operations with A, I/O instructions with storage are buffered. As soon as Read or Write signals are activated, Main Control relinquishes control of the storage I/O operation and returns to the main program.

Registers 00 through 17_8 of the Register File are reserved for I/O operations. Registers 00 through 07 are used for storing the modified instruction words containing the current character addresses. (Refer to Table 5-8). Registers 10_8 through 17 hold the modified sub-instruction words containing the last character addresses (± 1 depending upon the instruction parameters). In cases where addresses require modification to obtain dynamic I/O operations, care should be exercised to provide proper readout and restoration of the modified control bits.

During the execution of word addressed storage I/O instructions, the addresses 'm' and 'n' are shifted left two bit positions. From this time on and when they are stored in the Register File, they are recognized as character addresses.

Before executing an I/O instruction in Executive mode, the desired program state number (0 through 7) must be loaded into the lower three bits of the A register. The program state number is automatically transferred to the upper digit of Register File location 0X for referencing during buffered I/O tasks, thus enabling the A register to be used for other operations.

Table 5-8 and its accompanying example illustrate the relationship between the Register File addresses, their contents, and the individual instructions. Each I/O instruction should be referenced for a description of its particular parameters and a flowchart of the overall operation. The ACI instruction, described elsewhere in this section, should be consulted for special I/O channel considerations.

When performing I/O operations with peripheral equipment not equipped with a 12 to 6-bit disassembly feature, character oriented instructions should not be used, thus preventing erroneous transmission parity errors.

TABLE 5-8. MODIFIED I/O INSTRUCTION WORDS

	Instruction	Relative location of instruction words (See individual instructions)	Register File location	Contents of Register File location
Operations With Storage	73 (INPC)	P	1X	3 - - - - - - -
		P + 1	0X	* - - - - - - -
	74 (INPW)	P	1X	0 - - - - - - -
		P + 1	0X	* - - - - - - -
	75 (OUTC)	P	1X	1 - - - - - - -
		P + 1	0X	* - - - - - - -
	76 (OUTW)	P	1X	2 - - - - - - -
		P + 1	0X	* - - - - - - -
Operations With A		P	1X	7 - - - - - - -
	73 (INAC)	P + 1	0X	- - - - - - - -
		P	1X	4 - - - - - - -
	74 (INAW)	P + 1	0X	- - - - - - - -
		P	1X	5 - - - - - - -
	75 (OTAC)	P + 1	0X	- - - - - - - -
		P	1X	6 - - - - - - -
	76 (OTAW)	P + 1	0X	- - - - - - - -

X = An I/O channel designator "ch", 0, 1, 2, 3, 4, 5, 6, or 7.
* = The program number (lowest 3 bits of the 'A' Register)

Upper digit position (Blanks indicate digits unaltered by control logic)

EXAMPLE:

Execute the following INPW instruction.

```
P =     74 003200          (A) = 00000001
P + 1 = 20 003100
P + 2 = 01 003300             3306 I/O Channel
```

ANALYSIS: I/O Channel 2 is specified; thus Register File location 12 is used to store 04015000 and location 02 holds 10014400.

This instruction specifies 12- to 24-bit assembly, no interrupt upon completion, forward storage, and an unconditional jump to address 03300 as a reject instruction.

The first word address (m) of the block of storage assigned to receive data from an external equipment is m = 03100.

The last word address (n) of the assigned storage area (plus one) is n = 03200.

The first 12-bit byte is stored in bits 12 through 23 at address 003100, the second byte in bits 00 through 11, etc.

CLCA
Clear Channel Activity

23 18	17 12	11	10 09	08	07 00
77	51	▨	2	▨	cm

cm = channel mask
Bits 08 and 11 should be loaded with zeros.

Instruction Description: Clear only the selected I/O channel(s).

Comments: The peripheral equipment associated with the selected channel(s) are not cleared by executing this instruction. Bit 00 corresponds to channel 0, bit 01 corresponds to channel 1, etc. More than one channel may be set to "1" for multiple channel clearing.

IOCL
Clear I/O, Typewriter, and Search/Move

23 18	17 12	11 00
77	51	x

x = block control clearing mask

Instruction Description: This instruction may be used to clear the I/O channels. It also clears all associated peripheral equipment, the typewriter or the Search/ Move control according to bits set in the Block Control clearing mask. (See Table 5-9.)

TABLE 5-9. BLOCK CONTROL CLEARING MASK

Mask Bits	Mask Codes (x)	Controls Cleared
00	0001	I/O channel 0
01	0002	1
02	0004	2
03	0010	3
04	0020	4
05	0040	5
06	0100	6
07	0200	7
08	0400	Typewriter
09	1000	(see note)
10	2000	(see note)
11	4000	Search/Move

NOTE

If bits 09 and 10 are both set or both clear, the channel(s) specified by bits 00 through 07 of the mask are cleared, i.e., Read or Write, Status, and Channel Interrupt are cleared. A 5.5 usec Clear signal is also sent to the peripheral equipment and controllers connected to the selected channel(s).

If bit 09 is clear and bit 10 is set, the instruction will clear the channel(s) only and the 5.5 usec Clear signal is not transmitted. Bit 08 clears the typewriter as well as the Type Load or Type Dump logic in Block Control.

CON
Connect

23 18	17 15	14 12	11 00
77	0	ch	x

ch = I/O channel designator, 0-7
x = 12 bit connect code. Bits 09-11 select one of eight controllers which may be attached to channel ch. Bits 00-08 select the peripheral units connected to the controller.

Instruction Description: This instruction sends a 12-bit connect code along with a connect enable to an external equipment controller on I/O channel ch. If a Reply is received from the controller within 100 usec, the next instruction is read from address P + 2. If a Reject is received or there is no response within 100 usec, a reject instruction is read from address P + 1. If the I/O channel is busy, a reject instruction is read from address P + 1.

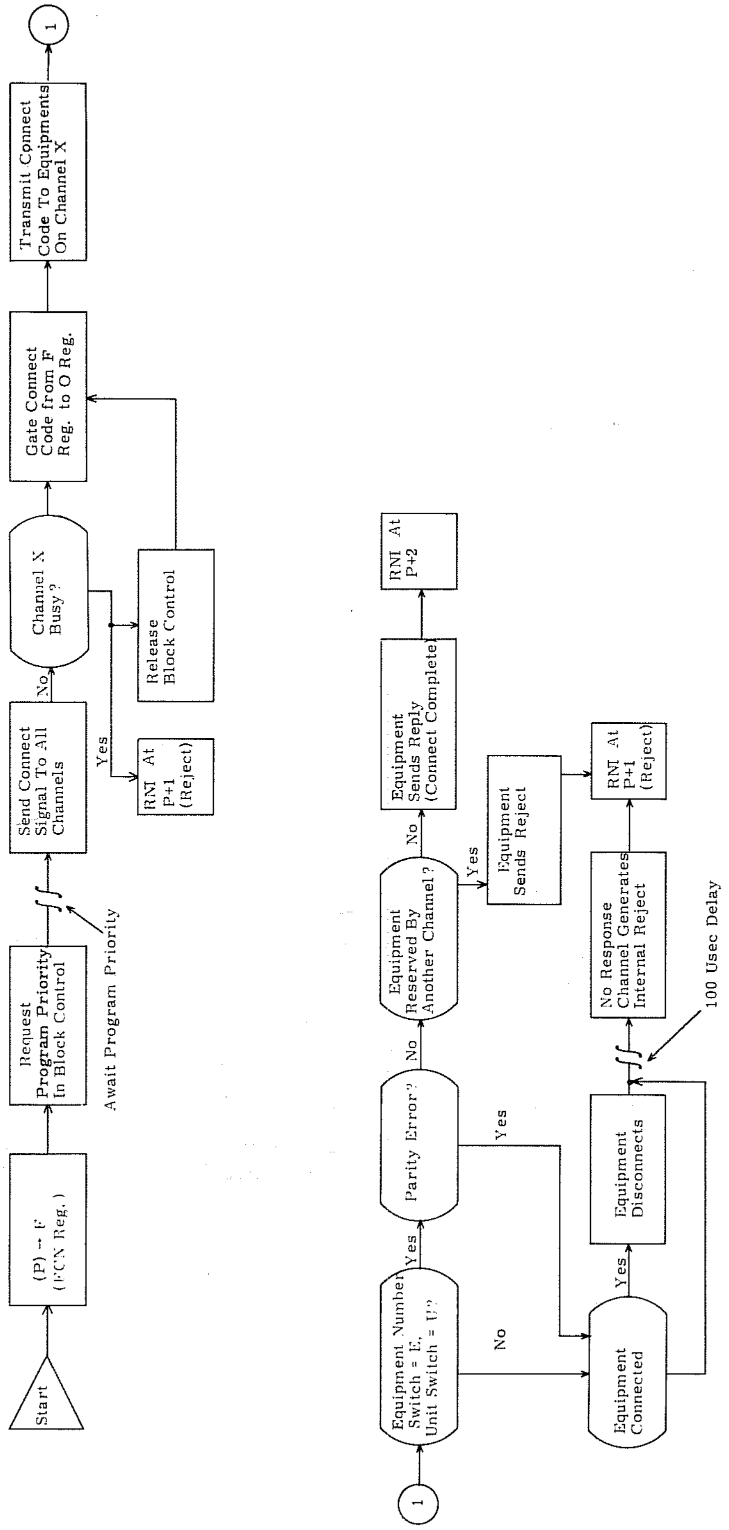


Figure 5-6. 77 Connect Operation

SEL
Select Function

23 18	17 15	14 12	11 00
77	I	ch	x

ch = I/O channel designator, 0-7

x = 12-bit function code. Each piece of external equipment has a unique set of function codes to specify operations within that device. Refer to the 3000 Series Computer Systems Peripheral Equipment Codes Publication No. 60113400 for a complete list of function codes.

Instruction Description: This instruction sends a 12-bit function code along with a function enable to the unit connected to I/O channel ch. If a Reply is received from the unit within 100 usec, the next instruction is read from P + 2. If a Reject is received or there is no response within 100 usec, or if the I/O channel is busy, a reject instruction is read from address P + 1.

The following conditions or combination of conditions result in a Reject:

1) No Unit or Equipment Connected: The referenced device is not connected to the system and cannot recognize a Select Function instruction. If no response is received within 100 usec, the Reject signal is generated automatically by the I/O channel.

2) Undefined Code: When the Function code x is not defined for the specific device, a Reject may be generated by the device. However, in some cases an undefined code will cause the device to generate a Reply although no operation is performed. (Refer to the reference manual pertaining to the specific peripheral device.)

3) Equipment or Unit Busy or Not Ready: The device cannot perform the operation specified by the function code x without damaging the equipment or losing data. For example, a Write End of File code is rejected by a tape unit if the tape unit is rewinding.

4) Channel Busy: The selected data channel is currently performing a Read or Write operation or a Reply or Reject from a previous operation is still present at the channel.

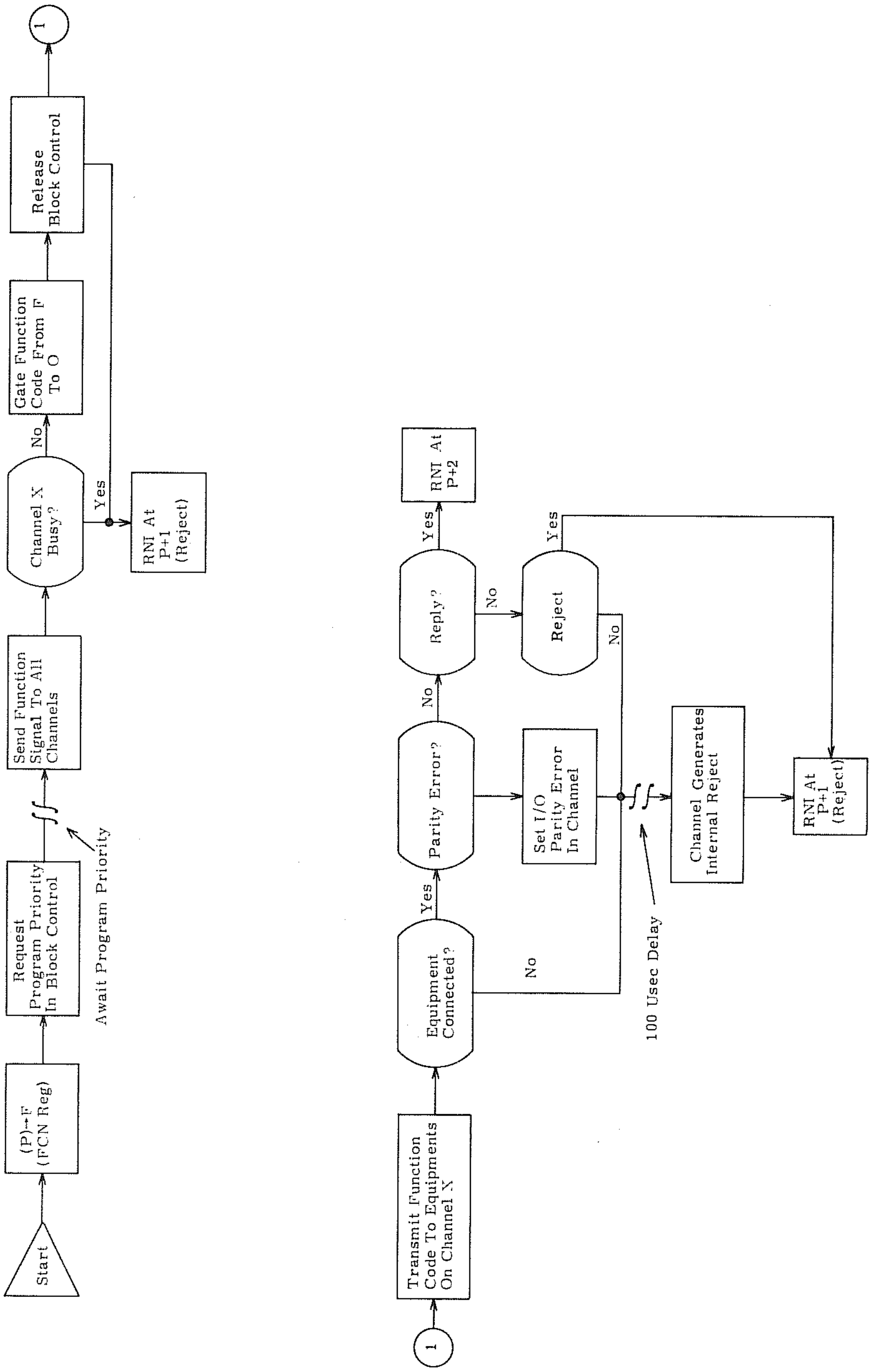

Start
(P)→F (FCN Reg)
Request Program Priority In Block Control
Await Program Priority
Send Function Signal To All Channels
Channel X Busy?
Yes
RNI At P+1 (Reject)
No
Gate Function Code From F To O
Release Block Control
1
1
Transmit Function Code To Equipments On Channel X
Equipment Connected?
Yes
No
Parity Error?
No
Set I/O Parity Error In Channel
Reply?
Yes
RNI At P+2
No
Reject
Yes
No
100 Usec Delay
Channel Generates Internal Reject
RNI At P+1 (Reject)

Figure 5-7. 77 Select Function Operation

CTI
Set Console Typewriter Input

23 18	17 12	11 00
77	75	▨

Bits 00 through 11 should be loaded with zeros.

Instruction Description: This instruction, like the TYPE LOAD switch, permits a block of data to be entered into storage as soon as the TYPE LOAD indicator lights. If a block of data smaller than the one defined by registers 23 and 33 is to be typed, the FINISH switch should be depressed when the typing is completed. If more data is entered than the defined block can hold, the excess data is lost. If a typing error occurs, the REPEAT button should be depressed. When either the FINISH or REPEAT switch is depressed, the typewriter input operation is terminated and the appropriate status bits (09 and 10) may be sensed with the PAUS instruction. (For additional information refer to the PAUS instruction.)

CTO
Set Console Typewriter Output

23 18	17 12	11 00
77	76	▨

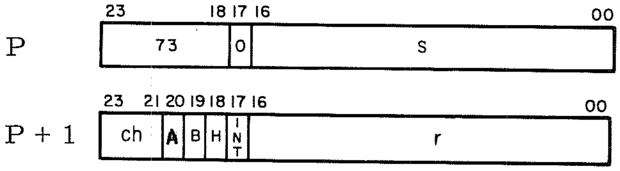
Bits 00 through 11 should be loaded with zeros.

Instruction Description: This instruction, like the TYPE DUMP switch, causes the typewriter to print out the block of data defined by the character addresses in registers 23 and 33.

NOTE

The CTI and CTO instructions are mutually exclusive. Typewriter busy should be checked before these instructions are used and before registers 23 and 33 are altered.

INPC
Character-Addressed Input to Storage



B = "1" for backward storage
ch = I/O channel designator, 0-7
H = "0" for 6- to 24-bit assembly
H = "1" for 12- to 24-bit assembly
INT = "1" for interrupt upon completion
r = first character address of I/O data block; becomes current address as I/O operation progresses
s = last character address of input data block, plus one (minus one, for backward storage)
A = "1" for word count control

Instruction Description: This instruction transfers a character address block of data, consisting of 6-bit characters or 12-bit bytes, from an external equipment to storage. During 6- to 24-bit assembly (H = 0), the lower 6 bits of successive data words (12-bit data words from the 3306, 24-bit data words from the 3307) are loaded into successive characters in storage, the first character being loaded into character address r. The next character is loaded into character address (r + 1) if doing forward storage (B = 0), into (r - 1) if doing backward storage (B = 1). During 12- to 24-bit assembly (H = 1), successive 12-bit bytes are loaded into successive halves of storage words. The first byte will be loaded into the upper or lower half of the storage word, depending upon character address r. If B = 0, the next byte is loaded into (r + 2); if B = 1, the next byte is loaded into (r - 2). During 12- to 24-bit assembly, the lowest bit of each character address is forced to remain a "0" in register 0X. This ensures that assembled bytes are in either the upper or the lower half of the word being stored.

If channel 'ch' is not busy, the buffered I/O operation with storage commences while Main Control performs an RNI at P + 3. Main Control continues executing the main program while the I/O operation occurs simultaneously. If channel 'ch' is initially busy, Main Control performs an RNI at P +2 and the I/O operation does not occur.

NOTES

When bit 20 of the subinstruction word at P + 1 is "1", the word count control feature allows this I/O operation to continue beyond an End-of-Record signal. In practice, when an End-of-Record signal is sent, the Read line drops but the buffer operation does not terminate. The Read signal sent to the external equipment then reappears until the word count is satisfied. This signal appears as a new input instruction to the external equipment, but as a continue Read to the I/O channel.

If H = "1", an even character count must be used. If the count is odd, the last character will be lost.

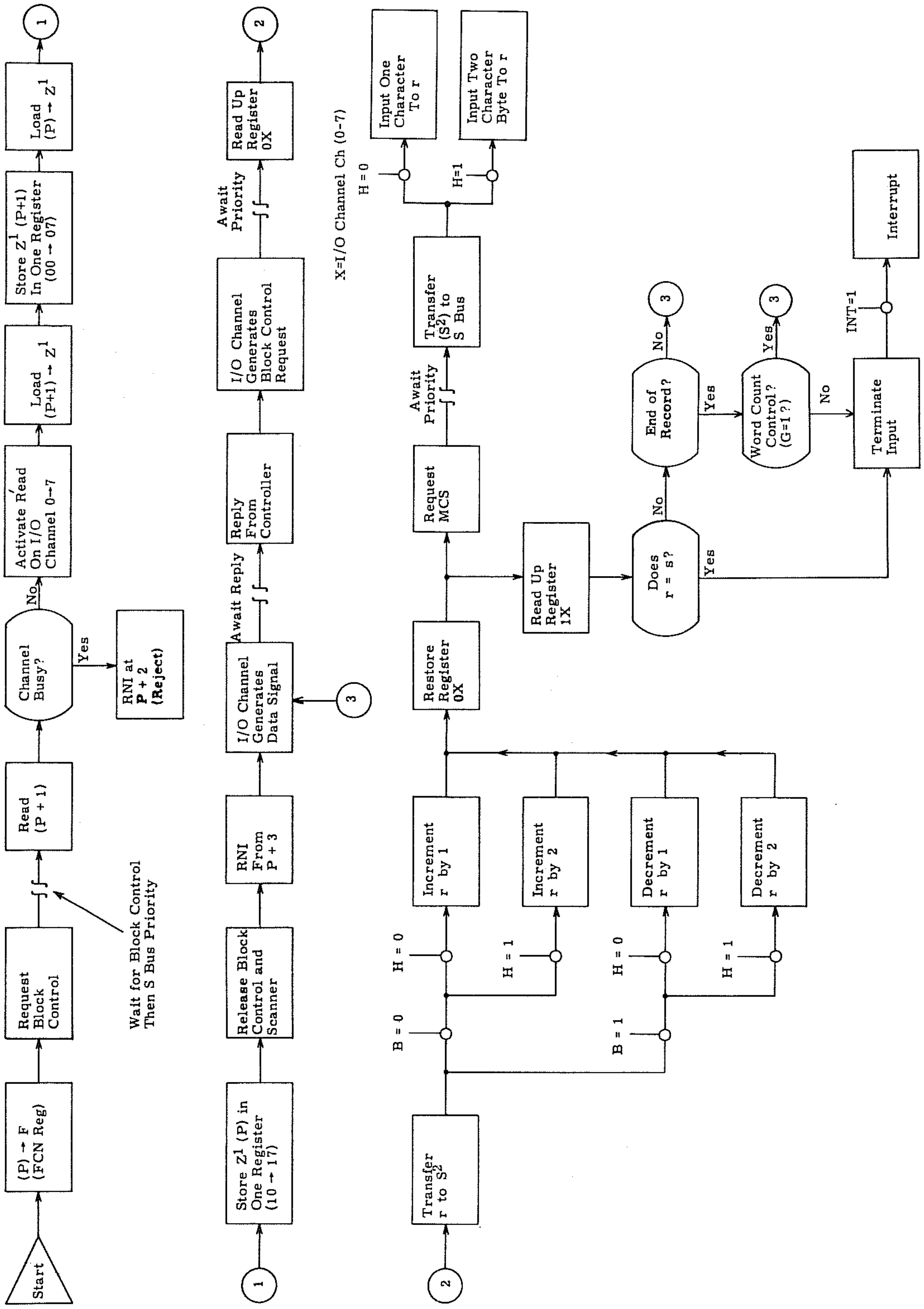
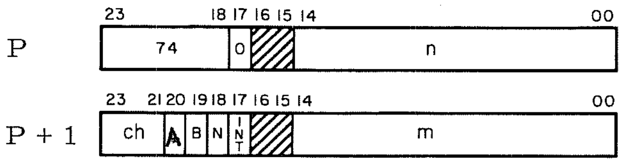


Figure 5-8. 73 I/O Operation with Storage

INPW
Word-Addressed Input to Storage



B = "1" for backward storage
ch = I/O channel designator, 0-7
INT = "1" for interrupt upon completion
N = "0" for 12- to 24-bit assembly
N = "1" for no assembly
m = first word address of I/O data block; becomes current address as I/O operation progresses
n = last word address of input data block, plus one (minus one, for backward storage)
A = "1" for word count control
Bits 15 and 16 at P and P + 1 should be loaded with zeros.

Instruction Description: This instruction transfers a word-addressed data block from an external equipment to storage. Transferring 12-bit bytes or 24-bit words depends upon the type of I/O channel used. The 3306 utilizes 12-bit bytes and the 3307 uses 24-bit words.

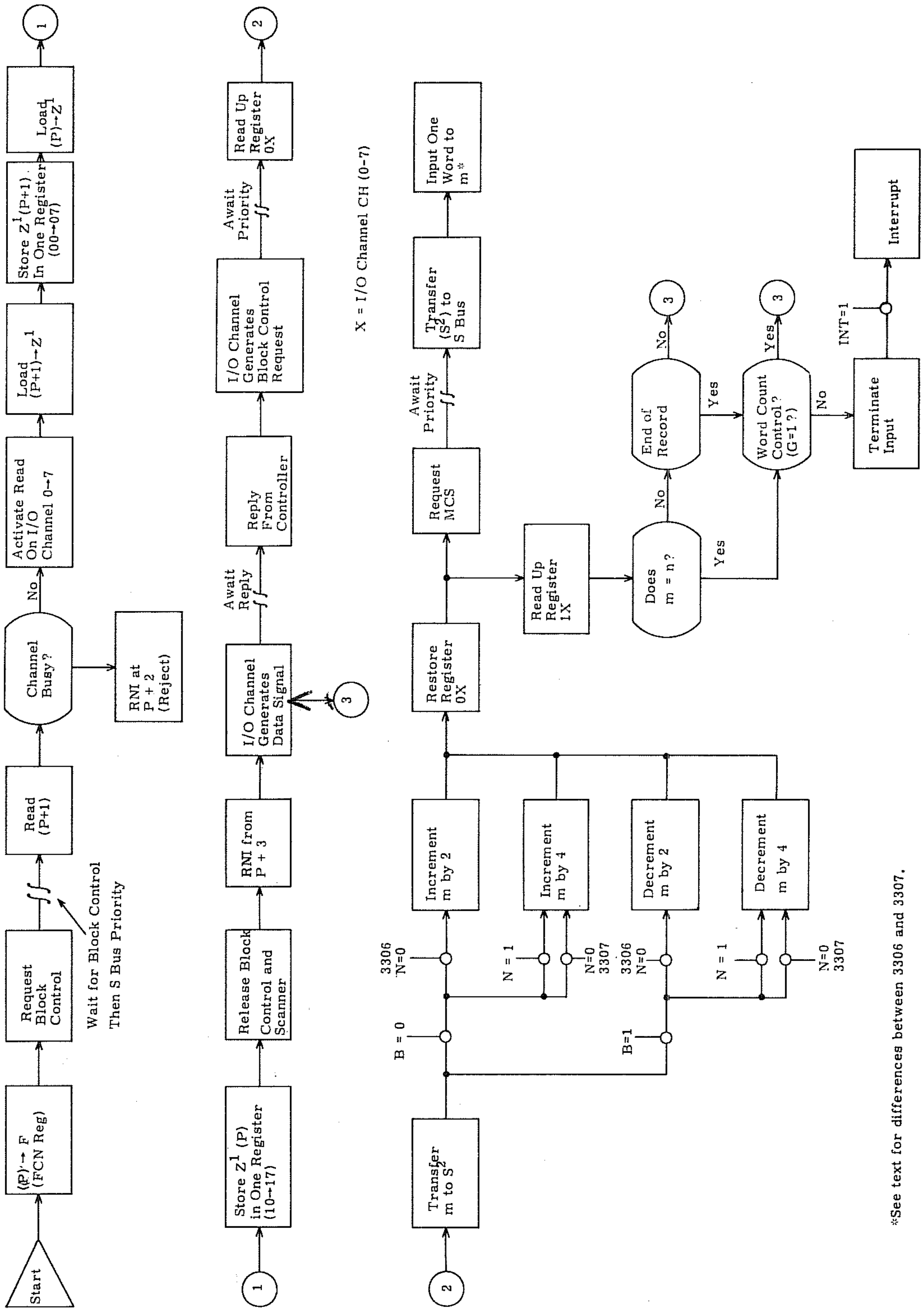
During forward storage and 12- to 24-bit assembly, the first byte of a block of data is stored in the upper half of the memory location specified by the storage address. Conversely, during backward storage, the first byte is stored in the lower half of the memory location.

If channel 'ch' is not busy, the buffered I/O operation with storage commences while Main Control performs an RNI at P + 3. Main Control continues executing the main program while the I/O operation occurs simultaneously. If channel 'ch' is initially busy, Main Control performs an RNI at P + 2 and the I/O operation does not occur.

NOTES

When bit 20 of the subinstruction word at P + 1 is "1", the word count control feature allows this I/O operation to continue beyond an End-of-Record signal. In practice, when an End-of-Record signal is sent, the Read line drops but the buffer operation does not terminate. The Read signal sent to the external equipment then reappears until the word count is satisfied. This signal appears as a new input instruction to the external equipment, but as a continue Read to the I/O channel.

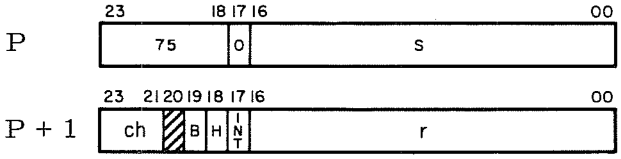
If N = 1 and a 3306 is used, the upper 12 bits of each storage word will be unchanged. If N = 1 and a 3307 is used with a 12-bit device, the upper 12 bits will be zeros.



*See text for differences between 3306 and 3307.

Figure 5-9. 74 I/O Operation with Storage

OUTC
Character-Addressed Output from Storage



B = "1" for backward storage
ch = I/O channel designator, 0-7
H = "0" for 24- to 6-bit disassembly
H = "1" for 24- to 12-bit disassembly
INT = "1" for interrupt upon completion
r = first character address of I/O data block; becomes current address as I/O operation progresses
s = last character address of output data block, plus one (minus one, for backward output)
Bit 20 at P + 1 should be loaded with a "0"

Instruction Description: This instruction transfers a character-addressed block of data, consisting of 6-bit characters or 12-bit bytes, from storage to an external equipment. During 24- to 6-bit disassembly (H = 0), the first character is transferred from character address r. The next character is transferred from (r + 1) if forward storage (B = 0), from (r - 1) if backward storage (B = 1). During 24- to 12-bit disassembly (H = 1), the first byte will be transferred from the upper or lower half of the storage word, depending upon character address r. If B = 0, the next byte is transferred from (r + 2); if B = 1, the next byte is transferred from (r - 2).

If channel 'ch' is not busy, the buffered I/O operation with storage commences while Main Control performs an RNI at P + 3. Main Control continues executing the main program while the I/O operation occurs simultaneously. If channel 'ch' is initially busy, Main Control performs an RNI at P + 2 and the I/O operation does not occur.

NOTE

If H = "1", an even character count must be used. If the count is odd, the last character will be lost.

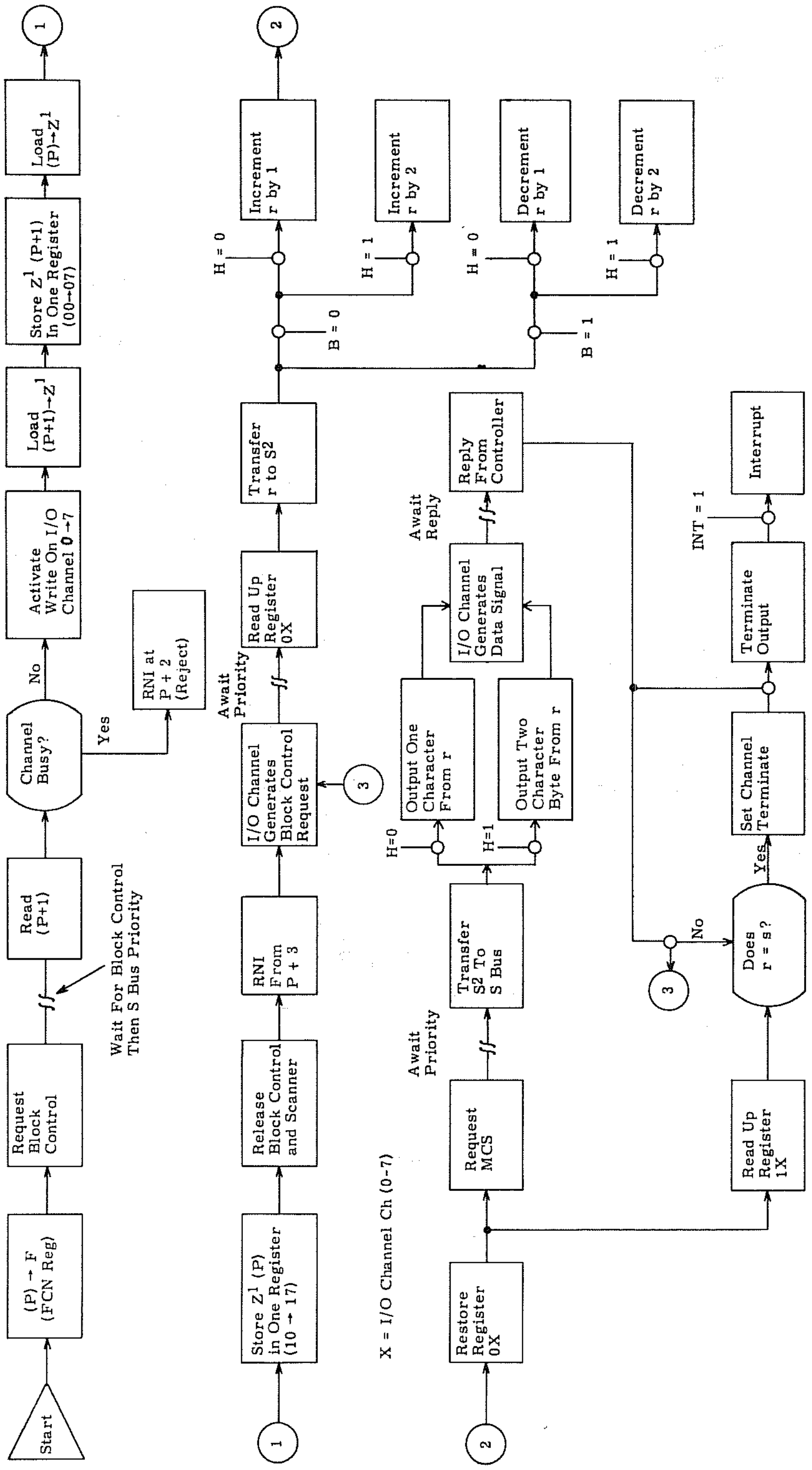
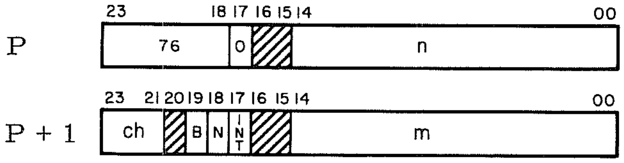


Figure 5-10. 75 I/O Operation with Storage

OUTW
Word-Addressed Output from Storage

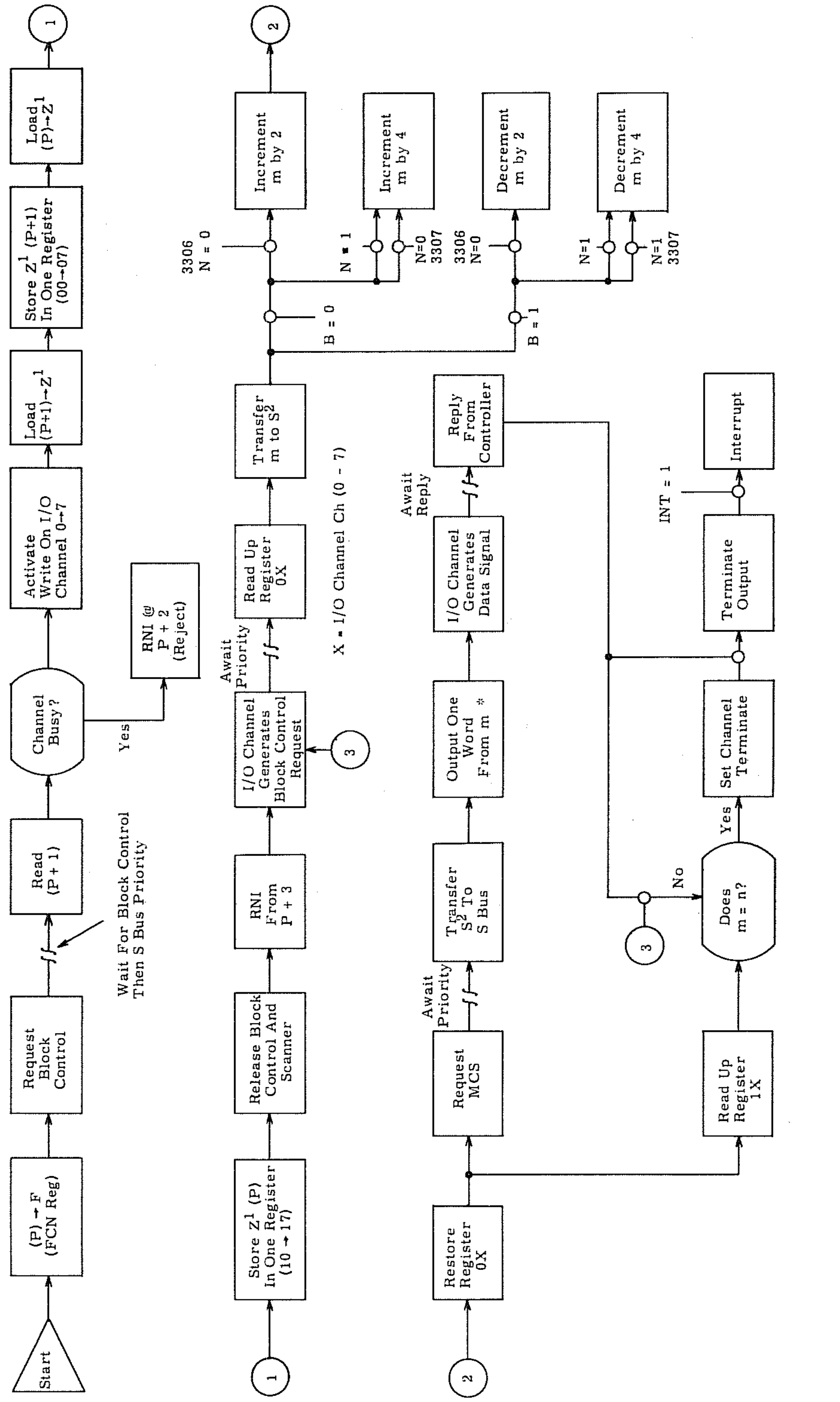


B = "1" for backward storage
ch = I/O channel designator, 0-7
INT = "1" for interrupt upon completion
m = first word address of I/O data block; becomes current address as I/O operation progresses
N = "0" for 24- to 12-bit disassembly
N = "1" for straight 12- or 24-bit data transfer
n = last word address of output data block, plus one (minus one, for backward output)
Bits 15 and 16 at P and bits 15, 16, and 20 of P + 1 should be loaded with zeros.

Instruction Description: This instruction transfers a word-addressed block of data consisting of 12-bit bytes or 24-bit words from storage to an external equipment.

With no disassembly, 12 or 24-bit transfer capability depends upon whether a 3306 or 3307 I/O channel is used. If an attempt is made to send a 24-bit word over a 3306 I/O channel, the upper byte will be lost.

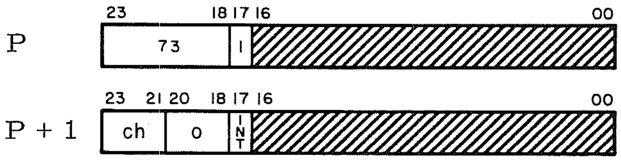
If channel 'ch' is not busy, the buffered I/O operation with storage commences while Main Control performs an RNI at P + 3. Main Control continues executing the main program while the I/O operation occurs simultaneously. If channel 'ch' is initially busy, Main Control performs an RNI at P + 2 and the I/O operation does not occur.



*See text for differences between 3306 and 3307.

Figure 5-11. 76 I/O Operation with Storage

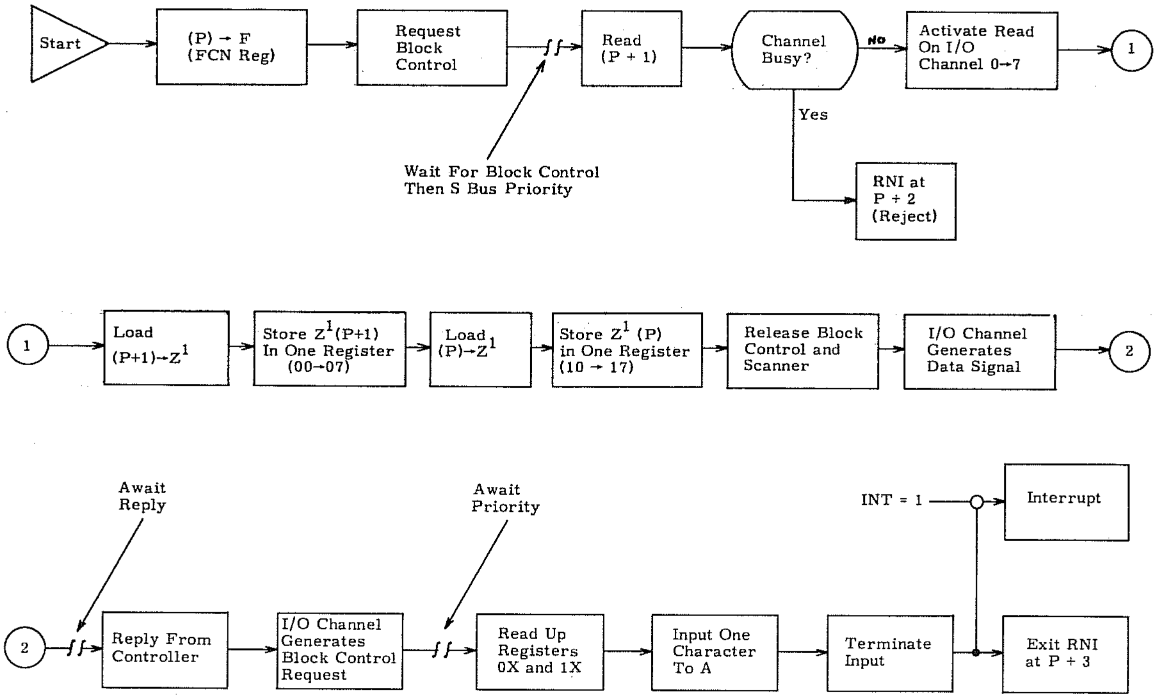
INAC
Input
Character to A



ch = I/O channel designator, 0-7
INT = "1" for interrupt upon completion

Bits 00 through 16 at P and P + 1 should be loaded with zeros.

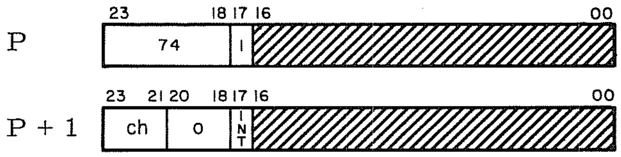
Instruction Description: This instruction transfers a 6-bit character from an external equipment into the lower 6 bits of the A register. (A) are cleared prior to loading, and the upper 18 bits remain cleared. When the I/O operation with A is completed, RNI at P + 3. If channel 'ch' is busy and the operation cannot be performed, RNI at P + 2.



X = I/O Channel Ch (0-7)

Figure 5-12. 73 I/O Operation with A

INAW
Input Word to A



ch = I/O channel designator, 0-7
INT = "1" for interrupt upon completion

Bits 00 through 16 at P and P + 1 should be loaded with zeros.

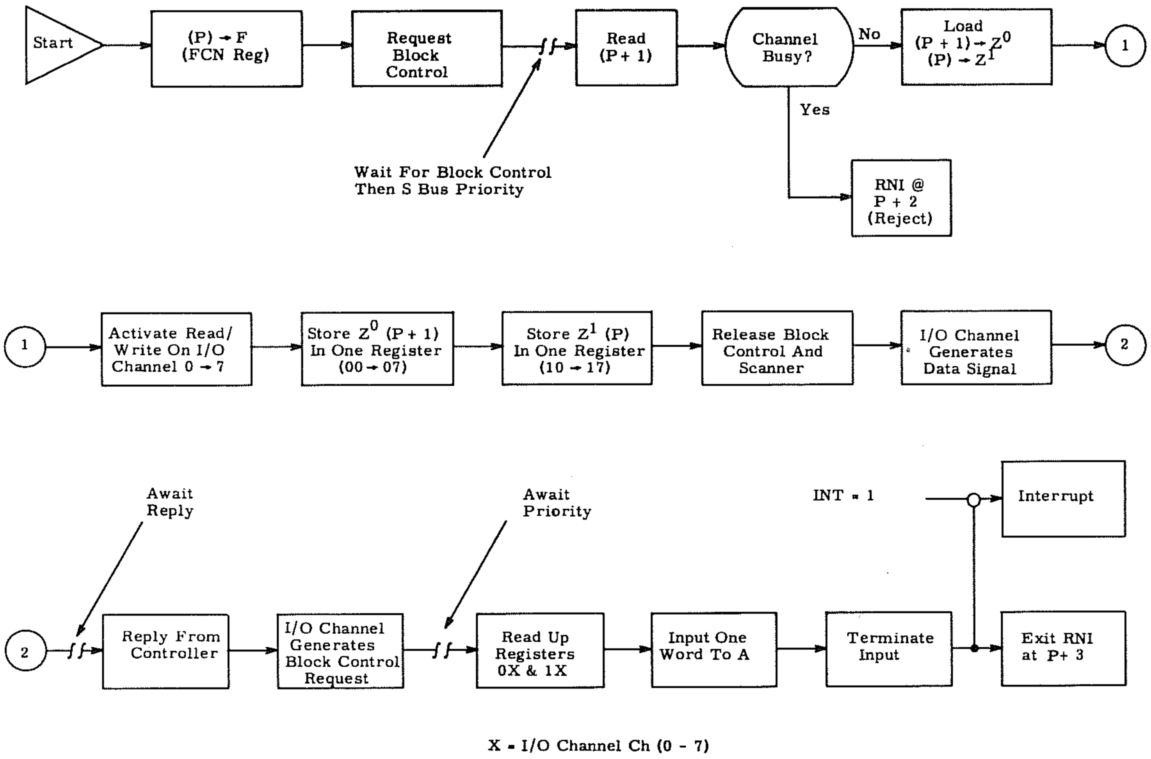
Instruction Description: This instruction transfers a 12-bit byte into the lower 12 bits of A or a 24-bit word into all of A from an external equipment. Transferring 12 or 24 bits depends upon whether a 3306 or 3307 I/O channel is used. (A) is cleared prior to loading and, in the case of a 12-bit input, the upper 12 bits remain cleared. When the I/O operation with A is completed, RNI at P + 3. If channel 'ch' is busy and the operation cannot be performed, RNI at P + 2.

NOTE

Bits 18, 19, and 20 may be all "0's" when a 3306 Data Channel is used. However, when bit 18 = "1", bit 19 = "0" or "1", and bit 20 = "0", this instruction can be used with either a 3306 or 3307 (when the 12- to 24-bit assembly feature is not utilized). This eliminates the need to alter a program when a 3307 is selected in place of a 3306 or vice versa.

If the assembly feature of the 3307 is utilized, bits 18, 19, and 20 take on the following significance:

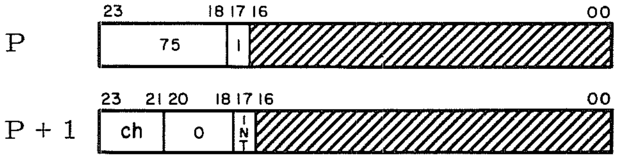
- For 12- to 24-bit forward assembly, bits 18, 19, and 20 = 0
- For 12- to 24-bit backward assembly, bits 18, 19, and 20 = 2



Start
(P) → F
(FCN Reg)
Request Block Control
Read (P + 1)
Channel Busy?
No
Load (P + 1) → Z0 (P) → Z1
1
Wait For Block Control Then S Bus Priority
Yes
RNI @ P + 2 (Reject)
1
Activate Read/ Write On I/O Channel 0 → 7
Store Z0 (P + 1) In One Register (00 → 07)
Store Z1 (P) In One Register (10 → 17)
Release Block Control And Scanner
I/O Channel Generates Data Signal
2
Await Reply
Await Priority
INT = 1
Interrupt
2
Reply From Controller
I/O Channel Generates Block Control Request
Read Up Registers 0X & 1X
Input One Word To A
Terminate Input
Exit RNI at P+ 3
X = I/O Channel Ch (0 - 7)

Figure 5-13. 74 I/O Operation with A

OTAC
Output Character from A



ch = I/O channel designator, 0-7
INT = "1" for interrupt upon completion

Bits 00 through 16 at P and P + 1 should be loaded with zeros.

Instruction Description: This instruction transfers a character from the lower 6 bits of A to an external equipment. The original (A) are retained. When the I/O operation with A is completed, RNI at P + 3. If channel 'ch' is busy and the operation cannot be performed, RNI at P + 2.

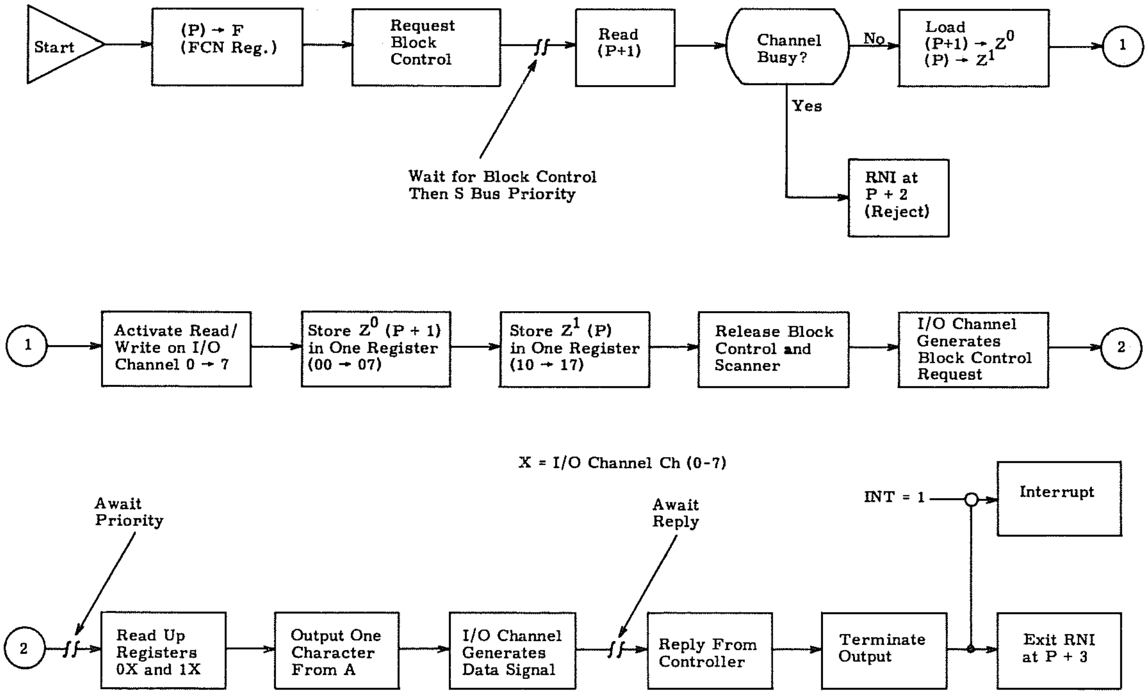
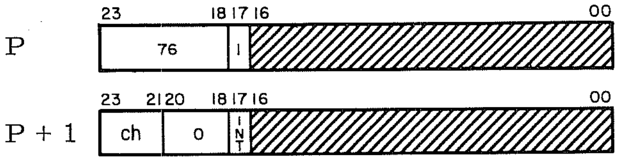


Figure 5-14. 75 I/O Operation with A

OTAW *Output Word from A*



ch = I/O channel designator, 0-7
INT = "1" for interrupt upon completion

Bits 00 through 16 at P and P + 1 should be loaded with zeros.

Instruction Description: This instruction transfers the lower 12 bits of A or (A) to an external equipment, depending upon the type of I/O channel (3306 or 3307) that is used. The original (A) remain unchanged. When the I/O operation with A is completed, RNI at P + 3. If channel 'ch' is busy and the operation cannot be performed, RNI at P + 2.

NOTE

Bits 18, 19, and 20 may be all "0's" when a 3306 Data Channel is used. However, when bit 18 = "1", bit 19 = "0" or "1", and bit 20 = "0", this instruction can be used with either a 3306 or 3307 (when the 24- to 12-bit disassembly feature is not utilized). This eliminates the need to alter a program when a 3307 is selected in place of a 3306, or vice versa.

If the disassembly feature of the 3307 is utilized, bits 18, 19, and 20 take on the following significance:

- For 24- to 12-bit forward disassembly, bits 18, 19, and 20 = 2
- For 24- to 12-bit backward disassembly, bits 18, 19, and 20 = 0

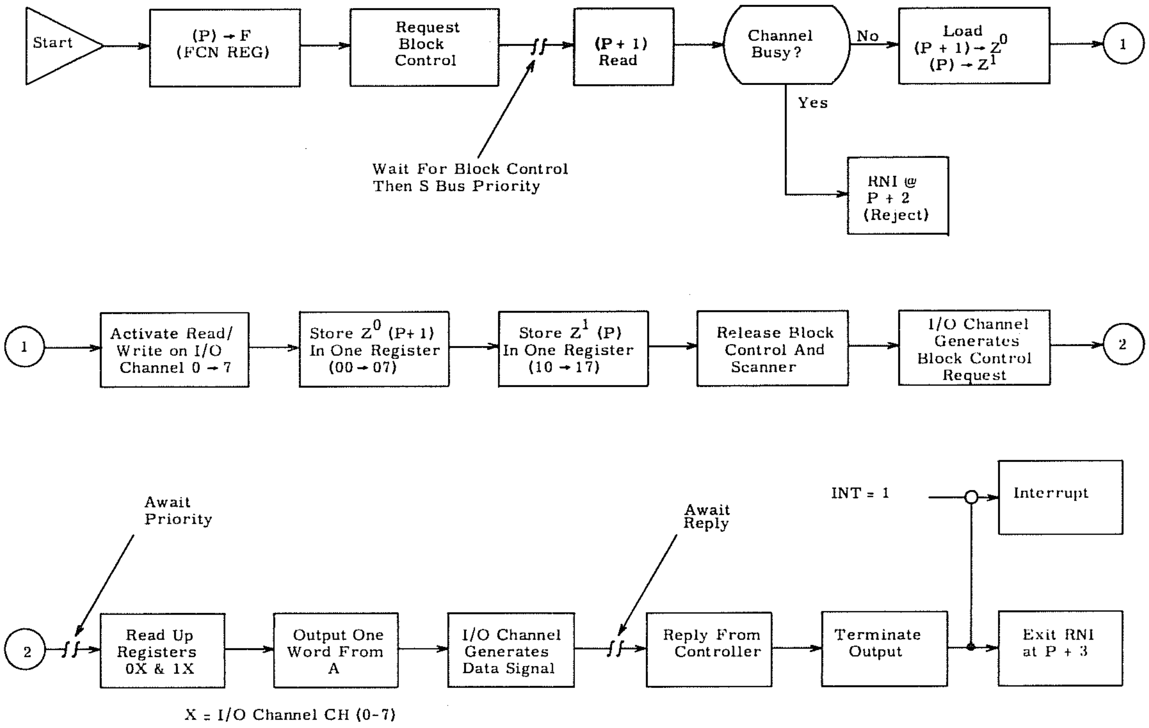
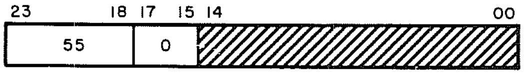


Figure 5-15. 76 I/O Operation with A

Relocation Control Instructions

Operation Field		Address Field	Interpretation
RIS	55		Relocate with (ISR)
ROS	55		Relocate with (OSR)
SBJP	77		Set boundary jump

RIS
Relocate
with Instruction State

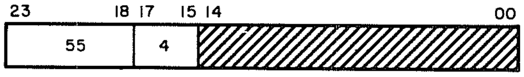


Bits 00 through 14 should be loaded with zeros.

Instruction Description: Clear bit 02 of the Condition register, enabling (ISR) to be used as the upper 3 bits of address for all storage references during Program State.

Comments: A Master Clear produces the same effect as this instruction.

ROS
Relocate with
Operand State

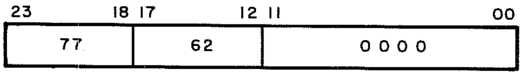


Bits 00 through 14 should be loaded with zeros.

Instruction Description: Set bit 02 of the Condition register, enabling (OSR) to be used as the upper 3 bits of address for all operand references in Executive mode.

Comments: Refer to 'Interrupts During Executive Mode' in Section 4 for additional information.

SBJP
Set
Boundary Jump



Instruction Description: Set a boundary jump condition flag (bit 00 of the condition register).

Comments: When the next jump instruction is executed, the boundary jump condition flag is cleared and the processor enters Program State. The (ISR) are appended to the jump address (m) for the RNI that follows. The (ISR) are used for both the STO and the RNI if the jump instruction is an RTJ (00. 7).

If the computer is interrupted, the condition is cleared as the CRA instruction used in interrupt processing is executed.

Multiprocessing Control Instructions

Operation Field		Address Field	Interpretation
IAPR	77		Interrupt associated processor
SDL	77		Set destructive Load condition

IAPR
Interrupt Associated Processor

23 18	17 12	11 00
77	57	▨

Bits 00 through 11 should be loaded with zeros.

Instruction Description: The processor (computer) executing this instruction sends an interrupt to an associated processor via a special cable. The interrupt remains active in the receiving computer until it is recognized.

SDL
Set Destructive Load

23 18	17 12	11 09	08 00
77	62	4	0 0 0

Instruction Description: Set the destructive load condition flag (bit 01 of the CR).

Comments: After this instruction is executed, the next Load A (LDA) instruction senses the flag and causes the following operations to occur:

1. Load (M) from LDA instruction into A and restore 77777777 into (M).
2. Clear the destructive load condition flag when executing the LDA instruction.

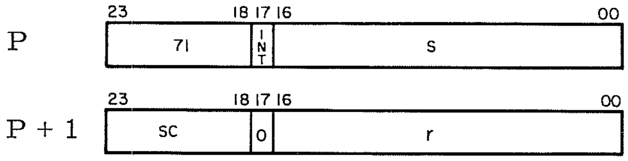
Refer to 'Interrupts During Executive Mode' in Section 4 for additional information.

This instruction is useful in controlling multiple CPU's during multi-processing instructions.

Character Search Instructions

Operation Field	Address Field	Interpretation
SRCE, INT SRCN, INT	71 SC, r, s 71 SC, r, s	Search for character equality Search for character inequality

SRCE Search for Character Equality



INT = "1" for interrupt upon completion
s = last character address of the search block, plus one
SC = 6-bit BCD scan character
r = first (current) character address of the search block

Instruction Description: This instruction initiates a search through a block of character addresses in storage looking for equality with the scan character, SC. If Search/Move control is not busy, the buffered search operation commences while Main Control performs an RNI at P + 3. Main Control continues executing the main program while the search operation occurs simultaneously. If Search/ Move control is initially busy, Main Control performs an RNI at P + 2 and the search operation does not occur.

As a search progresses, r is incremented until the search terminates when either a comparison occurs between the scan character 'SC' and a character in the storage field, or until r=s. If a comparison does occur, the address of the satisfying character may be determined by inspecting r. To do this, transfer the contents of register 20 to A with instruction TMA (53 0 20020).

Register 20 of the register file is reserved for the second instruction word which contains the current character address of the search block. Register 30 is reserved for the first instruction word which contains the last character address of the search block, plus one.

Figure 5-15 is a flow chart of steps that occur during a search operation.

Comments:

Before executing this instruction in Executive mode, the desired program state number (0 through 7) must be loaded into the lower three bits of the A register. This number is then automatically transferred to the upper 3 bits of Register File location 2X for referencing during the buffered operation. The OSR is not used.

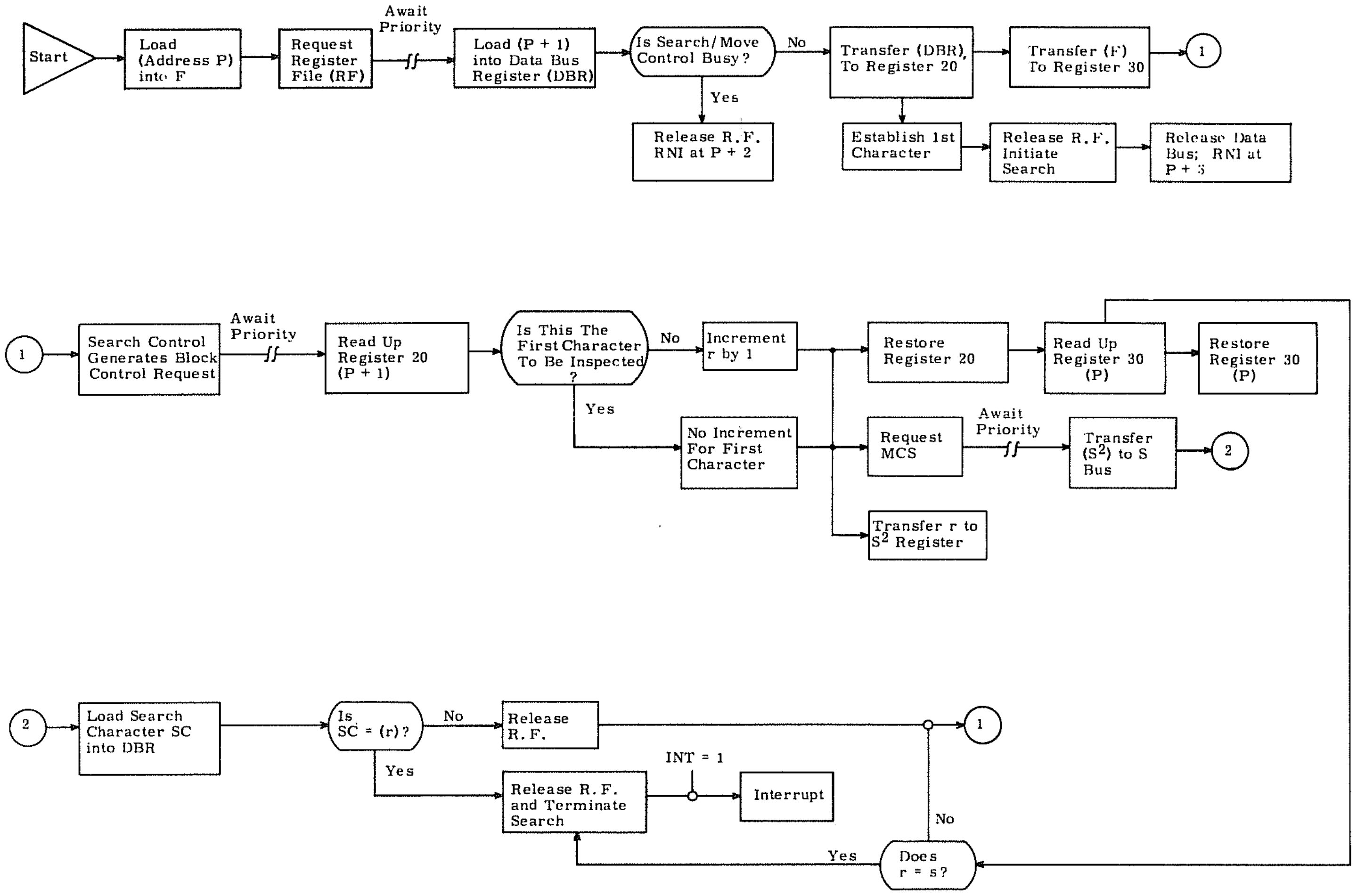
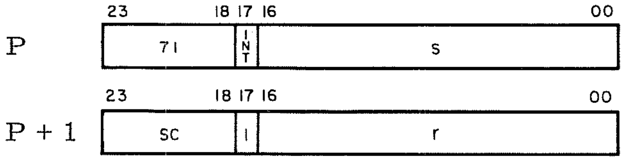

Start
Load (Address P) into F
Request Register File (RF)
Await Priority
Load (P + 1) into Data Bus Register (DBR)
Is Search/Move Control Busy?
No
Yes
Release R.F. RNI at P + 2
Transfer (DBR), To Register 20
Transfer (F) To Register 30
1
Establish 1st Character
Release R.F. Initiate Search
Release Data Bus; RNI at P + 3
1
Search Control Generates Block Control Request
Await Priority
Read Up Register 20 (P + 1)
Is This The First Character To Be Inspected ?
No
Yes
Increment r by 1
No Increment For First Character
Restore Register 20
Read Up Register 30 (P)
Restore Register 30 (P)
Request MCS
Await Priority
Transfer (S2) to S Bus
2
Transfer r to S2 Register
2
Load Search Character SC into DBR
Is SC = (r)?
No
Yes
Release R.F.
1
Release R.F. and Terminate Search
INT = 1
Interrupt
No
Yes
Does r = s?

Figure 5-16. SRCE Operation

SRCN
Search for Character Inequality



INT = "1" for interrupt upon completion
s = last character address of the search block, plus one
SC = 6-bit BCD scan character
r = first (current) character address of the search block

Instruction Description: This instruction initiates a search through a block of character addresses in storage looking for inequality with scan character SC. If Search/Move control is not busy, the buffered search operation occurs simultaneously. If Search/Move control is initially busy, Main Control performs an RNI at P + 2 and the search operation does not occur.

As a search progresses, r is incremented until the search terminates when either an unequal character comparison occurs between the search character SC and a character in storage, or until r = s. If an unequal character comparison does occur, the address of the satisfying character may be determined by inspecting r. To do this, transfer the contents of register 20 to A with instruction TMA (53 0 20020).

Register 20 of the register file is reserved for the second instruction word which contains the current character address of the search block. Register 30 is reserved for the first instruction word which contains the last character address, plus one of the search block.

Figure 5-16 is a flow chart of steps that occur during a search operation.

Comments:

Before executing this instruction in Executive mode, the desired program state number (0 through 7) must be loaded into the lower three bits of the A register. This number is then automatically transferred to the upper 3 bits of Register File location 2X for referencing during the buffered operation.

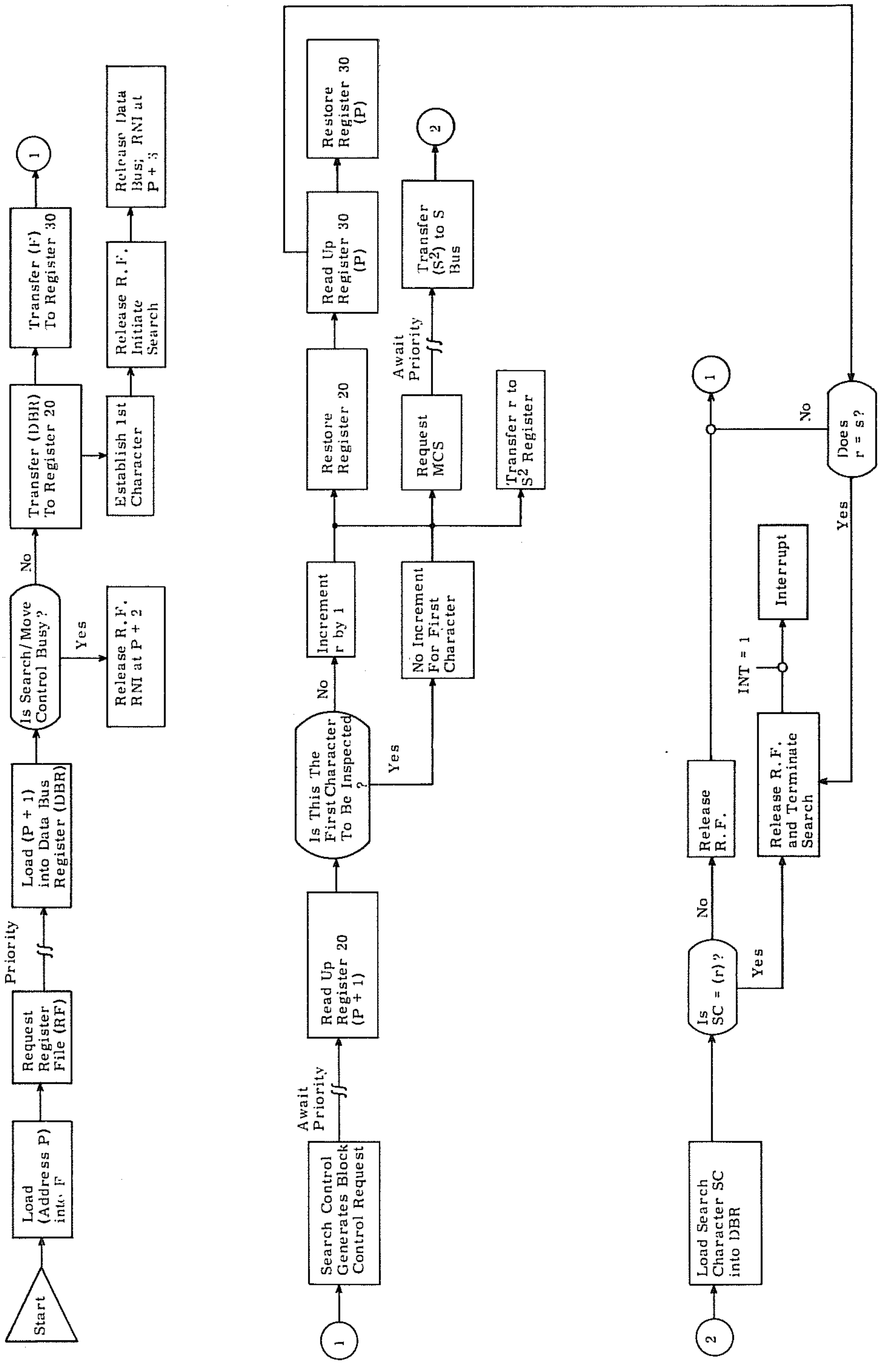
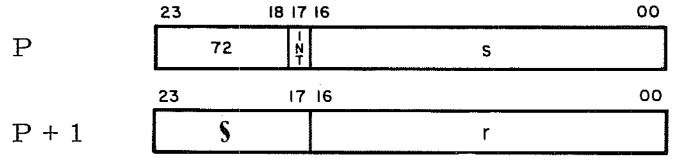

Start
Load (Address P) into F
Request Register File (RF)
Await Priority
Load (P + 1) into Data Bus Register (DBR)
Is Search/Move Control Busy?
Yes
Release R.F. RNI at P + 2
No
Transfer (DBR) To Register 20
Establish 1st Character
Release R.F. Initiate Search
Release Data Bus; RNI at P + 3
Transfer (F) To Register 30
1
1
Search Control Generates Block Control Request
Await Priority
Read Up Register 20 (P + 1)
Is This The First Character To Be Inspected ?
Yes
No Increment For First Character
No
Increment r by 1
Restore Register 20
Request MCS
Transfer r to S2 Register
Read Up Register 30 (P)
Restore Register 30 (P)
Await Priority
Transfer (S2) to S Bus
2
2
Load Search Character into DBR
Is SC = (r)?
No
Release R.F.
1
Yes
Release R.F. and Terminate Search
INT = 1
Interrupt
Does r = s?
No
Yes

Figure 5-17. SRCN Operation

MOVE INSTRUCTION

Operation Field	Address Field	Interpretation
MOVE, INT 72	$\mathcal{S}$, r, s	Move characters from r to s

MOVE
Move $\mathcal{S}$
Characters from r to s



INT = "1" for interrupt upon completion
s = first address of character block destination
$\mathcal{S}$ = field length of data block, 0-177_8* characters
r = first address of character block source

Instruction Description: This instruction moves a block of characters from one area of storage to another. If Search/Move control is not busy, the buffered move operation commences while Main Control performs an RNI at P + 3. Main Control continues executing the main program while the move operation occurs simultaneously. If Search/Move control is initially busy, Main Control performs an RNI at P + 2 and the move operation does not occur.

As a move operation progresses, r and s are incremented and $\mathcal{S}$ (number of characters) is decremented until $\mathcal{S}$ = 0. 128 characters or 32 words may be moved. When bits 00 and 01 of r and s are zero, and the field length is a multiple of four characters, data is moved word by word. This reduces the move time by 75% over a character by character move.

Register 21 of the Register File is reserved for the second instruction word which contains the first address of the character block source. Register 31 is reserved for the first instruction word which contains the first address of the character block destination.

Figure 5-17 is a flow chart of steps that occur during a move operation.

Comments:

Before executing this instruction in Executive mode, the desired program state number (0 through 7) must be loaded into the lower three bits of the A register. This number is then automatically transferred to the upper 3 bits of Register File location 2X for referencing during the buffered operation.

* = 1-177_8 represents a field length of 1 to 127 characters; 0 represents a field length of 128 characters.

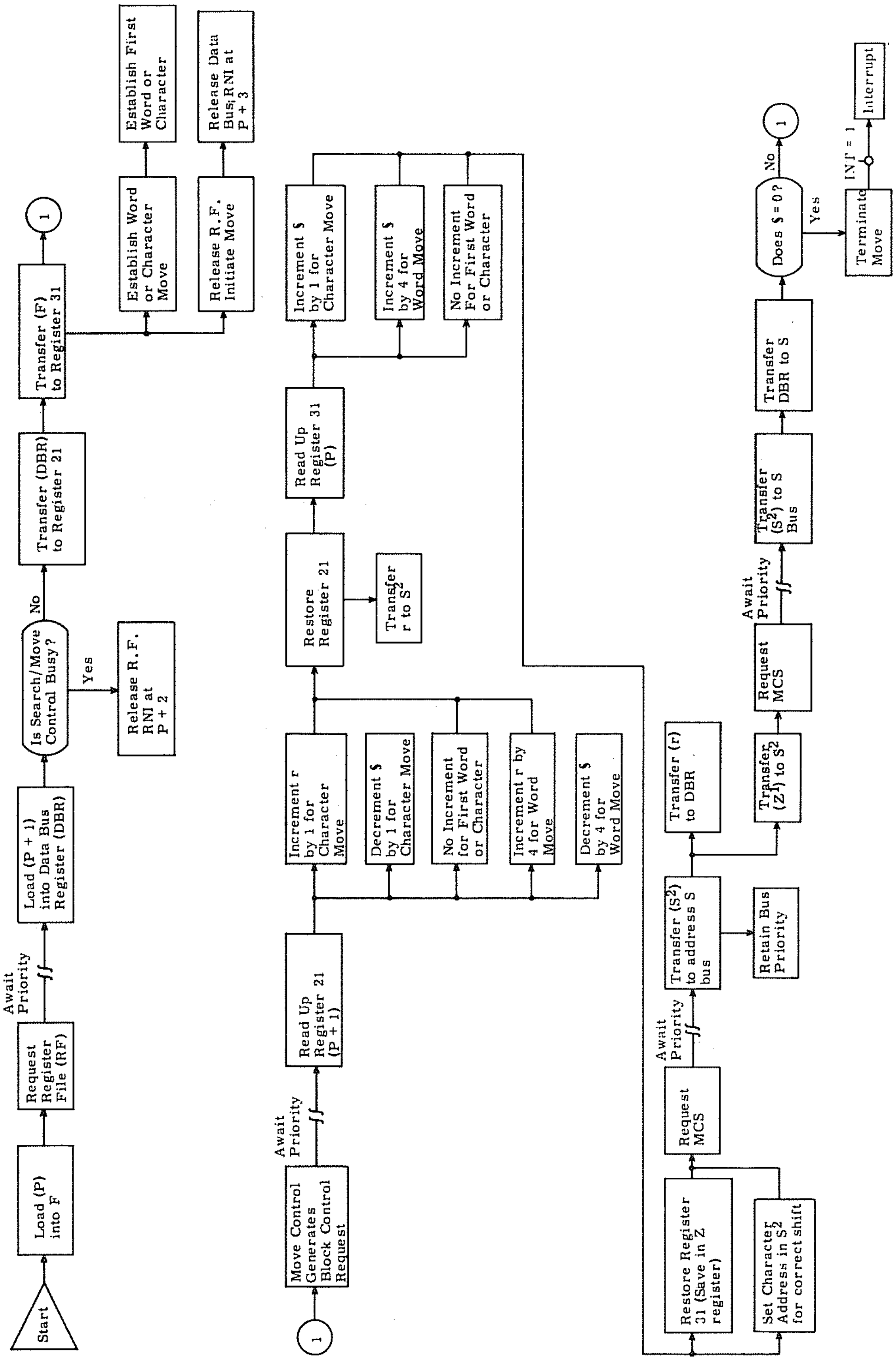

Start
Load (P) into F
Request Register File (RF)
Await Priority
Load (P + 1) into Data Bus Register (DBR)
Is Search/Move Control Busy?
Yes
Release R.F. RNI at P + 2
No
Transfer (DBR) to Register 21
Transfer (F) to Register 31
1
Establish Word or Character Move
Establish First Word or Character
Release R.F. Initiate Move
Release Data Bus, RNI at P + 3
1
Move Control Generates Block Control Request
Await Priority
Read Up Register 21 (P + 1)
Increment r by 1 for Character Move
Decrement $ by 1 for Character Move
No Increment for First Word or Character
Increment r by 4 for Word Move
Decrement $ by 4 for Word Move
Restore Register 21
Transfer r to S²
Read Up Register 31 (P)
Increment $ by 1 for Character Move
Increment $ by 4 for Word Move
No Increment For First Word or Character
Restore Register 31 (Save in Z register)
Set Character Address in S² for correct shift
Request MCS
Await Priority
Transfer (S²) to address S bus
Retain Bus Priority
Transfer (r) to DBR
Transfer (Z¹) to S²
Request MCS
Await Priority
Transfer (S²) to S Bus
Transfer DBR to S
Does $ = 0?
No
1
Yes
Terminate Move
INT = 1
Interrupt

Figure 5-18. Move Instruction

Business Data Processing Instructions

Two somewhat different sets of BDP instructions are available in 3300 systems. The 3312 optional Business Data Processing Unit and 3304-2 Business Data Processor execute the same instruction set. The 3304-2 Business Data Processor executes the second instruction set. The main differences between the two instruction sets are:

1. The 3312 and 3304-2 have instructions for BCD to ASCII conversion (66. 2) and ASCII to BCD conversion. These instructions are not available in the 3504-3.
2. The Compare instructions (67. 3) are quite different in the two BDP's.
3. There are minor differences in several other instructions.

The following table lists all of the BDP instructions and indicates where differences exist between the two instruction sets.

Table 5-10. BDP INSTRUCTION SET

Operation Field		Address Field	Interpretation
MVE	64. 0	r, B_r, ℓ_1, s, B_s, ℓ_2	Move field A to field C
MVE, dc	64. 0	r, B_r, s, B_s, ℓ_2	Move field A to field C with delimiting
MVBF	64. 1	r, B_r, ℓ_1, s, B_s, ℓ_2	Move field A to field C and blank fill
MVZF	64. 2	r, B_r, ℓ_1, s, B_s, ℓ_2	Move field A to field C and zero fill
MVZS	64. 3	r, B_r, ℓ_1, s, B_s, ℓ_2	Move field A to field C and suppress zeros
MVZS, dc	64. 3	r, B_r, s, B_s, ℓ_2	Move field A to field C and suppress zeros with delimiting
ZADM***	67. 2	r, B_r, ℓ_1, s, B_s, ℓ_2	Move field A to field C and add zeros
FRMT	64. 4	r, B_r, ℓ_1, s, B_s, ℓ_2	Move field A to field C and format with commas and decimal point
EDIT	64. 4	r, B_r, ℓ_1, s, B_s, ℓ_2	Move field A to field C and perform complete COBOL edit
SCAN, LR, EQ	65. 0	r, B_r, ℓ_2, sc	Scan field (left to right) for equal condition

*3312/3304-2 Only
**3304-3 Only
***Minor differences between 3312/3304-2 and 3304-3. See instruction descriptions.

Table 5-10. BDP INSTRUCTION SET (Cont'd)

Operation Field		Address Field	Interpretation
SCAN, LR, EQ, dc	65.0	r, B_r, $\$_2$, sc	Scan field (left to right) for equal condition with delimiting
SCAN, LR, NE	65.2	r, B_r, $\$_2$, sc	Scan field (left to right) for unequal condition
SCAN, LR, NE, dc	65.2	r, B_r, $\$_2$, sc	Scan field (left to right) for unequal condition with delimiting
SCAN, RL, EQ	65.1	r, B_r, $\$_2$, sc	Scan field (right to left) for equal condition
SCAN, RL, EQ, dc	65.1	r, B_r, $\$_2$, sc	Scan field (right to left) for equal condition with delimiting
SCAN, RL, NE	65.3	r, B_r, $\$_2$, sc	Scan field (right to left) for unequal condition
SCAN, RL, NE, dc	65.3	r, B_r, $\$_2$, sc	Scan field (right to left) for unequal condition with delimiting
CVDB	66.0	r, B_r, $\$_1$, m, B_m	Convert BCD field to binary field
CVBD	66.1	m, B_m, n, B_n	Convert binary field to BCD field
DTA*	66.2	r, B_r, $\$_2$, m, B_m	Convert BCD field to ASCII field
DTA, dc*	66.2	r, B_r, $\$_2$, M, B_m	Convert BCD field to ASCII delimiting
ATD*	66.3	m, B_m, $\$_2$, s, B_s	Convert ASCII field to BCD field
ATD, dc*	66.3	m, B_m, $\$_2$, s, B_s	Convert ASCII field to BCD, delimiting
PAK	66.4	r, B_r, $\$_2$, m, B_m	Convert 6-bit BCD to 4-bit BCD
UPAK	66.5	m, B_m, s, B_s, $\$_2$	Convert 4-bit BCD to 6-bit BCD
ADM***	67.0	r, B_r, $\$_1$, s, B_s, $\$_2$	Add field A to field C
SBM***	67.1	r, B_r, $\$_1$, s, B_s, $\$_2$	Subtract field A from field C
CMP*	67.3	r, B_r, $\$_1$, s, B_s, $\$_2$	Compare field A to field C
CMP, dc*	67.3	r, B_r, s, B_s, $\$_1$	Compare field A to field C delimiting
CMP**	67.3	r, B_r, $\$_1$, s, B_s, $\$_2$	Collating compare of field A with field C

*3312/3304-2 Only

**3304-3 Only

***Minor differences between 3312/3304-2 and 3304-3. See instruction descriptions.

Table 5-10. BDP INSTRUCTION SET (Cont'd)

Operation Field		Address Field	Interpretation
CMP, n**	67.3	r, B_r, $\S_1$, s, B_s, $\S_2$	Numeric compare of field A with field C
TST	67.4	r, B_r, $\S_1$	Test field A for sign
TSTN	67.4	r, B_r, $\S_1$	Test field A for numeric
LBR***	70.6	m	Load BDP
SBR	70.7	m	Store BDP

*3312/3304-2 Only

**3304-3 Only

***Minor differences between 3312/3304-2 and 3304-3. See instruction descriptions.

NOTE

> All instructions in this group (except LBR and SBR) are unconditionally trapped when the BDP MODE switch is OFF or the optional BDP is not present. LBR and SBR are also trapped if the switch is OFF during Non-Executive mode or Program state of Executive mode. However, during Monitor state, they are No-Ops.

Whenever one of the 64 - 70 instructions is read from memory during execution of a program, the Main Control section signals the BDP section of the Central Processor to assume control for the instruction. Main Control performs all required index and memory operations. Generally, the BDP instructions involve operations with variable length data fields and certain guidelines should be followed while programming.

In those instructions using two variable length data fields, care must be taken in assigning these fields to memory so that overlapping of processed data of the result field over unprocessed data of the source field does not occur. If overlapping occurs the results will be unpredictable.

Interrupts During BDP Instructions

Interrupts are recognized near the end of the first RNI of all instructions. However, after the first RNI of BDP instructions, Main Control continually tests for active interrupt conditions. If a selected interrupt (or Abnormal interrupt) condition becomes active, an Interrupt Stop signal is sent to the BDP section. The BDP relinquishes control after the current character operation is completed. The interrupt is actually recognized as Main Control rereads the instruction at P, or at the address of the next instruction if the current instruction was completed.

The BDP records interrupt recovery conditions (refer to the LBR instruction), and transfers operating information to the B^3 register. If recovery from interrupts is desired, the interrupt routine used must contain a SBR instruction to store the recorded interrupt recovery conditions, and a LBR instruction to return the recovery conditions to the BDP once the interrupt processing is completed. These conditions normally enable a restart to be made from the point of interrupt. Exceptions to the recovery start are: the 66.0 and 66.1 instructions always restart from the beginning if interrupted, and if the interrupt is because of an Illegal Write, the instructions 66.4 and 66.5 also restart from the beginning.

The B^3 index register has the following significance when a BDP instruction is interrupted:

Bits 00 - 11, record the count of the Field C characters processed prior to interrupt.

Bit 12 = "1", if a second pass was in progress.

Bit 13 = "1", if an arithmetic carry was generated on an ADM or SBM instruction during the iteration preceeding interrupt. This is an internal status bit used to enable interrupt recovery and does not indicate an Arithmetic Overflow at instruction completion.

Bit 14 = "1", if a BCD fault occurred.

BDP Condition Register

The BDP Condition register (BCR) is a 2-bit register that is set to indicate conditions existing directly after a business data processing instruction has been executed. The BCR is cleared upon execution of the next BDP instruction. The (BCR) can be sampled to condition jumps to address 'm' by the three jump instructions: JMP, ZRO: JMP, HI: JMP, LOW. Refer to the Jump Instructions group earlier in Section 5 for these instructions and the BCR codes.

Numeric Fields

Six-bit numeric BCD characters consist of a numeric portion (lower 4-bits) and a zone portion (upper 2 bits), the latter of which specifies sign for the character. When considering variable-length numeric fields, the convention followed is to designate the field sign with the sign of the lowest order (right-most) character in the field. This lowest order character is hereafter referred to as the sign character. The zone bits for all other characters in the field must equal zero.

The sign of fields in packed BCD (4-bit) is specified by two special 4-bit sign characters (1010_2 - positive, and 1011_2 - negative) in the lowest order character position.

The significance of zone bits and the numeric portion of 6-bit BCD characters are shown below:

Sign of BCD Field	Relative Bit Positions	
	6	5
+	0	0
+	0	1
-	1	0
+	1	1

NUMERIC BCD CODES (Lower 4 Bits)

Decimal Number	BCD Character Relative Bit Positions			
	4	3	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

EXAMPLE: Following is an example illustrating execution of a MVZF (64.2, D = 0)* instruction:

P 64000202
P + 1 = 27003000 $(B^1) = 00200$
P + 2 = 00140017 $(B^2) = 01000$

Analysis:

1. The unmodified character address 'r' is 00202.
2. $B_r = 3$, requiring (B^1) be added to r. If $(B^1) = 00200$ then R = 00402 which equals word address 00100 character position 2. This is the true address of the highest order character in field A.
3. $B_s = 2$, requiring (B^2) be added to the unmodified character address 's', 03000. If $(B^2) = 01000$, then S = 04000.
4. The length of the A field is 14_8 characters and the alloted length of the C field is 17_8 characters. The last three characters of field C will be filled with zeros. The last character of field C (a zero) will also contain the sign of the field.

(continued on next page)

*This instruction moves a string of 6-bit characters from field A to field C. If field C is longer, its remainder is filled with zeros. Refer to the BDP instruction descriptions later in this section for a more thorough explanation.

Character Positions

'A' Field Word Addresses	0	1	2	3
0 0 1 0 0	10	06	04	03
0 0 1 0 1	01	11	02	04
0 0 1 0 2	06	03	11	10
0 0 1 0 3	05	51	04	07

Highest order character in Field A

Operand specified in Field Aequals -431924639859

Lowest Order Character in Field A

$$\overbrace{5\qquad 1}$$

$$\underbrace{1\ 0}\ \underbrace{1\ 0\ 0\ 1}$$

Indicates a negative field

Indicates character is $11_8 = 9_{10}$

The operation proceeds as follows:

First character stored is highest order character at word address 01000, character position zero.

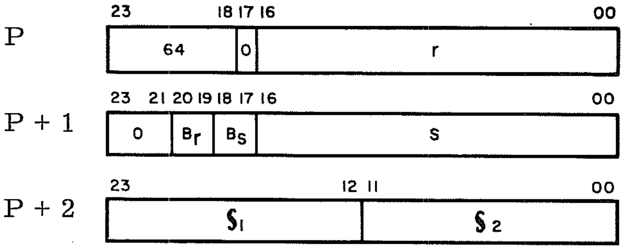
'C' Field Word Addresses				
0 0 7 7 7	03	01	00	02
0 1 0 0 0	04	03	01	11
0 1 0 0 1	02	04	06	03
0 1 0 0 2	11	10	05	11
0 1 0 0 3	00	00	40	07

Sign is no longer stored in this character as the lowest order character of Field C is now character position 2 of word address 01003.

Lowest order character in Field C contains the sign of Field C (minus) and a zero character.

Operand now stored in Field C equals -431924639859000

MVE
Move Field A to Field C



r = unmodified address of the highest order character in field A. R = r +[B_r]

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

s = unmodified address of the highest order character in field C. S = s + [B_s]

B_s = index register flag for field C (same bit functions as B_r)

$§_1$ = number of characters in field A to be moved

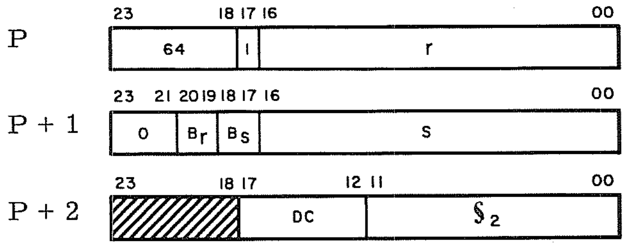
$§_2$ = number of available character positions in field C

Instruction Description: Move a field of up to 4095 6-bit alphanumeric characters from field A to field C, left to right. If field lengths are unequal, the length of the shorter field terminates the move and the remainder of the longer field is not moved or changed.

Comments: The BDP Condition register is set to the sign of field A if the sign character is moved. It is set to 00_2 for a positive sign (or no sign transferred) and to 10_2 for a negative sign.

Index register B^3 (bits 0-11) records the number of characters moved.

MVE, DC
Move Field A to Field C, Delimited



r = unmodified address of the highest order character in field A. $R = r + [B_r]$

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

s = unmodified address of the highest order character in field C. $S = s + [B_s]$

B_s = index register flag for field C (same bit functions as B_r)

$\S_2$ = number of available character positions in both field A and field C

DC = 6-bit delimiting character compared against the characters in field A

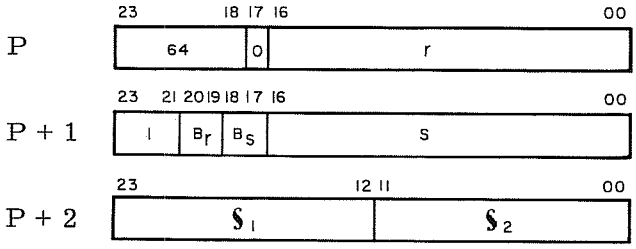
Bits 18 through 23 of P + 2 should be loaded with zeros.

Instruction Description: Move a field of up to 4095 6-bit alphanumeric characters from field A to field C, left to right.

Comments: The length of field C, $\S_2$, terminates the move operation. If the delimiting character is recognized at any time during the character move, the operation is terminated after the delimit character has been moved.

The BDP Condition register is not used for this instruction (always set to 00_2).

MVBF
Move Field A to Field C and Blank Fill



r = unmodified address of the highest order character in field A. R = r + [B_r]

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing
s = unmodified address of the highest order character in field C. S = s + [B_s]

B_s = index register flag for field C (same bit functions as B_r)
$§_1$ = number of characters in field A to be moved
$§_2$ = number of available character positions in field C

Instruction Description: Move a field of up to 4095 6-bit alphanumeric characters from ~~field A to~~ field C, left to right. If field lengths are unequal, the length of the shorter field terminates the move. If field C is longer, its remainder is filled with blanks. The sign is contained in character last moved from field A.

Comments: The BDP Condition register is set to the sign of field A if the sign character is moved. It is set to 00_2 for a positive sign (or no sign transferred) and to 10_2 for a negative sign.

Index register B^3 (bits 0-11) records the field C character count as the instruction progresses.

MVZF
Move Field A to Field C and Zero Fill

	23–18	17	16–00
P	64	0	r

	23–21	20–19	18–17	16–00
P + 1	2	B_r	B_s	s

	23–12	11–00
P + 2	$\S_1$	$\S_2$

r = unmodified address of the highest order character in field A. $R = r + [B_r]$

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

s = unmodified address of the highest order character in field C. $S = s + [B_s]$

B_s = index register flag for field C (same bit functions as B_r)

$\S_1$ = number of characters in field A to be moved

$\S_2$ = number of available character positions in field C

Instruction Description: Move a field of up to 4095 6-bit BCD numeric characters from field A to field C, left to right. If field lengths are unequal, the shorter field terminates the move. If field C is longer, its remainder is filled with zeros.

Comments: The zone bits of all field A characters are forced to zero when moved to field C, as is any field A character containing a 12_8 - 17_8 code.

The sign from field A is always transferred to the BDP Condition register and to the lowest order character in field C (which may be a zero-filled character), even if $\S_1 > \S_2$.

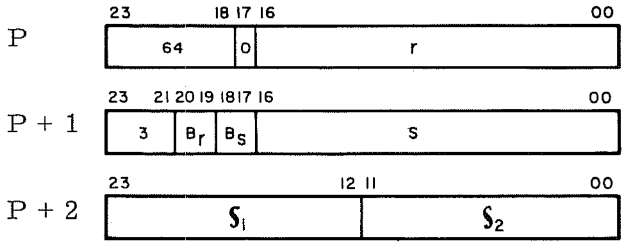
Index register B^3 (bits 0-11) records the field C character count as the instruction progresses.

A BCD fault is generated if one of the following conditions occur:

1. Zone portion of any character in field A (except sign character) does not equal zero.
2. Numeric portion of any character in field A contains a BCD code greater than 11_8, except the sign character where a 12_8 code is legal.
3. The sign character contains a 72_8 code.

Operation continues despite any BCD fault.

MVZS
Move Field A to Field C and Zero Suppress



r = unmodified address of the highest order character in field A. $R = r + [B_r]$

B_r = index register flag for field A
If $Br = 1$ or 3, use index register B^1
If $B_r = 2$, use index register B^2
If $B_r = 0$, no indexing

s = unmodified address of the highest order character in field C. $S = s + [B_s]$

B_s = index register flag for field C (same bit functions as B_r)

$\mathcal{S}_1$ = number of characters in field A to be moved

$\mathcal{S}_2$ = number of available character position in field C

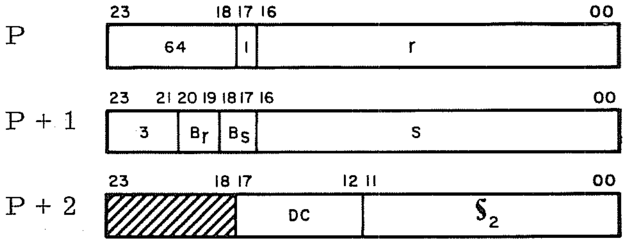
Instruction Description: Move a field of up to 4095 6-bit alphanumeric characters in field A to field C, left to right, and replace all leading zeros occurring in field A with blanks in field C. If field lengths are unequal, the shorter field terminates the move and the remainder of the longer field is not moved or changed.

Comments: If field A is longer than field C, the sign bits in field C may be invalid since sign bits are not checked or modified and consist of the zone bits of the last character moved.

The BDP Condition register contains the sign of the A field if the sign is moved (i.e., $\mathcal{S}_1 \lesssim \mathcal{S}_2$).

Index register B^3 (bits 0-11) records the field C character count as the instruction progresses.

MVZS, DC
Move Field A to Field C and Zero Suppress, Delimited



r = unmodified address of the highest order character in field A. $R = r + [B_r]$

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

s = unmodified address of the highest order character in field C. $S = s + [B_s]$

B_s = index register flag for field C (same bit functions as B_r)

$§_2$ = number of available character positions in both field A and field C

DC = 6-bit delimiting character compared against the characters in field A

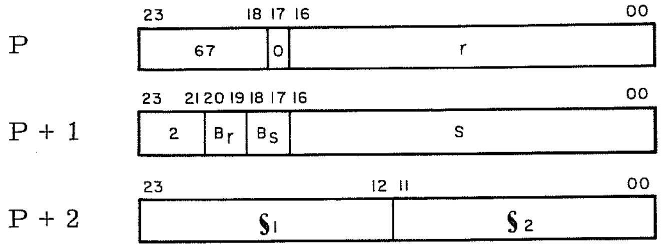
Bits 18 through 23 of P + 2 should be loaded with zeros.

Instruction Description: Move a field of up to 4095 6-bit alphanumeric characters from field A to field C, left to right, and replace all leading zeros occurring in field A with blanks in field C. The length of field C, $§_2$, terminates the move operation. If the delimiting character is recognized at any time during the character move, the operation is terminated after the delimit character has been moved. The remainder of field C remains unchanged.

Comments: The sign of the newly formed C field may be inavlid since sign bits are not checked or modified and consist of the zone bits of the last character moved.

The BDP Condition register is not used (always set to 00_2).

ZADM
Zero and Add Move



r = unmodified address of the lowest order character in field A. R = r + [B_r]

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

s = unmodified address of the lowest order character in field C. S = s + [B_s]

B_s = index register flag for field C (same bit functions as B_r)

$\S_1$ = number of characters in field A to be moved

$\S_2$ = number of available character positions in field C

Instruction Description: Move a field of up to 4095 6-bit BCD numeric characters from field A to field C, right to left. If field lengths are unequal, the shorter field terminates the move. If field C is longer, the remainder of field C is filled with zeros.

Comments: The algebraic sign of field A is obtained from the zone bits of the lowest order charcter in field A. (10_2 indicates a negative field, all other combinations indicate a positive field.) This sign is stored in the sign character of field C. The BDP Condition register is also set to the sign of field A.

Any field A character with a 12_8 - 17_8 code is forced to zero when moved.
A sign character of 40_8 in the result field is converted to 52_8, the code for magnetic tape character negative zero. A 60_8 code encountered in either field is treated as zero. *

Index register B^3 (bits 0-11) records the field C character count as the instruction progresses.

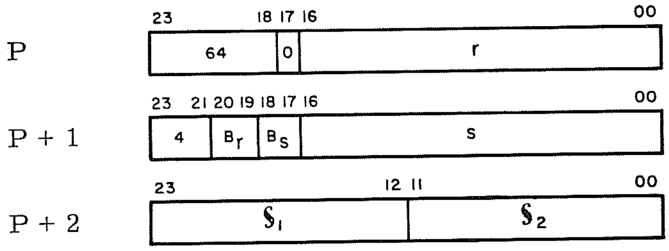
A BCD fault is generated if one of the following conditions occur:

1. Zone portion of any character in field A (except sign character) does not equal zero.
2. Numeric portion of any character in field A contains a BCD code greater than 11_8, except the sign character where a 12_8 code is legal.
3. The sign character contains a 72_8 code.

Operation continues despite any BCD fault.

*Applicable to 3304-3 only.

FRMT
Move Field A and Format in Field C



r = unmodified address of the highest order character in field A. R = r + [B_r]

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing.

s = unmodified address of the highest order character in field C. S = s + [B_s]

B_s = index register flag for field C (same bit functions as B_r)

$\S_1$ = number of characters in field A to be edited. (Values must be 2, 5, 10_8, 13_8, 16_8, etc.)

$\S_2$ = number of characters in field C. Values of $\S_2$ must be 3, 6, 12_8, 16_8, 22_8, etc. and include decimal point and commas).

Instruction Description: Move the numeric characters in field A from left to right into field C, replacing leading zeros with blanks and inserting a comma after every three characters moved. A decimal point is inserted in the third lowest order position of the C field.

Comments: Leading zeros in field A, together with normally inserted commas, are suppressed and replaced by blanks in field C. Zero suppression terminates when a non-zero character is encountered in field A or immediately before the decimal point is inserted.

The sign of field A is recorded in the BDP Condition register if the sign character is moved but does not appear in the resultant C field.

Index register B^3 (bits 0-11) records the C field character count as the instruction progresses.

A BCD fault is generated if one of the following conditions occur:

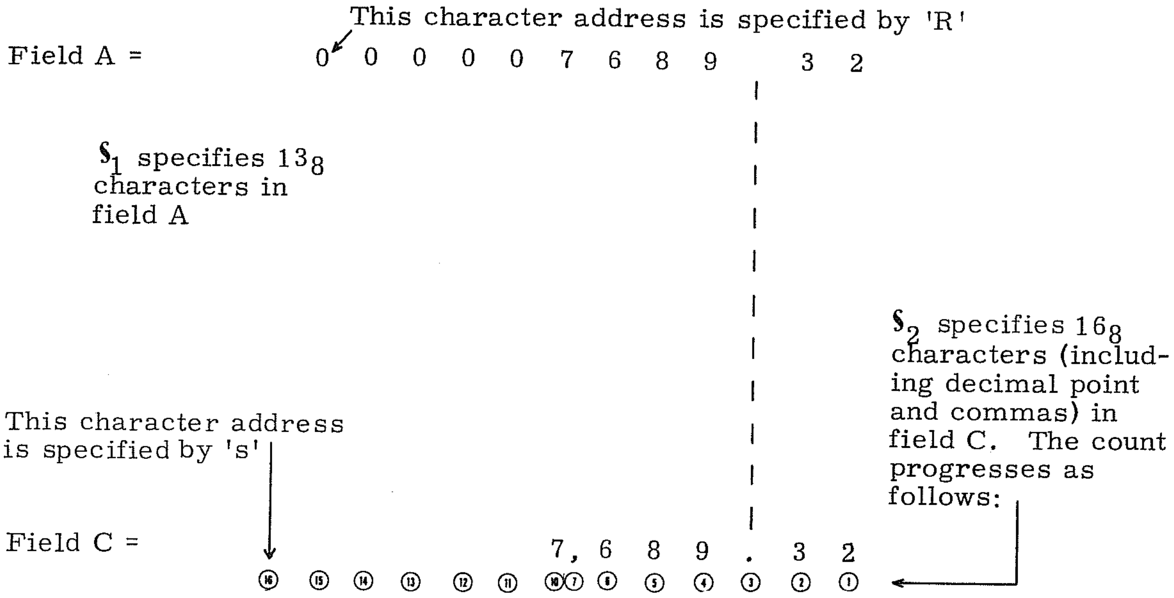
1. Zone portion (upper 2 bits) of any character in field A (except sign character) does not equal zero.
2. Numeric portion (lower 4 bits) of any character in field A contains a BCD code greater than 11_8 except the sign character, where a 12_8 code is legal.
3. The lowest order character (sign character) contains a 72_8 code.
4. Incorrect field length specified - a) the two fields are not aligned (Ex: $\S_1$ = 2 and $\S_2$ = 6), b) illegal count used (Ex: $\S_2 \neq 3, 6, 12_8, 22_8$, etc.).

Operation continues despite any BCD fault.

EXAMPLE:

$S_1 = 13_8$

$S_2 = 16_8$



This character address is specified by 'R'

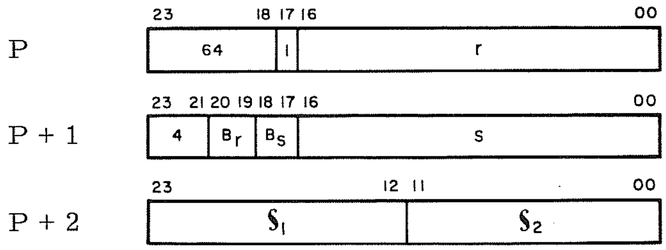
S_1 specifies 13_8 characters in field A

This character address is specified by 's'

S_2 specifies 16_8 characters (including decimal point and commas) in field C. The count progresses as follows:

The leading zeros are removed, leaving six blanks in field C.

EDIT
Move Field A and Edit in Field C



r = unmodified address of the highest order character in field A. R = r + [B_r]

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

s = unmodified address of the highest order character in field C. S = s + [B_s]

B_s = index register flag for field C (same bit functions as B_r)

S_1 = number of characters in field A to be edited

S_2 = number of characters in field C

Instruction Description: Perform an edit on the numeric characters in field A proceeding from left to right. The editing is performed character for character with respect to the COBOL type editing characters in field C. The resulting edited field is stored in Field C.

Comments: Programming consideration must be given to aligning the characters in field A with the proper editing characters in field C. The COBOL type editing characters used in field C and applicable to this instruction are listed here with their descriptions. Any other character in the C field is recognized as a 9.

The BDP Condition register is set to the sign of field if the sign character is moved. The conditions for a BCD Fault are the same as for FRMT, except condition 4 does not apply.

Index register B^3 (bits 0-11) records the field C character count as the instruction progresses.

Editing examples are listed following the character descriptions:

EDITING CHARACTERS

9 $ + - . , 0 B CR DB Z * /

- Direct Characters:

  9 When character 9 appears in the C field, it is replaced by the corresponding character in the A field.

- Insertion Characters:

When an insertion character is specified in the C field, it remains in that character position in the edited field. The insertion characters are:

$ + - . , 0 B CR DB /

$ When a single dollar sign is specified as the left-most symbol in a C field, it appears as the left-most character in the edited field. This character is included in the character count (S_2) of the edited field.

+ When a plus sign is specified as the first or last symbol of a C field, a plus sign is inserted in the indicated character position of the edited data, provided the field of data is positive or is unsigned. If the data is negative, a minus is inserted in the indicated character position. The sign is included in the character count (S_2) of the edited field.

- When a minus sign is specified as the first or last symbol of a C field, a minus sign is inserted in the indicated character position of the edited data, provided the field of data is negative. If the data is not negative, a blank is inserted in the indicated character position. The sign or blank is included in the character count (S_2) of the edited field.

. (decimal point) This character is used in a C field to represent an actual decimal point as opposed to an assumed decimal point. When used, a decimal point appears in the edited data as a character in the same character position as it appears in the C field and it is included in the character count of the edited field. A picture of a report item can never contain more than one decimal point, actual or assumed.

, When a comma is used in a C field, a comma is inserted in the corresponding character position of the edited data. It is included in the character count (S_2) of the edited field.

0 (zero) When a zero is used in a C field, a zero is inserted in the corresponding character position in the edited field. It is included in the character count (S_2) of the edited field.

B When the character, B, is used in a C field, a blank is inserted in the corresponding character position in the edited field. It is included in the character count (S_2) of the edited field.

CR The combined characters (CR) represent a credit in accounting operations and may be specified only at the right end of a C field. The symbol is inserted in the last two character positions of the edited field, provided the value of the data is negative. If the data is positive or unsigned these last two character positions are set to blanks. Since this symbol always results in two characters (CR or blanks) it is included as two characters in the character count (S_2) of the edited field.

DB The combined characters (DB) represent a debit and may be specified only at the right end of a C field. It has the same results as the credit symbol, using DB or blanks.

/ (slash) When a slash (/) is used in a C field, a slash is inserted in the corresponding character position in the edited field. It is included in the character count ($\S_2$) of the edited field.

- Replacement Characters:

A replacement character in a C field suppresses leading zeros in data and replaces them with other characters in the edited data. The replacement characters are:

Z * $ + -

Only one replacement character may be used in a picture, although a Z or asterisk (*) may be used with any of the insertion characters including:

$ + -

Z One Z character is specified at the left end of a C field for each leading zero in the A field that requires suppression and replacement by a blank in the edited field. Z's may be preceded by one of the three insertion characters $ + or - and interspersed with the four insertion characters . , 0 or B. Whether these insertion characters affect the result of the editing process depends on the nature of the data.

Suppressing leading zeros and inserting blanks ceases when one of the following conditions exists:

1. When the number of suppressed zeros equals the number of Z's specified in the C field.
2. When the first non-zero digit character in the A field is encountered.
3. When the position in the C field is reached where a decimal point insertion is specified. Zero suppression and blank replacement cannot continue beyond a decimal point, hence, a decimal point is never followed by blanks in an edited field.

   - If either a $ + - is specified before Z's, the character is inserted in the edited data regardless of leading zero suppression.
   - If either a comma, B, or 0 is encountered in the edit field before zero suppression has terminated, the character is not inserted in the edited data, but is suppressed and a blank is inserted.
   - In the special case where the edited data has a value of zero, the entire edited data is replaced by blanks if a 9 does not appear in the edit picture. This special rule overrides the condition that zero suppression terminates when a decimal point is encountered. If one of the three insertion characters $ + - is specified but the value of the edited field is zero, a blank is inserted instead of the insertion character.

* The asterisk (*) is specified in the same way and with the same results as the character Z, except that suppressed characters are replaced by asterisks instead of blanks. The rules for the Z character apply also to the use of the asterisk.

$ If one more dollar sign ($) than the number of leading zeros to be suppressed is specified at the left end of a C field, this dollar sign acts as an insertion character. Each of the other dollar signs corresponds to a leading zero to be suppressed. This use of the dollar sign has the same results as described for the Z character, except that the dollar sign is inserted directly preceding the first non-suppressed character. A dollar sign used in this way as a replacement character is known as a floating dollar sign as it virtually "floats" through all of the suppressed characters.

If one or more floating dollar signs are specified in a C field, the edited data always contains a dollar sign whether or not any suppression occurs since one of the dollar signs is an insertion character. Each dollar sign specified in a picture (including the insertion $) is included in determining the character count (S_2) of the edited field.

+ When a plus sign (+) is used as a replacement character, it functions as a floating plus sign and is specified in the C field one more time than the number of leading zeros to be suppressed. Its function is the same as the floating dollar sign, with the following exception:

> If the A field data is positive or unsigned a plus sign is inserted in the character position directly preceding the first non-suppressed character. If the A field data is negative, a minus sign is inserted in this character position in the edited field.

- When a minus sign is used as a replacement character it functions as a floating minus sign and is specified in the C field one more time than the number of leading zeros to be suppressed. Its function is the same as the floating dollar sign and floating plus sign with the following exception:

> If the A field data is negative, a minus sign is inserted in the character position directly preceding the first non-suppressed character. If the value of the A field data is positive or unsigned, a blank is inserted in this position in the edited data instead of a minus sign.

TABLE 5-11. EDITING EXAMPLES

Field A Data	Field C Editing Data	Resultant Field C Edited Data
4 8	$99	$ 4 8
4 8 3 4	$99.99	$ 4 8 . 3 4
4 8 3 4	9,999	4 , 8 3 4
2 9 2	+999	+ 2 9 2
2 9 $\overline{2}$	+999	- 2 9 2
2 9 $\overline{2}$	999-	2 9 2 -
2 9 2	999-	2 9 2 Δ
2 4 3 2 1	$BB999.99	$ Δ Δ 2 4 3 . 2 1
2 4 3 2 1	$00999.99	$ 0 0 2 4 3 . 2 1
1 1 3 $\overline{4}$	99.99CR	1 1 . 3 4 C R
1 1 3 4	99.99CR	1 1 . 3 4 Δ Δ
2 3 7 $\overline{6}$	99.99DB	2 3 . 7 6 D B
2 3 7 6	99.99DB	2 3 . 7 6 Δ Δ
0 0 9 2 3	ZZ999	Δ Δ 9 2 3
0 0 9 2 3	ZZZ99	Δ Δ 9 2 3
0 0 0 0 0 0	ZZZZ.ZZ	Δ Δ Δ Δ Δ Δ Δ
0 0 9 2 3	$***.99	$ * * 9 . 2 3
0 0 0 8 2 4	$$$$9.99	Δ Δ Δ $ 8 . 2 4
0 0 5 2 $\overline{6}$	---9.99	Δ Δ - 5 . 2 6
3 2 6 5	$$$.99	$ 3 2 . 6 5

UPPERMOST CHARACTER POSITIONS

THE SIGN IS CONTAINED IN THE LOWEST ORDER CHARACTER. ONLY MINUS SIGNS ARE SHOWN HERE

THE Δ FIGURE INDICATES A BLANK POSITION

Field A Data	Field C Editing Data	Resultant Field C Edited Data
00001234	$ZZZ,ZZZ.99	$ΔΔΔΔΔ12.34
00123456	$***,**9.99	$**1,234.56
12345634	$***,***.99	$123,456.34
00001$\overline{2}$	-ZZZ,ZZZ	-ΔΔΔΔΔ12
1234562$\overline{4}$	$ZZZ,ZZ9.99CR	$123,456.24CR
00123400	$$$$,$$9.99	ΔΔ$1,234.00
000000	$$$$,$$$.99	ΔΔΔΔΔΔΔΔΔΔΔ
0000125$\overline{6}$	----,---.99DB	ΔΔΔΔΔ-12.56DB

1. Only one replacement character of the set Z * $ + and - can be used within a single editing C field even though it may be specified more than once.

2. If one of the replacement characters Z or * is used with one of the insertion characters $ + or -, the plus sign or the minus sign may be specified as either the leftmost or rightmost character in the editing C field.

3. A plus sign and a minus sign may not be included in the same editing C field.

4. A leftmost plus sign and a dollar sign may not be included in the same editing C field.

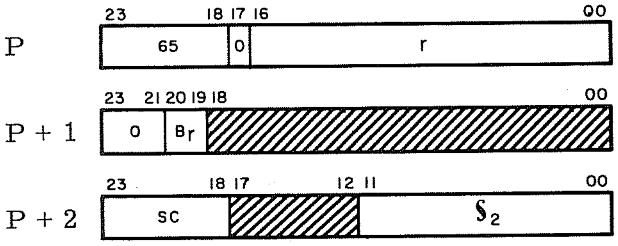
5. A leftmost minus sign and a dollar sign may not be included in the same editing C field.

6. The character 9 may not be specified to the left of a replacement character.

7. Symbols which may appear only once are: decimal point, CR, and DB.

8. The decimal point may not be the rightmost character in an editing C field.

SCAN, LR, EQ *Search Field A Left to Right for Equality*



r = unmodified address of the highest order character in field A. R = r + [B_r]

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

SC = 6-bit scan character compared against characters in field A

$§_2$ = number of characters to be searched

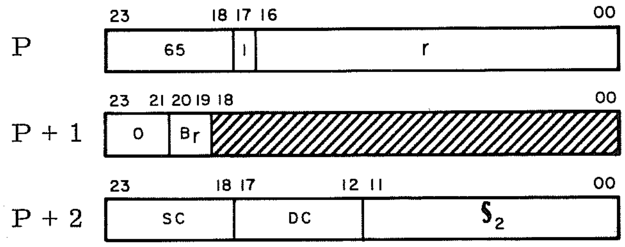
Bits 00 through 18 of P + 1 and bits 12 through 17 of P + 2 should be loaded with zeros.

Instruction Description: Search field A from left to right beginning with the 6-bit character at location R and RNI at P + 4 if a character is found that is equal to the scan character, SC. If a character is not found that equals the SC after the entire field defined by $§_2$ has been searched, RNI at P + 3.

Comments: If a character comparison occurs during the search, the number of searched characters is transferred to the lower 12 bits of B^3. If an unsuccessful search is made, then (B^3) = $§_2$. The upper 3 bits of B^3 are of no consequence in this instruction. BCD codes of 12_8, 32_8, and 52_8 do not compare equal to zero for this instruction.

The BDP Condition register is not used (always set to 00_2).

SCAN, LR, EQ, DC
Search Field A Left to Right for Equality, Delimited



r = unmodified address of the highest order character in field A. R = r + [B_r]

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

SC = 6-bit scan character compared against characters in field A.

$\mathcal{S}_2$ = number of characters to be searched.

DC = 6-bit delimiting character compared against characters in field A

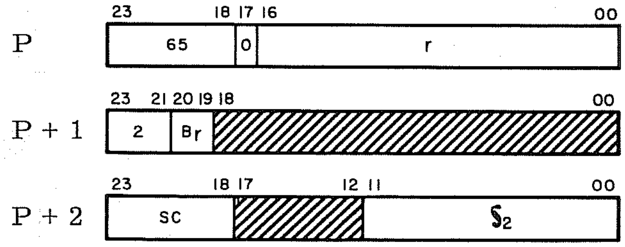
Bits 00 through 18 of P + 1 should be loaded with zeros.

Instruction Description: Search field A from left to right beginning with the 6-bit character at location R and RNI at P + 4 if a character is found that is equal to the scan character, SC. If a character is not found that equals the SC after the entire field defined by $\mathcal{S}_2$ has been searched or if a character is found that equals the delimiting character, DC, RNI at P+3. (If SC equals DC and comparison with field A occurs, RNI at P + 4).

Comments: If a character comparison is found during the search, the number of characters searched is transferred to the lower 12 bits of B^3. If an unsuccessful search is made, then (B^3) = $\mathcal{S}_2$. The upper 3-bits of B^3 are of no consequence in this instruction. BCD codes of 12_8, 32_8, and 52_8 do not compare equal to zero for this instruction.

The BDP Condition register is not used (always set to 00_2).

SCAN, LR, NE
Search Field A Left to Right for Inequality



r = unmodified address of the highest order character in field A. $R = r + [B_r]$

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

SC = scan character which is compared against characters in field A.

$\S_2$ = number of characters to be searched

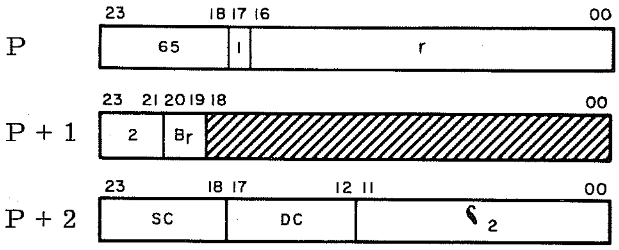
Bits 00 through 18 of P + 1 and bits 12 through 17 of P+ 2 should be loaded with zeros.

Instruction Description: Search field A from left to right beginning with the 6-bit character at location R, and RNI at P + 4 if a character is found that is not equal to the scan character ,SC. If a character is not found that is not equal to the SC after the entire field defined by $\S_2$ has been searched, RNI at P + 3.

Comments: If an unequal character comparison is found during the search, the number of characters searched is transferred to the lower 12 bits of B^3. If all characters searched are equal to the SC, then $(B^3) = \S_2$. The upper 3 bits of B^3 are of no consequence in this instruction. BCD codes of 12_8, 32_8, and 52_8 do not compare equal to zero for this instruction.

The BDP Condition register is not used (always set to 00_2).

SCAN, LR, NE, DC ***Search Field A Left to Right for Inequality, Delimited***



r = unmodified address of the highest order character in field A. R = r + [B_r]

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

SC = scan character which is compared against characters in field A.

$\mathcal{S}_2$ = number of characters to be searched

DC = 6-bit delimiting character compared against the characters in field A

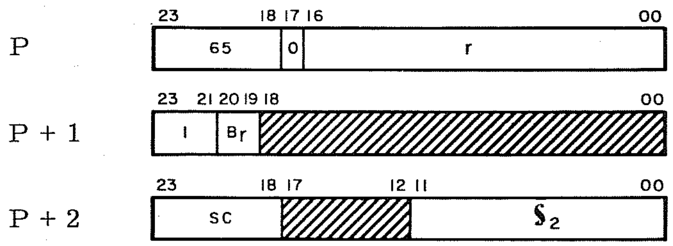
Bits 00 through 18 of P + 1 should be loaded with zeros.

Instruction Description: Search field A from left to right beginning with the 6-bit character at location R, and RNI at P + 4 if a character is found that is not equal to the scan character, SC. If a character is not found that is not equal to the SC after the entire field defined by $\mathcal{S}_2$ has been searched or if a character is found that equals the delimiting character, DC, RNI at P + 3. (If SC equals DC and an unequal comparison with field A occurs, RNI at P + 4).

Comments: If an unequal character comparison is found during the search, the number of characters searched is transferred to the lower 12 bits of B^3. If all characters searched are equal to the SC, then (B^3) = $\mathcal{S}_2$. The upper 3 bits of B^3 are of no consequence in this instruction. BCD codes of 12_8, 32_8 and 52_8 cause an unequal comparison to zero.

The BDP Condition register is not used (always set to 00_2).

SCAN, RL, EQ
Search Field A Right to Left for Equality



r = unmodified address of the lowest order character in field A. R = r + [B_r]

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

SC = scan character which is compared against characters to be searched.

$\S_2$ = number of characters to be searched

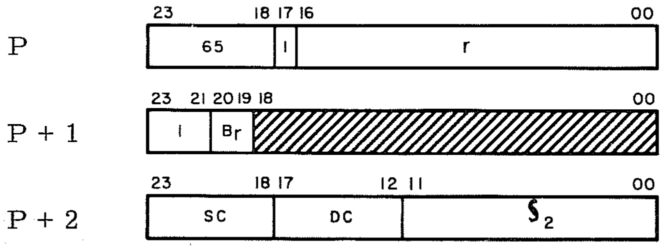
Bits 00 through 18 of P + 1 and bits 12 through 17 of P + 2 should be loaded with zeros.

Instruction Description: Search field A from right to left beginning with the 6-bit character at location R and RNI at P + 4 if a character is found that is identical to the scan character, SC. If a character is not found that equals the SC after the entire field defined by $\S_2$ has been searched, RNI at P + 3.

Comments: If a character comparison is found during the search, the number of characters searched is transferred to the lower 12 bits of B^3. If an unsuccessful search is made, then $(B^3) = \S_2$. The upper 3 bits of B^3 are of no consequence in this instruction. BCD codes of 12_8, 32_8, and 52_8 do not compare equal to zero.

The BDP Condition register is set to the sign of field A.

SCAN, RL, EQ, DC
Search Field A Right to Left for Equality, Delimited



r = unmodified address of the lowest order character in field A. $R = r + [B_r]$

B_r = index register flag for field A
If $B_r = 1$ or 3, use index register B^1
If $B_r = 2$, use index register B^2
If $B_r = 0$, no indexing
SC = scan character which is compared against characters in field A.
$\mathcal{S}_2$ = number of characters to be searched
DC = 6-bit delimiting character compared against the characters in field A

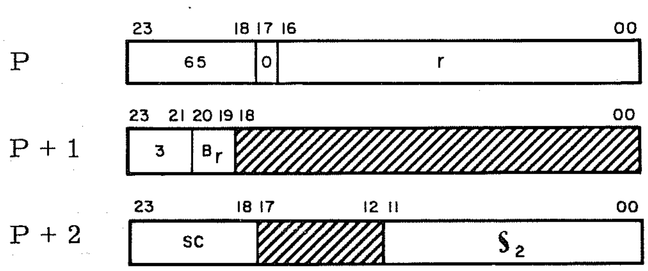
Bits 00 through 18 of P + 1 should be loaded with zeros.

Instruction Description: Search field A from right to left beginning with the 6-bit character at locations R and RNI at P + 4 if a character is found that is equal to the scan character, SC. If a character is not found that equals the SC after the entire field defined by $\mathcal{S}_2$ has been searched or if a character is found that equals the delimiting character , DC, RNI at P + 3 (if SC equals DC and an equal comparison with field A occurs. RNI at P + 4).

Comments: If a character comparison is found during the search, the number of characters searched is transferred to the lower 12 bits of B^3. If an unsuccessful search is made, then $(B^3) = \mathcal{S}_2$. The upper 3 bits of B^3 are of no consequence in this instruction. BCD codes of 12_8, 32_8, and 52_8 do not compare equal to zero.

The BDP Condition register is set to the sign of field A.

SCAN, RL, NE
Search Field A Right to Left for Inequality



r = unmodified address of the lowest order character in field A. $R = r + [B_r]$

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

SC = scan character which is compared against characters in field A

$\S_2$ = number of characters to be searched

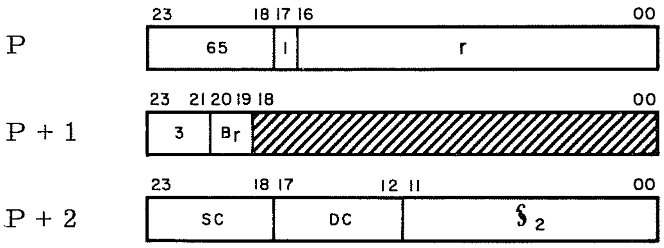
Bits 00 through 18 of P + 1 and bits 12 through 17 of P + 2 should be loaded with zeros.

Instruction Description: Search field A from right to left beginning with the 6-bit character at location R and RNI at P + 4 if a character is found that is not equal to the scan character, SC. If a character is not found that is not equal to the SC after the entire field defined by $\S_2$ has been searched, RNI at P + 3.

Comments: If an unequal character comparison is found during the search, the number of characters searched is transferred to the lower 12 bits of B^3. If all characters searched equal SC, then $(B^3) = \S_2$. The upper 3 bits of B^3 are of no consequence in this instruction. BCD codes 12_8, 32_8, and 52_8 cause an unequal comparison to zero.

The BDP Condition register is set to the sign of field A.

SCAN, RL, NE, DC
Search Field A Right to Left for Inequality, Delimited



r = unmodified address of the lowest order character in field A. R = r + [B_r]

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing
SC = scan character which is compared against characters in field A
$\S_2$ = number of characters to be searched
DC = 6-bit delimiting character compared against the characters in field A

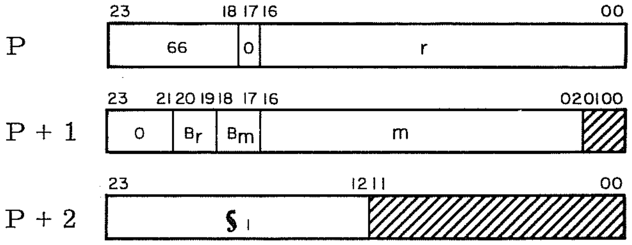
Bits 00 through 18 of P + 1 should be loaded with zeros.

Instruction Description: Search field A from right to left beginning with the 6-bit character at location R and RNI at P + 4 if a character is found that is not equal to the scan character, SC. If a character is not found that is not equal to the SC after the entire field defined by $\S_2$ has been searched, or if a character is found that equals the delimiting character, DC, RNI at P + 3 (If SC equals DC and an unequal comparison with field A occurs, RNI at P + 4).

Comments: If an unequal character comparison is found during the search, the number of characters searched is transferred to the lower 12 bits of B^3. If all characters searched equal SC, the $(B^3) = \S_2$. The upper 3 bits of B^3 are of no consequence in this instruction. BCD codes of 12_8, 32_8, and 52_8 cause an unequal comparison to zero.

The BDP Condition register is set to the sign of field A.

CVDB
Convert BCD to Binary



r = unmodified address of the highest order BCD character in field A. R = r + [B_r]

B_r = index register flag for field A.
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

m = unmodified address of the highest order bits in binary field C. M = m + [B_m]

B_m = index register flag for binary field C. (same bit functions as B_r). Bits 02-14 of the index register are used for word address indexing (13 bit index-sign extended). Bits 00 and 01 must be set to "1's".

$\S_1$ = number of BCD characters to be converted.

Bits 00 and 01 of P + 1, and bits 00 through 11 of P + 2 should be loaded with zeros.

Instruction Description: Convert a BCD number of up to 14_{10} numeric characters in magnitude (including sign) into its binary equivalent in one's complement notation and store the result in locations M and M + 1.

Comments: The resultant binary number is placed into the words at locations M and M + 1. The sign of the binary number is always stored in bit 23 of M. The maximum positive BCD number (fourteen 9's) is equal to 265714203643777_8. The complement of this number in M and M + 1 equals a negative field of fourteen 9's.

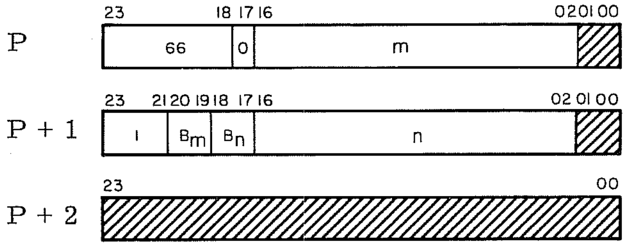
The BDP Condition register is set to the sign of field A. The B^3 register contains a count of the BCD characters converted.

A BCD fault is generated if one of the following conditions occur:

1. Zone portion (upper 2 bits) of any character (except sign character) does not equal zero.
2. Numeric portion (lower 4 bits) of any character in field A contains a BCD code greater than 11_8, except the sign character where a 12_8 code is legal.
3. The sign character contains a 72_8 code.
4. More than 14_{10} BCD characters are specified by $\S_1$.

Operation continues despite any BCD fault.

CVBD
Convert Binary to BCD



m = unmodified address of the highest order bits in binary field A.
$M = m + [B_m]$

B_m = index register flag for field A.
If B_m = 1 or 3, use index register B^1
If B_m = 2, use index register B^2
If B_m = 0, no indexing

Bits 02-14 of the index register are used for word address indexing (13 bit index-sign extended). Bits 00 and 01 must be set to "1's".

n = unmodified address of the highest order BCD character in field C.
$N = n + [B_n]$

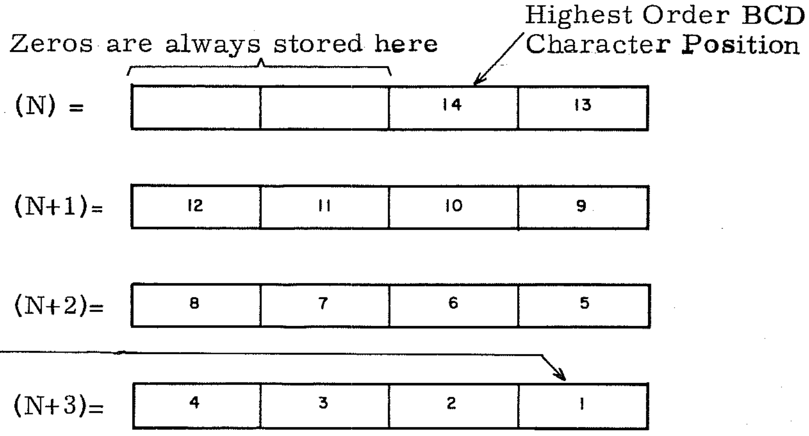
B_n = index register flag for field C. (same bit functions as B_m)

Bits 00 and 01 of P, bits 00 and 01 of P + 1, and bits 00 through 23 of P + 2 should be loaded with zeros.

Instruction Description: Convert the 48-bit binary number (including sign) at M and M + 1 into its signed, numeric BCD equivalent and store in field C.

Comments: (M) and (M + 1) are always analyzed for the binary number to be converted. This is true even if the binary number does not utilize all of the bit positions of M. The sign of the original binary number is located in bit 23 of M. The sign of the binary number is stored in the field C sign character and is set in the BDP Condition register.

A BCD fault is generated if the conversion results in a number of more than 14_{10} BCD characters. However the lower 14_{10} characters will be correct.



Highest Order BCD Character Position
Zeros are always stored here
(N) =
14
13
(N+1)=
12
11
10
9
(N+2)=
8
7
6
5
(N+3)=
4
3
2
1

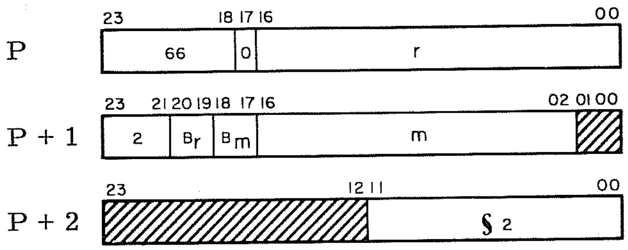
BCD character positions appear in this order. (This character position always contains the sign of the field.)

The original contents of index register B^3 is destroyed. Bits 0-11 are set to all zeros.

DTA
Convert
BCD to ASCII

NOTE

This instruction is available only in the 3312 and 3304-2.



r = unmodified address of the highest order character in BCD field A.
R = r + [B_r]

B_r = index register flag for field A.
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

m = unmodified word address of the highest order characters in ASCII field C.
M = m + [B_m]

B_m = index register flag for field C (same bit functions as B_r). Bits 02-14 of the index register are used for word address indexing (13 bit index-sign extended). Bits 00 and 01 must be set to "1's".

$§_2$ = number of characters to be converted

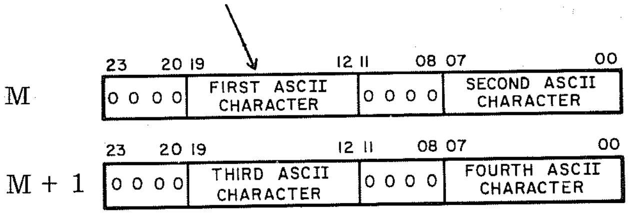
Bits 00 and 01 at P + 1 and bits 12 - 23 of P + 2 should be loaded with zeros.

Instruction Description: Convert a field of up to 4095 6-bit BCD characters in field A into the 8-bit American Standard Code for Information Interchange (ASCII) in field C.

Comments: Conversion proceeds from left to right (Refer to Appendix A for a comparative listing of BCD and ASCII codes.)

If the last ASCII character stored occupies bits 12-19, zeros are stored in bits 00-07. The BCD codes of 12_8, 32_8, and 52_8 are not treated as zero for this instruction. The count in the B^3 register upon completion equals four times the number of 24-bit words stored. The BDP Condition register is not used (always set to 00_2).

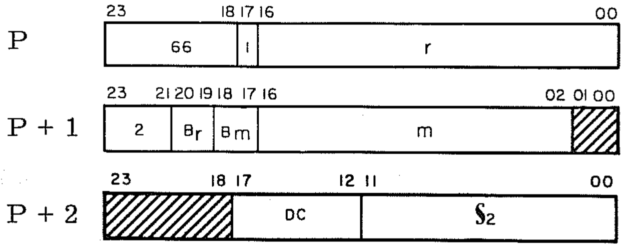
The first ASCII character is always placed in this character position of the word specified by M.



DTA, DC
Convert BCD to ASCII, Delimited

NOTE

This instruction is available only in the 3312 and 3304-2.



r = unmodified address of the highest order character in BCD field A.
$R = r + [B_r]$.

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

m = unmodified address of the highest order word in ASCII field C.
$M = m + [B_m]$

B_m = index register flag for field C (same bit functions as B_r). Bits 02-14 of the index register are used for word address indexing (13-bit index-sign extended). Bits 00 and 01 must be set to ones.

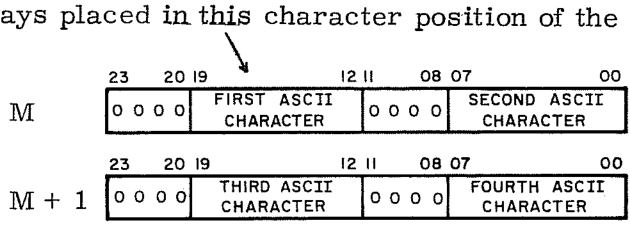
$\S_2$ = number of characters in ASCII field C

DC = 6-bit delimiting character compared against characters in field A.

Bits 18 through 23 at P + 2 and 00 and 01 at P + 1 should be loaded with zeros.

Instruction Description: Convert a field of up to 4095 6-bit BCD characters in field A into the 8-bit American Standard Code for Information Interchange (ASCII) in field C. The conversion proceeds from left to right. The operation is terminated when the number of characters specified by $\S_2$ has been converted or the delimiting character is recognized in field A. A character equal to the delimiting character is converted when encountered.

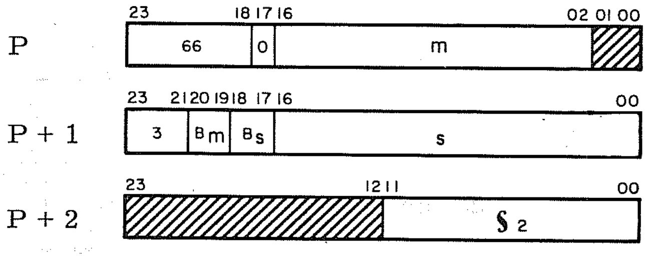
Comments: The BCD codes of 12_8, 32_8, and 52_8 are not treated as zero for this instruction. The count in the B^3 register upon completion equals four times the number of 24-bit words stored. The BDP condition register is not used (always set to 00_2).
The first ASCII character is always placed in this character position of the word specified by M.



ATD
Convert ASCII to BCD

NOTE

This instruction is available only in the 3312 and 3304-2.



m = unmodified word address of the highest order characters in ASCII field A.

M=m + B_m (Refer to diagram below.)

B_m = index register flag for field A
If B_m = 1 or 3, use index register B^1
If B_m = 2, use index register B^2
If B_m = 0, no indexing
Bits 02-14 of the index register are used for word address indexing (13-bit index-sign extended). Bits 00 and 01 must be set to ones.

s = unmodified address of the highest order character in BCD field C. S = [s + Bs]

B_s = index register flag for field C (same bit functions as B_m)

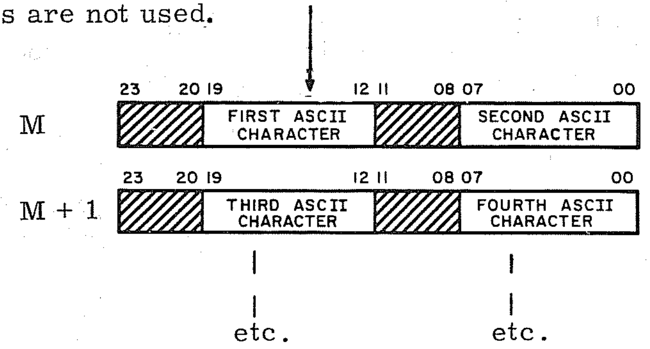
$\S_2$ = number of characters in BCD field C

Bits 00 and 01 of P and bits 12-23 of P + 2 should be loaded with zeros.

Instruction Description: Convert a field of up to 4095 8-bit American Standard Code for Information Interchange (ASCII) characters in field A into 6-bit BCD characters in field C. The conversion proceeds from left to right. The operation is terminated when $\S_2$ is exhausted.

Comments: A BCD fault is generated if bit positions 05 and 06 for any character contain both "1's" or both "0's" (an illegal character). An illegal character is set to zero in field C and conversion continues to completion. The B^3 register indicates the number of BCD characters stored. The BDP Condition register is not used (always set to 00_2).

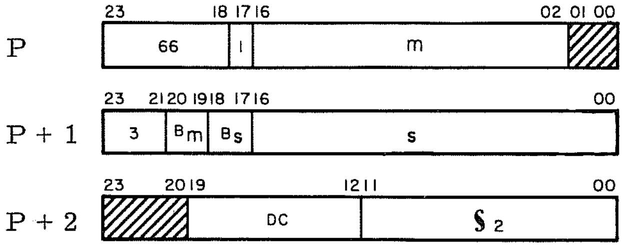
The first ASCII character is always located in this character position of the word specified by M. Shaded areas are not used.



ATD, DC
Convert ASCII to BCD, Delimited

NOTE

This instruction is available only in the 3312 and 3304-2.



m = unmodified word address of the highest order characters in ASCII field A. (Refer to diagram below.) M = m + B_m

B_m = index register flag for field A
If B_m = 1 or 3, use index register B^1
If B_m = 2, use index register B^2
If B_m = 0, no indexing
Bits 02-14 of the index register are used for word address indexing (13-bit index-sign extended). Bits 00 and 01 must be set to "1's".

s = unmodified address of the highest order character in BCD field C. S = [s + Bs]

B_s = index register flag for field C. (same bit functions as B_m)

$\S_2$ = number of characters in BCD field C

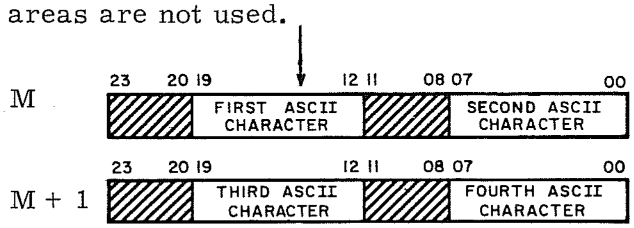
DC = 8-bit delimiting character compared against characters in field A

Bits 20 through 23 at P + 2 and 00 and 01 at P should be loaded with zeros.

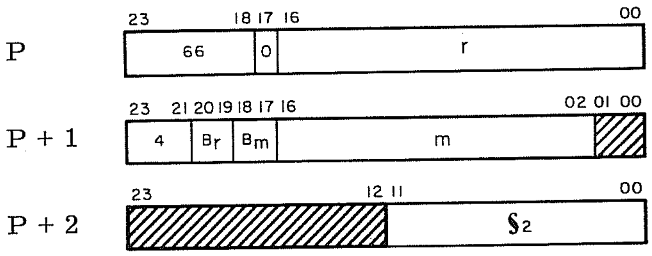
Instruction Description: Convert a field of up to 4095 8-bit American Standard Code for Information Interchange (ASCII) characters in field A into 6-bit BCD characters in field A. Conversion proceeds from left to right. The operation is terminated when $\S_2$ is exhausted or an ASCII character equal to the delimiting character is recognized. A character equal to the delimiting character is converted when encountered.

Comments: A BCD fault is generated if bit positions 05 and 06 for any character contain both "1's" or both "0's" (an illegal character). An illegal character is set to zero in field C and conversion continues to completion. The B^3 register indicates the number of BCD characters stored. The BDP Condition register is not used (always set to 00_2).

The first ASCII character is always derived from this character position of the word specified by M. Shaded areas are not used.



PAK
Convert 6-Bit BCD to 4-Bit BCD



r = unmodified address of the lowest order character in the A field.
$R = r + [B_r]$

Br = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

m = unmodified address of the lowest order word in field C.
$M = m + [B_m]$

B_m = index register flag for field C (same bit functions as B_r)
Bits 02-14 of the index register are used for word address indexing (13-bit index-sign extended). Bits 00 and 01 must be set to "1's".

$\S_2$ = number of 6-bit characters in field A to pack.

Bits 00 and 01 or P + 1, and bits 12 through 23 of P + 2 should be loaded with zeros.

Instruction Description: Convert from right to left, a field of numeric 6-bit BCD characters in field A into 4-bit BCD characters and store the result in field C. The zone (upper 2) bits of all 6-bit characters are removed.

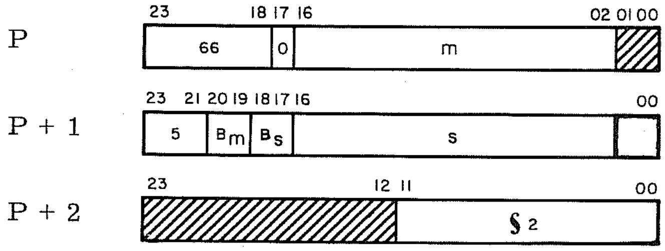
Comments: The sign of field A is converted into a 4-bit sign character (1010_2 - positive, 1011_2 - negative) and is placed in the low order character position of field C prior to packing BCD characters. Any A field character with a 12_8 - 17_8 code is forced to a zero in field C. A full word store is used. Any unfilled 4-bit characters in the highest order word are stored as zeros. The B^3 register contains a count equal to four times the number of 24-bit words stored. The BDP Condition register is not used (always set to 00_2).

A BCD fault is generated if one of the following conditions occur:

1. Zone portion (upper 2 bits) of any character (except sign character) does not equal zero.
2. Numeric portion (lower 4 bits) of any character in field A contains a BCD code greater than 11_8, except the sign character where a 12_8 code is legal.
3. The sign character in field A contains a 72_8 code.

Operation continues despite any BCD fault.

UPAK
Convert 4-Bit BCD to 6-Bit BCD



m = unmodified address of the lowest order word in field A.
$M = m + [B_r]$

B_m = index register flag for field A
If B_m = 1 or 3, use index register B^1
If B_m = 2, use index register B^2
If B_m = 0, no indexing
Bits 02-14 of the index register are used for word address indexing (13-bit index-sign extended).
Bits 00 and 01 must be set to "1's".

s = unmodified address of the lowest order character in field C. S = [s + Bs]

B_s = index register flag for field C (same bit functions as B_m)

S_2 = number of characters resulting in field C.

Bits 00 and 01 of P and bits 12 through 23 should be loaded with zeros.

Instruction Description: Convert from right to left, a field of packed numeric 4-bit BCD characters in field A into 6-bit BCD characters and store the result in field C. Zone bits of new 6-bit characters are set to '00' except in the lowest order character which receives the algebraic sign from the 4-bit sign character in field A. Field C contains one less character than field A due to the elimination of the 4-bit sign character.

Comments: The conversion commences with the sign character of the 4-bit BCD field. If the sign is positive (1010_2), '00' zone bits are stored in the field C sign character; if negative (1011_2), a '10' is stored. A result sign character code (zone and numeric) of 40_8 is converted to 52_8. Any A field character (other than the sign character) containing a 12_8-17_8 code results in a zero in field C.

The B^3 register indicates the number of 6-bit characters stored in field C. The BDP Condition register is not used (always set to 00_2).

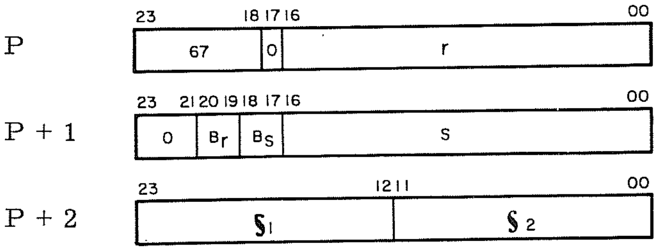
A BCD fault is generated if one of the following conditions occur:

1. The sign character in field A is other than 1010_2 or 1011_2 (the sign of field C is then set to zero), or
2. Any other character in field A contains a 12_8-17_8 code, except the second lowest character where a 12_8 code is legal.

Operation continues despite any BCD fault.

ADM
Add Field A to Field C

This description applies only to the 3312 and 3304-2. See next page for 3304-3.



r = unmodified address of lowest order character in field A. $R = r + [B_r]$

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

s = unmodified address of lowest order character in field C. $S = s + [B_s]$

B_s = index register flag for field C (same bit functions as B_r)

$§_1$ = 12-bit character count specifying the length of the A field

$§_2$ = 12-bit character count specifying the length of the C field

Instruction Description: Add the BCD contents of field A (addend) to field C (augend) proceeding from right to left. The sum appears in field C. The lowest order character of each field specifies sign. The sign of the sum is contained in the character defined by the original address S.

Comments: Field A may be shorter than field C as carries can be set into progressively higher order positions of field C. If any character position of either field contains a 12_8 - 17_8 code, it is converted to zero before the add operation.

The BDP Condition register indicates a positive sign (00) or negative sign (10) according to the result in field C. A zero result may have either a positive or negative sign.

Index register B^3 (bits 0-11) records the C field character count as the instruction progresses.

A BCD fault is generated if one of the following conditions occur:

1. An arithmetic carry is attempted out of the upper limit of field C.
2. $§_1 > §_2$
3. Zone portion (upper 2 bits) of characters in either field (except sign character) does not equal zero.
4. Numeric portion (lower 4 bits) of characters in either field contains a BCD code greater than 11_8, except the sign character where a 12_8 code is legal.
5. The lowest code character (sign character) in either field contains a 72_8 code.

Operation continues despite any BCD fault.

ADM
Add Field A to Field C

Applicable to 3304-3 only. See previous page for 3312 and 3304-2.

	23–18	17	16–00
P	67	0	r

	23–21	20–19	18–17	16–00
P + 1	0	B_r	B_s	s

	23–12	11–00
P + 2	$\S_1$	$\S_2$

r = unmodified address of lowest order character in field A. $R = r + [B_r]$

B_r = index register flag for field A
If $B_r = 1$ or 3, use index register B^1
If $B_r = 2$, use index register B^2
If $B_r = 0$, no indexing

s = unmodified address of lowest order character in field C. $S = s + [B_s]$

B_s = index register flag for field C (same bit functions as B_r)

$\S_1$ = 12-bit character count specifying the length of the A field

$\S_2$ = 12-bit character count specifying the length of the C field

Instruction Description: Add the BCD contents of field A (addend) to field C (augend) proceeding from right to left. The sum appears in field C. The lowest order character of each field specifies sign. The sign of the sum is specified by the character defined by the original address 'S'.

Comments: Field A may be shorter than field C as carries can be set into progressively higher order positions of field C. This instruction normally terminates when the C field is exhausted; however, an early exit occurs prior to this time when the following conditions are present:

1. A field exhausted,
2. ADD and signs alike,
3. No carry, and
4. The current cumulative arithmetic result is not zero.

A 60_8 code (blank) encountered in either field is treated as a 00_8 code. A sign character code of 40_8 in the result field is converted to a 52_8, the code for magnetic tape character negative zero. If any character position of either field contains a 12_8 - 17_8 code, it is converted to zero before the add operation.

The BDP Condition register indicates a positive sign (00) or negative sign (10) according to the result in field C. A zero result will always be positive.

Index register B^3 (bits 0-11) records the field C character count as the instruction progresses.

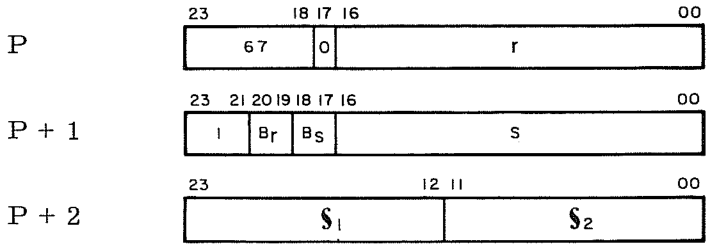
A BCD fault is generated if one of the following conditions occurs:

1. An arithmetic carry is attempted out of the upper limit of field C.
2. $\S_1 > \S_2$
3. Zone portion of characters in either field (except sign character) does not equal zero.
4. Numeric portion of characters in either field contains a BCD code greater than 11_8, except the sign character where a 12_8 code is legal.
5. The sign character in either field contains a 72_8 code.

Operation continues despite any BCD fault.

SBM
Subtract Field A from Field C

Applicable to 3312 and 3304-2 only. See next page for 3304-3.



r = unmodified address of the lowest order character in field A.
$R = r + [B_r]$

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

s = unmodified address of the lowest order character in field C.
$S = s + [B_s]$

B_s = index register flag for field C (same bit functions as B_r)

$\S_1$ = 12-bit character count specifying the length of the A field

$\S_2$ = 12-bit character count specifying the length of the C field

Instruction Description: Subtract the BCD contents of field A (subtrahend) from field C (minuend) proceeding from right to left. The difference appears in field C. The lowest order character in fields A and C contain the algebraic sign of their respective fields. The sign of the difference appears in the lowest order character in field C.

Comments: $\S_1$ must be $\leq \S_2$ since field C must accommodate the result of the subtraction.

If any character position of either field contains a 12_8 - 17_8 code, it is converted to zero before the subtract operation.

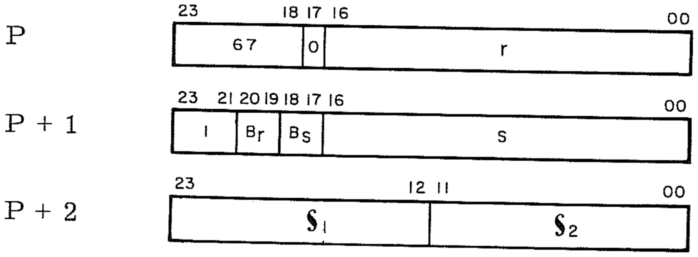
The BDP Condition register indicates a positive sign (00) or a negative sign (10) according to the result in field C. A negative zero result will never occur. BCD codes 12_8 - 17_8 in the lowest order character in either field are converted to zero.

Index register B^3 (bits 0-11) records the field C character count as the instruction progresses.

The conditions for generating a BCD fault are identical to those for the ADM instruction.

SBM
Subtract Field A from Field C

Applicable to 3304-3 only.
See previous page for
3312 and 3304-2.



r = unmodified address of the lowest order character in field A.
$R = r + [B_r]$

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

s = unmodified address of the lowest order character in field C.
$S = s + [B_s]$

B_s = index register flag for field C (same bit functions as B_r)

$\S_1$ = 12-bit character count specifying the length of the A field

$\S_2$ = 12-bit character count specifying the length of the C field

Instruction Description: Subtract the BCD contents of field A (subtrahend) from field C (minuend) proceeding from right to left. The difference appears in field C. The lowest order character of each field specifies sign. The sign of the difference appears in the lowest order character in field C.

Comments: $\S_1$ must be $\leq \S_2$ since field C must accommodate the result of the subtraction. This instruction normally terminates when the C field is exhausted; however, an early exit occurs prior to this time when the following conditions exist:

1. A field exhausted,
2. SUBTRACT and signs alike,
3. No carry, and
4. The current cumulative result is not zero.

A 60_8 code (blank) encountered in either field is treated as a 00_8 code. A sign character code of 40_8 in the result field is converted to a 52_8, the code for magnetic tape character negative zero. If any character position of either field contains a 12_8 - 17_8 code, it is converted to zero before the subtract operation.

The BDP Condition register indicates a positive sign (00) or a negative sign (10) according to the result in field C. A zero result will always be positive.

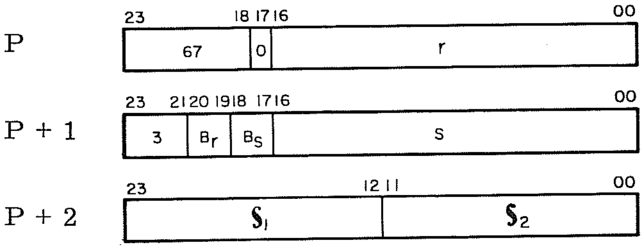
Index register B^3 (bits 0-11) records the field C character count as the instruction progresses.

The conditions for generating a BCD fault are identical to those for the ADM instruction.

CMP
Compare Field A to Field C

NOTE

This description applies to the 3312 and 3304-2 only. See next page for 3304-3.



r = unmodified address of the highest order character in field A
$R = r + (B_r)$

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

s = unmodified address of the highest order character in field C.
$S = s + (B_s)$

B_s = index register flag for field C (same bit function as B_r)

$§_1$ = 12 bit character count specifying the length of the A field

$§_2$ = 12 bit character count specifying the length of the C field

Instruction Description: Compare characters in field A with field C proceeding from left to right. Terminate the operation when an unequal character comparison occurs and RNI at P + 3. If the comparison condition is not satisfied when one of the fields has been completely examined, the remainder of the other field is examined for blanks (60 codes). If the remaining characters are all blanks, the compare operation is terminated as an equal comparison and the next instruction is read from P + 4. If the remainder of the larger field does not contain all blanks and:

If $§_1 > §_2$, then an A > C comparison condition exists.
If $§_2 > §_1$, then an A < C comparison condition exists.

Comments: The result of the comparison is entered into the BCD Condition register as described in the table below. If the fields are unequal, the next instruction is read from P + 3. If the fields are equal, the next instruction is read from P + 4. The count of C field characters processed is placed in the B^3 register upon completion of the instruction.

Comparison Condition	Contents of BDP Condition Register
A = C	00_2
A > C	01_2
A < C	10_2

CMP
Collating Compare

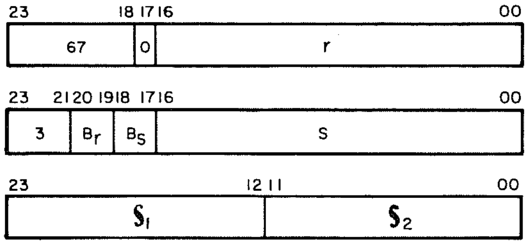
NOTE

This description applies to the 3304-3 only. See previous page for 3312 and 3304-2.

P

P + 1

P + 2



r = unmodified address of the highest order character in field A.
$R = r + [B_r]$

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

s = unmodified address of the highest order character in field C.
$S = s + [B_s]$

B_s = index register flag for field C (same bit function as B_r)

S_1 = 12-bit character count specifying the length of the A field

S_2 = 12-bit character count specifying the length of the C field

Instruction Description: Compare BCD characters in field A with corresponding characters in field C for inequality. The operation proceeds from left to right and terminates when an unequal comparison occurs or when both fields are exhausted. If inequality is detected, a table in storage that contains a pre-stored collating sequence is referenced to determine which of the two unequal characters ranks highest. Results of the compare are recorded in the BDP Condition register as follows:

Comparison Condition	Contents of BDP Condition Register
A = C	00_2
A > C	01_2
A < C	10_2

If the comparison is unequal, the next instruction is read from P + 3. If the two fields are identical, the next instruction comes from P + 4. At termination, B^3 holds the number of C field characters examined.

If field lengths are different and one field is depleted without finding inequality, the remainder of the longer field is examined for blanks (60_8 codes). If all blanks are found, an equal comparison (A=C) is recorded. If a non-blank is detected and

$\S_1 > \S_2$, then an A > C condition is recorded

$\S_2 > \S_1$, then an A < C condition is recorded

The user can establish any collating sequence he wishes in the storage table. The table is located in absolute addresses 00040_8 - 00057_8 (character addresses 00200_8 - 00277_8) in main core storage. It contains 64 character locations, one for each of the 64 possible 6-bit BCD code combinations that can appear in the A or C field. Each 6-bit BCD code selects one character location in the table as follows:

code 00_8 references location 00040_8, character 0

code 01 references location 00040, character 1

code 02 references location 00040, character 2

code 03 references location 00040, character 3

code 04 references location 00041, character 0

.

.

.

.

code 04 references location 00057, character 3

The contents of each character location in the table is a 6-bit quantity that represents the rank of the corresponding code within the collating sequence.

The character address corresponding to any 6-bit code can be determined by adding the 6-bit code to 00200_8. For example, the character address referenced by code 17_8 is 00217_8.

Example

Assume two fields of equal length, to be ordered according to the USASCII collating sequence pre-stored in the storage table:

Field A = 3N374	03	45	03	07	04
Field C = 3N3X2	03	45	03	54	02
				↑	

Step 1 - A character by character compare is performed; inequality between the two fields is detected at the 4th character.

Step 2 - The table is referenced to obtain the two corresponding ASCII collating positions for the unequal characters. The table addressing is automatically done by adding the table base address (00200_8) to the 6-bit code:

Field A table character address = 00200 + 07 = 00207

Field C table character address = 00200 + 54 = 00254

00040 (200 - 203)	20	21	22	23
00041 (204 - 207)	24	25	26	27
00053 (254 - 257)	12	03	74	36

Table containing ASCII collating sequence

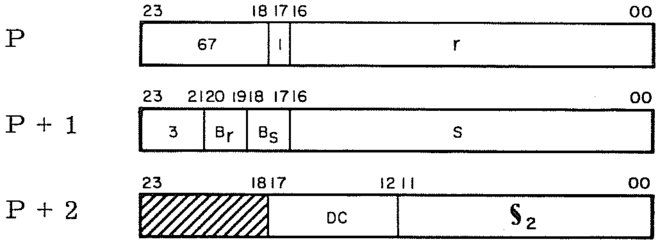
The table indicates that code 07 (from field A) ranks 27th within the collating sequence while code 54 (from field C) ranks only 12th. Thus an A > C result is recorded in the BDP Condition register as (01_2).

Step 3 - Perform an RNI at P + 3. B3 register would contain a character count of 4 upon the exit.

CMP, DC
Compare Field A to Field C, Delimited

NOTE

This description applies to the 3312 and 3304-2 only. See next page for 3304-3.



r = unmodified address of the highest order character in field A.
$R = r + (B_r)$

B_r = index register flag for field A
If B_r = 1, or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

s = unmodified address of the highest order character in field C.
$S = s + (B_s)$

B_s = index register flag for field C (same bit functions as B_r)

DC = 6-bit delimiting character

$\S_2$ = 12-bit character count specifying the length of field C

Field A always contains the same number of characters as field C during a delimited compare operation. Bits 18 through 23 of P + 2 should be loaded with zeros.

Instruction Description: Compare characters in field A with field C proceeding from left to right. Terminate the operation when: 1) an unequal character comparison occurs (the magnitudes of the unequal characters are noted - see table below), 2) all of the characters in the fields have been examined, 3) a character in either the A or C field equals the delimiting character. RNI at P + 3 for an unequal field comparison; RNI at P + 4 for an equal field comparison.

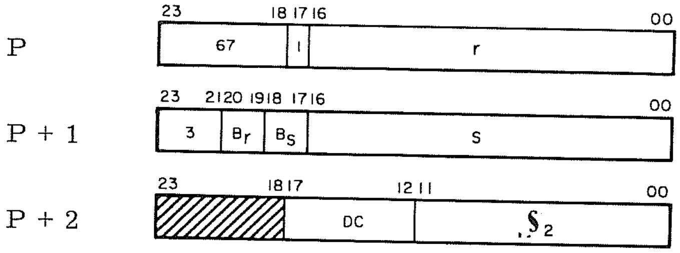
Comments: The count of the characters processed is placed in the B^3 register upon completion of the instruction. The following table describes the state of the BDP Condition register when the operation is terminated:

Terminating Comparison Condition	Contents of BDP Condition Register
A = C (through entire field or through delimiting character)	00_2
A > C	01_2
A < C	10_2

CMP, n Compare Field A to Field C, Numeric

NOTE

This description applies to the 3304-3 only. See previous page for 3312 and 3304-2.



r = unmodified address of the highest order character in field A.
$R = r + [B_r]$

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

s = unmodified address of the highest order character in field C.
$S = s + [B_s]$

B_s = index register flag for field C (same bit functions as B_r)

dc = 6-bit delimiting character

$\S_1$ = 12 bit character count specifying length of field A

$\S_2$ = 12-bit character count specifying the length of field C

Instruction Description: Compare numeric characters in field A with field C, proceeding from right to left. The operation terminates upon completion of the longer field. If field lengths are different, the remainder of the longer field is compared to zero.

For inequality (unequal comparison condition), a RNI is performed at P+3, otherwise the next instruction is fetched from P+4.

After completion of the compare, the high-low equal status is contained in the BDP Condition register:

Condition	Code
A = C	00_2
A > C	01_2
A < C	10_2

If the signs of the two fields are unlike, an exit is made with B^3 (lower 12 bits) equal to zero, otherwise B^3 contains the character count of the C field characters processed.

Comments: Positive numbers are considered greater than negative, except for ±0 where they are considered equal.

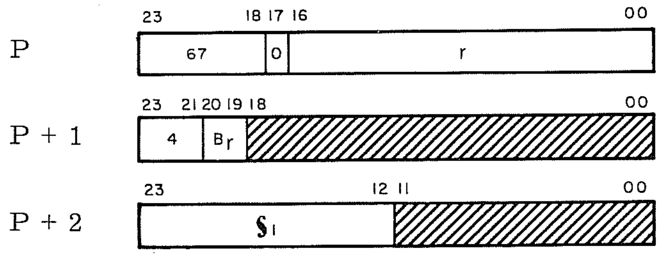
A character with a numeric portion greater then 11_8 is illegal (12_8 is legal in sign character). The numeric portion of an illegal character is converted to 00_8 and this value is used for the comparison. Only the numeric part is tested in the compare operation.

The BCD Fault is generated if one of the following conditions occur:

1. Zone portion of any character (except sign character) does not equal zero.
2. Numeric portion of any character contains BCD code greater than 11_8, except the sign character where a 12_8 code is legal.
3. The sign character contains a 72_8 code.

Operation continues despite any BCD fault.

TST
Test Field A for Sign



r = unmodified address of the lowest order BCD character in field A
R = r + (B_r)

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

$\S_1$ = number of BCD characters in field A to be tested

Bits 00 through 18 of P + 1, and bits 00 through 11 of P + 2 should be loaded with zeros.

Instruction Description: Examine field A from right to left and determine if it is zero, greater than zero, or less than zero. The BDP Condition register is set to reflect the test results (see table below). Field A is scanned until the first non-zero character is encountered or until $\S_1$ is exhausted.

Comments: If field A contains all zeros and the sign character is recognized as zero (contains BCD code of 60, 52, 40, 32, 20, 12, or 00), the Condition register is set to "00". If field A does not equal zero, the Condition register is set as follows:

Field A	Contents of BDP Condition Register
Positive - upper 2 bits of sign character are 00_2, 01_2, or 11_2	01_2
Negative - upper 2 bits of sign character are 10_2	10_2

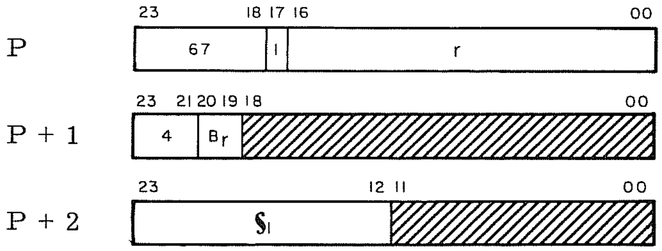
A BCD fault is generated if one of the following conditions occur:

1. Zone portion (upper 2 bits) of any character (except sign character) does not equal zero.
2. Numeric portion (lower 4 bits) of any character contains a BCD code greater than 11_8, except the sign character where a 12_8 code is legal.
3. The sign character contains a 72_8 code.

Operation continues despite any BCD fault.

Index register B^3 (bits 0-11) is not used (set to all zeros).

TSTN
Test Field A for Numeric



r = unmodified address of the lowest order BCD character in field A
$R = r + (B_r)$

B_r = index register flag for field A
If B_r = 1 or 3, use index register B^1
If B_r = 2, use index register B^2
If B_r = 0, no indexing

$\$_1$ = number of BCD characters in field A to be tested.

Bits 00 - 18 of P + 1, and bits 00 - 11 of P + 2 should be loaded with zeros.

Instruction Description: Examine the sign character of field A and record the results in the BDP Condition register (see Comments). Field A is scanned from right to left until $\$_1$ is exhausted. If the BCD Fault is not set upon completion of the instruction, field A is numeric. The BCD Fault can be sensed by instruction.

Comments: If field A contains all zeros and the sign character is recognized as zero (contains BCD code of 60, 52, 40, 32, 20, 12, or 00), the Condition register is set to "0". If field A does not equal zero, the Condition register is set as follows:

Field A	Contents of BDP Condition Register
Positive - upper 2 bits of sign character are 00_2, 01_2, or 11_2	01_2
Negative - upper 2 bits of sign character are 10_2	10_2

A BCD fault is generated if one of the following conditions occur:

1. Zone portion (upper 2 bits) of any character (except sign character) does not equal zero.
2. Numeric portion (lower 4 bits) of any character contains a BCD code greater than 11_8, except the sign character where a 12_8 code is legal.
3. The sign character contains a 72_8 code.

Operation continues despite any BCD fault.

Index register B^3 (bits 0-11) is not used (set to all zeros).

LBR
Load BCR

23 – 18	17 – 15	14 – 00
70	6	m

NOTE

This description applies to the 3304-3 only. See next page for 3312 and 3304-2.

m = storage address. Indexing not permitted.

Instruction Description: Load the BDP Condition Register and set interrupt recovery conditions within the BDP as defined by (m). The 24 bits of (m) have the following significance:

POSITION	FUNCTION
00 and 01	Contents of the BDP Condition register.
02	Edit flag indicating a (DB) or (CR) character detected.
03	Zero suppression operation in progress.
04	Edit flag indicating a floating sign ($, +, -) operation is in progress.
05	Edit flag indicating a $ sign is forced.
06	Edit flag indicating a + sign is forced or compare flag indicating field sign negative.
07	Edit flag indicating an * sign is forced or compare flag indicating A > C condition.
08	Edit flag indicating a floating character is forced or compare flag indicating C > A condition.
09	Operand equals zero.
10	Signs of operands unlike on ADM or SBM, or incorrect on EDIT.
11	Interrupt occurred during BDP operation.
12 - 23	Number of characters or words for Field A already processed.

Comments: This instruction must be used to load recovery conditions into the BDP before restarting a previously interrupted BDP instruction. The recovery information must have been stored in address m by a SBR (70. 7) instruction immediately after the interrupt occurred. The LBR instruction is trapped if the BDP MODE switch is OFF, except during Monitor state of Executive mode when it is a No - OP.

LBR
Load BDP

23 18	17 15	14 00
70	6	m

m = storage address. Indexing not permitted.

This description applies to the 3312 and 3304-2 only. See previous page for 3304-3.

Instruction Description: Load the BDP Condition Register and set interrupt recovery conditions within the BDP as defined by (m). The 24 bits of (m) have the following significance:

POSITION	FUNCTION
00 and 01	Contents of the BDP Condition Register
02	Edit flag indicating a (DB) or (CR) character detected.
03	Zero suppression operation in progress.
04	Edit flag indicating a floating sign ($, +, -) operation is in progress.
05	Edit flag indicating a $ sign is forced.
06	Edit flag indicating a + sign is forced.
07	Edit flag indicating an * sign is forced.
08	Edit flag indicating a floating character is forced.
09	Operand equals zero
10	Signs of operands unlike on ADM or SBM, or incorrect on EDIT.
11	Interrupt occurred during BDP operation.
12 - 23	Number of characters of words for Field A already processed.

Comments: This instruction must be used to load recovery conditions into the BDP before restarting a previously interrupted BDP instruction. The recovery information must have been stored in address m by a SBR (70. 7) instruction immediately after the interrupt occurred. The LBR instruction is trapped if the BDP MODE switch is OFF, except during Monitor state of Executive mode when it is a NO - OP.

SBR
Store (BDP)

23 18	17 15	14 00
70	7	m

m = storage address. Indexing not permitted.

Instruction Description: Store various operating conditions from within the BDP section at address 'm'. Refer to the LBR instruction for the bit functions of (m).

Comments: Execution of this instruction does not clear the operating conditions within the BDP. This instruction is trapped if the BDP MODE switch is OFF, except during Monitor state when it is a No-Op.

6. SOFTWARE SYSTEMS

GENERAL INFORMATION

Control Data supports its lower 3000 Series Computers with a library of excellent standard software products effectively covering a wide range of computer applications.

- Operating Systems exercise supervisory control
- Languages are oriented toward programming needs
- Utility routines perform tasks for user's programs
- Applications systems are specialized programs

This section briefly describes available software systems and also references obtainable documents. To obtain these documents, refer to the Literature Distribution Catalog for the correct Publication numbers.

OPERATING SYSTEMS

Operating systems provided by Control Data make efficient use of various hardware configurations. These operating systems provide automatic job monitoring and supervisory control during compliation, assembly, and execution of user's programs. Systems storage requirements are kept at a minimum and operator intervention reduced significantly by job stacking, automatic accounting and storage allocation, automatic assignment of input/output functions, and by operator messages produced on the standard output comment unit. Operating systems include the following:

- Real-Time SCOPE
- MASTER
- MSOS
- SCOPE Utility Routines

Real-Time SCOPE

An operating system which provides backgrounding, stacked job processing, and priority interrupt handling. Time is shared between a background program and the stacked jobs. The standard SCOPE features are included: I/O and status operations, debugging facilities, and library maintenance.

Documents

General Information
Real-Time SCOPE Operator's Manual
Real-Time SCOPE Reference

MASTER

A multiprogramming system that is adaptable to applications involving multi-access on-line input/output with and without real time calculations as well as to conventional and batch processing applications. MASTER uses mass storage for system storage and temporary storage of user programs, as well as for storage of user files. MASTER allocates tasks to available equipment and handles communication among tasks. The tasks are processed on a priority basis.

Documents

General Information
Reference

MSOS

Provides utilization of mass storage devices. The operating system, the related software packages and library, and user data areas are allocated to disk or similar storage. Time is shared between a background program and the stacked jobs. Background programs may operate in real time. The system also includes priority interrupt handling, I/O and status operations, debugging facilities, and library maintenance.

Documents

General Information
MSOS Reference
MSOS Operator's Manual
PRELIB MSOS

SCOPE Utility Routines

An open-ended peripheral processing package which allows transfer of data between peripheral units and storage media.

Documents

Reference

LANGUAGES

Programmers can choose the language best suited to the needs of their particular problems. Control Data has implemented programming languages which range from machine mnemonics to problem-oriented systems which closely resemble the natural expressions in particular fields of application. The languages include:

- FORTRAN
- COBOL
- ALGOL
- COMPASS
- Data Processing Package
- Report Generator

FORTRAN-32

A versatile mathematical compiler. Most programs written in FORTRAN II and FORTRAN IV are compilable with FORTRAN-32.

Documents

General Information
Reference
Instant FORTRAN
Library Routines
Library Functions

Mass Storage FORTRAN

Provides all the features of FORTRAN-32 and allows compilation and execution using mass storage devices.

Documents

General Information
Reference
FORTRAN/MASTER
Instant FORTRAN
Library Routines
Library Functions

COBOL 32

A data processing language based on the specifications set forth in the DOD reference of COBOL-61, Extended. This language provides fast compilation speeds and efficient object code.

Documents

General Information
Reference
Compatible COBOL
Version 2.0 Extensions and Revisions
Instant COBOL

COBOL 33

Provides all the features of COBOL 32 and utilizes the Business Data Processing hardware during execution of the COBOL object programs.

Documents

Reference

Mass Storage COBOL

Provides all the features of COBOL 32 and uses mass storage for compilation. Version 2 of Mass Storage COBOL contains mass storage statements and allows object programs to use mass storage.

Documents

General Information
Reference
Version 2.0 Extensions and Revisions

ALGOL

A compiler accepting an algorithmic language defined in the ALGOL-60 Revised Report in the Communication of the ACM, 1963, Vol. 6. Input/output procedures are those of the IFIP set and the complete ACM set.

Documents

General Information
Reference
Instant ALGOL
Functional Description ALGOL Compiler
Abnormal Object Time Termination Dump

COMPASS-32

A comprehensive assembly system, providing mnemonic machine operation codes, symbolic addressing, assembly-directing pseudo instructions, and programmer-defined macro instructions.

Documents

General Information
Compatible COMPASS Language Reference Manual
COMPASS/Tape SCOPE COMPASS/Disk SCOPE
  Programming Guide
Instant COMPASS
COMPASS/Real-Time SCOPE. COMPASS/MSOS

COMPASS-33

An extension of COMPASS-32 designed to process coding for the Control Data 3300.

Documents

Compatible COMPASS Language Reference Manual
Instant COMPASS
COMPASS/Real-Time SCOPE. COMPASS/MSOS
COMPASS/MASTER

Data Processing Package

Consists of a set of input/output and file description macro instructions. The set also includes macros to perform certain data manipulation and mathematical functions.

Documents

General Information
Reference

Report Generator

Facilitates the preparation of programs which produce a variety of reports from an input file.

Documents

General Information
Reference
Instant Report

INPUT/OUTPUT

Input/output control routines are included in the software library to provide access to a number of different I/O media through efficiently preprogrammed library routines.

I/O control programs include:

- RESPOND/MSOS
- MSIO
- SIPP

RESPOND/MSOS

A multi-access software package which operates as a background program under MSOS and provides users at remote terminals with the ability to access files of information contained on mass storage at the central computer site. Files may be submitted to the operating system for foreground processing. Records within a file may be added, deleted, modified, or displayed by action of the terminal operator.

MSIO

A file oriented input/output system consisting of two sections. One section provides physical I/O features for mass storage files. The other section provides logical record processing facilities such as blocking, deblocking, buffering, updating, inserting, and deleting for files on mass storage.

SIPP

Enables simultaneous execution of data transfer operations involving several peripheral units. If permitted by the operating system, SIPP can operate as a background program.

Documents

Reference

APPLICATIONS

Applications programs are tested working programs which perform specialized jobs in industry, business, and research. Applications programs include:

- PERT/TIME
- PERT/COST
- SORT
- Mass Storage SORT
- REGINA-I
- ADAPT

PERT/TIME

Utilizes a time-oriented network structure to provide a variety of reports reflecting the actual and scheduled progress of a project.

Documents

PERT General Information
Reference
Version 2.0 Extensions and Revisions

PERT/COST

Utilizes a cost-oriented breakdown structure to provide a variety of reports on actual and estimated costs over the life of a project.

Documents

PERT General Information
Reference

SORT

Produces a sequenced file of data records from random input. The internal phase makes use of the replacement selection sorting technique; the external phase may be either a balanced or poly-phase merge. The user has the option to enter own-code subroutines during the program.

Documents

General Information
Reference

Mass Storage SORT

Similar to the tape SORT except that disk storage is used during intermediate merge processing. The SORT may optionally employ a tag sorting method.

Documents

General Information
Reference

REGINA-I

A linear programming system; it provides an integer solution to the set of equations.

Documents

General Information
Reference

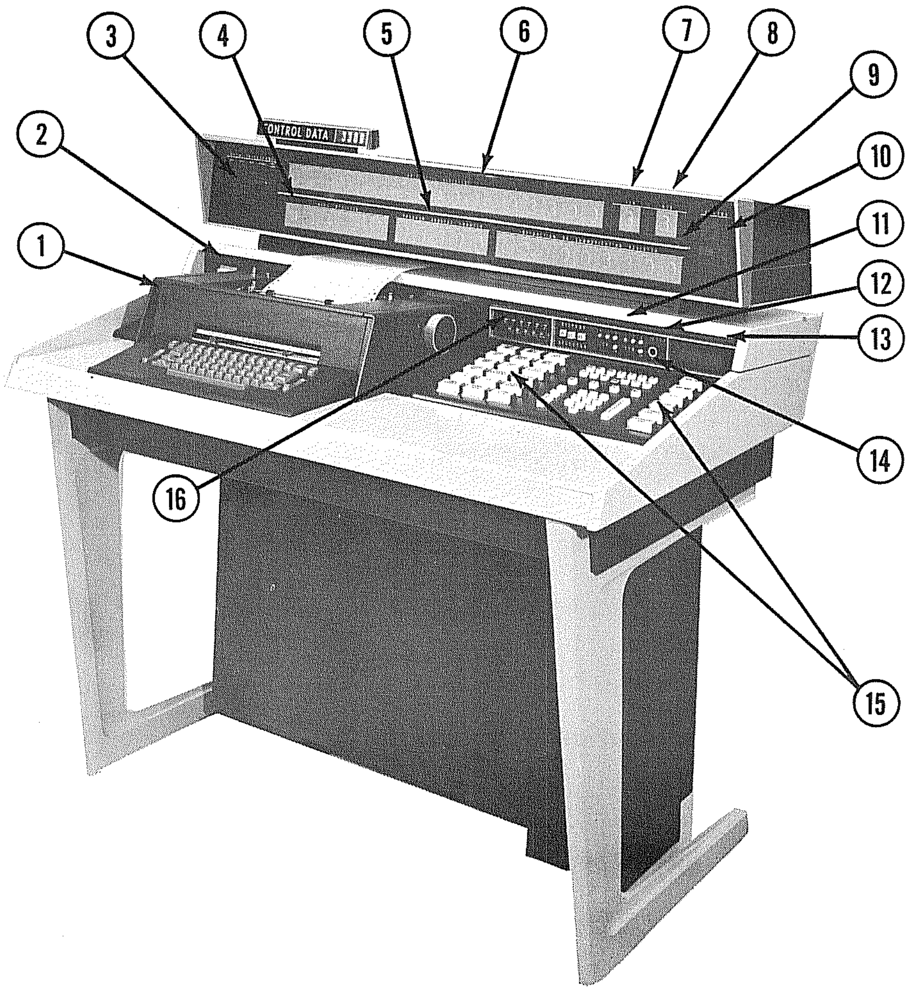
ADAPT

A system that prepares instructions for numerically controlled machine tools. The ADAPT language allows specification of the geometric properties of a part to be machined and the operations involved in producing the part. ADAPT is a subset of the more complex APT system.

Documents

General Information
Reference

7. CONSOLE AND POWER CONTROL PANEL



① Typewriter
② Typewriter Switches
③ Data Interchange Display
④ Index, or LJA, or CIR Register
⑤ P Register or Page Index File Address
⑥ A and Q, or E Register
⑦ Instruction State Register (ISR)
⑧ Operand State Register (OSR)
⑨ F or C Register
⑩ Status Display
⑪ ISR and OSR Entry Switches
⑫ Step Rate Control
⑬ Emergency Off Switch
⑭ Access Keyboard Switches
⑮ Console Condition Switches
⑯ Breakpoint Switch Assembly

Figure 7-1. 3300 Console

GENERAL INFORMATION

The 3300 desk console shown in Figure 7-1 enables the computer operator to control and observe computer operation. This section describes the operator's controls and the significance of the visual indicators. Also included in this section is a description of the Power Control Panel.

CONSOLE

The console provides an operator with visual displays to monitor the current status of computer, controls for setting certain conditions and performing operations, and a typewriter for direct input and output communications with the computer. Each of these areas are described in the following pages to familiarize the operator with the functions of the console.

Register Displays

Figure 7-2 shows the display locations of the operational registers described in Section 1. Entering data into the Communication register, Instruction State Register, or Operand State register is described below.

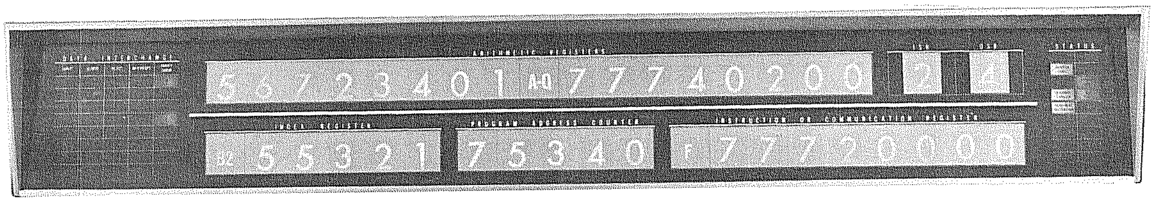


Figure 7-2. Register Display Area

Instruction and Communication Registers

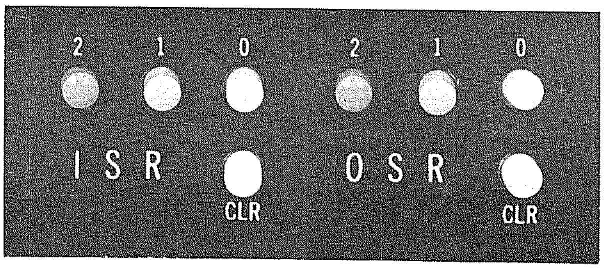
The Instruction register (F register) and Communication register (C register) share the same display area on the console. The F register is displayed when the access keyboard switches are inactive and the computer is not in the GO mode. The C register is displayed when data is being entered via the access keyboard switches.

Data entered into the A or Q registers must first pass through the Communication register. Starting with the uppermost digit, data is entered into the Communication register by first depressing a register switch and then depressing the numeric keyboard switches. A blue Active Digit indicator light is superimposed on each digit position of the Communication register as digit entry progresses. When data is entered into the B^1, B^2, B^3, or P registers, the Active Digit indicator automatically starts at the fifth digit position of the Communication register.

Depressing the TRANSFER switch causes the data to be transferred from the Communication register to the designated register. Immediately depressing the TRANSFER switch again results in transferring all zeros to the register.

Instruction State and Operand State Registers

The contents of the ISR or OSR may be changed by first clearing the register(s) and then depressing binary position switches to form the desired octal number. The switches may be depressed simultaneously or individually. The white register clearing switch and blue binary position switches are shown in Figure 7-3.

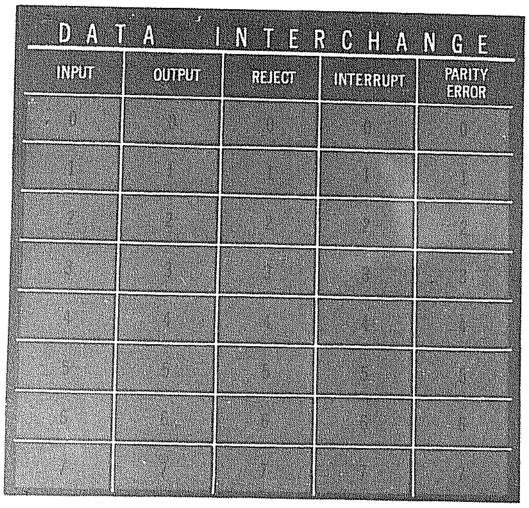


1756

Figure 7-3. ISR and OSR Display and Binary Entry Switches

Data Interchange Display

The Data Interchange Display, shown in Figure 7-4, enables the console operator to determine the status of each of the eight I/O channels (0 through 7). Each channel has its own set of Input, Output, Reject, Interrupt, and Parity Error indicators. Transient conditions may not be seen on the display due to the response time of the indicators.



1764

Figure 7-4. Data Interchange Display

TABLE 7-1. DATA INTERCHANGE INDICATOR DESCRIPTIONS

INDICATOR NAME	FUNCTION
INPUT	Glows when data is being received by the computer on the channel indicated.
OUTPUT	Glows when data is being transmitted by the computer on the channel indicated.
REJECT	Glows when a Reject signal is received from a peripheral equipment on the channel indicated.
INTERRUPT	Glows when an Interrupt is received from a peripheral equipment on the channel indicated. Indicator glows until the interrupting condition is cleared.
PARITY ERROR	Glows when a transmission parity error has occurred on the channel indicated. Indicator glows until the condition is recognized.

Status Display

The Status Display provides the operator with visual indications of the internal status of the computer. Operating status, fault conditions, and physical malfunctions are the general status areas associated with the Status Display indicators. Figure 7-5 shows the arrangement of the indicators on the Status Display and the function of each indicator is described in Table 7-2.

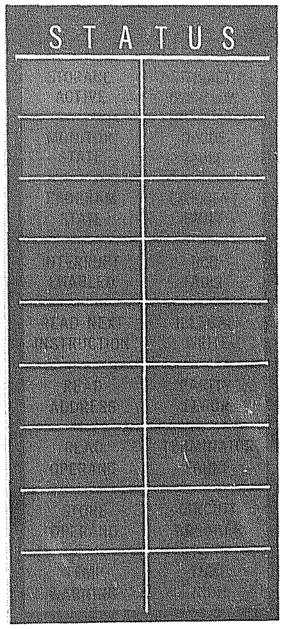


Figure 7-5. Status Display

TABLE 7-2. STATUS DISPLAY INDICATOR DESCRIPTIONS

INDICATOR NAME	FUNCTION
STORAGE ACTIVE	Indicates a storage reference is in progress. Indicator is common to all storage modules.
MONITOR STATE	Indicates the computer is operating in the Monitor State of Executive mode.
PROGRAM STATE	Indicates the computer is operating in the Program State of Executive mode.
INTERRUPT ENABLED	Indicates the interrupt system has been enabled by executing an EINT (77. 74) instruction.
READ NEXT INSTRUCTION	Indicates the computer is reading the next instruction of the program it is currently executing. Usually referred to as the RNI cycle.
READ ADDRESS	Indicates the computer is reading the lower 18 bits at a storage location to form a new address for indirect addressing. Usually referred to as the RADR cycle.
READ OPERAND	Indicates the computer is reading a 24-bit operand from storage for use with the instruction being executed. Usually referred to as the ROP cycle.
STORE OPERAND	Indicates the computer is storing a 24-bit operand that has been previously processed into a selected storage module. Usually referred to as the STO cycle.
TEMP WARNING	Indicates the temperature within the computer is abnormally high and is at least 80° F.
ARITHMETIC OVERFLOW	Indicates the capacity of the adder has been exceeded. Its capacity, including sign, is 24 or 48 bits for 24-bit precision or 48-bit precision, respectively.
DIVIDE FAULT	The divide fault indicates a quotient, including sign, exceeds 24 or 48 bits for 24-bit precision or 48-bit precision, respectively. Therefore, attempts to divide by too small a number, including positive and negative zero, result in a divide fault. During floating point division, a divide fault occurs if division by zero or by a number that is not in floating point format is attempted. If the divisor is not properly normalized a divide fault may also occur. Refer to Appendix B for a description of normalization.
EXPONENT FAULT	Indicates either an exponent overflow ($> +1777_8$) or an exponent underflow ($< -1777_8$) has occurred during a floating point arithmetic operation.

(Continued)

TABLE 7-2. STATUS DISPLAY INDICATOR DESCRIPTIONS (Cont'd)

INDICATOR NAME	FUNCTION
BCD FAULT	Indicates a BCD fault has occurred within the BDP module or a SBCD (77.72) instruction has been executed. Refer to Section 4, Interrupt System, for additional information.
ILLEGAL WRITE	Indicates an attempt has been made to write into a protected storage location or read from certain locations while operating in Executive mode. Refer to Section 4, Interrupt System, for additional information.
PARITY ERROR	Indicates a parity error occurred during a memory reference. Transmission parity errors do not affect this indicator.
TERMINATOR FAULT	Indicates that the internal terminator power supplies are not functioning properly.
CIRCUIT BREAKER	Indicates that one or more of the power system circuit breakers are open.
TEMP HIGH	If the TEMP WARNING indicators are glowing and an absolute temperature of 110° F is exceeded, the computer automatically shuts off logic power. The TEMP HIGH indicator for the particular computer section continues to glow until the temperature drops below the absolute limit. Secondary power must be manually reapplied before normal operation can resume.

Switches and Controls

Condition Switches

The condition switches are used mainly to set various operating and programming conditions. These 24 switches are located on both sides of the access keyboard switches and are shown in Figure 7-6 and described in Table 7-3. The typewriter control switches, located on the extreme left side of the console are described later in this Section.

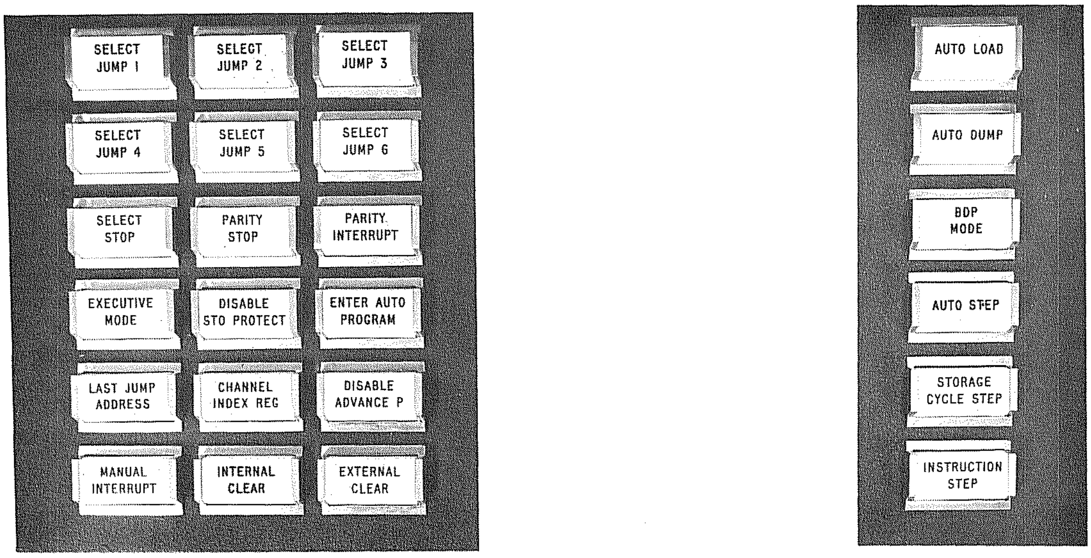


Figure 7-6. Condition Switches

TABLE 7-3. CONDITION SWITCHES DESCRIPTION

SWITCH NAME	FUNCTION
SELECT JUMP (1-6)	Switches are actuated in accordance with programs utilizing the Selective Jump instruction (SJ1-6 00j).
SELECT STOP	Stops the computer when the SLS (77.70) instruction is read. When the computer enters the GO mode again, the program resumes with the next instruction.
PARITY STOP	Causes the computer to halt when a storage parity error is detected.
PARITY INTERRUPT	Causes the computer to process the interrupt subroutine when a storage parity error is detected. Refer to Section 4, Interrupt System, for additional information.
EXECUTIVE MODE	Permits the computer to operate in the Executive mode. Initial state of Executive mode is always the Monitor State. Reactuating this switch permits the computer to operate in the non-Executive mode.
AUTO LOAD	If the computer has been Master Cleared and the AUTO LOAD switch is actuated, the computer automatically jumps to address 77740 if in the non-Executive mode or address 003700 in Executive mode and executes the instruction stored there. Refer to Auto Load/Auto Dump in Section 3.
AUTO DUMP	This switch performs the same function as the AUTO LOAD switch with the exception of jumping to address 77760 if in the non-Executive mode or address 003740 in Executive mode.

(Continued)

TABLE 7-3. CONDITION SWITCHES DESCRIPTION (Cont'd)

SWITCH NAME	FUNCTION
BDP MODE (Business Data Processor)	Actuating this switch with the BDP module in the system permits the BDP to directly execute the business oriented instructions. If the switch is not On, these instructions are trapped. Refer to Section 5, Instructions, for a list of the BDP instructions.
AUTO STEP	Permits instructions to be executed in a slow speed GO mode. The speed (3 to 50 instructions per second) is regulated by a variable Step Rate control on the Upper Console Switch Panel.
STORAGE CYCLE STEP	Enables the operator to step through an instruction one storage cycle at a time, i.e., RNI, RADR, ROP, or STO.
INSTRUCTION STEP	Enables the operator to execute a program, instruction by instruction. One instruction is executed each time the switch is pressed.
THERMOSTAT BY PASS	Allows computation to proceed regardless of abnormal temperatures within the computer.
DISABLE STO PROTECT	Disables the protection feature of the 15 storage protect switches. This switch has no effect on the protected Auto Load and Auto Dump or program protected storage areas.
ENTER AUTO PROGRAM	Allows the operator to enter the Auto Load and Auto Dump storage areas with different data.
LJA (Last Jump Address)	Actuating this momentary switch when the computer is stopped causes the storage address of the last jump instruction to be displayed on the console.
CHANNEL INDEX REG (Channel Index Register)	Pressing this switch when the computer is stopped causes the contents of the 3-bit Channel Index register to be displayed.
DISABLE ADVANCE P	Prevents the P register from being incremented. When the GO switch on the keyboard is pressed, the same instruction is repeatedly executed.
MANUAL INTERRUPT	Forces the computer into an interrupt routine if the computer is in the GO mode. If the computer is stopped when the switch is pressed, it goes into an interrupt routine as soon as the GO switch is pressed.
INTERNAL CLEAR	Master clears internal conditions and registers.
EXTERNAL CLEAR	Master clears all external equipments and the I/O channels.

Access Keyboard

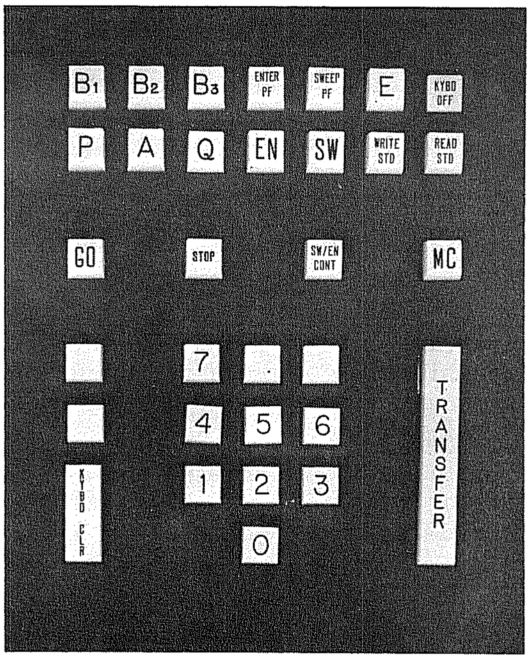
Figure 7-7 shows the access keyboard switches, used for manually entering and retrieving data from the computer and controlling its operation. Table 7-4 describes the individual keyboard switch functions.

Upper Console Switch Panel

The upper console switch panel shown in Figure 7-8 is used for:

- Selecting Index register B^1, B^2, or B^3 for display
- Operating the Breakpoint switch
- Entering data into the ISR or OSR
- Adjusting the Step Rate control
- Immediately removing computer power in the event of an emergency by depressing the EMERGENCY OFF switch

The Index register switches on the access keyboard are used for entering data. To display one of the three index registers, the appropriate upper console index register switch must be depressed. A complete description of the Breakpoint switch follows the access keyboard switch descriptions.



1757

Figure 7-7. Access Keyboard Switches

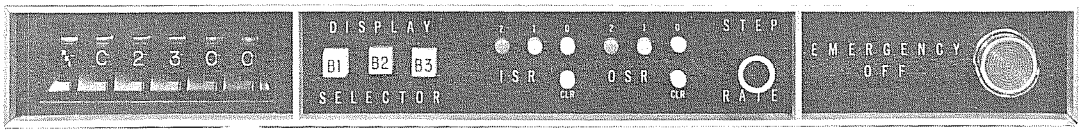


Figure 7-8. Upper Console Switch Panel

TABLE 7-4. ACCESS KEYBOARD SWITCHES

SWITCH NAME	FUNCTIONS
A	Causes both A and Q to be displayed, but permits entry only into A.
Q	Causes both A and Q to be displayed, but permits entry only into Q.
E	Causes E_U and E_L to be displayed. Manual entry is not possible.
P	Enables an address to be manually entered from the keyboard into the P register.
B_1, B_2, or B_3	Enables data to be manually entered into Index registers B^1, B^2, or B^3 from the keyboard. Appropriate Index register switch on the upper console switch panel must be depressed for register display.
EN (ENTER)	Permits data to be manually entered into storage while the computer is stopped. First address of sequence must be previously entered into P. Pressing the TRANSFER switch advances P.
ENTER PF	Permits data to be manually entered into the Page Index File while the computer is stopped. First address of sequence must be previously entered into the lower 7 bits of the P register.
SW (SWEEP)	Permits unexecuted instructions to be read from consecutive storage locations. First address of sequence must be previously entered into P. Pressing the TRANSFER switch advances P.
SWEEP PF	Permits page indexes to be read from consecutive Page Index File locations. First address of sequence must be previously entered into the lower 7 bits of the P register.
SW/EN CONT (SWEEP/ENTER CONTINUOUS)	Enables Sweep or Enter operations to proceed continuously through storage or the Page Index File without pressing the TRANSFER switch.

TABLE 7-4. ACCESS KEYBOARD SWITCHES (Cont'd)

SWITCH NAME	FUNCTION
WRITE STO (WRITE STORAGE)	Permits keyboard entry into the storage location specified by the thumb-wheel switches. Entry occurs each time the TRANSFER switch is pressed whether the computer is in the GO mode or stopped.
READ STO (READ STORAGE)	Permits the contents of the storage register location specified by the thumb-wheel switches to be displayed. The display rate is determined by the Step Rate control.
GO	Starts the program execution at the address specified by the P register. Not used for Sweep or Enter operations.
STOP	Stops the computer at the end of the current instruction.
0 THROUGH 7	These switches, when pressed one at a time, allow entry of that particular digit into the Communication register.
TRANSFER	Transfers data in the Communication register to a selected register or storage location.
MC (MASTER CLEAR)	Performs both an internal and external clear. Disabled when GO switch is depressed and the computer is in the GO mode.
KYBD CLR (KEYBOARD CLEAR)	Clears the Communication register.
KYBD OFF (KEYBOARD OFF)	Deactivates all access Keyboard controls.

Breakpoint Switch

The Breakpoint switch is a six-section, eight-position, thumb-wheel switch. The left-hand wheel selects the operating mode, and the other five wheels specify a register number or storage address. There are four mode positions on the mode selector switch with an OFF position between each mode; these modes are BPI, BPO, REG, and STO.

BPI and BPO Modes: The address on the S Bus is continually compared with the instruction or operand address specified by the Breakpoint digit switches. When the selector switch is set to BPI, the computer stops if these values become equal during an RNI (Read Next Instruction) sequence. When the mode selector switch is set to BPO, the computer stops if these values become equal during an ROP (Read Operand) or STO (Store) sequence.

REG and STO Modes: In these two modes, the operator may either monitor the contents of a register location or storage address specified by the thumb-wheel digit switches, or he may store a word in these locations. To monitor a storage location:

- Set the mode selector to REG (register file location) or STO (storage).
- Set the Breakpoint switch to the desired register number or storage address.
- Press the READ STO switch on the keyboard.
- Adjust the Step Rate control to vary the display rate.

The register or storage contents are repeatedly displayed in the Communication register at the selected repetition rate until another keyboard button is pressed to release READ STO. To write a word in storage:

- Set the mode selector to REG or STO.
- Set the Breakpoint switch to the desired register number or storage location.
- Press the WRITE STO switch on the keyboard.
- Enter data into the Communication register by depressing the numeric switches and finally the TRANSFER switch.

The data is entered into the desired storage location or Register File location at the end of the instruction that is currently being executed by the computer. Pressing any other register or mode selector switch releases WRITE STO operation.

Emergency Off Switch

This red momentary switch is used to remove power from the whole computer system in case of a fire or other emergency. It should not be used for a normal po'ver shutdown. Refer to the SOURCE POWER OFF switch description in the Power Control Panel description of this section.

Console Loudspeaker Volume Control

The console loudspeaker and its associated volume control are mounted underneath the console table. The loudspeaker receives its input from the upper 3 bits of the A register. Sound is produced when one or more of these bits are toggled at an audio frequency. Loudspeaker volume is controlled by rotating the volume control knob.

Examples of Keyboard Switch Functions :

1. To enter data into the A register:
    a. Depress the A register switch.

b. Enter all eight digits of the Communication register by depressing the appropriate numeric key switches.*

c. Depress the TRANSFER switch.

d. Depress the KEYBOARD OFF switch.

2. To enter data into the Q register:

   Depress the Q register switch and repeat steps b through d of example 1.

3. To enter the Program Address Counter (P register) with a specific address:

   a. Depress the P register switch.

   b. Enter the lower five digits of the Communication register by depressing the appropriate numeric key switches.

   c. Depress the TRANSFER switch.

   d. Depress the KEYBOARD OFF switch.

4. To enter an operand at a specific address:**

   a. Perform example 3.

   b. Depress the EN switch.

   c. Enter all eight digits of the Communication register by depressing the appropriate numeric key switches.

   d. Depress the TRANSFER switch.

   e. The count in the Program Address Counter has now incremented by one. If data is to be entered into this memory location, repeat steps c and d for as many succeeding entries as required.

   f. Depress the KEYBOARD OFF switch when all data has been entered into the successive group of memory locations.

5. To read an operand from a specific storage address:

   a. Perform example 3.

   b. Depress the SW switch.

   c. Depress the TRANSFER switch.

   d. The contents of the specified storage address are now displayed in the Communication register. (The Program Address Counter is not incremented when the TRANSFER switch is initially depressed.)

*If all eight digit positions of the Communication register are not entered before the Transfer switch is depressed, zeros will be entered into the remaining digit positions.

**The Breakpoint switch may be used in lieu of this operation. (Refer to Example d, Figure 7-9.)

e. If the TRANSFER switch is again depressed, the Program Address Counter is incremented by one, and the contents of the new address are displayed.

f. Depress the KEYBOARD OFF switch when all of the desired memory locations within a successive group have been examined.

NOTE

> Step 5 only permits the operator to examine the contents of specific storage locations. The instructions are not executed during this operation.

6. To enter zeros or another operand into all storage locations:

   a. Depress the EN switch.

   b. Enter all eight digits of the Communication register by depressing the appropriate numeric key switches.

   c. Depress the SW/EN CONT switch.

   d. Depress the STOP switch.

   e. Depress the KEYBOARD OFF switch.

7. The following procedure is applicable for sweeping storage during certain maintenance routines:

   a. Depress the SW switch.

   b. Depress the SW/EN CONT switch. This switch remains engaged until the STOP switch is depressed.

   c. Depress the STOP switch.

   d. Depress the KEYBOARD OFF switch.

8. To enter a 12-bit operand into a specific Page Index File (PIF) address:

   a. Set P to a specific PIF address (000-177) as outlined in example 3. (Only the lower 7-bits of P are recognized.)

   b. Depress the ENTER PF switch.

   c. Enter the lower four digits of the Communication register by depressing the appropriate numeric key switches.

   d. Depress the TRANSFER switch.

   e. The PIF address in the Program Address Counter has now incremented by one. If data is to be entered into this PIF location, repeat steps c and d for as many succeeding entries as required.

   f. Depress the KEYBOARD OFF switch when all data has been entered into the successive group of PIF locations.

9. To read an index from the PIF:

   a. Perform step a of example 8.

   b. Depress the SWEEP PF switch

c. Depress the TRANSFER switch.

d. The specified index of the PIF is now displayed in the lower 12-bits of the Communication register. (The Program Address Counter is not incremented when the TRANSFER switch is initially depressed.)

e. If the TRANSFER switch is again depressed, the Program Address Counter is incremented by one and the index of the new PIF address is displayed.

f. Depress the KEYBOARD OFF switch when all of the desired indexes within a successive group have been examined.

10. To enter zeros or another operand into all indexes of the PIF:

a. Depress the ENTER PF switch

b. Enter the lower four digits of the Communication register by depressing the appropriate numeric key switches.

c. Depress the SW/EN CONT switch. This switch remains engaged until the STOP switch is depressed.

d. Depress the KEYBOARD OFF switch.

11. The following procedure is applicable for sweeping all indexes of the PIF during certain maintenance routines:

a. Depress the SWEEP PF switch

b. Depress the SW/EN CONT switch. This switch remains engaged until the STOP switch is depressed.

c. Depress the STOP switch.

d. Depress the KEYBOARD OFF switch.

Examples of Console Switch Functions:

1. To enter a special routine into the non-Executive mode Auto Load storage area:

a. Depress the MC (Master Clear) keyboard switch.

b. Holding down the keyboard STOP switch, depress the AUTO LOAD switch. Release both switches. The P register should now read 77740. (Holding the STOP switch down prevents the computer from entering the GO mode and executing the previous Auto Load routine.)

c. Depress the ENTER AUTO PROGRAM switch.

d. Depress the keyboard EN switch.

e. Enter the first instruction of the new routine at address 77740 by depressing the appropriate numeric key switches.

f. Depress the keyboard TRANSFER switch.

g. Repeat steps e and f for addresses 77741 through 77757.

h. Depress the MC switch. This clears the registers and cancels the ENTER AUTO PROGRAM function.

i. Depress the KEYBOARD OFF switch.

2. To enter a special routine into the non-Executive mode Auto Dump storage area:

   Repeat steps a through i of example 1 using the AUTO DUMP switch and filling the storage area covered by addresses 77760 through 77777.

3. To execute the Auto Load routine:

   a. Depress the keyboard MC switch.

   b. Depress the AUTO LOAD switch. The computer automatically executes the Auto Load routine and stops when a stop or halt instruction is recognized. The Auto Load function is automatically cleared when the first I/O operation is completed.

4. To execute the Auto Dump routine:

   Perform steps a and b in example 3 but use the AUTO DUMP switch instead of the AUTO LOAD switch.

5. To execute a program at a Auto Step rate:

   a. Set the P register to the first address of the program to be executed.

   b. Depress the AUTO STEP switch.

   c. Adjust the STEP RATE display control.

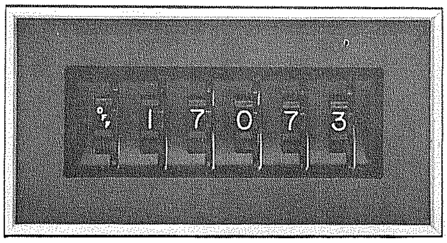
   d. Depress the AUTO STEP switch again to cancel the function and stop program execution. The only way to exit from the Auto Step mode is to depress the AUTO STEP switch again. In the Auto Step mode, halt and jump instructions are executed,but the computer does not stop. Neither will program execution be affected by depressing the STOP switch. The computer continues cycling through memory until the AUTO STEP switch is again depressed.

NOTE

To load or execute a subroutine in the Auto Load or Auto Dump areas while in Executive mode, perform the same operations as for non-Executive mode except that the addresses for the respective areas will be as follows:

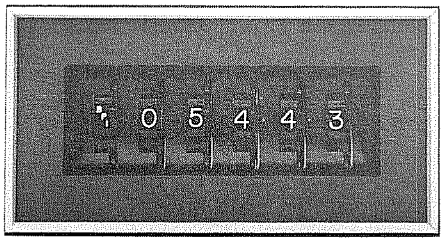
Auto Load:	003700 through 003737
Auto Dump:	003740 through 003777

EXAMPLE A



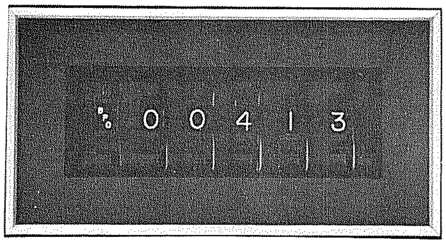
The Breakpoint switch is inoperative whenever an OFF designator is displayed. An OFF designator separates the REG, STO, BPI and BPO positions.

EXAMPLE B



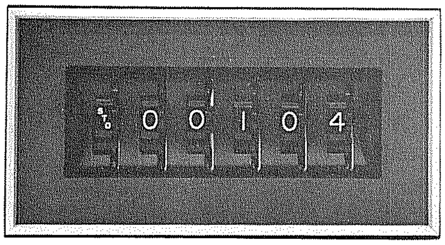
During the normal execution of a program, the computer stops when an RNI is attempted at memory location 05443. A jump to this location also causes the computer to stop. If the program references memory location 05443 for an operand, the computer ignores the Breakpoint switch.

EXAMPLE C



The computer stops only when an attempt is made to read or store an operand at address 00413.

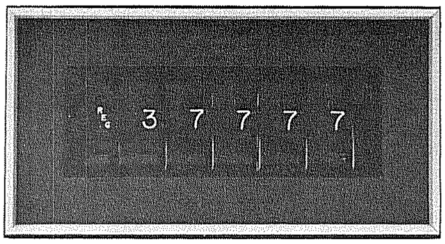
EXAMPLE D



If the WRITE STO switch on the keyboard is depressed and data has been entered into the Communication register, the data is transferred to memory location 00104 when the TRANSFER switch is depressed.

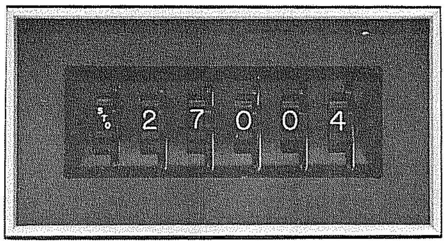
Figure 7-9. Breakpoint Switch Examples

EXAMPLE E



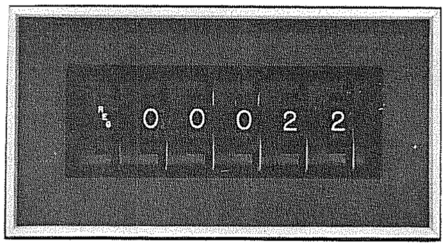
If the WRITE STO switch on the keyboard is depressed and data has been entered into the Communication register, the data will be transferred to register 77 when the TRANSFER switch is depressed. (Only the lower two digits are recognized when the designator switch is in the REG position. The programmer must use caution when writing into the Register File to prevent destruction of other data. Refer to Section 1, Table 1-3.)

EXAMPLE F



If the READ STO switch on the keyboard is depressed, the contents of memory location 27004 are displayed in the Communication register at a repetition rate determined by the Step Rate control. (If the memory location depicted by the Breakpoint switch exceeds the storage capacity of the system, the computer selects the address that corresponds to the storage capacity of the system.)

EXAMPLE G



If the READ STO switch on the keyboard is depressed, the contents of register 22 are displayed in the Communication register at a repetition rate determined by the Step Rate control. (Only the lower two digits are of consequence when the REG designator is displayed. In this case register 22, the real time clock, is being referenced.)

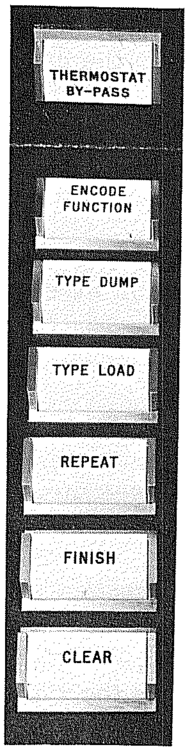
Figure 7-9. Breakpoint Switch Examples (Cont'd)

Typewriter

The console typewriter is an on-line input/output (I/O) device; i.e., it requires no connection to a communication channel and no function codes are issued. The typewriter receives output data directly from storage via the lower 6 bits of the Data Bus. Inputs to storage are handled in the same manner.

Used in conjunction with Block Control and the Register File, the typewriter may be used to enter a block of internal binary-coded characters into storage and to print out data from storage. The two storage addresses that define the limits of the block must be stored in the register file prior to an input or output operation. Register 23 contains the program state number and the initial character address of the block. Register 33 contains the last character address, plus one (refer to Section 1, Table 1-1 notes for Registers 23 and 33 operand formats). Because the initial character address is incremented for each storage reference, it always shows the address of the character currently being stored or dumped. Output operations occur at the rate of 15 characters per second. Input operations are limited by the operator's typing speed.

The console typewriter control switches are shown in Figure 7-10 and their functions are described in Table 7-5.



1758

Figure 7-10. Console Typewriter Control Switches

The general order of events when using the console typewriter for an input or output operation is:

- Check status
- Set registers 23 and 33 of the Register File to the appropriate addresses
- Set tabs, margins and spacing; turn on typewriter
- Clear
- Type out or type in

Status Checking

The programmer may wish to check the status of the typewriter before proceeding. This is done with the Pause instruction. Status response is returned to the computer via two status lines.

The typewriter control transmits two status signals that are checked by the Busy Comparison Mask using the Pause instruction. These status signals are:

Bit 09	Type Finish
Bit 10	Type Repeat

An additional status bit appears on sense line 08. This code is Type Busy and is transmitted by block control in the computation section when a typewriter operation has been selected. If the programmer is certain of the status of the typewriter, this operation may be omitted.

Set Registers 23 and 33

Registers 23 and 33 define the limits of the typewriter I/O operation. These registers are set by instruction or by entering the registers via the Breakpoint switch.

Set Tabs, Margins, and Spacing

All tabs, margins, and paper spacing must be set manually prior to the input or output operation. A tab may be set for each space on the typewriter between margins.

Clear

There are three types of Clears which may be used to clear all conditions (except Encode Function) existing in the typewriter control. These are:

- Internal Clear or a Master Clear

  This signal clears the typewriter control and sets the typewriter to lower case.

TABLE 7-5. CONSOLE TYPEWRITER SWITCHES AND INDICATORS

SWITCH	SWITCH (S) INDICATOR (I)	FUNCTION
ENCODE FUNCTION	S/I	This switch enables the typewriter to send to storage the special function codes for backspace, tab, carriage return, upper-case shift, and lower-case shift.
TYPE LOAD	S/I	This switch allows the computer to receive a block of input data from the typewriter. The TYPE LOAD indicator remains on until either the FINISH, REPEAT, or CLEAR button is pressed, or until the last character of the block has been stored. If the program immediately reactivates the typewriter, it may appear that the light does not go off.
TYPE DUMP	S/I	This switch causes the computer to send data to the typewriter for print-out. It is a momentary contact switch that is illuminated until the last character in the block has been printed or the CLEAR button is pressed.
REPEAT	S/I	This switch is pressed during a Type Load operation to indicate that a typing error occurred. This switch deactivates busy sense line 10 (see PAUS instruction). If the computer does not respond, this light remains on.
FINISH	S/I	This switch is pressed during a Type Load operation to indicate that there is no more data in the current block. This action is necessary if the block that the operator has entered is smaller than the block defined by registers 23 and 33. This switch also deactivates busy sense line 09. If the computer does not respond, this light remains on.
CLEAR	S/I	This switch clears the typewriter controls and sets the typewriter to lower case but does not cancel the ENCODE FUNCTION switch.

- Clear Channel, Search/Move Control, or Type Control instruction (77.51).

  This instruction selectively clears a channel, the S/M control, or, by placing a "1" in bit 08 of the instruction, the typewriter control, and sets the typewriter to lower case.

- Clear Switch on Typewriter

  This switch clears the typewriter control and sets the typewriter to lower case.

Type In and Type Load

Executing the CTI (77.75) instruction or pressing the TYPE LOAD switch on the console or typewriter permits the operator to enter data directly into storage from the typewriter. When the TYPE LOAD indicator on the console glows, the operator may begin typing. The Encode Function switch must be depressed to enable backspace, tab, carriage return, and case shifts to be transmitted to the computer during a typewriter input operation.

Input is in character mode only. As each character is typed, the information is transmitted via the Data Bus to the storage address specified by block control. This address is incremented as characters are transmitted. When the current address equals the terminating address, the TYPE LOAD indicator goes off and the operation is terminated. Data is lost if the operator continues typing after the TYPE LOAD indicator goes off.

Type Out and Type Dump

The typewriter begins to type out when the computation section executes a CTO (77.76) instruction,or when the operator presses the TYPE DUMP switch on the console. Single 6-bit characters are sent from storage to the typewriter via the lower 6 bits of the Data Bus. When the current address equals the terminating address, the TYPE DUMP indicator goes off and the operation is terminated.

During a Type Out operation, the keyboard is locked to prevent loss of data in the event a key is accidentally pressed.

Table 7-6 lists the internal BCD codes, typewriter printout and upper-or lower-case shift that applies to the console typewriter. All character transmission between the computation section and the typewriter is in the form of internal BCD. The typewriter logic makes the necessary conversion to the machine code.

NOTE

Shifting to upper case (57) or lower case (32) is not necessary except on keyboard letters where both upper and lower cases are available. The standard type set has two sets of upper case letters and no lower case letters. This eliminates the need for specifying a case shift.

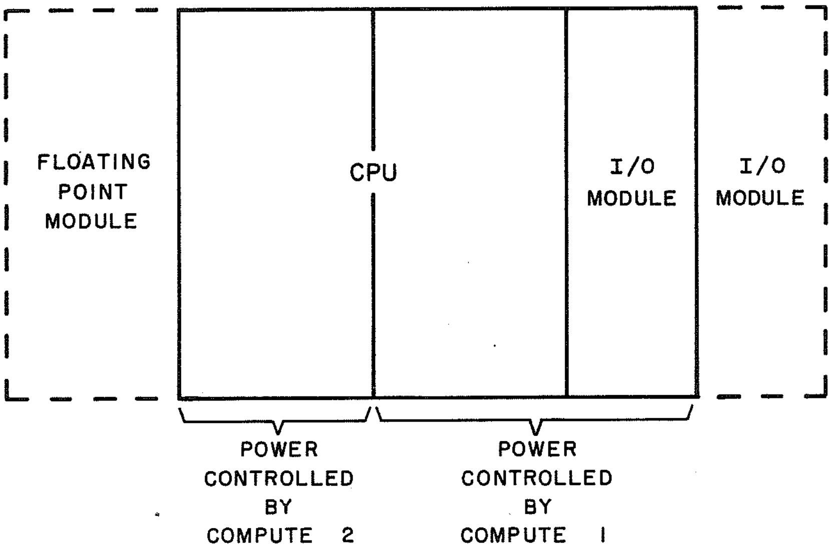
TABLE 7-6. CONSOLE TYPEWRITER CODES

PRINT-OUT	CASE	INTERNAL BCD CODE	PRINT-OUT	CASE	INTERNAL BCD CODE
0	L*	00	-	L	40
1	L	01	J	U or L	41
2	L	02	K	U or L	42
3	L	03	L	U or L	43
4	L	04	M	U or L	44
5	L	05	N	U or L	45
6	L	06	O	U or L	46
7	L	07	P	U or L	47
8	L	10	Q	U or L	50
9	L	11	R	U or L	51
±	U*	12	° (degree)	U	52
=	L	13	$	U	53
"	U	14	*	U	54
:	U	15	#	U	55
;	L	16	%	U	56
?	U	17	(Shift to UC)		57
+	U	20	(Space)		60
A	U or L	21	/	L	61
B	U or L	22	S	U or L	62
C	U or L	23	T	U or L	63
D	U or L	24	U	U or L	64
E	U or L	25	V	U or L	65
F	U or L	26	W	U or L	66
G	U or L	27	X	U or L	67
H	U or L	30	Y	U or L	70
I	U or L	31	Z	U or L	71
(Shift to LC)		32	&	U	72
.	U and L	33	,	U and L	73
)	U	34	(	U	74
'	L	35	(Tab)		75
@	U	36	(Backspace)		76
!	L	37	(Carriage Return)		77

*L = Lower Case; U = Upper Case

POWER CONTROL PANEL

The Power Control Panel module shown in Figure 7-11 controls the logic power supplied to the CPU and the first I/O module. Adjusting the +20 and -20 controls for 0% indication on. their respective meters provides exactly the proper operating power. The following illustration shows which part of the computer the Compute One and Compute Two controls govern.



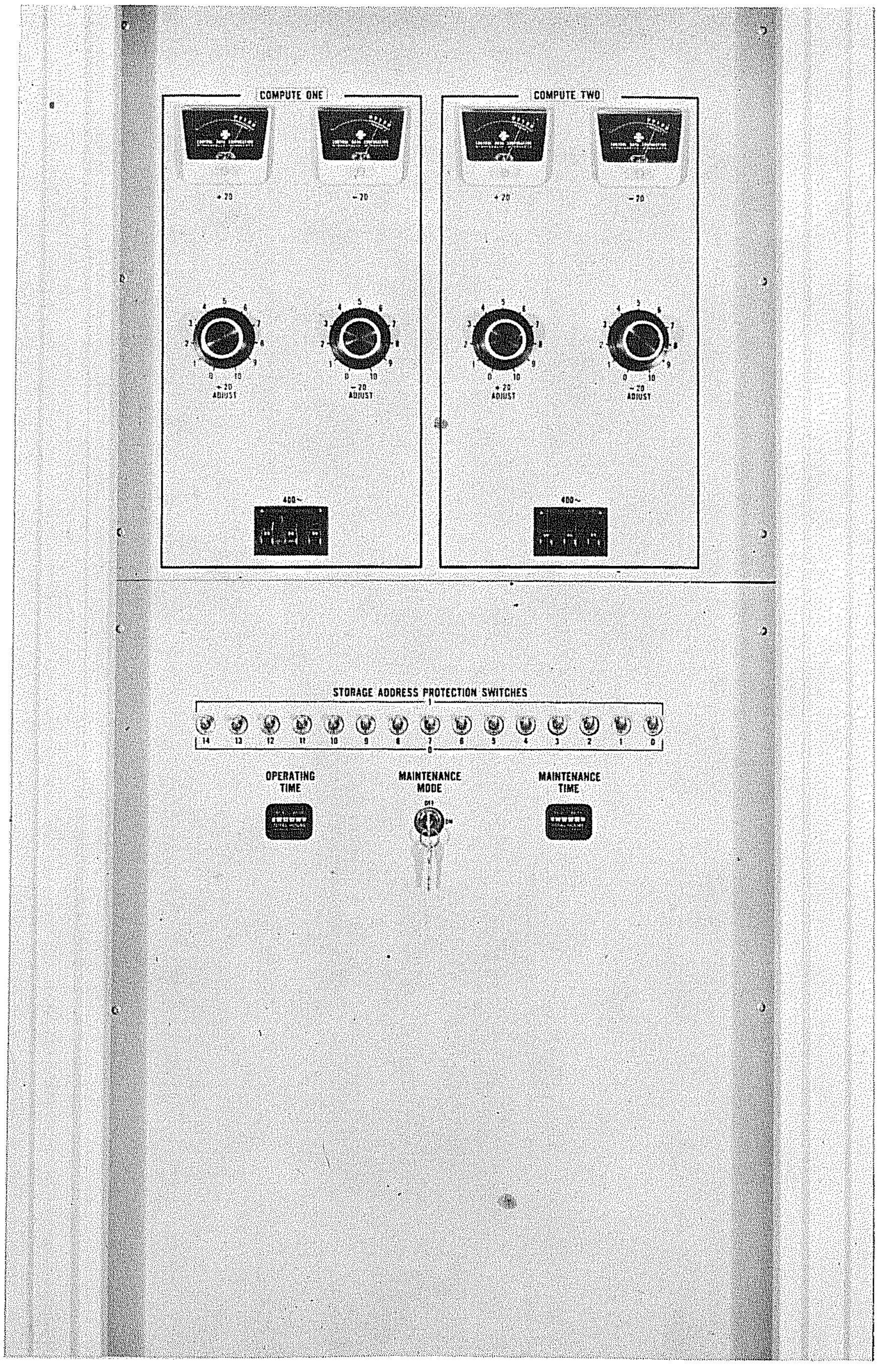
The two main circuit breakers must both be On before the system CPU is operative. Refer to the 3300 Customer Engineering manual for detailed maintenance information.

Elapsed Time Meters

Two elapsed time meters and a key-operated, two-position switch are located on the control panel. Turning the key-operated Maintenance Mode switch to ON connects the Maintenance Time meter to the computer to record the amount of time the computer is used during a maintenance period. Removing the key connects the Operating Time meter to the computer to record normal operating time. Customers renting the computer are often billed according to the time recorded on this meter. The sum of the times recorded on both meters indicates the total computer running time. Only one of the two meters can operate at any one time. Either meter is active for a minimum of one second when a storage cycle occurs.

Storage Protect Switches

The 15 Storage protect switches are described in Section 2.



1763

Figure 7-11. Power Control Panel

8. MULTIPROGRAMMING AND RELOCATION FEATURES

Multiprogramming in the 3300 Computer System enables the instructions of many programs to be sequentially executed by controlled time-sharing operations within a processor. With the Control Data Multiprogramming Modules, throughput is very high due to efficient use of hardware and optimum program scheduling. This feature is very desirable at installations where numerous jobs are run and computing time must be kept at a minimum. Systems equipped with the relocation feature can compute many programs on a time-shared basis or be switched into the non-Executive mode and process jobs according to control card job assignments.

EXECUTIVE MODE

A system equipped with relocation hardware and operating in the Executive Mode functions in either the Monitor State or the Program State.

Monitor State

The Monitor State is the initial operating state of a master cleared processor. The processor also reverts to this state if interrupted for any condition. All instructions may be executed in the Monitor State.

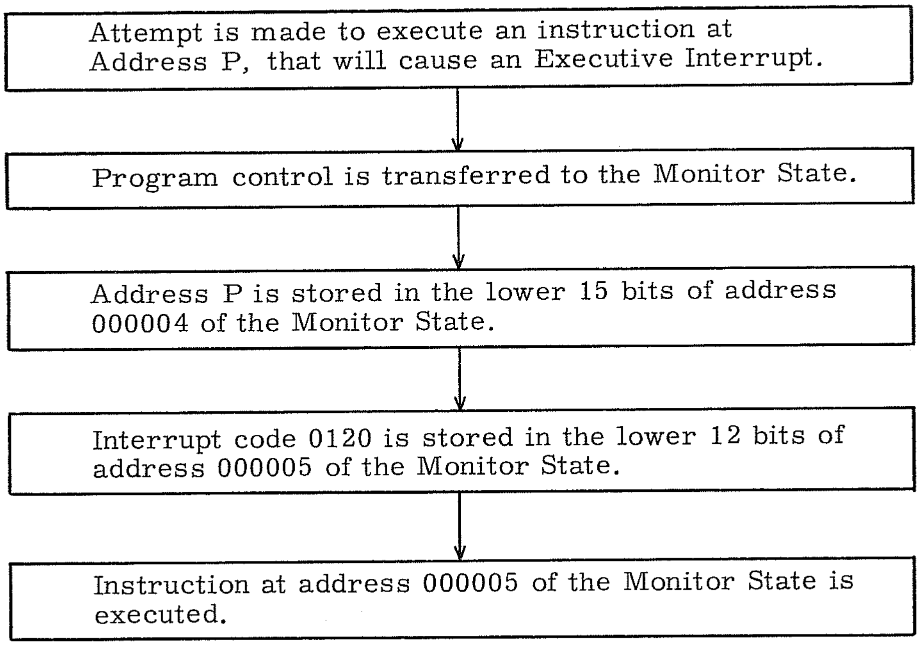
Program State

The Program State permits all but the following instructions to be executed:

1. A Halt instruction (00. 0)
2. Any of the instructions with function codes in the 71-77 range including the UCS, except the SFPF (77. 71) and SBCD (77. 72) instructions.
3. An inter-register transfer instruction that attempts to alter registers 00 through 37 of the register file.

If an attempt to execute one of these instructions occurs, an Executive interrupt is generated and operating control is transferred back to the Monitor State. The Executive interrupt is not masked and the interrupt system need not be enabled to recognize the interrupt when it occurs. Upon recognition, the Executive interrupt transfers program control to the Monitor State. The instruction that caused the interrupt is not executed. The following flow chart describes the sequence of events involved when an Executive Interrupt occurs.

EXECUTIVE INTERRUPT SEQUENCE



Attempt is made to execute an instruction at Address P, that will cause an Executive Interrupt.

Program control is transferred to the Monitor State.

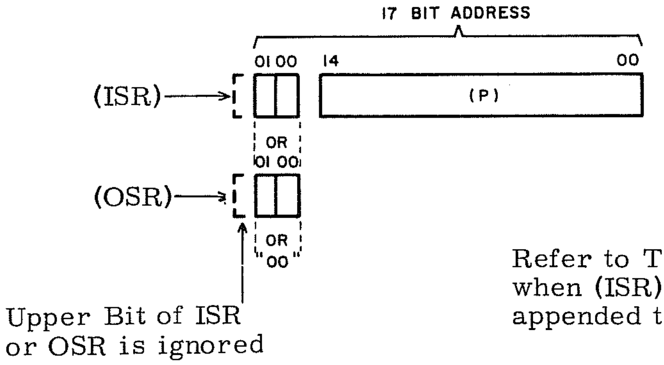
Address P is stored in the lower 15 bits of address 000004 of the Monitor State.

Interrupt code 0120 is stored in the lower 12 bits of address 000005 of the Monitor State.

Instruction at address 000005 of the Monitor State is executed.

MULTIPROGRAMMING AND RELOCATION

If the 3311 Multiprogramming option is not present in a 3300 system, the maximum number of MCS words is 131,072. The actual address referenced is as follows:



Refer to Table 8-1 for conditions when (ISR), (OSR), or zero is appended to address P.

If the APF (77.64) or PFA (77.65) instructions are executed they become no-operation instructions when the 3311 is not present. The keyboard sweep and enter functions with the Page Index File are also disabled. All other operating conditions are the same whether or not the 3311 is in the system.

A 3300 CPU can access up to 262,144 words of core storage when the 3311 Multiprogramming option and appropriate storage modules are present in the system. This is accomplished by augmenting the basic 15-bit address P with a 3-bit state number. The state number, along with a portion of the 15-bit address, becomes the direction path into a relocation path. From the Page Index File the correct page address is obtained for actual memory addressing.

Page Structure

Each page of memory is assigned 2,048 absolute memory locations. A fully expanded system contains 128 of these pages. Individual pages may be subdivided into four partial pages. A 1/4 page consists of 512 address locations. Programs may be allocated full pages, 3/4 page, 1/2 page or 1/4 page of memory.

To facilitiate addressing with the paging scheme, a word organized core matrix is used. This core matrix, called the Page Index File, is referenced by a program during a memory reference to obtain the physical page address or partial page address and provide memory protection.

Address Relocation

Figure 8-1 illustrates address bits at various stages of the relocation process. Those portions of the diagram accompanied by circled numbers are further described in the following numbered paragraphs.

① Program Address and Program Address Group

Any program executed by a 3300 is processed within the confines of a 15-bit program address structure. These 15 bits define the program or operand address related to the routine or subroutine being processed at a given instant. Figures 8-2 and 8.-3 illustrate the significance of these bits in the instruction words for both word addressing and character addressing.

The 15 bits used in word addressing define an absolute address assignment ranging from 00000 to 77777_8. Any program or group of programs within this range of addresses which can be compiled and loaded without conflicting addresses can be considered part of a program address group. Figure 8-4 is illustrative of a program address group consisting of five non-conflicting programs.

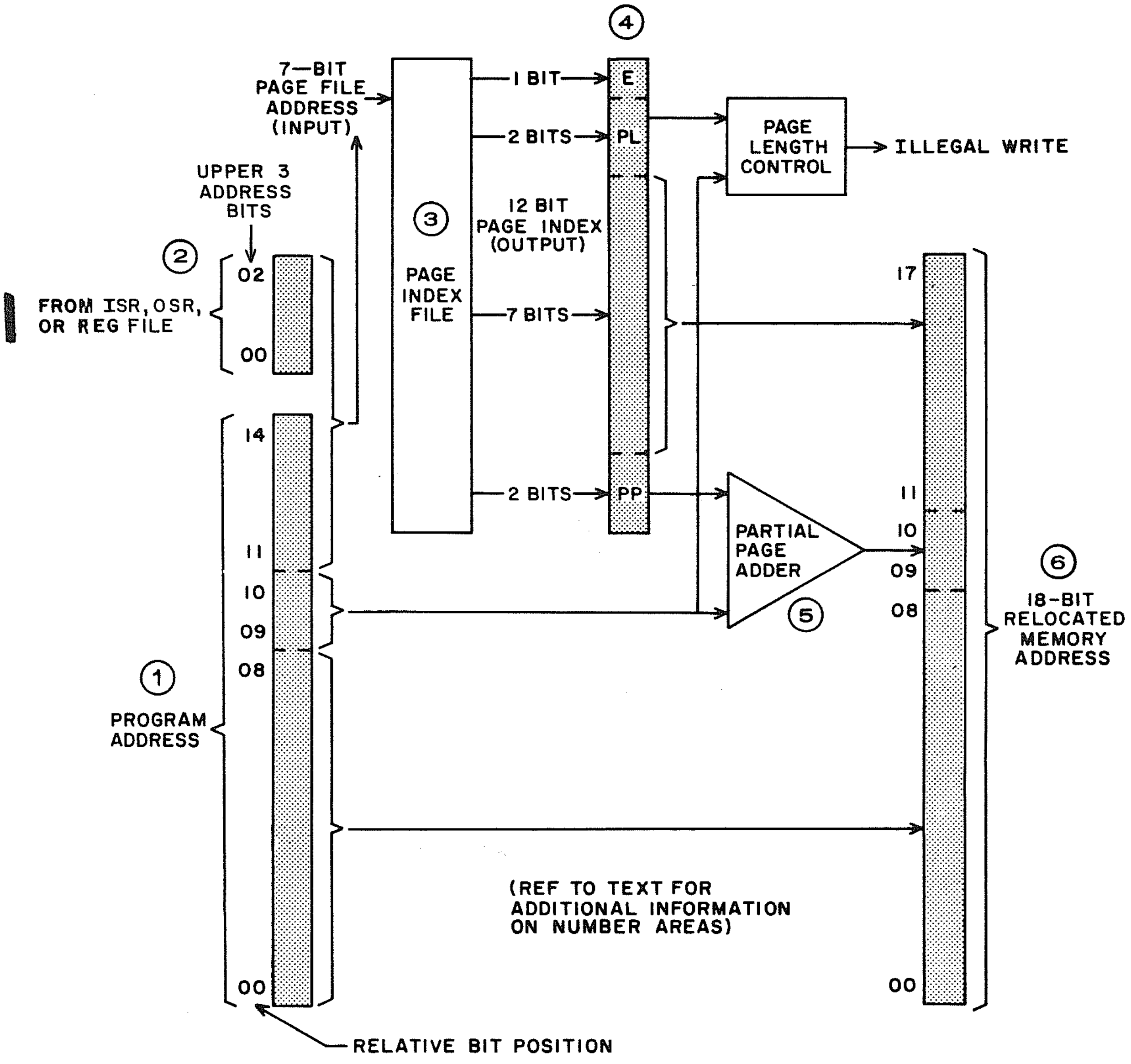


Figure 8-1. Address Relocation Process

A program address group may be considered apart from the physical memory structure since it is a group of sequentially numbered addresses representing one or more programs within 32,768 words of storage and not a discrete physical device. Many program address groups may be contained in storage; however, eight such groups are used in the 3300 to best optimize the memory system.

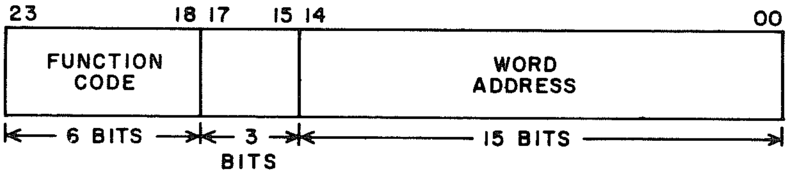


Figure 8-2. Word Addressing

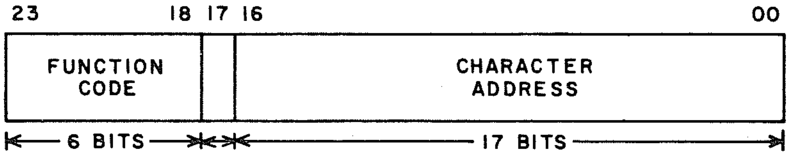


Figure 8-3. Character Addressing

② Upper Three Address Bits

The upper three address bits select 1 of 8 32K program address groups. The ISR and OSR define the specific program address group for all memory references except I/O and Search/Move operand references. The program address group being referenced for instructions and operands can assume any one of eight discrete values by modifying the contents of these 3-bit registers. By transferring dissimilar numbers into these registers, instructions and operands may reference different program address groups.

In I/O and Search/Move instructions a 32K program address group for operand references is selected by the lower three bits of the A register. The address group number must be entered into the A register before the I/O or Search/Move instruction is executed. When the operation starts, the address group number is transferred to a register file location that holds the updated 18-bit operand address throughout the operation.

Table 8-1 shows the source of the upper three address bits for the various operating modes.

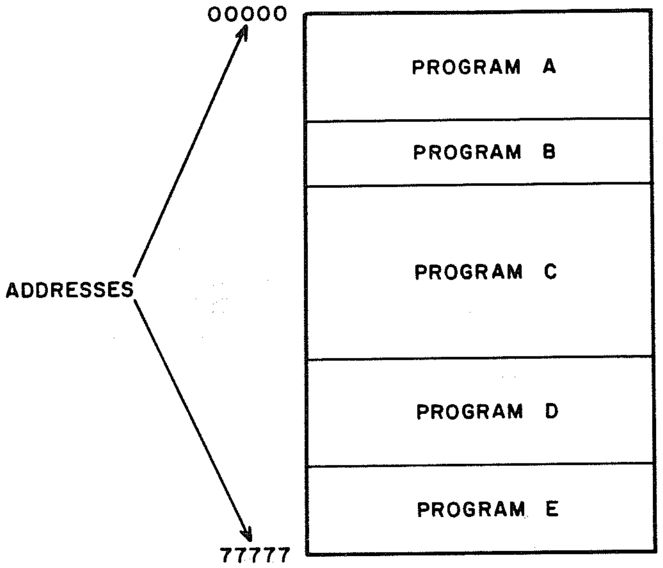


Figure 8-4. Program Address Group

TABLE 8-1. UPPER THREE ADDRESS BITS

Operational State of the Processor	Instruction Address	Operand* Address
Initial Monitor State	Zero	Zero
Monitor State and 55.4 (relocate to operand state) instruction executed	Zero	Contents of OSR
Transition from Monitor State to Program State	Contents of ISR	Contents of ISR
Program State and 55.4 (relocate to operand state) instruction executed	Contents of ISR	Contents of OSR
Program State and 55.0 (relocate to instruction state) instruction executed	Contents of ISR	Contents of ISR
Any interrupt condition to Monitor State	Zero	Zero

*EXCEPTIONS:

1. I/O and Search/Move instructions - Upper 3 bits of operand address originate in bits 0-3 of A register regardless of operating mode.
2. 77.75 (Set Typewriter Input) and 77.76 (Set Typewriter Output) instructions - Upper 3 bits of operand address must be preset in Register File 33 (bits 21-23).

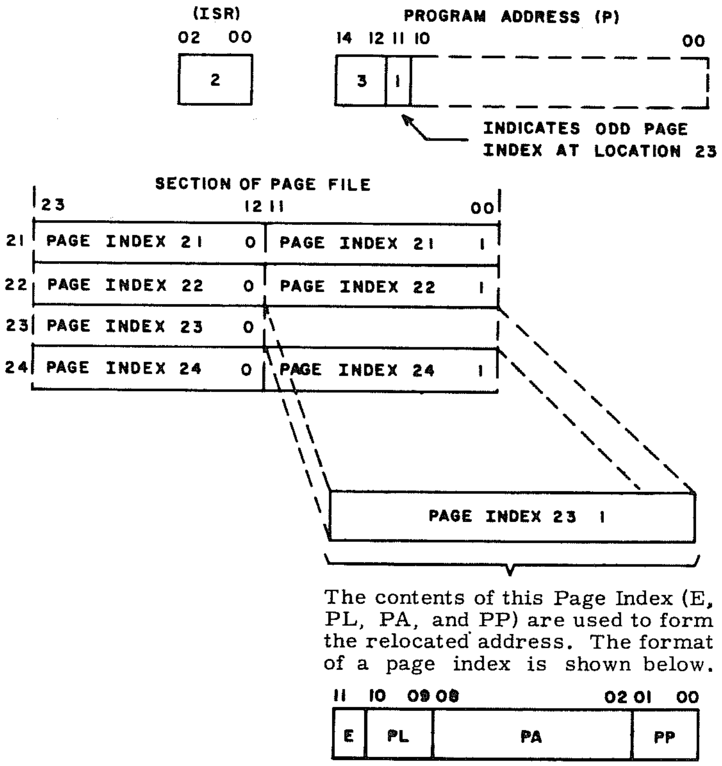
③ Page Index File

The Page Index File is functionally divided into eight distinct reference areas. One area is associated with each of eight possible numbers appearing in the ISR and OSR. Because of this direct relationship, each of the eight program address groups is permanently assigned a reference area in the Page Index File.

Each of the eight reference areas within the Page Index File consists of sixteen 12-bit Page Index Registers. This provides each of the program address groups exclusive use of 16 of these registers. By using the upper 4 bits of the program address for direction to the respective Page Index Registers, a direct and sequential relationship is established between the addresses in a program address group and a specific set of 16 Page Index registers. The Page Index File is actually constructed of 64 24-bit Page Index registers with dual 12-bit indexes. Only one of the 12-bit indexes is used during any specific reference.

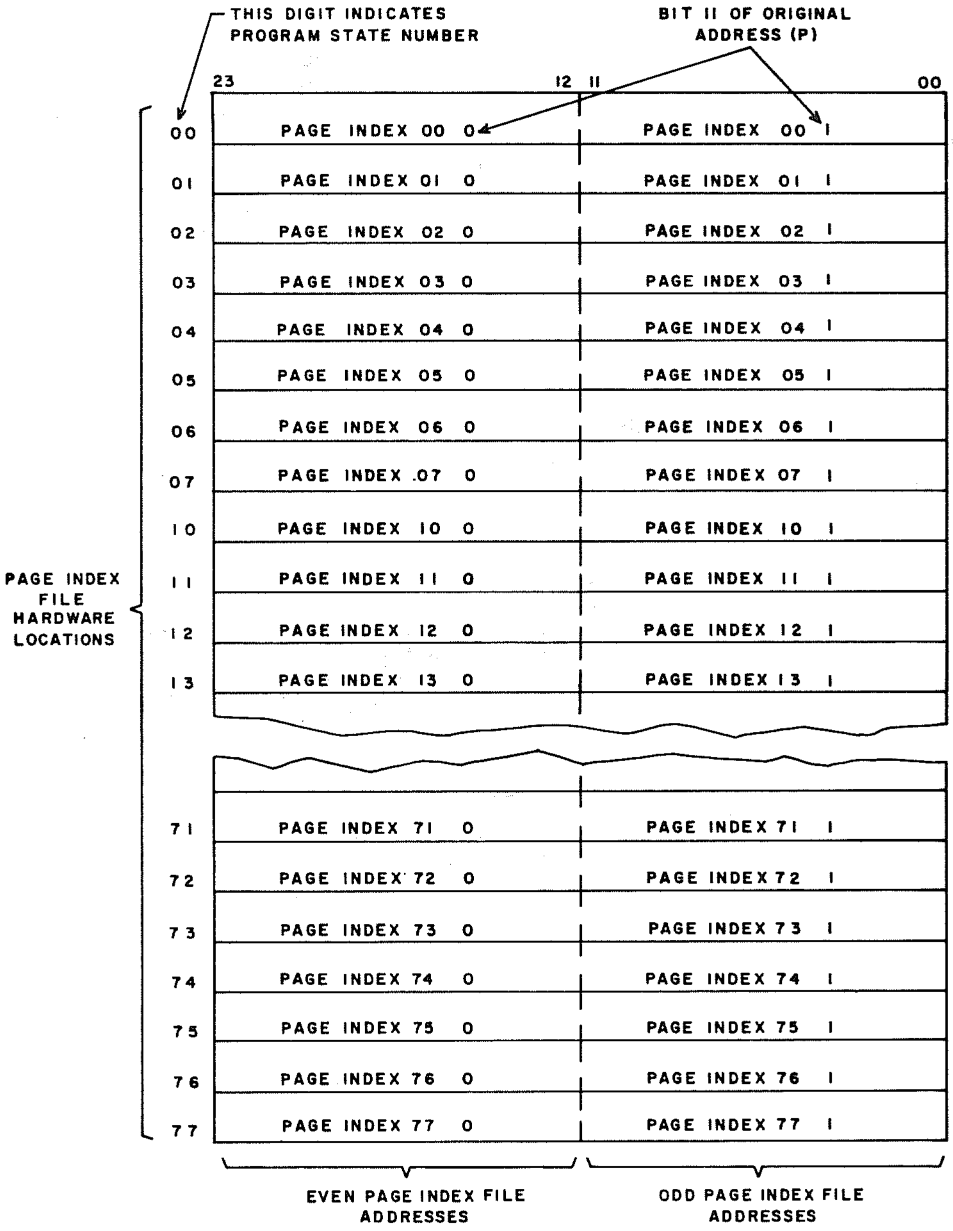
Figure 8-6 depicts the page indexes within the Page Index File and Figure 8-7 illustrates the relationship between program address groups, Page Index File, and a fully expanded core memory.

Bit 11 of the original 15-bit address determines which of the two page indexes at the Page File location will be used. Figure 8-5 shows a specific page index being referenced.


(ISR)
02 00
2
PROGRAM ADDRESS (P)
14 12 11 10 00
3
1
INDICATES ODD PAGE INDEX AT LOCATION 23
SECTION OF PAGE FILE
23 12 11 00
21 PAGE INDEX 21 0 PAGE INDEX 21 1
22 PAGE INDEX 22 0 PAGE INDEX 22 1
23 PAGE INDEX 23 0
24 PAGE INDEX 24 0 PAGE INDEX 24 1
PAGE INDEX 23 1
11 10 09 08 02 01 00
E
PL
PA
PP

The contents of this Page Index (E, PL, PA, and PP) are used to form the relocated address. The format of a page index is shown below.

Figure 8-5. Example of Page Index Referencing


THIS DIGIT INDICATES PROGRAM STATE NUMBER
BIT 11 OF ORIGINAL ADDRESS (P)
PAGE INDEX FILE HARDWARE LOCATIONS
EVEN PAGE INDEX FILE ADDRESSES
ODD PAGE INDEX FILE ADDRESSES

	23 12	11 00
00	PAGE INDEX 00 0	PAGE INDEX 00 1
01	PAGE INDEX 01 0	PAGE INDEX 01 1
02	PAGE INDEX 02 0	PAGE INDEX 02 1
03	PAGE INDEX 03 0	PAGE INDEX 03 1
04	PAGE INDEX 04 0	PAGE INDEX 04 1
05	PAGE INDEX 05 0	PAGE INDEX 05 1
06	PAGE INDEX 06 0	PAGE INDEX 06 1
07	PAGE INDEX .07 0	PAGE INDEX 07 1
10	PAGE INDEX 10 0	PAGE INDEX 10 1
11	PAGE INDEX 11 0	PAGE INDEX 11 1
12	PAGE INDEX 12 0	PAGE INDEX 12 1
13	PAGE INDEX 13 0	PAGE INDEX 13 1
71	PAGE INDEX 71 0	PAGE INDEX 71 1
72	PAGE INDEX 72 0	PAGE INDEX 72 1
73	PAGE INDEX 73 0	PAGE INDEX 73 1
74	PAGE INDEX 74 0	PAGE INDEX 74 1
75	PAGE INDEX 75 0	PAGE INDEX 75 1
76	PAGE INDEX 76 0	PAGE INDEX 76 1
77	PAGE INDEX 77 0	PAGE INDEX 77 1

Figure 8-6. Page Index File Address and Hardware Structure

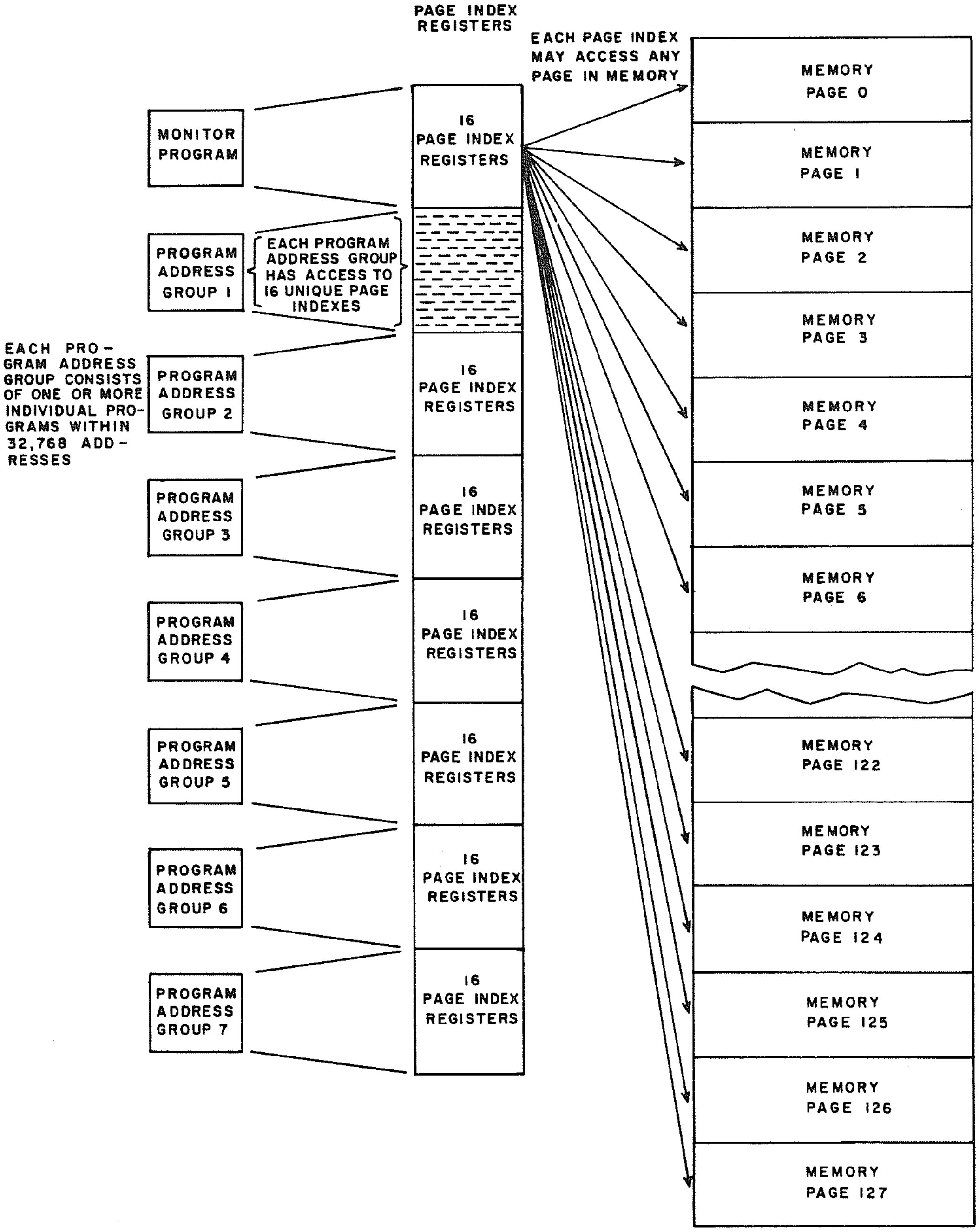
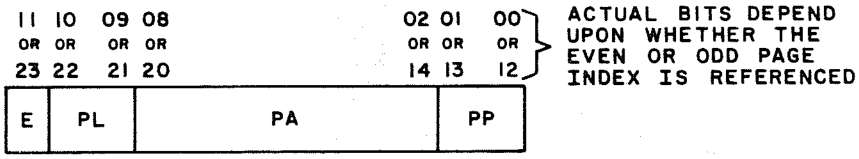


Figure 8-7. Relocation System Illustrating Memory Protection with Fully Expanded Memory (262K)

④ Page Index

Each page index has the same basic format. The significance of each designator during the relocation process is described below. Figure 8-8 shows the format for a page index while Figure 8-9 shows a view of the display panel on the relocation chassis.


11 OR 23
10 OR 22
09 OR 21
08 OR 20
02 OR 14
01 OR 13
00 OR 12
ACTUAL BITS DEPEND UPON WHETHER THE EVEN OR ODD PAGE INDEX IS REFERENCED
E
PL
PA
PP

E = EXCLUSION BIT (1 BIT)
PL = PAGE LENGTH DESIGNATOR (2 BITS)
PA = PAGE ADDRESS DESIGNATOR (7 BITS)
PP = PARTIAL PAGE DESIGNATOR (2 BITS)

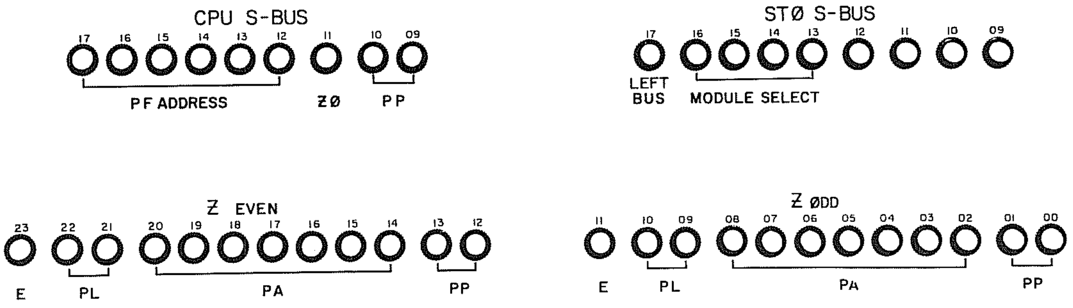
Figure 8-8. Page Index Format

E - Exclusion Bit

This designator may have one of three meanings:

1. If E = "0", the quantity expressed by PA defines a page* where either reading or writing is permitted.
2. If E = "1", and PL, PA or PP is a quantity other than zero, PA defines a page * where only reading is permitted. If a write is attempted, an Illegal Write interrupt is generated.
3. If E = "1" and PL, PA or PP are all equal to zero, an unaddressable page is defined and an Illegal Write interrupt is generated by the Page Index File.

* Refer to descriptions of PL and PP designators for page restrictions.



1587

Figure 8-9. Relocation Chassis Display Panel

PA - Seven bits are used to define the actual memory module being referenced. As stated earlier in this manual, there may be 128 segmented pages in a 3300 system with 262,144 words of core storage. Each page has a unique page address and addresses 000 through 177_8 define all of the possible pages.

A 3300 system with a fully expanded storage network has two address busses. Each bus has access to 131,072 words of the total 262,144 storage words. The uppermost bit of PA (bit 17 in the relocated address) determines which bus (right or left) is selected. This bit will be a "1" when the left bus is used and a "0" when the right bus is selected. Figure 8 -10 depicts the bus address system.

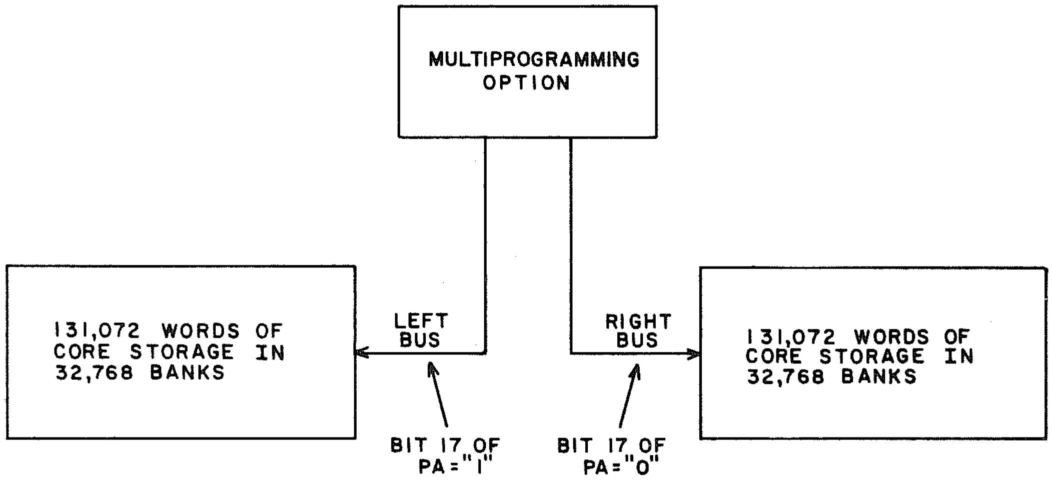
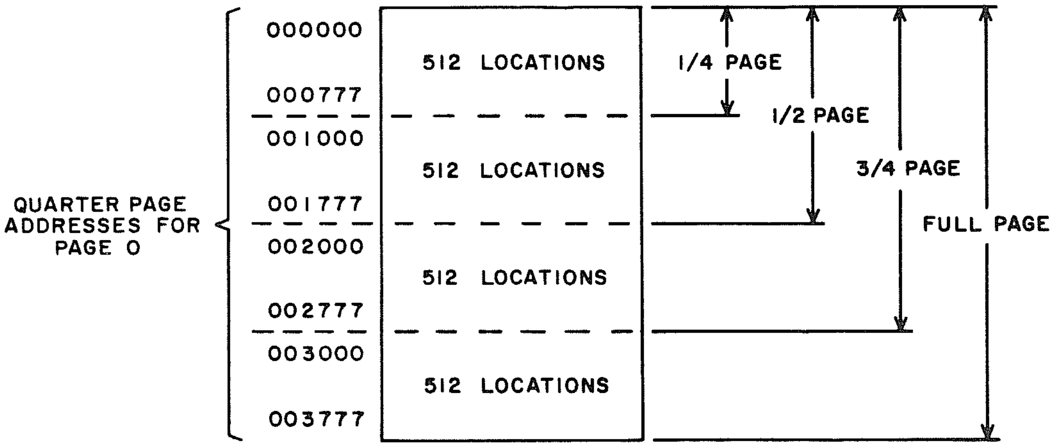

MULTIPROGRAMMING OPTION
131,072 WORDS OF CORE STORAGE IN 32,768 BANKS
LEFT BUS
RIGHT BUS
131,072 WORDS OF CORE STORAGE IN 32,768 BANKS
BIT 17 OF PA="1"
BIT 17 OF PA="0"

Figure 8-10. Storage Address Buses



NOTE: PP = 0 FOR THIS EXAMPLE

PL = 0 FOR FULL PAGE   PL = 2 FOR 1/2 PAGE

PL = 1 FOR 1/4 PAGE   PL = 3 FOR 3/4 PAGE

Figure 8 -11. Page Length Subdivisions

PL - Each page has 2,048 memory locations and is subdivided into quarters of 512 locations each. The PL designator defines how many quarters of a page can be referenced (beginning with the starting quarter specified by PP). A program is assigned the number of quarter pages it needs to reside in memory. Figure 8-11 illustrates the quarter sections of a page and the significance of the PL bits.

PP - The Partial Page designator is the address of the physical quarter page that will serve as the starting point of the page. Example A (Figure 8-12) shows the quarter page referenced for each of the PP designators. The significance of the PP designator in selecting the respective quarter page for addressing is described below.

EXAMPLE A

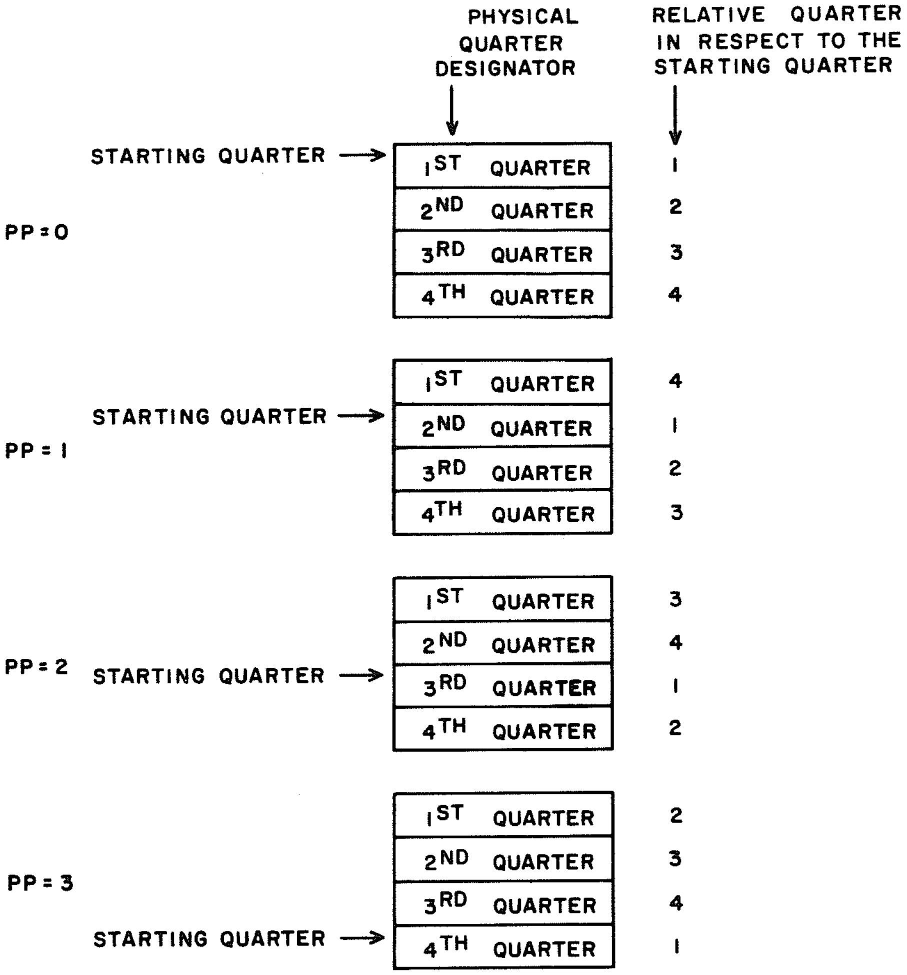


Figure 8-12. Quarter Page in Relation to PP Designator

If PP = 0, the relative page begins in the 1st physical quarter

If PP = 1, the relative page begins in the 2nd physical quarter

If PP = 2, the relative page begins in the 3rd physical quarter

If PP = 3, the relative page begins in the 4th physical quarter

⑤ Partial Page Adder

A special adder is used to combine the PP designator from the page index with bits 9 and 10 of the original address. The partial sum indicates the address of the physical quarter in which referencing will begin. Example B and Figure 8-13 show the actual quarter page in which addressing occurs for specific PL, PP, and bits 9 and 10 values.

EXAMPLE B

PL = 0

PP = 1

Bits 9 and 10 = 2

Analysis: A full page (PL = 0) is allocated, the relative page begins in the second physical quarter, and referencing begins in the fourth physical quarter, (physical quarter address 3).

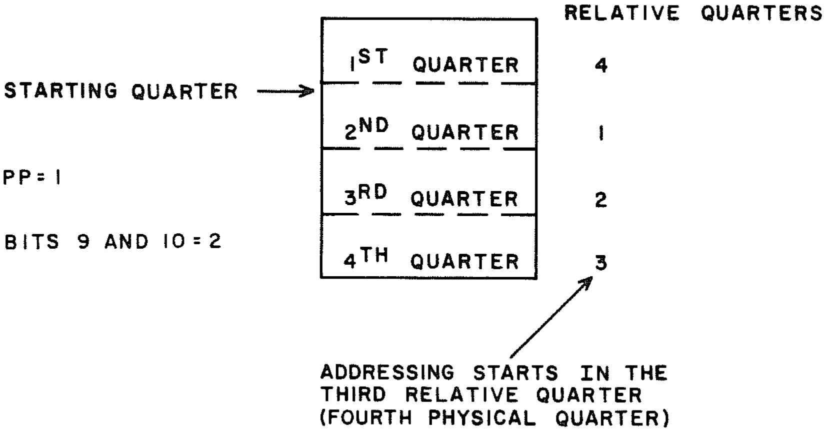


Figure 8-13. Starting Quarters versus Relative Quarters

It should be noted that if bits 9 and 10 of the original address specify a quarter page equal to or greater than that of the PL designator when PL ≠ zero, an Illegal Write interrupt will occur. An example of this condition would be a 1/4 page allocated but bits 9 and 10 equal to 3, thus specifying an address in the fourth quarter.

This interrupt will not occur during Monitor State or I/O operations.

⑥ Relocated Address

The 18-bit relocated address defines the actual core storage location being referenced.

The PA portion of the page index fills the upper seven bits of this address (S) bus to use and select the appropriate storage module. Bits 9 and 10 receive the output of the adder previously described and indicate the physical quarter page being referenced. The lower nine bits are unaltered from the original address and comprise the remainder of the relocated address.

Page Zero Consideration

If page Index File address zero is referenced in either the Program or Monitor state, the PA and PP designators for this page index will always be zero. As a result of this condition, page zero, which encompasses addresses 000000 through 003777, can be accessed and used for storing the Auto Load and Auto Dump routines. The Auto Load routine is contained in addresses 003700 through 003737 and the Auto Dump routine is stored in addresses 003740 through 003777.

APPENDIX A

CONTROL DATA 3100, 3200, 3300 COMPUTER SYSTEMS CHARACTER SET AND BCD/ASCII CODE CONVERSIONS

CONTROL DATA 3100, 3200, 3300 COMPUTER SYSTEMS CHARACTER SET

INTERNAL BCD CODES	EXTERNAL BCD CODES	CONSOLE TYPEWRITER CHARACTERS (USES INTERNAL BCD ONLY)	MAGNETIC TAPE UNIT CHARACTERS	PUNCHED CARD CODES
00	12	0 (zero)	0 (zero)	0
01	01	1	1	1
02	02	2	2	2
03	03	3	3	3
04	04	4	4	4
05	05	5	5	5
06	06	6	6	6
07	07	7	7	7
10	10	8	8	8
11	11	9	9	9
12	(illegal)	±	---	2, 8
13	13	=	#	3, 8
14	14	"	@	4, 8
15	15	:	---	5, 8
16	16	;	---	6, 8
17	17	?	(file mark)	7, 8
20	60	+	&	12
21	61	A	A	12, 1
22	62	B	B	12, 2
23	63	C	C	12, 3
24	64	D	D	12, 4
25	65	E	E	12, 5
26	66	F	F	12, 6
27	67	G	G	12, 7
30	70	H	H	12, 8
31	71	I	I	12, 9
32	72	(Shift to lower case)	+0	12, 0
33	73	.(period)	.	12, 3, 8
34	74	)	⌑	12, 4, 8
35	75	'(apostrophe)	---	12, 5, 8
36	76	@	---	12, 6, 8
37	77	!	---	12, 7, 8

(Continued on next page)

INTERNAL BCD CODES	EXTERNAL BDC CODES	CONSOLE TYPEWRITER CHARACTERS (USES INTERNAL BCD ONLY)	MAGNETIC TAPE UNIT CHARACTERS	PUNCHED CARD CODES
40	40	-(minus)	-(minus)	11
41	41	J	J	11, 1
42	42	K	K	11, 2
43	43	L	L	11, 3
44	44	M	M	11, 4
45	45	N	N	11, 5
46	46	O	O	11, 6
47	47	P	P	11, 7
50	50	Q	Q	11, 8
51	51	R	R	11, 9
52	52	° (degree)	-O	11, 0
53	53	$	$	11, 3, 8
54	54	*	*	11, 4, 8
55	55	#	---	11, 5, 8
56	56	%	---	11, 6, 8
57	57	(Shift to upper case)	---	11, 7, 8
60	20	(space)	(blank)	(blank)
61	21	/	/	0, 1
62	22	S	S	0, 2
63	23	T	T	0, 3
64	24	U	U	0, 4
65	25	V	V	0, 5
66	26	W	W	0, 6
67	27	X	X	0, 7
70	30	Y	Y	0, 8
71	31	Z	Z	0, 9
72	32	&	---	0, 2, 8
73	33	, (comma)	, (comma)	0, 3, 8
74	34	(	%	0, 4, 8
75	35	(tab)	---	0, 5, 8
76	36	(backspace)	---	0, 6, 8
77	37	(carriage return)	---	0, 7, 8

BCD/ASCII CONVERSION TABLE

6-BIT BCD CODE	8-BIT ASCII CHARACTER	BINARY STATUS OF ASCII CHARACTER (BIT POSITIONS)							
		7*	6	5	4	3	2	1	0
00	0	0	0	1	1	0	0	0	0
01	1	0	0	1	1	0	0	0	1
02	2	0	0	1	1	0	0	1	0
03	3	0	0	1	1	0	0	1	1
04	4	0	0	1	1	0	1	0	0
05	5	0	0	1	1	0	1	0	1
06	6	0	0	1	1	0	1	1	0
07	7	0	0	1	1	0	1	1	1
10	8	0	0	1	1	1	0	0	0
11	9	0	0	1	1	1	0	0	1
12	:	0	0	1	1	1	0	1	0
13	=	0	0	1	1	1	1	0	1
14	'	0	0	1	0	0	1	1	1
15	&	0	0	1	0	0	1	1	0
16	%	0	0	1	0	0	1	0	1
17	[	0	1	0	1	1	0	1	1
20	+	0	0	1	0	1	0	1	1
21	A	0	1	0	0	0	0	0	1
22	B	0	1	0	0	0	0	1	0
23	C	0	1	0	0	0	0	1	1
24	D	0	1	0	0	0	1	0	0
25	E	0	1	0	0	0	1	0	1
26	F	0	1	0	0	0	1	1	0
27	G	0	1	0	0	0	1	1	1
30	H	0	1	0	0	1	0	0	0
31	I	0	1	0	0	1	0	0	1
32	<	0	0	1	1	1	1	0	0
33	.	0	0	1	0	1	1	1	0
34	)	0	0	1	0	1	0	0	1
35	∧	0	1	0	1	1	1	1	0
36	"	0	0	1	0	0	0	1	0
37	;	0	0	1	1	1	0	1	1

*ASCII bit 7 is unassigned and "0" for all codes.

BCD/ASCII CONVERSION TABLE (Cont'd)

6-BIT BCD CODE	8-BIT ASCII CHARACTER	BINARY STATUS OF ASCII CHARACTER (BIT POSITIONS)							
		7*	6	5	4	3	2	1	0
40	-	0	0	1	0	1	1	0	1
41	J	0	1	0	0	1	0	1	0
42	K	0	1	0	0	1	0	1	1
43	L	0	1	0	0	1	1	0	0
44	M	0	1	0	0	1	1	0	1
45	N	0	1	0	0	1	1	1	0
46	O	0	1	0	0	1	1	1	1
47	P	0	1	0	1	0	0	0	0
50	Q	0	1	0	1	0	0	0	1
51	R	0	1	0	1	0	0	1	0
52	!	0	0	1	0	0	0	0	1
53	$	0	0	1	0	0	1	0	0
54	*	0	0	1	0	1	0	1	0
55	#	0	0	1	0	0	0	1	1
56	\	0	1	0	0	0	0	0	0
57	>	0	0	1	1	1	1	1	0
60	Blank	0	0	1	0	0	0	0	0
61	/	0	0	1	0	1	1	1	1
62	S	0	1	0	1	0	0	1	1
63	T	0	1	0	1	0	1	0	0
64	U	0	1	0	1	0	1	0	1
65	V	0	1	0	1	0	1	1	0
66	W	0	1	0	1	0	1	1	1
67	X	0	1	0	1	1	0	0	0
70	Y	0	1	0	1	1	0	0	1
71	Z	0	1	0	1	1	0	1	0
72	]	0	1	0	1	1	1	0	1
73	Comma	0	0	1	0	1	1	0	0
74	(	0	0	1	0	1	0	0	0
75	~	0	1	0	1	1	1	0	0
76	_	0	1	0	1	1	1	1	1
77	?	0	0	1	1	1	1	1	1

*ASCII bit 7 is unassigned and "0" for all codes.

APPENDIX B

SUPPLEMENTARY ARITHMETIC INFORMATION

B. SUPPLEMENTARY ARITHMETIC INFORMATION

NUMBER SYSTEMS

Any number system may be defined by two characteristics, the radix or base and the modulus. The radix or base is the number of unique symbols used in the system. The decimal system has ten symbols, 0 through 9. Modulus is the number of unique quantities or magnitudes a given system can distinguish. For example, an adding machine with ten digits, or counting wheels, would have a modulus of $10^{10}-1$. The decimal system has no modulus because an infinite number of digits can be written, but the adding machine has a modulus because the highest number which can be expressed is 9, 999, 999, 999.

Most number systems are positional; that is, the relative position of a symbol determines its magnitude. In the decimal system, a 5 in the units column represents a different quantity than a 5 in the tens column. Quantities equal to or greater than 1 may be represented by using the 10 symbols as coefficients of ascending powers of the base 10. The number 984_{10} is:

$$\begin{array}{rcrcr} 9 \times 10^2 & = & 9 \times 100 & = & 900 \\ +8 \times 10^1 & = & 8 \times 10 & = & 80 \\ +4 \times 10^0 & = & 4 \times 1 & = & \underline{4} \\ & & & & 984_{10} \end{array}$$

Quantities less than 1 may be represented by using the 10 symbols as coefficients of ascending negative powers of the base 10. The number 0.593_{10} may be represented as:

$$\begin{array}{rcrcl} 5 \times 10^{-1} & = & 5 \times .1 & = & .5 \\ +9 \times 10^{-2} & = & 9 \times .01 & = & .09 \\ +3 \times 10^{-3} & = & 3 \times .001 & = & \underline{.003} \\ & & & & 0.593_{10} \end{array}$$

Binary Number System

Computers operate faster and more efficiently by using the binary number system. There are only two symbols, 0 and 1; the base = 2. The following shows the positional value:

$$\begin{array}{ccccccccc} \ldots & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 & & \\ & 32 & 16 & 8 & 4 & 2 & 1 & & \text{Binary point} \end{array}$$

The binary number 0 1 1 0 1 0 represents:

$$\begin{array}{rclcr} 0 \times 2^5 & = & 0 \times 32 & = & 0 \\ +1 \times 2^4 & = & 1 \times 16 & = & 16 \\ +1 \times 2^3 & = & 1 \times 8 & = & 8 \\ +0 \times 2^2 & = & 0 \times 4 & = & 0 \\ +1 \times 2^1 & = & 1 \times 2 & = & 2 \\ +0 \times 2^0 & = & 0 \times 1 & = & \underline{0} \\ & & & & 26_{10} \end{array}$$

Fractional binary numbers may be represented by using the symbols as coefficients of ascending negative powers of the base.

	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5} ...
Binary Point	1/2	1/4	1/8	1/16	1/32

The binary number 0. 10 110 may be represented as:

$$\begin{array}{rclclcl} 1 \times 2^{-1} & = & 1 \times 1/2 & = & 1/2 & = & 8/16 \\ +0 \times 2^{-2} & = & 0 \times 1/4 & = & 0 & = & 0 \\ +1 \times 2^{-3} & = & 1 \times 1/8 & = & 1/8 & = & 2/16 \\ +1 \times 2^{-4} & = & 1 \times 1/16 & = & 1/16 & = & \underline{1/16} \\ & & & & & & 11/16_{10} \end{array}$$

Octal Number System

The octal number system uses eight discrete symbols, 0 through 7. With base eight the positional value is:

...	8^5	8^4	8^3	8^2	8^1	8^0
	32,768	4,096	512	64	8	1

The octal number 513_8 represents:

$$\begin{array}{rclcr} 5 \times 8^2 & = & 5 \times 64 & = & 320 \\ +1 \times 8^1 & = & 1 \times 8 & = & 8 \\ +3 \times 8^0 & = & 3 \times 1 & = & \underline{3} \\ & & & & 331_{10} \end{array}$$

Fractional octal numbers may be represented by using the symbols as coefficients of ascending negative powers of the base.

8^{-1}	8^{-2}	8^{-3}	8^{-4}	...
1/8	1/64	1/512	1/4096	

The octal number 0. 4520 represents:

$$\begin{array}{rclcr} 4 \times 8^{-1} & = & 4 \times 1/8 & = & 256/512 \\ +5 \times 8^{-2} & = & 5 \times 1/64 & = & 40/512 \\ +2 \times 8^{-3} & = & 2 \times 1/512 & = & \underline{2/512} \\ & & & & 298/512 = 149/256_{10} \end{array}$$

ARITHMETIC

Addition and Subtraction

Binary numbers are added according to the following rules:

$0 + 0 = 0$
$0 + 1 = 1$
$1 + 0 = 1$
$1 + 1 = 0$ with a carry of 1

The addition of two binary numbers proceeds as follows (the decimal equivalents verify the result):

Augend	0111	(7)
Addend	+0100	+(4)
Partial Sum	0011	
Carry	1	
Sum	1011	(11)

Subtraction may be performed as an addition:

8 (minuend)		8 (minuend)
-6 (subtrahend)	or	+4 (10's complement of subtrahend)
2 (difference)		2 (difference - omit carry)

The second method shows subtraction performed by the "adding the complement" method. The omission of the carry in the illustration has the effect of reducing the result by 10.

One's Complement: The computer performs all arithmetic and counting operations in the binary one's complement mode. In this system, positive numbers are represented by the binary equivalent and negative numbers in ones' complement notation.

The one's complement representation of a number is found by subtracting each bit of the number from 1. For example:

1111	
-1001	9
0110	(one's complement of 9)

This representation of a negative binary quantity may also be obtained by substituting "1's" for "0's" and "0's" for "1's".

The value zero can be represented in one's complement notation in two ways:

$0000 \rightarrow 00_2$ Positive (+) Zero
$1111 \rightarrow 11_2$ Negative (-) Zero

The rules regarding the use of these two forms for computation are:

- Both positive and negative zero are acceptable as arithmetic operands.
- If the result of an arithmetic operation is zero, it will be expressed as positive zero.

One's complement notation applies not only to arithmetic operations performed in A, but also to the modification of execution addresses in the F register. During address modification, the modified address will equal 77777_8 only if the unmodified execution address equals 77777_8 and b = 0 or $(B^b) = 77777_8$.

Multiplication

Binary multiplication proceeds according to the following rules:

$$0 \times 0 = 0$$
$$0 \times 1 = 0$$
$$1 \times 0 = 0$$
$$1 \times 1 = 1$$

Multiplication is always performed on a bit-by-bit basis. Carries do not result from multiplication, since the product of any two bits is always a single bit.

Decimal example:

Multiplicand	14	
Multiplier	12	
Partial Products	28	
	14	(shifted one place left)
Product	168_{10}	

The shift of the second partial product is a shorthand method for writing the true value 140.

Binary example:

Multiplicand (14)	1110	
Multiplier (12)	1100	
	0000	
	0000	shift to place
Partial Products	1110	digits in proper
	1110	columns
Product (168_{10})	10101000_2	

The computer determines the running subtotal of the partial products. Rather than shifting the partial product to the left to position it correctly, the computer right shifts the summation of the partial products one place before the next addition is made. When the multiplier bit is "1", the multiplicand is added to the running total and the results are shifted to the right one place. When the multiplier bit is "0", the partial product subtotal is shifted to the right (in effect, the quantity has been multiplied by 10_2).

Division

The following examples shows the familiar method of decimal division:

```
                    14      Quotient
Divisor     13  | 185       Dividend
                  13
                   55       Partial Dividend
                   52
                    3       Remainder
```

The computer performs division in a similar manner (using binary equivalents):

```
                          1110     Quotient (14)
Divisor      1101   | 10111001     Dividend
                      1101
                      10100
                       1101
                         1110      Partial Dividends
                         1101
                           11      Remainder (3)
```

However, instead of shifting the divisor right to position it for subtraction from the partial dividend (shown above), the computer shifts the partial dividend left, accomplishing the same purpose and permitting the arithmetic to be performed in the A register. The computer counts the number of shifts, which is the number of quotient digits to be obtained; after the correct number of counts, the routine is terminated.

CONVERSIONS

The procedures that may be used when converting from one number system to another are power addition, radix arithmetic, and substitution.

TABLE B-1. RECOMMENDED CONVERSION PROCEDURES (INTEGER AND FRACTIONAL)

Conversion	Recommended Method
Binary to Decimal	Power Addition
Octal to Decimal	Power Addition
Decimal to Binary	Radix Arithmetic
Decimal to Octal	Radix Arithmetic
Binary to Octal	Substitution
Octal to Binary	Substitution
GENERAL RULES	
$r_i > r_f$: use Radix Arithmetic, Substitution	
$r_i < r_f$: use Power Addition, Substitution	
r_i = Radix of initial system	
r_f = Radix of final system	

Power Addition

To convert a number from r_i to r_f ($r_i < r_f$) write the number in its expanded r_i polynomial form and simplify using r_f arithmetic.

EXAMPLE 1 Binary to Decimal (Integer)

$$\begin{aligned}010\ 111_2 &= 1(2^4) +0(2^3) +1(2^2) +1(2^1) +1(2^0)\\ &= 1(16) +0(8) +1(4) +1(2) +1(1)\\ &= 16 +0 +4 +2 +1\\ &= 23_{10}\end{aligned}$$

EXAMPLE 2 Binary to Decimal (Fractional)

$$\begin{aligned}.0101_2 &= 0(2^{-1}) +1(2^{-2}) +0(2^{-3}) +1(2^{-4})\\ &= 0 +1/4 +0 +1/16\\ &= 5/16_{10}\end{aligned}$$

EXAMPLE 3 Octal to Decimal (Integer)

$$\begin{aligned}324_8 &= 3(8^2) +2(8^1) +4(8^0)\\ &= 3(64) +2(8) +4(1)\\ &= 192 +16 +4\\ &= 212_{10}\end{aligned}$$

EXAMPLE 4 Octal to Decimal (Fractional)

$$\begin{aligned}.44_8 &= 4(8^{-1}) +4(8^{-2})\\ &= 4/8 +4/64\\ &= 36/64_{10}\\ &= 9/16_{10}\end{aligned}$$

Radix Arithmetic

To convert a whole number from r_i to r_f ($r_i > r_f$):

- Divide r_i by r_f using r_i arithmetic
- The remainder is the lowest order bit in the new expression
- Divide the integral part from the previous operation by r_f
- The remainder is the next higher order bit in the new expression
- The process continues until the division produces only a remainder which will be the highest order bit in the r_f expression.

To convert a fractional number from r_i to r_f:

- Multiply r_i by r_f using r_i arithmetic
- The integral part is the highest order bit in the new expression
- Multiply the fractional part from the previous operation by r_f
- The integral part is the next lower order bit in the new expression
- The process continues until sufficient precision is achieved or the process terminates.

EXAMPLE 1 Decimal to Binary (Integer)

45 ÷ 2 = 22 remainder 1; record	1
22 ÷ 2 = 11 remainder 0; record	0
11 ÷ 2 = 5 remainder 1; record	1
5 ÷ 2 = 2 remainder 1; record	1
2 ÷ 2 = 1 remainder 0; record	0
1 ÷ 2 = 0 remainder 1; record	1
Thus: $45_{10} = 101101_2$	101101

EXAMPLE 2 Decimal to Binary (Fractional)

.25 x 2 = 0.5; record	0
.5 x 2 = 1.0; record	1
.0 x 2 = 0.0; record	0
Thus: $.25_{10} = .010_2$	.010

EXAMPLE 3 Decimal to Octal (Integer)

273 ÷ 8 = 34 remainder 1; record	1
34 ÷ 8 = 4 remainder 2; record	2
4 ÷ 8 = 0 remainder 4; record	4
Thus: $273_{10} = 421_8$	421

EXAMPLE 4 Decimal to Octal (Fractional)

.55 x 8 = 4.4; record	4
.4 x 8 = 3.2; record	3
.2 x 8 = 1.6; record	1
-- --	-
-- --	-
	.431 . . .

Thus: $.55_{10} = .431\ldots_8$

Substitution

This method permits easy conversion between octal and binary representations of a number. If a number in binary notation is partitioned into triplets to the right and left of the binary point, each triplet may be converted into an octal digit. Similarly, each octal digit may be converted into a triplet of binary digits.

EXAMPLE 1 Binary to Octal

Binary =	110	000	.	001	010
Octal =	6	0	.	1 .	2

EXAMPLE 2 Octal to Binary

Octal =	6	5	0	.	2	2	7
Binary =	110	101	000	.	010	010	111

FIXED POINT ARITHMETIC

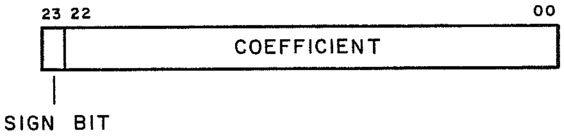
24-Bit Precision

Any number may be expressed in the form kB^n, where k is a coefficient, B a base number, and the exponent n the power to which the base number is raised.

A fixed point number assumes:

- The exponent $n = 0$ for all fixed point numbers.
- The coefficient, k, occupies the same bit positions within the computer word for all fixed point numbers.
- The radix (binary) point remains fixed with respect to one end of the expression.

A fixed point number consists of a sign bit and coefficient as shown below. The upper bit of any fixed point number designates the sign of the coefficient (23 lower order bits). If the bit is "1", the quantity is negative since negative numbers are represented in one's complement notation; a "0" sign bit signifies a positive coefficient.



The radix (binary) point is assumed to be immediately to the right of the lowest order bit (00).

In many instances, the values in a fixed point operation may be too large or too small to be expressed by the computer. The programmer must position the numbers within the word format so they can be represented with sufficient precision. The process, called scaling, consists of shifting the values a predetermined number of places. The numbers must be positioned far enough to the right in the register to prevent overflow but far enough to the left to maintain precision. The scale factor (number of places shifted) is expressed as the power of the base. For example, $5,100,000_{10}$ may be expressed as 0.51×10^7, 0.051×10^8, 0.0051×10^9, etc. The scale factors are 7, 8, and 9.

Since only the coefficient is used by the computer, the programmer is responsible for remembering the scale factors. Also, the possibility of an overflow during intermediate operations must be considered. For example, if two fractions in fixed point format are multiplied, the result is a number < 1. If the same two fractions are added, subtracted, or divided, the result may be greater than one and an overflow will occur. Similarly, if two integers are multiplied, divided, subtracted or added, the likelihood of an overflow is apparent.

48-Bit Precision (Double Precision)

The 48-bit Add, Subtract, Multiply and Divide instructions enable operands to be processed. The Multiply and Divide instructions utilize the E register and therefore are executed as trapped instructions if the applicable arithmetic option is not present in a system. Figure 5-5 in the Instruction Section illustrates the operand formats in 48-bit precision Multiply and Divide instructions.

FLOATING POINT ARITHMETIC

As an alternative to fixed point operation a method involving a variable radix point, called floating point, is used. This significantly reduces the amount of bookkeeping required on the part of the programmer.

By shifting the radix point and increasing or decreasing the value of the exponent, widely varying quantities which do not exceed the capacity of the machine may be handled.

Floating point numbers within the computer are represented in a form similar to that used in scientific notation, that is, a coefficient orfraction multiplied by a number raised to a power. Since the computer uses only binary numbers, the numbers are multiplied by powers of two.

$F \bullet 2^E$ where: F = fraction
E = exponent

In floating point, different coefficients need not relate to the same power of the base as they do in fixed point format. Therefore, the construction of a floating point number includes not only the coefficient but also the exponent.

NOTE

Refer to Figure 5-6 in the Instruction Section for the operand format and bit functions for specific floating point instructions.

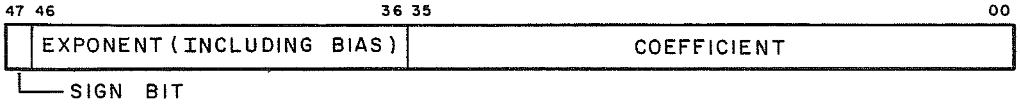
Coefficient

The coefficient consists of a 36-bit fraction in the 36 lower order positions of the floating point word. The coefficient is a normalized fraction; it is equal to or greater than 1/2 but less than 1. The highest order bit position (47) is occupied by the sign bit of the coefficient. If the sign bit is a "0", the coefficient is positive; a "1" bit denotes a negative fraction (negative fractions are represented in one's complement notation).

Exponent

The floating point exponent is expressed as an 11-bit quantity with a value ranging from 0000 to 3777_8. It is formed by adding a true positive exponent and a bias of 2000_8 or a true negative exponent and a bias of 1777_8. This results in a range of biased exponents as shown on the following page.

True Positive Exponent	Biased Exponent	True Negative Exponent	Biased Exponent
+0	2000	-0	2000*
+1	2001	-1	1776
+2	2002	-2	1775
--	----	--	----
--	----	--	----
+1776	3776	-1776	0001
+1777$_8$	3777$_8$	-1777$_8$	0000$_8$



*Minus zero is sensed as positive zero by the computer and is therefore biased by 2000$_8$ rather than 1777$_8$.

The exponent is biased so that floating point operands can be compared with each other in the normal fixed point mode.

As an example, compare the unbiased exponents of +52$_8$ and +0.02$_8$ (Example 1).

EXAMPLE 1

Number = +52$_8$

0	0 0 000 000 110	(36 bits)
Coefficient Sign	Exponent	Coefficient

Number = +0.02$_8$

0	1 1 111 111 011	(36 bits)
Coefficient Sign	Exponent	Coefficient

In this case +0.02 appears to be larger than +52 because of the larger exponent. If, however, both exponents are biased (Example 2), changing the sign of both exponents makes +52 greater than +0.02.

EXAMPLE 2

Number = +52$_8$

0	1 0 000 000 110	(36 bits)
Coefficient Sign	Exponent	Coefficient

Number = +0.02$_8$

0	0 1 111 111 011	(36 bits)
Coefficient Sign	Exponent	Coefficient

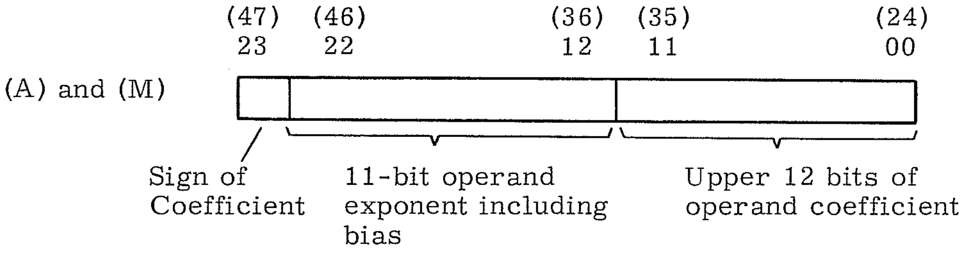
When bias is used with the exponent, floating point operation is more versatile since floating point operands can be compared with each other in the normal fixed point mode.

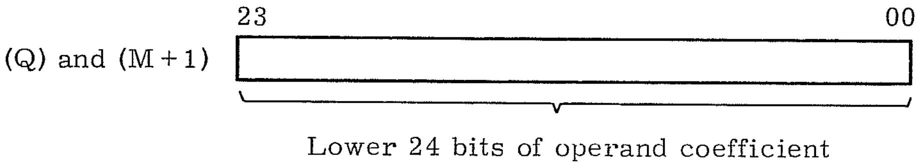
All floating point operations involve the A, Q, and E registers, plus two consecutive storage locations M and M + 1. The A and Q registers are treated as one 48-bit register. Indirect addressing and address modification are applicable to this whole group of instructions.

Operand Formats

The AQ register and the storage address contents have identical formats.

In both cases the maximum possible shift is 64 (77_8) bit positions. Since the coefficient consists of only 36 bits at the start, any shift greater than 36 positions will, of course, always result in an answer equal to the larger of the two original operands.


(47) 23
(46) 22
(36) 12
(35) 11
(24) 00
(A) and (M)
Sign of Coefficient
11-bit operand exponent including bias
Upper 12 bits of operand coefficient



Exponents

The 3100, 3200, 3300 Computers use an 11-bit exponent that is biased by 2000_8 for floating point operations. The effective modulus of the exponent is $\pm$ 1777_8 or $\pm$ 1023_{10}.

Exponent Equalization

During floating point addition and subtraction, the exponents involved are equalized prior to the operation.

- Addition - The coefficient of the algebraically smaller exponent is automatically shifted right in AQE until the exponents are equal. A maximum of 77_8 shifts may occur.

- Subtraction - If AQ contains the algebraically smaller exponent, the coefficient in AQ is shifted right in AQE until the exponents are equal. If (M) and (M + 1) have the smaller exponent, the complement of the coefficient of (M) and (M + 1) is shifted right in AQE until the exponents are equal or until a maximum of 77_8 shifts are performed.

Rounding

Rounding is an automatic floating point operation and is particularly necessary when floating point arithmetic operations yield coefficient answers in excess of 36 bits.

Although standard floating point format requires only a 36-bit coefficient, portions of the E register are used for extended coefficients. Refer to individual instruction descriptions for E register applications.

Rounding modifies the coefficient result of a floating point operation by adding or subtracting a "1" from the lowest bit position in Q without regard to the biased exponent. The coefficient of the answer in AQ passes through the adder with the rounding quantity before normalization. The conditions for rounding are classified according to arithmetic operation in Table B-2.

TABLE B-2. ROUNDED CONDITIONS FOLLOWING ARITHMETIC OPERATION

Arithmetic OPERATION	Bit 23 of the A Register	Bit 47 of the E Register or (Ratio of Residue/Divisor for Divide Only)	Applicable Rounding
ADD or SUBTRACT	0*	0	No
	0*	1	Add "1"
	1*	0	Subtract "1"
	1*	1	No
	Comments: Rounding occurs as a result of inequality between the sign bits of AQ and E.		
MULTIPLY	0	0	No
	0	1	Add "1"
	1	0	Subtract "1"
	1	1	No
	Comments: A floating point multiplication yields a 76 bit coefficient. Comparison between the sign bits of AQ and E indicates that the lower 36 bits are equal to or greater than 1/2 of the lowest order bit in AQ.		
DIVIDE	0	≥ 1/2 (absolute)	Add "1"
	0	≤ 1/2 (absolute)	No
	1	≥ 1/2 (absolute)	Subtract "1"
	1	≤ 1/2 (absolute)	No
	Comments: Rounding occurs if the answer resulting from the final residue division is equal to or greater than 1/2.		

*Condition of bit 23 of the A register immediately after equalization. (Refer to Exponent Equalization on preceeding page).

Normalizing

Normalizing brings the above answer back to a fraction with a value between one-half and one with the binary point to the left of the 36th bit of the coefficient. In other words, the final normalized coefficient in AQ will range in value from 2^{36} to $2^{37}-1$ including sign. Arithmetic control normalizes the answer by right or left shifting the coefficient the necessary number of places and adjusting the exponent. It does not shift the residue that is in E.

Faults

Three conditions are considered faults during the execution of floating point instructions:

- Exponent overflow ($> + 1777_8$)
- Exponent underflow ($< - 1777_8$)
- Division by zero, by too small a number, or by a number that is not in floating point format

These faults have several things in common:

- They can be sensed by the INS (77. 3) instruction
- Sensing automatically clears them
- The program should sense for these faults only after the floating point instructions have had sufficient time to go to completion
- They may be used to cause an interrupt

FIXED POINT/FLOATING POINT CONVERSIONS

Fixed Point To Floating Point

- Express the number in binary.
- Normalize the number. A normalized number has the most significant 1 positioned immediately to the right of the binary point and is expressed in the range $1/2 \leq k < 1$.
- Inspect the sign of the true exponent. If the sign is positive add 2000_8 (bias) to the true exponent of the normalized number. If the sign is negative, add the bias 1777_8 to the true exponent of the normalized number. In either case, the resulting exponent is the biased exponent.
- Assemble the number in floating point.
- Inspect the sign of the coefficient. If negative, complement the assembled floating point number to obtain the true floating point representation of the number. If the sign of the coefficient is positive, the assembled floating point number is the true representation.

EXAMPLE 1 Convert +4.0 to floating point

- The number is expressed in octal.
- Normalize 4.0 = $4.0 \times 8^0 = 0.100 \times 2^3$
- Since the sign of the true exponent is positive, add 2000_8 (bias) to the true exponent. Biased exponent = 2000 + 3.
- Assemble number in floating point format.
  Coefficient = $400\ 000\ 000\ 000_8$
  Biased Exponent = 2003_8
  Assembled word = $2003\ 400\ 000\ 000\ 000_8$
- Since the sign of the coefficient is positive, the floating point representation of +4.0 is as shown. If, however, the sign of the coefficient were negative, it would be necessary to complement the entire floating point word.

EXAMPLE 2 Convert -4.0 to floating point

- The number is expressed in octal.
- Normalize -4.0 = $-4.0 \times 8^0 = -0.100 \times 2^3$
- Since the sign of the true exponent is positive, add 2000_8 (bias) to the true exponent. Biased exponent = 2000 + 3.
- Assemble number in floating point format.
  Coefficient = $400\ 000\ 000\ 000_8$
  Biased Exponent = 2003_8
  Assembled word = $2003\ 400\ 000\ 000\ 000_8$
- Since the sign of the coefficient is negative, the assembled floating point word must be complemented. Therefore, the true floating point representation for -4.0 = $5774\ 377\ 777\ 777\ 777_8$.

EXAMPLE 3 Convert 0.5_{10} to floating point

- Convert to octal $0.5_{10} = 0.4_8$
- Normalize 0.4 = $0.4 \times 8^0 = 0.100 \times 2^0$
- Since the sign of the true exponent is positive, add 2000_8 (bias) to the true exponent. Biased exponent = 2000 + 0.
- Assemble number in floating point format.
  Coefficient = $400\ 000\ 000\ 000_8$
  Biased Exponent = 2000_8
  Assembled word = $2000\ 400\ 000\ 000\ 000_8$
- Since the sign of the coefficient is positive, the floating point representation of $+0.5_{10}$ is as shown. If, however, the sign of the coefficient were negative, it would be necessary to complement the entire floating point word. This example is a special case of floating point since the exponent of the normalized number is 0 and could be represented as -0. The exponent would then be biased by 1777_8 instead of 2000_8 because of the negative exponent. The 3100 and 3200, however, recognize -0 as +0 and bias the exponent by 2000_8.

EXAMPLE 4 Convert 0.04_8 to floating point

- The number is expressed in octal.
- Normalize $0.04 = 0.04 \times 8^0 = 0.4 \times 8^{-1} = 0.100 \times 2^{-3}$.
- Since the sign of the true exponent is negative, add 1777_8 (bias) to the true exponent. Biased exponent = 1777_8 + (-3) = 1774_8
- Assemble number in floating point format.
  Coefficient = 400 000 000 000_8
  Biased Exponent = 1774_8
  Assembled word = 1774 400 000 000 000_8
- Since the sign of the coefficient is positive, the floating point representation of 0.04_8 is as shown. If, however, the sign of the coefficient were negative, it would be necessary to complement the entire floating point word.

Floating Point to Fixed Point Format

- If the floating point number is negative, complement the entire floating point word and record the fact that the quantity is negative. The exponent is now in a true biased form.
- If the biased exponent is equal to or greater than 2000_8, subtract 2000_8 to obtain the true exponent; if less than 2000_8, subtract 1777_8 to obtain true exponent.
- Separate the coefficient and exponent. If the true exponent is negative, the binary point should be moved to the left the number of bit positions indicated by the true exponent. If the true exponent is positive, the binary point should be moved to the right the number of bit positions indicated by the true exponent.
- The coefficient has now been converted to fixed binary. The sign of the coefficient will be negative if the floating point number was complemented in step one. (The sign bit must be extended if the quantity is placed in a register.)
- Represent the fixed binary number in fixed octal notation.

EXAMPLE 1 Convert floating point number 2003 400 000 000 000_8 to fixed octal

- The floating point number is positive and remains uncomplemented.
- The biased exponent $> 2000_8$; therefore, subtract 2000_8 from the biased exponent to obtain the true exponent of the number. 2003 - 2000 = +3.
- Coefficient = 400 000 000 000_8 = $.100_2$. Move binary point to the right three places. Coefficient = 100.0_2.
- The sign of the coefficient is positive because the floating point number was not complemented in step one.
- Represent in fixed octal notation.
  $100.0 \times 2^0 = 4.0 \times 8^0$.

EXAMPLE 2 Convert floating point number $5774\ 377\ 777\ 777\ 777_8$ to fixed octal

- The sign of the coefficient is negative; therefore, complement the floating point number. Complement = $2003\ 400\ 000\ 000\ 000_8$
- The biased exponent (in complemented form) $> 2000_8$; therefore, subtract 2000_8 from the biased exponent to obtain the true exponent of the number 2003 - 2000 = +3.
- Coefficient = $4000\ 000\ 000\ 000_8 = 0.100_2$. Move binary point to the right three places. Coefficient = 100.0_2.
- The sign of the coefficient will be negative because the floating point number was originally complemented.
- Convert to fixed octal. $-100.0_2 = 4.0_8$.

EXAMPLE 3 Convert floating point number $1774\ 400\ 000\ 000\ 000_8$ to fixed octal

- The floating point number is positive and remains uncomplemented.
- The biased exponent $< 2000_8$; therefore, subtract 1777_8 from the biased exponent to obtain the true exponent of the number. $1774_8 - 1777_8 = -3$.
- Coefficient = $400\ 000\ 000\ 000_8 = .100_2$. Move binary point to the left three places. Coefficient = $.000100_2$.
- The sign of the coefficient is positive because the floating point number was not complemented in step one.
- Represent in fixed octal notation. $.000100_2 = .04_8$.

APPENDIX C

PROGRAMMING REFERENCE TABLES AND CONVERSION INFORMATION

TABLE OF POWERS OF TWO

2^n	n	2^{-n}
1	0	1.0
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.062 5
32	5	0.031 25
64	6	0.015 625
128	7	0.007 812 5
256	8	0.003 906 25
512	9	0.001 953 125
1 024	10	0.000 976 562 5
2 048	11	0.000 488 281 25
4 096	12	0.000 244 140 625
8 192	13	0.000 122 070 312 5
16 384	14	0.000 061 035 156 25
32 768	15	0.000 030 517 578 125
65 536	16	0.000 015 258 789 062 5
131 072	17	0.000 007 629 394 531 25
262 144	18	0.000 003 814 697 265 625
524 288	19	0.000 001 907 348 632 812 5
1 048 576	20	0.000 000 953 674 316 406 25
2 097 152	21	0.000 000 476 837 158 203 125
4 194 304	22	0.000 000 238 418 579 101 562 5
8 388 608	23	0.000 000 119 209 289 550 781 25
16 777 216	24	0.000 000 059 604 644 775 390 625
33 554 432	25	0.000 000 029 802 322 387 695 312 5
67 108 864	26	0.000 000 014 901 161 193 847 656 25
134 217 728	27	0.000 000 007 450 580 596 923 828 125
268 435 456	28	0.000 000 003 725 290 298 461 914 062 5
536 870 912	29	0.000 000 001 862 645 149 230 957 031 25
1 073 741 824	30	0.000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0.000 000 000 465 661 287 307 739 257 812 5
4 294 967 296	32	0.000 000 000 232 830 643 653 869 628 906 25
8 589 934 592	33	0.000 000 000 116 415 321 826 934 814 453 125
17 179 869 184	34	0.000 000 000 058 207 660 913 467 407 226 562 5
34 359 738 368	35	0.000 000 000 029 103 830 456 733 703 613 281 25
68 719 476 736	36	0.000 000 000 014 551 915 228 366 851 806 640 625
137 438 953 472	37	0.000 000 000 007 275 957 614 183 425 903 320 312 5
274 877 906 944	38	0.000 000 000 003 637 978 807 091 712 951 660 156 25
549 755 813 888	39	0.000 000 000 001 818 989 403 545 856 475 830 078 125
1 099 511 627 776	40	0.000 000 000 000 909 494 701 772 928 237 915 039 062 5
2 199 023 255 552	41	0.000 000 000 000 454 747 350 886 464 118 957 519 531 25
4 398 046 511 104	42	0.000 000 000 000 227 373 675 443 232 059 478 759 765 625
8 796 093 022 208	43	0.000 000 000 000 113 686 837 721 616 029 739 379 882 812 5
17 592 186 044 416	44	0.000 000 000 000 056 843 418 860 808 014 869 689 941 406 25
35 184 372 088 832	45	0.000 000 000 000 028 421 709 430 404 007 434 844 970 703 125
70 368 744 177 664	46	0.000 000 000 000 014 210 854 715 202 003 717 422 485 351 562 5
140 737 488 355 328	47	0.000 000 000 000 007 105 427 357 601 001 858 711 242 675 781 25
281 474 976 710 656	48	0.000 000 000 000 003 552 713 678 800 500 929 355 621 337 890 625
562 949 953 421 312	49	0.000 000 000 000 001 776 356 839 400 250 464 677 810 668 945 312 5
1 125 899 906 842 624	50	0.000 000 000 000 000 888 178 419 700 125 232 338 905 334 472 656 25
2 251 799 813 685 248	51	0.000 000 000 000 000 444 089 209 850 062 616 169 452 667 236 328 125
4 503 599 627 370 496	52	0.000 000 000 000 000 222 044 604 925 031 308 084 726 333 618 164 062 5
9 007 199 254 740 992	53	0.000 000 000 000 000 111 022 302 462 515 654 042 363 166 809 082 031 25
18 014 398 509 481 984	54	0.000 000 000 000 000 055 511 151 231 257 827 021 181 583 404 541 015 625
36 028 797 018 963 968	55	0.000 000 000 000 000 027 755 575 615 628 913 510 590 791 702 270 507 812 5
72 057 594 037 927 936	56	0.000 000 000 000 000 013 877 787 807 814 456 755 295 395 851 135 253 906 25
144 115 188 075 855 872	57	0.000 000 000 000 000 006 938 893 903 907 228 377 647 697 925 567 626 953 125
288 230 376 151 711 744	58	0.000 000 000 000 000 003 469 446 951 953 614 188 823 848 962 783 813 476 562 5
576 460 752 303 423 488	59	0.000 000 000 000 000 001 734 723 475 976 807 094 411 924 481 391 906 738 281 25
1 152 921 504 606 846 976	60	0 000 000 000 000 000 000 867 361 737 988 403 547 205 962 240 695 953 369 140 625

DECIMAL/BINARY POSITION TABLE

Largest Decimal Integer	Decimal Digits Req'd*	Number of Binary Digits	Largest Decimal Fraction
1		1	.5
3		2	.75
7		3	.875
15	1	4	.937 5
31		5	.968 75
63		6	.984 375
127	2	7	.992 187 5
255		8	.996 093 75
511		9	.998 046 875
1 023	3	10	.999 023 437 5
2 047		11	.999 511 718 75
4 095		12	.999 755 859 375
8 191		13	.999 877 929 687 5
16 383	4	14	.999 938 964 843 75
32 767		15	.999 969 482 421 875
65 535		16	.999 984 741 210 937 5
131 071	5	17	.999 992 370 605 468 75
262 143		18	.999 996 185 302 734 375
524 287		19	.999 998 092 651 367 187 5
1 048 575	6	20	.999 999 046 325 683 593 75
2 097 151		21	.999 999 523 162 841 796 875
4 194 303		22	.999 999 761 581 420 898 437 5
8 388 607		23	.999 999 880 790 710 449 218 75
16 777 215	7	24	.999 999 940 395 355 244 609 375
33 554 431		25	.999 999 970 197 677 612 304 687 5
67 108 863		26	.999 999 985 098 838 806 152 343 75
134 217 727	8	27	.999 999 992 549 419 403 076 171 875
268 435 455		28	.999 999 996 274 709 701 538 085 937 5
536 870 911		29	.999 999 998 137 354 850 769 042 968 75
1 073 741 823	9	30	.999 999 999 068 677 425 384 521 484 375
2 147 483 647		31	.999 999 999 534 338 712 692 260 742 187 5
4 294 967 295		32	.999 999 999 767 169 356 346 130 371 093 75
8 589 934 591		33	.999 999 999 883 584 678 173 065 185 546 875
17 179 869 183	10	34	.999 999 999 941 792 339 086 532 592 773 437 5
34 359 738 367		35	.999 999 999 970 896 169 543 266 296 386 718 75
68 719 476 735		36	.999 999 999 985 448 034 771 633 148 193 359 375
137 438 953 471	11	37	.999 999 999 992 724 042 385 816 574 096 679 687 5
274 877 906 943		38	.999 999 999 996 362 021 192 908 287 048 339 843 75
549 755 813 887		39	.999 999 999 998 181 010 596 454 143 524 169 921 875
1 099 511 627 775	12	40	.999 999 999 999 090 505 298 227 071 762 084 960 937 5
2 199 023 255 551		41	.999 999 999 999 545 252 649 113 535 881 042 480 468 75
4 398 046 511 103		42	.999 999 999 999 772 626 324 556 767 940 521 240 234 375
8 796 093 022 207		43	.999 999 999 999 886 313 162 278 383 970 260 620 117 187 5
17 592 186 044 415	13	44	.999 999 999 999 943 156 581 139 191 985 130 310 058 593 75
35 184 372 088 831		45	.999 999 999 999 971 578 290 569 595 992 565 155 029 296 875
70 368 744 177 663		46	.999 999 999 999 985 789 145 284 797 996 282 577 514 648 437 5
140 737 488 355 327	14	47	.999 999 999 999 992 894 572 642 398 998 141 288 757 324 218 75
281 474 976 710 655		48	.999 999 999 999 996 447 286 321 199 499 070 644 378 662 109 375
562 949 953 421 311		49	.999 999 999 999 998 223 643 160 599 749 535 322 189 331 054 687 5
1 125 899 906 842 623	15	50	.999 999 999 999 999 111 821 580 299 874 767 661 094 665 527 343 75
2 251 799 813 685 247		51	.999 999 999 999 999 555 910 790 149 937 383 830 547 332 763 671 875
4 503 599 627 370 495		52	.999 999 999 999 999 777 955 395 074 968 691 915 273 666 381 835 937 5
9 007 199 254 740 991		53	.999 999 999 999 999 888 977 697 537 484 345 957 636 833 190 917 968 75
18 014 398 509 481 983	16	54	.999 999 999 999 999 944 488 848 768 742 172 978 818 416 595 458 984 375
36 028 797 018 963 967		55	.999 999 999 999 999 972 244 424 384 371 086 489 409 208 297 729 492 187 5
72 057 594 037 927 935		56	.999 999 999 999 999 986 122 212 192 185 543 244 704 604 148 864 746 093 75
144 115 188 075 855 871	17	57	.999 999 999 999 999 993 061 106 096 092 771 622 352 302 074 432 373 046 875
288 230 376 151 711 743		58	.999 999 999 999 999 996 530 553 048 046 385 811 176 151 037 216 186 523 437 5
576 460 752 303 423 487		59	.999 999 999 999 999 998 265 276 524 023 192 905 588 075 518 608 093 261 718 75
1 152 921 504 606 846 975	18	60	.999 999 999 999 999 999 132 638 262 011 596 452 794 037 759 304 046 630 859 375

*Larger numbers within a digit group should be checked for exact number of decimal digits required.

Examples of use:

1. *Q*. What is the largest decimal value that can be expressed by 36 binary digits?
   A. 68,719,476,735.
2. *Q*. How many decimal digits will be required to express a 22-bit number?
   A. 7 decimal digits.

OCTAL ARITHMETIC MATRICES

ADDITION-SUBTRACTION

0	1	2	3	4	5	6	7
1	2	3	4	5	6	7	10
2	3	4	5	6	7	10	11
3	4	5	6	7	10	11	12
4	5	6	7	10	11	12	13
5	6	7	10	11	12	13	14
6	7	10	11	12	13	14	15
7	10	11	12	13	14	15	16

MULTIPLICATION-DIVISION

0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7
2	2	4	6	10	12	14	16
3	3	6	11	14	17	22	25
4	4	10	14	20	24	30	34
5	5	12	17	24	31	36	43
6	6	14	22	30	36	44	52
7	7	16	25	34	43	52	61

CONSTANTS

π	=	3.14159 26535 89793 23846 26433 83279 50
$\sqrt{3}$	=	1.732 050 807 569
$\sqrt{10}$	=	3.162 277 660 1683
e	=	2.71828 18284 59045 23536
ln 2	=	0.69314 71805 599453
ln 10	=	2.30258 50929 94045 68402
$\log_{10} 2$	=	0.30102 99956 63981
$\log_{10} e$	=	0.43429 44819 03251 82765
$\log_{10} \log_{10} e$	=	9.63778 43113 00537 -10
$\log_{10} \pi$	=	0.49714 98726 94133 85435
1 degree	=	0.01745 32925 11943 radians
1 radian	=	57.29577 95131 degrees
$\log_{10}(5)$	=	0.69897 00043 36019
7!	=	5040
8!	=	40320
9!	=	362,880
10!	=	3,628,800
11!	=	39,916,800
12!	=	479,001,600
13!	=	6,227,020,800
14!	=	87,178,291,200
15!	=	1,307,674,368,000
16!	=	20,922,789,888,000
$\frac{\pi}{180}$	=	0.01745 32925 19943 29576 92369 07684 9
$\left(\frac{\pi}{2}\right)^2$	=	2.4674 01100 27233 96
$\left(\frac{\pi}{2}\right)^3$	=	3.8757 84585 03747 74
$\left(\frac{\pi}{2}\right)^4$	=	6.0880 68189 62515 20
$\left(\frac{\pi}{2}\right)^5$	=	9.5631 15149 54004 49
$\left(\frac{\pi}{2}\right)^6$	=	15.0217 06149 61413 07
$\left(\frac{\pi}{2}\right)^7$	=	23.5960 40842 00618 62
$\left(\frac{\pi}{2}\right)^8$	=	37.0645 72481 52567 57
$\left(\frac{\pi}{2}\right)^9$	=	58.2208 97135 63712 59
$\left(\frac{\pi}{2}\right)^{10}$	=	91.4531 71363 36231 53
$\left(\frac{\pi}{2}\right)^{11}$	=	143.6543 05651 31374 95
$\left(\frac{\pi}{2}\right)^{12}$	=	225.6516 55645 350
$\left(\frac{\pi}{2}\right)^{13}$	=	354.4527 91822 91051 47
$\left(\frac{\pi}{2}\right)^{14}$	=	556.7731 43417 624

CONSTANTS (Continued)

π^2	=	9.86960 44010 89358 61883 43909 9988
$2\pi^2$	=	19.73920 88021 78717 23766 87819 9976
$3\pi^2$	=	29.60881 32032 68075 85680 31729 9964
$4\pi^2$	=	39.47841 76043 57434 47533 75639 9952
$5\pi^2$	=	49.34802 20054 46793 09417 19549 9940
$6\pi^2$	=	59.21762 64065 36151 71300 63459 9928
$7\pi^2$	=	69.08723 08076 25510 33184 07369 9916
$8\pi^2$	=	78.95683 52087 14868 95067 51279 9904
$9\pi^2$	=	88.82643 96098 04227 56950 95189 9892

$\sqrt{2}$	=	1.414 213 562 373 095 048 801 688
$1 + \sqrt{2}$	=	2.414 213 562 373 095 048 801 688
$(1 + \sqrt{2})^2$	=	5.828 427 124 746 18
$(1 + \sqrt{2})^4$	=	33.970 562 748 477 08
$(1 + \sqrt{2})^6$	=	197.994 949 366 116 30
$(1 + \sqrt{2})^8$	=	1153.999 133 448 220 72
$(1 + \sqrt{2})^{10}$	=	6725.999 851 323 208 02
$(1 + \sqrt{2})^{12}$	=	39201.999 974 491 027 40
$(1 + \sqrt{2})^{14}$	=	228485.999 995 622 956 38
$(1 + \sqrt{2})^{16}$	=	1331713.999 999 246 711
$(1 + \sqrt{2})^{18}$	=	7761797.999 999 884 751

Sin .5	=	0.47942 55386 04203
Cos .5	=	0.87758 25618 90373
Tan .5	=	0.54630 24898 43790
Sin 1	=	0.84147 09848 07896
Cos 1	=	0.54030 23058 68140
Tan 1	=	1.55740 77246 5490
Sin 1.5	=	0.99749 49866 04054
Cos 1.5	=	0.07073 72016 67708
Tan 1.5	=	14.10141 99471 707

OCTAL-DECIMAL INTEGER CONVERSION TABLE

	0	1	2	3	4	5	6	7
0000	0000	0001	0002	0003	0004	0005	0006	0007
0010	0008	0009	0010	0011	0012	0013	0014	0015
0020	0016	0017	0018	0019	0020	0021	0022	0023
0030	0024	0025	0026	0027	0028	0029	0030	0031
0040	0032	0033	0034	0035	0036	0037	0038	0039
0050	0040	0041	0042	0043	0044	0045	0046	0047
0060	0048	0049	0050	0051	0052	0053	0054	0055
0070	0056	0057	0058	0059	0060	0061	0062	0063
0100	0064	0065	0066	0067	0068	0069	0070	0071
0110	0072	0073	0074	0075	0076	0077	0078	0079
0120	0080	0081	0082	0083	0084	0085	0086	0087
0130	0088	0089	0090	0091	0092	0093	0094	0095
0140	0096	0097	0098	0099	0100	0101	0102	0103
0150	0104	0105	0106	0107	0108	0109	0110	0111
0160	0112	0113	0114	0115	0116	0117	0118	0119
0170	0120	0121	0122	0123	0124	0125	0126	0127
0200	0128	0129	0130	0131	0132	0133	0134	0135
0210	0136	0137	0138	0139	0140	0141	0142	0143
0220	0144	0145	0146	0147	0148	0149	0150	0151
0230	0152	0153	0154	0155	0156	0157	0158	0159
0240	0160	0161	0162	0163	0164	0165	0166	0167
0250	0168	0169	0170	0171	0172	0173	0174	0175
0260	0176	0177	0178	0179	0180	0181	0182	0183
0270	0184	0185	0186	0187	0188	0189	0190	0191
0300	0192	0193	0194	0195	0196	0197	0198	0199
0310	0200	0201	0202	0203	0204	0205	0206	0207
0320	0208	0209	0210	0211	0212	0213	0214	0215
0330	0216	0217	0218	0219	0220	0221	0222	0223
0340	0224	0225	0226	0227	0228	0229	0230	0231
0350	0232	0233	0234	0235	0236	0237	0238	0239
0360	0240	0241	0242	0243	0244	0245	0246	0247
0370	0248	0249	0250	0251	0252	0253	0254	0255

	0	1	2	3	4	5	6	7
0400	0256	0257	0258	0259	0260	0261	0262	0263
0410	0264	0265	0266	0267	0268	0269	0270	0271
0420	0272	0273	0274	0275	0276	0277	0278	0279
0430	0280	0281	0282	0283	0284	0285	0286	0287
0440	0288	0289	0290	0291	0292	0293	0294	0295
0450	0296	0297	0298	0299	0300	0301	0302	0303
0460	0304	0305	0306	0307	0308	0309	0310	0311
0470	0312	0313	0314	0315	0316	0317	0318	0319
0500	0320	0321	0322	0323	0324	0325	0326	0327
0510	0328	0329	0330	0331	0332	0333	0334	0335
0520	0336	0337	0338	0339	0340	0341	0342	0343
0530	0344	0345	0346	0347	0348	0349	0350	0351
0540	0352	0353	0354	0355	0356	0357	0358	0359
0550	0360	0361	0362	0363	0364	0365	0366	0367
0560	0368	0369	0370	0371	0372	0373	0374	0375
0570	0376	0377	0378	0379	0380	0381	0382	0383
0600	0384	0385	0386	0387	0388	0389	0390	0391
0610	0392	0393	0394	0395	0396	0397	0398	0399
0620	0400	0401	0402	0403	0404	0405	0406	0407
0630	0408	0409	0410	0411	0412	0413	0414	0415
0640	0416	0417	0418	0419	0420	0421	0422	0423
0650	0424	0425	0426	0427	0428	0429	0430	0431
0660	0432	0433	0434	0435	0436	0437	0438	0439
0670	0440	0441	0442	0443	0444	0445	0446	0447
0700	0448	0449	0450	0451	0452	0453	0454	0455
0710	0456	0457	0458	0459	0460	0461	0462	0463
0720	0464	0465	0466	0467	0468	0469	0470	0471
0730	0472	0473	0474	0475	0476	0477	0478	0479
0740	0480	0481	0482	0483	0484	0485	0486	0487
0750	0488	0489	0490	0491	0492	0493	0494	0495
0760	0496	0497	0498	0499	0500	0501	0502	0503
0770	0504	0505	0506	0507	0508	0509	0510	0511

0000	0000
to	to
0777	0511
(Octal)	(Decimal)

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
1000	0512	0513	0514	0515	0516	0517	0518	0519
1010	0520	0521	0522	0523	0524	0525	0526	0527
1020	0528	0529	0530	0531	0532	0533	0534	0535
1030	0536	0537	0538	0539	0540	0541	0542	0543
1040	0544	0545	0546	0547	0548	0549	0550	0551
1050	0552	0553	0554	0555	0556	0557	0558	0559
1060	0560	0561	0562	0563	0564	0565	0566	0567
1070	0568	0569	0570	0571	0572	0573	0574	0575
1100	0576	0577	0578	0579	0580	0581	0582	0583
1110	0584	0585	0586	0587	0588	0589	0590	0591
1120	0592	0593	0594	0595	0596	0597	0598	0599
1130	0600	0601	0602	0603	0604	0605	0606	0607
1140	0608	0609	0610	0611	0612	0613	0614	0615
1150	0616	0617	0618	0619	0620	0621	0622	0623
1160	0624	0625	0626	0627	0628	0629	0630	0631
1170	0632	0633	0634	0635	0636	0637	0638	0639
1200	0640	0641	0642	0643	0644	0645	0646	0647
1210	0648	0649	0650	0651	0652	0653	0654	0655
1220	0656	0657	0658	0659	0660	0661	0662	0663
1230	0664	0665	0666	0667	0668	0669	0670	0671
1240	0672	0673	0674	0675	0676	0677	0678	0679
1250	0680	0681	0682	0683	0684	0685	0686	0687
1260	0688	0689	0690	0691	0692	0693	0694	0695
1270	0696	0697	0698	0699	0700	0701	0702	0703
1300	0704	0705	0706	0707	0708	0709	0710	0711
1310	0712	0713	0714	0715	0716	0717	0718	0719
1320	0720	0721	0722	0723	0724	0725	0726	0727
1330	0728	0729	0730	0731	0732	0733	0734	0735
1340	0736	0737	0738	0739	0740	0741	0742	0743
1350	0744	0745	0746	0747	0748	0749	0750	0751
1360	0752	0753	0754	0755	0756	0757	0758	0759
1370	0760	0761	0762	0763	0764	0765	0766	0767

	0	1	2	3	4	5	6	7
1400	0768	0769	0770	0771	0772	0773	0774	0775
1410	0776	0777	0778	0779	0780	0781	0782	0783
1420	0784	0785	0786	0787	0788	0789	0790	0791
1430	0792	0793	0794	0795	0796	0797	0798	0799
1440	0800	0801	0802	0803	0804	0805	0806	0807
1450	0808	0809	0810	0811	0812	0813	0814	0815
1460	0816	0817	0818	0819	0820	0821	0822	0823
1470	0824	0825	0826	0827	0828	0829	0830	0831
1500	0832	0833	0834	0835	0836	0837	0838	0839
1510	0840	0841	0842	0843	0844	0845	0846	0847
1520	0848	0849	0850	0851	0852	0853	0854	0855
1530	0856	0857	0858	0859	0860	0861	0862	0863
1540	0864	0865	0866	0867	0868	0869	0870	0871
1550	0872	0873	0874	0875	0876	0877	0878	0879
1560	0880	0881	0882	0883	0884	0885	0886	0887
1570	0888	0889	0890	0891	0892	0893	0894	0895
1600	0896	0897	0898	0899	0900	0901	0902	0903
1610	0904	0905	0906	0907	0908	0909	0910	0911
1620	0912	0913	0914	0915	0916	0917	0918	0919
1630	0920	0921	0922	0923	0924	0925	0926	0927
1640	0928	0929	0930	0931	0932	0933	0934	0935
1650	0936	0937	0938	0939	0940	0941	0942	0943
1660	0944	0945	0946	0947	0948	0949	0950	0951
1670	0952	0953	0954	0955	0956	0957	0958	0959
1700	0960	0961	0962	0963	0964	0965	0966	0967
1710	0968	0969	0970	0971	0972	0973	0974	0975
1720	0976	0977	0978	0979	0980	0981	0982	0983
1730	0984	0985	0986	0987	0988	0989	0990	0991
1740	0992	0993	0994	0995	0996	0997	0998	0999
1750	1000	1001	1002	1003	1004	1005	1006	1007
1760	1008	1009	1010	1011	1012	1013	1014	1015
1770	1016	1017	1018	1019	1020	1021	1022	1023

1000	0512
to	to
1777	1023
(Octal)	(Decimal)

OCTAL-DECIMAL INTEGER CONVERSION TABLE (Cont'd)

2000	1024
to	to
2777	1535
(Octal)	(Decimal)

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
2000	1024	1025	1026	1027	1028	1029	1030	1031
2010	1032	1033	1034	1035	1036	1037	1038	1039
2020	1040	1041	1042	1043	1044	1045	1046	1047
2030	1048	1049	1050	1051	1052	1053	1054	1055
2040	1056	1057	1058	1059	1060	1061	1062	1063
2050	1064	1065	1066	1067	1068	1069	1070	1071
2060	1072	1073	1074	1075	1076	1077	1078	1079
2070	1080	1081	1082	1083	1084	1085	1086	1087
2100	1088	1089	1090	1091	1092	1093	1094	1095
2110	1096	1097	1098	1099	1100	1101	1102	1103
2120	1104	1105	1106	1107	1108	1109	1110	1111
2130	1112	1113	1114	1115	1116	1117	1118	1119
2140	1120	1121	1122	1123	1124	1125	1126	1127
2150	1128	1129	1130	1131	1132	1133	1134	1135
2160	1136	1137	1138	1139	1140	1141	1142	1143
2170	1144	1145	1146	1147	1148	1149	1150	1151
2200	1152	1153	1154	1155	1156	1157	1158	1159
2210	1160	1161	1162	1163	1164	1165	1166	1167
2220	1168	1169	1170	1171	1172	1173	1174	1175
2230	1176	1177	1178	1179	1180	1181	1182	1183
2240	1184	1185	1186	1187	1188	1189	1190	1191
2250	1192	1193	1194	1195	1196	1197	1198	1199
2260	1200	1201	1202	1203	1204	1205	1206	1207
2270	1208	1209	1210	1211	1212	1213	1214	1215
2300	1216	1217	1218	1219	1220	1221	1222	1223
2310	1224	1225	1226	1227	1228	1229	1230	1231
2320	1232	1233	1234	1235	1236	1237	1238	1239
2330	1240	1241	1242	1243	1244	1245	1246	1247
2340	1248	1249	1250	1251	1252	1253	1254	1255
2350	1256	1257	1258	1259	1260	1261	1262	1263
2360	1264	1265	1266	1267	1268	1269	1270	1271
2370	1272	1273	1274	1275	1276	1277	1278	1279

	0	1	2	3	4	5	6	7
2400	1280	1281	1282	1283	1284	1285	1286	1287
2410	1288	1289	1290	1291	1292	1293	1294	1295
2420	1296	1297	1298	1299	1300	1301	1302	1303
2430	1304	1305	1306	1307	1308	1309	1310	1311
2440	1312	1313	1314	1315	1316	1317	1318	1319
2450	1320	1321	1322	1323	1324	1325	1326	1327
2460	1328	1329	1330	1331	1332	1333	1334	1335
2470	1336	1337	1338	1339	1340	1341	1342	1343
2500	1344	1345	1346	1347	1348	1349	1350	1351
2510	1352	1353	1354	1355	1356	1357	1358	1359
2520	1360	1361	1362	1363	1364	1365	1366	1367
2530	1368	1369	1370	1371	1372	1373	1374	1375
2540	1376	1377	1378	1379	1380	1381	1382	1383
2550	1384	1385	1386	1387	1388	1389	1390	1391
2560	1392	1393	1394	1395	1396	1397	1398	1399
2570	1400	1401	1402	1403	1404	1405	1406	1407
2600	1408	1409	1410	1411	1412	1413	1414	1415
2610	1416	1417	1418	1419	1420	1421	1422	1423
2620	1424	1425	1426	1427	1428	1429	1430	1431
2630	1432	1433	1434	1435	1436	1437	1438	1439
2640	1440	1441	1442	1443	1444	1445	1446	1447
2650	1448	1449	1450	1451	1452	1453	1454	1455
2660	1456	1457	1458	1459	1460	1461	1462	1463
2670	1464	1465	1466	1467	1468	1469	1470	1471
2700	1472	1473	1474	1475	1476	1477	1478	1479
2710	1480	1481	1482	1483	1484	1485	1486	1487
2720	1488	1489	1490	1491	1492	1493	1494	1495
2730	1496	1497	1498	1499	1500	1501	1502	1503
2740	1504	1505	1506	1507	1508	1519	1510	1511
2750	1512	1513	1514	1515	1516	1517	1518	1519
2760	1520	1521	1522	1523	1524	1525	1526	1527
2770	1528	1529	1530	1531	1532	1533	1534	1535

3000	1536
to	to
3777	2047
(Octal)	(Decimal)

	0	1	2	3	4	5	6	7
3000	1536	1537	1538	1539	1540	1541	1542	1543
3010	1544	1545	1546	1547	1548	1549	1550	1551
3020	1552	1553	1554	1555	1556	1557	1558	1559
3030	1560	1561	1562	1563	1564	1565	1566	1567
3040	1568	1569	1570	1571	1572	1573	1574	1575
3050	1576	1577	1578	1579	1580	1581	1582	1583
3060	1584	1585	1586	1587	1588	1589	1590	1591
3070	1592	1593	1594	1595	1596	1597	1598	1599
3100	1600	1601	1602	1603	1604	1605	1606	1607
3110	1608	1609	1610	1611	1612	1613	1614	1615
3120	1616	1617	1618	1619	1620	1621	1622	1623
3130	1624	1625	1626	1627	1628	1629	1630	1631
3140	1632	1633	1634	1635	1636	1637	1638	1639
3150	1640	1641	1642	1643	1644	1645	1646	1647
3160	1648	1649	1650	1651	1652	1653	1654	1655
3170	1656	1657	1658	1659	1660	1661	1662	1663
3200	1664	1665	1666	1667	1668	1669	1670	1671
3210	1672	1673	1674	1675	1676	1677	1678	1679
3220	1680	1681	1682	1683	1684	1685	1686	1687
3230	1688	1689	1690	1691	1692	1693	1694	1695
3240	1696	1697	1698	1699	1700	1701	1702	1703
3250	1704	1705	1706	1707	1708	1709	1710	1711
3260	1712	1713	1714	1715	1716	1717	1718	1719
3270	1720	1721	1722	1723	1724	1725	1726	1727
3300	1728	1729	1730	1731	1732	1733	1734	1735
3310	1736	1737	1738	1739	1740	1741	1742	1743
3320	1744	1745	1746	1747	1748	1749	1750	1751
3330	1752	1753	1754	1755	1756	1757	1758	1759
3340	1760	1761	1762	1763	1764	1765	1766	1767
3350	1768	1769	1770	1771	1772	1773	1774	1775
3360	1776	1777	1778	1779	1780	1781	1782	1783
3370	1784	1785	1786	1787	1788	1789	1790	1791

	0	1	2	3	4	5	6	7
3400	1792	1793	1794	1795	1796	1797	1798	1799
3410	1800	1801	1802	1803	1804	1805	1806	1807
3420	1808	1809	1810	1811	1812	1813	1814	1815
3430	1816	1817	1818	1819	1820	1821	1822	1823
3440	1824	1825	1826	1827	1828	1829	1830	1831
3450	1832	1833	1834	1835	1836	1837	1838	1839
3460	1840	1841	1842	1843	1844	1845	1846	1847
3470	1848	1849	1850	1851	1852	1853	1854	1855
3500	1856	1857	1858	1859	1860	1861	1862	1863
3510	1864	1865	1866	1867	1868	1869	1870	1871
3520	1872	1873	1874	1875	1876	1877	1878	1879
3530	1880	1881	1882	1883	1884	1885	1886	1887
3540	1888	1889	1890	1891	1892	1893	1894	1895
3550	1896	1897	1898	1899	1900	1901	1902	1903
3560	1904	1905	1906	1907	1908	1909	1910	1911
3570	1912	1913	1914	1915	1916	1917	1918	1919
3600	1920	1921	1922	1923	1924	1925	1926	1927
3610	1928	1929	1930	1931	1932	1933	1934	1935
3620	1936	1937	1938	1939	1940	1941	1942	1943
3630	1944	1945	1946	1947	1948	1949	1950	1951
3640	1952	1953	1954	1955	1956	1957	1958	1959
3650	1960	1961	1962	1963	1964	1965	1966	1967
3660	1968	1969	1970	1971	1972	1973	1974	1975
3670	1976	1977	1978	1979	1980	1981	1982	1983
3700	1984	1985	1986	1987	1988	1989	1990	1991
3710	1992	1993	1994	1995	1996	1997	1998	1999
3720	2000	2001	2002	2003	2004	2005	2006	2007
3730	2008	2009	2010	2011	2012	2013	2014	2015
3740	2016	2017	2018	2019	2020	2021	2022	2023
3750	2024	2025	2026	2027	2028	2029	2030	2031
3760	2032	2033	2034	2035	2036	2037	2038	2039
3770	2040	2041	2042	2043	2044	2045	2046	2047

OCTAL-DECIMAL INTEGER CONVERSION TABLE (Cont'd)

	0	1	2	3	4	5	6	7
4000	2048	2049	2050	2051	2052	2053	2054	2055
4010	2056	2057	2058	2059	2060	2061	2062	2063
4020	2064	2065	2066	2067	2068	2069	2070	2071
4030	2072	2073	2074	2075	2076	2077	2078	2079
4040	2080	2081	2082	2083	2084	2085	2086	2087
4050	2088	2089	2090	2091	2092	2093	2094	2095
4060	2096	2097	2098	2099	2100	2101	2102	2103
4070	2104	2105	2106	2107	2108	2109	2110	2111
4100	2112	2113	2114	2115	2116	2117	2118	2119
4110	2120	2121	2122	2123	2124	2125	2126	2127
4120	2128	2129	2130	2131	2132	2133	2134	2135
4130	2136	2137	2138	2139	2140	2141	2142	2143
4140	2144	2145	2146	2147	2148	2149	2150	2151
4150	2152	2153	2154	2155	2156	2157	2158	2159
4160	2160	2161	2162	2163	2164	2165	2166	2167
4170	2168	2169	2170	2171	2172	2173	2174	2175
4200	2176	2177	2178	2179	2180	2181	2182	2183
4210	2184	2185	2186	2187	2188	2189	2190	2191
4220	2192	2193	2194	2195	2196	2197	2198	2199
4230	2200	2201	2202	2203	2204	2205	2206	2207
4240	2208	2209	2210	2211	2212	2213	2214	2215
4250	2216	2217	2218	2219	2220	2221	2222	2223
4260	2224	2225	2226	2227	2228	2229	2230	2231
4270	2232	2233	2234	2235	2236	2237	2238	2239
4300	2240	2241	2242	2243	2244	2245	2246	2247
4310	2248	2249	2250	2251	2252	2253	2254	2255
4320	2256	2257	2258	2259	2260	2261	2262	2263
4330	2264	2265	2266	2267	2268	2269	2270	2271
4340	2272	2273	2274	2275	2276	2277	2278	2279
4350	2280	2281	2282	2283	2284	2285	2286	2287
4360	2288	2289	2290	2291	2292	2293	2294	2295
4370	2296	2297	2298	2299	2300	2301	2302	2303

	0	1	2	3	4	5	6	7
4400	2304	2305	2306	2307	2308	2309	2310	2311
4410	2312	2313	2314	2315	2316	2317	2318	2319
4420	2320	2321	2322	2323	2324	2325	2326	2327
4430	2328	2329	2330	2331	2332	2333	2334	2335
4440	2336	2337	2338	2339	2340	2341	2342	2343
4450	2344	2345	2346	2347	2348	2349	2350	2351
4460	2352	2353	2354	2355	2356	2357	2358	2359
4470	2360	2361	2362	2363	2364	2365	2366	2367
4500	2368	2369	2370	2371	2372	2373	2374	2375
4510	2376	2377	2378	2379	2380	2381	2382	2383
4520	2384	2385	2386	2387	2388	2389	2390	2391
4530	2392	2393	2394	2395	2396	2397	2398	2399
4540	2400	2401	2402	2403	2404	2405	2406	2407
4550	2408	2409	2410	2411	2412	2413	2414	2415
4560	2416	2417	2418	2419	2420	2421	2422	2423
4570	2424	2425	2426	2427	2428	2429	2430	2431
4600	2432	2433	2434	2435	2436	2437	2438	2439
4610	2440	2441	2442	2443	2444	2445	2446	2447
4620	2448	2449	2450	2451	2452	2453	2454	2455
4630	2456	2457	2458	2459	2460	2461	2462	2463
4640	2464	2465	2466	2467	2468	2469	2470	2471
4650	2472	2473	2474	2475	2476	2477	2478	2479
4660	2480	2481	2482	2483	2484	2485	2486	2487
4670	2488	2489	2490	2491	2492	2493	2494	2495
4700	2496	2497	2498	2499	2500	2501	2502	2503
4710	2504	2505	2506	2507	2508	2509	2510	2511
4720	2512	2513	2514	2515	2516	2517	2518	2519
4730	2520	2521	2522	2523	2524	2525	2526	2527
4740	2528	2529	2530	2531	2532	2533	2534	2535
4750	2536	2537	2538	2539	2540	2541	2542	2543
4760	2544	2545	2546	2547	2548	2549	2550	2551
4770	2552	2553	2554	2555	2556	2557	2558	2559

4000 to 4777 (Octal) 2048 to 2559 (Decimal)

Octal	Decimal
10000	4096
20000	8192
30000	12288
40000	16384
50000	20480
60000	24576
70000	28672

	0	1	2	3	4	5	6	7
5000	2560	2561	2562	2563	2564	2565	2566	2567
5010	2568	2569	2570	2571	2572	2573	2574	2575
5020	2576	2577	2578	2579	2580	2581	2582	2583
5030	2584	2585	2586	2587	2588	2589	2590	2591
5040	2592	2593	2594	2595	2596	2597	2598	2599
5050	2600	2601	2602	2603	2604	2605	2606	2607
5060	2608	2609	2610	2611	2612	2613	2614	2615
5070	2616	2617	2618	2619	2620	2621	2622	2623
5100	2624	2625	2626	2627	2628	2629	2630	2631
5110	2632	2633	2634	2635	2636	2637	2638	2639
5120	2640	2641	2642	2643	2644	2645	2646	2647
5130	2648	2649	2650	2651	2652	2653	2654	2655
5140	2656	2657	2658	2659	2660	2661	2662	2663
5150	2664	2665	2666	2667	2668	2669	2670	2671
5160	2672	2673	2674	2675	2676	2677	2678	2679
5170	2680	2681	2682	2683	2684	2685	2686	2687
5200	2688	2689	2690	2691	2692	2693	2694	2695
5210	2696	2697	2698	2699	2700	2701	2702	2703
5220	2704	2705	2706	2707	2708	2709	2710	2711
5230	2712	2713	2714	2715	2716	2717	2718	2719
5240	2720	2721	2722	2723	2724	2725	2726	2727
5250	2728	2729	2730	2731	2732	2733	2734	2735
5260	2736	2737	2738	2739	2740	2741	2742	2743
5270	2744	2745	2746	2747	2748	2749	2750	2751
5300	2752	2753	2754	2755	2756	2757	2758	2759
5310	2760	2761	2762	2763	2764	2765	2766	2767
5320	2768	2769	2770	2771	2772	2773	2774	2775
5330	2776	2777	2778	2779	2780	2781	2782	2783
5340	2784	2785	2786	2787	2788	2789	2790	2791
5350	2792	2793	2794	2795	2796	2797	2798	2799
5360	2800	2801	2802	2803	2804	2805	2806	2807
5370	2808	2809	2810	2811	2812	2813	2814	2815

	0	1	2	3	4	5	6	7
5400	2816	2817	2818	2819	2820	2821	2822	2823
5410	2824	2825	2826	2827	2828	2829	2830	2831
5420	2832	2833	2834	2835	2836	2837	2838	2839
5430	2840	2841	2842	2843	2844	2845	2846	2847
5440	2848	2849	2850	2851	2852	2853	2854	2855
5450	2856	2857	2858	2859	2860	2861	2862	2863
5460	2864	2865	2866	2867	2868	2869	2870	2871
5470	2872	2873	2874	2875	2876	2877	2878	2879
5500	2880	2881	2882	2883	2884	2885	2886	2887
5510	2888	2889	2890	2891	2892	2893	2894	2895
5520	2896	2897	2898	2899	2900	2901	2902	2903
5530	2904	2905	2906	2907	2908	2909	2910	2911
5540	2912	2913	2914	2915	2916	2917	2918	2919
5550	2920	2921	2922	2923	2924	2925	2926	2927
5560	2928	2929	2930	2931	2932	2933	2934	2935
5570	2936	2937	2938	2939	2940	2941	2942	2943
5600	2944	2945	2946	2947	2948	2949	2950	2951
5610	2952	2953	2954	2955	2956	2957	2958	2959
5620	2960	2961	2962	2963	2964	2965	2966	2967
5630	2968	2969	2970	2971	2972	2973	2974	2975
5640	2976	2977	2978	2979	2980	2981	2982	2983
5650	2984	2985	2986	2987	2988	2989	2990	2991
5660	2992	2993	2994	2995	2996	2997	2998	2999
5670	3000	3001	3002	3003	3004	3005	3006	3007
5700	3008	3009	3010	3011	3012	3013	3014	3015
5710	3016	3017	3018	3019	3020	3021	3022	3023
5720	3024	3025	3026	3027	3028	3029	3030	3031
5730	3032	3033	3034	3035	3036	3037	3038	3039
5740	3040	3041	3042	3043	3044	3045	3046	3047
5750	3048	3049	3050	3051	3052	3053	3054	3055
5760	3056	3057	3058	3059	3060	3061	3062	3063
5770	3064	3065	3066	3067	3068	3069	3070	3071

5000 to 5777 (Octal) 2560 to 3071 (Decimal)

OCTAL-DECIMAL INTEGER CONVERSION TABLE (Cont'd)

6000 3072
to to
6777 3583
(Octal) (Decimal)

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
6000	3072	3073	3074	3075	3076	3077	3078	3079
6010	3080	3081	3082	3083	3084	3085	3086	3087
6020	3088	3089	3090	3091	3092	3093	3094	3095
6030	3096	3097	3098	3099	3100	3101	3102	3103
6040	3104	3105	3106	3107	3108	3109	3110	3111
6050	3112	3113	3114	3115	3116	3117	3118	3119
6060	3120	3121	3122	3123	3124	3125	3126	3127
6070	3128	3129	3130	3131	3132	3133	3134	3135
6100	3136	3137	3138	3139	3140	3141	3142	3143
6110	3144	3145	3146	3147	3148	3149	3150	3151
6120	3152	3153	3154	3155	3156	3157	3158	3159
6130	3160	3161	3162	3163	3164	3165	3166	3167
6140	3168	3169	3170	3171	3172	3173	3174	3175
6150	3176	3177	3178	3179	3180	3181	3182	3183
6160	3184	3185	3186	3187	3188	3189	3190	3191
6170	3192	3193	3194	3195	3196	3197	3198	3199
6200	3200	3201	3202	3203	3204	3205	3206	3207
6210	3208	3209	3210	3211	3212	3213	3214	3215
6220	3216	3217	3218	3219	3220	3221	3222	3223
6230	3224	3225	3226	3227	3228	3229	3230	3231
6240	3232	3233	3234	3235	3236	3237	3238	3239
6250	3240	3241	3242	3243	3244	3245	3246	3247
6260	3248	3249	3250	3251	3252	3253	3254	3255
6270	3256	3257	3258	3259	3260	3261	3262	3263
6300	3264	3265	3266	3267	3268	3269	3270	3271
6310	3272	3273	3274	3275	3276	3277	3278	3279
6320	3280	3281	3282	3283	3284	3285	3286	3287
6330	3288	3289	3290	3291	3292	3293	3294	3295
6340	3296	3297	3298	3299	3300	3301	3302	3303
6350	3304	3305	3306	3307	3308	3309	3310	3311
6360	3312	3313	3314	3315	3316	3317	3318	3319
6370	3320	3321	3322	3323	3324	3325	3326	3327

	0	1	2	3	4	5	6	7
6400	3328	3329	3330	3331	3332	3333	3334	3335
6410	3336	3337	3338	3339	3340	3341	3342	3343
6420	3344	3345	3346	3347	3348	3349	3350	3351
6430	3352	3353	3354	3355	3356	3357	3358	3359
6440	3360	3361	3362	3363	3364	3365	3366	3367
6450	3368	3369	3370	3371	3372	3373	3374	3375
6460	3376	3377	3378	3379	3380	3381	3382	3383
6470	3384	3385	3386	3387	3388	3389	3390	3391
6500	3392	3393	3394	3395	3396	3397	3398	3399
6510	3400	3401	3402	3403	3404	3405	3406	3407
6520	3408	3409	3410	3411	3412	3413	3414	3415
6530	3416	3417	3418	3419	3420	3421	3422	3423
6540	3424	3425	3426	3427	3428	3429	3430	3431
6550	3432	3433	3434	3435	3436	3437	3438	3439
6560	3440	3441	3442	3443	3444	3445	3446	3447
6570	3448	3449	3450	3451	3452	3453	3454	3455
6600	3456	3457	3458	3459	3460	3461	3462	3463
6610	3464	3465	3466	3467	3468	3469	3470	3471
6620	3472	3473	3474	3475	3476	3477	3478	3479
6630	3480	3481	3482	3483	3484	3485	3486	3487
6640	3488	3489	3490	3491	3492	3493	3494	3495
6650	3496	3497	3498	3499	3500	3501	3502	3503
6660	3504	3505	3506	3507	3508	3509	3510	3511
6670	3512	3513	3514	3515	3516	3517	3518	3519
6700	3520	3521	3522	3523	3524	3525	3526	3527
6710	3528	3529	3530	3531	3532	3533	3534	3535
6720	3536	3537	3538	3539	3540	3541	3542	3543
6730	3544	3545	3546	3547	3548	3549	3550	3551
6740	3552	3553	3554	3555	3556	3557	3558	3559
6750	3560	3561	3562	3563	3564	3565	3566	3567
6760	3568	3569	3570	3571	3572	3573	3574	3575
6770	3576	3577	3578	3579	3580	3581	3582	3583

7000 3584
to to
7777 4095
(Octal) (Decimal)

	0	1	2	3	4	5	6	7
7000	3584	3585	3586	3587	3588	3589	3590	3591
7010	3592	3593	3594	3595	3596	3597	3598	3599
7020	3600	3601	3602	3603	3604	3605	3606	3607
7030	3608	3609	3610	3611	3612	3613	3614	3615
7040	3616	3617	3618	3619	3620	3621	3622	3623
7050	3624	3625	3626	3627	3628	3629	3630	3631
7060	3632	3633	3634	3635	3636	3637	3638	3639
7070	3640	3641	3642	3643	3644	3645	3646	3647
7100	3648	3649	3650	3651	3652	3653	3654	3655
7110	3656	3657	3658	3659	3660	3661	3662	3663
7120	3664	3665	3666	3667	3668	3669	3670	3671
7130	3672	3673	3674	3675	3676	3677	3678	3679
7140	3680	3681	3682	3683	3684	3685	3686	3687
7150	3688	3689	3690	3691	3692	3693	3694	3695
7160	3696	3697	3698	3699	3700	3701	3702	3703
7170	3704	3705	3706	3707	3708	3709	3710	3711
7200	3712	3713	3714	3715	3716	3717	3718	3719
7210	3720	3721	3722	3723	3724	3725	3726	3727
7220	3728	3729	3730	3731	3732	3733	3734	3735
7230	3736	3737	3738	3739	3740	3741	3742	3743
7240	3744	3745	3746	3747	3748	3749	3750	3751
7250	3752	3753	3754	3755	3756	3757	3758	3759
7260	3760	3761	3762	3763	3764	3765	3766	3767
7270	3768	3769	3770	3771	3772	3773	3774	3775
7300	3776	3777	3778	3779	3780	3781	3782	3783
7310	3784	3785	3786	3787	3788	3789	3790	3791
7320	3792	3793	3794	3795	3796	3797	3798	3799
7330	3800	3801	3802	3803	3804	3805	3806	3807
7340	3808	3809	3810	3811	3812	3813	3814	3815
7350	3816	3817	3818	3819	3820	3821	3822	3823
7360	3824	3825	3826	3827	3828	3829	3830	3831
7370	3832	3833	3834	3835	3836	3837	3838	3839

	0	1	2	3	4	5	6	7
7400	3840	3841	3842	3843	3844	3845	3846	3847
7410	3848	3849	3850	3851	3852	3853	3854	3855
7420	3856	3857	3858	3859	3860	3861	3862	3863
7430	3864	3865	3866	3867	3868	3869	3870	3871
7440	3872	3873	3874	3875	3876	3877	3878	3879
7450	3880	3881	3882	3883	3884	3885	3886	3887
7460	3888	3889	3890	3891	3892	3893	3894	3895
7470	3896	3897	3898	3899	3900	3901	3902	3903
7500	3904	3905	3906	3907	3908	3909	3910	3911
7510	3912	3913	3914	3915	3916	3917	3918	3919
7520	3920	3921	3922	3923	3924	3925	3926	3927
7530	3928	3929	3930	3931	3932	3933	3934	3935
7540	3936	3937	3938	3939	3940	3941	3942	3943
7550	3944	3945	3946	3947	3948	3949	3950	3951
7560	3952	3953	3954	3955	3956	3957	3958	3959
7570	3960	3961	3962	3963	3964	3965	3966	3967
7600	3968	3969	3970	3971	3972	3973	3974	3975
7610	3976	3977	3978	3979	3980	3981	3982	3983
7620	3984	3985	3986	3987	3988	3989	3990	3991
7630	3992	3993	3994	3995	3996	3997	3998	3999
7640	4000	4001	4002	4003	4004	4005	4006	4007
7650	4008	4009	4010	4011	4012	4013	4014	4015
7660	4016	4017	4018	4019	4020	4021	4022	4023
7670	4024	4025	4026	4027	4028	4029	4030	4031
7700	4032	4033	4034	4035	4036	4037	4038	4039
7710	4040	4041	4042	4043	4044	4045	4046	4047
7720	4048	4049	4050	4051	4052	4053	4054	4055
7730	4056	4057	4058	4059	4060	4061	4062	4063
7740	4064	4065	4066	4067	4068	4069	4070	4071
7750	4072	4073	4074	4075	4076	4077	4078	4079
7760	4080	4081	4082	4083	4084	4085	4086	4087
7770	4088	4089	4090	4091	4092	4093	4094	4095

OCTAL-DECIMAL FRACTION CONVERSION TABLE

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.000	.000000	.100	.125000	.200	.250000	.300	.375000
.001	.001953	.101	.126953	.201	.251953	.301	.376953
.002	.003906	.102	.128906	.202	.253906	.302	.378906
.003	.005859	.103	.130859	.203	.255859	.303	.380859
.004	.007812	.104	.132812	.204	.257812	.304	.382812
.005	.009765	.105	.134765	.205	.259765	.305	.384765
.006	.011718	.106	.136718	.206	.261718	.306	.386718
.007	.013671	.107	.138671	.207	.263671	.307	.388671
.010	.015625	.110	.140625	.210	.265625	.310	.390625
.011	.017578	.111	.142578	.211	.267578	.311	.392578
.012	.019531	.112	.144531	.212	.269531	.312	.394531
.013	.021484	.113	.146484	.213	.271484	.313	.396484
.014	.023437	.114	.148437	.214	.273437	.314	.398437
.015	.025390	.115	.150390	.215	.275390	.315	.400390
.016	.027343	.116	.152343	.216	.277343	.316	.402343
.017	.029296	.117	.154296	.217	.279296	.317	.404296
.020	.031250	.120	.156250	.220	.281250	.320	.406250
.021	.033203	.121	.158203	.221	.283203	.321	.408203
.022	.035156	.122	.160156	.222	.285156	.322	.410156
.023	.037109	.123	.162109	.223	.287109	.323	.412109
.024	.039062	.124	.164062	.224	.289062	.324	.414062
.025	.041015	.125	.166015	.225	.291015	.325	.416015
.026	.042968	.126	.167968	.226	.292968	.326	.417968
.027	.044921	.127	.169921	.227	.294921	.327	.419921
.030	.046875	.130	.171875	.230	.296875	.330	.421875
.031	.048828	.131	.173828	.231	.298828	.331	.423828
.032	.050781	.132	.175781	.232	.300781	.332	.425781
.033	.052734	.133	.177734	.233	.302734	.333	.427734
.034	.054687	.134	.179687	.234	.304687	.334	.429687
.035	.056640	.135	.181640	.235	.306640	.335	.431640
.036	.058593	.136	.183593	.236	.308593	.336	.433593
.037	.060546	.137	.185546	.237	.310546	.337	.435546
.040	.062500	.140	.187500	.240	.312500	.340	.437500
.041	.064453	.141	.189453	.241	.314453	.341	.439453
.042	.066406	.142	.191406	.242	.316406	.342	.441406
.043	.068359	.143	.193359	.243	.318359	.343	.443359
.044	.070312	.144	.195312	.244	.320312	.344	.445312
.045	.072265	.145	.197265	.245	.322265	.345	.447265
.046	.074218	.146	.199218	.246	.324218	.346	.449218
.047	.076171	.147	.201171	.247	.326171	.347	.451171
.050	.078125	.150	.203125	.250	.328125	.350	.453125
.051	.080078	.151	.205078	.251	.330078	.351	.455078
.052	.082031	.152	.207031	.252	.332031	.352	.457031
.053	.083984	.153	.208984	.253	.333984	.353	.458984
.054	.085937	.154	.210937	.254	.335937	.354	.460937
.055	.087890	.155	.212890	.255	.337890	.355	.462890
.056	.089843	.156	.214843	.256	.339843	.356	.464843
.057	.091796	.157	216796	.257	.341796	.357	.466796
.060	.093750	.160	.218750	.260	.343750	.360	.468750
.061	.095703	.161	.220703	.261	.345703	.361	.470703
.062	.097656	.162	.222656	.262	.347656	.362	.472656
.063	.099609	.163	.224609	.263	.349609	.363	.474609
.064	.101562	.164	.226562	.264	.351562	.364	.476562
.065	.103515	.165	.228515	.265	.353515	.365	.478515
.066	.105468	.166	.230468	.266	.355468	.366	.480468
.067	.107421	.167	.232421	.267	.357421	.367	.482421
.070	.109375	.170	.234375	.270	.359375	.370	.484375
.071	.111328	.171	.236328	.271	.361328	.371	.486328
.072	.113281	.172	.238281	.272	.363281	.372	.488281
.073	.115234	.173	.240234	.273	.365234	.373	.490234
.074	.117187	.174	.242187	.274	.367187	.374	.492187
.075	.119140	.175	.244140	.275	.369140	.375	.494140
.076	.121093	.176	.246093	.276	.371093	.376	.496093
.077	.123046	.177	.248046	.277	.373046	.377	.498046

OCTAL-DECIMAL FRACTION CONVERSION TABLE

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.400	.500000	.500	.625000	.600	.750000	.700	.875000
.401	.501953	.501	.626953	.601	.751953	.701	.876953
.402	.503906	.502	.628906	.602	.753906	.702	.878906
.403	.505859	.503	.630859	.603	.755859	.703	.880859
.404	.507812	.504	.632812	.604	.757812	.704	.882812
.405	.509765	.505	.634765	.605	.759765	.705	.884765
.406	.511718	.506	.636718	.606	.761718	.706	.886718
.407	.513671	.507	.638671	.607	.763671	.707	.888671
.410	.515625	.510	.640625	.610	.765625	.710	.890625
.411	.517578	.511	.642578	.611	.767578	.711	.892578
.412	.519531	.512	.644531	.612	.769531	.712	.894531
.413	.521484	.513	.646484	.613	.771484	.713	.896484
.414	.523437	.514	.648437	.614	.773437	.714	.898437
.415	.525390	.515	.650390	.615	.775390	.715	.900390
.416	.527343	.516	.652343	.616	.777343	.716	.902343
.417	.529296	.517	.654296	.617	.779296	.717	.904296
.420	.531250	.520	.656250	.620	.781250	.720	.906250
.421	.533203	.521	.658203	.621	.783203	.721	.908203
.422	.535156	.522	.660156	.622	.785156	.722	.910156
.423	.537109	.523	.662109	.623	.787109	.723	.912109
.424	.539062	.524	.664062	.624	.789062	.724	.914062
.425	.541015	.525	.666015	.625	.791015	.725	.916015
.426	.542968	.526	.667968	.626	.792968	.726	.917968
.427	.544921	.527	.669921	.627	.794921	.727	.919921
.430	.546875	.530	.671875	.630	.796875	.730	.921875
.431	.548828	.531	.673828	.631	.798828	.731	.923828
.432	.550781	.532	.675781	.632	.800781	.732	.925781
.433	.552734	.533	.677734	.633	.802734	.733	.927734
.434	.554687	.534	.679687	.634	.804687	.734	.929687
.435	.556640	.535	.681640	.635	.806640	.735	.931640
.436	.558593	.536	.683593	.636	.808593	.736	.933593
.437	.560546	.537	.685546	.637	.810546	.737	.935546
.440	.562500	.540	.687500	.640	.812500	.740	.937500
.441	.564453	.541	.689453	.641	.814453	.741	.939453
.442	.566406	.542	.691406	.642	.816406	.742	.941406
.443	.568359	.543	.693359	.643	.818359	.743	.943359
.444	.570312	.544	.695312	.644	.820312	.744	.945312
.445	.572265	.545	.697265	.645	.822265	.745	.947265
.446	.574218	.546	.699218	.646	.824218	.746	.949218
.447	.576171	.547	.701171	.647	.826171	.747	.951171
.450	.578125	.550	.703125	.650	.828125	.750	.953125
.451	.580078	.551	.705078	.651	.830078	.751	.955078
.452	.582031	.552	.707031	.652	.832031	.752	.957031
.453	.583984	.553	.708984	.653	.833984	.753	.958984
.454	.585937	.554	.710937	.654	.835937	.754	.960937
.455	.587890	.555	.712890	.655	.837890	.755	.962890
.456	.589843	.556	.714843	.656	.839843	.756	.964843
.457	.591796	.557	.716796	.657	.841796	.757	.966796
.460	.593750	.560	.718750	.660	.843750	.760	.968750
.461	.595703	.561	.720703	.661	.845703	.761	.970703
.462	.597656	.562	.722656	.662	.847656	.762	.972656
.463	.599609	.563	.724609	.663	.849609	.763	.974609
.464	.601562	.564	.726562	.664	.851562	.764	.976562
.465	.603515	.565	.728515	.665	.853515	.765	.978515
.466	.605468	.566	.730468	.666	.855468	.766	.980468
.467	.607421	.567	.732421	.667	.857421	.767	.982421
.470	.609375	.570	.734375	.670	.859375	.770	.984375
.471	.611328	.571	.736328	.671	.861328	.771	.986328
.472	.613281	.572	.738281	.672	.863281	.772	.988281
.473	.615234	.573	.740234	.673	.865234	.773	.990234
.474	.617187	.574	.642187	.674	.867187	.774	.992187
.475	.619140	.575	.744140	.675	.869140	.775	.994140
.476	.621093	.576	.746093	.676	.871093	.776	.996093
.477	.623046	.577	.748046	.677	.873046	.777	.998046

OCTAL-DECIMAL FRACTION CONVERSION TABLE (Cont'd)

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.000000	.000000	.000100	.000244	.000200	.000488	.000300	.000732
.000001	.000003	.000101	.000247	.000201	.000492	.000301	.000736
.000002	.000007	.000102	.000251	.000202	.000495	.000302	.000740
.000003	.000011	.000103	.000255	.000203	.000499	.000303	.000743
.000004	.000015	.000104	.000259	.000204	.000503	.000304	.000747
.000005	.000019	.000105	.000263	.000205	.000507	.000305	.000751
.000006	.000022	.000106	.000267	.000206	.000511	.000306	.000755
.000007	.000026	.000107	.000270	.000207	.000514	.000307	.000759
.000010	.000030	.000110	.000274	.000210	.000518	.000310	.000762
.000011	.000034	.000111	.000278	.000211	.000522	.000311	.000766
.000012	.000038	.000112	.000282	.000212	.000526	.000312	.000770
.000013	.000041	.000113	.000286	.000213	.000530	.000313	.000774
.000014	.000045	.000114	.000289	.000214	.000534	.000314	.000778
.000015	.000049	.000115	.000293	.000215	.000537	.000315	.000782
.000016	.000053	.000116	.000297	.000216	.000541	.000316	.000785
.000017	.000057	.000117	.000301	.000217	.000545	.000317	.000789
.000020	.000061	.000120	.000305	.000220	.000549	.000320	.000793
.000021	.000064	.000121	.000308	.000221	.000553	.000321	.000797
.000022	.000068	.000122	.000312	.000222	.000556	.000322	.000801
.000023	.000072	.000123	.000316	.000223	.000560	.000323	.000805
.000024	.000076	.000124	.000320	.000224	.000564	.000324	.000808
.000025	.000080	.000125	.000324	.000225	.000568	.000325	.000812
.000026	.000083	.000126	.000328	.000226	.000572	.000326	.000816
.000027	.000087	.000127	.000331	.000227	.000576	.000327	.000820
.000030	.000091	.000130	.000335	.000230	.000579	.000330	.000823
.000031	.000095	.000131	.000339	.000231	.000583	.000331	.000827
.000032	.000099	.000132	.000343	.000232	.000587	.000332	.000831
.000033	.000102	.000133	.000347	.000233	.000591	.000333	.000835
.000034	.000106	.000134	.000350	.000234	.000595	.000334	.000839
.000035	.000110	.000135	.000354	.000235	.000598	.000335	.000843
.000036	.000114	.000136	.000358	.000236	.000602	.000336	.000846
.000037	.000118	.000137	.000362	.000237	.000606	.000337	.000850
.000040	.000122	.000140	.000366	.000240	.000610	.000340	.000854
.000041	.000125	.000141	.000370	.000241	.000614	.000341	.000858
.000042	.000129	.000142	.000373	.000242	.000617	.000342	.000862
.000043	.000133	.000143	.000377	.000243	.000621	.000343	.000865
.000044	.000137	.000144	.000381	.000244	.000625	.000344	.000869
.000045	.000141	.000145	.000385	.000245	.000629	.000345	.000873
.000046	.000144	.000146	.000389	.000246	.000633	.000346	.000877
.000047	.000148	.000147	.000392	.000247	.000637	.000347	.000881
.000050	.000152	.000150	.000396	.000250	.000640	.000350	.000885
.000051	.000156	.000151	.000400	.000251	.000644	.000351	.000888
.000052	.000160	.000152	.000404	.000252	.000648	.000352	.000892
.000053	.000164	.000153	.000408	.000253	.000652	.000353	.000896
.000054	.000167	.000154	.000411	.000254	.000656	.000354	.000900
.000055	.000171	.000155	.000415	.000255	.000659	.000355	.000904
.000056	.000175	.000156	.000419	.000256	.000663	.000356	.000907
.000057	.000179	.000157	.000423	.000257	.000667	.000357	.000911
.000060	.000183	.000160	.000427	.000260	.000671	.000360	.000915
.000061	.000186	.000161	.000431	.000261	.000675	.000361	.000919
.000062	.000190	.000162	.000434	.000262	.000679	.000362	.000923
.000063	.000194	.000163	.000438	.000263	.000682	.000363	.000926
.000064	.000198	.000164	.000442	.000264	.000686	.000364	.000930
.000065	.000202	.000165	.000446	.000265	.000690	.000365	.000934
.000066	.000205	.000166	.000450	.000266	.000694	.000366	.000938
.000067	.000209	.000167	.000453	.000267	.000698	.000367	.000942
.000070	.000213	.000170	.000457	.000270	.000701	.000370	.000946
.000071	.000217	.000171	.000461	.000271	.000705	.000371	.000949
.000072	.000221	.000172	.000465	.000272	.000709	.000372	.000953
.000073	.000225	.000173	.000469	.000273	.000713	.000373	.000957
.000074	.000228	.000174	.000473	.000274	.000717	.000374	.000961
.000075	.000232	.000175	.000476	.000275	.000720	.000375	.000965
.000076	.000236	.000176	.000480	.000276	.000724	.000376	.000968
.000077	.000240	.000177	.000484	.000277	.000728	.000377	.000972

OCTAL-DECIMAL FRACTION CONVERSION TABLE (Cont'd)

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.000400	.000976	.000500	.001220	.000600	.001464	.000700	.001708
.000401	.000980	.000501	.001224	.000601	.001468	.000701	.001712
.000402	.000984	.000502	.001228	.000602	.001472	.000702	.001716
.000403	.000988	.000503	.001232	.000603	.001476	.000703	.001720
.000404	.000991	.000504	.001235	.000604	.001480	.000704	.001724
.000405	.000995	.000505	.001239	.000605	.001483	.000705	.001728
.000406	.000999	.000506	.001243	.000606	.001487	.000706	.001731
.000407	.001003	.000507	.001247	.000607	.001491	.000707	.001735
.000410	.001007	.000510	.001251	.000610	.001495	.000710	.001739
.000411	.001010	.000511	.001255	.000611	.001499	.000711	.001743
.000412	.001014	.000512	.001258	.000612	.001502	.000712	.001747
.000413	.001018	.000513	.001262	.000613	.001506	.000713	.001750
.000414	.001022	.000514	.001266	.000614	.001510	.000714	.001754
.000415	.001026	.000515	.001270	.000615	.001514	.000715	.001758
.000416	.001029	.000516	.001274	.000616	.001518	.000716	.001762
.000417	.001033	.000517	.001277	.000617	.001522	.000717	.001766
.000420	.001037	.000520	.001281	.000620	.001525	.000720	.001770
.000421	.001041	.000521	.001285	.000621	.001529	.000721	.001773
.000422	.001045	.000522	.001289	.000622	.001533	.000722	.001777
.000423	.001049	.000523	.001293	.000623	.001537	.000723	.001781
.000424	.001052	.000524	.001296	.000624	.001541	.000724	.001785
.000425	.001056	.000525	.001300	.000625	.001544	.000725	.001789
.000426	.001060	.000526	.001304	.000626	.001548	.000726	.001792
.000427	.001064	.000527	.001308	.000627	.001552	.000727	.001796
.000430	.001068	.000530	.001312	.000630	.001556	.000730	.001800
.000431	.001071	.000531	.001316	.000631	.001560	.000731	.001804
.000432	.001075	.000532	.001319	.000632	.001564	.000732	.001808
.000433	.001079	.000533	.001323	.000633	.001567	.000733	.001811
.000434	.001083	.000534	.001327	.000634	.001571	.000734	.001815
.000435	.001087	.000535	.001331	.000635	.001575	.000735	.001819
.000436	.001091	.000536	.001335	.000636	.001579	.000736	.001823
.000437	.001094	.000537	.001338	.000637	.001583	.000737	.001827
.000440	.001098	.000540	.001342	.000640	.001586	.000740	.001831
.000441	.001102	.000541	.001346	.000641	.001590	.000741	.001834
.000442	.001106	.000542	.001350	.000642	.001594	.000742	.001838
.000443	.001110	.000543	.001354	.000643	.001598	.000743	.001842
.000444	.001113	.000544	.001358	.000644	.001602	.000744	.001846
.000445	.001117	.000545	.001361	.000645	.001605	.000745	.001850
.000446	.001121	.000546	.001365	.000646	.001609	.000746	.001853
.000447	.001125	.000547	.001369	.000647	.001613	.000747	.001857
.000450	.001129	.000550	.001373	.000650	.001617	.000750	.001861
.000451	.001132	.000551	.001377	.000651	.001621	.000751	.001865
.000452	.001136	.000552	.001380	.000652	.001625	.000752	.001869
.000453	.001140	.000553	.001384	.000653	.001628	.000753	.001873
.000454	.001144	.000554	.001388	.000654	.001632	.000754	.001876
.000455	.001148	.000555	.001392	.000655	.001636	.000755	.001880
.000456	.001152	.000556	.001396	.000656	.001640	.000756	.001884
.000457	.001155	.000557	.001399	.000657	.001644	.000757	.001888
.000460	.001159	.000560	.001403	.000660	.001647	.000760	.001892
.000461	.001163	.000561	.001407	.000661	.001651	.000761	.001895
.000462	.001167	.000562	.001411	.000662	.001655	.000762	.001899
.000463	.001171	.000563	.001415	.000663	.001659	.000763	.001903
.000464	.001174	.000564	.001419	.000664	.001663	.000764	.001907
.000465	.001178	.000565	.001422	.000665	.001667	.000765	.001911
.000466	.001182	.000566	.001426	.000666	.001670	.000766	.001914
.000467	.001186	.000567	.001430	.000667	.001674	.000767	.001918
.000470	.001190	.000570	.001434	.000670	.001678	.000770	.001922
.000471	.001194	.000571	.001438	.000671	.001682	.000771	.001926
.000472	.001197	.000572	.001441	.000672	.001686	.000772	.001930
.000473	.001201	.000573	.001445	.000673	.001689	.000773	.001934
.000474	.001205	.000574	.001449	.000674	.001693	.000774	.001937
.000475	.001209	.000575	.001453	.000675	.001697	.000775	.001941
.000476	.001213	.000576	.001457	.000676	.001701	.000776	.001945
.000477	.001216	.000577	.001461	.000677	.001705	.000777	.001949

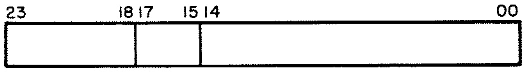
APPENDIX D

INSTRUCTION FORMATS AND NOTES

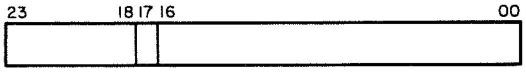
D. INSTRUCTION FORMATS AND NOTES

The formats below correspond to the mnemonic instructions listed in Table D-1.

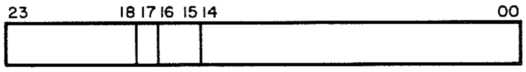
F1



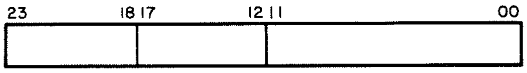
F2



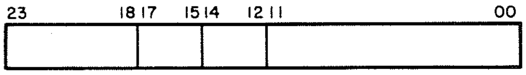
F3



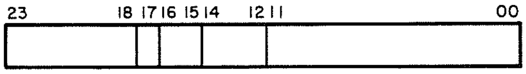
F4



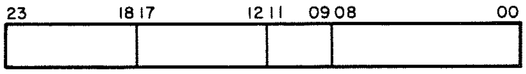
F5



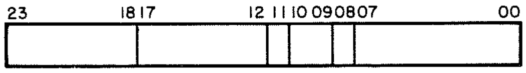
F6



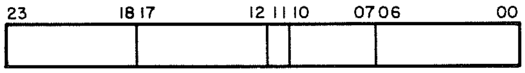
F7



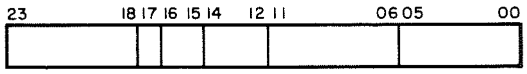
F8



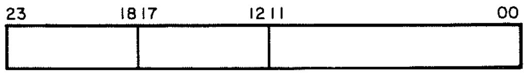
F9



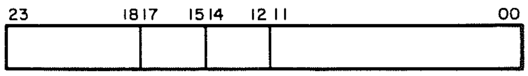
F10

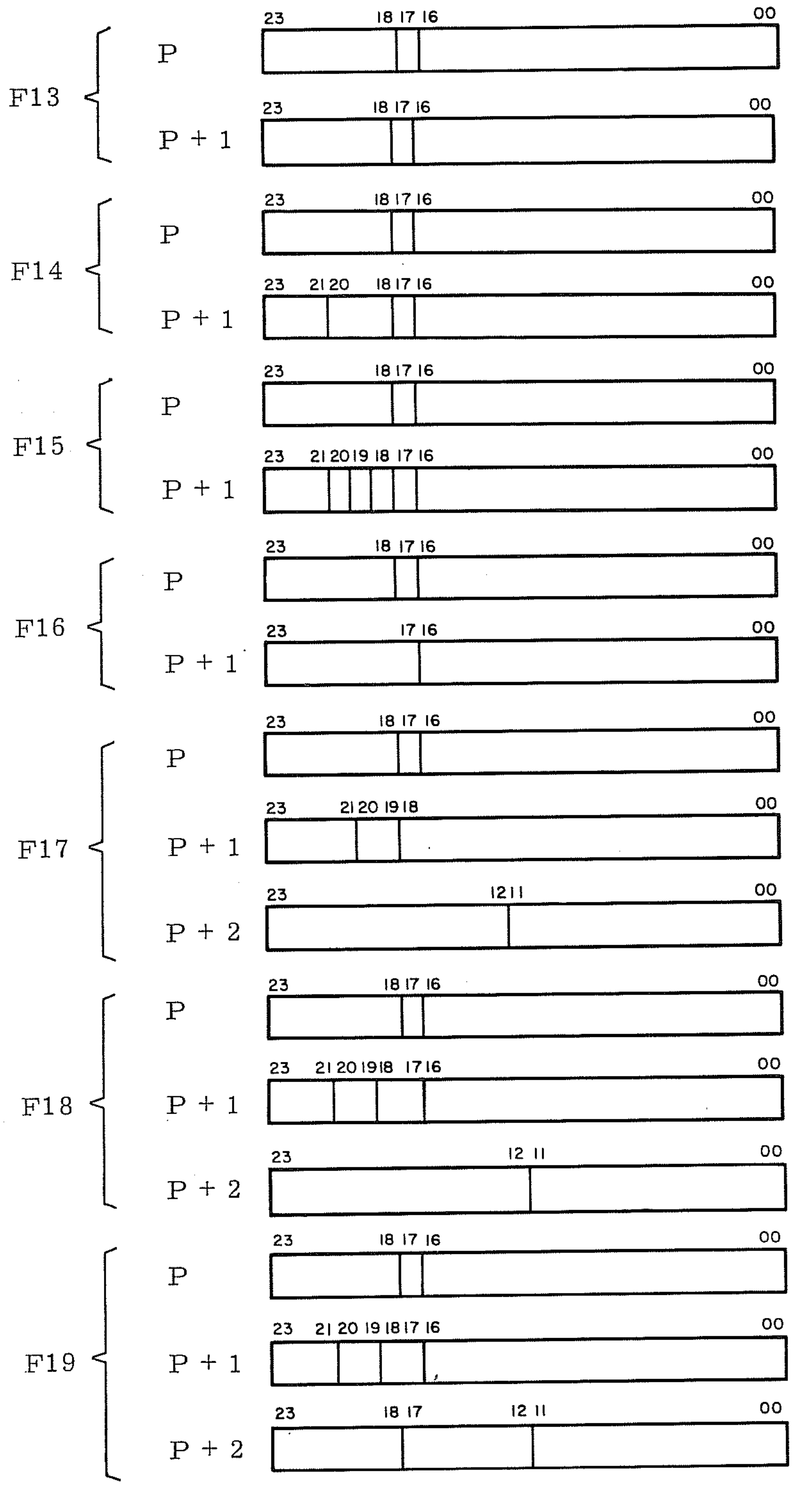


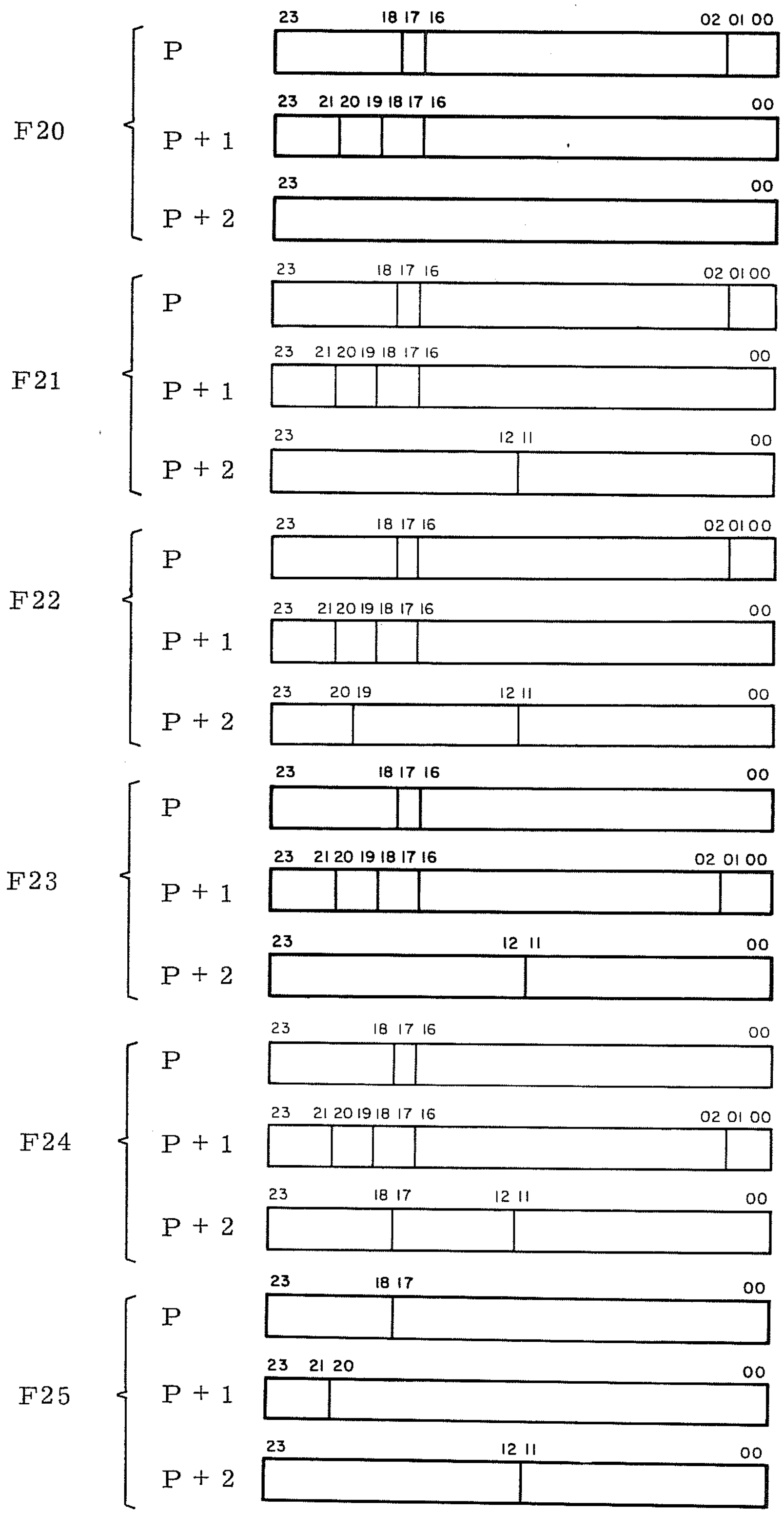
F11



F12




F13
P
23 18 17 16 00
P + 1
23 18 17 16 00
F14
P
23 18 17 16 00
P + 1
23 21 20 18 17 16 00
F15
P
23 18 17 16 00
P + 1
23 21 20 19 18 17 16 00
F16
P
23 18 17 16 00
P + 1
23 17 16 00
F17
P
23 18 17 16 00
P + 1
23 21 20 19 18 00
P + 2
23 12 11 00
F18
P
23 18 17 16 00
P + 1
23 21 20 19 18 17 16 00
P + 2
23 12 11 00
F19
P
23 18 17 16 00
P + 1
23 21 20 19 18 17 16 00
P + 2
23 18 17 12 11 00


F20
P
23 18 17 16 02 01 00
P + 1
23 21 20 19 18 17 16 00
P + 2
23 00
F21
P
23 18 17 16 02 01 00
P + 1
23 21 20 19 18 17 16 00
P + 2
23 12 11 00
F22
P
23 18 17 16 02 01 00
P + 1
23 21 20 19 18 17 16 00
P + 2
23 20 19 12 11 00
F23
P
23 18 17 16 00
P + 1
23 21 20 19 18 17 16 02 01 00
P + 2
23 12 11 00
F24
P
23 18 17 16 00
P + 1
23 21 20 19 18 17 16 02 01 00
P + 2
23 18 17 12 11 00
F25
P
23 18 17 00
P + 1
23 21 20 00
P + 2
23 12 11 00

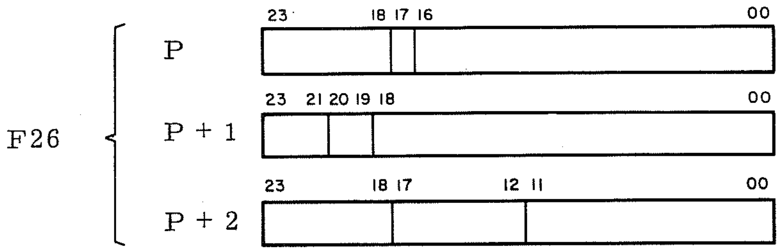

F26
P
23
18 17 16
00
P + 1
23
21 20 19 18
00
P + 2
23
18 17
12 11
00

TABLE D-1. INSTRUCTION FORMATS

Mnemonic Code	Basic Octal Code	Instruction Format	Page No.
ACI	77	F4	5-38
ACR	77	F7	5-40
ADA, I	30	F3	5-60
ADAQ, I	32	F3	5-61
ADM	67	F18	5-68
AEU	55	F1	5-36
AIA	53	F6	5-33
AIS	77	F7	5-37
ANA	17	F1	5-73
ANA, S	17	F1	5-73
ANI	17	F3	5-73
ANQ	17	F1	5-74
ANQ, S	17	F1	5-74
AOS	77	F4	5-37
APF	77	F9	5-39
AQA	53	F5	5-32
AQE	55	F1	5-36
AQJ, EQ	03	F3	5-46
AQJ, GE	03	F3	5-46
AQJ, LT	03	F3	5-46
AQJ, NE	03	F3	5-46
ASE	04	F1	5-29
ASE, S	04	F1	5-29
ASG	05	F1	5-30
ASG, S	05	F1	5-30
ATD	66	F21	5-119
ATD, D	66	F22	5-120
AZJ, EQ	03	F3	5-45
AZJ, GE	03	F3	5-45
AZJ, LT	03	F3	5-45
AZJ, NE	03	F3	5-45
CIA	77	F4	5-38
CILO	77	F8	5-91

TABLE D-1. INSTRUCTION FORMATS (Cont'd)

Mnemonic Code	Basic Octal Code	Instruction Format	Page No.
CINS	77	F5	5-86
CLCA	77	F8	5-94
CMP	67	F18	5-79
CMP, DC	67	F19	5-80
CON	77	F12	5-95
COPY	77	F5	5-83
CPR, I	52	F3	5-77
CRA	77	F4	5-40
CTI	77	F4	5-97
CTO	77	F4	5-97
CVBD	66	F20	5-116
CVDB	66	F23	5-115
DINT	77	F4	5-89
DTA	66	F23	5-117
DTA, DC	66	F24	5-118
DVA, I	51	F3	5-62
DVAQ, I	57	F3	5-63
EAQ	55	F1	5-36
ECHA	11	F2	5-26
ECHA, S	11	F2	5-26
EDIT	64	F18	5-149
EINT	77	F4	5-89
ELQ	55	F1	5-36
ENA	14	F1	5-25
ENA, S	14	F1	5-25
ENI	14	F3	5-25
ENQ	14	F1	5-25
ENQ, S	14	F1	5-25
EUA	55	F1	5-36
EXS	77	F5	5-83
FAD, I	60	F3	5-65
FDV, I	63	F3	5-66

TABLE D-1. INSTRUCTION FORMATS (Cont'd)

Mnemonic Code	Basic Octal Code	Instruction Format	Page No.
FMU, I	62	F3	5-66
FRMT	64	F18	5-147
FSB, I	61	F3	5-65
HLT	00	F1	5-24
IAI	53	F6	5-33
IAPR	77	F4	5-113
IJD	02	F3	5-44
IJI	02	F3	5-43
INA	15	F1	5-27
INA, S	15	F1	5-27
INAC, INT	73	F14	5-106
INAW, INT	74	F14	5-107
INCL	77	F4	5-89
INI	15	F3	5-27
INPC, INT, B, H, G	73	F15	5-98
INPW, INT, B, N, G	74	F15	5-100
INQ	15	F1	5-27
INQ, S	15	F1	5-27
INS	77	F5	5-85
INTS	77	F5	5-84
IOCL	77	F4	5-94
ISA	77	F7	5-37
ISD	10	F3	5-31
ISE	04	F3	5-28
ISG	05	F3	5-30
ISI	10	F3	5-31
JAA	77	F4	5-40
JMP, HI A	70	F1	5-42
JMP, LOW A	70	F1	5-42
JMP, ZRO A	70	F1	5-42
LACH	22	F2	5-49
LBR	70	F1	5-155

TABLE D-1. INSTRUCTION FORMATS (Cont'd)

Mnemonic Code	Basic Octal Code	Instruction Format	Page No.
LCA, I	24	F3	5-50
LCAQ, I	26	F3	5-51
LDA, I	20	F3	5-49
LDAQ, I	25	F3	5-50
LDI, I	54	F3	5-52
LDL, I	27	F3	5-50
LDQ, I	21	F3	5-51
LPA, I	37	F3	5-73
LQCH	23	F2	5-52
MEQ	06	F1	5-75
MOVE, INT	72	F16	5-138
MTH	07	F1	5-76
MUA, I	50	F3	5-62
MUAQ, I	56	F3	5-63
MVBF	64	F18	5-142
MVE	64	F18	5-140
MVE, D	64	F19	5-141
MVZF	64	F18	5-143
MVZS	64	F18	5-144
MVZS, D	64	F19	5-145
OSA	77	F4	5-37
OTAC, INT	75	F14	5-109
OTAW, INT	76	F14	5-110
OUTC, INT, B, H	75	F15	5-102
OUTW, INT, B, N	76	F15	5-104
PAK	66	F18	5-121
PAUS	77	F11	5-87
PFA	77	F9	5-39
PRP	77	F11	5-88
QEL	55	F1	5-36
QSE	04	F1	5-29
QSE, S	04	F1	5-29

TABLE D-1. INSTRUCTION FORMATS (Cont'd)

Mnemonic Code	Basic Octal Code	Instruction Format	Page No.
QSG	05	F1	5-30
QSG, S	05	F1	5-30
RAD, I	34	F3	5-60
RIS	55	F1	5-112
ROS	55	F1	5-112
RTJ	00	F1	5-47
SACH	42	F2	5-54
SBA, I	31	F3	5-61
SBAQ, I	33	F3	5-61
SBCD	77	F4	5-91
SBJP	77	F4	5-112
SBM	67	F18	5-69
SBR	70	F1	5-155
SCA, I	36	F3	5-72
SCAN, LR, EQ, DC	65	F26	5-130
SCAN, LR, NE, DC	65	F26	5-132
SCAN, RL, EQ, DC	65	F26	5-134
SCAN, RL, NE, DC	65	F26	5-136
SCAN, LR, EQ	65	F26	5-129
SCAN, LR, NE	65	F26	5-131
SCAN, RL, EQ	65	F26	5-133
SCAN, RL, NE	65	F26	5-135
SCAQ	13	F3	5-59
SCHA, I	46	F3	5-56
SCIM	77	F4	5-90
SDL	77	F7	5-113
SEL	77	F12	5-96
SFPF	77	F4	5-91
SHA	12	F3	5-57
SHAQ	13	F3	5-59

TABLE D-1. INSTRUCTION FORMATS (Cont'd)

Mnemonic Code	Basic Octal Code	Instruction Format	Page No.
SHQ	12	F3	5-59
SJ1	00	F1	5-41
SJ2	00	F1	5-41
SJ3	00	F1	5-41
SJ4	00	F1	5-41
SJ5	00	F1	5-41
SJ6	00	F1	5-41
SLS	77	F4	5-24
SQCH	43	F2	5-55
SRCE, INT	71	F13	5-125
SRCN, INT	71	F13	5-127
SSA, I	35	F3	5-72
SSH	10	F1	5-57
SSIM	77	F4	5-90
STA, I	40	F3	5-53
STAQ, I	45	F3	5-54
STI, I	47	F3	5-56
STQ, I	41	F3	5-55
SWA, I	44	F3	5-56
TAI	53	F6	5-33
TAM	53	F10	5-34
TIA	53	F6	5-33
TIM	53	F10	5-35
TMA	53	F10	5-34
TMAV	77	F4	5-81
TMI	53	F10	5-35
TMQ	53	F10	5-34
TQM	53	F10	5-34
TST	67	F17	5-82
TSTN	67	F17	5-82.0
UCS	77	F4	5-24
UJP, I	01	F3	5-41

TABLE D-1. INSTRUCTION FORMATS (Cont'd)

Mnemonic Code	Basic Octal Code	Instruction Format	Page No.
UPAK	66	F18	5-122
XOA	16	F1	5-71
XOA, S	16	F1	5-71
XOI	16	F3	5-71
XOQ	16	F1	5-71
XOQ, S	16	F1	5-71
ZADM	67	F18	5-146

APPENDIX E

MULTIPROGRAMMING AND RELOCATION SUPPLEMENTARY INFORMATION

E. MULTIPROGRAMMING AND RELOCATION SUPPLEMENTARY INFORMATION

Multiprogramming in the 3300 Computer System enables the instructions of many programs to be sequentially executed by controlled time-sharing operations within a processor. With the Control Data Multiprogramming Modules, throughput is very high due to efficient use of hardware and optimum program scheduling. This feature is very desirable at installations where numerous jobs are run and computing time must be kept at a minimum. Systems equipped with the relocation feature can compute many programs on a time-shared basis or be switched into the non-Executive mode and process jobs according to control card job assignments.

EXECUTIVE MODE

A system equipped with relocation hardware and operating in the Executive Mode functions in either the Monitor State or the Program State.

Monitor State

The Monitor State is the initial operating state of a master cleared processor. The processor also reverts to this state if interrupted for any condition. All instructions may be executed in the Monitor State.

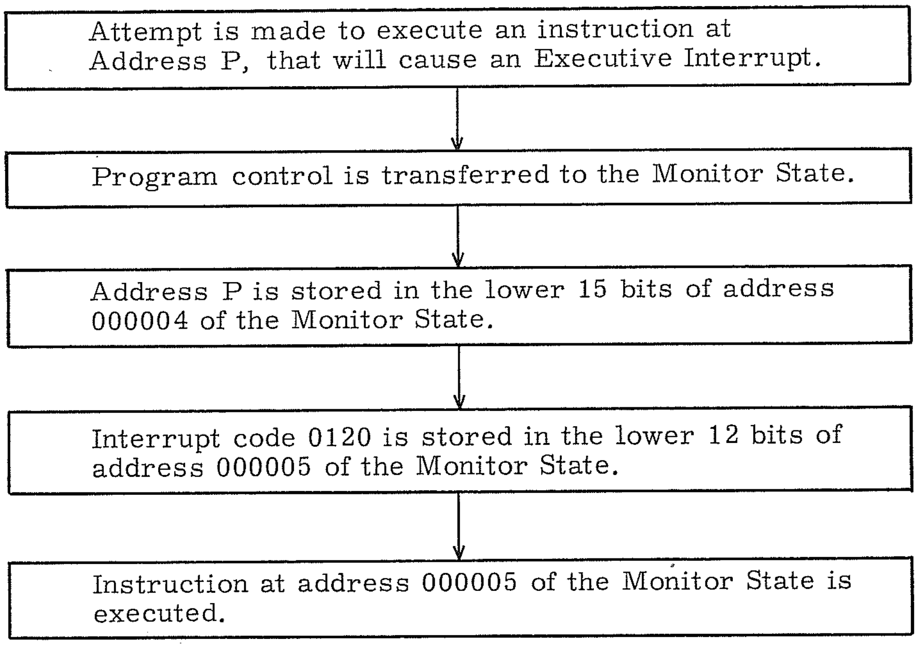
Program State

The Program State permits all but the following instructions to be executed:

1. A Halt instruction (00.0)
2. Any of the instructions with function codes in the 71-77 range including the UCS, except the SFPF (77.71) and SBCD (77.72) instructions.
3. An inter-register transfer instruction that attempts to alter registers 00 through 37 of the register file.

If an attempt to execute one of these instructions occurs, an Executive interrupt is generated and operating control is transferred back to the Monitor State. The Executive interrupt is not masked and the interrupt system need not be enabled to recognize the interrupt when it occurs. Upon recognition, the Executive interrupt transfers program control to the Monitor State. The instruction that caused the interrupt is not executed. The following flow chart describes the sequence of events involved when an Executive Interrupt occurs.

EXECUTIVE INTERRUPT SEQUENCE



Attempt is made to execute an instruction at Address P, that will cause an Executive Interrupt.

Program control is transferred to the Monitor State.

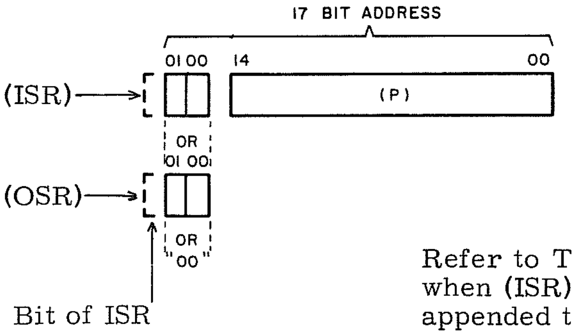
Address P is stored in the lower 15 bits of address 000004 of the Monitor State.

Interrupt code 0120 is stored in the lower 12 bits of address 000005 of the Monitor State.

Instruction at address 000005 of the Monitor State is executed.

MULTIPROGRAMMING AND RELOCATION

If the 3311 Multiprogramming option is not present in a 3300 system, the maximum number of MCS words is 131,072. The actual address referenced is as follows:



Upper Bit of ISR or OSR is ignored

Refer to Table E-1 for conditions when (ISR), (OSR), or zero is appended to address P.

If the APF (77.64) or PFA (77.65) instructions are executed they become no-operation instructions when the 3311 is not present. The keyboard sweep and enter functions with the Page Index File are also disabled. All other operating conditions are the same whether or not the 3311 is in the system.

A 3300 CPU can access up to 262,144 words of core storage when the 3311 Multiprogramming option and appropriate storage modules are present in the system. This is accomplished by augmenting the basic 15-bit address P with a 3-bit state number. The state number, along with a portion of the 15-bit address, becomes the direction path into a relocation path. From the Page Index File the correct page address is obtained for actual memory addressing.

Page Structure

Each page of memory is assigned 2,048 absolute memory locations. A fully expanded system contains 128 of these pages. Individual pages may be subdivided into four partial pages. A 1/4 page consists of 512 address locations. Programs may be allocated full pages, 3/4 page, 1/2 page or 1/4 page of memory.

To facilitiate addressing with the paging scheme, a word organized core matrix is used. This core matrix, called the Page Index File, is referenced by a program during a memory reference to obtain the physical page address or partial page address and provide memory protection.

Address Relocation

Figure E-1 illustrates address bits at various stages of the relocation process. Those portions of the diagram accompanied by circled numbers are further described in the following numbered paragraphs.

① Program Address and Program Address Group

Any program executed by a 3300 is processed within the confines of a 15-bit program address structure. These 15 bits define the program or operand address related to the routine or subroutine being processed at a given instant. Figures E-2 and E-3 illustrate the significance of these bits in the instruction words for both word addressing and character addressing.

The 15 bits used in word addressing define an absolute address assignment ranging from 00000 to 77777_8. Any program or group of programs within this range of addresses which can be compiled and loaded without conflicting addresses can be considered part of a program address group. Figure E-4 is illustrative of a program address group consisting of five non-conflicting programs.

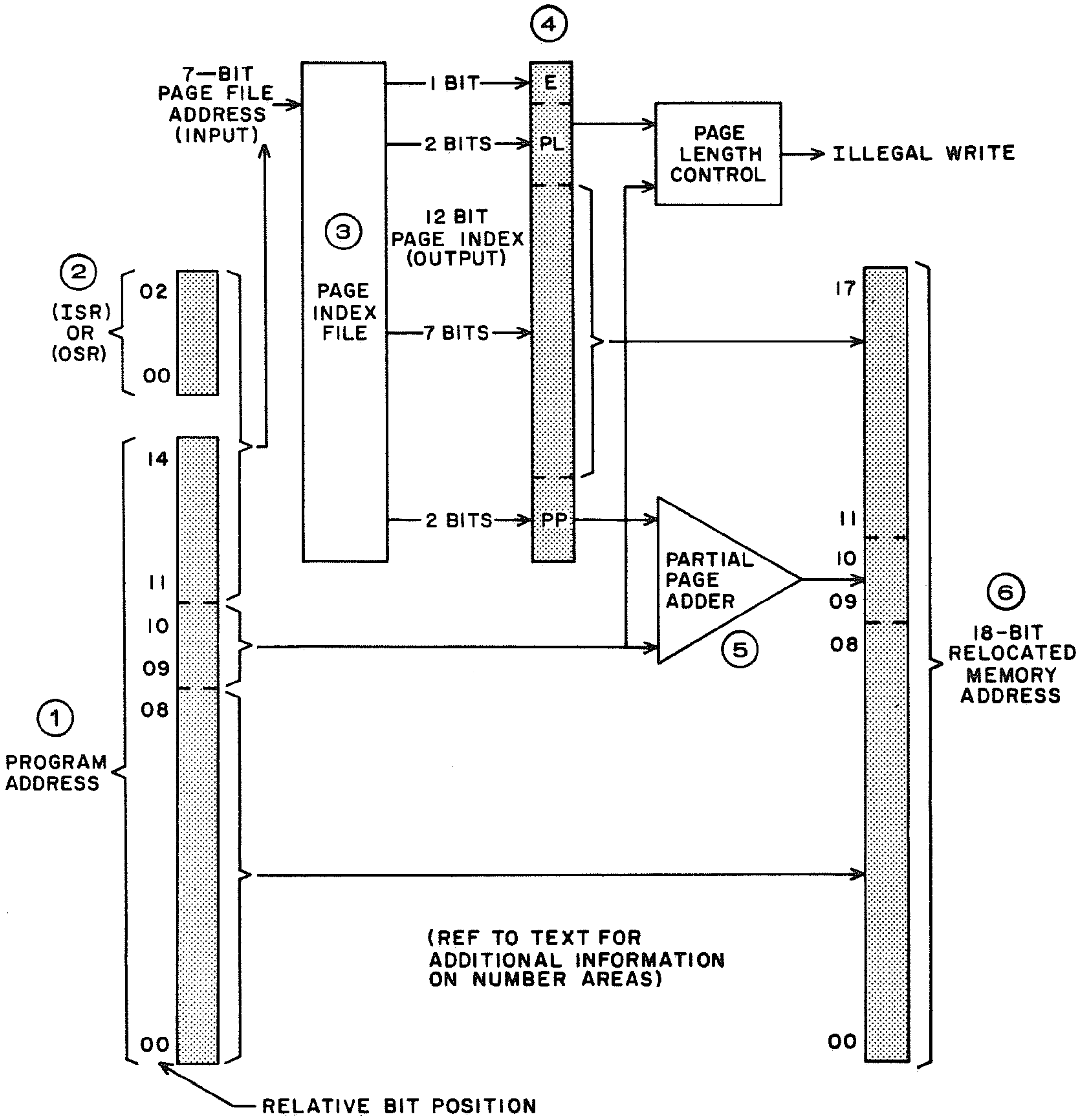


Figure E-1. Address Relocation Process

A program address group may be considered apart from the physical memory structure since it is a group of sequentially numbered addresses representing one or more programs within 32, 768 words of storage and not a discrete physical device. Many program address groups may be contained in storage; however, eight such groups are used in the 3300 to best optimize the memory system.

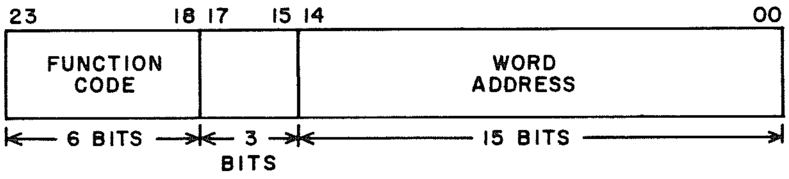


Figure E-2. Word Addressing

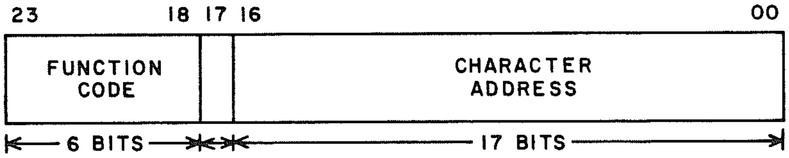


Figure E-3. Character Addressing

② Instruction State Register (ISR) and Operand State Register (OSR)

The ISR and OSR define the specific program address group currently being accessed by a processor. The program address group being referenced for instructions and operands can assume any one of eight discrete values by modifying the contents of these single digit registers. By transferring dissimilar numbers into these registers, instructions and operands may reference different program address groups.

The contents of these registers can only be changed by the Executive routine in the Monitor State.

The program address group that is currently valid for memory references is selected by the contents of the ISR or OSR. Table E-1 describes the selecting conditions.

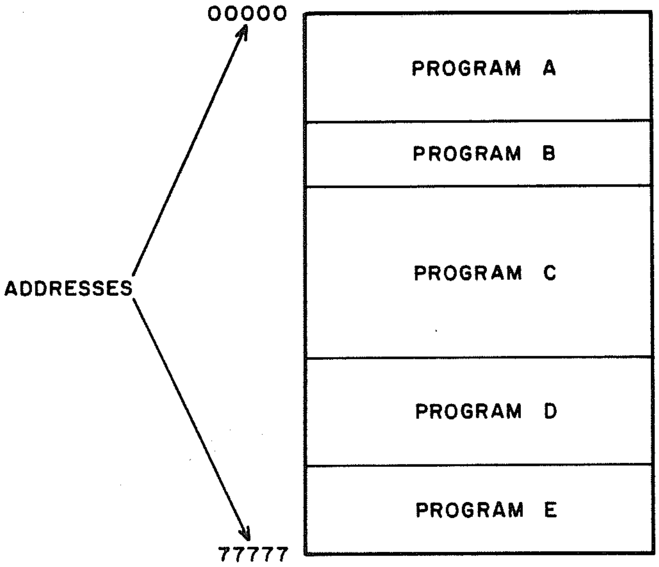


Figure E-4. Program Address Group

TABLE E-1. INSTRUCTION AND OPERAND REFERENCING

Operational State of the Processor	Instructions Referenced With:	Operands Referenced With:
Initial Monitor State	Zero	Zero
Monitor State and 55.4 (relocate to operand state) instruction executed	Zero	Contents of OSR
Transition from Monitor State to Program State	Contents of ISR	Contents of ISR*
Program State and 55.4 (relocate to operand state) instruction executed	Contents of ISR	Contents of OSR
Program State and 55.0 (relocate to instruction state) instruction executed	Contents of ISR	Contents of ISR
Any interrupt condition to Monitor State	Zero	Zero

*Transition from Monitor State to Program State does not change the operand address mode.

③ Page Index File

The Page Index File is functionally divided into eight distinct reference areas. One area is associated with each of eight possible numbers appearing in the ISR and OSR. Because of this direct relationship, each of the eight program address groups is permanently assigned a reference area in the Page Index File.

Each of the eight reference areas within the Page Index File consists of sixteen 12-bit Page Index Registers. This provides each of the program address groups exclusive use of 16 of these registers. By using the upper 4 bits of the program address for direction to the respective Page Index Registers, a direct and sequential relationship is established between the addresses in a program address group and a specific set of 16 Page Index registers. The Page Index File is actually constructed of 64 24-bit Page Index registers with dual 12-bit indexes. Only one of the 12-bit indexes is used during any specific reference.

Figure E-6 depicts the page indexes within the Page Index File and Figure E-7 illustrates the relationship between program address groups, Page Index File, and a fully expanded core memory.

Bit 11 of the original 15-bit address determines which of the two page indexes at the Page File location will be used. Figure E-5 shows a specific page index being referenced.

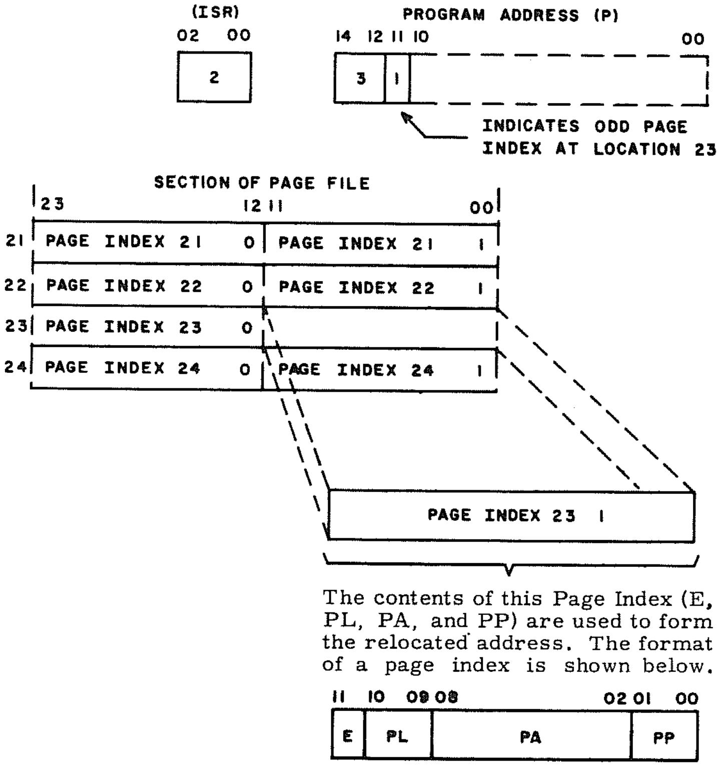
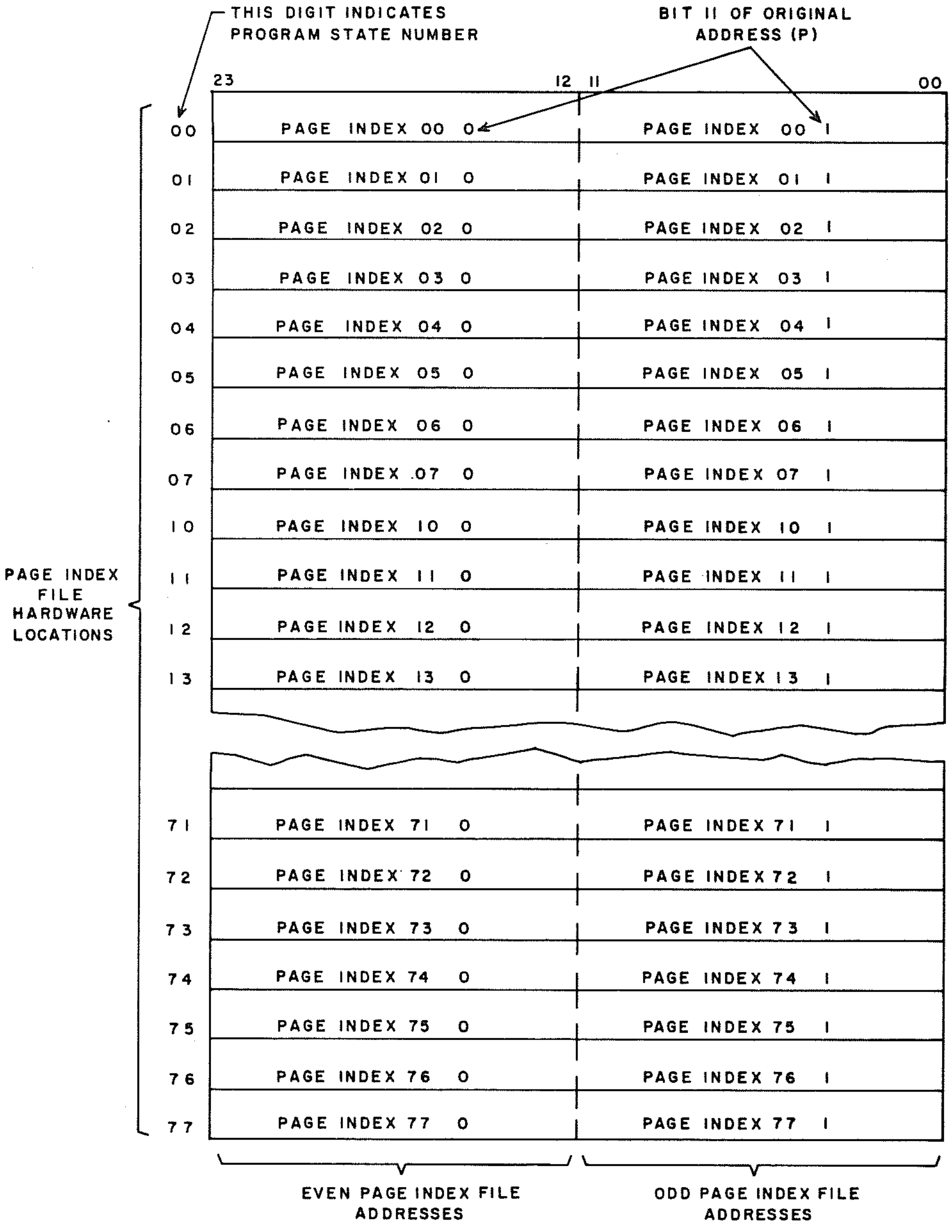

(ISR)
02 00
2
PROGRAM ADDRESS (P)
14 12 11 10 00
3
1
INDICATES ODD PAGE INDEX AT LOCATION 23
SECTION OF PAGE FILE
23 12 11 00
21 PAGE INDEX 21 0 PAGE INDEX 21 1
22 PAGE INDEX 22 0 PAGE INDEX 22 1
23 PAGE INDEX 23 0
24 PAGE INDEX 24 0 PAGE INDEX 24 1
PAGE INDEX 23 1
The contents of this Page Index (E, PL, PA, and PP) are used to form the relocated address. The format of a page index is shown below.
11 10 09 08 02 01 00
E
PL
PA
PP

Figure E-5. Example of Page Index Referencing


THIS DIGIT INDICATES PROGRAM STATE NUMBER
BIT 11 OF ORIGINAL ADDRESS (P)
23
12
11
00
PAGE INDEX FILE HARDWARE LOCATIONS
EVEN PAGE INDEX FILE ADDRESSES
ODD PAGE INDEX FILE ADDRESSES

	23–12	11–00
00	PAGE INDEX 00 0	PAGE INDEX 00 1
01	PAGE INDEX 01 0	PAGE INDEX 01 1
02	PAGE INDEX 02 0	PAGE INDEX 02 1
03	PAGE INDEX 03 0	PAGE INDEX 03 1
04	PAGE INDEX 04 0	PAGE INDEX 04 1
05	PAGE INDEX 05 0	PAGE INDEX 05 1
06	PAGE INDEX 06 0	PAGE INDEX 06 1
07	PAGE INDEX .07 0	PAGE INDEX 07 1
10	PAGE INDEX 10 0	PAGE INDEX 10 1
11	PAGE INDEX 11 0	PAGE INDEX 11 1
12	PAGE INDEX 12 0	PAGE INDEX 12 1
13	PAGE INDEX 13 0	PAGE INDEX 13 1
71	PAGE INDEX 71 0	PAGE INDEX 71 1
72	PAGE INDEX 72 0	PAGE INDEX 72 1
73	PAGE INDEX 73 0	PAGE INDEX 73 1
74	PAGE INDEX 74 0	PAGE INDEX 74 1
75	PAGE INDEX 75 0	PAGE INDEX 75 1
76	PAGE INDEX 76 0	PAGE INDEX 76 1
77	PAGE INDEX 77 0	PAGE INDEX 77 1

Figure E-6. Page Index File Address and Hardware Structure

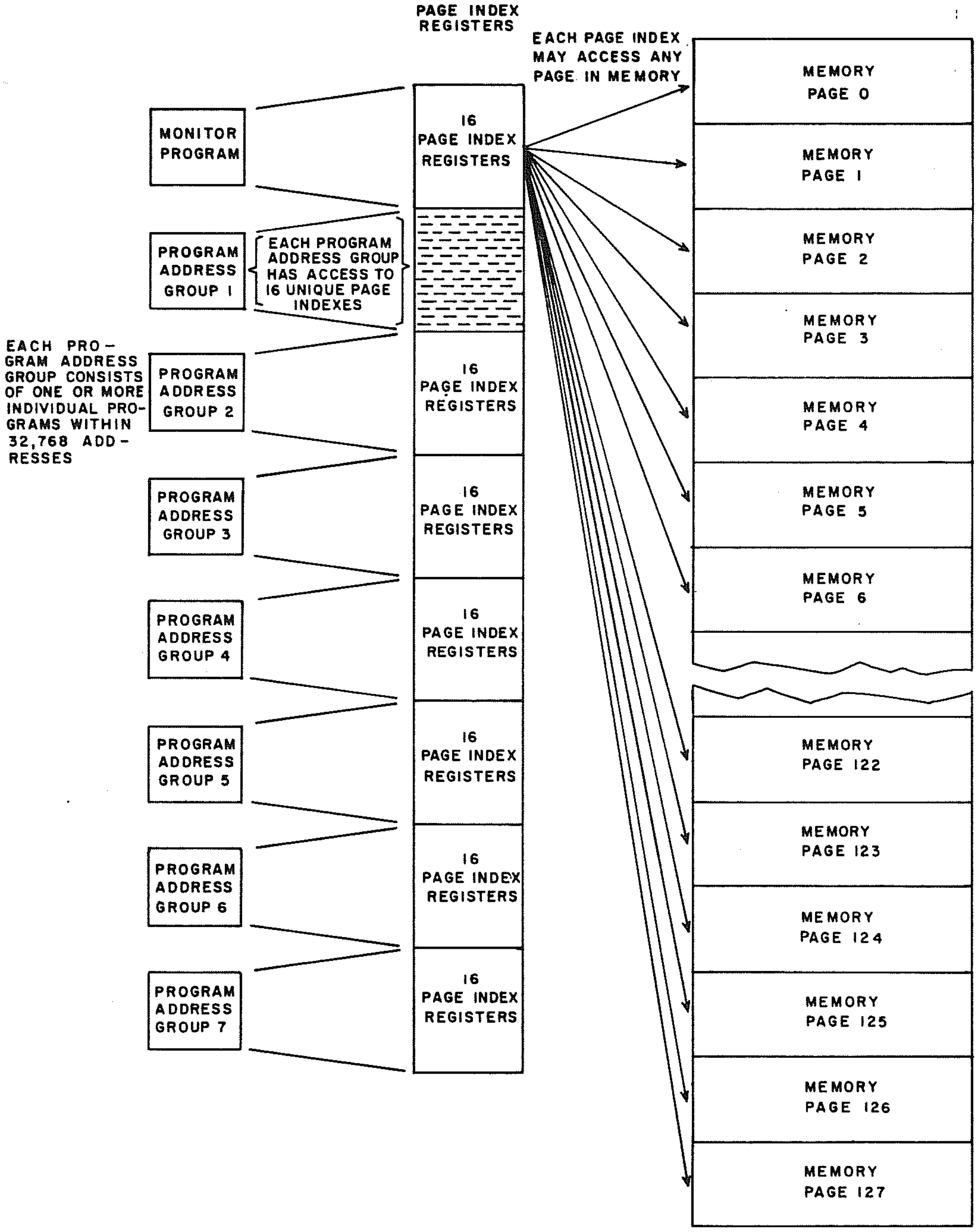
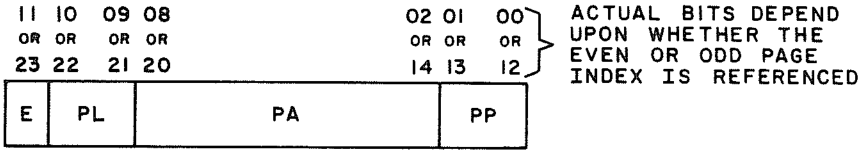


Figure E-7. Relocation System Illustrating Memory Protection with Fully Expanded Memory (262K)

④ Page Index

Each page index has the same basic format. The significance of each designator during the relocation process is described below. Figure E-8 shows the format for a page index while Figure E-9 shows a view of the display panel on the relocation chassis.


11 OR 23
10 OR 22
09 OR 21
08 OR 20
02 OR 14
01 OR 13
00 OR 12
ACTUAL BITS DEPEND UPON WHETHER THE EVEN OR ODD PAGE INDEX IS REFERENCED
E
PL
PA
PP

E = EXCLUSION BIT (1 BIT)
PL = PAGE LENGTH DESIGNATOR (2 BITS)
PA = PAGE ADDRESS DESIGNATOR (7 BITS)
PP = PARTIAL PAGE DESIGNATOR (2 BITS)

Figure E-8. Page Index Format

E - Exclusion bit

This designator may have one of three meanings:

1. If E = "0", the quantity expressed by PA defines a page* where either reading or writing is permitted.
2. If E = "1", and PL, PA or PP is a quantity other than zero, PA defines a page * where only reading is permitted. If a write is attempted, an Illegal Write interrupt is generated.
3. If E = "1" and PL, PA or PP are all equal to zero, an unaddressable page is defined and an Illegal Write interrupt is generated by the Page Index File.

* Refer to descriptions of PL and PP designators for page restrictions.

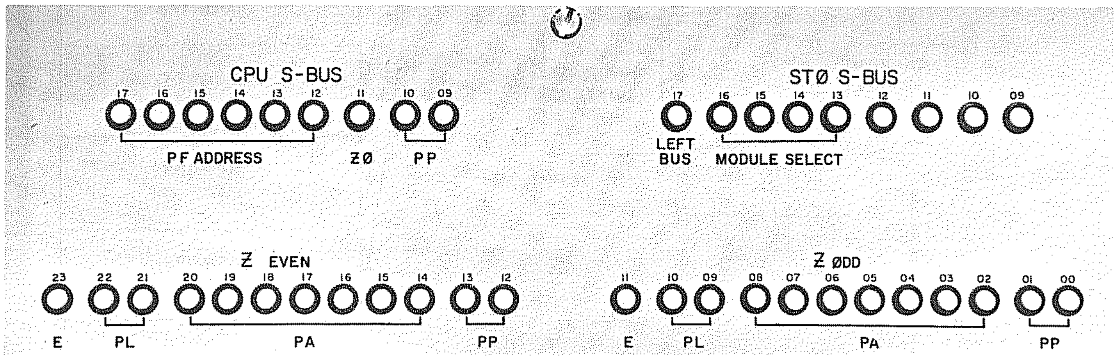


Figure E-9. Relocation Chassis Display Panel

PA - Seven bits are used to define the actual memory module being referenced. As stated earlier in this manual, there may be 128 segmented pages in a 3300 system with 262,144 words of core storage. Each page has a unique page address and addresses 000 through 177_8 define all of the possible pages.

A 3300 system with a fully expanded storage network has two address busses. Each bus has access to 131,072 words of the total 262,144 storage words. The uppermost bit of PA (bit 17 in the relocated address) determines which bus (right or left) is selected. This bit will be a "1" when the left bus is used and a "0" when the right bus is selected. Figure E-10 depicts the bus address system.

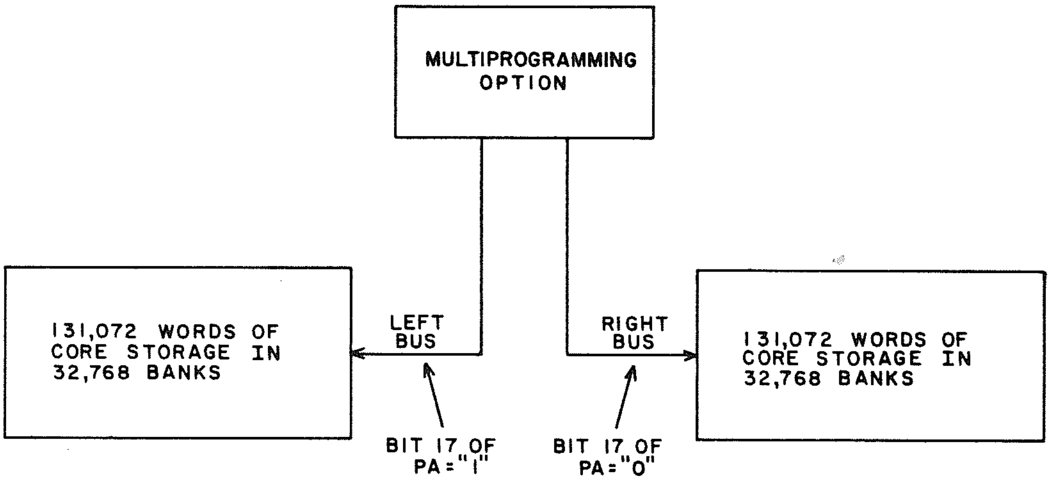

MULTIPROGRAMMING OPTION
131,072 WORDS OF CORE STORAGE IN 32,768 BANKS
LEFT BUS
RIGHT BUS
131,072 WORDS OF CORE STORAGE IN 32,768 BANKS
BIT 17 OF PA="1"
BIT 17 OF PA="0"

Figure E-10. Storage Address Buses

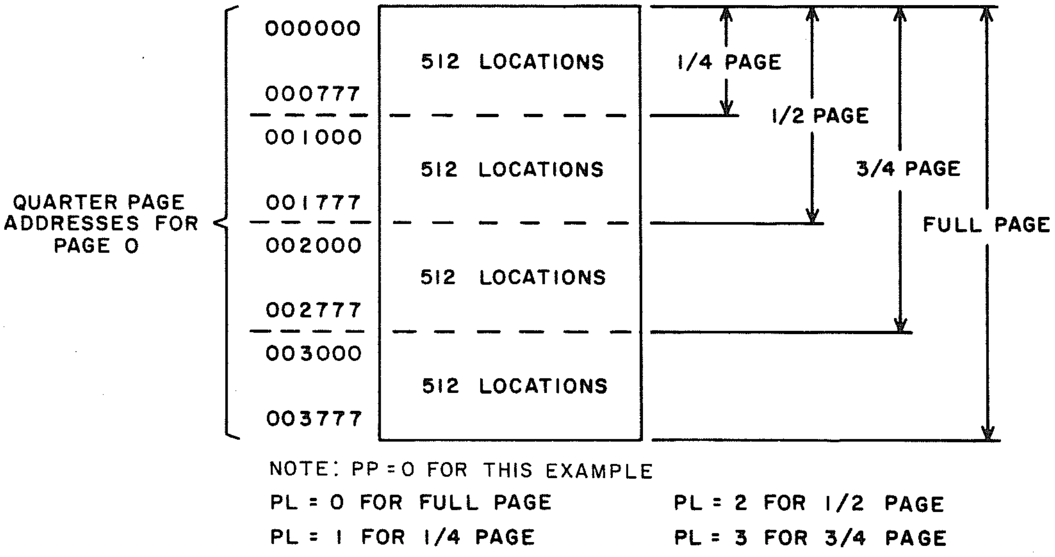


Figure E-11. Page Length Subdivisions

PL - Each page has 2,048 memory locations and is subdivided into quarters of 512 locations each. The PL designator defines how many quarters of a page can be referenced (beginning with the starting quarter specified by PP). A program is assigned the number of quarter pages it needs to reside in memory. Figure E-11 illustrates the quarter sections of a page and the significance of the PL bits.

PP - The Partial Page designator is the address of the physical quarter page that will serve as the starting point of the page. Example A (Figure E-12) shows the quarter page referenced for each of the PP designators. The significance of the PP designator in selecting the respective quarter page for addressing is described below.

EXAMPLE A

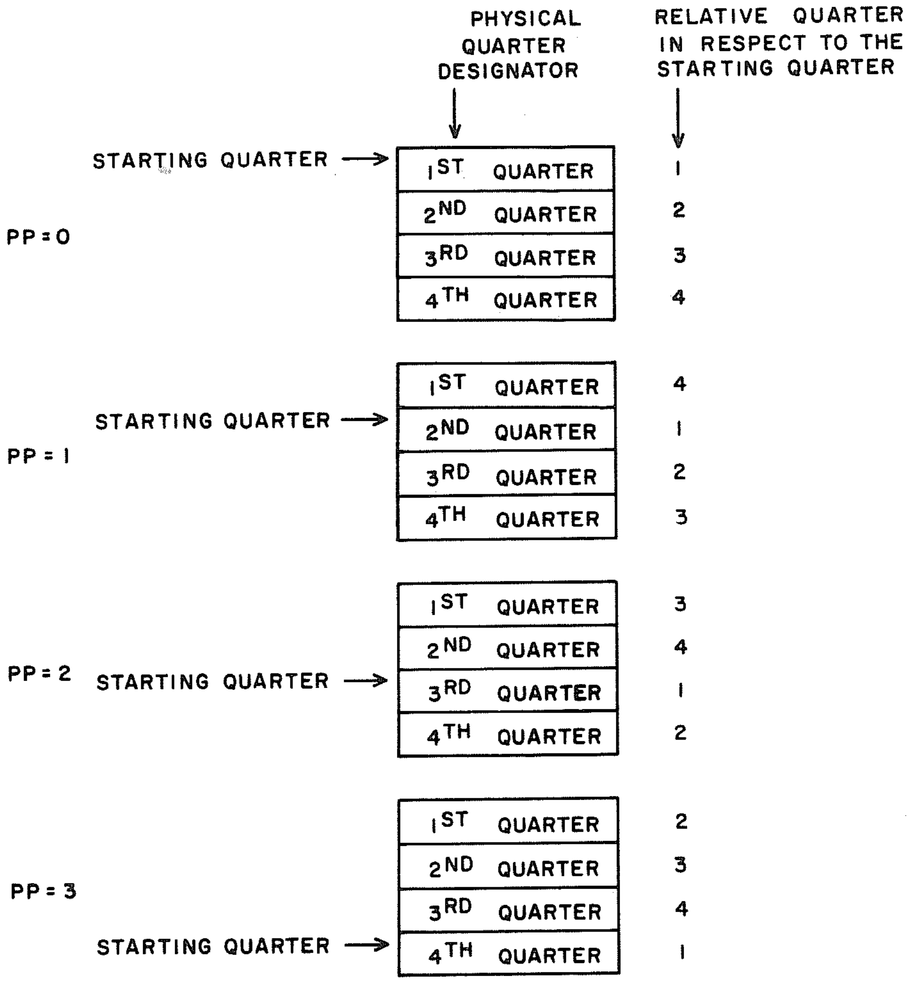


Figure E-12. Quarter Page in Relation to PP Designator

If PP = 0, the relative page begins in the 1st physical quarter

If PP = 1, the relative page begins in the 2nd physical quarter

If PP = 2, the relative page begins in the 3rd physical quarter

If PP = 3, the relative page begins in the 4th physical quarter

⑤ Partial Page Adder

A special adder is used to combine the PP designator from the page index with bits 9 and 10 of the original address. The partial sum indicates the address of the physical quarter in which referencing will begin. Example B and Figure E-13 show the actual quarter page in which addressing occurs for specific PL, PP, and bits 9 and 10 values.

EXAMPLE B

PL = 0

PP = 1

Bits 9 and 10 = 2

Analysis: A full page (PL = 0) is allocated, the relative page begins in the second physical quarter, and referencing begins in the fourth physical quarter, (physical quarter address 3).

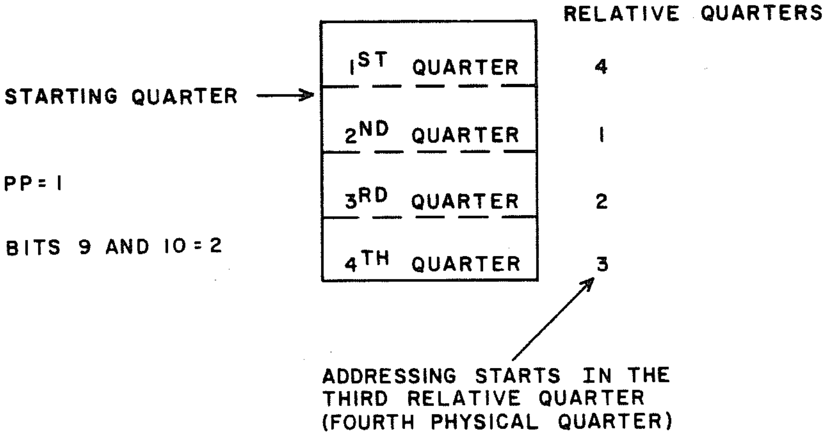


Figure E-13. Starting Quarters versus Relative Quarters

It should be noted that if bits 9 and 10 of the original address specify a quarter page equal to or greater than that of the PL designator when PL ≠ zero, an Illegal Write interrupt will occur. An example of this condition would be a 1/4 page allocated but bits 9 and 10 equal to 3, thus specifying an address in the fourth quarter.

This interrupt will not occur during Monitor State or I/O operations.

⑥ Relocated Address

The 18-bit relocated address defines the actual core storage location being referenced.

The PA portion of the page index fills the upper seven bits of this address (S) bus to use and select the appropriate storage module. Bits 9 and 10 receive the output of the adder previously described and indicate the physical quarter page being referenced. The lower nine bits are unaltered from the original address and comprise the remainder of the relocated address.

Page Zero Consideration

If page Index File address zero is referenced in either the Program or Monitor state, the PA and PP designators for this page index will always be zero. As a result of this condition, page zero, which encompasses addresses 000000 through 003777, can be accessed and used for storing the Auto Load and Auto Dump routines. The Auto Load routine is contained in addresses 003700 through 003737 and the Auto Dump routine is stored in addresses 003740 through 003777.

APPENDIX F

BUSINESS DATA PROCESSING SUPPLEMENTARY INFORMATION

F. BUSINESS DATA PROCESSING SUPPLEMENTARY INFORMATION

The performance of a 3300 computing system is further enhanced by the addition of a 3312 Business Data Processor (BDP) which eliminates the need for simulating subroutines. Business-oriented moves and edits, searches, code conversions and translations as well as BCD arithmetic operations are executed directly by the BDP. Since the BDP is designated as an integral part of the computer system and interfaces directly to the magnetic core storage modules, the instructions are executed with great speed. The BDP expands the basic instruction repertoire to cover instructions inherent to business data processing applications.

The logical organization of the BDP and its comprehensive instruction repertoire emphasizes its usefulness as a logical field processor. This organization permits truly efficient processing of highly structured data files. Some examples of applications which exploit these features are data collection, production control, inventory accounting, and financial and accounting systems.

CHARACTERISTICS OF BDP

The following are characteristics of the 3300 Business Data Processor:

- Character addressing and manipulation

  internal BCD 6-bit characters
  24-bit words
- Manipulation of data organized in variable or fixed length fields
- Memory to memory operations
- Field limits expressed by either programmer-defined delimiter or by length
- COBOL specification compatibility
- Error indications on arithmetic overflow and illegal characters
- Algebraic sign control

- Extremely fast and comprehensive data processing instruction set

  moves and edits
  searches
  ASCII/BCD code conversions and translations
  arithmetic functions

- System interrupt capability retained without loss of data

Business data processing instructions are listed in Section 5. A general description of each of the instruction categories is listed below:

MOVES AND EDITS

The following capabilities are features of this instruction category:

- Ability to transfer variable length data fields from one area of storage
- Both fields may specify any 6-bit character location in storage as the beginning address
- Both fields may be independently indexed
- Up to 4095 characters may be processed
- Operations may be terminated by specifying lengths of fields or by encountering delimiting characters; field lengths may differ
- Data moved from a source field to a receiving field may be manipulated and/or modified as follows:
  - ► Single character or block of characters transferred without modification
  - ► Move with blanks inserted in any remaining character positions in the receiving field
  - ► Move with zeros inserted in any remaining character positions in the receiving field
  - ► Move with leading zeros replaced with blanks and zone (sign) bits stripped during the transfer
  - ► Move with edit functions performed: insertion of commas, decimal point with suppression of leading zeros, or complete formatted edit with insertion of character set as defined in DOD COBOL-61 Extended specification

Instructions in this group are particularly useful in data processing applications involving character manipulation, formatting for printing of integer quantities, point alignment problems, etc. Editing functions are accomplished by hardware rather than a complex subroutine, resulting in extremely fast processing times.

SEARCHES

The following capabilities are features of this category of instructions:

- Any 6-bit character location in storage may be specified as the location of the first character to be searched
- Up to 4095 characters may be examined
- Indexing may be accomplished on the search field
- Search key (character) specified by programmer and contained in instruction word
- Search may be terminated by:
  - Locating object character
  - Completing specified number of searches without locating object character
  - Encountering delimiter character without locating object character
- At conclusion of search operation, an index register holds number of characters searched to aid in determining location of character meeting search condition; this information placed in Central Processor Index register
- Program control at search termination branches to either of two points, depending on result of search
- Searches may be of the following types:
  - Search successive character locations (either left to right or right to left) in a field for an object character equal to the search key
  - Search successive character locations (either left to right or right to left) in a field for an object character unequal to the search key
  - Search successive character locations (from left to right) in a field for an object character equal to the search key and jump; jump is to normal termination point plus the number of characters searched

CONVERSIONS AND TRANSLATIONS

Instructions in the category provide conversion and translation abilities to efficiently process data of varying formats. Translating codes prepares data for various operations preliminary to the actual data processing. Translations to and from the American Standard Code for Information Interchange (ASCII) code provide compatibility with other systems data handling schemes. An ASCII to BCD conversion table can be found in Appendix A.

The following conversions and translations may be effected with this category of instructions:

- Convert BCD to binary
- Convert binary to BCD
- Translate to ASCII
- Translate from ASCII
- Pack (convert numeric 6-bit digits into 4-bit BCD characters)
- Unpack (convert numeric 4-bit digits into 6-bit characters)

ARITHMETIC FUNCTIONS

The following capabilities are features of this category of instructions:

- Arithmetic performed on 6-bit BCD characters
- Both fields may specify any 6-bit character location in storage as the beginning address
- Both fields may be independently indexed
- Algebraic sign control
- Arithmetic overflow fault indicator provided
- Arithmetic operations from right to left
- Compare - comparisons of two fields for equal, unequal
  - ► Compare left to right
  - ► Delimiter or number of characters (up to 4095) specifies number of characters to be compared
  - ► High-low indication held in condition register for examination by Jump instructions
- Test instructions examine field for: greater than zero, zero, or less than zero. The result of the test sets a BCD condition register to +, 0, or -
- Jump instructions in the CPU may be used to examine arithmetic result flags in the BDP

Consult the individual business data processing instructions in Section 5 for detailed information pertaining to each type of instruction.

BCD CHARACTERS AND ALGEBRAIC SIGN POSITIONS

Six-bit BCD characters are used during most BDP operations. The following diagrams, example, and tables show the character placements and sign positions:

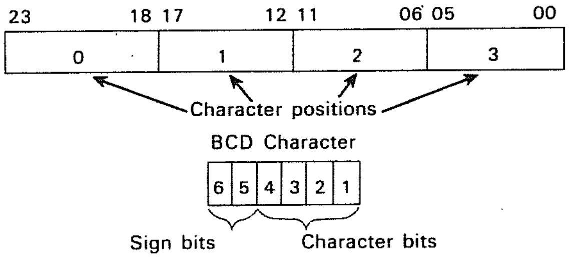


Figure F-1. BCD Word and Character Format

TABLE F-1. BCD SIGN BIT POSITIONS

Sign of BCD Field	Relative Bit Positions	
	6	5
+	0	0
+	0	1
-	1	0
+	1	1

TABLE F-2. DECIMAL/BCD CHARACTER FORMAT

Decimal Number	BCD Character Relative Bit Positions			
	4	3	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

EXAMPLE: Execute the MVZF instruction at P, P + 1, and P + 2.

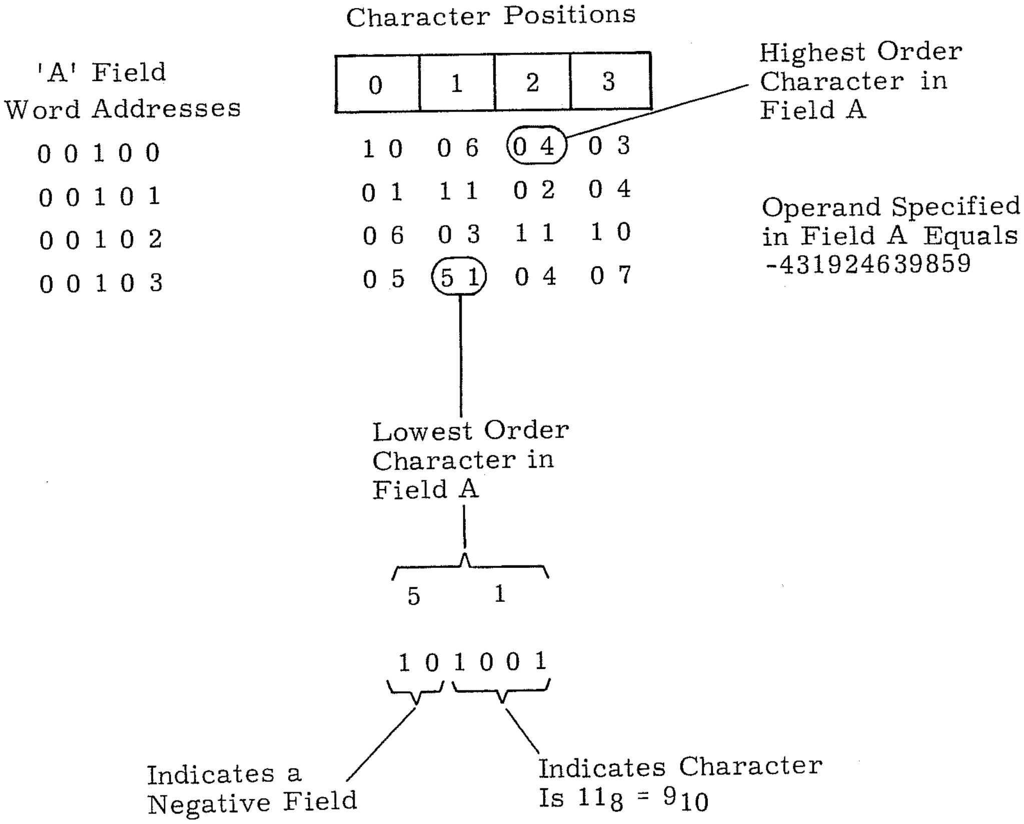
P = 64000202

P + 1 = 27003000 $(B^1) = 00200$

P + 2 = 00140017 $(B^2) = 01000$

Analysis:

1. The unmodified character address 'r' is 00202.
2. $B_r = 3$, requiring (B^1) be added to r. If $(B^1) = 00200$ then R = 00402 which equals word address 00100 character position 2. This is the true address of the highest order character in field A.
3. $B_s = 2$, requiring (B^2) be added to the unmodified character address 's', 03000. If $(B^2) = 01000$, then S = 04000.
4. The length of the A field is 14_8 characters and the alloted length of the C field is 17_8 characters. The last three characters of field C will be filled with zeros. The last character of field C (a zero) will also contain the sign of the field.



Character Positions

'A' Field Word Addresses	0	1	2	3
00100	10	06	04	03
00101	01	11	02	04
00102	06	03	11	10
00103	05	51	04	07

Highest Order Character in Field A
Operand Specified in Field A Equals -431924639859
Lowest Order Character in Field A
5 1
101001
Indicates a Negative Field
Indicates Character Is $11_8 = 9_{10}$

The operation proceeds as follows:

First Character Stored Is Highest Order Character At Word Address 01000, Character Position Zero.

'C' Field Word Addresses				
00777	03	01	00	02
01000	04	03	01	11
01001	02	04	06	03
01002	11	10	05	11
01003	00	00	40	07

Sign Is No Longer Stored In This Character As The Lowest Order Character Of Field C Is Now Character Position 2 Of Word Address 01003.

Lowest Order Character In Field C Contains the Sign of Field C (minus) and a Zero Character.

Operand Now Stored In Field C Equals -431924639859000

GLOSSARY

The definitions of terms in this glossary are general, and are oriented toward their application to the 3300 computer. The definitions should not be construed as absolute or applicable for all Control Data products.

A REGISTER - Principal arithmetic register; operates as a 24-bit additive accumulator (modulus 2^{24}-1).

ABSOLUTE ADDRESS - An address at a specific memory location.

ACCESS TIME - The time needed to perform a storage reference, either read or write. In effect, the access time of a computer is one storage reference cycle.

ACCUMULATOR - A register with provisions for the addition of another quantity to its content.

ADDER - A device capable of forming the sum of two or more quantities.

ADDRESS - A 15-bit operand which identifies a particular storage location; a 17-bit operand which identifies a particular character location in storage.

ADDRESS MODIFICATION - Normally the derivation of a storage address from the sum of the execution address and the contents of the specified index register.

AND FUNCTION - A logical function in Boolean algebra that is satisfied (has the value "1") only when all of its terms are "1's". For any other combination of values it is not satisfied and its value is "0".

ARGUMENT - An operand or parameter used by a program or an instruction.

ASCII CODE - American Standard Code for Information Interchange 8-bit character code (eighth bit is actually unassigned).

ASSEMBLER - A program which translates statements to machine language. Normally, one source language statement results in the generation of one line of object code.

B^1, B^2, B^3 REGISTERS - Index registers used primarily for address modification and/or counting.

BASE - A quantity which defines some system of representing numbers by positional notation; radix.

BDP - (1) Business Data Processor (3312); provides the necessary hardware to execute business oriented instructions. Contains its own translation and control logic but must be used with CPU.

(2) Business Data Processing; processing business oriented data.

BINARY CODED DECIMAL (BCD) - A form of decimal notation where decimal digits are represented by a binary code.

BIT - Binary digit, either "1" or "0".

BLOCK - A sequential group of storage words or characters in storage.

BOOTSTRAP - Any short program which facilitates loading of the appropriate system executive.

BREAKPOINT - A point in a routine at which the computer may be stopped by manual switches for a visual check of progress.

BUFFER - Any area that is used to hold data temporarily for input or output, normally storage.

BYTE - A portion of a computer word, usually 6 or 12 bits.

CAPACITY - The upper and lower limits of the numbers which may be processed in a register or the quantity of information which may be stored in a storage unit. If the capacity of a register is exceeded, an overflow is generated.

CHANNEL - An input/output (I/O) transmission path that connects the computer to an external equipment; 3306 or 3307.

CHARACTER - A group of bits which represents a digit, letter, or symbol from the typewriter.

CLEAR - An operation that removes a quantity from a register by placing every stage of the register in the "0" state. The initial contents of the register are destroyed by the Clear operation.

COMMAND - A control signal; also used synonymously with Instruction.

COMPILER - A program with the capability to generate more than one line of machine code (instruction or data word) from one source language statement.

COMPLEMENT - Noun: See One's Complement or Two's Complement. Verb: A command which produces the one's complement of a given quantity.

CONTENT - The quantity or word held in a register or storage location.

CORE - A ferromagnetic toroid used as the bi-stable device for storing a bit in a memory plane.

COUNTER - A register or storage location, the contents of which may be incremented or decremented.

CPU - Central Processing Unit; controls all sequential operations within the computer. See Main Control.

D - Delimiting indicator (refer to Delimiting).

DELIMITING - During a BDP character operation, character delimiting may sometimes be used where the operation is terminated if during the course of the operation a character is recognized as equal (or unequal in some instructions) to a fixed comparison (or delimiting) character.

DOUBLE PRECISION - Providing greater precision in the results of arithmetic operations by appending 24 additional bits of lesser significance to the initial operands.

ENTER - The operation where the current contents of a register or storage location are replaced by some defined operand.

EQUALIZE - Adjusting the operand of the algebraically smaller exponent to equal the larger prior to adding or subtracting the floating point coefficients.

EXCLUSIVE OR - A logical function in Boolean algebra that is satisfied (has the value "1") when any of its terms are "1". It is not satisfied when all its terms are "1" or when all its terms are "0".

EXECUTION ADDRESS - The lower 15 or 17 bits of a 24-bit instruction. Most often used to specify the storage address of an operand. Sometimes used as the operand.

EXECUTIVE MODE - An operating mode in which address relocation may occur. An efficient operating mode consisting of two possible states: Monitor State and Program State.

EXIT - Initiation of a second control sequence by the first, occurring when the first is near completion; the circuit involved in exiting.

F REGISTER - Program Control register. Holds a program step while the single 24-bit instruction contained in it is executed.

FAULT - Operational difficulty which lights an indicator or for which interrupt may be selected.

FILE MANAGER - A software system operating in conjunction with MSIO and providing a central repository for all data accruing in a particular data center, i.e., a system which creates a file for the user.

FIXED POINT - A notation or system of arithmetic in which all numeric quantities are expressed by a predetermined number of digits with the binary point implicitly located at some predetermined position; contrasted with floating point.

FLIP-FLOP (FF) - A bi-stable storage device. A "1" input to the set side puts the FF in the "1" state; a "1" input to the clear side puts the FF in the "0" state. The FF remains in a state indicative of its last "1" input. A stage of a register consists of a FF.

FLOATING POINT - A means of expressing a number, X, by a pair of numbers, Y and Z, such that $X = Yn^z$. Z is an integer called the exponent or characteristic; n is a base, usually 2 or 10; and Y is called the fraction or mantissa.

FUNCTION CODE - See Operation Code.

INCREASE - The increase operation adds a quantity to the contents of the specified register.

INDEX DESIGNATOR - A 2-bit quantity in an instruction; usually specifies an index register whose contents are to be added to the execution address; sometimes specifies the conditions for executing the instruction.

INDIRECT ADDRESSING - A method of address modification whereby the lower 18 bits of the specified address become the new execution address and index designator.

INSTRUCTION - A 24- or 48-bit quantity consisting of an operation code and several other designators.

INTEGRATED REGISTER FILE - The upper 64_{10} locations of core storage; reserved for special operations with Block Control.

INTERRUPT - A signal which results in transfer of control, following completion of the current instruction cycle, to a fixed storage location.

INTERRUPT REGISTER - A 24-bit register whose individual bits are set to "1" by the occurrence of specific interrupt conditions, either internal or external.

INTERRUPT MASK REGISTER - A 24-bit register whose individual bits match those of the Interrupt register. Setting bits of the Interrupt Mask register to "1's" is one of the conditions for selecting interrupt.

INVERTER - A circuit which provides as an output a signal that is opposite to its input. An inverter output is "1" only if all the separate OR inputs are "0".

ISR - Instruction State Register; 3-bit register defining the program address group being referenced for instructions.

JUMP - An instruction which alters the normal sequence control of the computer and, conditionally or unconditionally, specifies the location of the next instruction.

LIBRARY - Any collection of programs (routines) and/or subprograms (subroutines).

LOAD - The Load operation is composed of two steps: a) the register is cleared, and b) the contents of storage location M are copied into the cleared register.

LOCATION - A storage position holding one computer word, usually designated by a specific address.

LOGICAL PRODUCT - In Boolean algebra, the AND function of several terms. The product is "1" only when all the terms are "1"; otherwise it is "0". Sometimes referred to as the result of bit-by-bit multiplication.

LOGICAL SUM - In Boolean algebra, the OR function of several terms. The sum is "1" when any or all of the terms are "1"; it is "0" only when all are "0".

LOOP - Repetition of a group of instructions in a routine.

MACRO CODE - A method of defining a subroutine which can be generated and/or inserted by the assembler.

MAIN CONTROL - The sequence of events within the CPU controlling the various operations of program execution; synonomous with Program Control.

MASK - In the formation of the logical product of two quantities, one quantity may mask the other; i.e., determine what part of the other quantity is to be considered. If the mask is "0", that part of the other quantity is unused; if the mask is "1", the other quantity is used.

MASTER - Multiple Access, Shared Time, Executive Routine; an advanced time-sharing operating system for 3300 and 3500 computers equipped with the 3311 multiprogramming option.

MASTER CLEAR - A general command produced by pressing one of three switches; a) Internal Master Clear - clears all operational registers and control FFs in the processor; b) External Master Clear - clears all external equipments and the communication channels; c) Master Clear - a keyboard switch that performs both an Internal and External clear.

MCS - Magnetic Core Storage; see CORE.

MNEMONIC CODE - A three-or four-letter code which represents the function or purpose of an instruction. Also called alphabetic code.

MODULUS - An integer which describes certain arithmetic characteristics of registers, especially counters and accumulators, within a digital computer. The modulus of a device is defined by r^n for an open-ended device and r^n-1 for a closed (end-around) device, where r is the base of the number system used and n is the number of digit positions (stages) in the device. Generally, devices with modulus r^n use two's complement arithmetic; devices with modulus r^n-1 use one's complement.

MONITOR STATE - An operating state under Executive mode in which all 3300 instructions may be executed. If the 3311 Multiprogramming option is in a system, address relocation is possible.

MSIO - Mass Storage Input/Output; a basic file oriented I/O program for operating with mass storage devices and magnetic tape units.

MULTI-PROCESSING - Simultaneous instruction processing; multi-processing is accomplished in the 3300 by using an additional CPU.

MULTIPROGRAMMING - Alternately servicing instructions from two or more programs as opposed to completing one job at a time. Multiprogramming utilizes the time-sharing capabilities of the computer under the guidance of an executive monitor.

NON-EXECUTIVE MODE - An operating mode in which instructions are sequentially executed and which permits a 3300 to perform identically to a 3200.

NO-OP - No-Operation.

NO-OPERATION - Usually an undefined octal code that produces no useful function. Some 3300 instructions are No-Operation (NO-OP) instructions if execution is attempted in non-Executive mode.

NORMALIZE - To adjust the exponent and mantissa of a floating point result so that the mantissa lies in the prescribed standard (normal) range.

NORMAL JUMP - An instruction that jumps from one sequence of instructions to a second and makes no preparation for returning to the first sequence. Also referred to as an unconditional jump.

NUMERIC CODING - A system of abbreviation in which all information is reduced to numerical quantities. Also called absolute or machine language coding.

OBJECT PROGRAM - The machine language version of the source program.

ONE'S COMPLEMENT - With reference to a binary number, that number which results from subtracting each bit of a given number from "1". The one's complement of a number is formed by complementing each bit of it individually, that is, changing a "1" to "0" and a "0" to a "1". A negative number is expressed by the one's complement of the corresponding positive number.

ON-LINE OPERATION - A type of system application in which the input or output data to or from the system is fed directly from or to the external equipment.

OPERAND - Usually refers to the quantity specified by the execution address.

OPERATION CODE (Function Code) - A 6-bit quantity in an instruction specifying the operation to be performed.

OPERATIONAL REGISTERS - Registers which are displayed on the operator's section of the console.

OR FUNCTION - A logical function in Boolean algebra that is satisfied (has the value "1") when any of its terms are "1". It is not satisfied when all terms are "0". Often called the inclusive OR function.

OSR - Operand State Register; 3-bit register defining the program address group being referenced for operands.

OVERFLOW - The capacity of a register is exceeded.

P REGISTER - The Program Address Counter (P register) is a one's complement additive register (modulus 2^{15}-1) which defines the storage addresses containing the individual program steps.

PAGE - 2,048 absolute memory locations which may be subdivided into four partial pages. Storage allocation is made in whole number multiples of quarter pages.

PAGE INDEX FILE - A word-organized core matrix consisting of 12-bit page indexes. The contents of the page indexes are used during address relocation.

PARAMETER - An operand used by a program or subroutine.

PARITY CHECK - A summation check in which a group of binary digits are added and the sum checked against a previously computed parity digit; i.e., a check which tests whether the number of ones is odd or even.

PICTURE - A compact way of describing a data item. Among the things it may specify are size, class, sign, and editing.

PROGRAM - A precise sequence of instructions that accomplishes the solution of a problem. Also called a routine.

PROGRAM ADDRESS GROUP - A group of sequentially numbered addresses representing one or more programs within 32,768 words of storage. It is not a discrete physical device.

PROGRAM ADDRESS REGISTER - Synonomous with P register.

PROGRAM CONTROL - Synonomous with Main Control.

PROGRAM STATE - A highly efficient operating state of Executive mode in which all 3300 instructions may be executed except those instructions that call for I/O operations, alter certain register file locations, or halt the computer.

PROGRAM STATE NUMBER - One of seven Program Address Groups.

PSEUDO CODE - A statement requesting a specific operation by the assembler or compiler.

Q REGISTER - Auxiliary 24-bit arithmetic register which assists the A register in the more complicated arithmetic operations.

RADIX - The number of different digits that can occur in a digit position for a specific number system. It may be referred to as the base of a number system.

RANDOM ACCESS - Access to storage under conditions in which the next position from which information is to be obtained can be independent of the previous one.

READ - To remove a quantity from a storage location.

REGISTER - The internal logic used for temporary storage or for holding a quantity during computation.

REJECT - A signal generated under certain circumstances by either the external equipment or the processor during the execution of I/O instructions.

RELOCATION - Making efficient use of all memory locations by reassignment through the use of a memory paging system under control of a monitor program.

REPLACE - When used in the title of an instruction, the result of the execution of the instruction is stored in the location from which the initial operand was obtained. When replace is used in the description of an instruction, the contents of a location or register are substituted by the operand. The Replace operation implies clearing the register or portion of the register in preparation for the new quantity.

REPLY - A response signal in I/O operations that indicates a positive response to some previous operation or request signal.

REPORT GENERATOR - A language and compiler to reduce the programming necessary to generate reports.

RETURN JUMP - An instruction that jumps from a sequence of instructions to initiate a second sequence and prepares for continuing the first sequence after the second is completed.

ROUTINE - The sequence of operations which the computer performs; also called a program.

S REGISTER - The 13-bit S register displays the address of the storage word currently being referenced.

SCALE FACTOR - One or more coefficients by which quantities are multiplied or divided so that they lie in a given range of magnitude.

SCAN - Synonomous with Search.

SEARCH - Searching a field of characters for a certain condition or a specific character.

SHIFT - To move the bits of a quantity right or left.

SIGN BIT - In registers where a quantity is treated as signed by use of one's complement notation, the bit in the highest order stage of the register. If the bit is "1", the quantity is negative; if the bit is "0", the quantity is positive.

SIGN CHARACTER - A unique character that indicates the algebraic sign (positive or negative) of a given field of characters. The upper two bits of a normal 6-bit character may be used or in some instances a specially formed 4-bit character exits.

SIGN EXTENSION - The duplication of the sign bit in the higher order stages of a register.

SOFTWARE - Programs and/or subroutines.

SOURCE LANGUAGE - The language used by the programmer to define his program.

STAGE - The FFs and inverters associated with a bit position of a register.

STATUS - The state or condition of circuits within the processor, I/O channels, or external equipment.

STORAGE CONTROL - The sequence of events within a particular storage module, controlling various internal storage operations.

STORE - To transmit information to a device from which the unaltered information can later be obtained. The Store operation is essentially the reverse of the Load operation. Storage location M is cleared, and the contents of the register are copied into M.

SUBROUTINE - A set of instructions that is used at more than one point in program operation.

SYMBOLIC CODING - A system of abbreviation used in preparing information for input into a computer; e.g., Shift Q would be SHQ. (See Mnemonic)

TOGGLE - To complement each specified bit of a quantity; i.e., "1" to "0" or "0" to "1".

TRANSMIT (Transfer) - The term transfer implies register contents are moved; i.e., the contents of register 1 are copied into register 2. Unless specifically stated, the contents are not changed during transmission. The term transmit is often used synonymously with transfer.

TWO'S COMPLEMENT - Number that results from subtracting each bit of a number from "0". The two's complement may be formed by complementing each bit of the given number and then adding one to the result, performing the required carries.

UNDERFLOW - An illegal change of sign from - to +, e.g., subtracting from a quantity so that the result would be less than $-(2^n-1)$, where n is the modulus. In floating point notation, this occurs where the value of the exponent becomes less than $2^{-10}+1$ (-1777_8).

WORD - The content of a storage location; it can be an instruction or 24 bits of data.

WRITE - To enter a quantity into a storage location.

X REGISTER - An arithmetic transfer register, nonaddressable and not displayed.

Z REGISTER - A 28-bit storage data register; receives the data and parity bits as they are read from storage or written into storage. Nonaddressable but displayed on the 'T' panel in the storage module.

TABLE 1. OCTAL LISTING OF INSTRUCTIONS

Octal Code	Mnemonic Code	Address Field	Instruction Description	Page No.
00.0	HLT	m	Unconditional halt, RNI @ m on restarting	5-24
00.1	SJ1	m	If jump key 1 is set, jump to m	5-41
00.2	SJ2	m	If jump key 2 is set, jump to m	5-41
00.3	SJ3	m	If jump key 3 is set, jump to m	5-41
00.4	SJ4	m	If jump key 4 is set, jump to m	5-41
00.5	SJ5	m	If jump key 5 is set, jump to m	5-41
00.6	SJ6	m	If jump key 6 is set, jump to m	5-41
00.7	RTJ	m	P+1 → m (address portion), RNI @ m+1, return to m for P+1	5-47
01	UJP,I	m,b	Unconditional jump to m	5-41
02.0	No Operation			
02.1-3	IJI	m,b	If $(B^b) = 0$, RNI @ P+1; if $(B^b) \neq 0$, $(B^b) + 1 \rightarrow B^b$, RNI @ m	5-43
02.4	No Operation			
02.5-7	IJD	m,b	If $(B^b) = 0$, RNI @ P+1; if $(B^b) \neq 0$, $(B^b) - 1 \rightarrow B^b$, RNI @ m	5-44
03.0	AZJ,EQ	m	If (A) = 0, RNI @ m, otherwise RNI @ P+1	5-45
03.1	AZJ,NE	m	If (A) ≠ 0, RNI @ m, otherwise RNI @ P+1	5-45
03.2	AZJ,GE	m	If (A) ≥ 0, RNI @ m, otherwise RNI @ P+1	5-45
03.3	AZJ,LT	m	If (A) < 0, RNI @ m, otherwise RNI @ P+1	5-45
03.4	AQJ,EQ	m	If (A) = (Q), RNI @ m, otherwise RNI @ P+1	5-46
03.5	AQJ,NE	m	If (A) ≠ (Q), RNI @ m, otherwise RNI @ P+1	5-46
03.6	AQJ,GE	m	If (A) ≥ (Q), RNI @ m, otherwise RNI @ P+1	5-46
03.7	AQJ,LT	m	If (A) < (Q), RNI @ m, otherwise RNI @ P+1	5-46
04.0	ISE	y	If y = 0, RNI @ P+2, otherwise RNI @ P+1	5-28
04.1-3	ISE	y,b	If $y = (B^b)$, RNI @ P+2, otherwise RNI @ P+1	5-28
04.4	ASE,S	y	If y = (A), RNI @ P + 2, otherwise RNI @ P + 1. Sign of y extended.	5-29
04.5	QSE,S	y	If y = (Q), RNI @ P + 2, otherwise RNI @ P + 1. Sign of y extended.	5-29
04.6	ASE	y	If y = (A), RNI @ P + 2, otherwise RNI @ P + 1. Lower 15 bits of A are used.	5-29
04.7	QSE	y	If y = (Q), RNI @ P + 2, otherwise RNI @ P + 1. Lower 15 bits of Q are used.	5-29

TABLE 1. OCTAL LISTING OF INSTRUCTIONS (Cont'd)

Octal Code	Mnemonic Code	Address Field	Instruction Description	Page No.
05.0	ISG	y	If y = 0, RNI @ P+2, otherwise RNI @P+1	5-30
05.1-3	ISG	y,b	$(B^b) \geq y$,RNI@P+2,otherwise RNI @ P+1	5-30
05.4	ASG,S	y	If (A) $\geq$ y, RNI @ P+2, otherwise RNI @ P+1. Sign of y is extended	5-30
05.5	QSG,S	y	If (Q) $\geq$ y, RNI @ P+2, otherwise RNI @ P+1. Sign of y is extended	5-30
05.6	ASG	y	If (A) $\geq$ y, RNI @ P+2, otherwise RNI@P+1	5-30
05.7	QSG	y	If (Q)$\geq$y,RNI @ P+2,otherwise RNI @ P+1	5-30
06	MEQ	m,i	$(B^1) - i \rightarrow B^1$; if (B^1) negative,RNI @P+1; if (B^1) positive,test (A) = (Q) $\wedge$ (M); if true, RNI @ P+2; if false, repeat sequence	5-73
07	MTH	m,i	$(B^2)-i \rightarrow B^2$; if (B^2) negative RNI @ P+1; if (B^2) positive, test (A) $\geq$(Q) $\wedge$ (M); if true, RNI @ P+2; if false, repeat sequence	5-74
10.0	SSH	m	Test sign of (m), shift left one place, end around, replace in storage. If sign negative, RNI @P+2; otherwise RNI @ P+1	5-57
10.1-3	ISI	y,b	If (B^b)= y, clear B^b and RNI @ P+2; if $(B^b) \neq$ y, $(B^b) + 1 \rightarrow B^b$, RNI @ P+1	5-31
10.4-7	ISD	y,b	If (B^b) = y, clear B^b and RNI @ P+2; if $(B^b) \neq$ y, $(B^b) - 1 \rightarrow B^b$, RNI @ P+1	5-31
11.0-3	ECHA	z	z → A, lower 17 bits of A are used	5-26
11.4-7	ECHA,S	z	z → A, sign of z extended	5-26
12.0-3	SHA	k, b	Shift (A). Shift count K=k+(B^b) (signs of k and B^b extended.)If bit 23 of K="1", shift right; complement of lower 6 bits equals shift magnitude. If bit 23 of K="0", shift left; lower 6 bits equals shift magnitude. Left shifts end around;right shifts end off.	5-57
12.4-7	SHQ	k, b	Shift (Q). Shift count K=k+(B^b)(signs of k and B^b extended. If bit 23 of K="1", shift right;complement of lower 6 bits equals shift magnitude. If bit 23 of K="0", shift left;lower 6 bits equals shift magnitude. Left shifts end around;right shifts end off.	5-59
13.0-3	SHAQ	k, b	Shift (AQ) as one register. Shift count K= k+(B^b) (signs of k and B^b extended). If bit 23 of K="1", shift right;complement of lower 6 bits equals shift magnitude. If bit 23 of K="0", shift left;lower 6 bits equal shift magnitude. Left shifts end around; right shifts end off.	5-59

TABLE 1. OCTAL LISTING OF INSTRUCTIONS (Cont'd)

Octal Code	Mnemonic Code	Address Field	Instruction Description	Page No.
13.4-7	SCAQ	k, b	Shift (AQ) left end around until upper 2 bits of A are unequal. Residue K = k - shift count. If b = 1, 2, or 3, K → B^b; if b = 0, K is discarded.	5-59 5-59
14.0	No Operation		No operation (COMPASS assembled NOP)	
14.1-3	ENI	y, b	Clear B^b, enter y	5-25
14.4	ENA, S	y	Clear A, enter y, sign of y extended	5-25
14.5	ENQ, S	y	Clear Q, enter y, sign of y extended	5-25
14.6	ENA	y	Clear A, enter y	5-25
14.7	ENQ	y	Clear Q, enter y	5-25
15	No Operation			
15.1-3	INI	y, b	Increase (B^b) by y, signs of y and B^b extended	5-27 5-27
15.4	INA, S	y	Increase (A) by y, sign of y extended	5-27
15.5	INQ, S	y	Increase (Q) by y, sign of y extended	5-27
15.6	INA	y	Increase (A) by y	5-27
15.7	INQ	y	Increase (Q) by y	5-27
16.0	No Operation			
16.1-3	XOI	y, b	$y \vee (B^b) \rightarrow B^b$	5-69
16.4	XOA, S	y	$y \vee (A) \rightarrow A$, sign of y extended	5-69
16.5	XOQ, S	y	$y \vee (Q) \rightarrow Q$, sign of y extended	5-69
16.6	XOA	y	$y \vee (A) \rightarrow A$	5-69
16.7	XOQ	y	$y \vee (Q) \rightarrow Q$	5-69
17.0	No Operation			
17.1-3	ANI	y, b	$y \wedge (B^b) \rightarrow B^b$	5-71
17.4	ANA, S	y	$y \wedge (A) \rightarrow A$, sign of y extended	5-71
17.5	ANQ, S	y	$y \wedge (Q) \rightarrow Q$, sign of y extended	5-72
17.6	ANA	y	$y \wedge (A) \rightarrow A$	5-71
17.7	ANQ	y	$y \wedge (Q) \rightarrow Q$	5-72
20	LDA, I	m, b	(M) → A	5-49
21	LDQ, I	m, b	(M) → Q	5-51
22	LACH	r, 1	(R) → A. Load lower 6 bits of A	5-49
23	LQCH	r, 2	(R) → Q. Load lower 6 bits of Q	5-52

TABLE 1. OCTAL LISTING OF INSTRUCTIONS (Cont'd)

Octal Code	Mnemonic Code	Address Field	Instruction Description	Page No.
24	LCA,I	m,b	$(\overline{M}) \rightarrow A$	5-50
25	LDAQ,I	m,b	$(M) \rightarrow A$, $(M + 1) \rightarrow Q$	5-50
26	LCAQ,I	m,b	$(\overline{M}) \rightarrow A$, $(\overline{M + 1}) \rightarrow Q$	5-51
27	LDL,I	m,b	$(M) \wedge (Q) \rightarrow A$	5-50
30	ADA,I	m,b	Add (M) to $(A) \rightarrow A$	5-60
31	SBA,I	m,b	(A) minus $(M) \rightarrow A$	5-61
32	ADAQ,I	m,b	Add $(M, M + 1)$ to $(AQ) \rightarrow AQ$	5-61
33	SBAQ,I	m,b	(AQ) minus $(M, M + 1) \rightarrow AQ$	5-61
34	RAD,I	m,b	Add (M) to $(A) \rightarrow (M)$	5-60
35	SSA,I	m,b	Where (M) contains a "1" bit, set the corresponding bit in A to "1"	5-70
36	SCA,I	m,b	Where (M) contains a "1" bit, complement the corresponding bit in A	5-70
37	LPA,I	m,b	$(M) \wedge (A) \rightarrow A$	5-71
40	STA,I	m,b	$(A) \rightarrow (M)$	5-53
41	STQ,I	m,b	$(Q) \rightarrow (M)$	5-55
42	SACH	r,2	$(A_{00-05}) \rightarrow R$	5-54
43	SQCH	r,1	$(Q_{00-05}) \rightarrow R$	5-55
44	SWA,I	m,b	$(A_{00-14}) \rightarrow (M_{00-14})$	5-56
45	STAQ,I	m,b	$(AQ) \rightarrow (M, M + 1)$	5-54
46	SCHA,I	m,b	$(A_{00-16}) \rightarrow (M_{00-16})$	5-56
47	STI,I	m,b	$(B^b) \rightarrow (M_{00-14})$	5-56
50	MUA,I	m,b	Multiply (A) by $(M) \rightarrow QA$; lowest order bits of product in A	5-62 5-62
51	DVA,I	m,b	$(AQ) \div (M) \rightarrow A$, remainder → Q	5-62
52	CPR,I	m,b	$\left.\begin{array}{l}(M) > (A),\ \text{RNI @ P+1}\\(Q) > (M),\ \text{RNI @ P+2}\\(A) \geq (M) \geq (Q),\ \text{RNI @ P+3}\end{array}\right\}$ (A) and (Q) are unchanged	5-75
53.(0-3)0	TIA	b	Clear (A), $(B^b) \rightarrow A_{00-14}$	5-33
53.(5-7)0	TAI	b	$(A_{00-14}) \rightarrow B^b$	5-33
53.(0-3)1	TMQ	v	$(v) \rightarrow Q$	5-34
53.(4-7)1	TQM	v	$(Q) \rightarrow v$	5-34
53.(0-3)2	TMA	v	$(v) \rightarrow A$	5-34
53.(4-7)2	TAM	v	$(A) \rightarrow v$	5-34

TABLE 1. OCTAL LISTING OF INSTRUCTIONS (Cont'd)

Octal Code	Mnemonic Code	Address Field	Instruction Description	Page No.
53.(1-3)3	TMI	v,b	$(V_{00-14}) \rightarrow B^b$	5-35
53.43	TIM	v,b	$(B^b) \rightarrow v_{00-14}$	5-35
53.04	AQA		Add (A) to (Q) → A	5-32
53.(1-3)4	AIA	b	Add (A) to $(B^b) \rightarrow A$	5-33
53.(5-7)4	IAI	b	Add (A) to $(B^b) \rightarrow B^b$. Sign of B^b extended prior to addition	5-33
			All other combinations of 53 are undefined and will be rejected by the assembler	
54	LDI,I	m,b	$(M_{00-14}) \rightarrow B^b$	5-52
55.0	RIS		Use (ISR) in address relocation for operands. RELOCATE TO INSTRUCTION STATE	5-109
55.1	ELQ		$(E_L) \rightarrow Q$	5-36
55.2	EUA		$(E_U) \rightarrow A$	5-36
55.3	EAQ		$(E_UE_L) \rightarrow AQ$	5-36
55.4	ROS		Use (OSR) in address relocation for operands. RELOCATE TO OPERAND STATE	5-109
55.5	QEL		$(Q) \rightarrow E_L$	5-36
55.6	AEU		$(A) \rightarrow E_U$	5-36
55.7	AQE		$(AQ) \rightarrow E_UE_L$	5-36
56	MUAQ,I	m,b	Multiply (AQ) by (M,M + 1) → AQE	5-63
57	DVAQ,I	m,b	(AQE) ÷ (M,M + 1) → AQ and remainder with sign extended to E. Divide fault halts operation and program advances to next instruction	5-63
60	FAD,I	m,b	Floating point addition of (M,M + 1) to (AQ) → AQ	5-65
61	FSB,I	m,b	Floating point subtraction of (M,M + 1) from (AQ) → AQ	5-65
62	FMU,I	m,b	Floating point multiplication of (AQ) and (M,M + 1) → AQ	5-66
63	FDV,I	m,b	Floating point division of (AQ) by (M,M + 1) → AQ, remainder with sign extended to E	5-66
64.0	MVE	r, B_r, $\S_1$, s, B_s, $\S_2$	Move characters from fld A → fld C according to parameters given	5-123

TABLE 1. OCTAL LISTING OF INSTRUCTIONS (Cont'd)

Octal Code	Mnemonic Code	Address Field	Instruction Description	Page No.
64.0	MVE,DC	r, B_r, s, B_s, $\S_2$	Move characters from fld A → fld C according to parameters given. Delimiting character possibility	5-124
64.1	MVBF	r, B_r, $\S_1$, s, B_s, $\S_2$	Move characters from fld A → fld C; if fld C > fld A, blank fill	5-125
64.2	MVZF	r, B_r, $\S_1$, s, B_s, $\S_2$	Move characters from fld A → fld C; if fld C > fld A, zero fill	5-126
64.3	MVZS	r, B_r, $\S_1$, s, B_s, $\S_2$	Move characters from fld A → fld C; suppress leading zeros	5-127
64.3	MVZS,DC	r, B_r, s, B_s, $\S_2$	Move characters from fld A → fld C; suppress leading zeros. Delimiting character possibility	5-128
64.4	EDIT	r, B_r, $\S_1$, s, B_s, $\S_2$	Fld A → fld C with COBOL type of editing specified by picture previously stored in fld C	5-132
64.4	FRMT	r, B_r, $\S_1$, s, B_s, $\S_2$	Fld A → fld C with editing specified by picture previously stored in fld; limited to specific types of editing to allow processing in a single scan.	5-130
65.0	SCAN, LR,EQ	r, B_r, $\S_2$, SC	Scans fld A from left to right, stop on = condition	5-138
65.0	SCAN, LR, EQ, DC	r, B_r, $\S_2$, SC	Scans fld A from left to right, stop on = condition. Delimiting character possibility	5-139
65.1	SCAN, RL,EQ	r, B_r, $\S_2$, SC	Scans fld A from right to left, stop on = condition	5-142
65.1	SCAN, RL, EQ, DC	r, B_r, $\S_2$, SC	Scans fld A from right to left, stop on = condition. Delimiting character possibility	5-143
65.2	SCAN, LR,NE	r, B_r, $\S_2$, SC	Scans fld A from left to right, stop on ≠ condition	5-140
65.2	SCAN, LR, NE DC	r, B_r, $\S_2$, SC,	Scan fld A from left to right, stop on ≠ condition. Delimiting character possibility	5-141
65.3	SCAN, RL, NE	r, B_r, $\S_2$, SC	Scans fld A from right to left, stop on ≠ condition	5-144

TABLE 1. OCTAL LISTING OF INSTRUCTIONS (Cont'd)

Octal Code	Mnemonic Code	Address Field	Instruction Description	Page No.
65.3	SCAN, RL, NE, DC	r, B_r, $\S_2$, SC,	Scans fld A from right to left, stop on ≠ condition. Delimiting character possibility	5-145
66.0	CVDB	r, B_r, $\S_1$, m, B_m	Convert BCD fld A to binary fld → C	5-146
66.1	CVBD	m, B_m, n, B_n	Convert binary fld A to BCD → fld C	5-147
66.2	DTA*	r, B_r, $\S_2$, m, B_m	Translate BCD fld A to ASCII → fld C	5-148
66.2	DTA, DC*	r, B_r, $\S_2$, m, B_m	Translate BCD fld A to ASCII → fld C with delimiting character possibility	5-149
66.3	ATD*	m, B_m, $\S_2$, s, B_s	Translate ASCII fld A to BCD → fld C	5-150
66.3	ATD, DC*	m, B_m, $\S_2$, s, B_s	Translate ASCII fld A to BCD → fld C with delimiting character possibility	5-151
66.4	PAK	r, B_r $\S_2$, m, B_m	Pack 6-bit BCD fld A into 4-bit BCD fld C	5-152
66.5	UPAK	m, B_m, s, B_s, $\S_2$	Unpack 4-bit BCD fld A into 6-bit BCD fld C	5-153
67.0	ADM	r, B_r, $\S_1$, s, B_s, $\S_2$	Add fld A to fld C → fld C	5-154
67.1	SBM	r, B_r, $\S_1$, s, B_s, $\S_2$	Subtract fld A from fld C → fld C	5-156
67.2	ZADM	r, B_r, $\S_1$, s, B_s, $\S_2$	Clear fld C; fld A → fld C, right justify	5-129

*Available in 3312 and 3304-2 only.

TABLE 1. OCTAL LISTING OF INSTRUCTIONS (Cont'd)

Basic Octal Code	Mnemonic Code	Address Field	Instruction Description	Page No.
67.3	CMP*	r, B_r, $\S_1$, s, B_s, $\S_2$	Compares fld A to fld C, exits upon encountering ≠ characters	5-158
67.3	CMP, DC*	r, B_r, s, B_s, $\S_2$	Compares fld A to fld C, exits upon encountering ≠ characters; delimiting character possibility	5-162
67.3	CMP**	r, B_r, S_1, s, B_s, S_2	Collating Compare of Field A with Field C	5-159
67.3	CMP, N**	r, B_r, S_1, s, B_s, S_2	Numeric compare of Field A with Field C	5-159
67.4	TST	r, Br, $\S_1$	Test fld A, +, -, or 0	5-165
67.4	TSTN	r, B_r, $\S_1$	Test fld A for numeric	5- 166
70.0	JMP, HI	m	Jump if BDP condition register > 0 or +	5-42
70.1	JMP, ZRO	m	Jump if BDP condition register = 0	5-42
70.2	JMP, LOW	m	Jump if BDP condition register < 0 or -	5-42
70.6	LBR	m	Load BCR and restore BDP conditions from data at 'm'	5- 167
70.7	SBR	m	Store (BCR) and BDP conditions at 'm' for interrupt recovery.	5- 168
71	SRCE, INT	SC, r, s	Search for equality of scan character 'SC' in a field beginning at location r until an equal character is found, or until character location s is reached;	5- 111
71	SRCN, INT	SC, r, s	Same as SRCE except search condition is for inequality	5- 113
72	MOVE, INT	§, r, s	Move (§) characters from r to s	5- 115
73.0-3	INPC, INT B, H, G	ch, r, s	A 6- or 12-bit character is read from peripheral device and stored in memory at a given location	5- 95
73.4-7	INAC, INT	ch	(A) is cleared and a 6-bit character is transferred from a peripheral device to the lower 6 bits of A	5- 103
74.0-3	INPW, INT, B, N, G	ch, m, n	Word address is placed in bits 00-14, 12- or 24-bit words are read from a peripheral device and stored in memory	5- 97

*Available in 3312 and 3304-2 only.
**Available in 3304-3 only.

TABLE 1. OCTAL LISTING OF INSTRUCTIONS (Cont'd)

Basic Octal Code	Mnemonic Code	Address Field	Instruction Description	Page No.
74.4-7	INAW, INT	ch	(A) is cleared and a 12-or 24-bit word is read from a peripheral device into the lower 12 bits or all of A (word size depends on I/O channel)	5-104
75.0-3	OUTC, INT,B,H	ch,r,s	Storage words disassembled into 6- or 12-bit characters and sent to a peripheral device	5-99
75.4-7	OTAC, INT	ch	Character from lower 6 bits of A is sent to a peripheral device,(A) retained	5-106
76.0-3	OUTW, INT,B,N	ch,m,n	Words read from storage to a peripheral device	5-101
76.4-7	OTAW, INT	ch	Word from lower 12 bits or all of A (depending on type of I/O channel) sent to a peripheral device	5-107
77	CON	x,ch	If channel ch is busy, reject instruction RNI @ P + 1. If channel ch is not busy, 12-bit connect code sent on channel ch with connect enable, RNI @ P + 2	5-90
77.1	SEL	x, ch	If channel ch is busy, read reject instruction from P + 1. If channel ch is not busy, a 12-bit function code is sent on channel ch with a function enable, RNI @ P + 2	5-92
77.2 ch, X; X ≠ 0	EXS	x, ch	Sense external status if "1" bits occur on status lines in any of the same positions as "1" bits in the mask, RNI @ P + 1. If no comparison, RNI @ P + 2	5-78
77.2 ch, X; X = 0	COPY	ch	External status code from I/O channel ch → lower 12 bits of A, contents of interrupt mask register → upper 12 bits of A; RNI @ P + 1	5-78
77.3 ch, X; X ≠ 0	INS	x, ch	Sense internal status if "1" bits occur on status lines in any of the same positions as "1" bits in the mask, RNI @ P + 1. If no comparison, RNI @ P + 2	5-80
77.3 ch, X; X = 0	CINS	ch	Interrupt mask and internal status to A	5-81
77.4	INTS	x, ch	Sense for interrupt condition: If "1" bits occur simultaneously in interrupt lines and in the interrupt mask, RNI @ P + 1; if not, RNI @ P + 2	5-79
77.50	INCL	x	Interrupt faults defined by x are cleared	5-84

TABLE 1. OCTAL LISTING OF INSTRUCTIONS (Cont'd)

Basic Octal Code	Mnemonic Code	Address Field	Instruction Description	Page No.
77.51	IOCL	x	Clears I/O channel or search/move control as defined by bits 00-07, 08, and 11 of x.	5- 89
77.511	CILO	x	Lockout external interrupt on masked channels (x), until channel(s) is not busy.	5- 86
77.512	CLCA	x	Clear the specified channel, but not external equipment. CLEAR CHANNEL ACTIVITY	5- 89
77.52	SSIM	x	Selectively set interrupt mask register for each "1" bit in x. The corresponding bit in the mask register is set to "1"	5- 85
77.53	SCIM	x	Selectively clear interrupt mask register for each "1" bit in x. The corresponding bit in the mask register is set to "0"	5- 85
77.54	ACI		A_{00-02} → CIR A TO CHANNEL INDEX REGISTER	5-38
77.55	CIA		Clear A; Channel index register → A_{00-02}	5-38
77.56	JAA		Last executed jump address → A JUMP ADDRESS TO A	5-40
77.57	IAPR		Interrupt associated processor	5- 110
77.60	PAUS	x	Sense busy lines. If "1" appears on a line corresponding to "1" bits in x, do not advance P. If P is inhibited for longer than 40 ms, read reject instruction from P + 1. If no comparison, RNI @ P + 2	5- 82
77.61X X ≠ 0	PRP		Same as PAUS except real-time clock cannot increment during the pause PRIORITY PAUSE	5- 83
77.61X X = 0	TMAV		Initiate memory request. If reply occurs within 5 usec, RNI @ P + 2; if not, RNI @ P + 1. Storage address is (B^2) with (OSR) or zero appended. TEST MEMORY AVAILABILITY	5- 77
77.62	SBJP		Transfers system from Monitor State to Program State when next jump instruction is executed. SET BOUNDARY JUMP	5- 109
77.624	SDL		Causes next LDA instruction to: 1. (M) → A 2. Store 77777777 @ M SET DESTRUCTIVE LOAD	5- 110
77.63	CRA		Clear A; Condition register → A_{00-05}	5-40
77.634	ACR		A_{00-05} → Condition register	5-40

TABLE 1. OCTAL LISTING OF INSTRUCTIONS (Cont'd)

Basic Octal Code	Mnemonic Code	Address Field	Instruction Description	Page No.
77.64	APF	w, 2	A_{00-11} to page file A TO PAGE FILE	5-39
77.65	PFA	w, 2	Clear A, page file index → A_{00-11} PAGE FILE TO A	5-39
77.66	AOS		A_{00-02} → OSR A TO OPERAND STATE REGISTER	5-37
77.664	AIS		A_{00-02} → ISR A TO INSTRUCTION STATE REGISTER	5-37
77.67	OSA		Clear A; OSR → A_{00-02} OPERAND STATE REGISTER TO A	5-37
77.674	ISA		Clear A; ISR → A_{00-02} INSTRUCTION STATE REGISTER TO A	5-37
77.70	SLS		Program stops if Selective Stop switch is on; upon restarting, RNI @ P + 1	5-24
77.71	SFPF		Set floating point fault logic	5-86
77.72	SBCD		Set BCD fault logic	5-86
77.73	DINT		Disables interrupt control	5-84
77.74	EINT		Interrupt control is enabled, allows one more instruction to be executed before interrupt	5-84
77.75	CTI		Set Type In } Beginning character address must be preset in location 23 of register file and last character address + 1 must be preset in location 33 of the file	5-94
77.76	CTO		Set Type Out	
77.77	UCS		Unconditional stop. Upon restarting, RNI @ P + 1	5-24

TABLE 2. ALPHAMNEMONIC LISTING OF INSTRUCTIONS

Mnemonic Code	Basic Octal Code	Address Field	Instruction Description	Page No.
ACI	77		$A_{00-02} \rightarrow$ CIR A TO CHANNEL INDEX REGISTER	5-38
ACR	77		A_{00-05} to Condition register	5-40
ADA, I	30	m, b	Add (M) to (A) → A	5-60
ADAQ, I	32	m, b	Add (M, M + 1) to (AQ) → AQ	5-61
ADM	67	r, B_r, $\S_1$, s, B_s, $\S_2$	Add fld A to fld C → fld C	5-154
AEU	55		(A) → E_U	5-36
AIA	53	b	Add (A) to (B^b) → A	5-33
AIS	77		$A_{00-02} \rightarrow$ ISR A TO INSTRUCTION STATE REGISTER	5-37
ANA	17	y	y ∧ (A) → A	5-71
ANA,S	17	y	y ∧ (A) → A, sign of y extended	5-71
ANI	17	y,b	y ∧ (B^b) → B^b	5-71
ANQ	17	y	y ∧ (Q) → Q	5-72
ANQ,S	17	y	y ∧ (Q) → Q, sign of y extended	5-72
AOS	77		$A_{00-02} \rightarrow$ OSR A TO OPERAND STATE REGISTER	5-37
APF	77	w, 2	$A_{00-11} \rightarrow$ page file A TO PAGE FILE	5-39
AQA	53		Add (A) to (Q) → A	5-32
AQE	55		(AQ) → $E_U E_L$	5-36
AQJ,EQ	03	m	If (A) = (Q), RNI @ m, otherwise RNI @ P + 1	5-46
AQJ,GE	03	m	If (A) ≥ (Q), RNI @ m, otherwise RNI @ P + 1	5-46
AQJ,LT	03	m	If (A) < (Q), RNI @ m, otherwise RNI @ P + 1	5-46
AQJ,NE	03	m	If (A) ≠ (Q), RNI @ m, otherwise RNI @ P + 1	5-46
ASE	04	y	If y = (A), RNI @ P + 2, otherwise RNI @ P + 1. Lower 15 bits of A are used.	5-29
ASE,S	04	y	If y = (A), RNI @ P + 2, otherwise RNI @ P + 1, sign of y is extended	5-29
ASG	05	y	If (A) ≥ y, RNI @ P + 2, otherwise RNI @ P + 1	5-30

TABLE 2. ALPHAMNEMONIC LISTING OF INSTRUCTIONS (Cont'd)

Mnemonic Code	Basic Octal Code	Address Field	Instruction Description	Page No.
ASG,S	05	y	If (A) $\geq$ y, RNI @ P + 2, otherwise RNI @ P + 1, sign of y is extended	5-30
ATD	66	m,B_m, S_2, s, B_s	Translate ASCII fld A to BCD → fld C	5-150
ATD,DC	66	m,B_m, S_2, s, B_s	Translate ASCII fld to BCD → fld C with delimiting character possibility	5-151
AZJ,EQ	03	m	If (A) = 0, RNI @ m, otherwise RNI @ P + 1	5-45
AZJ,GE	03	m	If (A) $\geq$ 0, RNI @ m, otherwise RNI @ P + 1	5-45
AZJ,LT	03	m	If (A) < 0, RNI @ m, otherwise RNI @ P + 1	5-45
AZJ,NE	03	m	If (A) ≠ 0, RNI @ m, otherwise RNI @ P + 1	5-45
CIA	77		Clear A; Channel index register → A_{00-02}	5-38
CILO	77	x	Lockout external interrupt on masked channels (x), until channel(s) is not busy	5-86
CINS	77	ch	Interrupt mask and internal status to A	5-81
CLCA	77	x	Clear the specified channel, but not external equipment. CLEAR CHANNEL ACTIVITY	5-89
CMP*	67	r,B_r, S_1, s, B_s, S_2	Compares fld A to fld C, exits upon encountering ≠ characters	5-158
CMP,DC*	67	r, B_r, s, B_s S_2	Compares fld A to fld C, exits upon encountering ≠ characters; delimiting character possibility	5-162
CMP**	67	r, B_r, S_1, s, B_s, S_2	Collating compare of Field A with Field C	5-159
CMP, N*	67	r, B_r, S_1; s, B_s, S_2	Numeric compare of Field A with Field C	5-159
CON	77	x, ch	If channel ch is busy, reject instruction, RNI @ P+1. If channel ch is not busy, 12-bit connect code sent on channel ch with connect enable, RNI @ P+2.	5-90
COPY	77	ch	External status code from I/O channel ch to lower 12-bits of A, contents of interrupt mask register to upper 12-bits of A RNI @ P+1	5-78
CPR,I	52	m,b	(M) > (A), RNI @ P+1 (Q) > (M), RNI @ P + 2 (A) $\geq$ (M) $\geq$ (Q), RNI @ P+3 } (A) and (Q) are unchanged	5-75
CRA	77		Condition register to A_{00-05}	5-40

*Available in 3312 and 3304-2 only
**Available in 3304-3 only

TABLE 2. ALPHAMNEMONIC LISTING OF INSTRUCTIONS (Cont'd)

Mnemonic Code	Basic Octal Code	Address Field	Instruction Description	Page No.
CTI	77		Set Type In } Beginning character address must be preset in location 23 of register file and last character address + 1 must be preset in location 33 of the file	5-94
CTO	77		Set Type Out	
CVBD	66	r, B_r, m, B_m	Convert binary fld A to BCD → fld C	5-147
CVDB	66	m, Bm, §1, s, B_s	Convert BCD fld A to binary → fld C	5-146
DINT	77		Disables interrupt control	5-84
DTA	66	r, B_r, §2, m, B_m	Translate BCD fld A to ASCII → fld C	5-148
DTA,DC	66	r, B_r, §2, m, B_m	Translate BCD fld A to ASCII → fld C with delimiting character possibility	5-149
DVA,I	51	m,b	(AQ) ÷ (M) → A, remainder → Q	5-62
DVAQ,I	57	m,b	(AQE) ÷ (M,M + 1) → AQ and remainder with sign extended to E. Divide fault halts operation and program advances to next instruction	5-63
EAQ	55		$(E_UE_L) \rightarrow AQ$	5-36
ECHA	11	z	z → A, lower 17 bits of A are used	5-26
ECHA,S	11	z	z → A, sign of z extended	5-26
EDIT	64	r, B_r, §1, s, B_s, §2	Fld A → fld C with COBOL type of editing specified by picture previously stored in fld C	5-132
EINT	77		Interrupt control is enabled. Allows one more instruction to be executed before interrupt	5-84
ELQ	55		$(E_L) \rightarrow Q$	5-36
ENA	14	y	Clear A, enter y	5-25
ENA,S	14	y	Clear A, enter y, sign of y extended	5-25
ENI	14	y,b	Clear B^b, enter y	5-25
ENQ	14	y	Clear Q, enter y	5-25

TABLE 2. ALPHAMNEMONIC LISTING OF INSTRUCTIONS (Cont'd)

Mnemonic Code	Basic Octal Code	Address Field	Instruction Description	Page No.
ENQ,S	14	y	Clear Q, enter y, sign of y extended	5-25
EUA	55		$(E_U) \rightarrow A$	5-36
EXS	77	x,ch	Sense external status if "1" bits occur on status lines in any of the same positions as "1" bits in the mask, RNI @ P+1. If no comparison RNI @ P+2.	5-78
FAD,I	60	m,b	Floating point addition of (M,M + 1) to (AQ) → AQ	5-65
FDV,I	63	m,b	Floating point division of (AQ) by (M,M + 1) → AQ, Remainder with sign extended to E.	5-66
FMU,I	62	m,b	Floating point multiplication of (AQ) and (M,M + 1) → AQ	5-66
FRMT	64	r,Br, $\$_1$,s, B_s,$\$_2$	Fld A → fld C with editing specified by picture previously stored in fld; limited to specific types of editing to allow processing in a single scan.	5-130
FSB,I	61	m,b	Floating point subtraction of (M,M + 1) from (AQ) → AQ	5-65
HLT	00	m	Unconditional halt, RNI @ m upon restarting	5-24
IAI	53	b	Add (A) to $(B^b) \rightarrow B^b$. Sign of B^b extended prior to addition	5-33
IAPR	77		Interrupt associated processor	5-110
IJD	02	m,b	If $(B^b) = 0$, RNI @ P + 1; if $(B^b) \neq 0$, $(B^b) - 1 \rightarrow B^b$, RNI @ m	5-44
IJI	02	m,b	If $(B^b) = 0$, RNI @ P + 1; if $(B^b) \neq 0$, $(B^b) + 1 \rightarrow B^b$. RNI @ m	5-43
INA	15	y	Increase (A) by y	5-27
INA,S	15	y	Increase (A) by y, sign of y extended	5-27
INAC, INT	73	ch	(A) is cleared and a 6-bit character is transferred from a peripheral device to the lower 6 bits of A	5-103
INAW, INT	74	ch	(A) is cleared and a 12- or 24-bit word is read from a peripheral device into the lower 12 bits or all of A (word size depends on I/O channel)	5-104
INCL	77	x	Interrupt faults defined by x are cleared	5-84
INI	15	y,b	Increase (B^b) by y, signs of y and B^b extended	5-27

TABLE 2. ALPHAMNEMONIC LISTING OF INSTRUCTIONS (Cont'd)

Mnemonic Code	Basic Octal Code	Address Field	Instruction Description	Page No.
INPC, INT,B, H,G	73	ch,r,s	A 6- or 12-bit character is read from a peripheral device and stored in memory at a given location	5-95
INPW, INT,B, N,G	74	ch,m, n	Word Address is placed in bits 00-14, 12- or 24-bit words are read from a peripheral device and stored in memory	5-97
INQ	15	y	Increase (Q) by y	5-27
INQ,S	15	y	Increase (Q) by y, sign of y extended	5-27
INS	77	x,ch	Sense internal status if "1" bits occur on status lines in any of the same positions as "1" bits in the mask, RNI @ P + 1. If no comparison, RNI @ P + 2	5-80
INTS	77	c,ch	Sense for interrupt condition; if "1" bits occur simultaneously in interrupt lines and in the interrupt mask, RNI @ P + 1; if not, RNI @ P + 2	5-79
IOCL	77	x	Clears I/O channel or search/move control as defined by bits 00-07, 08, and 11 of x.	5-89
ISA	77		Clear A, ISR $\rightarrow A_{00-02}$ INSTRUCTION STATE REGISTER TO A	5-37
ISD	10	y,b	If $(B^b) = y$, clear B^b and RNI @ P + 2; if $(B^b) \neq y$, $(B^b) - 1 \rightarrow B^b$, RNI @ P + 1	5-31
ISE	04	y	If y = 0, RNI @ P+2, otherwise RNI @ P + 1	5-28
ISE	04	y,b	If $y = (B^b)$, RNI @ P + 2, otherwise RNI @ P + 1	5-28
ISG	05	y	If y = 0, RNI @ P + 2, otherwise RNI @ P + 1	5-30
ISG	05	y,b	If $(B^b) \geq y$, RNI @ P + 2, otherwise RNI @ P + 1	5-30
ISI	10	y,b	If $(B^b) = y$, clear B^b and RNI @ P + 2; if $(B^b) \neq y$, $(B^b) + 1 \rightarrow B^b$, RNI @ P + 1	5-31
JAA	77		Last executed jump address → A JUMP ADDRESS TO A	5-40
JMP,HI	70	m	Jump if BDP condition register >0 or +	5-42
JMP, LOW	70	m	Jump if BDP condition register <0 or -	5-42
JMP, ZRO	70	m	Jump if BDP condition register = 0	5-42

TABLE 2. ALPHAMNEMONIC LISTING OF INSTRUCTIONS (Cont'd)

Mnemonic Code	Basic Octal Code	Address Field	Instruction Description	Page No.
LACH	22	r, 1	(R) → A; load lower 6 bits of A	5-49
LBR	70	m	Load BCR and restore BDP conditions from data at 'm'	5-167
LCA,I	24	m,b	$(\overline{M}) \rightarrow A$	5-50
LCAQ,I	26	m,b	$(\overline{M}) \rightarrow A, (\overline{M + 1}) \rightarrow Q$	5-51
LDA,I	20	m,b	$(M) \rightarrow A$	5-49
LDAQ,I	25	m,b	$(M) \rightarrow A, (M + 1) \rightarrow Q$	5-50
LDI,I	54	m,b	$(M_{00-14}) \rightarrow B^b$	5-52
LDL,I	27	m,b	$(M) \wedge (Q) \rightarrow A$	5-50
LDQ,I	21	m,b	$(M) \rightarrow Q$	5-51
LPA,I	37	m,b	$(M) \wedge (A) \rightarrow A$	5-71
LQCH	23	r,2	(R) → Q; load lower 6 bits of Q	5-52
MEQ	06	m,i	$(B^1) - i \rightarrow B^1$; if (B^1) negative, RNI @ P + 1; if (B^1) positive, test $(A) = (Q) \wedge (M)$; if true, RNI @ P + 2, if false, repeat sequence	5-73
MOVE, INT	72	§,r,s	Move (§) characters from r to s	5-115
MTH	07	m,i	$(B^2) - i \rightarrow B^2$, if (B^2) negative, RNI @ P + 1; if (B^2) positive, test $(A) \geq (Q) \wedge (M)$; if true, RNI @ P + 2; if false,repeat sequence	5-74
MUA,I	50	m,b	Multiply (A) by (M) → QA; lowest order bits of product in A	5-62
MUAQ,I	56	m,b	Multiply (AQ) by (M,M + 1) → AQE	5-63
MVBF	64	r, B_r, $§_1$, s, B_s, $§_2$	Move characters from fld A → fld C; if fld C > fld A, blank fill	5-125
MVE	64	r, B_r, $§_1$, s, B_s, $§_2$	Move characters from fld A → fld C according to parameters given	5-123
MVE,DC	64	r, B_r, s, B_s, $§_2$	Move characters from fld A → fld C according to parameters given. Delimiting character possibility	5-124
MVZF	64	r, B_r, $§_1$, s, B_s, $§_2$	Move characters from fld A → fld C; if fld C > fld A, zero fill	5-126

TABLE 2. ALPHAMNEMONIC LISTING OF INSTRUCTIONS (Cont'd)

Mnemonic Code	Basic Octal Code	Address Field	Instruction Description	Page No.
MVZS	64	r, B_r, $\S_1$, s, B_s, $\S_2$	Move characters from fld A → fld C; suppress leading zeros	5-127
MVZS,DC	64	r, B_r, s, B_s, $\S_2$	Move characters from fld A → fld C; suppress leading zeros. Delimiting character possibility	5-128
OSA	77		Clear A; OSR → A_{00-02} OPERAND STATE REGISTER TO A	5-37
OTAC, INT	75	ch	Character from lower 6 bits of A is sent to peripheral device ,(A) retained	5-106
OTAW, INT	76	ch	Word from lower 12 bits or all of A (depending on type of I/O channel) sent to a peripheral device	5-107
OUTC, INT,B, H	75	ch,r,s	Storage words disassembled into 6 or 12-bit characters and sent to a peripheral device	5-99
OUTW, INT,B,N	76	ch,m,n	Words read from storage to peripheral device	5-101
PAK	66	r,B_r,$\S_2$ m, B_m	Pack 6-bit BCD fld A into 6-bit BCD fld C	5-152
PAUS	77	x	Sense busy lines. If "1" appears on a line corresponding to "1" bits in x, do not advance P. If P is inhibited for longer than 40 ms, read reject instruction from P + 1. If no comparison, RNI @ P + 2	5-82
PFA	77	w, 2	Clear A, page file index → A_{00-11} PAGE FILE TO A	5-39
PRP	77		Same as PAUS except real time clock cannot increment during the pause PRIORITY PAUSE	5-83
QEL	55		(Q) → E_L	5-36
QSE	04	y	If y = (Q), RNI @ P + 2, otherwise RNI @ P + 1; lower 15 bits of Q are used	5-29
QSE,S	04	y	If y = (Q), RNI @ P + 2, otherwise RNI @ P + 1, sign of y is extended	5-29
QSG	05	y	If (Q) ≥ y, RNI @ P + 2, otherwise RNI @ P + 1	5-30
QSG,S	05	y	If (Q) ≥ y, RNI @ P + 2, otherwise RNI @ P + 1, sign of y is extended	5-30

TABLE 2. ALPHAMNEMONIC LISTING OF INSTRUCTIONS (Conc'd)

Mnemonic Code	Basic Octal Code	Address Field	Instruction Description	Page No.
RAD,I	34	m,b	Add (M) to (A) → (M)	5-60
RIS	55		Use (ISR) in address relocation for operands RELOCATE TO INSTRUCTION STATE	5-109
ROS	55		Use (OSR) in address relocation for operands. RELOCATE TO OPERAND STATE	5-109
RTJ	00	m	P + 1 → M (address portion) RNI @ m + 1, return to m for P + 1	5-47
SACH	42	r,2	$(A_{00-05}) \rightarrow R$	5-54
SBA,I	31	m,b	(A) minus (M) → A	5-61
SBAQ,I	33	m,b	(AQ) minus (M,M + 1) → AQ	5-61
SBCD	77		Set BCD fault logic	5-86
SBJP	77		Transfers system from Monitor State to Program State when next jump instruction is executed. SET BOUNDRY JUMP	5-109
SBM	67	r, B_r, $\S_1$, s, B_s, $\S_2$	Subtract fld A from fld C → fld C	5-156
SBR	70	m	Store (BCR) and BDP conditions at 'm' for interrupt recovery	5-168
SCA, I	36	m, b	Where (M) contains a "1" bit, complement the corresponding bit in A	5-70
SCAN, LR, EQ, DC	65	r, B_r, $\S_2$, SC	Scans fld A from left to right, stop on = condition. Delimiting character possibility	5-139
SCAN, LR NE, DC	65	r,B_r,$\S_2$ SC	Scans fld A from left to right, stop on ≠ condition. Delimiting character possibility	5-141
SCAN, RL, EQ, DC	65	r, B_r, $\S_2$, SC	Scans fld A from right to left, stop on = condition. Delimiting character possibility	5-143
SCAN, RL, NE, DC	65	r, Br, $\S_2$, SC	Scans fld A from right to left, stop on ≠ condition. Delimiting character possibility	5-145
SCAN, LR, EQ	65	r, Br, $\S_2$, SC	Scans fld A from left to right, stop on = condition	5-138
SCAN, LR, NE	65	r, B_r, $\S_2$,SC	Scans fld A from left to right, stop on ≠ condition	5-140

TABLE 2. ALPHAMNEMONIC LISTING OF INSTRUCTIONS (Cont'd)

Mnemonic Code	Basic Octal Code	Address Field	Instruction Description	Page No.
SCAN, RL, EQ	65	r, B_r, $\mathcal{S}_2$, SC	Scans fld A from right to left, stop on = condition	5-142
SCAN, RL, NE	65	r, B_r, $\mathcal{S}_2$, SC	Scans fld A from right to left, stop on ≠ condition	5-144
SCAQ	13	k, b	Shift (AQ) left end around until upper 2 bits of A are unequal. Residue K = k shift count. If b = 1, 2, or 3, K → B^b; if b = 0, K is discarded	5-59
SCHA, I	46	m, b	$(A_{00-16}) \rightarrow (M_{00-16})$	5-56
SCIM	77	x	Selectively clear Interrupt Mask Register for each "1" bit in x. The corresponding bit in the mask register is set to "0".	5-85
SDL	77		Causes next LDA instruction to: 1. (M) → A 2. Store 77777777 @ M SET DESTRUCTIVE LOAD	5-110
SEL	77	x, ch	If channel ch is busy, read reject instruction from P + 1. If channel ch is not busy, a 12-bit function code is sent on channel ch with a function enable RNI @ P + 2	5-92
SFPF	77		Set floating point fault logic	5-86
SHA	12	k, b	Shift (A). Shift count K=k + (B^b) (signs of k and B^b extended). If bit 23 of K="1", shift right; complement of lower 6 bits equals shift magnitude. If bit 23 of K = "0", shift left; lower 6 bits equal shift magnitude. Left shifts end around; right shifts end off	5-57
SHAQ	13	k, b	Shift (AQ) as one register. Shift count K = k + B^b (signs of k and B^b extended). If bit 23 of K = "1", shift right and complement of lower 6 bits equals shift magnitude. If bit 23 of K = "0", shift left and lower 6 bits equal shift magnitude. Left shifts end around; right shifts end off	5-59
SHQ	12	k, b	Shift (Q), Shift count K=k + (B^b) (signs of k and B^b extended). If bit 23 of K = "1", shift right, complement of lower 6 bits equals shift magnitude. If bit 23 of K = "0", shift left, lower 6 bits equal shift magnitude. Left shifts end around; right shifts end off	5-59

TABLE 2. ALPHAMNEMONIC LISTING OF INSTRUCTIONS (Cont'd)

Mnemonic Code	Basic Octal Code	Address Field	Instruction Description	Page No.
SJ1	00	m	If jump key 1 is set, jump to m	5-41
SJ2	00	m	If jump key 2 is set, jump to m	5-41
SJ3	00	m	If jump key 3 is set, jump to m	5-41
SJ4	00	m	If jump key 4 is set, jump to m	5-41
SJ5	00	m	If jump key 5 is set, jump to m	5-41
SJ6	00	m	If jump key 6 is set, jump to m	5-41
SLS	77		Program stops if Selective Stop switch is on; upon restarting RNI @ P + 1	5-24
SQCH	43	r,1	$(Q_{00-05}) \rightarrow R$	5-55
SRCE, INT	71	SC,r,s	Search for equality of scan character SC in a field beginning at location r until an equal character is found, or until character location s is reached.	5-111
SRCN, INT	71	SC,r,s	Same as SRCE except search condition is for inequality	5-113
SSA,I	35	m,b	Where (M) contains a "1" bit, set the corresponding bit in A to "1"	5-70
SSH	10	m	Test sign of (m), shift (m) left one place, end around and replace in storage. If sign negative, RNI @ P + 2; otherwise RNI @ P + 1	5-57
SSIM	77	x	Selectively set interrupt mask register for each "1" bit in x. The corresponding bit in the mask register is set to "1"	5-85
STA,I	40	m,b	$(A) \rightarrow (M)$	5-53
STAQ,I	45	m,b	$(AQ) \rightarrow (M, M + 1)$	5-54
STI,I	47	m,b	$(B^b) \rightarrow (M_{00-14})$	5-56
STQ,I	41	m,b	$(Q) \rightarrow (M)$	5-55
SWA,I	44	m,b	$(A_{00-14}) \rightarrow (M_{00-14})$	5-56
TAI	53	b	$(A_{00-14}) \rightarrow B^b$	5-33
TAM	53	v	$(A) \rightarrow v$	5-34
TIA	53	b	Clear (A), $(B^b) \rightarrow A_{00-14}$	5-33
TIM	53	v,b	$(B^b) \rightarrow v_{00-14}$	5-35
TMA	53	v	$(v) \rightarrow A$	5-34

TABLE 2. ALPHAMNEMONIC LISTING OF INSTRUCTIONS (Cont'd)

Mnemonic Code	Basic Octal Code	Address Field	Instruction Description	Page No.
TMAV	77		Initiate memory request. If reply occurs within 5 usec, RNI at P + 2; if not, RNI at P + 1. Storage address is (B^2) with (OSR) or zero appended. TEST MEMORY AVAILABILITY	5-77
TMI	53		$(v_{00-14}) \rightarrow B^b$	5-35
TMQ	53	v	(v) → Q	5-34
TQM	53	v	(Q) → v	5-34
TST	67	r, B_r, $\S_1$	Test fld A, +, -, or 0	5-165
TSTN	67	r, B_r, $\S_1$	Test fld A for numeric	5-166
UCS	77		Unconditional stop. Upon restarting RNI @ P + 1	5-24
U JP, I	01	m, b	Unconditional jump to M.	5-41
UPAK	66	m, B_m, s, B_s, $\S_2$	Unpack 4-bit BCD fld A into 6-bit BCD fld C	5-153
XOA	16	y	y v (A) → A,	5- 69
XOA,S	16	y	y v (A) → A, sign of y extended	5- 69
XOI	16	y,b	y v $(B^b) \rightarrow B^b$	5- 69
XOQ	16	y	y v (Q) → Q	5- 69
XOQ,S	16	y	y v (Q) → Q, sign of y extended	5- 69
ZADM	67	r, B_r, $\S_1$, s, B_s, $\S_2$	Clear fld C; fld A → fld C, right justify	5- 129

INDEX

COMMENT SHEET

MANUAL TITLE CONTROL DATA 3300 COMPUTER SYSTEM

Reference Manual

PUBLICATION NO. 60157000 REVISION K

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 U.S.A

AA 3419

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

FOLD ON DOTTED LINES AND STAPLE

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

QUICK REFERENCE INSTRUCTION INDEX

ADDITIONAL INSTRUCTION INFORMATION

CONTROL DATA
CORPORATION

CORPORATE HEADQUARTERS, 8100 34th AVE. SO., MINNEAPOLIS, MINN. 55440
SALES OFFICES AND SERVICE CENTERS IN MAJOR CITIES THROUGHOUT THE WORLD

LITHO IN U.S.A.