SPECTRA 70

RADIO CORPORATION OF AMERICA • ELECTRONIC DATA PROCESSING

PROCESSOR

70/25

REFERENCE MANUAL

70/25 PROCESSOR REFERENCE MANUAL

SPECTRA 70

RADIO CORPORATION OF AMERICA · ELECTRONIC DATA PROCESSING

PROCESSOR

7025

REFERENCE MANUAL

RADIO CORPORATION OF AMERICA

70 - 25 - 601

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.

First Printing: March, 1965

RCA 70/25 System

CONTENTS

Page

APPENDICES

SYSTEM DESCRIPTION

INTRODUCTION

♦ The RCA Model 70/25 Processor, second member of the Spectra 70 Series, is a small-to-medium scale computer designed to satisfy a wide variety of data processing requirements. The 70/25 is organized as a powerful data processor with the capability to concurrently perform up to 16 input/output operations in addition to its compute operations. This simultaneity is achieved by including the following in a 70/25 configuration:

1. *Eight selector channels (two may be high speed)* — Each selector channel controls one-device subsystem (from 1 to 16 devices). One device may be operating at one time on each selector channel.
2. *Multiplexor channel* — Up to 115 devices may be connected to the multiplexor channel and up to eight of these devices may operate simultaneously.

The above input/output simultaneity coupled with its communications capabilities make the 70/25 a highly efficient vehicle for high-speed remote processing.

The 70/25 is designed not only to support a large complex of systems (multisystems operation) but also to stand by itself as a small-to-medium size data processor.

To fulfill the growth requirements of the user, provision has been made for program compatibility between the Model 70/25 Processor and the larger processors in the Spectra 70 Series.

ORGANIZATION OF DATA

♦ The following definitions describe the various levels of data organization for the 70/25 Processor:

Bit:	is a single binary digit having the value of either zero or one.
Byte:	consists of eight information bits and a parity bit. It represents two decimal digits, one alphabetic character or one special symbol.
Halfword:	consists of two consecutive bytes beginning on a high-speed memory location that is a multiple of two.
Word:	consists of four consecutive bytes beginning on a high-speed memory location that is a multiple of four.
Double word:	consists of eight consecutive bytes beginning on a high-speed memory location that is a multiple of eight.
Item/Field:	consists of any number of bytes that specify a particular unit of information (numeric field, alphabetic name, street address, stock number, etc.).
Record:	consists of one or more related items.

DATA FORMATS

♦ The basic unit of information in the 70/25 Processor is a *byte*. A byte consists of eight information bits and one parity bit. The parity bit ensures the accuracy of all bytes accessed by the processor. The byte is the smallest addressable unit in the 70/25. It represents one alphanumeric character, two decimal digits, or eight binary digits.

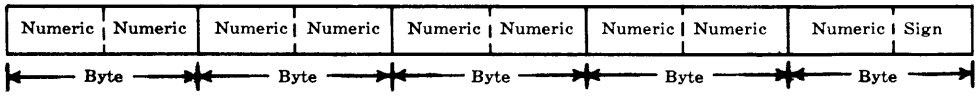
DATA FORMATS (Cont'd)

The internal code representation in the 70/25 Processor is the *E*xtended *B*inary-*C*oded-*D*ecimal *I*nterchange *C*ode (EBCDIC). Appendix E contains a complete listing of each 70/25 code with its corresponding printed symbol and bit configuration.

The formats for data in high-speed memory are *packed-decimal and zoned.*

♦ In packed-decimal format, one byte represents two numeric digits. All decimal numerics must be packed because all decimal arithmetic functions operate on this format. The numerals, zero (0) through nine (9), are coded $(0000)_2$ through $(1001)_2$ and are the only legitimate digits in packed-decimal format.

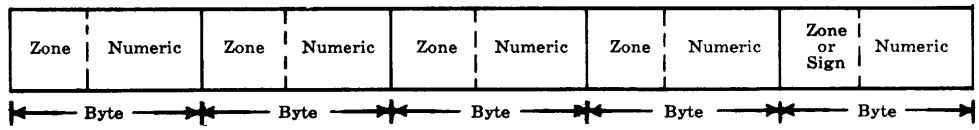
Packed-Decimal Format



The rightmost half-byte (4 bits) of a field represents the sign. The only EBCDIC code that is machine-generated to represent the plus (+) sign of a positive field is $(1100)_2$. The code $(1101)_2$ is the only EBCDIC code that is machine-generated to represent the minus (−) sign of a negative field. It should be noted, however, that the codes $(1010)_2$, $(1110)_2$, and $(1111)_2$ representing plus (+) signs and the code $(1011)_2$ representing a minus (−) sign are accepted by the machine. (This variety permits the processor to handle other than EBCDIC code.) Nevertheless, if an arithmetic operation is performed on a field, the sign of the result will be in EBCDIC code.

Zoned Format

♦ In zoned format, one byte represents one alphanumeric digit. Alphanumeric data *must* be in zoned format. The right half-byte (4 bits) is the number and the left half-byte (4 bits) is the zone.



When changing from packed decimal format to zoned format, the code $(1111)_2$ is generated in the zone portion of a field. The zone portion of the rightmost byte of a numeric field is the sign of the field.

RCA 70/25 PROCESSOR

INTRODUCTION

♦ The RCA Model 70/25 Processor is a word-organized, variable-address, digital computer consisting of *high-speed memory, program control,* and *input/output control.*

HIGH-SPEED MEMORY

♦ High-speed memory consists of planes of magnetic cores. These planes are 64 x 64 strings; each string is four bytes in depth resulting in a basic block of 16,384 bytes of separately addressable core memory. High-speed memory is field-expandable from 16,384 bytes to 32,768 bytes or to 65,536 bytes.

One byte is the smallest addressable unit in the 70/25 Processor. Memory cycle time is 1.5 microseconds which is the time required to transfer a four-byte word from the 70/25 memory to the memory register and to regenerate the word in storage. Data is manipulated in high-speed memory one byte at a time with the exception of the Move instruction. The Move instruction can operate on four bytes with one memory access.

Each byte in high-speed memory is binarily addressed. Sixteen-bit addresses permit accessing up to 65,536 bytes. Memory wrap-around occurs at 16,384 bytes; 32,768 bytes or 65,536 bytes depending on the size of high-speed memory.

Since binary addresses are cumbersome to work with, the hexadecimal numbering system has been adopted to represent characters and addresses in the 70/25 Processor. The hexadecimal system has a base of 16. The first ten marks are represented by decimal numbers zero (0) through nine (9); marks eleven through sixteen are represented by the letters A through F.

The basic hexadecimal marking system and its binary and decimal equivalent are specified in table 1.

Table 1. Basic Hexadecimal Marking System

Hexadecimal (Base 16)	Binary (Base 2)	Decimal (Base 10)
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

HIGH-SPEED MEMORY (Cont'd)

The first 50 bytes and the last 100 bytes of high-speed memory are reserved for use by the processor. The first 50 bytes serve as registers and intermediate storage areas during the handling of input/output operations and interrupt mechanization. The last 100 bytes are used for hardware utility registers, a Timer register, and 15 General-Purpose registers. If a multiplexor channel is included in the system, high-speed memory immediately preceding the last 100 bytes (eight bytes for each device on the multiplexor) must also be reserved. Specific memory allocations are defined on pages 15 through 18.

THE BASIC INSTRUCTION

♦ The RCA Model 70/25 Processor includes a standard set of 32 instructions. This instruction repertory performs arithmetic, data handling, decision, control, and input/output operations.

All instructions must start on halfword boundaries (even-numbered byte locations. The basic format of the instruction is two, four or six bytes long. Twenty-two instructions have a two-address format and are six bytes in length. Nine single-address instructions are in the set and are four bytes long. One instruction is two bytes long. Indirect addressing is not provided. Table 2 illustrates these instruction formats.

Table 2. Instruction Formats

Halfword			Halfword		Halfword	
Byte 1	Byte 2		Byte 3	Byte 4	Byte 5	Byte 6
8 OP	8 L		4 B_1	12 D_1	4 B_2	12 D_2
8 OP	4 L_1	4 L_2	4 B_1	12 D_1	4 B_2	12 D_2
8 OP	4 T	4 U	4 B_1	12 D_1	4 B_2	12 D_2
8 OP	8 M		4 B_2	12 D_2		
8 OP	4 T	4 U	4 B_2	12 D_2		
8 OP	4 R_1	4 R_3	4 B_2	12 D_2		
8 OP	4 R_1	4 R_2				

Legend:

OP — operation code.
B_1 — register containing base address of first operand.
B_2 — register containing base address of second operand.
D_1 — address of leftmost byte of the displacement component of the first operand.
D_2 — address of leftmost byte of the displacement component of the second operand.
L — one less than the length of the first and/or second operand.
L_1 — one less than the length of the first operand.
L_2 — one less than the length of the second operand.
M — mask for Branch On Condition, Test Under Mask, and Halt and Branch.
T — input/output trunk referenced.
U — input/output device referenced.

THE BASIC INSTRUCTION (Cont'd)

Legend: (Cont'd)

R_1 — specifies the first general register to be loaded or stored.

R_2 — specifies the register containing the branch address for Branch and Link (BALR).

R_3 — specifies the last general register to be loaded or stored.

The operand addresses are generated from two binary numbers. The base address is a 16-bit binary number held in the Base Address register specified by the B field (B_1/B_2) of the instruction. The displacement is a 12-bit number contained in the D field (D_1/D_2) of the instruction. This displacement provides for relative addressing up to 4,095 bytes beyond the base address. In forming the address, the displacement and the contents of the Base Address register are added together as absolute binary integers. The Base Address registers and their specified B fields are defined in table 3.

Table 3. Base Address Registers

Register	B Field
1	$(0001)_2$
2	$(0010)_2$
3	$(0011)_2$
4	$(0100)_2$
5	$(0101)_2$
6	$(0110)_2$
7	$(0111)_2$
8	$(1000)_2$
9	$(1001)_2$
10	$(1010)_2$
11	$(1011)_2$
12	$(1100)_2$
13	$(1101)_2$
14	$(1110)_2$
15	$(1111)_2$

If $(0000)_2$ appears in the B field of an instruction, it indicates that no base address is to be used; it has no relationship with register zero. Register zero on the 70/25 is the Timer register.

PROGRAM CONTROL

♦ The function of the program control unit in the Model 70/25 Processor is to interpret and to execute the instructions stored in high-speed memory. The program control unit provides the necessary registers and indicators to monitor sequence of operations, to perform automatic accuracy checks, and to communicate with the RCA standard interface in the control of input/output devices.

The program control not only executes each instruction but also takes each instruction from high-speed memory and places it in the proper registers. This process is called staticizing. The total time to staticize an instruction is as follows:

Two-byte instruction — 4.5 microseconds
Four-byte instruction — 9 microseconds
Six-byte instruction — 13.5 microseconds

Figure 1 shows the interrelationship of the registers and indicators and depicts the flow of information through the processor. Table 4 contains a brief description of the functions of these registers and indicators.

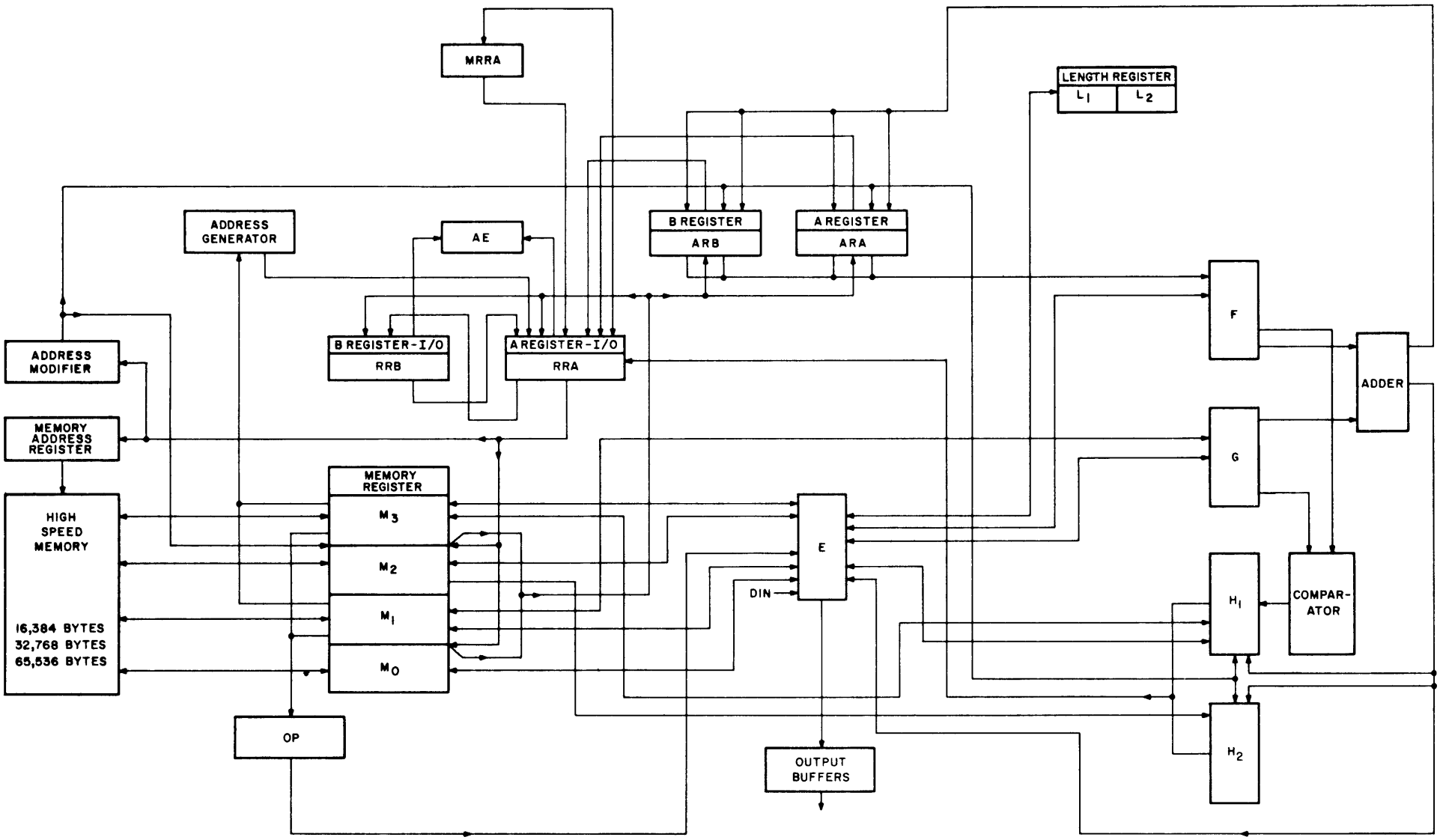


Figure 1. RCA 70/25 Processor Schematic Diagram

PROGRAM CONTROL (Cont'd)

Table 4. Functions of Registers and Indicators (See figure 1.)

Register/Indicator	Function
Address Generator	Two-byte register that produces the address of the program counter for either the *Processing State* (P_1) or the *Interrupt State* (P_2). The address generator also generates the address of the Base Address registers, as specified by the instruction being executed, and the reserved high-speed memory addresses if an input/output operation is being executed.
Memory Address Register (MAR)	Two-byte register that holds the address of the high-speed memory location to be processed.
Memory Register (M)	Four-byte register (M_0, M_1, M_2, M_3) that contains the byte(s) read from, or to be written to, high-speed memory.
Address Modifier	Increments or decrements the contents of the Memory Address Register.
Operation Register (OP)	One-byte register that holds the operation code of the instruction being processed.
A Register (ARA)	Two-byte register that holds the sum of the contents of the general register addressed by the B_1 field and the D_1 address of the instruction.
B Register (ARB)	Two-byte register that holds the sum of the contents of the general register addressed by the B_2 field and the D_2 address of the instruction.
A Register I/O (RRA)	Two-byte register that holds the sum of the contents of the general register addressed by the B_1 field and the D_1 address of an input/output instruction. This register is also used as an address interchange register and as an address storage register for the address modifier.
B Register I/O (RRB)	Two-byte register that holds the sum of the contents of the general register addressed by the B_2 field and the D_2 address of an input/output instruction.
AE	Address equality circuit for input/output operations.
MRRA Register	Two-byte register that stores the current A address of a multiplexor operation when it must wait because of input/output priority.
Adder	Used in the addition of the contents of the F and G registers.
Comparator	Used in the comparison of the contents of the F and G registers.
Output Buffers	Used as storage for data being transferred from high-speed memory to peripheral devices.
Length Register	One-byte register that holds L_1 and L_2, M, L, or R of the instruction being processed.
F Register	One-byte register that is used as temporary storage for data. It is also used to feed the adder or comparator.

PROGRAM CONTROL (Cont'd)

Table 4. Functions of Registers and Indicators (Cont'd)

Register/Indicator	Function
G Register	One-byte register that is used as temporary storage for data. It is also used to feed the adder or comparator.
H_1 Register	One-byte register used as temporary storage for data or for the most significant byte of an address.
H_2 Register	One-byte register used as temporary storage for data or for the least significant byte of an address.
E Register	One-byte register used as an interchange to send data to or receive data from other registers.
Condition Code Indicators	Used in conjunction with the adder/comparator to indicate positive, negative, zero, or overflow results. On occurrence of an interrupt, the condition code indicators also specify the type of interrupt that occurred. In conjunction with input/output instructions, they indicate if the instruction was successful. *Note:* The condition code is the only hardware register or indicator that the programmer can test. The condition code indicators can be sensed by a Branch on Condition instruction. The condition code indicators are stored in reserved high-speed memory only when an interrupt occurs.

INTERRUPT MECHANIZATION

♦ The 70/25 Processor has two distinct *processor states,* each having its own program counter. They provide fast interrupt servicing and facilitate program control. The processor states and their functions are as follows:

Processing State (P_1) — is the state in which the user's program is executed. This state is capable of being interrupted. Once interrupted, conditions existing at the time of interrupt are automatically stored and control is then transferred to the *Interrupt State (P_2).*

Interrupt State (P_2) — is the state in which a program analysis of the interrupt is made. Control is then transferred back to the *Processing State (P_1)* where the interrupt is serviced. The P_2 state cannot be interrupted.

Program interruption capabilities are provided in the 70/25 Processor as follows:

1. *External request or termination interrupt from an input-output device* — normal processing is interrupted upon request from an interrogating typewriter, a communications control, or a data exchange control. These requests are to process remote inquiries or data transmissions. An interrupt also occurs upon termination of an input/output operation.
2. *Internal operation code trap* — an interrupt occurs when an undefined operation code in the 70/25 instruction set is recognized. Operations not included in the 70/25 instruction set can be simulated by this feature.

INTERRUPT MECHANIZATION (Cont'd)

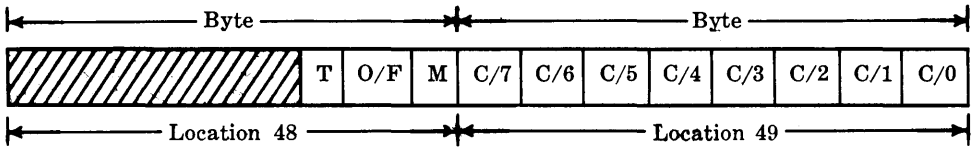
3. *Arithmetic overflow or divide exception* — an interrupt occurs on all arithmetic overflow conditions. The conditions that cause a divide exception interrupt are defined under the Divide instruction. (See page 40.)

4. *Timer interrupt* — the timer may be set or altered to provide for interruption of normal processing when an overflow from bit-position 2^{23} in the Timer register occurs.

When an interrupt signal is received by the processor, the interrupt indicator is set. The interrupt takes place as soon as the current instruction terminates. Figure 2 shows the sequence of events when the interrupt occurs.

EXPOSITORY NOTES: (See figure 2.)

Block 1 — The interrupt mask in reserved high-speed memory locations 48 and 49 is checked against the hardware interrupt indicator to determine if the interrupt is permitted. The bit significance of locations 48 and 49 is as follows:



Location 49 — C/0 through C/7 = Channel zero through Channel seven

Location 48 — M = Multiplexor Channel
O/F = Arithmetic overflow or divide exception
T = Timer

High-speed memory location 49 corresponds to the eight input/output channels. An input/output channel interrupt occurs upon request from an interrogating typewriter, a communications control, or a data exchange control if any of these devices are connected to a channel. An interrupt also occurs upon termination of an input/output device connected to a channel. If the mask bit is a zero for the channel requesting the interrupt, the interrupt remains pending and the program continues processing. For example, the bit configuration $(11101111)_2$ in location 49 prohibits any interrupts by channel four (C/4). If the mask bit is a one, the interrupt is taken. If the interrupt is prohibited, it remains pending until the interrupt is taken or a Post Status instruction is issued to the trunk and device.

The rightmost three bits of location 48 correspond to the timer interrupt, to the arithmetic overflow/divide exception interrupt, and to the multiplexor channel interrupt.

Timer — A timer interrupt is caused by an overflow from bit-position 2^{23} in the Timer register (general register zero). If the timer interrupt is prohibited (mask = 0) the interrupt remains pending until it is permitted (mask = 1) or until the Timer register is accessed by a Load Multiple instruction. The Timer register continues to increment even though the timer interrupt is prohibited.

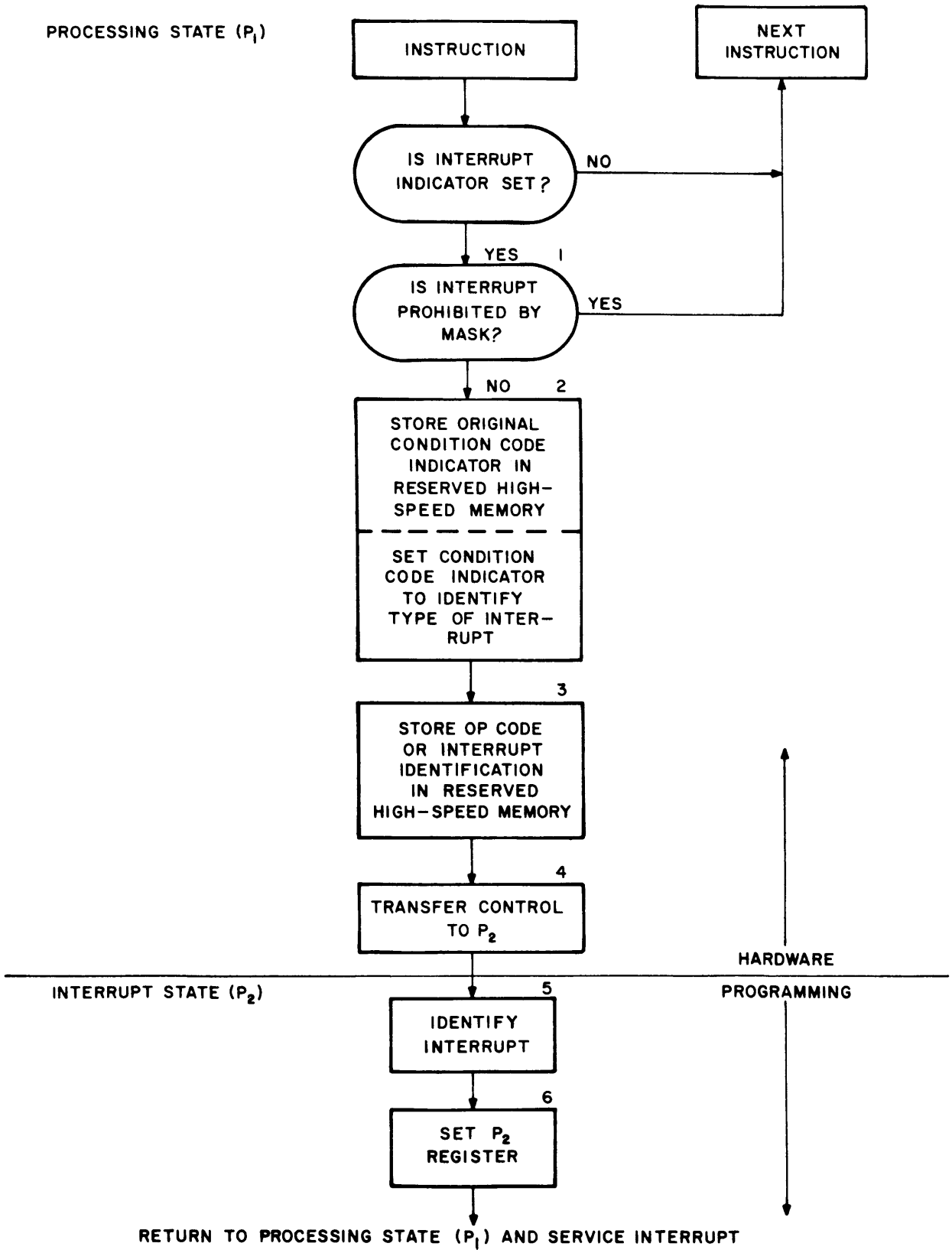


Figure 2. Interrupt Mechanization

EXPOSITORY NOTES (See figure 2.) (Cont'd)

Block 1 — Arithmetic overflow/divide exception (Cont'd) — An arithmetic overflow/divide exception interrupt can occur as a result of an Add Decimal, Subtract Decimal, Add Binary, or Divide instruction. If this interrupt condition occurs and the mask bit indicates that it is permitted (mask bit = 1), the operation code is stored in reserved high-speed memory location 42. This permits the program to differentiate between an *arithmetic overflow interrupt* and a *divide exception interrupt*. If this interrupt condition occurs and the mask bit indicates that it is prohibited (mask bit = 0), the interrupt condition is reset. It does *not* remain pending. If this interrupt condition occurs in the *Interrupt State (P_2)*, the interrupt is not taken. Nevertheless, an arithmetic overflow in the P_2 state can be determined by testing the condition code indicators.

Multiplexor — A multiplexor interrupt occurs upon request from an interrogating typewriter, a communications control, or a data exchange control if they are attached to the multiplexor channel. An interrupt also occurs upon termination of an input/output operation for a device connected to the multiplexor. If the interrupt is prohibited (mask = 0), it remains pending until the interrupt is taken or a Post Status instruction is issued to the trunk and device.

Important: Operation code trap interrupts can not be masked and must be taken.

Block 2 — If the interrupt is taken, the condition code setting (at the time of interrupt) is stored in reserved high-speed memory location 43. The condition code indicator is then set to indicate the cause of the interrupt as follows:

0 — external device request or termination

1 — operation code trap

2 — arithmetic overflow or divide exception

3 — timer

Block 3 — If the interrupt is an operation code trap or an arithmetic overflow/divide exception, the operation code causing the interrupt is stored in reserved high-speed memory location 42.

If an external device interrupt (request or termination) occurs, the standard device byte, trunk number, and device number of the interrupting device are stored in reserved high-speed memory locations 46 and 47.

Block 4 — Control is transferred to the instruction address located in the program counter (reserved high-speed memory locations 40 and 41) for the *Interrupt State (P_2)*. No further interrupts can take place until control is transferred back to the *Processing State (P_1)*. If an operation code trap occurs in P_2, the processor comes to an orderly halt (i.e., all input/output operations in progress go to completion before the computer halts).

EXPOSITORY NOTES (See figure 2.) (Cont'd)

Block 5 — Programming in the *Interrupt State (P_2)* analyzes the interrupt to determine the action to be taken. The condition code, operation code, and interrupt identification that were stored by hardware in reserved high-speed memory provide the interrupt information.

Block 6 — When the interrupt has been identified, the program in the *Interrupt State (P_2)* must execute a Set P_2 Register instruction to return to the *Processing State (P_1)*. The Set P_2 Register instruction:

1. restores the condition code indicator (as it appears in reserved high-speed memory location 43).
2. sets the program counter for the *Interrupt State (P_2)* (reserved high-speed memory locations 44 and 45).
3. transfers control back to the *Processing State (P_1)* to the address specified by the P_1 program counter (reserved high-speed memory locations 40 and 41).

The interrupt priority in the 70/25 is as follows:

1. Operation code trap.
2. Arithmetic overflow or divide exception.
3. Timer.
4. External device request or termination — The priority for input/output interrupts is (a) high-speed selector channel (b) selector channel (c) multiplexor channel. (The devices on the multiplexor channel have a priority depending upon the device number. The lower the device number, the higher the priority.)

INPUT/OUTPUT CONTROL

♦ The 70/25 Processor communicates with all input/output devices through the RCA standard interface.

Selector Channels

♦ The 70/25 can have a total of either eight selector channels *or* six selector channels and two high-speed selector channels. If high-speed selector channels are included in the system, they must be designated as 0 and 1. Each selector channel contains one standard interface trunk which, in turn, controls one device subsystem (from 1 to 16 devices). Since each selector channel has its own set of registers, all may operate simultaneously.

Multiplexor Channel

♦ In addition to the selector channels, a multiplexor channel may be included in a 70/25 system. The multiplexor channel contains eight standard interface trunks; each trunk controls one device subsystem (from one to 16 devices). A maximum of 115 devices may be connected to the multiplexor channel. Each device has its own set of registers in reserved high-speed memory. Up to eight-way simultaneity can be achieved on the multiplexor channel.

Input/Output Operation

♦ An input/output operation takes place as follows:

1. The input/output instruction is staticized and, if the addressed device is available, the operation proceeds.

Input/Output Operation (Cont'd)

2. Upon completion of the instruction, the final D_1 address plus one (minus one if the operation was read reverse) is stored in the D_1 final location for the particular channel in reserved high-speed memory and a termination interrupt occurs. If the termination interrupt is permitted, the trunk number, device number, and standard device byte are automatically stored in reserved high-speed memory locations 46 and 47. If the termination interrupt is prohibited by the mask, it remains pending until the interrupt is permitted or a Post Status instruction is issued to the trunk and device.

The following information can be used by the program to determine the status of an input/output instruction after it has been attempted:

Condition Code

♦ Condition codes can be sensed by the program to indicate whether or not the attempted input/output instruction had been successful. The condition code settings are listed below in table 5.

Table 5. Condition Code Settings

Condition Code	Description
0	Instruction was accepted.
1	Device is inoperable.
2	Interrupt is pending.

Notes:

1. *Condition code 0,* indicates that the device addressed was available and that the instruction was accepted.
2. *Condition code 1,* indicates that the device was inoperable or the channel and/or device addressed was invalid. The attempted input/output instruction is bypassed and the next instruction in sequence is staticized.
3. *Condition code 2,* indicates that an interrupt (external device request or termination) is pending on the channel addressed. The attempted input/output instruction is bypassed and the next instruction in sequence is staticized. An input/output instruction can not be executed to this channel until the interrupt has been serviced, or a Post Status instruction is executed which resets the interrupt pending condition.

Standard Device Byte

♦ The standard device byte indicates the status of the device following an input/output instruction. It is placed into reserved high-speed memory when:

1. An input/output operation is terminated and a termination interrupt is permitted. The standard device byte is automatically stored in reserved high-speed memory location 46.
2. A Post Status instruction is executed. The standard device byte for the device referenced is placed into reserved high-speed memory for the selector channel or multiplexor channel addressed.

The standard device byte is defined in table 6.

Standard Device Byte (Cont'd)

Table 6. Standard Device Byte

"1" bit in	Description
2^0	Not applicable
2^1	Device inoperable
2^2	Secondary indicator
2^3	Device end
2^4	Not applicable
2^5	Not applicable
2^6	Termination interrupt pending
2^7	External device request interrupt pending

Notes:

1. *Device inoperable bit* is set when the device referenced is inoperable.
2. *Secondary indicator bit* is set when the device referenced has additional indicators to be tested. These indicators can be brought into high-speed memory by using the Sense instruction.
3. *Device end bit* is set when the device referenced has terminated and may accept another operation.
4. *Termination interrupt pending bit* is set when an input/output termination condition exists in the device referenced.
5. *External device request interrupt pending bit* is set when an interrogating typewriter, a data exchange control, or a communications control requires servicing.

Sense Byte

♦ The sense byte can be brought into high-speed memory from the particular device referenced by using the Sense instruction. It contains status information for the device referenced. (The exact status information sent is defined in the Spectra 70 input/output supplementary publications for the individual units.)

Notes

♦ If an illegal input/output instruction is attempted (i.e., rewind the printer or write to the card reader), the condition code indicates that the instruction is accepted. However, the secondary indicator bit in the standard device byte is set and the sense byte indicates an illegal operation.

If an input/output instruction is attempted to a device that is busy, the instruction is restaticized until the device becomes available. The program is interruptable at the beginning of each restaticizing.

If a Post Status instruction is executed and the referenced device is busy, control is transferred to the address specified in the D_2 portion of the instruction.

RESERVED HIGH SPEED MEMORY

Lower Memory

♦ The first 50 bytes of high-speed memory are reserved for use by the processor. They serve as registers and intermediate storage areas during the handling of input/output operations and interrupt mechanization. The specific lower-memory allocations are specified in table 7.

Table 7. Lower-Memory Allocations

Location	Description
0-31	Channel status indicators — indicate the status of a channel following an input/output operation. Each of the eight input/output channels requires four bytes of high-speed memory for the storage of information: *Byte* — *Channel* 0- 3 — 0 4- 7 — 1 8-11 — 2 12-15 — 3 16-19 — 4 20-23 — 5 24-27 — 6 28-31 — 7 The four bytes per trunk are assigned as follows: Bytes 1, 2 — D_1 final register contents at input/output termination. Byte 3 — Standard device byte (placed in this location when a Post Status instruction is executed). Byte 4 — Reserved for future enhancement.
32-39	Reserved for use by the processor and can not be used by programming.
40-41	Program counter for *Processing State* (P_1) — contains the address of the next instruction to be executed in the *Processing State* (P_1).
42	Operation code — contains the operation code and length of the last instruction interrupted when an operation code trap or an arithmetic overflow/divide exception interrupt occurs. The length of the instruction is indicated in the two high-order bits of byte 42 as follows: $(00)_2$ = two-byte instruction $(01)_2$ or $(10)_2$ = four-byte instruction $(11)_2$ = six-byte instruction The six low-order bits contain the remainder of the operation code.
43	Condition Code — contains the condition code indicator (in the two low-order bits) for the *Processing State* (P_1) when an interrupt occurs. This code ranges from $(00)_2$ to $(11)_2$.
44-45	Program counter for *Interrupt State* (P_2) — contains the address of the next instruction to be executed in the *Interrupt State* (P_2).
46-47	Interrupt identification — contains information concerning an interrupt as follows: 46 = the standard device byte of the interrupting device. 47 = the trunk and device causing the interrupt.

RESERVED HIGH SPEED MEMORY
Lower Memory (Cont'd)

Table 7. Lower-Memory Allocations (Cont'd)

Location	Description
48-49	Interrupt mask — permits or inhibits an interrupt. The bit significance of locations 48 and 49 is as follows: ← Byte → ← Byte → ▨ T O/F M C/7 C/6 C/5 C/4 C/3 C/2 C/1 C/0 ← Location 48 → ← Location 49 → Location 49 — C/0 through C/7 = Channel zero through Channel seven. Location 48 — M = Multiplexor. O/F = Arithmetic overflow or divide exception. T = Timer. If a mask bit is zero, the specified interrupt is inhibited. The interrupt remains pending unless it is an arithmetic overflow/divide exception. (See page 11.) If a mask bit is one, the interrupt is taken.

The complete map of the first 50 high-speed memory locations (lower-memory) is shown in table 8.

Table 8. Reserved High-Speed Memory Layout (Lower-Memory)

Location Decimal	Location Hexadecimal	Byte	Byte	Byte	Byte
0000-0003	0000-0003	D_1 Final for	16 Channel/Trunk 0	8 Standard Device Byte	8 Reserved
0004-0007	0004-0007	D_1 Final for	16 Channel/Trunk 1	8 Standard Device Byte	8 Reserved
0008-0011	0008-000B	D_1 Final for	16 Channel/Trunk 2	8 Standard Device Byte	8 Reserved
0012-0015	000C-000F	D_1 Final for	16 Channel/Trunk 3	8 Standard Device Byte	8 Reserved
0016-0019	0010-0013	D_1 Final for	16 Channel/Trunk 4	8 Standard Device Byte	8 Reserved
0020-0023	0014-0017	D_1 Final for	16 Channel/Trunk 5	8 Standard Device Byte	8 Reserved
0024-0027	0018-001B	D_1 Final for	16 Channel/Trunk 6	8 Standard Device Byte	8 Reserved
0028-0031	001C-001F	D_1 Final for	16 Channel/Trunk 7	8 Standard Device Byte	8 Reserved
0032-0039	0020-0027	Reserved For Hardware Use Only.			
0040-0043	0028-002B	(P_1) Program Counter 16		8 Operation Code	▨ 6 / 2 CC
0044-0047	002C-002F	(P_2) Program Counter 16		8 Interrupt Standard Device Byte	4 Interrupt Trunk No. / 4 Interrupt Device No.
0048-0049	0030-0031	▨ 5 / 1 T / 1 O/F / 1 M		8 Interrupt Mask for Channel 0-7	

▨ — These bit positions are not used and must be set to zeros.

NOTE: Numbers in upper right-corner of blocks indicate the number of bits used.

RESERVED HIGH SPEED MEMORY (Cont'd)

Upper Memory

♦ The last 100 bytes of high-speed memory (regardless of memory size) are reserved for use by the hardware and cannot be used by the programmer. The specific upper-memory allocations are specified in table 9.

Table 9. Upper-Memory Allocations

Bytes (Counting from the last HSM location downward)	Description
99-40	Fifteen general-purpose registers — each general-purpose register uses 4 bytes of reserved high-speed memory. The low-order two bytes (16 bits) are used as Base Address registers. The value contained in the register is used by the instruction address to form the high-speed memory location address. The general-purpose registers are also used as operands in the Branch and Link, Branch on Count, Load Multiple, and Store Multiple instructions.
39-36	Timer register — A Timer register is provided on the 70/25 as a standard feature. It occupies general register zero and uses the low-order 24 bits. A one is added to the low-order bit of the Timer register either 50 (50-cycle power) or 60 (60-cycle power) times per second. A timer interrupt occurs when overflow takes places in the 24th bit position (2^{23}) of the Timer register. The value of the timer may be obtained or altered at any time by using the Store Multiple or Load Multiple instructions. Using 60-cycle power, the interrupt interval may be varied from 17 milliseconds to 77⅔ hours. A computer halt prevents the timer from incrementing.
35- 0	Reserved for use by the processor, and can not be used by programming.

If a multiplexor channel is included in the system, the area immediately preceding the last 100 reserved bytes of upper high-speed memory must also be reserved. Each device connected to the multiplexor channel requires eight bytes of reserved high-speed memory. The addressing scheme used for the devices on the multiplexor channel requires that the trunk and device number be complemented. This technique places the lower-numbered devices in the higher-numbered high-speed memory locations. The first eight-byte group of memory available is for device 13 located 104 bytes from the top of memory (the first 100 bytes are reserved as shown above). Consequently, device numbers 13-127 *only* may be connected to the multiplexor channel providing a total of 115 devices. The eight bytes per multiplexor device are assigned as follows:

Bytes 1, 2 — D_1 final register contents at input/output termination.
Bytes 3, 4 — D_2 address of input/output instruction.
Byte 5 — Operation code of input/output instruction.
Byte 6 — Standard device byte (placed in this location when a Post Status instruction is executed).
Bytes 7, 8 — Not used and must be zeros.

RESERVED HIGH SPEED MEMORY
Upper Memory (Cont'd)

A complete map of the upper high-speed memory reserved locations is shown in table 10. (The locations are shown in decimal and hexadecimal for each high-speed memory size.)

Table 10. Reserved High-Speed Memory Layout (Upper-Memory)

Byte	Byte	Byte	Byte	Byte	Byte	Byte	Byte
GENERAL REGISTER #14 (4 Bytes) 16,376–16,379 3FF8–3FFB 32,760–32,763 7FF8–7FFB 65,528–65,531 FFF8–FFFB				GENERAL REGISTER #15 (4 Bytes) 16,380–16,383 3FFC–3FFF 32,764–32,767 7FFC–7FFF 65,532–65,535 FFFC–FFFF			
GENERAL REGISTER #12 (4 Bytes) 16,368–16,371 3FF0–3FF3 32,752–32,755 7FF0–7FF3 65,520–65,523 FFF0–FFF3				GENERAL REGISTER #13 (4 Bytes) 16,372–16,375 3FF4–3FF7 32,756–32,759 7FF4–7FF7 65,524–65,527 FFF4–FFF7			
GENERAL REGISTER #10 (4 Bytes) 16,360–16,363 3FE8–3FEB 32,744–32,747 7FE8–7FEB 65,512–65,515 FFE8–FFEB				GENERAL REGISTER #11 (4 Bytes) 16,364–16,367 3FEC–3FEF 32,748–32,751 7FEC–7FEF 65,516–65,519 FFEC–FFEF			
GENERAL REGISTER #8 (4 Bytes) 16,352–16,355 3FE0–3FE3 32,736–32,739 7FE0–7FE3 65,504–65,507 FFE0–FFE3				GENERAL REGISTER #9 (4 Bytes) 16,356–16,359 3FE4–3FE7 32,740–32,743 7FE4–7FE7 65,508–65,511 FFE4–FFE7			
GENERAL REGISTER #6 (4 Bytes) 16,344–16,347 3FD8–3FDB 32,728–32,731 7FD8–7FDB 65,496–65,499 FFD8–FFDB				GENERAL REGISTER #7 (4 Bytes) 16,348–16,351 3FDC–3FDF 32,732–32,735 7FDC–7FDF 65,500–65,503 FFDC–FFDF			
GENERAL REGISTER #4 (4 Bytes) 16,336–16,339 3FD0–3FD3 32,720–32,723 7FD0–7FD3 65,488–65,491 FFD0–FFD3				GENERAL REGISTER #5 (4 Bytes) 16,340–16,343 3FD4–3FD7 32,724–32,727 7FD4–7FD7 65,492–65,495 FFD4–FFD7			
GENERAL REGISTER #2 (4 Bytes) 16,328–16,331 3FC8–3FCB 32,712–32,715 7FC8–7FCB 65,480–65,483 FFC8–FFCB				GENERAL REGISTER #3 (4 Bytes) 16,332–16,335 3FCC–3FCF 32,716–32,719 7FCC–7FCF 65,484–65,487 FFCC–FFCF			
TIMER REGISTER (4 Bytes) 16,320–16,323 3FC0–3FC3 32,704–32,707 7FC0–7FC3 65,472–65,475 FFC0–FFC3				GENERAL REGISTER #1 (4 Bytes) 16,324–16,327 3FC4–3FC7 32,708–32,711 7FC4–7FC7 65,476–65,479 FFC4–FFC7			
RESERVED FOR HARDWARE USE (40 Bytes) 16,280–16,319 3F98–3FBF 32,664–32,703 7F98–7FBF 65,432–65,471 FF98–FFBF							
Byte	Byte	Byte	Byte	Byte	Byte	Byte	Byte
MULTIPLEXOR DEVICE #13 (8 Bytes)							
D_1 Final		D_2 Address		OP Code	Stand Device	Not Used	
16,272–16,279 3F90–3F97 32,656–32,663 7F90–7F97 65,424–65,431 FF90–FF97							
904 BYTES FOR MULTIPLEXOR DEVICES #14 → #126							
MULTIPLEXOR DEVICE #127 (8 Bytes)							
D_1 Final		D_2 Address		OP Code	Stand Device	Not Used	
15,360–15,367 3C00–3C07 31,744–31,751 7C00–7C07 64,512–64,519 FC00–FC07							

RCA 70/25 INSTRUCTIONS

General

♦ The RCA 70/25 Processor contains a standard set of 32 instructions. These instructions may be classified into four general categories:

Data Handling Instructions

♦ The data handling instructions consist of five non-arithmetic instructions that manipulate data stored in high-speed memory. These instructions are:

Move (MVC)
Edit (ED)
Pack (PACK)
Unpack (UNPK)
Translate (TR)

Arithmetic and Logical Instructions

♦ The arithmetic and logical instructions consist of four decimal instructions and five logical instructions that permit bit manipulation and address modification. These instructions are:

Add Decimal (AP)
Add Binary (AB)
Subtract Decimal (SP)
Subtract Binary (SB)
Multiply Decimal (MP)
Divide Decimal (DP)
Logical AND (NC)
Logical OR (OC)
Exclusive OR (XC)

Decision and Control Instructions

♦ Eleven decision and control instructions are available to perform the following functions:

1. conditional and unconditional transfer of control
2. data and address comparison
3. control of the interrupt system
4. control of the processor state in which the computer is operating
5. loading and storing of general registers
6. stop the processor

These instructions are:

Branch On Condition (BC)
Branch and Link (BAL)
Branch and Link (BALR)
Branch On Count (BCT)
Compare Decimal (CP)
Compare Logical (CLC)
Set P_2 Register (STP2)
Test Under Mask (TM)
Load Multiple (LM)
Store Multiple (STM)
Halt and Branch (HB)

Input/Output Instructions

♦ Seven input/output instructions are available to provide for the communication between the processor and all input/output devices through the RCA standard interface.

Each 70/25 instruction is described in detail. All operation codes are shown in hexadecimal; all addresses are shown in decimal with the exception of those denoted by subscript 16. Staticizing time has been included in all of the instructions. These instructions are:

Read Forward (RDF)
Read Reverse (RDR)
Write (WR)
Write Control (WRC)
Write Erase (WRE)
Sense (IOS)
Post Status (PS)

Move (MVC)

General Description

♦ This instruction transfers a specified number of consecutive bytes from one high-speed memory location to another. From 1 to 256 bytes may be transferred.

Format

8	8	4	12	4	12
OP	L	B_1	D_1	B_2	D_2

OP — $(D2)_{16}$

L — number of bytes minus one to be transferred.

B_1/D_1 — HSM location to receive the first byte transferred.

B_2/D_2 — HSM location of the first byte to be transferred.

Direction of Operation

♦ Left to right.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address of the leftmost byte in the first operand. The contents of the general register, specified by B_2, are added to the contents of D_2 to obtain the B_2/D_2 address of the leftmost byte of the second operand. The B_1/D_1 address is placed in the A register; the B_2/D_2 address is placed in the B register.

The byte specified by the B register is transferred to the HSM location specified by the A register. The contents of the A and B registers are incremented by one; the contents of the L register are decremented by one. If the L register = $(FF)_{16}$, the instruction is terminated; if not, the cycle is repeated.

Condition Code

♦ Unchanged.

Timing

♦ t (μsec) = 13.5 + 3W + 3B

where: W = number of words.

B = number of bytes.

The 70/25 moves full words when on a word boundary. Before and after the word boundary, it moves one byte at a time. The timing above assumes that both operands are on word boundaries.

Example

Instruction

OP	L	B_1	D_1	B_2	D_2
D2	003	13	0086	13	0092

General register 13 contains 04000.

HSM before execution

4086	4087	4088	4089	4090	4091	4092	4093	4094	4095
7	0	2	5	—	—	R	C	A	*

HSM after execution

4086	4087	4088	4089	4090	4091	4092	4093	4094	4095
R	C	A	*	—	—	R	C	A	*

Edit (ED)

General Description

♦ This instruction alters the second operand from packed format to zoned format. The second operand also is edited under the control of the first operand. The result is then stored in the first operand location.

Format

8	8	4	12	4	12
OP	L	B_1	D_1	B_2	D_2

OP — $(DE)_{16}$

L — number of bytes minus one in the first operand (edit mask).

B_1/D_1 — HSM location of the MSD of the edit mask and the result.

B_2/D_2 — HSM location of the MSD of the data field.

Direction of Operation

♦ Left to right.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address of the leftmost byte in the first operand. The contents of the general register, specified by B_2, are added to the contents of D_2 to obtain the B_2/D_2 address of the leftmost byte of the second operand. The B_1/D_1 address is placed in the A register; the B_2/D_2 address is placed in the B register.

A description of the edit process is as follows:

1. The first byte of the edit mask acts as the fill character and is left in position. Transfer of edited data starts at HSM location one to the right of the fill byte.
2. Zero suppression with the specified fill character occurs until one of the following conditions takes place:
   a. A non-zero numeric that corresponds to a digit-select character in the edit mask is found in the data field.
   b. A significance start character is found in the edit mask. A significance start character indicates that all characters to the right of it are not to be suppressed.
3. After zero suppression has terminated, data and edit symbols are inserted according to the edit mask.
4. If the sign of the data field is positive, all remaining positions in the mask/result area are suppressed with the fill character.
5. If the sign of the data field is negative, all characters in the remaining positions of the mask/result area are retained.
6. Multiple fields may be edited by placing a field separator in the edit mask. The location of this field separator is between the end of one field and the start of the next field. It resets the edit operation to start on the second field and the fill character replaces the field separator.
7. All data characters inserted in the result have $(1111)_2$ placed in the zone portion of the edited character.
8. A significance start character is overlayed by the corresponding data digit. If the corresponding digit is zero, it is overlayed with the fill character.

Condition Code

- 0 — result field is zero.
  1 — result is less than zero.
  2 — result is greater than zero.
  3 — not used.

Timing

- t (μsec) = 13.5 + 1.5 (2I + 2F + 2.5D)
  where: I = number of inserts.
  F = number of fills.
  D = number of significant digits.

Examples

- d = insertion of digit $(0010\ 0000)_2$.
  s = significance start $(0010\ 0001)_2$.
  b = field separator $(0010\ 0010)_2$.

Example #1

Instruction

OP	L	B_1	D_1	B_2	D_2
DE	010	2	0000	2	1000

General Register 2 contains 02000.

HSM before execution

2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010
—	d	d	,	d	d	s	.	d	d	—

Edit mask

3000	3001	3002	3003
0 0	0 1	6 7	5 —

Data field

HSM after execution

2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010
—	—	—	—	—	1	6	.	7	5	—

Result

Condition Code = 1.

Example #2

Using same edit mask

3000	3001	3002	3003
0 0	0 0	0 0	9 +

Data field

HSM after execution

2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010
—	—	—	—	—	—	—		0	9	—

Result

Condition Code = 2.

Pack (PACK)

General Description

♦ This instruction alters the second operand from zoned format to packed format. The result is placed in the first operand. Zone bits of the rightmost byte are interpreted as the sign. High-order zeros are inserted when the first operand is longer than the second. High-order digits are ignored when the second operand is longer than the first.

Format

8	4	4	4	12	4	12
OP	L_1	L_2	B_1	D_1	B_2	D_2

OP — $(F2)_{16}$

L_1 — number of bytes minus one in the first operand.

L_2 — number of bytes minus one in the second operand.

B_1/D_1 — HSM location of the MSD of the packed result.

B_2/D_2 — HSM location of the MSD of the operand to be packed.

Direction of Operation

♦ Right to left.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address of the leftmost byte in the first operand to be altered. The length (L_1) specifies the number of bytes that are added to the location obtained above (B_1/D_1), thus giving the processor the address of the rightmost byte of the first operand to be altered. The length of the operand may be from one to 16 bytes since L_1 may be from 0000–1111. The second address (B_2/D_2) is obtained in a similar manner except that B_2, D_2, and L_2 are used. The B_1/D_1 address is placed in the A register; the B_2/D_2 address is placed in the B register.

The byte specified by the B register is placed in the G register. The B register and L_2 are decremented by one. The byte now specified by the B register is placed in the F register. The low-order four bits of F are placed in the high-order four bits of G and the contents of the F/G registers are placed in the HSM location specified by the A register. The contents of the A register, B register, L_1, and L_2 are decremented by one.

If $L_1 = (F)_{16}$, the instruction is terminated. If $L_2 = (F)_{16}$ and $L_1 \neq (F)_{16}$, high-order zeros are filled in the field specified by the A register. If neither L_1 nor $L_2 = (F)_{16}$, the cycle is repeated.

For the first byte accessed, the high-order four bits of F and the low-order four bits of F are reversed and restored in F. The contents of F are then written to the HSM location specified by the A register. The A register, B register, L_1, and L_2 are then decremented by one and the cycle is repeated.

Condition Code

♦ Unchanged.

Timing

♦ $t\ (\mu sec) = 19.5 + 1.5N_1 + 3N_2$

where: N_1 = number of bytes in first operand.

N_2 = number of bytes in second operand.

Pack (PACK)

Example

Instruction

OP	L_1	L_2	B_1	D_1	B_2	D_2
F2	03	03	04	0000	05	0100

General register 4 contains 03000.
General register 5 contains 02000.

HSM before execution

3000	3001	3002	3003
X	X	X	X

2100	2101	2102	2103
Z 2	Z 5	Z 6	— 3

HSM after execution

3000	3001	3002	3003
0 0	0 2	5 6	3 —

2100	2101	2102	2103
Z 2	Z 5	Z 6	— 3

Condition code is unchanged.

(UNPK) Unpack

General Description

♦ This instruction alters the second operand from packed format to zoned format. The result is placed in the first operand. The bits $(1111)_2$ are inserted in the zone portion for all bytes except the sign position. For the byte containing the sign, the low-order four bits and the high-order four bits are exchanged and placed in the result. High-order zeros are inserted when the first operand is longer than the second. High-order digits are ignored when the second operand is longer than the first.

Format

8	4	4	4	12	4	12
OP	L_1	L_2	B_1	D_1	B_2	D_2

OP — $(F3)_{16}$

L_1 — number of bytes minus one in the first operand.

L_2 — number of bytes minus one in the second operand.

B_1/D_1 — HSM location of the MSD of the result in zoned format.

B_2/D_2 — HSM location of the MSD of the operand to be changed to zoned format.

Direction of Operation

♦ Right to left.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address of the leftmost byte in the first operand to be altered. The length (L_1) specifies the number of bytes that are added to the HSM location obtained above (B_1/D_1), thus giving the processor the address of the rightmost byte in the first operand to be altered. The length of the operand may be from one to 16 bytes since L_1 may be from 0000–1111. The second address (B_2/D_2) is obtained in a similar manner except that B_2, D_2, and L_2 are used. The B_1/D_1 address is placed in the A register; the B_2/D_2 address is placed in the B register.

The byte specified by the B register is placed in the G register. The low-order four bits of G are transferred to the low-order four bits of the F register. The high-order four bits of F are set to the code $(1111)_2$. The contents of F are written to the location specified by the A register. The A register and L_1 are decremented by one. Next, the high-order four bits of G are transferred to the low-order four bits of G; the high-order four bits are set to the code $(1111)_2$. The contents of G are written to the HSM location specified by the A register. The A register, B register, L_1, and L_2 are decremented by one.

If $L_1 = (F)_{16}$, the instruction is terminated. If $L_2 = (F)_{16}$ and $L_1 \neq (F)_{16}$, high-order zeros are filled in the field specified by the A register. If neither L_1 nor $L_2 = (F)_{16}$, the cycle is repeated.

For the first byte accessed, the high-order four bits of G and the low-order four bits of G are reversed and restored in G. The contents of G are then written to the HSM location specified by the A register. The A register, B register, L_1, and L_2 are decremented by one and the cycle is repeated.

Condition Code

♦ Unchanged.

Unpack (UNPK)

Timing

♦ t (μsec) = $21 + 3N_1 + 1.5N_2$

where: N_1 = number of bytes in first operand.

N_2 = number of bytes in second operand.

Example

Instruction

OP	L_1	L_2	B_1	D_1	B_2	D_2
F3	05	02	00	0200	04	0157

General register 4 contains 04000.

HSM before execution

0200	0201	0202	0203	0204	0205
X	X	X	X	X	X

4157	4158	4159
6 1	2 7	5 —

HSM after execution

0200	0201	0202	0203	0204	0205
Z 0	Z 6	Z 1	Z 2	Z 7	— 5

4157	4158	4159
6 1	2 7	5 —

where $Z = (1111)_2 = (F)_{16}$

Condition code is unchanged.

Translate (TR)

General Description

♦ This instruction causes the variable-length operand, specified by the first address, to be translated, byte for byte, according to the translation table specified by the second address. The result replaces the bytes within the field specified by the first address. The second operand is not altered unless an overlap occurs.

Format

8	8	4	12	4	12
OP	L	B_1	D_1	B_2	D_2

OP — $(DC)_{16}$

L — number of bytes minus one in the first operand.

B_1/D_1 — HSM location of the leftmost byte of the first operand.

B_2/D_2 — HSM location of the leftmost byte of the second operand (Translate Table).

Direction of Operation

♦ Left to right.

Outline of Operation

♦ The bytes of the first operand are termed the argument bytes. The bytes of the second operand are termed the function bytes. Processing of the first operand is from left to right, one byte at a time. Each argument byte is added to the second operand address, which is the starting location of the table. This value, in turn, addresses a function byte within the table. The function byte at this location then replaces the original argument byte of the first operand. The operation terminates when the first operand bytes have been exhausted.

Condition Code

♦ Unchanged.

Timing

♦ t (μsec) = 13.5 + 6.75N
where: N = number of bytes in first operand.

Example

Instruction

OP	L	B_1	D_1	B_2	D_2
DC	0002	4	0785	0	0300

General register 4 contains 03000.

HSM before execution

3785	3786	3787
$(C3)_{16}$	$(C8)_{16}$	$(C4)_{16}$

$(C3)_{16} = (195)_{10}$ $(C8)_{16} = (200)_{10}$ $(C4)_{16} = (196)_{10}$

0300	0495	0496	0500
$(00)_{16}$	$(F3)_{16}$	$(F8)_{16}$	$(F4)_{16}$

HSM after execution

3785	3786	3787
$(F3)_{16}$	$(F4)_{16}$	$(F8)_{16}$

0300	0495	0496	0500
$(00)_{16}$	$(F3)_{16}$	$(F8)_{16}$	$(F4)_{16}$

Add Decimal (AP)

General Description

♦ This instruction performs an algebraic addition of two packed-decimal fields and places the sum in the area originally occupied by the first operand field (augend). These fields need not be of equal length.

Format

8	4	4	4	12	4	12
OP	L_1	L_2	B_1	D_1	B_2	D_2

OP — $(FA)_{16}$

L_1 — number of bytes minus one in the first operand (augend) and the sum.

L_2 — number of bytes minus one in the second operand (addend).

B_1/D_1 — HSM location of the MSD of the first operand (augend) and the sum.

B_2/D_2 — HSM location of the MSD of the second operand (addend).

Direction of Operation

♦ Right to left.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address of the leftmost byte in the first operand to be added. The length (L_1) specifies the number of bytes that are added to the location obtained above (B_1/D_1), thus giving the processor the address of the rightmost byte in the first operand to be added. The length of the operand may be from one to 16 bytes since L_1 may be from 0000–1111. The second address (B_2/D_2) is obtained in a similar manner except that B_2, D_2, and L_2 are used. The B_1/D_1 address is placed in the A register; the B_2/D_2 address is placed in the B register.

The byte specified by the B register is placed in the G register. The byte specified by the A register is placed in the F register. G and F now contain one byte (two decimal digits) each. The contents of G and F are added by the adder circuit; the result is placed in the HSM location specified by the A register. The A register, B register, L_1, and L_2 are decremented by one.

If $L_1 = (F)_{16}$, the instruction is terminated. If $L_2 = (F)_{16}$ and $L_1 \neq (F)_{16}$, the field specified by the B register is assumed to contain high-order zeros. If neither L_1 nor $L_2 = (F)_{16}$, the cycle is repeated.

Upon termination, the condition code is set to indicate positive, negative, or zero result. If the field specified by the A register is not large enough to hold the result, an overflow condition exists and the condition code is set to 3.

Because the first byte, accessed by the A and B registers, contains the sign of the field, only the high-order four bits of G and F are added during the first cycle. The rightmost four bits designate the sign control in the algebraic addition.

Condition Code

♦ 0 — sum is zero.

1 — sum is less than zero.

2 — sum is greater than zero.

3 — overflow.

Add Decimal (AP)

Timing

♦ t (μsec) = $21.75 + 2.25N_1 + 1.5N_2$

where: N_1 = number of bytes in first operand.

N_2 = number of bytes in second operand.

Interrupt Action

♦ An overflow interrupt can occur in the RCA 70/25 while executing the Add Decimal instruction. This can only occur if the overflow interrupt bit of the Interrupt Mask is set to a one, and there is a carry out of the high-order position of the first operand. If the interrupt is not desired, a test of the condition code indicates whether or not an overflow has occurred.

Special Conditions

♦ 1. The sign is taken from the rightmost four bits of the least significant byte.

2. If the second operand is shorter than the first operand, high-order zeros are supplied for the second operand.

3. If the second operand is longer than the first operand, the high-order digits of the second operand are dropped. Dropping of the high-order digits, even if significant, does not affect the condition code or sign of the result.

4. Overflow is based on a carry out of the high-order position of the first operand.

5. A zero is always positive.

6. Digits or signs are not checked for validity.

7. Operands may overlap if their rightmost bytes coincide.

8. B_1 or B_2 address components that are zero, specify that no general register is to be used in computing the address of an operand.

Example

Instruction

OP	L_1	L_2	B_1	D_1	B_2	D_2
FA	03	02	03	0200	03	0000

General register 3 contains 00500.

HSM before execution

0700	0701	0702	0703
0 0	1 2	3 8	9 +

0500	0501	0502
9 2	3 7	5 +

HSM after execution

0700	0701	0702	0703
0 1	0 4	7 6	4 +

0500	0501	0502
9 2	3 7	5 +

Condition code setting = 2.

Add Binary (AB)

General Description

♦ This instruction performs a binary addition of the first and second operands and places the sum in the area originally occupied by the first operand (augend). If the length of the second operand is greater than the first, high-order bytes are dropped. A carry in the first operand causes an overflow indicator to be set.

Format

OP — $(F6)_{16}$

L_1 — number of bytes minus one in the first operand (augend) and result.

L_2 — number of bytes minus one in the second operand (addend).

B_1/D_1 — HSM location of the MSD of the first operand (augend) and result.

B_2/D_2 — HSM location of the MSD of the second operand (addend).

Direction of Operation

♦ Right to left.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address of the leftmost byte in the first operand to be added. The length (L_1) specifies the number of bytes that are added to the location obtained above (B_1/D_1), thus giving the processor the address of the rightmost byte in the first operand to be added. The length of the operand may be from one to 16 bytes since L_1 may be from 0000–1111. The second address (B_2/D_2) is obtained in a similar manner except that B_2, D_2, and L_2 are used. The B_1/D_1 address is placed in the A register; the B_2/D_2 address is placed in the B register.

The byte specified by the B register is placed in the G register. The byte specified by the A register is placed in the F register. G and F now contain one byte each. The contents of G and F are added binarily by the adder circuit; the result is placed in the HSM location specified by the A register. The A register, B register, L_1, and L_2 are decremented by one.

If $L_1 = (F)_{16}$, the instruction is terminated. If $L_2 = (F)_{16}$, and $L_1 \neq (F)_{16}$, the field specified by the B register is assumed to contain high-order zeros. If neither L_1 nor $L_2 = (F)_{16}$, the cycle is repeated.

Upon termination, the condition code is set to indicate positive, negative, or zero result. If the field, specified by the A register is not large enough to hold the result, an overflow condition exists and the condition code is set to 3.

Condition Code

♦ 0 — sum is zero.

1 — not used.

2 — sum is greater than zero.

3 — overflow.

Timing

♦ t (μsec) $= 21.75 + 2.25N_1 + 1.5N_2$

where: N_1 = length (in bytes) of first operand.

N_2 = length (in bytes) of second operand.

Add Binary (AB)

Interrupt Action

♦ An overflow interrupt can occur in the RCA 70/25 while executing the Add Binary instruction. This can only occur if the overflow interrupt bit of the Interrupt Mask is set to a one, and the first operand is not long enough to hold the result. If the interrupt is not desired, then a test of the condition code indicates whether or not an overflow has occurred.

Special Conditions

♦ 1. If the second operand is longer than the first operand, the high-order digits of the second operand are dropped. Dropping of high-order digits, even if significant, does not affect the condition code.

2. Overflow is based on a carry out of the high-order position of the first operand.

Examples

Example #1

Instruction

OP	L_1	L_2	B_1	D_1	B_2	D_2
F6	02	01	13	0000	13	0003

General register 13 contains 02000.

HSM before execution

2000	2001	2002	2003	2004
1111 0010	1111 0101	1111 0110	1111 0001	1111 0011

HSM after execution

2000	2001	2002	2003	2004
1111 0011	1110 0111	1110 1001	1111 0001	1111 0011

Condition Code = 2.

Example #2

An example using the Add Binary instruction for address modification is as follows:

Instruction

OP	L_1	L_2	B_1	D_1	B_2	D_2
F6	01	01	4	0004	4	1000

General register 4 contains 02000.

$\overbrace{OP = (FA)_{16}}\ \overbrace{L = (09)_{16}}\ \overbrace{B_1 = (0)_{16}}\ \overbrace{D_1 = (0B6)_{16}}\ \overbrace{B_2 = (0)_{16}}\ \overbrace{D_2 = (07F)_{16}}$

HSM before execution

2000	2001	2002	2003	2004	2005
1111 1010	0000 1001	0000 0000	1011 0110	0000 0000	0111 1111

3000	3001
$(00)_{16}$ 0000 0000	$(64)_{16}$ 0110 0100

HSM after execution

2000	2001	2002	2003	2004	2005
1111 1010	0000 1001	0000 0000	1011 0110	0000 0000	1110 0011

$\overbrace{OP = (FA)_{16}}\ \overbrace{L = (09)_{16}}\ \overbrace{B_1 = (0)_{16}}\ \overbrace{D_1 = (0B6)_{16}}\ \overbrace{B_2 = (0)_{16}}\ \overbrace{D_2 = (0E3)_{16}}$

This example has incremented the D_2 address of a Decimal Add instruction located at 2000 by a factor of $(100)_{10}$ or $(64)_{16}$.

Subtract Decimal (SP)

General Description

♦ This instruction performs an algebraic subtraction of two packed-decimal fields and places the difference in the area originally occupied by the first operand field (minuend). The fields need not be of equal length.

Format

8	4	4	4	12	4	12
OP	L_1	L_2	B_1	D_1	B_2	D_2

OP — $(FB)_{16}$

L_1 — number of bytes minus one in the first operand (minuend) and difference.

L_2 — number of bytes minus one in the second operand (subtrahend).

B_1/D_1 — HSM location of the MSD of the first operand (minuend) and difference.

B_2/D_2 — HSM location of the MSD of the second operand (subtrahend).

Direction of Operation

♦ Right to left.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address of the leftmost byte in the first operand to be subtracted. The length (L_1) specifies the number of bytes that are added to the location obtained above, thus giving the processor the address of the rightmost byte in the first operand to be subtracted. The length of the operand may be from one to 16 bytes since L_1 may be from 0000–1111. The second address (B_2/D_2) is obtained in a similar manner except that B_2, D_2, and L_2 are used. The B_1/D_1 address is placed in the A register; the B_2/D_2 address is placed in the B register.

The byte specified by the B register is placed in the G register. The byte specified by the A register is placed in the F register. G and F now contain one byte (two decimal digits) each. The contents of G and F are subtracted by the adder circuit; the result is placed in the HSM location specified by the A register. The A register, B register, L_1, and L_2 are decremented by one.

If $L_1 = (F)_{16}$, the instruction is terminated. If $L_2 = (F)_{16}$ and $L_1 \neq (F)_{16}$, the field specified by the B register is assumed to contain high-order zeros. If neither L_1 nor $L_2 = (F)_{16}$, the cycle is repeated.

Upon termination, the condition code is set to indicate positive, negative, or zero result. If the field specified by the A register is not large enough to hold the result, an overflow condition exists and the condition code is set to 3.

Because the first byte, accessed by the A and B registers, contains the sign of the field, only the high-order four bits of G and F are subtracted during the first cycle. The rightmost four bits designate the sign control in the algebraic subtraction.

Condition Code

♦ 0 — difference is zero.

1 — difference is less than zero.

2 — difference is greater than zero.

3 — overflow.

Subtract Decimal (SP)

Timing

♦ t (μsec) = $21.75 + 2.25N_1 + 1.5N_2$

where: N_1 = number of bytes in first operand.

N_2 = number of bytes in second operand.

Interrupt Action

♦ An overflow interrupt can occur in the RCA 70/25 while executing the Subtract Decimal instruction. This can only occur if the overflow interrupt bit of the Interrupt Mask is set to a one, and there is a carry out of the high-order position of the first operand. If the interrupt is not desired, a test of the condition code indicates whether or not an overflow has occurred.

Special Conditions

1. The sign is taken from the rightmost four bits of the least significant byte.
2. If the second operand is shorter than the first operand, high-order zeros are supplied for the second operand.
3. If the second operand is longer than the first operand, the high-order digits of the second operand are dropped. Dropping of high-order digits, even if significant, does not affect the condition code or sign of the result.
4. Overflow is based on a carry out of the high-order position of the first operand.
5. A zero is always considered positive.
6. Digits or sign are not checked for validity.
7. Operands may overlap if their rightmost bytes coincide.
8. B_1 or B_2 address components that are zero, specify that no general register is to be used in computing the address of an operand.

Example

Instruction

OP	L_1	L_2	B_1	D_1	B_2	D_2
FB	02	02	4	0205	5	0030

General register 4 contains 00400.
General register 5 contains 00600.

HSM before execution

0605	0606	0607
8 3	2 7	1 –

0630	0631	0632
2 9	3 8	6 +

HSM after execution

0605	0606	0607
1 2	6 5	7 –

0630	0631	0632
2 9	3 8	6 +

Condition code setting = 3 (overflow).

Subtract Binary (SB)

General Description

♦ This instruction performs a binary subtraction of the second operand from the first operand. The difference is placed in the area originally occupied by the first operand (minuend).

Format

8	4	4	4	12	4	12
OP	L_1	L_2	B_1	D_1	B_2	D_2

OP — $(F7)_{16}$

L_1 — number of bytes minus one in the first operand (minuend) and difference.

L_2 — number of bytes minus one in the second operand (subtrahend).

B_1/D_1 — HSM location of the MSD of the first operand (minuend) and difference.

B_2/D_2 — HSM location of the MSD of the second operand (subtrahend).

Direction of Operation

♦ Right to left.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address of the leftmost byte in the first operand to be subtracted. The length (L_1) specifies the number of bytes that are added to the location obtained above (B_1/D_1), thus giving the processor the address of the rightmost byte in the first operand to be subtracted. The length of the operand may be from one to 16 bytes since L_1 may be from 0000–1111. The second address (B_2/D_2) is obtained in a similar manner except that B_2, D_2, and L_2 are used. The B_1/D_1 address is placed in the A register; the B_2/D_2 address is placed in the B register.

The byte specified by the B register is placed in the G register. The byte specified by the A register is placed in the F register. G and F now contain one byte each. The contents of G and F are subtracted binarily by the adder circuit; the result is placed in the HSM location specified by the A register. The A register, B register, L_1, and L_2 are decremented by one.

If $L_1 = (F)_{16}$, the instruction is terminated. If $L_2 = (F)_{16}$ and $L_1 \neq (F)_{16}$, the field specified by the B register is assumed to contain high-order zeros. If neither L_1 nor $L_2 = (F)_{16}$, the cycle is repeated.

Upon termination, the condition code is set to indicate positive, negative, or zero result.

Condition Code

♦ 0 — difference is zero.

1 — difference is less than zero.

2 — difference is greater than zero.

3 — not used.

Timing

♦ t (μsec) $= 21.75 + 2.25N_1 + 1.5N_2$

where: N_1 = length (in bytes) of first operand.

N_2 = length (in bytes) of second operand.

Special Condition

♦ If the second operand is longer than the first operand, the high-order digits of the second operand are dropped. Dropping of high-order digits, even if significant, does not affect the condition code.

Example

Instruction

OP	L_1	L_2	B_1	D_1	B_2	D_2
F7	02	01	04	0000	04	0003

General register 4 contains 02000.

HSM before execution

2000	2001	2002	2003	2004
1111 1001	1111 0100	1111 0010	1111 0001	1111 0111

HSM after execution

2000	2001	2002	2003	2004
1111 1001	0000 0010	1111 1011	1111 0001	1111 0111

Condition code = 2.

Multiply Decimal (MP)

General Description

♦ This instruction multiplies two packed-decimal fields and places the product in the area originally occupied by the first operand field (multiplicand). The sign of the product is determined by the rules of algebra.

Format

8	4	4	4	12	4	12
OP	L_1	L_2	B_1	D_1	B_2	D_2

OP — $(FC)_{16}$

L_1 — number of bytes minus one in the first operand (multiplicand) and product.

L_2 — number of bytes minus one in the second operand (multiplier).

B_1/D_1 — HSM location of the MSD of the first operand (multiplicand) and product.

B_2/D_2 — HSM location of the MSD of the second operand (multiplier).

Direction of Operation

♦ Right to left.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address of the leftmost byte in the first operand to be multiplied. The length (L_1) specifies the number of bytes that are added to the location obtained above (B_1/D_1), thus giving the processor the address of the rightmost byte in the first operand to be multiplied. The length of the operand may be from one to 16 bytes since L_1 may be from 0000–1111. The second address (B_2/D_2) is obtained in a similar manner except that B_2, D_2, and L_2 are used. The B_1/D_1 address is placed in the A register; the B_2/D_2 address is placed in the B register.

Multiplication takes place and the product is stored in the first operand field.

Condition Code

♦ Unchanged.

Timing

♦ $t\ (\mu sec) = 26.25 + 9N_1 - 1.5N_2 + C[3.75\ (N_1 - N_2) + 3]$

where: C = sum of the value of multiplier digits.

N_1 = number of bytes in first operand.

N_2 = number of bytes in second operand.

If $(N_1 - N_2)$ is less than zero, the result is the same as $(N_1 - N_2)$ being equal to zero.

Special Conditions

1. The sign is taken from the rightmost four bits of the least significant byte.
2. The number of digits in the product is the sum of the number of digits in both operands.
3. The maximum product size is 31 digits.
4. No overflow indication is given; therefore, the multiplicand field must have sufficient field size for the development of the product.
5. A zero is always considered positive.
6. Digits or signs are not checked for validity.
7. Operands may overlay if their rightmost bytes coincide.
8. B_1 or B_2 address components that are zero, specify that no general register is to be used in computing the address of an operand.

Example

Instruction

OP	L_1	L_2	B_1	D_1	B_2	D_2
FC	04	01	06	0500	06	0700

General register 6 contains 01000.

HSM before execution

1500	1501	1502	1503	1504
0 0	0 0	4 3	1 2	6 +

1700	1701
2 8	0 −

HSM after execution

1500	1501	1502	1503	1504
0 1	2 0	7 5	2 8	0 −

1700	1701
2 8	0 −

Condition code is unchanged.

Divide Decimal (DP)

General Description

♦ This instruction divides two packed-decimal fields and places the result (signed quotient plus signed remainder) in the area originally occupied by the first operand field (dividend). The quotient is placed leftmost in the first operand field. The remainder is placed rightmost in the first operand field and has a size equal to the divisor size.

Format

8	4	4	4	12	4	12
OP	L_1	L_2	B_1	D_1	B_2	D_2

OP — $(FD)_{16}$

L_1 — number of bytes minus one in the first operand (dividend) and result.

L_2 — number of bytes minus one in the second operand (divisor).

B_1/D_1 — HSM location of the MSD of the first operand (dividend) and result.

B_2/D_2 — HSM location of the MSD of the second operand (divisor).

Direction of Operation

♦ Right to left.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address of the leftmost byte in the first operand to be divided. The length (L_1) specifies the number of bytes that are added to the location obtained above (B_1/D_1), thus giving the processor the address of the rightmost byte in the first operand to be divided. The length of the operand may be from one to 16 bytes since L_1 may be from 0000–1111. The second address (B_2/D_2) is obtained in a similar manner except that B_2, D_2, and L_2 are used. The B_1/D_1 address is placed in the A register; the B_2/D_2 address is placed in the B register.

Division takes place and the result is placed in the first operand field.

Condition Code

♦ Unchanged.

Timing

♦ $t\ (\mu sec) = 22.5 + 29.25N_1 - 27N_2 + 37.5N_2\ (N_1–N_2)$

where: N_1 = number of bytes in first operand.

N_2 = number of bytes in second operand.

Interrupt Action

♦ A divide-exception interrupt can occur in the RCA 70/25 while executing the Divide Decimal instruction. This can only occur if the arithmetic overflow/divide exception interrupt mask bit is set to a one. The condition for a divide exception interrupt can be determined by a trial subtraction. The leftmost digit of the divisor field is aligned with the leftmost digit minus one of the dividend field. When the divisor, so aligned, is less than or equal to the dividend, a divide exception interrupt is indicated. Also, a decimal divide exception interrupt occurs if the dividend does not have at least one leading zero.

Special Conditions

1. The sign is taken from the rightmost four bits of the least significant byte.
2. The maximum dividend is 31 digits and a sign.
3. The smallest remainder size is one digit and sign, therefore, the maximum quotient size is 29 digits and sign.

Special Conditions (Cont'd)

4. The sign of the quotient is determined by the rules of algebra; the remainder has the sign of the dividend.
5. The dividend must contain at least one leading zero.

Example

Instruction

OP	L_1	L_2	B_1	D_1	B_2	D_2
FD	04	01	05	0300	06	0100

General register 5 contains 01200.
General register 6 contains 04000.

HSM before execution

1500	1501	1502	1503	1504
0 1	6 4	7 6	1 2	6 +

4100	4101
3 4	1 —

HSM after execution

1500	1501	1502	1503	1504
4 8	3 1	7 —	0 2	9 +

4100	4101
3 4	1 —

Condition code is unchanged.

Logical AND (NC)

General Description

♦ This instruction performs a logical "AND" operation on two operands of equal length according to the rules specified below. The result is placed in the area originally occupied by the first operand.

Format

8	8	4	12	4	12
OP	L	B_1	D_1	B_2	D_2

OP — $(D4)_{16}$

L — number of bytes minus one in each operand.

B_1/D_1 — HSM location of the MSD of the first operand and result.

B_2/D_2 — HSM location of the MSD of the second operand.

Direction of Operation

♦ Left to right.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address of the leftmost byte in the first operand to be manipulated. The second address (B_2/D_2) is obtained in a similar manner except that B_2 and D_2 are used. The B_1/D_1 address is placed in the A register; the B_2/D_2 address is placed in the B register.

The byte specified by the B register is placed in the G register. The byte specified by the A register is placed in the F register. G and F are combined bit-by-bit according to the following rules:

Rules for Logical "AND" Operation

Bit in first operand (A)	Bit in second operand (B)	Bit in the result
0	0	0
0	1	0
1	0	0
1	1	1

The result is placed in the HSM location specified by the A register. The contents of the A register and the B register are incremented by one; the contents of L are decremented by one. If L = $(FF)_{16}$, the instruction is terminated; otherwise, the cycle is repeated.

Condition Code

♦ 0 — result is zero.

1 — result is not zero.

2 — not used.

3 — not used.

Timing

♦ t (μsec) = 13.5 + 3.75N

where: N = length (in bytes) of the operand.

Logical OR (OC)

General Description

♦ This instruction performs a logical "OR" operation on two operands of equal length according to the rules specified below. The result is placed in the area originally occupied by the first operand.

Format

8	8	4	12	4	12
OP	L	B_1	D_1	B_2	D_2

OP — $(D6)_{16}$

L — number of bytes minus one in each operand.

B_1/D_1 — HSM location of the MSD of the first operand and result.

B_2/D_2 — HSM location of the MSD of the second operand.

Direction of Operation

♦ Left to right.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address of the leftmost byte in the first operand to be manipulated. The second address (B_2/D_2) is obtained in a similar manner except that B_2 and D_2 are used. The B_1/D_1 address is placed in the A register; the B_2/D_2 address is placed in the B register.

The byte specified by the B register is placed in the G register. The byte specified by the A register is placed in the F register. G and F are combined bit-by-bit according to the following rules:

Rules for Logical "OR" Operation

Bit in first operand (A)	Bit in second operand (B)	Bit in the result
0	0	0
0	1	1
1	0	1
1	1	1

The result is placed in the HSM location specified by the A register. The contents of the A register and the B register are incremented by one; the contents of L are decremented by one. If L = $(FF)_{16}$, the instruction is terminated; otherwise, the cycle is repeated.

Condition Code

♦ 0 — result is zero.

1 — result is not zero.

2 — not used.

3 — not used.

Timing

♦ t (μsec) = 13.5 + 3.75N

where: N = length (in bytes) of the operand.

Exclusive OR (XC)

General Description

♦ This instruction performs an exclusive "OR" operation on two operands of equal length according to the rules specified below. The result is placed in the area originally occupied by the first operand.

Format

8	8	4	12	4	12
OP	L	B_1	D_1	B_2	D_2

OP — $(D7)_{16}$

L — number of bytes minus one in each operand.

B_1/D_1 — HSM location of the MSD of the first operand and result.

B_2/D_2 — HSM location of the MSD of the second operand.

Direction of Operation

♦ Left to right.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address of the leftmost byte in the first operand to be manipulated. The second address (B_2/D_2) is obtained in a similar manner except that B_2 and D_2 are used. The B_1/D_1 address is placed in the A register; the B_2/D_2 address is placed in the B register.

The byte specified by the B register is placed in the G register. The byte specified by the A register is placed in the F register. G and F are combined bit-by-bit according to the following rules:

Rules for Exclusive "OR" Operation

Bit in first operand (A)	Bit in second operand (B)	Bit in the result
0	0	0
0	1	1
1	0	1
1	1	0

The result is placed in the HSM location specified by the A register. The contents of the A register and the B register are incremented by one; the contents of L are decremented by one. If L = $(FF)_{16}$, the instruction is terminated; otherwise, the cycle is repeated.

Condition Code

♦ 0 — result is zero.

1 — result is not zero.

2 — not used.

3 — not used.

Timing

♦ t (μsec) = 13.5 + 3.75N

where: N = length (in bytes) of the operand.

Branch On Condition (BC)

General Description

♦ This instruction transfers control in accordance with the condition code(s) sensed. The condition codes or combination thereof are specified by the mask field. If the specified condition code is not sensed, the next instruction in sequence will be executed.

Format

8	8	4	12
OP	M	B_2	D_2

OP — $(47)_{16}$

M — specifies the condition code to be tested as follows:
- 2^0 — 2^3 — are ignored and must be zeros.
- 2^4 — condition code 3.
- 2^5 — condition code 2.
- 2^6 — condition code 1.
- 2^7 — condition code 0.

B_2/D_2 — HSM address of the next instruction to be executed if the conditions specified by the mask are set.

Outline of Operation

♦ The condition code is tested for the conditions specified in the mask (M). If the condition code is set to any of these conditions, the contents of the general register (specified by B_2) is added to the contents of the displacement field (D_2) to obtain the address of the next instruction to be executed.

The following mask field settings specify which condition code to test:

$(0000\ \ 0000)_2$ = NO OP
$(0001\ \ 0000)_2$ = Condition Code 3
$(0010\ \ 0000)_2$ = Condition Code 2
$(0100\ \ 0000)_2$ = Condition Code 1
$(1000\ \ 0000)_2$ = Condition Code 0

Any of the above bit configurations may be combined; i.e., a mask setting of $(1111\ \ 0000)_2$ indicates to branch on any condition code which is, in effect, an unconditional branch.

Condition Code

♦ Unchanged.

Timing

♦ t (μsec) = 11.25 if branching occurs.
t (μsec) = 9 if no branch.

Example

Instruction

OP	M	B_2	D_2
$(47)_{16}$	$(30)_{16}$	03	0750

$(30)_{16} = (0011\ \ 0000)_2$

General register 3 contains 03000.

If condition code 2 or 3 is set, control is transferred to the instruction at 3750.

Branch and Link (BAL)

General Description

♦ This instruction stores the contents of the P counter in the general register specified by the first address (R_1). The branch location, specified by the second address, is then put into the P counter. The P counter stored is determined by the state in which the program is operating.

Format

8	4	4	4	12
OP	R_1		B_2	D_2

OP — $(45)_{16}$

R_1 — general register in which the P counter is to be stored.

B_2/D_2 — HSM location of the next instruction to be executed.

Outline of Operation

♦ The contents of the general register, specified by B_2, are added to the contents of the displacement field (D_2) to obtain the branch address. The computation of the branch address is performed prior to the storing of the P counter. After storing the P counter in the general register specified by R_1, the branch address is then placed in the P counter.

Condition Code

♦ Unchanged.

Timing

♦ t (μsec) = 12.75

Example

Instruction

OP	R_1		B_2	D_2
45	04		03	0164

General register 3 contains 04000.
General register 4 contains 00000.
P counter contains 05460.
Execution of this instruction causes the following register changes:
General register 4 contains 05460.
P counter contains 04164.

Branch and Link (BALR)

General Description

♦ This instruction stores the contents of the P counter in the general register specified by the first address (R_1). The branch location in the general register, specified by the second address (R_2), is then placed into the P counter. The P counter stored is determined by the state in which the program is operating. If the R_2 field contains zero, the P counter is stored in R_1 but branching does not occur.

Format

8	4	4
OP	R_1	R_2

OP — $(05)_{16}$

R_1 — general register in which the P counter is to be stored.

R_2 — general register that contains the HSM location of the next instruction to be executed.

Outline of Operation

♦ The address in the P counter is stored in the general register specified by the first address. The branch address in the register, specified by the second address (R_2), is then stored in the P counter. If the second address (R_2) is zero, the P counter is stored but branching does not occur.

Condition Code

♦ Unchanged.

Timing

♦ t (μsec) = Branch — 10.50.
t (μsec) = No branch — 6.75.

Example

Instruction

OP	R_1	R_2
05	01	0000

General register 1 contains 00000.
P counter contains 01540.
Execution of this instruction causes the following changes:
General register 1 contains 01540.
P counter contains 01540.

Branch On Count (BCT)

General Description

♦ This instruction decrements the contents of the general register, specified by the first address, by one. When the result is non-zero, the next instruction to be executed is specified by the second address (B_2/D_2). When the result is zero, the next sequential instruction is executed. The branch address is determined prior to decrementing the count in the general register.

Format

OP — $(46)_{16}$

R_1 — indicates the general register to be decremented.

B_2/D_2 — HSM location of the next instruction to be executed when the contents of the general register equal non-zero after decrementing.

Outline of Operation

♦ The contents of the general register, specified by B_2, is added to the contents of the displacement field (D_2) to obtain the branch address. The contents of the general register, specified by R_1, is then decremented by one. If the result is non-zero, the branch address, computed above (B_2/D_2), indicates the next instruction to be executed. If the result is zero, the next sequential instruction is executed.

Condition Code

♦ Unchanged.

Timing

♦ t (μsec) = 15.75 if branching occurs.
t (μsec) = 11.25 if no branching occurs.

Special Conditions

1. An initial value of zero in the general register, specified by R_1, causes the general register to be set to all ones and the branch is taken.
2. General register 0 is the Timer register in the RCA 70/25 and is automatically incremented each 1/60th of a second. Therefore, a designation of register 0 in the R_1 field of the Branch on Count instruction causes the timer to be inaccurate.

Examples

Example #1

Instruction location 01000

OP	R_1		B_2	D_2
46	05		03	0000

General register 3 contains 02000.
General register 5 contains 04096.
Execution of this instruction causes the following to occur:
General register 5 now contains 04095.
Control is transferred to the instruction at location 02000.

Example #2

Using the same instruction format but with the register values changed.
General register 3 contains 03000.
General register 5 contains 00001.
Execution of this instruction causes the following to occur:
General register 5 now contains 00000.
Control now passes to next sequential instruction (location 01004).

Compare Decimal (CP)

General Description

♦ This instruction algebraically compares two operands and indicates the result of the comparison by a specific condition code. The first operand is compared against the second and each operand must be in packed format.

Format

8	4	4	4	12	4	12
OP	L_1	L_2	B_1	D_1	B_2	D_2

OP — $(F9)_{16}$

L_1 — number of bytes minus one in the first operand.

L_2 — number of bytes minus one in the second operand.

B_1/D_1 — HSM location of the MSD of the first operand.

B_2/D_2 — HSM location of the MSD of the second operand.

Direction of Operation

♦ Right to left.

Outline of Operation

♦ The contents of the general register, specified by B_1, is added to the contents of (D_1) to obtain the B_1/D_1 address of the leftmost byte in the first operand to be compared. The length (L_1) specifies the number of bytes that are added to the location obtained above (B_1/D_1), thus giving the processor the address of the rightmost byte in the first operand to be compared. The length of the operand may be from one to 16 bytes since L_1 may be from 0000–1111. The second address (B_2/D_2) is obtained in a similar manner except that B_2, D_2, and L_2 are used. The B_1/D_1 address is placed in the A register; the B_2/D_2 address is placed in the B register.

The byte addressed by the B register is placed in the G register. The byte addressed by the A register is placed in the F register. G and F now contain one byte (two decimal digits) each. The contents of F and G are compared by the comparator circuit. The result of the comparison is stored in the condition code indicators. The A register, B register, L_1, and L_2 are decremented by one.

If $L_1 = (F)_{16}$, the instruction is terminated. If $L_2 = (F)_{16}$ and $L_1 \neq (F)_{16}$, the field specified by the B register is assumed to contain high-order zeros. If neither L_1 nor $L_2 = (F)_{16}$, the cycle is repeated.

Upon termination, the condition code is stored to indicate positive, negative, or zero result.

Because the first byte accessed by the A and B registers contains the sign of the field, the low-order four bits of G and F are compared during the first cycle and the condition code is set if the signs are not the same.

Condition Code

♦ 0 — operands are equal.

1 — first operand low.

2 — first operand high.

3 — not used.

Timing

♦ t (μsec) $= 19.5 + 1.5N_1 + 2.25N_2$

where: N_1 = number of bytes in first operand.

N_2 = number of bytes in second operand.

Example

Instruction

OP	L_1	L_2	B_1	D_1	B_2	D_2
F9	05	02	04	0350	00	1027

General register 4 contains 07000.

7350	7351	7352	7353	7354	7355
0 0	0 6	0 2	3 5	0 0	0 —

1027	1028	1029
3 2	7 1	4 +

Condition code setting = 1 (First operand is low).

General Description

♦ This instruction binarily compares two operands of equal lengths and indicates the result of the comparison by a specific condition code. The binary digits of the first operand are compared (left to right) with the binary digits of the second operand. This instruction terminates when an inequality is found.

Format

8	8	4	12	4	12
OP	L	B_1	D_1	B_2	D_2

OP — $(D5)_{16}$

L — number of bytes minus one to be compared.

B_1/D_1 — HSM location of the MSD of the first operand.

B_2/D_2 — HSM location of the MSD of the second operand.

Direction of Operation

♦ Left to right.

Outline of Operation

♦ The contents of the general register, specified by B_1, is added to the contents of D_1 to obtain the location of the leftmost byte in the first operand to be compared. The second address is obtained in a similar manner except that D_2 and B_2 are used. The B_1/D_1 address is placed in the A register; the B_2/D_2 address is placed in the B register.

The byte specified by the B register is placed in the G register. The byte specified by the A register is placed in the F register. G and F are compared bit-by-bit in the comparator circuit. If an inequality is found, the condition code is set to 1 or 2; i.e., first operand low will set condition code to 1, and first operand high will set condition code to 2. The instruction is then terminated by the inequality. If the bits compared were equal, the contents of the A register and B register are incremented by one and the contents of L are decremented by one. If $L = (FF)_{16}$, the instruction is terminated, and the condition code is set to zero to indicate equal operands. If $L \neq (FF)_{16}$, the cycle is repeated.

Condition Code

♦ 0 — operands are equal.

1 — first operand is low.

2 — first operand is high.

3 — not used.

Timing

♦ t (μsec) = 14.5 + 3B

where: B = number of bytes compared before and inequality occurs.

Example

Instruction

OP	L	B_1	D_1	B_2	D_2
D5	0002	03	0096	04	0400

General register 3 contains 02000.
General register 4 contains 03000.

Example (Cont'd)

HSM before and after execution

2096	2097	2098
A	B	C
1100 0001	1100 0010	1100 0011

3400	3401	3402
1	2	3
1100 0001	1111 0010	1111 0011

First operand is low — condition code = 1.

General Description

♦ This instruction transfers program control from the *Interrupt State* to the *Processing State*. The P_2 counter is loaded with a value in the first operand address. Control is transferred to the P_1 counter. It also restores the condition code indicator of P_1.

Format

8	8	4	12
OP	M	B_1	D_1

OP — $(82)_{16}$

M — not used.

B_1/D_1 — value to be placed in the program counter (P_2).

Outline of Operation

♦ The original condition code, stored in reserved memory location 41 when control was transferred to the P_2 state, resets the condition code indicators. The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address. This address is loaded into reserved memory locations 44 and 45 (P_2 counter). Control is then transferred to the instruction specified by the program counter for P_1 (locations 40 and 41).

Note: This instruction may be executed in the *Processing State (P_1)*.

Condition Code

♦ Set by reserved memory location 43.

Timing

♦ t (μsec) = 12.75

Example

Instruction

OP	M	B_1	D_1
$(82)_{16}$	$(00)_{16}$	04	0500

General register 4 contains 02000.

HSM before execution

0042	0043	0044	0045
1	2	X	X

$(09C4)_{16} = (2500)_{10}$

0042	0043	0044	0045
1	2	$(09)_{16}$	$(C4)_{16}$

Control is transferred to the instruction specified by the P_1 counter.

Test Under Mask (TM)

General Description

♦ This instruction binarily compares two bytes and indicates the result of the comparison by the condition code. The byte indicated by the second operand is compared bit-by-bit with the mask byte (first operand).

Format

8	8	4	12
OP	M	B_2	D_2

OP — $(91)_{16}$

M — mask byte to indicate which bits are to be tested for a one bit.

B_2/D_2 — HSM location of the byte to be tested.

Outline of Operation

♦ The contents of the general register, specified by B_2, are added to the contents of D_2 to obtain the B_2/D_2 address of the byte to be tested. The B_2/D_2 address is placed in the B register.

The byte, specified by the B register, is placed in the G register. The mask is contained in the L register. If the mask bit is zero, the storage bit is ignored. If the masked bit is one, the status of the storage bit is set in the condition code. The cycle is repeated until all eight bits have been checked.

Condition Code

♦ 0 — selected bits are all zeros; mask is all zero.

1 — selected bits are mixed zeros and ones.

2 — not used.

3 — selected bits are all ones.

Timing

♦ t (μsec) = 10.5

Example

Instruction

OP	M	B_2	D_2
91	0033	09	0094

General register 9 contains 06000.

M = $(0033)_{10}$ = $(0010\ 0001)_2$

Location 6094 = $(6)_{10}$ = $(0000\ 0110)_2$

M indicates to check only bits 2^0 and 2^5.

Because bits 2^0 and 2^5 at location 6094 are equal to 0, condition code zero is set.

General Description

♦ This instruction loads a set of contiguous general registers, starting with the one specified by the first address and ending with the third address, with operands from storage. The second address specifies the storage location of the first operand (word) to be loaded into the general registers.

Format

8	4	4	4	12
OP	R_1	R_3	B_2	D_2

OP — $(98)_{16}$

R_1 — specifies the first general register to be loaded.

R_3 — specifies the last general register to be loaded.

B_2/D_2 — HSM location of the first operand (word) that is to be loaded into the first general register.

Outline of Operation

♦ The contents of the general register, specified by B_2, are added to the contents of D_2 to obtain the B_2/D_2 address of the first operand (word). The loading places the first word (32 bits) from storage into the first general register specified by R_1; the next sequential word (32 bits) into the next sequential register; etc. This loading continues until the general register specified by R_3 has been loaded.

Condition Code

♦ Unchanged.

Timing

♦ t (μsec) = 9 + 3.75R

where: R = number of general registers to be loaded.

Special Conditions

1. The number of the register specified by R_3 must be greater than or equal to the number of the register specified by R_1.
2. While only the low-order 16 bits are used in the general registers one through 15, a complete 32-bit word is loaded from storage into the general register.
3. The HSM location addressed by the B_2/D_2 address components must be on an even-word boundary.
4. General register zero (0) is the Timer register and uses the low-order 24 bits.

Example

Instruction

OP	R_1	R_3	B_2	D_2
98	03	05	02	0000

General register 2 contains $(00\ 00\ 07\ D0)_{16} = (2000)_{10}$.
General register 3 contains $(00\ 00\ 00\ 00)_{16}$.
General register 4 contains $(00\ 00\ 0D\ 60)_{16}$.
General register 5 contains $(00\ 00\ 0C\ 4D)_{16}$.
Location 2000–2003 contains $(00\ 00\ 50\ 00)_{16}$.
Location 2004–2007 contains $(00\ 00\ 65\ 00)_{16}$.
Location 2008–2011 contains $(00\ 00\ 05\ 00)_{16}$.

Example (Cont'd)

Execution of this instruction causes the following to occur:
General register 3 now contains $(00\ 00\ 50\ 00)_{16}$.
General register 4 now contains $(00\ 00\ 65\ 00)_{16}$.
General register 5 now contains $(00\ 00\ 05\ 00)_{16}$.

General Description

♦ This instruction stores a set of contiguous general registers, starting with the one specified by the first address and ending with the third address, into HSM locations starting with the second address.

Format

8	4	4	4	12
OP	R_1	R_3	B_2	D_2

OP — $(90)_{16}$

R_1 — specifies the first general register that is to be stored.

R_3 — specifies the last general register that is to be stored.

B_2/D_2 — HSM location where the first general register is to be stored.

Outline of Operation

♦ The contents of the general register, specified by B_2, are added to the contents of D_2 to obtain the location where the first general register, specified by R_1, is to be stored. The registers are stored in ascending order starting with R_1 and continuing through R_3. All 32 bits of the register are stored.

Condition Code

♦ Unchanged.

Timing

♦ t (μsec) = 9 + 3.75R
where: R = number of general registers to be stored.

Special Conditions

1. The number of the register specified by R_3 must be greater than or equal to the number of the register specied by R_1.
2. A complete 32-bit word is stored from each general register.
3. The HSM location addressed by the B_2/D_2 address components must be on an even-word boundary.

Example

Instruction

OP	R_1	R_3	B_2	D_2
90	03	05	02	0000

General register 2 contains $(00\ 00\ 07\ D0)_{16} = (2000)_{10}$.
General register 3 contains $(00\ 00\ 00\ 00)_{16}$.
General register 4 contains $(00\ 00\ 0D\ 60)_{16}$.
General register 5 contains $(00\ 00\ 0C\ 4D)_{16}$.
Location 2000–2003 contains $(00\ 00\ 50\ 00)_{16}$.
Location 2004–2007 contains $(00\ 00\ 65\ 00)_{16}$.
Location 2008–2011 contains $(00\ 00\ 05\ 00)_{16}$.

Execution of this instruction causes the following to occur:
Location 2000–2003 now contains $(00\ 00\ 00\ 00)_{16}$.
Location 2004–2007 now contains $(00\ 00\ 0D\ 60)_{16}$.
Location 2008–2011 now contains $(00\ 00\ 0C\ 4D)_{16}$.

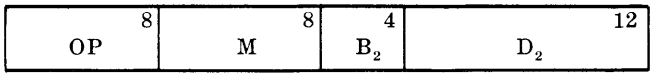
Halt and Branch (HB)

General Description

♦ This instruction stops the computer immediately. Depressing the START button causes control to transfer to the instruction specified by B_2/D_2.

Format

M



OP — $(81)_{16}$

M — any eight-bit byte to identify the halt.

B_2/D_2 — HSM address of the next instruction to be executed when the START button on the console is depressed.

Outline of Operation

♦ The computer is halted. Upon depression of the START button, operation continues at the address specified by the B_2/D_2 address.

Notes:

1. If this instruction is executed while input/output operations are in progress, the input/output operations will be completed before the computer comes to a halt.
2. The P counter is not loaded with the branch address until the START button is depressed. While the machine is halted, the P counter is loaded with the branch address of the byte following the Halt and Branch instruction.

Condition Code

♦ Unchanged.

Timing

♦ t (μsec) = 1.5

Read Forward (RDF)

General Description

♦ This instruction transfers information from the selected device, via the designated trunk, into high-speed memory.

Format

8	4	4	4	12	4	12
OP	T	U	B_1	D_1	B_2	D_2

OP — $(E5)_{16}$

*T — trunk number (0-7).

*U — device number.

B_1/D_1 — HSM location to receive the first byte transferred from the selected device.

B_2/D_2 — HSM location of the last byte to be transferred from the selected device.

Note: A Read Auxiliary instruction of the RCA 70/15 is treated by the 70/25 as a Read Forward instruction. The D_1 address, ignored by the 70/15, is used by the 70/25 to determine D_1 final. It will not cause an operation code trap interrupt.

Direction of Operation

♦ Left to right.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address which specifies the initial storage location of the information to be transferred. The contents of the general register, specified by B_2, are added to the contents of D_2 to obtain the B_2/D_2 address, which specifies the terminal address of the instruction.

The B_1/D_1 address is placed in the A register-I/0; the B_2/D_2 address is placed in the B register-I/0. The contents of the A register are incremented by one for each byte read until there is an A-B register equality, at which time the instruction terminates.

Upon completion of this instruction, the final D_1 address plus one will be available in the proper area reserved in memory for the particular trunk referenced. If the B register specifies an address greater than the actual transfer of information, the device terminates the instruction.

Condition Code

♦ 0 — instruction accepted.

1 — device inoperable.

2 — interrupt pending.

3 — reserved for future expansion.

Interrupt Action

♦ A termination interrupt occurs at the completion of this instruction provided that the interrupt mask for the addressed trunk is set to one (interrupt permitted).

---

* If this instruction is addressed to a device on the multiplexor channel, the format for T and U is as follows:

1. The leftmost bit of the T field is a one (1).
2. The remaining bits of the T field plus the U field (7 bits) designate the device number (13-127).

Read Reverse (RDR)

General Description

♦ This instruction transfers information from the selected device, via the designated trunk, into high-speed memory in a reverse direction.

Format

8	4	4	4	12	4	12
OP	T	U	B_1	D_1	B_2	D_2

OP — $(E2)_{16}$

*T — trunk number.

*U — device number.

B_1/D_1 — HSM location to receive the first byte transferred from the selected device.

B_2/D_2 — HSM location to receive the last byte transferred from the selected device.

Direction of Operation

♦ Right to left

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address which specifies the initial storage location of the information to be transferred. The contents of the general register, specified by B_2, are added to the contents of D_2 to obtain the B_2/D_2 address which specifies the terminal address of the instruction.

The B_1/D_1 address is placed in the A register-I/0; the B_2/D_2 address is placed in the B register-I/0. The contents of the A register are decremented by one for each byte read until there is A-B register equality, at which time the instruction terminates.

Upon completion of this instruction, the final D_1 address minus one will be available in the proper area reserved in memory for the particular trunk referenced. If the B register specifies an address less than the actual transfer of information, the device terminates the instruction.

Condition Code

♦ 0 — instruction accepted.

1 — device inoperable.

2 — interrupt pending.

3 — reserved for future expansion.

Interrupt Action

♦ A termination interrupt occurs at the completion of this instruction provided that the interrupt mask for the addressed trunk is set to one (interrupt permitted).

---

* If this instruction is addressed to a device on the multiplexor channel, the format for T and U is as follows:

1. The leftmost bit of the T field is a one (1).
2. The remaining bits of the T field plus the U field (7 bits) designate the device number (13-127).

Write (WR)

General Description

♦ This instruction transfers information from high-speed memory, via the designated trunk, to the selected device.

Format

8	4	4	4	12	4	12
OP	T	U	B_1	D_1	B_2	D_2

OP — $(E3)_{16}$

*T — trunk number.

*U — device number.

B_1/D_1 — HSM location of the first byte to be transferred to the selected device.

B_2/D_2 — HSM location of the last byte to be transferred to the selected device.

Direction of Operation

♦ Left to right.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address which specifies the initial location to be transferred. The contents of the general register, specified by B_2, are added to the contents of D_2 to obtain the B_2/D_2 address which specifies the terminating location.

The B_1/D_1 address is placed in the A register-I/0; the B_2/D_2 address is placed in the B register-I/0. The contents of the A register are incremented for each byte transferred until there is an A-B register equality, at which time the instruction terminates.

Upon completion of this instruction, the final D_1 address plus one will be available in the proper area reserved in memory for the particular trunk referenced.

Condition Code

♦ 0 — instruction accepted.

1 — device inoperable.

2 — interrupt pending.

3 — reserved for future expansion.

Interrupt Action

♦ A termination interrupt occurs at the completion of this instruction provided that the interrupt mask for the addressed trunk is set to one (interrupt permitted).

* If this instruction is addressed to a device on the multiplexor channel, the format for T and U is as follows:

1. The leftmost bit of the T field is a one (1).
2. The remaining bits of the T field plus the U field (7 bits) designate the device number (13-127).

Write Control (WRC)

General Description

♦ This instruction sends control information from high-speed memory, via the designated trunk, to the selected device. The exact control information sent is defined in the Spectra 70 input/output supplementary publications for the individual devices.

Format

8	4	4	4	4	12	12
OP	T	U	B_1	B_2	D_1	D_2

OP — $(E7)_{16}$

*T — trunk number.

*U — device number.

B_1/D_1 — HSM location of the initial byte containing the control information.

B_2/D_2 — HSM location of the last byte containing the control information.

Note: Bit configurations for particular control bytes are defined in the Spectra 70 I/0 supplementary publications for the individual devices.

Direction of Operation

♦ Left to right.

Outline of Operation

♦ The contents of the general register, specified by B_1, are added to the contents of D_1 to obtain the B_1/D_1 address which is the starting location of the control information. The contents of the general register, specified by B_2, are added to the contents of D_2 to obtain the B_2/D_2 address which is the terminating location of control information.

The B_1/D_1 address is placed in the A register-I/0; the B_2/D_2 address is placed in the B register-I/0. The contents of the A register are incremented by one for each byte transferred until there is an A-B register equality, at which time the instruction terminates. The device terminates this instruction if it receives the required number of control bytes prior to A-B register equality.

Upon completion of this instruction, the final D_1 address plus one will be available in the proper area reserved in memory for the particular trunk referenced.

Condition Code

♦ 0 — instruction accepted.

1 — device inoperable.

2 — interrupt pending.

3 — reserved for future expansion.

Interrupt Action

♦ A termination interrupt occurs at the completion of this instruction provided that the interrupt mask for the addressed trunk is set to one (interrupt permitted).

---

* If this instruction is addressed to a device on the multiplexor channel, the format for T and U is as follows:

1. The leftmost bit of the T field is a one (1).
2. The remaining bits of the T field plus the U field (7 bits) designate the device number (13-127).

Write Erase (WRE)

General Description

♦ This instruction transfers complete blanks to tape. It may be used to skip over tape flaws and is readily incorporated into rollback routines.

Format

8	4	4	4	12	4	12
OP	T	U	B_1	D_1	B_2	D_2

OP — $(E4)_{16}$

*T — trunk number.

*U — device number.

B_1/D_1 — HSM location of the first byte to be used to erase tape.

B_2/D_2 — HSM location of the last byte to be used to erase tape.

Direction of Operation

♦ Left to right.

Outline of Operation

♦ Complete blanks, whose length is determined by the difference between the B_1/D_1 and the B_2/D_2 addresses, are transferred to tape.

Upon completion of this instruction the final D_1 address plus one is stored in the reserved area in memory for the referenced trunk. It is not necessary to have the respective memory locations contain blanks since this instruction actually creates a gap by degaussing the tape.

Condition Code

♦ 0 — instruction accepted.

1 — device inoperable.

2 — interrupt pending.

3 — reserved for future expansion.

Interrupt Action

♦ A termination interrupt occurs at the completion of this instruction provided that the interrupt mask for the addressed trunk is set to one (interrupt permitted).

---

* If this instruction is addressed to a device on the multiplexor channel, the format for T and U is as follows:

1. The leftmost bit of the T field is a one (1).
2. The remaining bits of the T field plus the U field (7 bits) designate the device number (13-127).

Sense (IOS)

General Description

♦ This instruction places status information in high-speed memory. The exact status sent is defined in the input/output supplements for the individual units.

Format

8	4	4	4	12	4	12
OP	T	U	B_1	D_1	B_2	D_2

OP — $(E1)_{16}$

*T — trunk number (0-7).

*U — device number.

B_1/D_1 — HSM location to receive first byte of status information.

B_2/D_2 — HSM location to receive last byte of status information.

Direction of Operation

♦ Left to right.

Outline of Operation

♦ When the selected device is available (control and device are not busy) the Sense instruction is performed and status information about the selected device is transferred to high-speed memory. The Sense instruction resets all bits in the sense byte and 2^1 and 2^2 bits of the standard device byte. The Sense instruction does not cause a termination interrupt. No processing is allowed for the duration of the Sense instruction. Upon completion of this instruction, the final D_1 address plus one is available in the proper area reserved in memory for the particular trunk referenced.

Notes:

1. If an interrupt is pending on the addressed trunk, this instruction is not executed.
2. If the device addressed fails to respond (inoperable), this instruction is not executed. If the device is mechanically inoperable only, this instruction is executed.

Condition Code

♦ 0 — instruction accepted.

1 — device inoperable.

2 — interrupt pending.

3 — reserved for future expansion.

---

* If this instruction is addressed to a device on the multiplexor channel, the format for T and U is as follows:
  1. The leftmost bit of the T field is a one (1).
  2. The remaining bits of the T field plus the U field (7 bits) designate the device number (13-127).

Post Status (PS)

General Description

♦ This instruction stores the standard device byte in the input/output reserved memory location for the selected trunk. The standard device byte provides information to the programmer as to the status of the device upon completion of an input/output operation.

Format

8	4	4	4	12
OP	T	U	B_2	D_2

OP — $(66)_{16}$

*T — trunk number (0-7).

*U — device number.

Definition of standard device byte:

2^0 — not applicable.
2^1 — device inoperable.
2^2 — secondary indicator.
2^3 — device end.
2^4 — not applicable.
2^5 — not applicable.
2^6 — interrupt pending (termination).
2^7 — external device request interrupt pending.

B_2/D_2 — HSM location of next instruction to be executed if the device referenced is busy.

Direction of Operation

♦ Not applicable.

Outline of Operation

♦ The standard device byte is sent to the reserved location for the addressed trunk for interrogation by the programmer. Post Status does not cause a termination interrupt. No processing is allowed for the duration of the Post Status instruction.

Notes:

1. This instruction is performed if the device addressed is inoperable.
2. This instruction resets the 2^6 and 2^7 bits of the standard device byte.

Condition Code

♦ 0 — instruction accepted.

1 — device inoperable.

2 — interrupt pending.

3 — reserved for future expansion.

---

* If this instruction is addressed to a device on the multiplexor channel, the format for T and U is as follows:
  1. The leftmost bit of the T field is a one (1).
  2. The remaining bits of the T field plus the U field (7 bits) designate the device number (13-127).

APPENDIX A—SUMMARY OF INSTRUCTIONS

Data Handling, Arithmetic and Decision Instructions

0 7			8 15		16 31	32 47				
Op	Mnemonic	Instruction	L		B_1/D_1	B_2/D_2	Cond. Code	Condition Code Settings	Operation	Page Ref.
			8 11 L_1	12 15 L_2						
D2	MVC	Move	Length of the field addressed by B_1/D_1 and B_2/D_2		Leftmost address of receiving field	Leftmost address of sending field	No		L to R	21
DE	ED	Edit	Length of the field addressed by B_1/D_1 and B_2/D_2		Leftmost address of edit mask and result	Leftmost address of field to be edited	Yes	0 — Result = Zero 1 — Result < Zero 2 — Result > Zero 3 — Not used	L to R	22
F2	PACK	Pack	Length of the field addressed by B_1/D_1	Length of the field addressed by B_2/D_2	Leftmost address of field to receive packed data	Leftmost address of zoned data field	No		R to L	24
F3	UNPK	Unpack	Length of the field addressed by B_1/D_1	Length of the field addressed by B_2/D_2	Leftmost address of field to receive zoned data	Leftmost address of packed data field	No		R to L	26
DC	TR	Translate	Length of the field addressed by B_1/D_1 and B_2/D_2		Leftmost address of field to be translated and result	Leftmost address of translate table	No		L to R	28
FA	AP	Add Decimal	Length of the field addressed by B_1/D_1	Length of the field addressed by B_2/D_2	Leftmost address of augend and sum	Leftmost address of addend	Yes	0 — Sum = Zero 1 — Sum < Zero 2 — Sum > Zero 3 — Overflow	R to L	30
F6	AB	Add Binary	Length of the field addressed by B_1/D_1	Length of the field addressed by B_2/D_2	Leftmost address of augend and sum	Leftmost address of addend	Yes	0 — Sum = Zero 1 — Not used 2 — Sum > Zero 3 — Overflow	R to L	32
FB	SP	Subtract Decimal	Length of the field addressed by B_1/D_1	Length of the field addressed by B_2/D_2	Leftmost address of minuend and difference	Leftmost address of subtrahend	Yes	0 — Diff = Zero 1 — Diff < Zero 2 — Diff > Zero 3 — Overflow	R to L	34

Notes:

1. Length is always the total number of bytes minus one.
2. All bits not used must be zeros.

APPENDIX A — SUMMARY OF INSTRUCTIONS

Data Handling, Arithmetic and Decision Instructions (Cont'd)

0 7 8 15 16 31 32 47

Op	Mnemonic	Instruction	L		B_1/D_1	B_2/D_2	Cond. Code	Condition Code Settings	Operation	Page Ref.
			8 11 L_1	12 15 L_2						
F7	SB	Subtract Binary	Length of the field addressed by B_1/D_1	Length of the field addressed by B_2/D_2	Leftmost address of minuend and difference	Leftmost address of subtrahend	Yes	0 — Diff = Zero 1 — Diff < Zero 2 — Diff > Zero 3 — Not used	R to L	36
FC	MP	Multiply Decimal	Length of the field addressed by B_1/D_1	Length of the field addressed by B_2/D_2	Leftmost address of multiplicand and product	Leftmost address of multiplier	No		R to L	38
FD	DP	Divide Decimal	Length of the field addressed by B_1/D_1	Length of the field addressed by B_2/D_2	Leftmost address of dividend and quotient	Leftmost address of divisor	No		R to L	40
D4	NC	Logical And	Length of the field addressed by B_1/D_1 and B_2/D_2		Leftmost address of first operand and result	Leftmost address of second operand	Yes	0 — Result = Zero 1 — Result ≠ Zero 2 — Not used 3 — Not used	L to R	42
D6	OC	Logical Or	Length of the field addressed by B_1/D_1 and B_2/D_2		Leftmost address of first operand and result	Leftmost address of second operand	Yes	0 — Result = Zero 1 — Result ≠ Zero 2 — Not used 3 — Not used	L to R	43
D7	XC	Exclusive Or	Length of the field addressed by B_1/D_1 and B_2/D_2		Leftmost address of first operand and result	Leftmost address of second operand	Yes	0 — Result = zero 1 — Result ≠ Zero 2 — Not used 3 — Not used	L to R	44
F9	CP	Compare Decimal	Length of the field addressed by B_1/D_1	Length of the field addressed by B_2/D_2	Leftmost address of first operand	Leftmost address of second operand	Yes	0 — $D_1 = D_2$ 1 — $D_1 < D_2$ 2 — $D_1 > D_2$ 3 — Not used	R to L	50
D5	CLC	Compare Logical	Length of the field addressed by B_1/D_1 and B_2/D_2		Leftmost address of first operand	Leftmost address of second operand	Yes	0 — $D_1 = D_2$ 1 — $D_1 < D_2$ 2 — $D_1 > D_2$ 3 — Not used	L to R	52

Notes:

1. Length is always the total number of bytes minus one.
2. All bits not used must be zeros.

APPENDIX A—SUMMARY OF INSTRUCTIONS

Control Instructions

0 7 8 15 16 31

Op	Mnemonic	Instruction	M		B_1/D_1	Cond. Code	Condition Code Settings	Page Ref.
			R_1	R_X				
47	BC	Branch on Condition	High-order four bits specify condition code to be tested		Address of next Instruction if condition is present	No		45
45	BAL	Branch and Link	High-order four bits specify the register that will contain contents of P counter		Address of the next instruction to be executed	No		46
05	BALR	Branch and Link	Register in which the P counter is to be stored	Register that contains the address of the next instruction to be executed				47
46	BCT	Branch on Count	High-order four bits specify the register that contains the count		Address of the next instruction if count is *not* zero	No		48
82	STP2	Set P_2 Register	Not used		Value to be placed in P_2 counter	No		54
91	TM	Test Under Mask	Mask to be compared against data		Address of data byte to be compared	Yes	0 — Selected bits all zero; Mask all zero 1 — Selected bits both zero and one 3 — Selected bits all one	55
98	LM	Load Multiple	Address of first register to be loaded	Address of last register to be loaded	Leftmost address of the locations to be loaded into the registers	No		56
90	STM	Store Multiple	Address of first register to be stored	Address of last register to be stored	Leftmost address of the locations to receive the contents of the registers	No		58
81	HB	Halt and Branch	Any eight-bit code to identify the halt		Address of next instruction to be executed when Start button is depressed	No		59

Note:

All bits not used must be zeros.

APPENDIX A — SUMMARY OF INSTRUCTIONS

Input/Output Instructions

0 7 | 8 11 | 12 15 | 16 31 | 32 47

Op	Mnemonic	Instruction	T	U	B_1/D_1	B_2/D_2	Cond. Code	Cond. Code Settings	Remarks	Operation	Page Ref.
E5	RDF	Read Forward	Trunk number	Device number	Address to receive first character	Address to receive last character	Yes	0 — Instruction accepted 1 — Device inoperable 2 — Interrupt pending 3 — Not used		L to R	60
E2	RDR	Read Reverse	Trunk number	Device number	Address to receive first character	Address to receive last character	Yes	0 — Instruction accepted 1 — Device inoperable 2 — Interrupt pending 3 — Not used		R to L	61
E3	WR	Write	Trunk number	Device number	Address of first character to be written	Address of last character to be written	Yes	0 — Instruction accepted 1 — Device inoperable 2 — Interrupt pending 3 — Not used		L to R	62
E7	WRC	Write Control	Trunk number	Device number	Address of first character to be written	Address of last character to be written	Yes	0 — Instruction accepted 1 — Device inoperable 2 — Interrupt pending 3 — Not used		L to R	63
E4	WRE	Write Erase	Trunk number	Device number	Address of first character to be erased	Address of last character to be erased	Yes	0 — Instruction accepted 1 — Device inoperable 2 — Interrupt pending 3 — Not used	Length of erase determined by difference between B_1/D_1 and B_2/D_2	L to R	64
E1	IOS	Sense	Trunk number	Device number	Address to receive first character of status information	Address to receive last character of status information	Yes	0 — Instruction accepted 1 — Device inoperable 2 — Interrupt pending 3 — Not used		L to R	65
66	PS	Post Status	Trunk number	Device number	Address of next instruction to be executed if selected device is busy	Not used	Yes	0 — Instruction accepted 1 — Device inoperable 2 — Interrupt pending 3 — Not used		Not Applicable	66

Note:

All bits not used must be zeros.

APPENDIX B

Instruction Timing Summary

Instruction	Staticizing (μs)	Execution (μs)
DATA HANDLING INSTRUCTIONS		
Edit	13.5	1.5 (2I + 2F + 2.5D)
Move	13.5	3W + 3B
Pack	13.5	$6 + 1.5N_1 + 3N_2$
Unpack	13.5	$7.5 + 3N_1 + 1.5N_2$
Translate	13.5	6.75N
ARITHMETIC INSTRUCTIONS		
Add Binary	13.5	$8.25 + 2.25N_1 + 1.5N_2$
Add Decimal	13.5	$8.25 + 2.25N_1 + 1.5N_2$
Subtract Binary	13.5	$8.25 + 2.25N_1 + 1.5N_2$
Subtract Decimal	13.5	$8.25 + 2.25N_1 + 1.5N_2$
Multiply Decimal	13.5	$12.75 + 9N_1 - 1.5N_2 +$ C $[3.75\ (N_1 - N_2) + 3]$
Divide Decimal	13.5	$9 + 29.25N_1 - 27N_2 +$ $37.5N_2\ (N_1 - N_2)$
Logical AND	13.5	3.75N
Logical OR	13.5	3.75N
Exclusive OR	13.5	3.75N
DECISION AND CONTROL INSTRUCTIONS		
Branch and Link (BAL)	9	3.75
Branch and Link (BALR)	4.5	Branch = 6; No Branch = 2.25
Branch On Condition	9	Branch = 2.25; No Branch = 0.5
Branch On Count	9	Branch = 6.75; No Branch = 2.25
Compare Decimal	13.5	$6 + 1.5N_1 + 2.25N_2$
Compare Logical	13.5	1.5 + 3B
Halt and Branch	9	1.5
Load Multiple	9	3.75R
Set P_2 Register	9	3
Store Multiple	9	3.75R
Test Under Mask	9	1.5
INPUT/OUTPUT INSTRUCTIONS		
Post Status	9	Branch = 2.25; No Branch = 0.5
Write Erase	13.5	Refer to Spectra 70 input/output supplementary publications for additional timing information
Read Forward	13.5	
Read Reverse	13.5	
Sense	13.5	
Write	13.5	
Write Control	13.5	

Legend:

B — number of bytes processed (or number of bytes outside full word boundaries).
C — sum of value of multiplier digits.
D — number of digits inserted.
F — number of fill characters inserted.
I — number of edit symbols inserted.
N — total number of bytes.
N_1 — number of bytes in first operand.
N_2 — number of bytes in second operand.
R — number of registers.
W — number of four-byte words.

APPENDIX C

Reserved Memory Locations

Location (Byte)		Use
Decimal	Hexadecimal	
0000–0003	0000–0003	Status of channel 0
0004–0007	0004–0007	Status of channel 1
0008–0011	0008–000B	Status of channel 2
0012–0015	000C–000F	Status of channel 3
0016–0019	0010–0013	Status of channel 4
0020–0023	0014–0017	Status of channel 5
0024–0027	0018–001B	Status of channel 6
0028–0031	001C–001F	Status of channel 7
0032–0039	0020–0027	Reserved for use by the processor
0040–0041	0028–0029	Program counter for Processing State (P_1)
0042	002A	Operation code storage
0043	002B	Condition code storage
0044–0045	002C–002D	Program counter for Interrupt State (P_2)
0046–0047	002E–002F	Interrupt identification
0048–0049	0030–0031	Interrupt mask

Upper Memory (Byte 0 = last byte in memory)

0–59 — 15 general purpose registers

60–63 — Timer register

64–103 — Reserved for use by the processor

104 + — Multiplexor device status information (if multiplexor is present) — 8 bytes per device

APPENDIX D
EXTENDED BINARY-CODED-DECIMAL INTERCHANGE CODE (EBCDIC)

BIT POSITIONS→01 23 / →4567	00 00	00 01	00 10	00 11	01 00	01 01	01 10	01 11	10 00	10 01	10 10	10 11	11 00	11 01	11 10	11 11
0000	NULL				SP	&	—									0
0001							/		a	j			A	J		1
0010									b	k	s		B	K	S	2
0011									c	l	t		C	L	T	3
0100	PF	RES	BYP	PN					d	m	u		D	M	U	4
0101	HT	NL	LF	RS					e	n	v		E	N	V	5
0110	LC	BS	EOB	UC					f	o	w		F	O	W	6
0111	DEL	IL	PRE	EOT					g	p	x		G	P	X	7
1000									h	q	y		H	Q	Y	8
1001									i	r	z		I	R	Z	9
1010			SM		¢	!		:								
1011					.	$	,	#								
1100					<	*	%	@								
1101					(	)	—	'								
1110					+	;	>	=								
1111					|	¬	?	"								

Bit Positions: 0 1 2 3 4 5 6 7

Significance: 2^7 2^6 2^5 2^4 2^3 2^2 2^1 2^0

Note:
Chart is read by order of significance as designated by "Bit Positions," i.e., 0 is 2^7 bit, 1 is 2^6 bit . . . etc.

For example:
E is 11 00 0101

Control Characters:

NULL — All Zero-Bits
PF — Punch Off
HT — Horizontal Tab
LC — Lower Case
DEL — Delete
RES — Restore
NL — New Line
BS — Backspace
IL — Idle
BYP — Bypass
LF — Line Feed
EOB — End of Block
PRE — Prefix
PN — Punch On
RS — Reader Stop
UC — Upper Case
EOT — End of Transmission
SM — Set Mode
SP — Space

APPENDIX E
CHARACTER CODES

8-Bit BCD Code	Character Set Punch Combination	Printer Graphics	Decimal	Hexadecimal
00000000	12,0,9,8,1		0	00
00000001	12,9,1		1	01
00000010	12,9,2		2	02
00000011	12,9,3		3	03
00000100	12,9,4		4	04
00000101	12,9,5		5	05
00000110	12,9,6		6	06
00000111	12,9,7		7	07
00001000	12,9,8		8	08
00001001	12,9,8,1		9	09
00001010	12,9,8,2		10	0A
00001011	12,9,8,3		11	0B
00001100	12,9,8,4		12	0C
00001101	12,9,8,5		13	0D
00001110	12,9,8,6		14	0E
00001111	12,9,8,7		15	0F
00010000	12,11,9,8,1		16	10
00010001	11,9,1		17	11
00010010	11,9,2		18	12
00010011	11,9,3		19	13
00010100	11,9,4		20	14
00010101	11,9,5		21	15
00010110	11,9,6		22	16
00010111	11,9,7		23	17
00011000	11,9,8		24	18
00011001	11,9,8,1		25	19
00011010	11,9,8,2		26	1A
00011011	11,9,8,3		27	1B
00011100	11,9,8,4		28	1C
00011101	11,9,8,5		29	1D
00011110	11,9,8,6		30	1E
00011111	11,9,8,7		31	1F
00100000	11,0,9,8,1		32	20
00100001	0,9,1		33	21
00100010	0,9,2		34	22
00100011	0,9,3		35	23
00100100	0,9,4		36	24
00100101	0,9,5		37	25
00100110	0,9,6		38	26
00100111	0,9,7		39	27
00101000	0,9,8		40	28
00101001	0,9,8,1		41	29
00101010	0,9,8,2		42	2A
00101011	0,9,8,3		43	2B
00101100	0,9,8,4		44	2C
00101101	0,9,8,5		45	2D
00101110	0,9,8,6		46	2E
00101111	0,9,8,7		47	2F
00110000	12,11,0,9,8,1		48	30
00110001	9,1		49	31
00110010	9,2		50	32
00110011	9,3		51	33
00110100	9,4		52	34
00110101	9,5		53	35
00110110	9,6		54	36
00110111	9,7		55	37
00111000	9,8		56	38
00111001	9,8,1		57	39
00111010	9,8,2		58	3A
00111011	9,8,3		59	3B
00111100	9,8,4		60	3C
00111101	9,8,5		61	3D
00111110	9,8,6		62	3E
00111111	9,8,7		63	3F
01000000		space	64	40
01000001	12,0,9,1		65	41
01000010	12,0,9,2		66	42
01000011	12,0,9,3		67	43
01000100	12,0,9,4		68	44
01000101	12,0,9,5		69	45
01000110	12,0,9,6		70	46
01000111	12,0,9,7		71	47
01001000	12,0,9,8		72	48
01001001	12,8,1		73	49
01001010	12,8,2	¢ (cents)	74	4A
01001011	12,8,3	. (period)	75	4B

APPENDIX E

CHARACTER CODES (Cont'd)

8-Bit BCD Code	Character Set Punch Combination	Printer Graphics	Decimal	Hexadecimal
01001100	12,8,4	< (Less than)	76	4C
01001101	12,8,5	((left parens)	77	4D
01001110	12,8,6	+ (plus)	78	4E
01001111	12,8,7	| (stroke)	79	4F
01010000	12	& (ampersand)	80	50
01010001	12,11,9,1		81	51
01010010	12,11,9,2		82	52
01010011	12,11,9,3		83	53
01010100	12,11,9,4		84	54
01010101	12,11,9,5		85	55
01010110	12,11,9,6		86	56
01010111	12,11,9,7		87	57
01011000	12,11,9,8		88	58
01011001	11,8,1	∧ (logical AND)	89	59
01011010	11,8,2	! (exclamation)	90	5A
01011011	11,8,3	$ (dollar sign)	91	5B
01011100	11,8,4	* (asterisk)	92	5C
01011101	11,8,5	) (right parens)	93	5D
01011110	11,8,6	; (semicolon)	94	5E
01011111	11,8,7	¬ (logical NOT)	95	5F
01100000	11	– (minus)	96	60
01100001	0,1	/ (virgule)	97	61
01100010	11,0,9,2		98	62
01100011	11,0,9,3		99	63
01100100	11,0,9,4		100	64
01100101	11,0,9,5		101	65
01100110	11,0,9,6		102	66
01100111	11,0,9,7		103	67
01101000	11,0,9,8		104	68
01101001	0,8,1		105	69
01101010	12,11		106	6A
01101011	0,8,3	, (comma)	107	6B
01101100	0,8,4	% (percent)	108	6C
01101101	0,8,5	_ (underline)	109	6D
01101110	0,8,6	> (greater than)	110	6E
01101111	0,8,7	? (question mark)	111	6F
01110000	12,11,0		112	70
01110001	12,11,0,9,1		113	71
01110010	12,11,0,9,2		114	72
01110011	12,11,0,9,3		115	73
01110100	12,11,0,9,4		116	74
01110101	12,11,0,9,5		117	75
01110110	12,11,0,9,6		118	76
01110111	12,11,0,9,7		119	77
01111000	12,11,0,9,8		120	78
01111001	8,1		121	79
01111010	8,2	: (colon)	122	7A
01111011	8,3	# (number)	123	7B
01111100	8,4	@ (at rate of)	124	7C
01111101	8,5	' (apostrophe)	125	7D
01111110	8,6	= (equal)	126	7E
01111111	8,7	" (quotes)	127	7F
10000000	12,0,8,1		128	80
10000001	12,0,1		129	81
10000010	12,0,2		130	82
10000011	12,0,3		131	83
10000100	12,0,4		132	84
10000101	12,0,5		133	85
10000110	12,0,6		134	86
10000111	12,0,7		135	87
10001000	12,0,8		136	88
10001001	12,0,9		137	89
10001010	12,0,8,2		138	8A
10001011	12,0,8,3		139	8B
10001100	12,0,8,4		140	8C
10001101	12,0,8,5		141	8D
10001110	12,0,8,6		142	8E
10001111	12,0,8,7		143	8F
10010000	12,11,8,1		144	90
10010001	12,11,1		145	91
10010010	12,11,2		146	92
10010011	12,11,3		147	93
10010100	12,11,4		148	94
10010101	12,11,5		149	95
10010110	12,11,6		150	96
10010111	12,11,7		151	97

APPENDIX E
CHARACTER CODES (Cont'd)

8-Bit BCD Code	Character Set Punch Combination	Printer Graphics	Decimal	Hexadecimal
10011000	12,11,8		152	98
10011001	12,11,9		153	99
10011010	12,11,8,2		154	9A
10011011	12,11,8,3		155	9B
10011100	12,11,8,4		156	9C
10011101	12,11,8,5		157	9D
10011110	12,11,8,6		158	9E
10011111	12,11,8,7		159	9F
10100000	11,0,8,1		160	A0
10100001	11,0,1		161	A1
10100010	11,0,2		162	A2
10100011	11,0,3		163	A3
10100100	11,0,4		164	A4
10100101	11,0,5		165	A5
10100110	11,0,6		166	A6
10100111	11,0,7		167	A7
10101000	11,0,8		168	A8
10101001	11,0,9		169	A9
10101010	11,0,8,2		170	AA
10101011	11,0,8,3		171	AB
10101100	11,0,8,4		172	AC
10101101	11,0,8,5		173	AD
10101110	11,0,8,6		174	AE
10101111	11,0,8,7		175	AF
10110000	12,11,0,8,1		176	B0
10110001	12,11,0,1		177	B1
10110010	12,11,0,2		178	B2
10110011	12,11,0,3		179	B3
10110100	12,11,0,4		180	B4
10110101	12,11,0,5		181	B5
10110110	12,11,0,6		182	B6
10110111	12,11,0,7		183	B7
10111000	12,11,0,8		184	B8
10111001	12,11,0,9		185	B9
10111010	12,11,0,8,2		186	BA
10111011	12,11,0,8,3		187	BB
10111100	12,11,0,8,4		188	BC
10111101	12,11,0,8,5		189	BD
10111110	12,11,0,8,6		190	BE
10111111	12,11,0,8,7		191	BF
11000000	12,0		192	C0
11000001	12,1	A	193	C1
11000010	12,2	B	194	C2
11000011	12,3	C	195	C3
11000100	12,4	D	196	C4
11000101	12,5	E	197	C5
11000110	12,6	F	198	C6
11000111	12,7	G	199	C7
11001000	12,8	H	200	C8
11001001	12,9	I	201	C9
11001010	12,0,9,8,2		202	CA
11001011	12,0,9,8,3		203	CB
11001100	12,0,9,8,4		204	CC
11001101	12,0,9,8,5		205	CD
11001110	12,0,9,8,6		206	CE
11001111	12,0,9,8,7		207	CF
11010000	11,0		208	D0
11010001	11,1	J	209	D1
11010010	11,2	K	210	D2
11010011	11,3	L	211	D3
11010100	11,4	M	212	D4
11010101	11,5	N	213	D5
11010110	11,6	O	214	D6
11010111	11,7	P	215	D7
11011000	11,8	Q	216	D8
11011001	11,9	R	217	D9
11011010	12,11,9,8,2		218	DA
11011011	12,11,9,8,3		219	DB
11011100	12,11,9,8,4		220	DC
11011101	12,11,9,8,5		221	DD
11011110	12,11,9,8,6		222	DE
11011111	12,11,9,8,7		223	DF
11100000	0,8,2	Blank	224	E0
11100001	11,0,9,1		225	E1
11100010	0,2	S	226	E2
11100011	0,3	T	227	E3

APPENDIX E
CHARACTER CODES (Cont'd)

8-Bit BCD Code	Character Set Punch Combination	Printer Graphics	Decimal	Hexadecimal
11100100	0,4	U	228	E4
11100101	0,5	V	229	E5
11100110	0,6	W	230	E6
11100111	0,7	X	231	E7
11101000	0,8	Y	232	E8
11101001	0,9	Z	233	E9
11101010	11,0,9,8,2		234	EA
11101011	11,0,9,8,3		235	EB
11101100	11,0,9,8,4		236	EC
11101101	11,0,9,8,5		237	ED
11101110	11,0,9,8,6		238	EE
11101111	11,0,9,8,7		239	EF
11110000	0	0	240	F0
11110001	1	1	241	F1
11110010	2	2	242	F2
11110011	3	3	243	F3
11110100	4	4	244	F4
11110101	5	5	245	F5
11110110	6	6	246	F6
11110111	7	7	247	F7
11111000	8	8	248	F8
11111001	9	9	249	F9
11111010	12,11,0,9,8,2		250	FA
11111011	12,11,0,9,8,3		251	FB
11111100	12,11,0,9,8,4		252	FC
11111101	12,11,0,9,8,5		253	FD
11111110	12,11,0,9,8,6		254	FE
11111111	12,11,0,9,8,7	(lozenge) ¤	255	FF

APPENDIX F

Power of Two Table

N	2^N
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1,024
11	2,048
12	4,096
13	8,192
14	16,384
15	32,768
16	65,536

APPENDIX G
HEXADECIMAL-DECIMAL NUMBER CONVERSION

♦ This Appendix contains the necessary reference information for the conversion of decimal numbers to hexadecimal numbers and the conversion of binary numbers to decimal or hexadecimal.

Example #1 $(00111010)_2 = (3A)_{16} = (58)_{10}$

Example #2 $(FC)_{16} = (11111100)_2 = (252)_{10}$

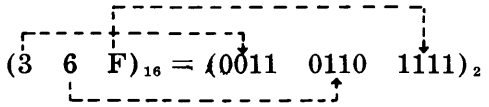
In the conversion of a hexadecimal number to its decimal value the marks (0-F) represent a multiplier and their position (reading right to left) within the hexadecimal number represent the exponent of the base. Each mark is multiplied by the base raised to the appropriate power and the summation of their product is the decimal value of the number.

Example #3

$$(36F)_{16} = 3\ (16^2) + 6\ (16^1) + 15\ (16^0)$$
$$\qquad\qquad\qquad\qquad\qquad\qquad\quad F$$

$$(36F)_{16} = 3\ (256) + 6\ (16) + 15\ (1) = (879)_{10}$$

To convert hexadecimal to binary substitute the binary equivalent of the hexadecimal mark into its appropriate position as follows:



HEXADECIMAL-DECIMAL NUMBER CONVERSION TABLE

♦ The table in this Appendix provides for direct conversion of decimal and hexadecimal numbers in these ranges:

Hexadecimal	*Decimal*
000 to FFF	0000 to 4095

For numbers outside the range of the table, add the following values to the table figures:

Hexadecimal	*Decimal*
1000	4096
2000	8192
3000	12288
4000	16384
5000	20480
6000	24576
7000	28672
8000	32768
9000	36864
A000	40960
B000	45056
C000	49152
D000	53248
E000	57344
F000	61440

HEXADECIMAL-DECIMAL NUMBER CONVERSION TABLE (Cont'd)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	0000	0001	0002	0003	0004	0005	0006	0007	0008	0009	0010	0011	0012	0013	0014	0015
01	0016	0017	0018	0019	0020	0021	0022	0023	0024	0025	0026	0027	0028	0029	0030	0031
02	0032	0033	0034	0035	0036	0037	0038	0039	0040	0041	0042	0043	0044	0045	0046	0047
03	0048	0049	0050	0051	0052	0053	0054	0055	0056	0057	0058	0059	0060	0061	0062	0063
04	0064	0065	0066	0067	0068	0069	0070	0071	0072	0073	0074	0075	0076	0077	0078	0079
05	0080	0081	0082	0083	0084	0085	0086	0087	0088	0089	0090	0091	0092	0093	0094	0095
06	0096	0097	0098	0099	0100	0101	0102	0103	0104	0105	0106	0107	0108	0109	0110	0111
07	0112	0113	0114	0115	0116	0117	0118	0119	0120	0121	0122	0123	0124	0125	0126	0127
08	0128	0129	0130	0131	0132	0133	0134	0135	0136	0137	0138	0139	0140	0141	0142	0143
09	0144	0145	0146	0147	0148	0149	0150	0151	0152	0153	0154	0155	0156	0157	0158	0159
0A	0160	0161	0162	0163	0164	0165	0166	0167	0168	0169	0170	0171	0172	0173	0174	0175
0B	0176	0177	0178	0179	0180	0181	0182	0183	0184	0185	0186	0187	0188	0189	0190	0191
0C	0192	0193	0194	0195	0196	0197	0198	0199	0200	0201	0202	0203	0204	0205	0206	0207
0D	0208	0209	0210	0211	0212	0213	0214	0215	0216	0217	0218	0219	0220	0221	0222	0223
0E	0224	0225	0226	0227	0228	0229	0230	0231	0232	0233	0234	0235	0236	0237	0238	0239
0F	0240	0241	0242	0243	0244	0245	0246	0247	0248	0249	0250	0251	0252	0253	0254	0255
10	0256	0257	0258	0259	0260	0261	0262	0263	0264	0265	0266	0267	0268	0269	0270	0271
11	0272	0273	0274	0275	0276	0277	0278	0279	0280	0281	0282	0283	0284	0285	0286	0287
12	0288	0289	0290	0291	0292	0293	0294	0295	0296	0297	0298	0299	0300	0301	0302	0303
13	0304	0305	0306	0307	0308	0309	0310	0311	0312	0313	0314	0315	0316	0317	0318	0319
14	0320	0321	0322	0323	0324	0325	0326	0327	0328	0329	0330	0331	0332	0333	0334	0335
15	0336	0337	0338	0339	0340	0341	0342	0343	0344	0345	0346	0347	0348	0349	0350	0351
16	0352	0353	0354	0355	0356	0357	0358	0359	0360	0361	0362	0363	0364	0365	0366	0367
17	0368	0369	0370	0371	0372	0373	0374	0375	0376	0377	0378	0379	0380	0381	0382	0383
18	0384	0385	0386	0387	0388	0389	0390	0391	0392	0393	0394	0395	0396	0397	0398	0399
19	0400	0401	0402	0403	0404	0405	0406	0407	0408	0409	0410	0411	0412	0413	0414	0415
1A	0416	0417	0418	0419	0420	0421	0422	0423	0424	0425	0426	0427	0428	0429	0430	0431
1B	0432	0433	0434	0435	0436	0437	0438	0439	0440	0441	0442	0443	0444	0445	0446	0447
1C	0448	0449	0450	0451	0452	0453	0454	0455	0456	0457	0458	0459	0460	0461	0462	0463
1D	0464	0465	0466	0467	0468	0469	0470	0471	0472	0473	0474	0475	0476	0477	0478	0479
1E	0480	0481	0482	0483	0484	0485	0486	0487	0488	0489	0490	0491	0492	0493	0494	0495
1F	0496	0497	0498	0499	0500	0501	0502	0503	0504	0505	0506	0507	0508	0509	0510	0511
20	0512	0513	0514	0515	0516	0517	0518	0519	0520	0521	0522	0523	0524	0525	0526	0527
21	0528	0529	0530	0531	0532	0533	0534	0535	0536	0537	0538	0539	0540	0541	0542	0543
22	0544	0545	0546	0547	0548	0549	0550	0551	0552	0553	0554	0555	0556	0557	0558	0559
23	0560	0561	0562	0563	0564	0565	0566	0567	0568	0569	0570	0571	0572	0573	0574	0575
24	0576	0577	0578	0579	0580	0581	0582	0583	0584	0585	0586	0587	0588	0589	0590	0591
25	0592	0593	0594	0595	0596	0597	0598	0599	0600	0601	0602	0603	0604	0605	0606	0607
26	0608	0609	0610	0611	0612	0613	0614	0615	0616	0617	0618	0619	0620	0621	0622	0623
27	0624	0625	0626	0627	0628	0629	0630	0631	0632	0633	0634	0635	0636	0637	0638	0639
28	0640	0641	0642	0643	0644	0645	0646	0647	0648	0649	0650	0651	0652	0653	0654	0655
29	0656	0657	0658	0659	0660	0661	0662	0663	0664	0665	0666	0667	0668	0669	0670	0671
2A	0672	0673	0674	0675	0676	0677	0678	0679	0680	0681	0682	0683	0684	0685	0686	0687
2B	0688	0689	0690	0691	0692	0693	0694	0695	0696	0697	0698	0699	0700	0701	0702	0703
2C	0704	0705	0706	0707	0708	0709	0710	0711	0712	0713	0714	0715	0716	0717	0718	0719
2D	0720	0721	0722	0723	0724	0725	0726	0727	0728	0729	0730	0731	0732	0733	0734	0735
2E	0736	0737	0738	0739	0740	0741	0742	0743	0744	0745	0746	0747	0748	0749	0750	0751
2F	0752	0753	0754	0755	0756	0757	0758	0759	0760	0761	0762	0763	0764	0765	0766	0767
30	0768	0769	0770	0771	0772	0773	0774	0775	0776	0777	0778	0779	0780	0781	0782	0783
31	0784	0785	0786	0787	0788	0789	0790	0791	0792	0793	0794	0795	0796	0797	0798	0799
32	0800	0801	0802	0803	0804	0805	0806	0807	0808	0809	0810	0811	0812	0813	0814	0815
33	0816	0817	0818	0819	0820	0821	0822	0823	0824	0825	0826	0827	0828	0829	0830	0831
34	0832	0833	0834	0835	0836	0837	0838	0839	0840	0841	0842	0843	0844	0845	0846	0847
35	0848	0849	0850	0851	0852	0853	0854	0855	0856	0857	0858	0859	0860	0861	0862	0863
36	0864	0865	0866	0867	0868	0869	0870	0871	0872	0873	0874	0875	0876	0877	0878	0879
37	0880	0881	0882	0883	0884	0885	0886	0887	0888	0889	0890	0891	0892	0893	0894	0895

HEXADECIMAL-DECIMAL NUMBER CONVERSION TABLE (Cont'd)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
38	0896	0897	0898	0899	0900	0901	0902	0903	0904	0905	0906	0907	0908	0909	0910	0911
39	0912	0913	0914	0915	0916	0917	0918	0919	0920	0921	0922	0923	0924	0925	0926	0927
3A	0928	0929	0930	0931	0932	0933	0934	0935	0936	0937	0938	0939	0940	0941	0942	0943
3B	0944	0945	0946	0947	0948	0949	0950	0951	0952	0953	0954	0955	0956	0957	0958	0959
3C	0960	0961	0962	0963	0964	0965	0966	0967	0968	0969	0970	0971	0972	0973	0974	0975
3D	0976	0977	0978	0979	0980	0981	0982	0983	0984	0985	0986	0987	0988	0989	0990	0991
3E	0992	0993	0994	0995	0996	0997	0998	0999	1000	1001	1002	1003	1004	1005	1006	1007
3F	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1020	1021	1022	1023
40	1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039
41	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1050	1051	1052	1053	1054	1055
42	1056	1057	1058	1059	1060	1061	1062	1063	1064	1065	1066	1067	1068	1069	1070	1071
43	1072	1073	1074	1075	1076	1077	1078	1079	1080	1081	1082	1083	1084	1085	1086	1087
44	1088	1089	1090	1091	1092	1093	1094	1095	1096	1097	1098	1099	1100	1101	1102	1103
45	1104	1105	1106	1107	1108	1109	1110	1111	1112	1113	1114	1115	1116	1117	1118	1119
46	1120	1121	1122	1123	1124	1125	1126	1127	1128	1129	1130	1131	1132	1133	1134	1135
47	1136	1137	1138	1139	1140	1141	1142	1143	1144	1145	1146	1147	1148	1149	1150	1151
48	1152	1153	1154	1155	1156	1157	1158	1159	1160	1161	1162	1163	1164	1165	1166	1167
49	1168	1169	1170	1171	1172	1173	1174	1175	1176	1177	1178	1179	1180	1181	1182	1183
4A	1184	1185	1186	1187	1188	1189	1190	1191	1192	1193	1194	1195	1196	1197	1198	1199
4B	1200	1201	1202	1203	1204	1205	1206	1207	1208	1209	1210	1211	1212	1213	1214	1215
4C	1216	1217	1218	1219	1220	1221	1222	1223	1224	1225	1226	1227	1228	1229	1230	1231
4D	1232	1233	1234	1235	1236	1237	1238	1239	1240	1241	1242	1243	1244	1245	1246	1247
4E	1248	1249	1250	1251	1252	1253	1254	1255	1256	1257	1258	1259	1260	1261	1262	1263
4F	1264	1265	1266	1267	1268	1269	1270	1271	1272	1273	1274	1275	1276	1277	1278	1279
50	1280	1281	1282	1283	1284	1285	1286	1287	1288	1289	1290	1291	1292	1293	1294	1295
51	1296	1297	1298	1299	1300	1301	1302	1303	1304	1305	1306	1307	1308	1309	1310	1311
52	1312	1313	1314	1315	1316	1317	1318	1319	1320	1321	1322	1323	1324	1325	1326	1327
53	1328	1329	1330	1331	1332	1333	1334	1335	1336	1337	1338	1339	1340	1341	1342	1343
54	1344	1345	1346	1347	1348	1349	1350	1351	1352	1353	1354	1355	1356	1357	1358	1359
55	1360	1361	1362	1363	1364	1365	1366	1367	1368	1369	1370	1371	1372	1373	1374	1375
56	1376	1377	1378	1379	1380	1381	1382	1383	1384	1385	1386	1387	1388	1389	1390	1391
57	1392	1393	1394	1395	1396	1397	1398	1399	1400	1401	1402	1403	1404	1405	1406	1407
58	1408	1409	1410	1411	1412	1413	1414	1415	1416	1417	1418	1419	1420	1421	1422	1423
59	1424	1425	1426	1427	1428	1429	1430	1431	1432	1433	1434	1435	1436	1437	1438	1439
5A	1440	1441	1442	1443	1444	1445	1446	1447	1448	1449	1450	1451	1452	1453	1454	1455
5B	1456	1457	1458	1459	1460	1461	1462	1463	1464	1465	1466	1467	1468	1469	1470	1471
5C	1472	1473	1474	1475	1476	1477	1478	1479	1480	1481	1482	1483	1484	1485	1486	1487
5D	1488	1489	1490	1491	1492	1493	1494	1495	1496	1497	1498	1499	1500	1501	1502	1503
5E	1504	1505	1506	1507	1508	1509	1510	1511	1512	1513	1514	1515	1516	1517	1518	1519
5F	1520	1521	1522	1523	1524	1525	1526	1527	1528	1529	1530	1531	1532	1533	1534	1535
60	1536	1537	1538	1539	1540	1541	1542	1543	1544	1545	1546	1547	1548	1549	1550	1551
61	1552	1553	1554	1555	1556	1557	1558	1559	1560	1561	1562	1563	1564	1565	1566	1567
62	1568	1569	1570	1571	1572	1573	1574	1575	1576	1577	1578	1579	1580	1581	1582	1583
63	1584	1585	1586	1587	1588	1589	1590	1591	1592	1593	1594	1595	1596	1597	1598	1599
64	1600	1601	1602	1603	1604	1605	1606	1607	1608	1609	1610	1611	1612	1613	1614	1615
65	1616	1617	1618	1619	1620	1621	1622	1623	1624	1625	1626	1627	1628	1629	1630	1631
66	1632	1633	1634	1635	1636	1637	1638	1639	1640	1641	1642	1643	1644	1645	1646	1647
67	1648	1649	1650	1651	1652	1653	1654	1655	1656	1657	1658	1659	1660	1661	1662	1663
68	1664	1665	1666	1667	1668	1669	1670	1671	1672	1673	1674	1675	1676	1677	1678	1679
69	1680	1681	1682	1683	1684	1685	1686	1687	1688	1689	1690	1691	1692	1693	1694	1695
6A	1696	1697	1698	1699	1700	1701	1702	1703	1704	1705	1706	1707	1708	1709	1710	1711
6B	1712	1713	1714	1715	1716	1717	1718	1719	1720	1721	1722	1723	1724	1725	1726	1727
6C	1728	1729	1730	1731	1732	1733	1734	1735	1736	1737	1738	1739	1740	1741	1742	1743
6D	1744	1745	1746	1747	1748	1749	1750	1751	1752	1753	1754	1755	1756	1757	1758	1759
6E	1760	1761	1762	1763	1764	1765	1766	1767	1768	1769	1770	1771	1772	1773	1774	1775
6F	1776	1777	1778	1779	1780	1781	1782	1783	1784	1785	1786	1787	1788	1789	1790	1791

HEXADECIMAL-DECIMAL NUMBER CONVERSION TABLE (Cont'd)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
70	1792	1793	1794	1795	1796	1797	1798	1799	1800	1801	1802	1803	1804	1805	1806	1807
71	1808	1809	1810	1811	1812	1813	1814	1815	1816	1817	1818	1819	1820	1821	1822	1823
72	1824	1825	1826	1827	1828	1829	1830	1831	1832	1833	1834	1835	1836	1837	1838	1839
73	1840	1841	1842	1843	1844	1845	1846	1847	1848	1849	1850	1851	1852	1853	1854	1855
74	1856	1857	1858	1859	1860	1861	1862	1863	1864	1865	1866	1867	1868	1869	1870	1871
75	1872	1873	1874	1875	1876	1877	1878	1879	1880	1881	1882	1883	1884	1885	1886	1887
76	1888	1889	1890	1891	1892	1893	1894	1895	1896	1897	1898	1899	1900	1901	1902	1903
77	1904	1905	1906	1907	1908	1909	1910	1911	1912	1913	1914	1915	1916	1917	1918	1919
78	1920	1921	1922	1923	1924	1925	1926	1927	1928	1929	1930	1931	1932	1933	1934	1935
79	1936	1937	1938	1939	1940	1941	1942	1943	1944	1945	1946	1947	1948	1949	1950	1951
7A	1952	1953	1954	1955	1956	1957	1958	1959	1960	1961	1962	1963	1964	1965	1966	1967
7B	1968	1969	1970	1971	1972	1973	1974	1975	1976	1977	1978	1979	1980	1981	1982	1983
7C	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999
7D	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
7E	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031
7F	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047
80	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063
81	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079
82	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095
83	2096	2097	2098	2099	2100	2101	2102	2103	2104	2105	2106	2107	2108	2109	2110	2111
84	2112	2113	2114	2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127
85	2128	2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143
86	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156	2157	2158	2159
87	2160	2161	2162	2163	2164	2165	2166	2167	2168	2169	2170	2171	2172	2173	2174	2175
88	2176	2177	2178	2179	2180	2181	2182	2183	2184	2185	2186	2187	2188	2189	2190	2191
89	2192	2193	2194	2195	2196	2197	2198	2199	2200	2201	2202	2203	2204	2205	2206	2207
8A	2208	2209	2210	2211	2212	2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223
8B	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239
8C	2240	2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254	2255
8D	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271
8E	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2287
8F	2288	2289	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303
90	2304	2305	2306	2307	2308	2309	2310	2311	2312	2313	2314	2315	2316	2317	2318	2319
91	2320	2321	2322	2323	2324	2325	2326	2327	2328	2329	2330	2331	2332	2333	2334	2335
92	2336	2337	2338	2339	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351
93	2352	2353	2354	2355	2356	2357	2358	2359	2360	2361	2362	2363	2364	2365	2366	2367
94	2368	2369	2370	2371	2372	2373	2374	2375	2376	2377	2378	2379	2380	2381	2382	2383
95	2384	2385	2386	2387	2388	2389	2390	2391	2392	2393	2394	2395	2396	2397	2398	2399
96	2400	2401	2402	2403	2404	2405	2406	2407	2408	2409	2410	2411	2412	2413	2414	2415
97	2416	2417	2418	2419	2420	2421	2422	2423	2424	2425	2426	2427	2428	2429	2430	2431
98	2432	2433	2434	2435	2436	2437	2438	2439	2440	2441	2442	2443	2444	2445	2446	2447
99	2448	2449	2450	2451	2452	2453	2454	2455	2456	2457	2458	2459	2460	2461	2462	2463
9A	2464	2465	2466	2467	2468	2469	2470	2471	2472	2473	2474	2475	2476	2477	2478	2479
9B	2480	2481	2482	2483	2484	2485	2486	2487	2488	2489	2490	2491	2492	2493	2494	2495
9C	2496	2497	2498	2499	2500	2501	2502	2503	2504	2505	2506	2507	2508	2509	2510	2511
9D	2512	2513	2514	2515	2516	2517	2518	2519	2520	2521	2522	2523	2524	2525	2526	2527
9E	2528	2529	2530	2531	2532	2533	2534	2535	2536	2537	2538	2539	2540	2541	2542	2543
9F	2544	2545	2546	2547	2548	2549	2550	2551	2552	2553	2554	2555	2556	2557	2558	2559
A0	2560	2561	2562	2563	2564	2565	2566	2567	2568	2569	2570	2571	2572	2573	2574	2575
A1	2576	2577	2578	2579	2580	2581	2582	2583	2584	2585	2586	2587	2588	2589	2590	2591
A2	2592	2593	2594	2595	2596	2597	2598	2599	2600	2601	2602	2603	2604	2605	2606	2607
A3	2608	2609	2610	2611	2612	2613	2614	2615	2616	2617	2618	2619	2620	2621	2622	2623
A4	2624	2625	2626	2627	2628	2629	2630	2631	2632	2633	2634	2635	2636	2637	2638	2639
A5	2640	2641	2642	2643	2644	2645	2646	2647	2648	2649	2650	2651	2652	2653	2654	2655
A6	2656	2657	2658	2659	2660	2661	2662	2663	2664	2665	2666	2667	2668	2669	2670	2671
A7	2672	2673	2674	2675	2676	2677	2678	2679	2680	2681	2682	2683	2684	2685	2686	2687

HEXADECIMAL-DECIMAL NUMBER CONVERSION TABLE (Cont'd)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A8	2688	2689	2690	2691	2692	2693	2694	2695	2696	2697	2698	2699	2700	2701	2702	2703
A9	2704	2705	2706	2707	2708	2709	2710	2711	2712	2713	2714	2715	2716	2717	2718	2719
AA	2720	2721	2722	2723	2724	2725	2726	2727	2728	2729	2730	2731	2732	2733	2734	2735
Ab	2736	2737	2738	2739	2740	2741	2742	2743	2744	2745	2746	2747	2748	2749	2750	2751
AC	2752	2753	2754	2755	2756	2757	2758	2759	2760	2761	2762	2763	2764	2765	2766	2767
AD	2768	2769	2770	2771	2772	2773	2774	2775	2776	2777	2778	2779	2780	2781	2782	2783
AE	2784	2785	2786	2787	2788	2789	2790	2791	2792	2793	2794	2795	2796	2797	2798	2799
AF	2800	2801	2802	2803	2804	2805	2806	2807	2808	2809	2810	2811	2812	2813	2814	2815
B0	2816	2817	2818	2819	2820	2821	2822	2823	2824	2825	2826	2827	2828	2829	2830	2831
B1	2832	2833	2834	2835	2836	2837	2838	2839	2840	2841	2842	2843	2844	2845	2846	2847
B2	2848	2849	2850	2851	2852	2853	2854	2855	2856	2857	2858	2859	2860	2861	2862	2863
B3	2864	2865	2866	2867	2868	2869	2870	2871	2872	2873	2874	2875	2876	2877	2878	2879
B4	2880	2881	2882	2883	2884	2885	2886	2887	2888	2889	2890	2891	2892	2893	2894	2895
B5	2896	2897	2898	2899	2900	2901	2902	2903	2904	2905	2906	2907	2908	2909	2910	2911
B6	2912	2913	2914	2915	2916	2917	2918	2919	2920	2921	2922	2923	2924	2925	2926	2927
B7	2928	2929	2930	2931	2932	2933	2934	2935	2936	2937	2938	2939	2940	2941	2942	2943
B8	2944	2945	2946	2947	2948	2949	2950	2951	2952	2953	2954	2955	2956	2957	2958	2959
B9	2960	2961	2962	2963	2964	2965	2966	2967	2968	2969	2970	2971	2972	2973	2974	2975
BA	2976	2977	2978	2979	2980	2981	2982	2983	2984	2985	2986	2987	2988	2989	2990	2991
BB	2992	2993	2994	2995	2996	2997	2998	2999	3000	3001	3002	3003	3004	3005	3006	3007
BC	3008	3009	3010	3011	3012	3013	3014	3015	3016	3017	3018	3019	3020	3021	3022	3023
BD	3024	3025	3026	3027	3028	3029	3030	3031	3032	3033	3034	3035	3036	3037	3038	3039
BE	3040	3041	3042	3043	3044	3045	3046	3047	3048	3049	3050	3051	3052	3053	3054	3055
BF	3056	3057	3058	3059	3060	3061	3062	3063	3064	3065	3066	3067	3068	3069	3070	3071
C0	3072	3073	3074	3075	3076	3077	3078	3079	3080	3081	3082	3083	3084	3085	3086	3087
C1	3088	3089	3090	3091	3092	3093	3094	3095	3096	3097	3098	3099	3100	3101	3102	3103
C2	3104	3105	3106	3107	3108	3109	3110	3111	3112	3113	3114	3115	3116	3117	3118	3119
C3	3120	3121	3122	3123	3124	3125	3126	3127	3128	3129	3130	3131	3132	3133	3134	3135
C4	3136	3137	3138	3139	3140	3141	3142	3143	3144	3145	3146	3147	3148	3149	3150	3151
C5	3152	3153	3154	3155	3156	3157	3158	3159	3160	3161	3162	3163	3164	3165	3166	3167
C6	3168	3169	3170	3171	3172	3173	3174	3175	3176	3177	3178	3179	3180	3181	3182	3183
C7	3184	3185	3186	3187	3188	3189	3190	3191	3192	3193	3194	3195	3196	3197	3198	3199
C8	3200	3201	3202	3203	3204	3205	3206	3207	3208	3209	3210	3211	3212	3213	3214	3215
C9	3216	3217	3218	3219	3220	3221	3222	3223	3224	3225	3226	3227	3228	3229	3230	3231
CA	3232	3233	3234	3235	3236	3237	3238	3239	3240	3241	3242	3243	3244	3245	3246	3247
CB	3248	3249	3250	3251	3252	3253	3254	3255	3256	3257	3258	3259	3260	3261	3262	3263
CC	3264	3265	3266	3267	3268	3269	3270	3271	3272	3273	3274	3275	3276	3277	3278	3279
CD	3280	3281	3282	3283	3284	3285	3286	3287	3288	3289	3290	3291	3292	3293	3294	3295
CE	3296	3297	3298	3299	3300	3301	3302	3303	3304	3305	3306	3307	3308	3309	3310	3311
CF	3312	3313	3314	3315	3316	3317	3318	3319	3320	3321	3322	3323	3324	3325	3326	3327
D0	3328	3329	3330	3331	3332	3333	3334	3335	3336	3337	3338	3339	3340	3341	3342	3343
D1	3344	3345	3346	3347	3348	3349	3350	3351	3352	3353	3354	3355	3356	3357	3358	3359
D2	3360	3361	3362	3363	3364	3365	3366	3367	3368	3369	3370	3371	3372	3373	3374	3375
D3	3376	3377	3378	3379	3380	3381	3382	3383	3384	3385	3386	3387	3388	3389	3390	3391
D4	3392	3393	3394	3395	3396	3397	3398	3399	3400	3401	3402	3403	3404	3405	3406	3407
D5	3408	3409	3410	3411	3412	3413	3414	3415	3416	3417	3418	3419	3420	3421	3422	3423
D6	3424	3425	3426	3427	3428	3429	3430	3431	3432	3433	3434	3435	3436	3437	3438	3439
D7	3440	3441	3442	3443	3444	3445	3446	3447	3448	3449	3450	3451	3452	3453	3454	3455
D8	3456	3457	3458	3459	3460	3461	3462	3463	3464	3465	3466	3467	3468	3469	3470	3471
D9	3472	3473	3474	3475	3476	3477	3478	3479	3480	3481	3482	3483	3484	3485	3486	3487
DA	3488	3489	3490	3491	3492	3493	3494	3495	3496	3497	3498	3499	3500	3501	3502	3503
DB	3504	3505	3506	3507	3508	3509	3510	3511	3512	3513	3514	3515	3516	3517	3518	3519
DC	3520	3521	3522	3523	3524	3525	3526	3527	3528	3529	3530	3531	3532	3533	3534	3535
DD	3536	3537	3538	3539	3540	3541	3542	3543	3544	3545	3546	3547	3548	3549	3550	3551
DE	3552	3553	3554	3555	3556	3557	3558	3559	3560	3561	3562	3563	3564	3565	3566	3567
DF	3568	3569	3570	3571	3572	3573	3574	3575	3576	3577	3578	3579	3580	3581	3582	3583

HEXADECIMAL-DECIMAL NUMBER CONVERSION TABLE (Cont'd)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
E0	3584	3585	3586	3587	3588	3589	3590	3591	3592	3593	3594	3595	3596	3597	3598	3599
E1	3600	3601	3602	3603	3604	3605	3606	3607	3608	3609	3610	3611	3612	3613	3614	3615
E2	3616	3617	3618	3619	3620	3621	3622	3623	3624	3625	3626	3627	3628	3629	3630	3631
E3	3632	3633	3634	3635	3636	3637	3638	3639	3640	3641	3642	3643	3644	3645	3646	3647
E4	3648	3649	3650	3651	3652	3653	3654	3655	3656	3657	3658	3659	3660	3661	3662	3663
E5	3664	3665	3666	3667	3668	3669	3670	3671	3672	3673	3674	3675	3676	3677	3678	3679
E6	3680	3681	3682	3683	3684	3685	3686	3687	3688	3689	3690	3691	3692	3693	3694	3695
E7	3696	3697	3698	3699	3700	3701	3702	3703	3704	3705	3706	3707	3708	3709	3710	3711
E8	3712	3713	3714	3715	3716	3717	3718	3719	3720	3721	3722	3723	3724	3725	3726	3727
E9	3728	3729	3730	3731	3732	3733	3734	3735	3736	3737	3738	3739	3740	3741	3742	3743
EA	3744	3745	3746	3747	3748	3749	3750	3751	3752	3753	3754	3755	3756	3757	3758	3759
EB	3760	3761	3762	3763	3764	3765	3766	3767	3768	3769	3770	3771	3772	3773	3774	3775
EC	3776	3777	3778	3779	3780	3781	3782	3783	3784	3785	3786	3787	3788	3789	3790	3791
ED	3792	3793	3794	3795	3796	3797	3798	3799	3800	3801	3802	3803	3804	3805	3806	3807
EE	3808	3809	3810	3811	3812	3813	3814	3815	3816	3817	3818	3819	3820	3821	3822	3823
EF	3824	3825	3826	3827	3828	3829	3830	3831	3832	3833	3834	3835	3836	3837	3838	3839
F0	3840	3841	3842	3843	3844	3845	3846	3847	3848	3849	3850	3851	3852	3853	3854	3855
F1	3856	3857	3858	3859	3860	3861	3862	3863	3864	3865	3866	3867	3868	3869	3870	3871
F2	3872	3873	3874	3875	3876	3877	3878	3879	3880	3881	3882	3883	3884	3885	3886	3887
F3	3888	3889	3890	3891	3892	3893	3894	3895	3896	3897	3898	3899	3900	3901	3902	3903
F4	3904	3905	3906	3907	3908	3909	3910	3911	3912	3913	3914	3915	3916	3917	3918	3919
F5	3920	3921	3922	3923	3924	3925	3926	3927	3928	3929	3930	3931	3932	3933	3934	3935
F6	3936	3937	3938	3939	3940	3941	3942	3943	3944	3945	3946	3947	3948	3949	3950	3951
F7	3952	3953	3954	3955	3956	3957	3958	3959	3960	3961	3962	3963	3964	3965	3966	3967
F8	3968	3969	3970	3971	3972	3973	3974	3975	3976	3977	3978	3979	3980	3981	3982	3983
F9	3984	3985	3986	3987	3988	3989	3990	3991	3992	3993	3994	3995	3996	3997	3998	3999
FA	4000	4001	4002	4003	4004	4005	4006	4007	4008	4009	4010	4011	4012	4013	4014	4015
FB	4016	4017	4018	4019	4020	4021	4022	4023	4024	4025	4026	4027	4028	4029	4030	4031
FC	4032	4033	4034	4035	4036	4037	4038	4039	4040	4041	4042	4043	4044	4045	4046	4047
FD	4048	4049	4050	4051	4052	4053	4054	4055	4056	4057	4058	4059	4060	4061	4062	4063
FE	4064	4065	4066	4067	4068	4069	4070	4071	4072	4073	4074	4075	4076	4077	4078	4079
FF	4080	4081	4082	4083	4084	4085	4086	4087	4088	4089	4090	4091	4092	4093	4094	4095