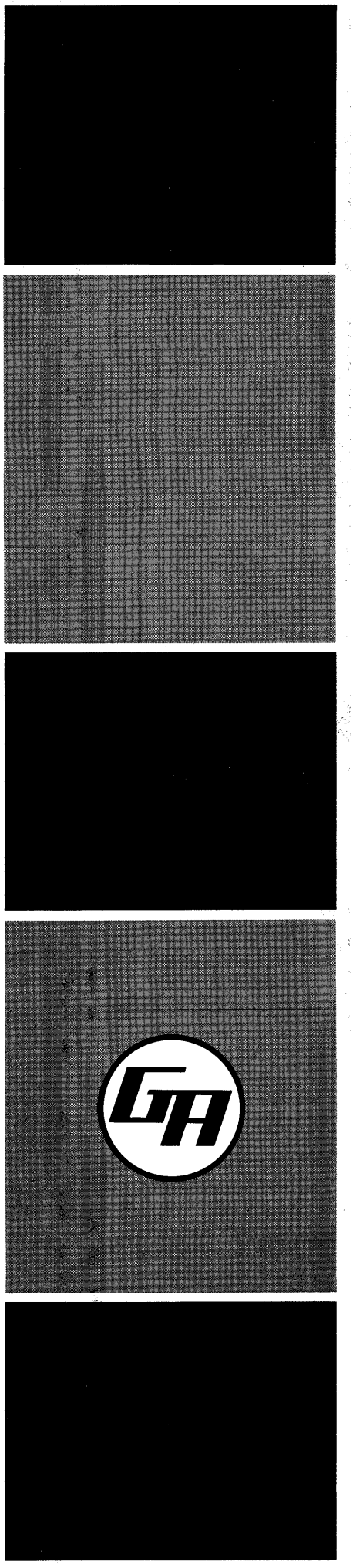
REFERENCE MANUAL

GA 18/30 INDUSTRIAL SUPERVISORY SYSTEM



GENERAL AUTOMATION, INC.

GA 18/30 INSTRUCTIONS

Mnemonic	Instruction Name	Execution Time (μs)*	Reference Manual Page
LOAD/STORE			
LDS	Load Status	1.2	3-2
STS	Store Status	1.2	3-2
LDX	Load Index	1.2	3-3
STX	Store Index	2.4	3-4
LD	Load Accumulator	2.4	3-4
LDD	Load Double	3.6	3-4
STO	Store Accumulator	2.4	3-4
STD	Store Double	3.6	3-5
ARITHMETIC/LOGICAL			
A	Add	2.4	3-5
AD	Add Double	3.6	3-5
S	Subtract	2.4	3-6
SD	Subtract Double	3.6	3-6
M	Multiply	12.0	3-6
D	Divide	13.2	3-6
AND	Logical And	2.4	3-7
OR	Logical Or	2.4	3-7
EOR	Logical Exclusive Or	2.4	3-7
SHIFT			
SLA	Shift Left Logical A	1.2 + 1.2(N/4)	3-8
SLT	Shift Left Logical A and Q	1.2 + 1.2(N/4)	3-9
SLCA	Shift Left and Count A	1.2 + 1.2(N/4)	3-9
SLC	Shift Left and Count A and Q	1.2 + 1.2(N/4)	3-9
SRA	Shift Right Logical A	1.2 + 1.2(N/4)	3-9
SRT	Shift Right A and Q	1.2 + 1.2(N/4)	3-10
RTE	Rotate Right A and Q	1.2 + 1.2(N/4)	3-10
BRANCH			
BSI	Branch and Store Instruction Register	2.4**	3-12
BSC (BOSC)	Branch or Skip on Condition (Branch Out of Interrupt)	1.2**	3-13
MDX	Modify Index and Skip	1.2	3-13
CMP	Compare	2.4	3-14
DCM	Compare Double	3.6	3-15
REGISTER TRANSFER			
RTR	Register Transfer	2.4	3-16
RCP	Register Transfer and Complement	2.4	3-16
RIC	Register Transfer and Increment	2.4	3-16
RDC	Register Transfer and Decrement	2.4	3-16
I/O AND CONTROL			
WAIT	Wait	1.2	3-17
NOP	No Operation	1.2	3-17
XIO	Execute I/O	3.6 or 4.8	3-17

*Time is shown for single word instruction. Add 1.2 μs for double word instruction and 1.2 μs for indirect addressing. N represents number of shifts.

**See instruction description for additional timing information.

GA 18/30 INSTRUCTION FORMATS

SINGLE WORD FORMAT: OP (0 1 2 3) | F (4 5 6 7) | T | DISPLACEMENT (8 9 10 11 12 13 14 15)

DOUBLE WORD FORMAT: OP (0 1 2 3) | F (4 5 6 7) | T | IA | BO | CONDITIONS (8 9 10 11 12 13 14 15) | ADDRESS (0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15)

Mnemonic	Name	SINGLE WORD FORMAT OP	F	T	DISPLACEMENT	DOUBLE WORD FORMAT OP	F	T	CONDITIONS	ADDRESS
WAIT	Wait	0	0		00	-	-		-	-
		3	0		00	-	-		-	-
		3	8		00	-	-		-	-
XIO	Execute I/O	0	8-B		00-FF	0	C-F		00 or 80	0000-FFFF
SLA	Shift Left Logical A	1	0-3		00-3F	-	-		-	-
SLCA	Shift Left and Count A	1	0-3		40-7F	-	-		-	-
SLT	Shift Left Logical A and Q	1	0-3		80-BF	-	-		-	-
SLC	Shift Left and Count A and Q	1	0-3		C0-FF	-	-		-	-
SRA	Shift Right Logical A	1	8-B		00-3F	-	-		-	-
SRT	Shift Right A and Q	1	8-B		80-BF	-	-		-	-
RTE	Rotate Right A and Q	1	8-B		C0-FF	-	-		-	-
LDS	Load Status	2	0		00-03	-	-		-	-
STS	Store Status	2	8-B		00-FF	2	C-F		10, 40, 41, 80, C0, C1	0000-FFFF
BSI	Branch and Store Instruction Register	4	0-3		00-FF	4	4-7		00-FF	0000-FFFF
BSC (BOSC)	Branch or Skip on Condition	4	8		00-7F	4	C-F		00-FF	0000-FFFF
RCP	Register Transfer and Complement	5	0 or 8		00-FF	-	-		-	-
RIC	Register Transfer and Increment	5	1 or 9		00-FF	-	-		-	-
RDC	Register Transfer and Decrement	5	2 or A		00-FF	-	-		-	-
RTR	Register Transfer	5	3 or B		00-FF	-	-		-	-
LDX	Load Index	6	0-3		00-FF	6	4-7		00 or 80	0000-FFFF
STX	Store Index	6	8-B		00-FF	6	C-F		00 or 80	0000-FFFF
MDX	Modify Index and Skip	7	0-3		00-FF	7	4-7		00-FF	0000-FFFF
NOP	No Operation	7	8		00	-	-		-	-
A	Add	8	0-3		00-FF	8	4-7		00 or 80	0000-FFFF
AD	Add Double	8	8-B		00-FF	8	C-F		00 or 80	0000-FFFF
S	Subtract	9	0-3		00-FF	9	4-7		00 or 80	0000-FFFF
SD	Subtract Double	9	8-B		00-FF	9	C-F		00 or 80	0000-FFFF
M	Multiply	A	0-3		00-FF	A	4-7		00 or 80	0000-FFFF
D	Divide	A	8-B		00-FF	A	C-F		00 or 80	0000-FFFF
CMP	Compare	B	0-3		00-FF	B	4-7		00 or 80	0000-FFFF
DCM	Compare Double	B	8-B		00-FF	B	C-F		00 or 80	0000-FFFF
LD	Load Accumulator	C	0-3		00-FF	C	4-7		00 or 80	0000-FFFF
LDD	Load Double	C	8-B		00-FF	C	C-F		00 or 80	0000-FFFF
STO	Store Accumulator	D	0-3		00-FF	D	4-7		00 or 80	0000-FFFF
STD	Store Double	D	8-B		00-FF	D	C-F		00 or 80	0000-FFFF
AND	Logical And	E	0-3		00-FF	E	4-7		00 or 80	0000-FFFF
OR	Logical Or	E	8-B		00-FF	E	C-F		00 or 80	0000-FFFF
EOR	Logical Exclusive Or	F	0-3, 8-B		00-FF	F	4-7, C-F		00 or 80	0000-FFFF

OP — Operation Code; F — Format Designator; T — Index Tag; IA — Indirect Address Flag;

BO — Branch Out of Interrupt Flag

88A00026A-A

May 1969
$10.00

REFERENCE MANUAL

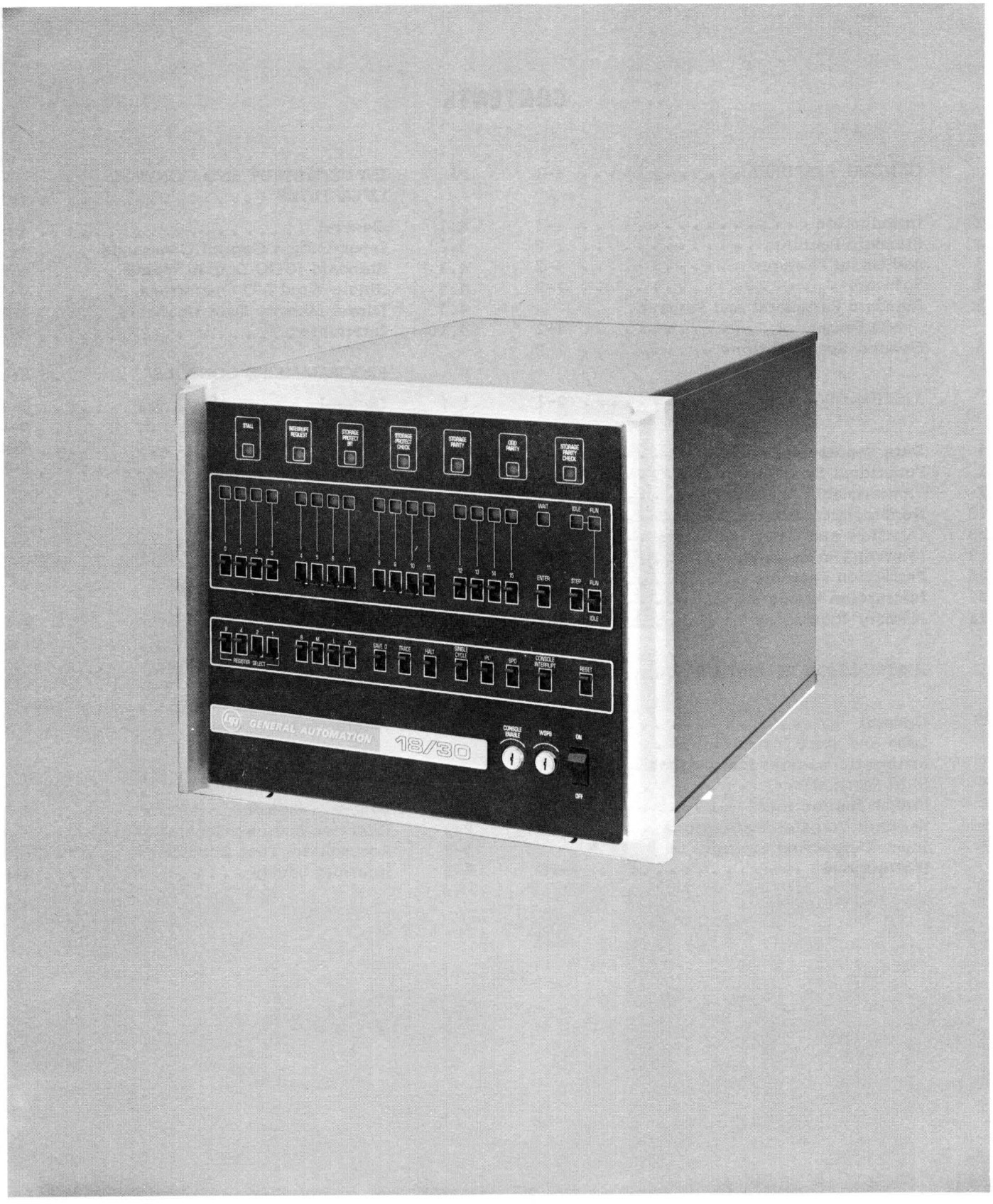
GA 18/30 INDUSTRIAL SUPERVISORY SYSTEM

GENERAL AUTOMATION, INC.
Automation Products Division
706 West Katella, Orange, California 92668 (714) 633-1091

REVISION

SYMBOL	DESCRIPTION	APPROVED	DATE
A	Original Issue		May 69

CONTENTS



GA 18/30 Industrial Supervisory System

SECTION I
GENERAL FEATURES

1.1 INTRODUCTION

When the demand arose for a solution to the problem of developing a small, low-cost, automation computer with the reliability and versatility required for special purpose electronic control systems, General Automation pioneered the solution with the SPC-12 Automation Minicomputer — the first industrial minicomputer to represent advancements in value, reliability, and solution completion.

Now, General Automation has pioneered another first in automation technology — the GA 18/30 Industrial Supervisory System. Reflecting the same automation know-how, planning, and attention to user needs that produced the SPC-12, the GA 18/30 is the total solution to today's and tomorrow's real-time data acquisition, process supervision, and communication and process control problems.

The GA 18/30, with its powerful computational capability, operational safety features, and expandable I/O system, is the ideal solution for applications ranging from missile systems checkout to medical research, and in industries as diverse as food processing and steel production. Typical of GA 18/30 applications are:

- Engineering analysis and design optimization
- Production unit checkout and evaluation
- High-speed data acquisition, data logging, and display
- Process simulation
- Industrial process supervision and control

Among the factors that make the GA 18/30 the leader in its class are:

- Size. Including power supply and cooling, it is only 15-3/4 inches high, 22-1/2 inches deep, 19 inches wide, and weighs just 85 pounds (105 pounds with all additional features).
- Reliability. Worst case design featuring wire-free construction, integrated and all-silicon circuits, wide-temperature (lithium) memories, and exceptional speed, temperature, power, and noise margins ensure reliable operation in the most demanding industrial environments.
- Safety. Memory write protection, stall alarm, and marginal power detection and automatic restart features guarantee confidence for on-line control applications. A keylock switch for disabling console controls adds an additional measure of operational security.
- Memory Expansion. The standard 4096-word memory can be expanded to 32,768 words in blocks of 4096 words without the need for additional external enclosures.
- Speed. A fully parallel arithmetic unit featuring hardware multiply and divide, plus a 960-nanosecond memory, provides more computation in less time.
- Powerful Instruction Repertoire. The basic repertoire contains 32 instructions with numerous subsets. Both single- and double-precision arithmetic instructions are standard. Most instructions can have single or double word formats.
- IBM 1800/1130 Instruction Compatible. Instruction compatibility with the IBM 1800 and 1130 extends to the binary coding level. Additionally, the GA 18/30 contains an extra class of register transfer instructions.
- Multiple Addressing Modes. Single word instructions can specify addressing relative to a base register (instruction counter or any one of three index registers). Double word instructions can specify direct, indexed, or indirect addressing. Double word instructions can directly address 32,768 memory locations obviating the need for memory paging.
- Expandable I/O System. The standard I/O system provides the capability for I/O control, sensing, and single word data transfer operations. Up to five direct memory data channels can be used for block-transfer capability. The direct memory data channels operate directly with memory on a cycle-steal basis. Eight interrupt levels (internal, trace,

and six external levels) are provided as part of the priority interrupt feature. The external interrupts can be expanded to a maximum of 59 levels. All external interrupt levels can be masked, unmasked, and triggered under program control

1.2 STANDARD FEATURES

Standard features of the GA 18/30 include:

- All-silicon logic circuits
- 18-bit memory word length
- Memory parity
- Memory write protection (controllable for each storage location)
- 4096-word memory, expandable to 32,768 words
- 960-nanosecond memory cycle time
- Fully parallel operation
- General purpose register block with 16 registers for program and I/O control:
    - Instruction counter
    - Three index registers for base-relative addressing (single word instructions) or address modification (double word instructions)
    - Double-precision accumulator
    - Ten direct memory transfer control registers
- Automatic priority interrupt system consisting of:
    - One internal level
    - One trace interrupt
    - Six external levels (expansion to 59 levels)
- Multiple addressing modes
    - Direct to 32,768 memory locations (double word instructions) without paging
    - Base-relative (single word instructions)
    - Indexed (double word instructions)
    - Indirect (double word instructions)
- Double-precision arithmetic
- Hardware multiply and divide
- Programmer's console

1.3 ADDITIONAL FEATURES

Additional features available for the GA 18/30 include:

- Memory expansion to 32,768 words in blocks of 4096 words
- A marginal power detection and automatic restart feature for automatic and safe shutdown in the event of a power failure and unattended startup when power returns
- An operations monitor alarm (stall alarm) for a positive indication of erroneous program sequencing
- Five direct memory data channels for block-transfer of data directly between memory and I/O devices with a minimum of program intervention and with block chaining capability
- External priority interrupt expansion to a maximum of 59 external interrupt levels
- Three crystal-controlled interval timers with time bases of 0.1, 1, and 10 milliseconds

1.4 SOFTWARE

A complete system of supporting software is available for the GA 18/30, including an assembler, a FORTRAN compiler, utility programs, and a subroutine library.

1.5 STANDARD PERIPHERAL AND PROCESS I/O EQUIPMENT

A full line of peripheral I/O devices and local and remote analog and digital process I/O equipment is available for use with the computer. These include mass storage devices (both drum and disc), magnetic and paper tape handlers, card readers and punches, strip and line printers, Teletype devices, cathode-ray tube displays, analog-to-digital and digital-to-analog units, multiplexers, and communications equipment.

1.6 GENERAL SPECIFICATIONS

Table 1-1 lists the general specifications for the computer.

Table 1-1. General Specifications

Characteristic	Specification
Type	Industrial automation-oriented computer with binary, fixed word length parallel processor
Memory	Random access, wide temperature, lithium magnetic core; 960-ns cycle time; 16-bit data word plus parity bit and storage protection bit; eight memory sizes (4096 words to 32,768 words in 4096-word increments)
Addressing Methods	Direct addressing to 32,768 memory locations; indexing (three 16-bit hardware index registers); indirect addressing; relative addressing; base addressing
Arithmetic	16-bit parallel, binary, fixed point, two's complement; single or double precision; program testable Overflow and Carry indicators
Instructions	32 basic instructions with numerous subsets; instruction repertoire compatible with IBM 1800 and 1130
Instruction Classes	Load and store (single and double word), arithmetic, logical, shift, register transfer, branch and skip, input/output and control
Speed	Typical execution time: Add (single precision) 2.4 μs Add (double precision) 3.6 μs Load and store (single precision) 2.4 μs Load and store (double precision) 3.6 μs Multiply 12.0 μs Divide 13.2 μs Skip 1.2 to 1.32 μs Register transfer 2.4 μs Branch 1.32 to 2.4 μs I/O transfers 4.8 μs Direct memory transfers 1.2 μs Maximum access time on highest priority channel 1.32 μs
Input/Output	16-bit parallel I/O bus; up to 61 automatic priority interrupt levels featuring program triggering and mask control of each external (59 maximum) interrupt level; console data entry from switches via sense instruction. Five direct memory data channels (cycle stealing); three crystal-controlled interval timers; stall alarm; marginal power detection and automatic restart; Teletypewriter plus wide range of peripheral equipment

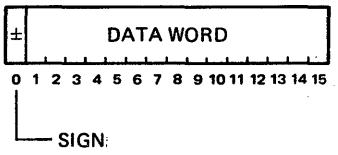
Table 1-1. General Specifications (Cont.)

Characteristic	Specification
Programmer's Console	16 data switches and indicators; register selection switches for 20 registers; key-operated console lockout switch; key-operated storage protect lockout switch; alarm indicators; 13 control switches
Software	Program compatible with IBM 1800 and 1130 systems; hardware test and verify routines; basic utility systems; subroutine library; FORTRAN compiler and assembler
Size	19 inches wide, 15-3/4 inches high, 22-1/2 inches deep with power supply and cooling
Weight	85 to 105 pounds (depending on additional features) with power supply and cooling
Operating Environment	0° to 50°C (32° to 122°F), 10 to 90% relative humidity (no condensation)
Power	115 (±11.5)V, 47 to 63 Hz, single phase, 800 watts
Installation	One-piece packaging including enclosure, power supply and cooling permits table top or standard 19-inch rack mounting

SECTION II
SYSTEM ORGANIZATION

2.1 DATA REPRESENTATION

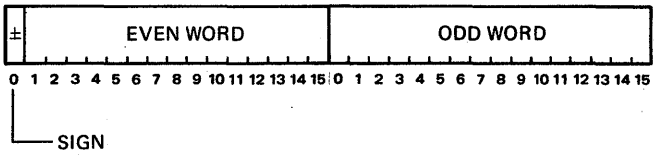
The basic data element used in the GA 18/30 is a single precision 16-bit data word with the following format:



Bit position 15 represents the least significant portion of the word and bit position 0, the most significant. The sign (polarity) of the data word is represented by bit position 0. Positive numbers are stored in the computer in true binary form with a sign bit of 0. Negative numbers are stored in two's complement form with a sign bit of 1.

The largest single precision positive number that can be stored is +32,767 (2^{15}-1). This is represented by a sign bit of 0 and 1's in all other bit positions. The largest single precision negative number that can be stored is -32,768 (-2^{15}). This is represented by a sign bit of 1, and 0's in all other bit positions. Zero is represented by 0's in all bit positions including the sign position.

Two adjacent memory locations can be used to store double precision numbers in the following format:



When double precision numbers are stored in memory, the most significant half of the number is always stored at an even address, and the least significant half, at the next highest odd address. Double precision words are always referenced by the address of the most significant word (even address). The largest double precision positive number that can be stored is +2,147,483,647 (2^{31}-1), and the largest negative number is -2,147,483,648 (-2^{31}).

Most instructions in the computer can be represented either by a single 16-bit word (single word instructions) or by two 16-bit words (double word instructions). Double word instructions require two consecutive memory storage locations, and, like double precision numbers, are referenced by the address of the most significant word. However, unlike double precision numbers, the first word is not restricted to even addresses.

The basic 16-bit organization of the computer lends itself to the use of hexadecimal notation for representing the binary instructions, addresses, and data manipulated by the computer. In hexadecimal notation a 16-bit binary number can be expressed in only four hexadecimal digits. The hexadecimal numbering system uses digits 0 through 9 and letters A through F to represent decimal numbers 0 through 15. The hexadecimal and binary equivalents for decimal numbers 0 through 15 are shown in the following table:

Decimal	Binary	Hexadecimal
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
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

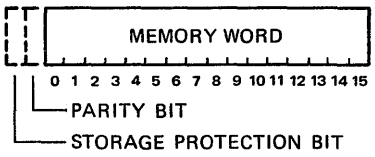
When hexadecimal notation is used in this manual, it is indicated by a string of hexadecimal digits enclosed in single quotation marks and preceded by the letter X. For example, the decimal number 16,383 would be shown in hexadecimal notation as X'3FFF'. Appendix A contains additional information on hexadecimal arithmetic, including conversion tables.

2.2 FUNCTIONAL DESCRIPTION

Figure 2-1 shows the functional organization of the computer including the major paths of data flow. The basic operational elements of the computer can generally be grouped into four functional sections: memory, control, arithmetic/logic, and input/output. These sections are described in the following paragraphs.

2.3 MEMORY

The standard memory system of the computer includes 4096 words of random access, lithium magnetic core storage with a read-restore time of 960 nanoseconds. Core storage is expandable in 4096-word increments to a maximum of 32,768 words. Each core storage location can hold an 18-bit memory word consisting of 16 data bits, 1 parity bit, and 1 storage protection bit:



The memory parity bit is set for odd parity (including the storage protection bit) each time a word is written into memory. Each time a word is read from memory it is checked for odd parity. If a parity error occurs, it is indicated by the STORAGE PARITY CHECK indicator on the programmer's console and the internal interrupt level (X'0008') is activated.

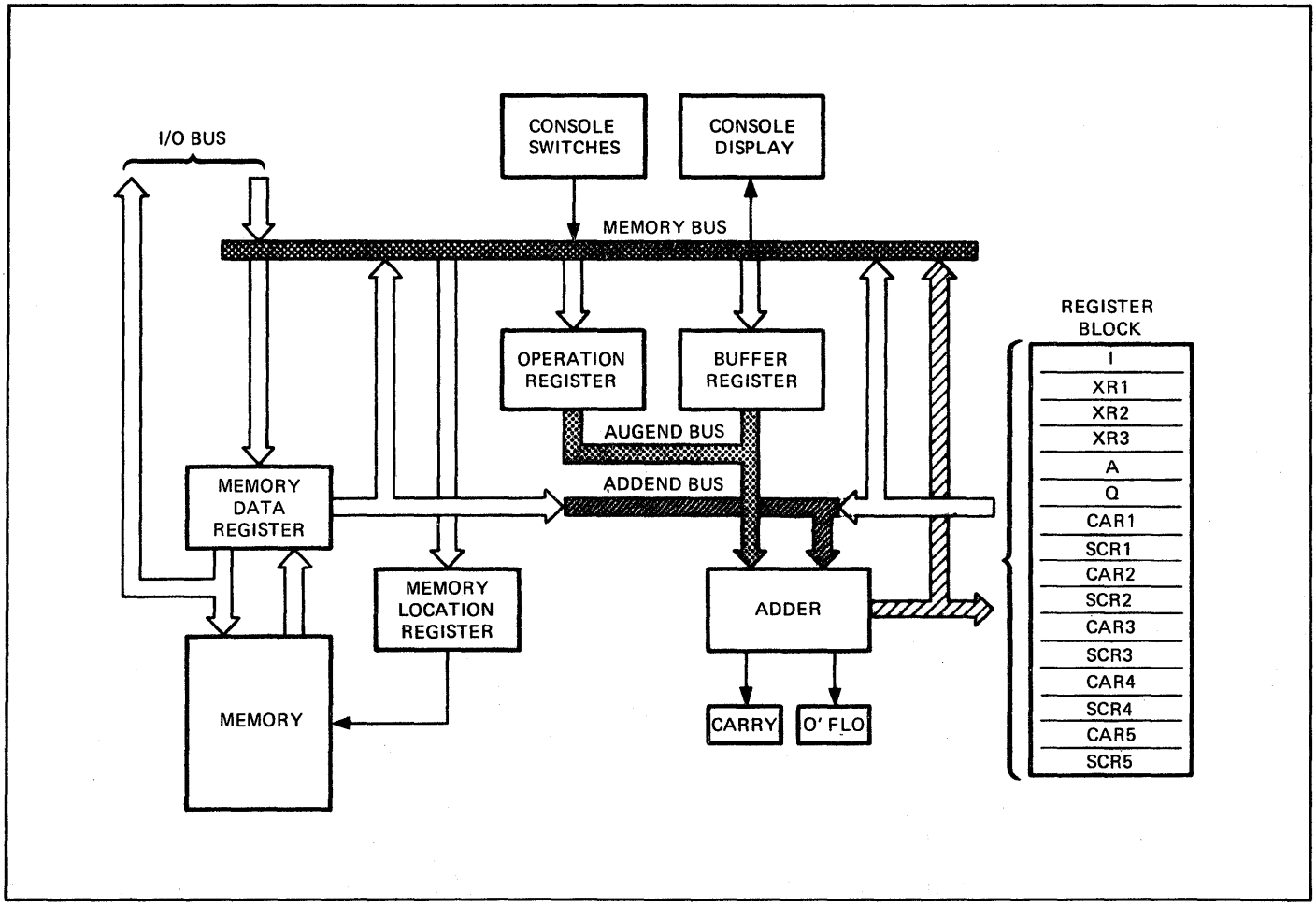


Figure 2-1. Basic Computer Organization

The storage protection bit is used to prevent the inadvertant loss of stored information by inhibiting writing into protected locations. When the storage protection bit of any location is set to 1, any attempt to write into that location results in an internal interrupt (X'0008'), and an indication of the violation is displayed by the STORAGE PROTECT CHECK indicator on the programmer's console. For detailed information on the storage protection feature, refer to paragraph 2.22.

The processor can directly address all storage locations in memory. Memory addresses range from X'0000' to X'7FFF' (for a maximum size memory of 32,768 words). For memory sizes of 4096, 8192, 16,384, 24,576, and 32,768 words, circular (wrap-around) addressing is used so that the next address following the highest address is always X'0000'. Certain memory locations are reserved for special uses by the processor. These locations and their uses are listed in table 2-1.

Two registers are used in all operations requiring memory access: the memory data register (M-register) and the memory location (L-register). All information is fetched from memory and stored into memory via the memory data register. The address of the memory location from which data is fetched or into which data is stored is always held in the memory location register.

2.4 ARITHMETIC/LOGIC

The arithmetic and logical operations of the computer are primarily performed by the 16-bit parallel adder and three registers – the accumulator (A-register), the accumulator extension (Q-register), and the buffer register (B-register).

The accumulator is the primary arithmetic register of the processor. It is used in all addition, subtraction, multiplication, and division operations, as well as all logical and shift operations. The accumulator extension register is used with the accumulator for double-precision addition and subtraction and for multiplication, division, and double register shift operations. Both the accumulator and its extension are programmable registers.

The buffer register is used mainly as an intermediate working register. It is not a programmable register.

Two indicators – Carry and Overflow – are associated with the adder. These indicators can be tested by the program with a Branch or Skip on Condition instruction or a double word Branch and Store Instruction Register instruction.

2.5 CONTROL

After a program is stored in memory, its execution is controlled primarily by the contents of the instruction counter, and operation and index registers.

The instruction counter (I-register) operates as the program counter, or sequencer. It supplies the memory location register with the addresses of the instructions from which the computer operates. It is automatically advanced during the execution of each instruction (except for branch instructions) so

Table 2-1. Reserved Memory Locations

Address		Use
Decimal	Hexadecimal	
4	4	Interval timer A
5	5	Interval timer B
6	6	Interval timer C
8	8	Internal interrupt level
9	9	Trace interrupt level
11 thru 16	B thru 10	Standard external interrupt levels
17 thru 69	11 thru 45	Optional external interrupt levels

that it holds the address of the next instruction in sequence. During the execution of a branch instruction it is loaded with the branch address specified by the instruction if the branch conditions are satisfied.

As instructions are fetched from memory, they are automatically loaded into the operation register (O-register). The unique codes assigned to each instruction are then decoded to generate the control signals used by the processor to perform the specified operation. If the instruction is one that requires a memory access, the address information provided by the instruction is used to develop the effective address, which is then loaded into the memory location register for the memory access.

The index registers can be used to modify the address portion of double word instructions, or they can be used to hold a base address for relative addressing during the execution of single word instructions. Single word instructions can specify addressing relative to the instruction counter or to any of the index registers (up to 127 locations forward or 128 backward from the base address held in the specified register).

2.6 INPUT/OUTPUT

The transfer of data between the computer and external devices is carried out through a parallel 16-bit I/O bus. The standard I/O system provides the capability for controlling and sensing the conditions of external devices and processes, and for transferring 16-bit data words between the computer and external devices.

The addition of direct memory data channels to the basic I/O system provides the capability for transferring blocks of 16-bit words, one word at a time, directly between memory and external devices on a cycle-steal basis; that is, the direct memory data channel can delay program execution for one memory cycle while one word is transferred directly between memory and the device. The processor contains 10 control registers (two for each channel) that are used by the direct memory data channels in controlling block-transfer operations. These are the channel address registers (CAR1 through CAR5) and the scan control registers (SCR1 through SCR5).

The I/O system features six external priority interrupt levels with expansion to 59 levels available. Each external interrupt level can be masked or unmasked under program control, and interrupt requests can be initiated on each level by the program.

Each interrupt level can accommodate as many as 16 separate interrupt request lines.

2.7 PROGRAMMABLE REGISTERS

The processor contains a register block of 16 high-speed, integrated circuit registers that can be manipulated under program control. Included in the register block are the instruction counter, three index registers, the accumulator, the accumulator extension, five channel address registers, and five scan control registers. The channel address and scan control registers are used with the direct memory data channels.

Each of these registers can be selected for display on the programmer's console by the REGISTER SELECT switches. The addresses used for selecting these registers from the console and for specifying the registers in the register transfer instructions are as follows:

Hexadecimal Address	Register
0	Instruction counter (I)
1	Index register (XR1)
2	Index register (XR2)
3	Index register (XR3)
4	Accumulator (A)
5	Accumulator extension (Q)
6	Channel address register (CAR1)
7	Scan control register (SCR1)
8	Channel address register (CAR2)
9	Scan control register (SCR2)
A	Channel address register (CAR3)
B	Scan control register (SCR3)
C	Channel address register (CAR4)
D	Scan control register (SCR4)
E	Channel address register (CAR5)
F	Scan control register (SCR5)

2.8 INSTRUCTION COUNTER

The instruction counter (I-register) is a 16-bit register that normally holds the address of the next sequential instruction. At the start of each instruction cycle, its contents are automatically transferred to the memory location register to fetch the next instruction from memory, and then its contents are increased by one count in readiness for fetching the next instruction from memory. During the execution of single word instructions the instruction counter can also be designated as the base address register for relative addressing.

The operation of the instruction counter in fetching a double word instruction from memory is as follows:

a. At the beginning of the instruction cycle the contents of the instruction counter are automatically transferred to the memory location register to fetch the first word of the instruction. The instruction counter is then advanced by one count.

b. The first word of the instruction is fetched from memory and loaded into the operation register for decoding.

c. The contents of the instruction counter are then transferred to the memory location register to fetch the second (address) word from memory. Again, the instruction counter is advanced by one count.

d. The second word of the instruction is fetched from memory and loaded into the buffer register. If indexing is specified, the contents of the specified index register are added to the address word before it is loaded into the buffer register. If indirect addressing is not specified, the buffer register now contains the effective address of the operand.

e. If indirect addressing is specified, the contents of the buffer register are transferred to the memory location register to fetch the effective address from memory.

f. The effective address is then fetched from memory and loaded into the buffer register.

g. The execution cycle of the instruction now begins with the transfer of the effective address from the buffer register to the memory location register to fetch the operand from memory.

2.9 INDEX REGISTERS

Three index registers (XR1, XR2, XR3) are provided. These registers can be used as base address registers during the execution of single word memory addressing instructions or for address modification of double word instructions. The operation of these registers for memory addressing and address modification is described in paragraph 2.26.

2.10 ACCUMULATOR

The 16-bit accumulator (A-register) is the primary arithmetic register of the computer. It normally provides one of the operands to the adder for arithmetic and logical operations and stores the result or a portion of the result. Its contents can be loaded from memory, stored into memory, shifted right or left, and manipulated by specific arithmetic and logical instructions.

2.11 ACCUMULATOR EXTENSION

The 16-bit accumulator extension (Q-register) is used as the low-order extension of the accumulator during multiplication, division, double precision load, store, addition, subtraction, and double register shift operations.

2.12 CHANNEL ADDRESS AND SCAN CONTROL REGISTERS

The 16-bit channel address and scan control registers (CAR1 through CAR5 and SCR1 through SCR5) are the primary control registers for the direct memory data channels. One channel address register and one scan control register are used with each channel. The operation of these registers is described in paragraph 4.6.

2.13 NONPROGRAMMABLE REGISTERS

The computer contains four hardware registers that are used during and affected by the execution of most instructions, but that cannot be directly manipulated under program control. Two of the registers — the memory location register and the operation register — directly control the automatic execution of the stored program. The other two — the memory data and buffer registers — are used for transferring data and for working storage.

2.14 MEMORY LOCATION REGISTER

The 16-bit memory location register (L-register) contains the address of the memory location into which instruction or data words are stored or from which they are fetched. It supplies the address to the address decoding logic when a memory access is required.

2.15 OPERATION REGISTER

The 16-bit operation register (O-register) contains the instruction currently being executed.

It automatically receives the instruction word as it is fetched from memory. The instruction is then decoded to provide the control signals for the processor to execute the instruction and perform the specified operation.

2.16 MEMORY DATA REGISTER

The 18-bit memory data register (M-register) is used to transfer information in and out of memory. In addition to the 16 data bits, it also holds the parity and storage protection bits.

2.17 BUFFER REGISTER

The 16-bit buffer register is used as a temporary working storage register.

2.18 OVERFLOW AND CARRY INDICATORS

Two indicators – Carry and Overflow – are associated with the accumulator. The Carry indicator is turned on (set) during an adder operation when a carry or borrow occurs from the high-order position of the accumulator. It changes with each add or subtract operation and is also affected by the Shift Left, Load Status, Store Status, and Compare instructions.

The Overflow indicator can be turned on (set) during addition, subtraction, or division operations. It indicates that the result of the operation exceeds the capacity of the accumulator. Once it is turned on, the Overflow indicator can be reset only by testing or by executing a Load Status or Store Status instruction.

2.19 INTERVAL TIMERS

Three interval timers are provided for supplying real-time information to the program. The timers have fixed time intervals and operate through three memory locations as follows:

Timer	Location	Time Interval
A	X'0004'	0.1 ms
B	X'0005'	1.0 ms
C	X'0006'	10.0 ms

The timers are started and stopped by the program with a dedicated version of the Execute I/O instruction. After a timer has been started, it is automatically advanced by one count at the end of each time interval (0.1, 1, or 10 ms). When the count reaches the highest positive value ($2^{15}-1$), it continues through the negative values, from highest to lowest, until it reaches zero. At that time, the counter requests an interrupt on the external interrupt level to which it is assigned. The timer does not stop after reaching zero, but continues to operate until stopped by the program. The interval timers will operate properly even though their locations may be storage protected.

Interval timer operation can be inhibited from the programmer's console when the computer is in idle mode by setting the HALT switch to the lower (on) position. If the switch is turned on during a timing operation, the operation continues to completion, and then further operations are inhibited.

The Execute I/O instruction for starting and stopping the timers is described in paragraph 4.3.

2.20 PROTECTION FEATURES

The protective features of the computer include the memory parity and memory storage protection features, and the stall alarm and marginal power detection and automatic restart features.

2.21 MEMORY PARITY

When the program attempts to fetch a word from a memory location containing incorrect (even) parity, an internal interrupt request is automatically generated (bit 2 of internal interrupt level status word turned on), and the STORAGE PARITY CHECK indicator on the programmer's console is turned on.

Since a parity error could occur when an instruction, an address, or an operand is read from memory, the correct instruction may be executed with incorrect data, or an incorrect instruction may be executed, in which case program recovery may not be possible.

2.22 MEMORY STORAGE PROTECTION

The storage protection feature protects the contents of specified memory locations from being lost during the execution or debugging of a program by the erroneous storing of information into these locations by the program. Each memory storage location contains a bit designated as the storage protection bit that can be turned on or off to inhibit or permit writing into the location. When the storage protection bit is on, information can be read from, but cannot be written into, the location. When the bit is off, both reading and writing are permitted.

Once the storage protection bit has been turned on, any attempt to write into the protected location except by the interval timers causes an internal

interrupt request to be generated (bit 3 of the internal interrupt level status word turned on). The violation is also indicated by the STORAGE PROTECT CHECK indicator on the programmer's console.

The storage protection bits can be turned on or off by the double word Store Status instruction (BO = 1). This instruction can change the status of the bits only when it is executed with the WSPB switch unlocked. Otherwise the instruction has no effect.

2.23 OPERATIONS MONITOR ALARM (STALL ALARM)

The stall alarm protects the system against abnormal operation and provides both visual and electrical signals to warn of the abnormality. It is particularly valuable in detecting improper program sequencing caused by component breakdowns or previously undetected program errors. Once activated, the stall alarm can cause instruction execution to proceed to an orderly halt, provides a stall signal (100-ma drive) on the I/O interface, and removes the signal from the system safe line. The stall condition is indicated by the STALL ALARM indicator on the programmer's console.

The stall alarm is controlled by an internal timer that is started when the CONSOLE ENABLE switch is locked. Once the timer is started, it must be reset by an XIO Reset Stall Alarm instruction within a preselected time interval to prevent the stall alarm from being activated. If the instruction is not executed before the time interval elapses, the alarm is activated. Once activated, the stall alarm can be reset by the XIO instruction. It can also be reset from the programmer's console by unlocking the CONSOLE ENABLE switch. As long as the switch is unlocked, the stall alarm is inoperative.

The XIO instruction for resetting the stall alarm timer is described in paragraph 4.3.

2.24 MARGINAL POWER DETECTION AND AUTOMATIC RESTART

The marginal power detection and automatic restart option protects the integrity of the system from unreliable operation of its power source.

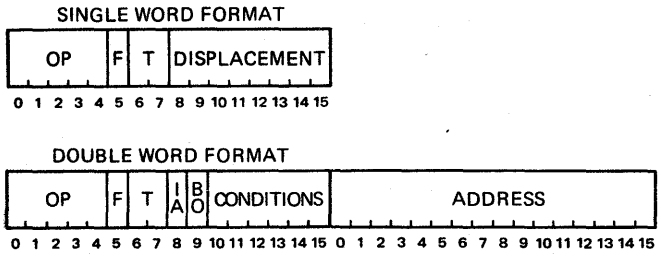
When power falls below safe operating limits, the marginal power detection circuits sense the condition and automatically generate an internal interrupt request (bit 0 of the internal interrupt level status word turned on). Approximately 500 microseconds later the system is automatically shut down in a reset condition. The 500-microsecond period provides time for the program, after sensing the interrupt, to store volatile information before the system shuts down.

Following the 500-microsecond period, the system shuts down nonsynchronously; therefore, part of the shutdown program must turn off the timers, turn off the direct memory data channels, and set the computer in the wait condition.

When power again returns to normal, the automatic restart circuits initiate a restart sequence which ends with an internal interrupt request (bit 1 of the internal interrupt level status word turned on). This interrupt can then be used to restore the conditions that existed before the system shut down.

2.25 INSTRUCTION WORDS

Most of the computer instructions can have either a single word or double word format as shown in the following diagrams:



The fields within each format are used as follows:

Field	Use
OP	Specifies the operation to be performed, such as add, subtract, multiply, divide, and so on.
F	Specifies whether the instruction is single word or double word. If F is 0, the instruction is single word; if F is 1, the instruction is double word.
T	Specifies the base address register (I, XR1, XR2, XR3) for single word instructions, or specifies the index register to be used for address modification of double word instructions. For shift instructions, this field specifies the location of the shift count. For register transfer instructions, this field is an extension of the OP field.
Displacement	Used in determining the effective address for single word instructions. The expanded displacement

Field	Use
	(EDISP) is normally added with the instruction counter (T = 00), or with one of the index registers (T = 01, 10, 11) to derive the effective address. The expanded displacement is the value obtained when the displacement is automatically expanded to a 16-bit value by duplicating bit position 8 (sign bit) in bit positions 0 through 7.
IA	Normally specifies indirect addressing. This bit has special uses for the Load Index and Modify Index and Skip instructions.
BO	Specifies that the Branch or Skip on Conditions (BSC) instruction is to be interpreted as a Branch Out of Interrupt (BOSC) instruction.
Conditions	Specifies the conditions of the accumulator that are to be tested during a BSC or double word BSI instruction.
Address	Contains the reference address for a double word memory addressing instruction. This address can be modified by the contents of an index register (T = 01, 10, 11) or can be used as an indirect address (IA = 1).

2.26 MEMORY ADDRESSING

Four methods of memory addressing can be used by the computer instructions:

a. Direct addressing

b. Base-relative addressing

c. Indexed addressing

d. Indirect addressing

Direct addressing can be performed only by double word instructions. This is the normal mode of addressing when both the T and IA fields of the instruction are zero and when F equals 1. When direct addressing is specified, the address field of the instruction is used as the effective address. Double word instructions can directly address up to 32,768 memory locations.

Base-relative addressing is performed by single word instructions. A 16-bit effective address is developed for single word instructions by adding the expanded displacement (EDISP) with a specified base register. The base register can be either the instruction counter (which normally contains the address of the next instruction in sequence), or any one of the index registers. The tag (T) field of the instruction specifies the base register as follows:

Tag	Base Register
00	I-register
01	XR1
10	XR2
11	XR3

Since the expanded displacement can range in value from -128 to +127 (bit 8 of the displacement is treated as a sign), base-relative addressing permits single word instructions to address 127 locations forward or 128 backward from the base address contained in the specified register.

Indexed addressing can be performed only by double word instructions. When indexed addressing is specified (T = 01, 10, or 11), the contents of the selected index register are added to the address field to derive the effective address.

Indirect addressing can be performed only by double word instructions. When indirect addressing is specified (IA = 1), the contents of the location specified by the address field of the instruction are fetched from memory and are used as the effective address. If both indexed and indirect addressing are specified in the same instruction, indexing is performed before indirect addressing.

Table 2-2 summarizes effective address computation for both single and double word instructions.

Table 2-2. Effective Address Generation

Instruction Type	Tag (T)	IA	Effective Address
Single word	00	-	(I) + EDISP*
	01	-	(XR1) + EDISP
	10	-	(XR2) + EDISP
	11	-	(XR3) + EDISP
Double word	00	0	Address
	01	0	Address + (XR1)
	10	0	Address + (XR2)
	11	0	Address + (XR3)
Double word	00	1	(Address)
	01	1	(Address + (XR1))
	10	1	(Address + (XR2))
	11	1	(Address + (XR3))

*EDISP can range from -128 to +127.

SECTION III
INSTRUCTION REPERTOIRE

3.1 GENERAL

This section describes the operations performed by the computer instructions. The instruction descriptions are grouped according to the following functional classes:

a. Load and Store

b. Arithmetic

c. Shift

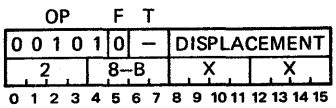
d. Branch

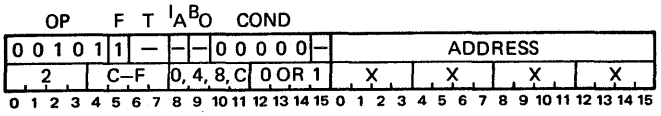
e. Register Transfer

f. Input/Output and Control

The following format is used in describing each instruction:

MNEMONIC INSTRUCTION NAME TIMING





DESCRIPTION

OPERATION STATEMENT

INDICATORS

AFFECTED

MNEMONIC. The mnemonic is the abbreviation that is recognized and used by the assembler to produce the unique operation code for each instruction.

INSTRUCTION NAME. The instruction name is the descriptive title of the instruction.

TIMING. Instruction execution time is given for the single word version of the instruction. An additional 1.2 microseconds should be added for double word instructions, and another 1.2 microseconds, if the instruction specifies indirect addressing. For shift instructions the execution time is given as $1.2 + 1.2(N/4)$ microseconds, where N represents the number of shifts and N/4 must be an integer. All special timing considerations, other than those specified here, are defined in the instruction descriptions.

FORMAT DIAGRAMS. The instruction format diagrams show the binary and hexadecimal configuration for both the single word and double word (if applicable) versions of each instruction. The upper half of the format shows the binary configuration; the lower half, the hexadecimal configuration. Both the binary and hexadecimal notations are shown for fields in which the contents are fixed. Abbreviations or words are used to identify fields in which the contents are variable. The following abbreviations are used in the format diagrams:

BO	Branch out of interrupt flag
COND	Condition
F	Format designator
IA	Indirect address flag
N	Number of shifts
OP	Operation code
RD	Destination register
RS	Source register
T	Indexed addressing tag
X	Variable hexadecimal value

DESCRIPTION. The instruction description explains the operations performed by the computer in executing the instruction.

OPERATION STATEMENT. The operation statement summarizes instruction operation in symbolic notation or simplified diagrams wherever possible. Symbolic terms and abbreviations used in the operation statement are:

DISP	Displacement (contents of bit positions 8 through 15 of the instruction)

EA	Effective address. The effective address is the final 16-bit address value developed for the instruction after all address arithmetic has been performed
EDISP	Expanded displacement. The value obtained when the displacement is automatically expanded to a 16-bit value by duplicating bit position 8 (sign bit) in bit positions 0 through 7
( )	Contents of register or memory location specified
x	Multiplication
÷	Division
+	Addition
-	Subtraction
∩	Logical product (AND)
∪	Logical sum (OR)
⊕	Logical comparison (exclusive OR)
>	Greater than
<	Less than
=	Equal
⟶	Replace

INDICATORS. The instruction's effect on the Carry and Overflow indicators is specified.

AFFECTED. All programmable registers and memory locations that can be affected by the instruction are listed.

3.2 LOAD AND STORE INSTRUCTIONS

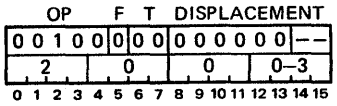
The following load and store instructions are available to the programmer:

LDS	Load Status
STS	Store Status
LDX	Load Index
STX	Store Index
LD	Load Accumulator
LDD	Load Double
STO	Store Accumulator
STD	Store Double

All of the instructions, with the exception of Load Status, can be used in either the single or double word format. The Load Status instruction can be used only in the single word format.

Indirect addressing can be specified for all of the double word instructions except Load Index. Although the indirect address (IA) bit is used in this instruction, its interpretation is different from that of normal indirect addressing. The way in which this bit is used is explained in the description of the Load Index instruction.

LDS LOAD STATUS 1.2 μs



The Load Status instruction copies the status of bits 14 and 15 of its displacement field into the Carry and Overflow indicators. Bit 14 is copied into the Carry indicator, and bit 15 is copied into the Overflow indicator. A 1-bit turns the corresponding indicator on; a 0-bit turns it off. The contents of Carry and Overflow are normally copied into the displacement field of the Load Status instruction by a Store Status instruction.

This instruction is a single-word instruction only. Its displacement field is used only for changing the status of the Carry and Overflow indicators, not for memory addressing.

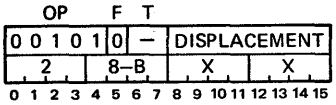
$$(\text{DISP})_{14} \longrightarrow (\text{CARRY})$$

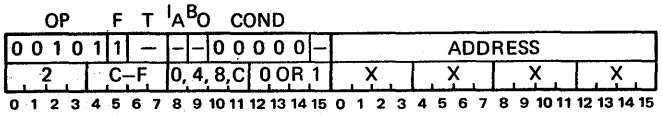
$$(\text{DISP})_{15} \longrightarrow (\text{OVERFLOW})$$

Indicators: Carry and Overflow are set or reset according to the states of displacement bits 14 and 15.

Affected: None

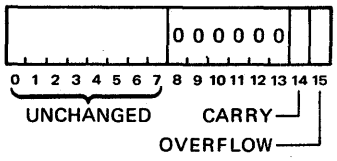
STS STORE STATUS 2.4 μs





The Store Status instruction performs either of the following operations:

a. If the single word instruction is used, or if bit 9 (BO) of the double word instruction is zero, the status of the Carry and Overflow indicators is stored in bit positions 14 and 15 of the location specified by the effective address. Carry status is stored in bit position 14; Overflow status, in bit position 15. The contents of bits 0 through 7 of the effective address remain unchanged, and bits 8 through 13 are reset to zero.



b. If bit 9 (BO) of the double word instruction is a 1, the storage protection bit of the memory location specified by the effective address is set or cleared according to the status of condition bit 15, as follows:

Bit 15	Result
0	Clear storage protection bit
1	Set storage protection bit

All other bits except the parity bit are unchanged. The storage protection bits can be changed only when the WSPB switch on the control console is unlocked (otherwise, the instruction performs no operation). For this function, the instruction requires 1.32 microseconds for execution.

For single word instruction or double word instruction with BO = 0:

$$(\text{CARRY}) \longrightarrow (\text{EA})_{14}$$

$$(\text{OVERFLOW}) \longrightarrow (\text{EA})_{15}$$

$$0\text{'s} \longrightarrow (\text{EA})_{8\text{-}13}$$

For double word instruction with BO = 1:

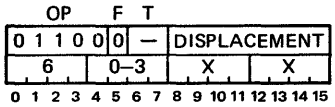
Clear storage protection bit if bit 15 is 0

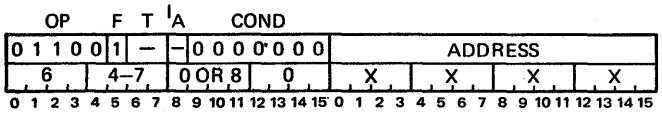
Set storage protection bit if bit 15 is 1

Indicators: Carry and Overflow are reset as they are stored.

Affected: $(\text{EA})_{8\text{-}15}$

LDX LOAD INDEX 1.2 μs





The Load Index instruction loads the expanded displacement, the address field, or the contents of the location specified by the address field into the register specified by the tag (T) field. The register is specified as follows:

T	Register
00	I
01	XR1
10	XR2
11	XR3

When the single word instruction is used, the expanded displacement is loaded into the specified register.

When the double word instruction is used, the IA bit of the instruction determines whether the contents of the address field or the contents of the location specified by the address field are loaded into the register. If bit IA is 0, the contents of the address field of the instruction are loaded into the specified register; if bit IA is 1, the contents of the location specified by the address field are loaded into the specified register.

For single word instruction:

$$\text{EDISP} \longrightarrow (\text{I, XR1, XR2, or XR3})$$

For double word instruction with IA = 0:

$$\text{ADDRESS} \longrightarrow (\text{I, XR1, XR2, or XR3})$$

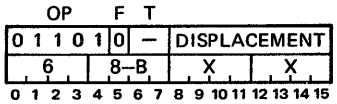
For double word instruction with IA = 1:

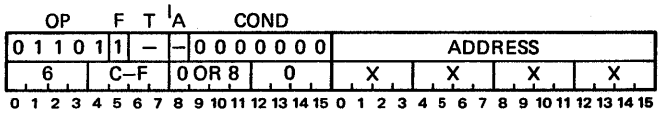
$$(\text{ADDRESS}) \longrightarrow (\text{I, XR1, XR2, or XR3})$$

Indicators: Carry and Overflow are not affected.

Affected: (I), (XR1), (XR2), or (XR3)

STX STORE INDEX **2.4 μs**





The Store Index instruction stores the contents of the register specified by the tag (T) field at the effective address. The register is specified as follows:

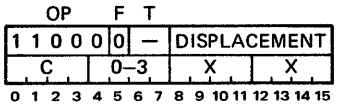
T	Register
00	I
01	XR1
10	XR2
11	XR3

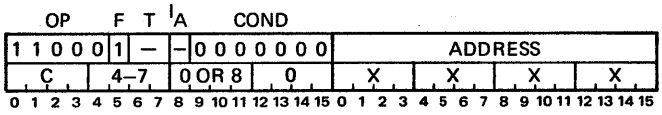
(I, XR1, XR2, or XR3) ⟶ (EA)

Indicators: Carry and Overflow are not affected.

Affected: (EA)

LD LOAD ACCUMULATOR **2.4 μs**





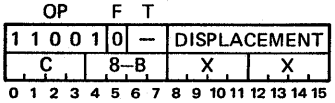
The Load Accumulator instruction loads the contents of the effective address into the accumulator. The contents of the effective address are not changed.

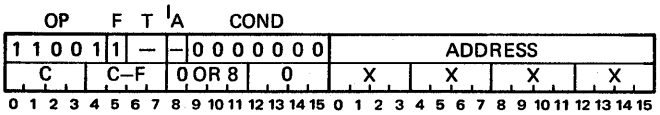
(EA) ⟶ (A)

Indicators: Carry and Overflow are not affected.

Affected: (A)

LDD LOAD DOUBLE **3.6 μs**





The Load Double instruction loads the contents of the effective address into the accumulator and loads the contents of the next higher address (effective address plus one) into the accumulator extension register (Q). For correct operation the effective address of the instruction must be an even address. If it is not, the contents of the effective address are loaded into the accumulator extension, and the contents of the next lower address (effective address minus one) are loaded into the accumulator. The contents of the effective address are not changed.

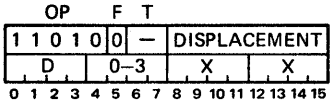
(EA) ⟶ (A)

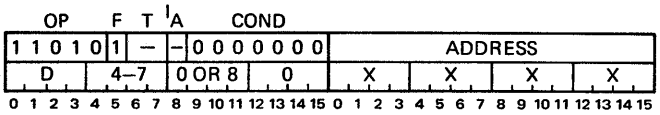
(EA+1) ⟶ (Q)

Indicators: Carry and Overflow are not affected.

Affected: (A), (Q)

STO STORE ACCUMULATOR **2.4 μs**





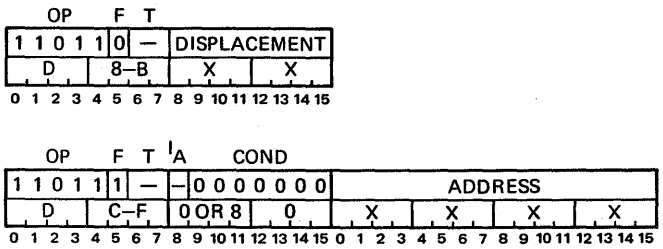
The Store Accumulator instruction stores the contents of the accumulator at the effective address. The contents of the accumulator are not changed.

(A) ⟶ (EA)

Indicators: Carry and Overflow are not affected.

Affected: (EA)

STD STORE DOUBLE **3.6 μs**



The Store Double instruction stores the contents of the accumulator at the effective address and stores the contents of the accumulator extension register (Q) at the next higher address (effective address plus one). For correct operation the effective address of the instruction must be an even address. If it is odd, the contents of the accumulator extension are stored at the effective address, and the contents of the accumulator are stored at the next lower address (effective address minus one). The contents of the accumulator and its extension are not changed.

$$(A) \longrightarrow (EA)$$

$$(Q) \longrightarrow (EA+1)$$

Indicators: Carry and Overflow are not affected.

Affected: (EA) and (EA+1)

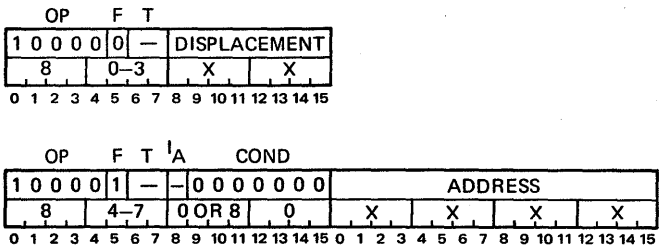
3.3 ARITHMETIC/LOGICAL INSTRUCTIONS

The following arithmetic/logical instructions are available to the programmer:

A	Add
AD	Add Double
S	Subtract
SD	Subtract Double
M	Multiply
D	Divide
AND	Logical And
OR	Logical Or
EOR	Logical Exclusive Or

The arithmetic/logical instructions can be used in either the single or double word format, and indirect addressing can be specified for all double word instructions.

A ADD **2.4 μs**



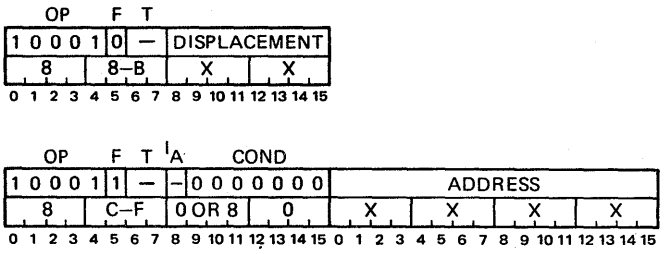
The Add instruction adds the contents of the effective address to the contents of the accumulator and loads the sum into the accumulator. The addition is performed in two's complement arithmetic with negative operands and sums in two's complement form. The contents of the effective address are not changed.

$$(EA) + (A) \longrightarrow (A)$$

Indicators: Overflow is set (turned on) if the sum exceeds the capacity of the accumulator (greater than $2^{15}-1$, or less than -2^{15}). Otherwise, Overflow is not affected. Carry is set if there is a carry out of bit position 0.

Affected: (A)

AD ADD DOUBLE **3.6 μs**



The Add Double instruction adds the contents of the effective address and the next higher address, treated as one signed 32-bit number, to the contents of the accumulator and accumulator extension, and loads the sum into the accumulator and extension. The most significant 15 bits and sign of the sum are loaded into the accumulator; the least significant 16 bits, into the accumulator extension. The contents of the effective address and next higher address are not changed.

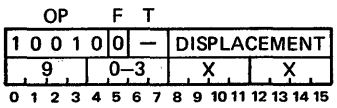
For correct operation the effective address must be an even address.

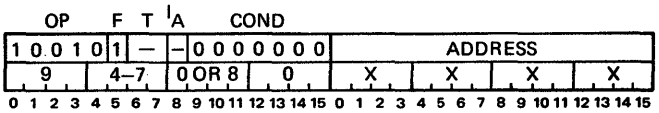
$$(EA, EA+1) + (A, Q) \longrightarrow (A, Q)$$

Indicators: Carry is set if there is a carry out of bit position 0 of the accumulator. Overflow is set if the sum exceeds the capacity of the accumulator and extension (greater than $2^{31}-1$ or less than -2^{31}). Otherwise, Overflow is not affected.

Affected: (A), (Q)

S SUBTRACT 2.4 μs





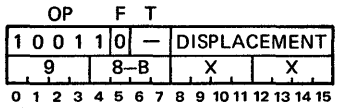
The Subtract instruction subtracts the contents of the effective address from the contents of the accumulator and loads the result into the accumulator. The subtraction is performed in two's complement arithmetic with negative operands and differences in two's complement form. The contents of the effective address are not changed.

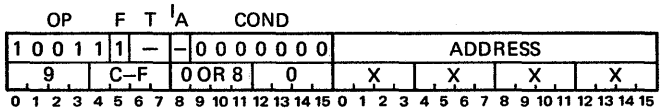
$$(A) - (EA) \longrightarrow (A)$$

Indicators: Carry is set if there is a borrow from bit position 0. Overflow is set if the difference exceeds the capacity of the accumulator (greater than $2^{15}-1$, or less than -2^{15}). Otherwise, Overflow is not affected.

Affected: (A)

SD SUBTRACT DOUBLE 3.6 μs





The Subtract Double instruction subtracts the contents of the effective address and the next higher address (treated as one signed 32-bit number) from the contents of the accumulator and accumulator extension, and loads the difference into the accumulator and extension.

The most significant 15 bits and sign of the difference are loaded into the accumulator; the least significant 16 bits, into the accumulator extension. The contents of the effective address and next higher address are not changed.

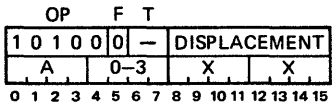
For correct operation the effective address must be an even address.

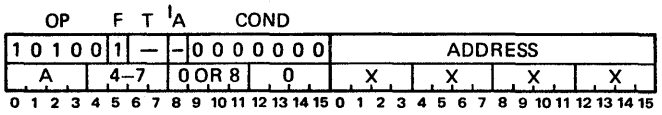
$$(A, Q) - (EA, EA+1) \longrightarrow (A, Q)$$

Indicators: Carry is set if there is a borrow from bit position 0 of the accumulator. Overflow is set if the difference exceeds the capacity of the accumulator and extension (greater than $2^{31}-1$, or less than -2^{31}). Otherwise, Overflow is not affected.

Affected: (A), (Q)

M MULTIPLY 12.0 μs





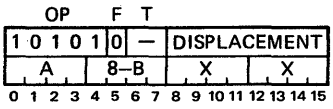
The Multiply instruction multiplies the contents of the effective address (multiplicand) by the contents of the accumulator (multiplier), and loads the double precision product into the accumulator and accumulator extension. The most significant 15 bits and sign of the product are loaded into the accumulator; the least significant 16 bits, into the accumulator extension. The contents of the effective address are not changed.

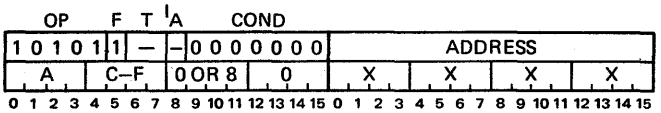
$$(EA) \times (A) \longrightarrow (A, Q)$$

Indicators: Carry and Overflow are not affected.

Affected: (A), (Q)

D DIVIDE 13.2 μs





The Divide instruction divides the double precision contents of the accumulator and extension (dividend) by the contents of the effective address (divisor), and loads the quotient into the accumulator and the remainder into the accumulator extension. The sign of the remainder is the same as that of the dividend or is zero. The contents of the effective address are not changed.

The largest dividend that can be correctly divided is $2^{30} + 2^{15}-1$ when the largest negative divisor (-2^{15}) is used.

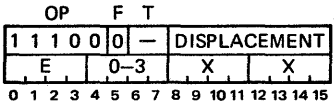
$$(A, Q) \div (EA) \longrightarrow (A)$$

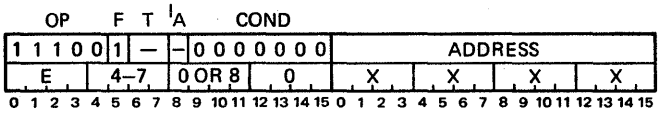
$$\text{Remainder} \longrightarrow (Q)$$

Indicators: Carry is not affected. Overflow is set if division by zero is attempted, or if division is attempted that would result in a quotient greater than $2^{15}-1$ or less than -2^{15}. If overflow occurs, the contents of the accumulator and accumulator extension are not predictable. Division by zero leaves the contents of the accumulator and accumulator extension unchanged.

Affected: (A), (Q)

AND LOGICAL AND **2.4 μs**





The Logical And instruction forms the logical product of the contents of the effective address and the contents of the accumulator and loads the result into the accumulator. The logical product is formed as follows:

Bit at Effective Address	Corresponding Bit in Accumulator	Result
0	0	0
0	1	0
1	0	0
1	1	1

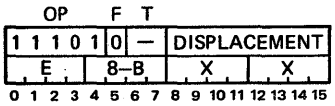
The contents of the effective address are not changed.

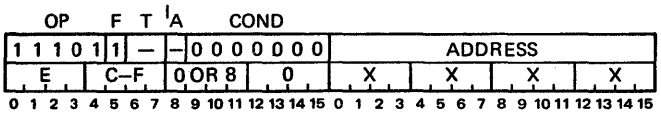
$$(EA) \cap (A) \longrightarrow (A)$$

Indicators: Carry and Overflow are not affected.

Affected: (A)

OR LOGICAL OR **2.4 μs**





The Logical Or instruction forms the logical sum of the contents of the effective address and the contents of the accumulator, and loads the result into the accumulator. The logical sum is formed as follows:

Bit at Effective Address	Corresponding Bit in Accumulator	Result
0	0	0
0	1	1
1	0	1
1	1	1

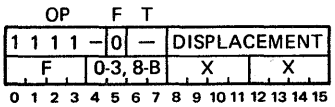
The contents of the effective address are not changed.

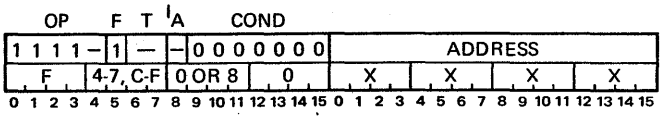
$$(EA) \cup (A) \longrightarrow (A)$$

Indicators: Carry and Overflow are not affected.

Affected: (A)

EOR LOGICAL EXCLUSIVE OR **2.4 μs**





The Logical Exclusive Or instruction logically compares the contents of the effective address and

the contents of the accumulator, and loads the result into the accumulator. The logical comparison is performed as follows:

Bit at Effective Address	Corresponding Bit in Accumulator	Result
0	0	0
0	1	1
1	0	1
1	1	0

The contents of the effective address are not changed.

$$(EA) \oplus (A) \longrightarrow (A)$$

Indicators: Carry and Overflow are not affected.

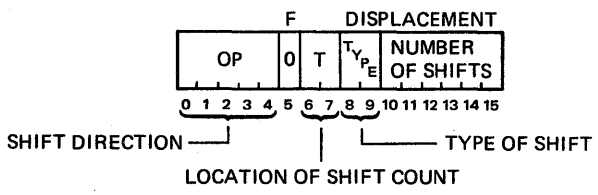
Affected: (A)

3.4 SHIFT INSTRUCTIONS

The following shift instructions are available to the programmer:

SLA	Shift Left Logical A
SLT	Shift Left Logical A and Q
SLCA	Shift Left and Count A
SLC	Shift Left and Count A and Q
SRA	Shift Right Logical A
SRT	Shift Right A and Q
RTE	Rotate Right A and Q

All shift instructions are single word instructions with the following format:



The operation code (OP), along with bits 8 and 9 (TYPE) of the displacement field, specifies the direction and type of shift operation. The contents of these fields are shown in the format diagrams for each instruction. The tag (T) field specifies the location of the shift count as follows:

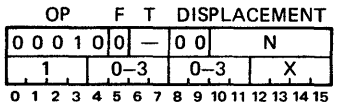
Tag	Location of Shift Count
00	Bits 10 through 15 of displacement
01	Bits 10 through 15 of XR1
10	Bits 10 through 15 of XR2
11	Bits 10 through 15 of XR3

The execution time for each shift instruction depends on the number of shifts performed. The time is determined from the following formula (where N equals the number of shifts):

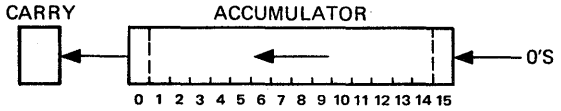
$$\text{Execution time } (\mu s) = 1.2 + 1.2(N/4)$$

For example, the execution of a shift instruction specifying a shift of 5, 6, 7 or 8 places requires 3.6 microseconds (N/4 must be an integer).

SLA SHIFT LEFT LOGICAL A **1.2 + 1.2(N/4) μs**



The Shift Left Logical A instruction shifts the contents of the accumulator to the left (toward bit 0). The number of bit positions shifted is specified by the shift count (low-order six bits of displacement, XR1, XR2, or XR3 as indicated by the tag field). As the contents are shifted, the bits shifted out of bit position 0 are shifted into the Carry indicator, and zeros are copied into vacated bit positions.

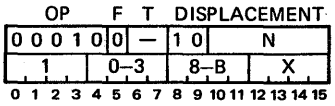


If the shift count specified by the instruction is zero, the instruction performs no operation.

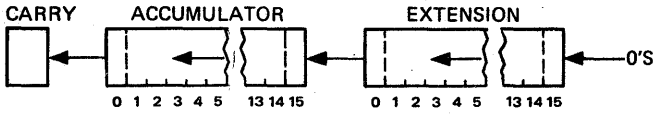
Indicators: Carry is set each time a 1 is shifted out of bit position 0 and reset each time a 0 is shifted out. Overflow is not affected.

Affected: (A)

SLT SHIFT LEFT LOGICAL A AND Q 1.2 + 1.2(N/4) μs



The Shift Left Logical A and Q instruction shifts the contents of the accumulator and accumulator extension (treated as one 32-bit double precision register) to the left (toward bit 0 of the accumulator). The number of bit positions shifted is specified by the shift count (low-order six bits of displacement, XR1, XR2, or XR3 as indicated by the tag field). As the contents are shifted, the bits shifted out of bit position 0 of the accumulator are shifted into the Carry indicator, and zeros are copied into vacated bit positions.

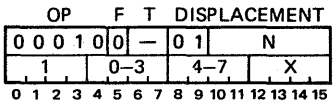


If the shift count specified by the instruction is zero, the instruction performs no operation.

Indicators: Carry is set each time a 1 is shifted out of bit position 0 of the accumulator and reset each time a 0 is shifted out. Overflow is not affected.

Affected: (A) and (Q)

SLCA SHIFT LEFT AND COUNT A 1.2 + 1.2(N/4) μs



The Shift Left and Count A instruction performs exactly as the Shift Left Logical A instruction if the tag (T) field is zero. If the tag field contents are other than zero, the instruction operates as follows:

a. Before shifting commences, the shift count is transferred from the six low-order bits of the specified index register to the shift counter.

b. The contents of the accumulator are shifted to the left (toward bit position 0), and the contents of the shift counter are decreased by one count each time a shift of one position is made. As the contents are shifted, zeros are copied into vacated bit positions.

c. Shifting continues until either the contents of the shift counter reach zero or until an attempt is made to shift a 1 out of bit position 0 of the accumulator. For the latter case the 1 remains in bit position 0.

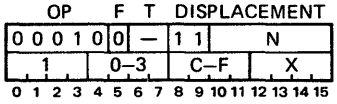
d. After shifting stops, the contents of the shift counter are copied back into the six low-order bit positions of the index register, and bits 8 and 9 of the index register are reset to zero. Bits 0 through 7 are not changed.

If the shift count initially specified by the instruction is zero or if bit position 0 of the accumulator initially contains a 1, the instruction performs no operation.

Indicators: For a tag field of zero, the Carry indicator behaves as described for the Shift Left Logical A instruction. For tag fields other than zero, Carry is reset at the end of the instruction if a 0 is in bit position 0 of the accumulator. It is set at the end of the instruction if a 1 is in bit position 0 of the accumulator. Overflow is not affected.

Affected: (A)

SLC SHIFT LEFT AND COUNT A AND Q 1.2 + 1.2(N/4) μs

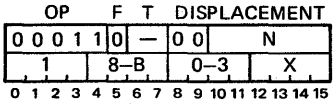


The Shift Left and Count A and Q instruction operates the same as Shift Left and Count A except that both the accumulator and accumulator extension are shifted as one 32-bit double precision register. Bits shifted out of bit position 0 of the extension are shifted into bit position 15 of the accumulator, and zeros are copied into vacated positions.

Indicators: Same as Shift Left and Count A.

Affected: (A) and (Q)

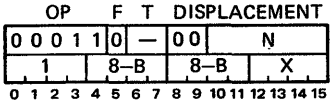
SRA SHIFT RIGHT LOGICAL A 1.2 + 1.2(N/4) μs



The Shift Right Logical A instruction shifts the contents of the accumulator to the right (toward bit 15). The number of bit positions shifted is

specified by the shift count (six low-order bits of displacement, XR1, XR2, or XR3 as indicated by the tag field). As the contents are shifted, zeros are copied into vacated bit positions, and the bits shifted out of bit position 15 are lost.

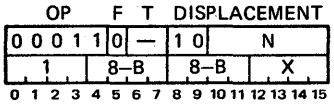
If a shift count of zero is specified the instruction performs no operation.



Indicators: Carry and Overflow are not affected.

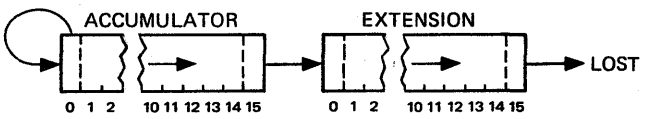
Affected: (A)

SRT SHIFT RIGHT A AND Q **1.2 + 1.2(N/4) μs**



The Shift Right A and Q instruction shifts the contents of the accumulator and accumulator extension (treated as one 32-bit double precision register) to the right (toward bit 15 of the extension). The number of bit positions shifted is specified by the shift count (six low-order bits of displacement, XR1, XR2, or XR3 as indicated by the tag field). As the contents are shifted, the value of the sign bit (bit position 0 of the accumulator) is copied into vacated positions, and the bits shifted out of bit position 15 of the extension are lost.

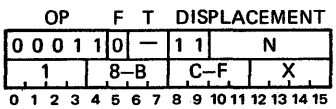
If a shift count of zero is specified the instruction performs no operation.



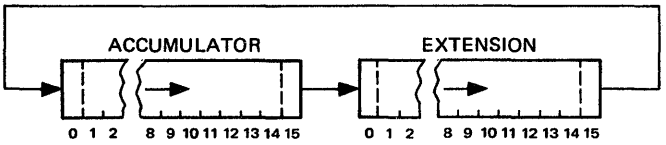
Indicators: Carry and Overflow are not affected.

Affected: (A) and (Q)

RTE ROTATE RIGHT A AND Q **1.2 + 1.2(N/4) μs**



The Rotate Right A and Q instruction shifts the contents of the accumulator and accumulator extension (treated as one 32-bit double precision register) to the right (toward bit 15 of the extension). The bits shifted out of bit 15 of the extension are recirculated and copied into bit position 0 of the accumulator. The number of bit positions shifted is specified by the shift count (six low-order bits of displacement, XR1, XR2, or XR3 as indicated by the tag field). If a shift count of zero is specified the instruction performs no operation.



Indicators: Carry and Overflow are not affected.

Affected: (A) and (Q)

3.5 BRANCH INSTRUCTIONS

The following branch instructions are available to the programmer:

BSI	Branch and Store Instruction Register
BSC (BOSC)	Branch or Skip on Condition (Branch Out of Interrupt)
MDX	Modify Index and Skip
CMP	Compare
DCM	Compare Double

All of the branch instructions can be used as either single or double word instructions. Indirect addressing can be specified for all of the double word instructions except Modify Index and Skip. The indirect address (IA) bit is used in this instruction, but its interpretation is different from that of normal indirect addressing. The way in which this bit is used is explained in the description of the Modify Index and Skip instruction.

Associated with the accumulator are six testable indicators that can be used to control conditional skip or branch operations. These indicators can be tested by the double word BSI instruction and by both the single and double word BSC instructions. Bits 10 through 15 of the displacement or condition field of the instruction specify the condition or conditions to be tested as follows:

Bit	Condition
10	Accumulator zero
11	Accumulator negative
12	Accumulator positive (greater than zero)
13	Accumulator even
14	Carry indicator reset (off)
15	Overflow indicator reset (off)

Testing the accumulator indicators with the single word BSC instruction causes a one-word skip to occur if one or more of the specified conditions are true. If none of the specified conditions are true, the skip does not occur; instead, the normal program sequence is followed.

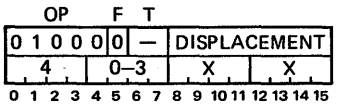
Testing the accumulator indicators with the double word BSI or BSC instruction causes a branch to occur if none of the specified conditions are true. If one or more of the specified conditions are true, the branch does not occur and the normal program sequence is followed.

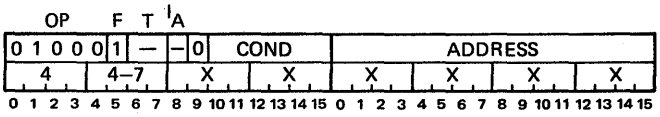
Table 3-1 shows the bit configurations of the displacement or condition field for testing the various conditions of the accumulator.

Table 3-1. Accumulator Test Conditions

Condition Bits						Skip Condition	Branch Condition
10 Zero	11 Minus	12 Plus	13 Even	14 Carry	15 O'flo		
0	0	0	0	0	0	Never	Always
0	0	0	0	0	1	No overflow	Overflow
0	0	0	0	1	0	No carry	Carry
0	0	0	0	1	1	No overflow or carry	Overflow and carry
0	0	0	1	0	0	Even	Odd
0	0	1	0	0	0	Plus	Minus or zero
0	0	1	1	0	0	Even or Plus	Odd and minus
0	1	0	0	0	0	Minus	Plus or zero
0	1	0	1	0	0	Even or minus	Odd and plus
0	1	1	0	0	0	Not zero	Zero
1	0	0	0	0	0	Zero	Not zero
1	0	1	0	0	0	Not minus	Minus
1	1	0	0	0	0	Not plus	Plus
1	1	1	0	0	0	Always	Never

BSI **BRANCH AND STORE INSTRUCTION REGISTER** **1.32 or 2.4 μs**





The Branch and Store Instruction Register instruction is an unconditional branch instruction when the single word version is used and a conditional branch instruction when the double word version is used.

The single word instruction stores the contents of the instruction counter (address of next instruction) at the effective address and then transfers control to the instruction stored at the location immediately following the effective address (effective address plus one). The stored address provides a link for returning control from a subroutine back to the point in the program at which the branch occurred. An indirect Branch or Skip on Condition instruction is used to return from the subroutine. When this instruction is executed, the link address is fetched from memory and restored in the instruction counter. Program execution then resumes with the instruction following the original branch instruction. Figure 3-1 shows an example of the use of the Branch and Store Instruction Register instruction.

The double word instruction performs the same operation as the single word instruction, but its execution conditionally depends on the result of the test specified by the condition field of the instruction (paragraph 3.5). If any one of the specified conditions is true, the instruction does not perform the storing and branch operation; instead, program execution continues with the next instruction in normal sequence. If none of the conditions are true, the instruction performs the storing and branch operations as described for the single word instruction.

All interrupts are inhibited following the execution of this instruction; thus, the instruction immediately following a BSI instruction must be executed before any interrupt can occur. For additional information on interrupt polling and acknowledgement during the execution of a BSI instruction, refer to paragraph 4.19.

The execution time for the single word instruction, and for the double word instruction when the branch condition is met, is 2.4 microseconds. The double word instruction requires 1.32 microseconds when the branch condition is not met.

Single word instruction:

$$(I) \longrightarrow (EA)$$

$$(EA+1) \longrightarrow (I)$$

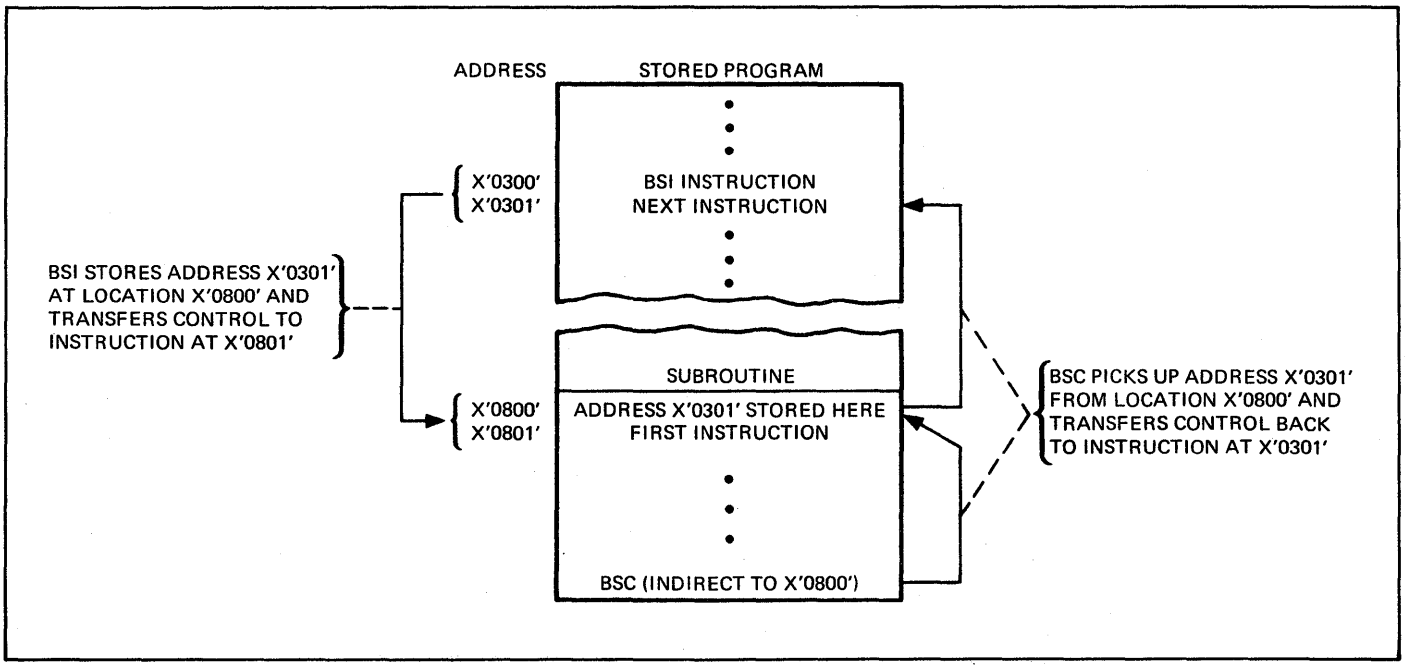


Figure 3-1. Example of BSI Instruction

Double word instruction:

Condition true: Continue normal instruction sequence

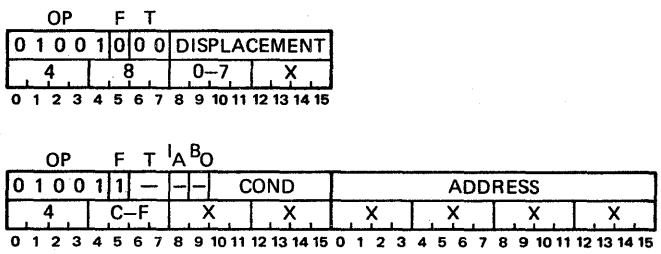
Condition false: (I) ⟶ (EA)

(EA+1) ⟶ (I)

Indicators: For the single word instruction, Carry and Overflow are not affected. For the double word instruction, Overflow is reset if tested.

Affected: (I) and (EA)

BSC (BOSC) BRANCH OR SKIP ON CONDITION (BRANCH OUT OF INTERRUPT) **1.2 or 1.32 μs**



The Branch or Skip on Condition instruction is a conditional skip instruction when the single word version is used and a conditional branch instruction when the double word version is used. A special version of the double word instruction, called Branch Out of Interrupt (BOSC), is used to reset the highest priority active interrupt after it has been serviced.

The single word instruction tests for the condition or conditions specified by bits 10 through 15 of the displacement field (paragraph 3.5) and skips the next single word instruction if any condition is true. If none of the conditions are true, program execution continues with the next instruction in normal sequence.

The double word instruction tests for the condition or conditions specified by the condition field (paragraph 3.5) and branches to the effective address if none of the specified conditions are true. If any condition is true, program execution continues with the next instruction in normal sequence.

The double word instruction with indirect addressing specified (IA = 1) is normally used to return control from a subroutine or interrupt servicing routine to the main program or point of interruption through the link address stored by the BSI instruction that caused entry into the subroutine or servicing routine.

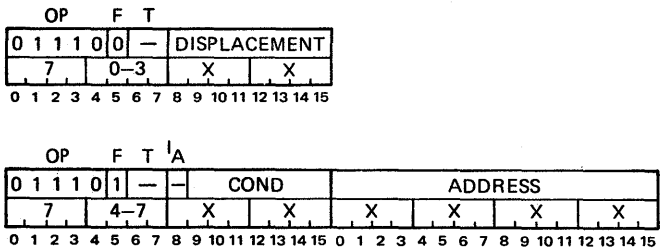
When bit BO of the double word instruction is a 1 (Branch Out of Interrupt), the instruction not only performs the conditional branch operation, but also resets the highest priority active interrupt level. This instruction is normally used at the end of the interrupt servicing routine to transfer control back to the main program. Resetting the active interrupt permits interrupts of equal or lower priority to be accepted and serviced. If the active interrupt is not reset, interrupts of equal or lower priority are recorded by the interrupt logic, but they cannot be accepted until the active interrupt is reset; therefore, it is important that the BOSC instruction, and not the BSC instruction, be used for branching out of an interrupt servicing routine. It is also important to remember that the interrupt is reset only if the conditional branch occurs. If the branch is not taken, the interrupt is not reset. If the conditional branch occurs, all interrupts are inhibited for one instruction.

The execution time for the single word instruction is 1.2 microseconds when the skip condition is not met, and 1.32 microseconds when the condition is met. The execution time for the double word instruction is 2.4 microseconds when the branch condition is met, and 1.32 microseconds when it is not met.

Indicators: Carry is not reset if tested. Overflow is reset when tested.

Affected: (I)

MDX MODIFY INDEX AND SKIP **1.2 μs**



The Modify Index and Skip instruction modifies either the contents of an index register, the instruction counter, or a memory location and then, depending on the modified result, either continues program execution with the next instruction in normal sequence or advances the instruction counter an additional count before continuing the program sequence.

The single word instruction can be used to modify an index register or the instruction counter by the addition of the value of the expanded displacement.

The double word instruction can be used to modify an index register by the addition of the contents of a memory location, the address field of the instruction, or the expanded displacement. It can also be used to modify the contents of a memory location by the addition of the expanded displacement.

The various operations performed by this instruction are determined by the value of the tag (T) field and IA bit as follows:

Single Word Instruction

Tag	Operation
00	The expanded displacement is added to the contents of the instruction counter. No skip occurs.
01, 10, or 11	The expanded displacement is added to the contents of the specified index register. If the contents of the index register equal zero after the operation or if a change in sign occurs, the instruction counter is advanced an additional count. For correct operation this version of the instruction should always be followed by a single word instruction.

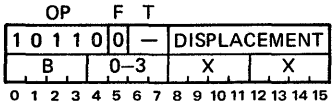
Double Word Instruction

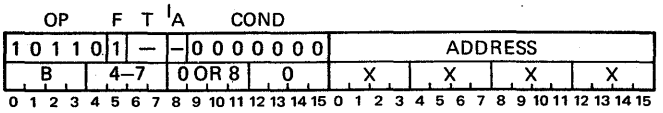
Tag	IA	Operation
00	0	The expanded displacement is added to the contents of the effective address. If the contents of the effective address equal zero after the operation or if a change in sign occurs, the instruction counter is advanced an additional count. For correct operation this version of the instruction should always be followed by a single word instruction.
01, 10 or 11	0	The address field of the instruction is added to the contents of the specified index register. If the contents of the index register equal zero after the operation or if a change in sign occurs, the instruction counter is advanced an additional count. For correct operation this version of the instruction should always be followed by a single word instruction.
00	1	The expanded displacement is added to the contents of the effective address. If the contents of the effective address equal zero after the operation or if a change in sign occurs, the instruction counter is advanced an additional count. For correct operation this version of the instruction should always be followed by a single word instruction.
01, 10, or 11	1	The contents of the location specified by the address field are added to the specified index register. If the contents of the index register equal zero after the operation or if a change in sign occurs, the instruction counter is advanced an additional count. For correct operation this version of the instruction should always be followed by a single word instruction.

Indicators: Carry and Overflow are not affected.

Affected: (I), (XR1), (XR2), or (XR3) for a single word instruction; (EA), (XR1), (XR2), or (XR3) for a double word instruction.

CMP COMPARE **2.4 μs**





The Compare instruction algebraically compares the contents of the accumulator with the contents of the effective address and then performs a skip of 0, 1, or 2 memory words depending on the result of the comparison as follows:

If (A) (EA), no skip occurs

If (A) (EA), a one-word skip occurs

If (A) (EA), a two-word skip occurs

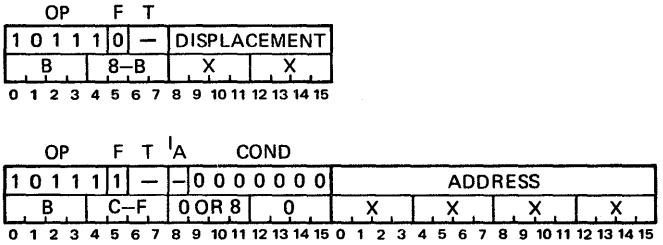
Execution of this instruction requires 2.52 microseconds if a one-word or two-word skip occurs.

For correct operation this instruction should always be followed by a single-word instruction.

Indicators: Carry may be affected. Overflow is not affected.

Affected: (I)

DCM COMPARE DOUBLE 3.6 μs



The Compare Double instruction algebraically compares the contents of the accumulator and the accumulator extension with the contents of the effective address and the next higher address (effective address plus one) and then performs a skip of 0, 1, or 2 memory words depending on the result of the comparison as follows:

If (A, Q) > (EA, EA+1), no skip occurs

If (A, Q) < (EA, EA+1), a one-word skip occurs

If (A, Q) = (EA, EA+1), a two-word skip occurs

For correct operation the effective address of this instruction must be an even address. This instruction should always be followed by a single word instruction.

Indicators: Carry may be affected. Overflow is not affected.

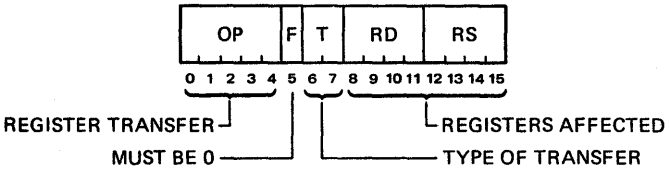
Affected: (I)

3.6 REGISTER TRANSFER INSTRUCTIONS

The following register transfer instructions are available to the programmer:

RTR	Register Transfer
RCP	Register Transfer and Complement
RIC	Register Transfer and Increment
RDC	Register Transfer and Decrement

The register transfer instructions are single word instructions with the following format:



The operation code (OP), along with the tag (T) field, specifies the type of register transfer operation to be performed as follows:

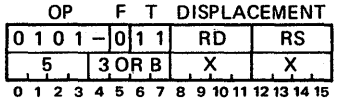
Tag	Operation
00	Transfer and complement
01	Transfer and increment
10	Transfer and decrement
11	Transfer

The registers selected for the operation are specified by the RD (destination register) and RS (source register) fields as follows:

RD or RS (Hex)	Register
0	I
1	XR1
2	XR2
3	XR3
4	A
5	Q
6	CAR1
7	SCR1
8	CAR2
9	SCR2

RD or RS (Hex)	Register
A	CAR3
B	SCR3
C	CAR4
D	SCR4
E	CAR5
F	SCR5

RTR REGISTER TRANSFER 2.4 μs



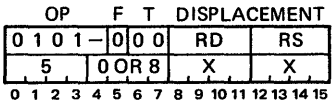
The Register Transfer instruction copies the contents of the source register (specified by RS) into the destination register (specified by RD). The contents of the source register are not changed.

$$(RS) \longrightarrow (RD)$$

Indicators: Carry and Overflow are not affected.

Affected: (A), (Q), (I), (XR1), (XR2), (XR3), (CAR1) through (CAR5), or (SCR1) through (SCR5) if specified as the destination register.

RCP REGISTER TRANSFER AND COMPLEMENT 2.4 μs



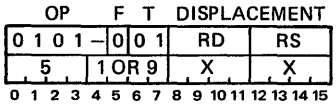
The Register Transfer and Complement instruction copies the one's complement of the contents of the source register (specified by RS) into the destination register (specified by RD). The contents of the source register are not changed.

$$\overline{(RS)} \longrightarrow (RD)$$

Indicators: Carry and Overflow are not affected.

Affected: (A), (Q), (I), (XR1), (XR2), (XR3), (CAR1) through (CAR5), or (SCR1) through (SCR5) if specified as the destination register.

RIC REGISTER TRANSFER AND INCREMENT 2.4 μs



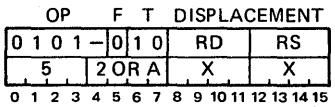
The Register Transfer and Increment instruction copies the contents of the source register (specified by RS) into the destination register (specified by RD) and then adds 1 to the new contents of the destination register. The contents of the source register are not changed.

$$(RS) + 1 \longrightarrow (RD)$$

Indicators: Carry and Overflow are not affected.

Affected: (A), (Q), (I), (XR1), (XR2), (XR3), (CAR1) through (CAR5), or (SCR1) through (SCR5) if specified as the destination register.

RDC REGISTER TRANSFER AND DECREMENT 2.4 μs



The Register Transfer and Decrement instruction copies the contents of the source register (specified by RS) into the destination register (specified by RD) and then subtracts 1 from the new contents of the destination register. The contents of the source register are not changed.

$$(RS) - 1 \longrightarrow (RD)$$

Indicators: Carry and Overflow are not affected.

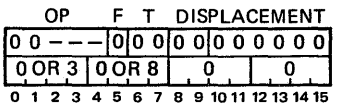
Affected: (A), (Q), (I), (XR1), (XR2), (XR3), (CAR1) through (CAR5), or (SCR1) through (SCR5) if specified as the destination register.

3.7 INPUT/OUTPUT AND CONTROL INSTRUCTIONS

Three control instructions are available to the programmer: Wait, No Operation, and Execute I/O. The Wait, and No Operation instructions are single word instructions. Execute I/O can be used either as a single word instruction or as a double word instruction, and indirect addressing can be specified for the double word version.

For detailed information on input/output and control operations, refer to section IV.

WAIT **WAIT** **1.2 μs**



Two versions of the Wait instruction can be executed: a standard Wait (operation codes 30 or 38) or a pseudo Wait (operation code 00). Both are single word instructions only.

When the standard Wait instruction is executed, the processor ceases operation, except for direct memory data and interval timing operations, until an interrupt occurs or until the computer is manually restarted from the programmer's console. When an interrupt occurs, the computer executes a forced BSI instruction to enter the interrupt servicing routine. The last instruction of the servicing routine is normally a Branch Out of Interrupt (BOSC) instruction that, when executed, transfers control to the next instruction in sequence following the Wait instruction.

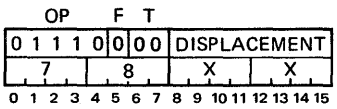
The pseudo Wait instruction is identical to the standard Wait except that all interrupts are inhibited while the computer is in the wait condition.

The wait condition caused by either the standard Wait or pseudo Wait instruction can be cleared from the programmer's console by unlocking the CONSOLE ENABLE switch and then pressing the STEP switch.

Indicators: Carry and Overflow are not affected.

Affected: None

NOP **NO OPERATION** **1.2 μs**



The No Operation instruction causes the instruction counter to advance one count. Program execution then continues with the next instruction in normal sequence.

For correct operation the tag (T) field of this instruction must be zero. If it is not, the instruction counter could advance a second count under either of the following conditions:

a. If the contents of the instruction counter plus the expanded displacement would equal zero.

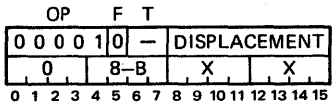
b. If the contents of the instruction counter plus the expanded displacement would change sign.

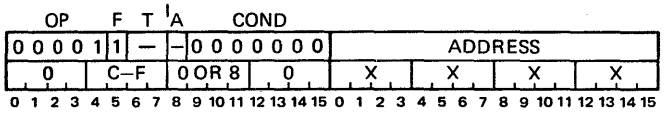
The No Operation instruction is a single word instruction only.

Indicators: Carry and Overflow are not affected.

Affected: None

XIO **EXECUTE I/O** **3.6 or 4.8 μs**





The Execute I/O instruction is a multipurpose instruction that performs the operation specified by a double word I/O Control Command (IOCC) stored at the effective address and the next higher address (effective address plus one). For correct operation the effective address must be an even address.

The IOCC double word specifies one of the following types of operations and provides the address information required for the operation:

a. Sense device

b. Sense interrupt level

c. Control (internal or external)

d. Read

e. Write

f. Initialize read (direct memory data channel)

g. Initialize write (direct memory data channel)

Detailed descriptions of these operations and the IOCC double word format for each operation are contained in section IV.

All interrupts are inhibited following the execution of an XIO instruction. Therefore, the instruction immediately following an XIO instruction must be executed before any interrupt can occur. For additional information on interrupt polling and acknowledgment during the execution of an XIO instruction, refer to paragraph 4.19.

The execution time of the single word XIO instruction for sense device, sense interrupt level, and control operations is 3.6 microseconds. For single word read and write, the execution time is 4.8 microseconds. For single word initialize read and initialize write the execution time is 6.0 microseconds. An additional 1.2 microseconds is required for double word instructions and for indirect addressing.

Indicators: Carry and Overflow are not affected.

Affected: (A) for sense device and sense interrupt level operations only.

SECTION IV
INPUT/OUTPUT AND CONTROL OPERATIONS

4.1 GENERAL

All input and output operations of the computer, including direct memory transfer, interrupt control, and certain internal control operations, are initiated by one multipurpose, input/output instruction — Execute I/O (XIO). This instruction can be used in either of two ways: (1) to perform directly an internal control or single word input/output operation, or (2) to set up the direct memory data channels for block-transfer operations.

When the XIO instruction is used for internal control or single word input/output operations, each specific operation is accomplished by the execution of one XIO instruction. The operation performed by the instruction can be any one of the following types:

a. Sense. A status word representing the operational status of an I/O device, process, or internal condition is read into the accumulator, or a status word indicating those devices requesting interrupt recognition on an interrupt level is read into the accumulator.

b. Control. The operating condition of an I/O device or internal feature is changed.

c. Read. One word is transferred from an I/O device into a specified memory location.

d. Write. One word is transferred from a specified memory location to an I/O device.

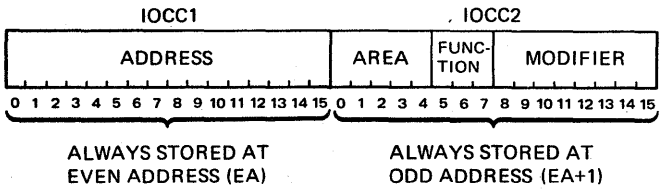
A direct memory transfer operation may be initialized by an XIO instruction. The XIO instruction may perform either one of the following functions:

a. Initialize Read. The conditions that are required for a device to transfer single or multiple blocks of data words directly into memory are established. Thereafter, the actual transfer of data is controlled by the direct memory data channel and the I/O device without further program intervention.

b. Initialize Write. The conditions that are required to transfer single or multiple blocks of data words directly from memory to an I/O device are established. Thereafter, the actual transfer of data is controlled by the direct memory data channel and the I/O device without further program intervention.

4.2 INPUT/OUTPUT CONTROL COMMANDS

The Execute I/O instruction by itself does not specify a particular input/output or control operation. Instead, the effective address of the instruction is used to fetch from memory a double word input/output control command (IOCC) that specifies the operation and all parameters defining the operation to be performed. The IOCC double word has the following format:



Field	Description
Area	Specifies the I/O device or I/O group affected by the operation. If the area code specifies an I/O group, the modifier field is used to designate a particular unit within the group. An area code of zero (00000) is used for the special operations described in paragraph 4.3.
Function	Specifies the operation to be performed as follows:

Code	Operation
000	Reserved for special uses.
001	Write. This code causes a single word to be transferred from the memory location specified by the address field (IOCC1) to the I/O device.
010	Read. This code causes a single word to be transferred from the I/O device to the memory location specified by the address field (IOCC1).

Field	Description
	011 Sense interrupt level. This code causes the interrupt level status word (ILSW) for the highest active interrupt level to be read into the accumulator. The previous contents of the accumulator are destroyed. For detailed information on interrupts, see paragraph 4.10.
	100 Control. This code causes the I/O device or computer to perform the control operation specified by the address or modifier field.
	101 Initialize write. This code sets up the I/O device and direct memory data channel for transferring blocks of data words from memory to the I/O device. When this function is used, bits 8 through 10 of the modifier field specify the direct memory data channel to be used.
	110 Initialize read. This code sets up the I/O device and direct memory data channel for transferring blocks of data words from the I/O device to memory. When this function is used, bits 8 through 10 of the modifier field specify the direct memory data channel to be used.
	111 Sense device. This code causes the current status of the I/O device or process to be read into the accumulator. The previous contents of the accumulator are destroyed.
Modifier	Provides additional detail not specified by the area or function fields. Specific uses of the modifier field for IOCC's with an area code of zero are given in paragraph 4.3.
Address	Provides the following information depending on the function specified: a. For initialize write (101) or initialize read (110) function, the address field specifies the starting address of the I/O table containing the control words and data for a direct memory transfer operation. b. For a control (100) function the address field can be used to designate a specific control operation. c. For a sense interrupt level (011) or sense device (111) function, the address field is not used. d. For a write (001) or read (010) function, the address field specifies the memory location from which or into which the data word is transferred.

4.3 STANDARD IOCC DOUBLE WORDS

Although the IOCC configurations are usually tailored for each individual application, certain standard configurations have been preassigned. Standard IOCC double words have been preassigned for the following operations:

a. Start/stop interval timers

b. Sense interval timers

c. Sense console data switches

d. Read console data switches

e. Mask/unmask interrupt levels

f. Set program interrupt register

g. Sense interrupt level status word

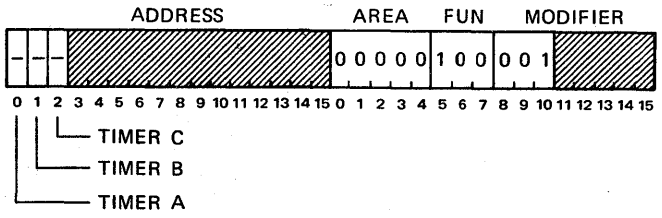
h. Sense console interrupt status word

i. Reset operations monitor alarm

j. Initialize Teletype or inhibit direct memory transfer

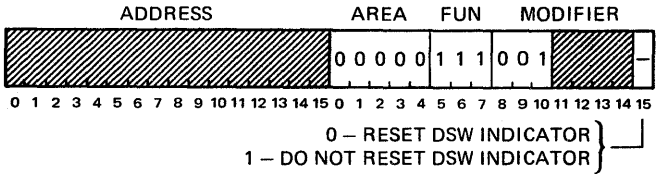
These operations are specified by IOCC's with an area code of zero and bits 8 through 10 of the modifier field designating the operation as shown in the following descriptions:

START/STOP INTERVAL TIMERS

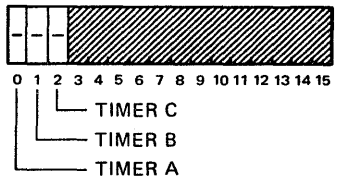


This IOCC is used to start or stop the interval timers. Bits 0 through 2 designate timer A (memory location X'0004'), timer B (memory location X'0005'), and timer C (memory location X'0006'), respectively. A 1-bit in the designated bit position starts the corresponding timer operating; a 0-bit stops the timer. For detailed information on interval timer operation, see paragraph 2.19.

SENSE INTERVAL TIMERS STATUS WORD

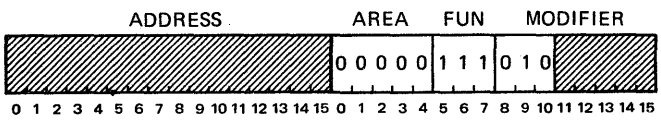


This IOCC copies the device status word (DSW) for the interval timers into the accumulator. The format of the status word is as follows:



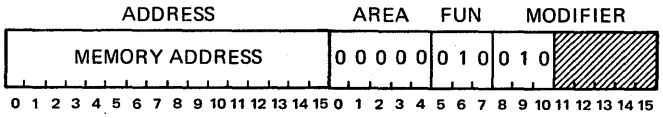
Bits 0 through 2 of the DSW identify the timer requesting the interrupt. A 1-bit indicates that an interrupt has been requested by the corresponding timer. Modifier bit 15 of the IOCC specifies whether the DSW indicators are reset as the status word is copied into the accumulator. They are reset if modifier bit 15 contains a 1. The previous contents of the accumulator are destroyed by this IOCC.

SENSE CONSOLE DATA SWITCHES



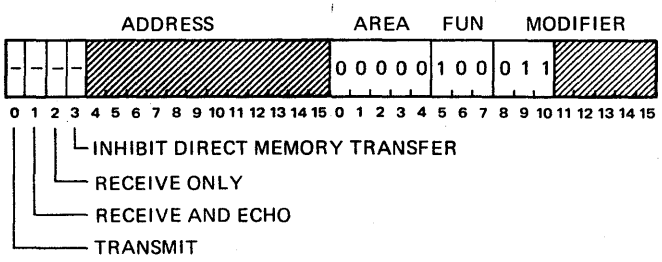
This IOCC copies the status of console data switches 0 through 15 into the correspondingly numbered bit positions of the accumulator. The previous contents of the accumulator are destroyed.

READ CONSOLE DATA SWITCHES



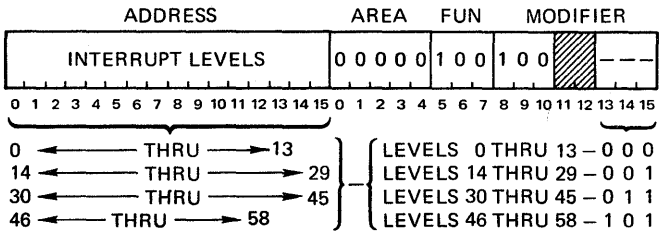
This IOCC stores the status of console data switches 0 through 15 at the memory location specified by IOCC1. The contents of the accumulator are not affected.

INITIALIZE TELETYPE OR INHIBIT DIRECT MEMORY TRANSFER



This IOCC is used to prepare the Teletype for transmitting or receiving data (with or without echo) or to inhibit direct memory transfer operations, as specified by bits 0 through 3 of IOCC1. A 1-bit in any of these bit positions causes the specified operation. The operation is undefined if a 1 appears in more than one of the bit positions 0, 1 or 2.

MASK/UNMASK INTERRUPT LEVELS



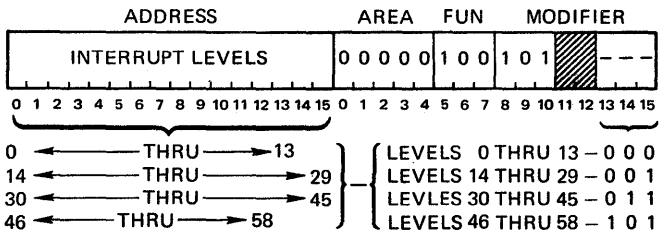
This IOCC is used to mask or unmask the external interrupt levels (internal and trace interrupts cannot be masked). The status of bit positions 0 through 13, 0 through 15, or 0 through 12 of IOCC1, depending on modifier bits 13 through 15, are copied into the interrupt mask register. Each bit position represents

one interrupt level, and the status of the bit specifies whether the level is masked (1) or unmasked (0). The level represented by each bit position is determined by modifier bits 13 through 15 as follows:

Modifier Bits 13 thru 15	Interrupt Levels
000	Bits 0 through 13 of IOCC1 represent interrupt levels 0 through 13
001	Bits 0 through 15 of IOCC1 represent interrupt levels 14 through 29
011	Bits 0 through 15 of IOCC1 represent interrupt levels 30 through 45
101	Bits 0 through 12 of IOCC1 represent interrupt levels 46 through 58

When an interrupt level is masked (corresponding bit position of IOCC1 contains a 1), all interrupt requests on that interrupt level are prevented from being acknowledged until another IOCC is executed to unmask the level. All external interrupt levels are automatically masked when the RESET switch on the programmer's console is pressed.

SET INTERRUPT REGISTER



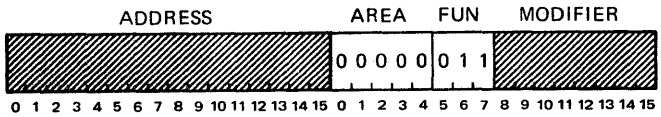
This IOCC triggers an external interrupt or interrupts from within the program by setting the program interrupt register (PIR). The PIR has one bit for each interrupt level. The status of the bit specifies whether the level is triggered (1) or not (0). The status of bit positions 0 through 13, 0 through 15, or 0 through 12 of IOCC1, depending on modifier bits 13 through 15, designate the bits of the PIR to be set. If more than one interrupt level is triggered, only the highest unmasked level is recognized. Once a bit of the PIR has been set, it remains set until either the associated interrupt level becomes active or it is reset by a subsequent set-PIR function. The PIR is not affected by the interrupt mask register. The entire PIR is reset when the RESET switch on the programmer's console is pressed. The PIR retains information even though the interrupt mask register is set. It is therefore possible to set a programmed interrupt at an interrupt level which is masked or becomes masked before the interrupt occurs. Subsequently, when the interrupt level becomes unmasked, the programmed interrupt will occur.

The level represented by each bit position of the PIR is exactly the same as in the Mask/Unmask Interrupt Levels IOCC. The specified interrupt levels are triggered only if they are unmasked and do not become masked before the forced interrupt BSI instruction is executed. Since this command triggers the interrupt level itself and not an individual interrupt, no bits are turned on in the interrupt level status word. For detailed information on interrupt operation, see paragraph 4.10.

NOTE

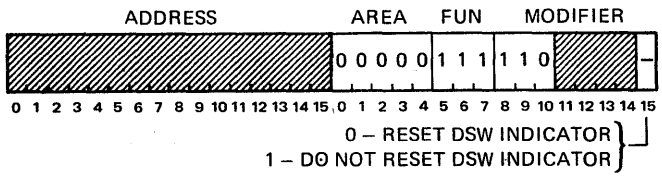
For correct operation the program-triggered interrupt must be allowed to occur as soon as possible since no other interrupts can occur in the meantime.

SENSE INTERRUPT LEVEL STATUS WORD

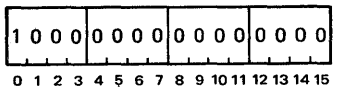


This IOCC copies the status word for the highest priority interrupt level requesting service into the accumulator. The status of the indicators in the devices assigned to the ILSW are not reset. The previous contents of the accumulator are destroyed by this IOCC.

SENSE CONSOLE INTERRUPT STATUS WORD

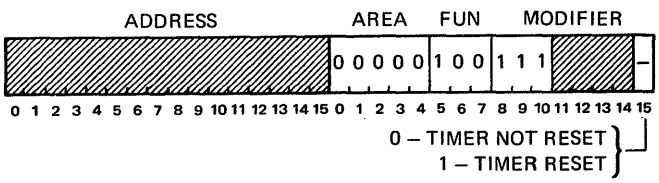


This IOCC copies the device status word (DSW) for the console interrupt into the accumulator. The format of the status word is as follows:



Modifier bit 15 of the IOCC specifies whether the DSW indicator is reset as the status word is copied into the accumulator. It is reset if modifier bit 15 contains a 1. The previous contents of the accumulator are destroyed by this IOCC.

RESET OPERATIONS MONITOR ALARM (STALL ALARM)



This IOCC resets the stall alarm timer to zero each time it is executed with modifier bit 15 set to 1. If bit 15 is set to 0, the timer is not reset. For detailed information on the operation of the stall alarm, see paragraph 2.23.

4.4 SINGLE WORD I/O OPERATIONS

Transferring data between the computer and a low-speed I/O device is normally accomplished by executing one or more XIO instructions for each single word of data transferred. The XIO instruction with its corresponding IOCC can be used to issue a control command to a device, to test the operational status of a device, to transfer a data word to or from the device, or to test the interrupt level status word in response to a service request interrupt from a device.

An XIO instruction specifying a control function IOCC can be used to direct an I/O device to perform such control operations as feed card, rewind tape, select stacker, and so on.

An XIO instruction specifying a sense device function IOCC can be used to test the operational status of a device before a data transfer is made. Since it is possible to execute an XIO instruction to transfer data to an I/O device that is busy responding to a previous XIO instruction, it is up to the program to test the status of the device to ensure that no data is lost. Executing a sense device IOCC causes the addressed device to enter its status word into the accumulator. The status word can then be tested to determine the operating state of the device.

An XIO instruction specifying a read or write function IOCC can be used to transfer a single 16-bit data word between the computer and I/O device. The XIO instruction is terminated immediately after the data word is transferred. If more than one word is to be transferred, the I/O device normally initiates a service request interrupt to signal that it is ready to send or receive another word of data. The program then branches to an interrupt routine.

An XIO instruction specifying a Sense Interrupt Level IOCC is normally used in the interrupt routine to read the interrupt level status word into the accumulator. The status word is then tested to determine which device or process assigned to that level is requesting the interrupt. Once this is determined, an XIO instruction specifying a sense device IOCC can be used to determine if the interrupt is the result of a service request or an error condition.

For a description of interrupt processing and status words, refer to paragraph 4.10.

4.5 DIRECT MEMORY DATA CHANNELS

The available direct memory data channels provide the capability for transferring blocks of data words directly between memory and high-speed I/O devices with a minimum of program intervention. An XIO instruction specifying an initialize read or initialize write IOCC is used to establish the initial conditions for the transfer. Then normal program execution can continue while the I/O device and direct memory data channel control the transfer operation. When the I/O device is ready to transfer or receive a data word, the channel stops program execution for one memory cycle while one data word is transferred directly to or from memory. Program execution resumes until the device is ready for another transfer. The direct memory data channel keeps stealing memory cycles and transferring data words until the specified block or blocks of words have been transferred.

All direct memory data channels can be inhibited by an XIO instruction specifying the Inhibit Direct Memory Transfer IOCC. Once the channels are inhibited, they can be returned to normal by pressing the RESET switch on the programmer's console or by executing an XIO Initialize Read or Initialize Write instruction.

All five direct memory data channels can be set up to operate concurrently. When more than one channel requests service at the same time, the requests are serviced on the basis of channel priority with channel 1 having the highest priority and channel 5, the lowest. The maximum access time for the highest priority channel is 1.36 microseconds. The maximum access time for the lowest priority channel (with all other channels chaining) is 21 microseconds.

Although I/O devices can be connected to any channel, devices that are to be operated simultaneously must be assigned to separate channels.

4.6 BLOCK TRANSFER CONTROL

Associated with each direct memory data channel are two computer registers that are the primary control registers for block-transfer operations. These

are the channel address register (CAR1 through CAR5) and the scan control register (SCR1 through SCR5).

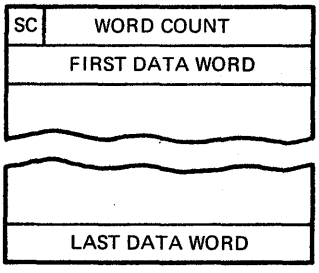
The channel address register is loaded with the starting address of a block of memory locations, called the I/O table, from which or into which the data is to be transferred. The address is loaded into the register by the XIO instruction that initializes the transfer operation. The contents of the channel address register are increased by one after each one-word transfer so that the register always contains the address from which or into which the next word is to be transferred.

The scan control register is loaded with the word count specifying the number of data words in the block to be transferred. The word count is initially stored in bits 2 through 15 of the first word in the first I/O table or in bits 2 through 15 of the second word of subsequent I/O tables if chaining is specified. The contents of the scan control register are decreased by one after each one-word transfer until the register contains zero. If chaining is not specified, the transfer operation is terminated at this point; if chaining is specified, the word count for the next block or the same block is loaded into the register and the transfer continues.

4.7 I/O TABLES

All block-transfer operations carried out through a direct memory data channel are performed to or from an I/O table in memory. Only one table is used for single-block transfer, but several tables (or the same table) may be used for multiple block transfers. The tables are chained together (hence the term chaining) by the last entry in each table. This entry contains the starting address of the next table. The tables are not restricted to any particular portion of memory.

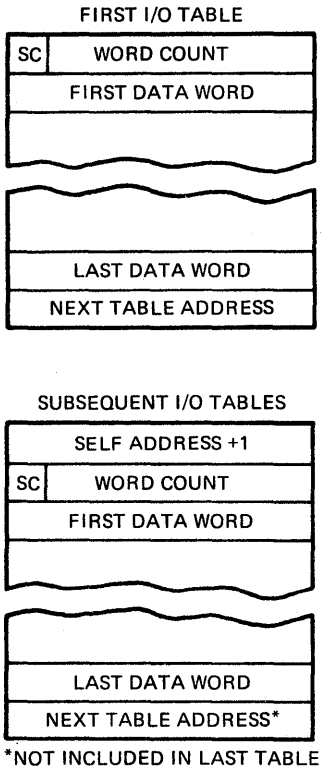
The format of the I/O table for a single-block transfer operation is shown in the following diagram:



The first word of the table contains two scan control (SC) bits and a 14-bit word count specifying the number of data words in the table. During the initialize or chaining cycles, the scan control bits are transferred to the I/O device where they are used to control the transfer operation, as follows:

SC	Operation
00	Transfer to or from this table and stop at end of table with an interrupt
01	Transfer to or from this table and stop at the end of table with no interrupt

The format of the I/O tables for a multiple-block transfer operation are shown in the following diagram:



The first word of the first table and the second word of subsequent tables contain the scan control bits and block word count. The following scan control bits are used for the first I/O table and all subsequent tables except the last:

SC	Operation
10	Transfer to or from this table and then chain to the beginning of this table or another table with an interrupt at the end of this table
11	Transfer to or from this table and then chain to the beginning of this table or another table with no interrupt

The scan control bits for the last table must be either 00 or 01 as defined for the single-block I/O table.

The last entry of all tables except the last contains the starting address of the next table. This address is loaded into the channel address register after the last data word in the current I/O table has been transferred.

The first word of all I/O tables except the first table contains its own address plus one. This word is used to check for correct chaining as described in paragraph 4.9.

4.8 SINGLE-BLOCK TRANSFER OPERATION

The sequence of operations for a single-block transfer operation proceeds as follows:

a. The program stores the scan control bits and word count in the first word of the I/O table.

b. The program executes an XIO instruction referencing an intialize read or initialize write function IOCC.

c. The area code of the IOCC selects the I/O device, and the device designates the data channel it will use. If no channel is designated by the device, channel DC5 is automatically addressed.

d. The address portion of the IOCC is loaded into the channel address register for the selected channel.

e. A memory cycle request is made, and the address is transferred from the channel address register to the memory location register.

f. The first word is fetched from the I/O table while the address in the channel address register is increased by one. The scan control bits (0 and 1) are transferred through the out bus to the I/O device, and the word count (bits 2 through 15) is transferred to the scan control register. With the completion of the first memory cycle, control is returned to the program.

g. When the I/O device is ready to transfer or receive the first data word, the direct memory data channel steals another memory cycle and the first data word is transferred to or from the I/O table. During this cycle the channel address register is increased by one and the word count in the scan control register is decreased by one. At the end of the cycle, control is again returned to the program. This sequence is repeated each time the I/O device is ready to send or receive another data word until the word count in the scan control register reaches zero or until the operation is stopped by the program with a control function XIO instruction.

4.9 MULTIPLE-BLOCK TRANSFER OPERATION (CHAINING)

The sequence of operations for a multiple-block transfer operation proceeds as follows:

a. The first I/O table is transferred as described in paragraph 4.8 except that the operation does not stop when the scan control register reaches zero. Instead, three additional memory cycles are used to chain from the first I/O table to the next one.

b. During the first memory cycle, the address stored in the location immediately following the last data word in the first I/O table is transferred into the channel address register. This address specifies the starting address of the next I/O table.

c. During the second cycle, the first word is fetched from the new I/O table. This word contains the starting address of the table plus one. The channel address register is then increased by one, and a comparison is made between the channel address register and the first word of the new I/O table to check for correct chaining. If the channel address register and first word do not agree, bit 4 of the internal interrupt level status word is turned on to activate the internal interrupt level, and the transfer operation is terminated.

d. During the third cycle, the second word is fetched from the new I/O table. The scan control bits from this word are transferred to the I/O device, and the word count is loaded into the scan control register. From this point on the transfer operation is again under control of the I/O device and direct memory data channel.

If more I/O tables are to be chained the operations described in steps a through d are performed to go from one table to another. When the last table is reached, the operation proceeds as for a single-block transfer.

4.10 INTERRUPTS

The interrupt feature of the computer permits the normal program sequence to be interrupted in response to a variety of internal or external stimuli that require special or immediate attention. The following types of conditions are typical of those that are used to interrupt normal operation:

a. Internal error conditions such as a memory parity error, a memory protection violation, or a direct memory data channel check error.

b. Completion of the timing period of an interval timer.

c. A request for service from an I/O device.

d. A change in the condition of a controlled process that necessitates an alternative program sequence.

4.11 INTERRUPT LEVELS

Up to 61 interrupt levels are normally available with each level having a unique memory location assignment (table 4-1) and a unique priority. The internal and trace interrupt levels plus six external interrupt levels (0 through 5) are provided as part of the standard interrupt feature. Up to 53 additional external interrupt levels (6 through 58) are optionally available.

a. Up to 16 interrupt request lines can be connected to each interrupt level (excluding trace).

b. Each interrupt level (excluding trace) may have an interrupt level status word (ILSW) of up to 16 bits to identify the source of the interrupt request.

c. Each interrupt request from an I/O device or process can result from more than one condition in the device or process.

d. Each I/O device or process has a 16-bit status word to identify, among other conditions, the specific condition responsible for an interrupt request. The status word for an I/O device is called a device status word (DSW); that for a process is called a process interrupt status word (PISW).

Table 4-1. Interrupt Levels

Interrupt Level*	Priority	Address		Availability
		Decimal	Hexadecimal	
Internal	1	8	8	Standard
Trace	62	9	9	Standard
External				
0	2	11	B	Standard
1	3	12	C	
2	4	13	D	
3	5	14	E	
4	6	15	F	
5	7	16	10	
6	8	17	11	Optional
7	9	18	12	
8	10	19	13	
9	11	20	14	
10	12	21	15	
11	13	22	16	
12	14	23	17	

*All levels except trace have interrupt level status words, and all levels except internal and trace can be masked.

Table 4-1. Interrupt Levels (Cont.)

Interrupt Level*	Priority	Address		Availability
		Decimal	Hexadecimal	
13	15	24	18	
14	16	25	19	
15	17	26	1A	
16	18	27	1B	
17	19	28	1C	
18	20	29	1D	
19	21	30	1E	
20	22	31	1F	
21	23	32	20	
22	24	33	21	
23	25	34	22	
24	26	35	23	
25	27	36	24	Optional
26	28	37	25	
27	29	38	26	
28	30	39	27	
29	31	40	28	
30	32	41	29	
31	33	42	2A	
32	34	43	2B	
33	35	44	2C	
34	36	45	2D	
35	37	46	2E	
36	38	47	2F	
37	39	48	30	

*All levels except trace have interrupt level status words, and all levels except internal and trace can be masked.

Table 4-1. Interrupt Levels (Cont.)

Interrupt Level*	Priority	Address		Availability
		Decimal	Hexadecimal	
38	40	49	31	Optional
39	41	50	32	
40	42	51	33	
41	43	52	34	
42	44	53	35	
43	46	54	36	
44	47	55	37	
45	48	56	38	
46	49	57	39	
47	50	58	3A	
48	51	59	3B	
49	52	60	3C	
50	53	61	3D	
51	54	62	3E	
52	55	63	3F	
53	56	64	40	
54	57	65	41	
55	58	66	42	
56	59	67	43	
57	60	68	44	
58	61	69	45	

*All levels except trace have interrupt level status words, and all levels except internal and trace can be masked.

When more than one interrupt request line is connected to an interrupt level, the individual request or requests causing the interrupt level to be activated must be identified by the program. This is accomplished in an interrupt routine by first executing an XIO instruction to sense the interrupt level status word to determine the I/O device or process that is requesting the interrupt, and then executing another XIO instruction to sense the device or process interrupt status word to determine the condition responsible for the request.

The first request to activate an interrupt level prevents succeeding requests on that level and all lower priority levels from interrupting until a Branch Out of Interrupts instruction is executed to reset the active level. Although interrupt requests on the same or lower levels cannot interrupt the servicing routine, those requests that occur on the level being serviced can be interrogated and serviced by the program by again sensing the interrupt level status word before the Branch Out of Interrupts instruction is executed.

4.12 INTERNAL INTERRUPT LEVEL

The internal interrupt level is a standard feature of the computer. It has the highest priority of all interrupt levels and is activated by any of the following processor conditions:

a. Detection of a marginal power condition by the marginal power detection feature (paragraph 2.24).

b. Application of power with the automatic restart feature present (paragraph 2.24).

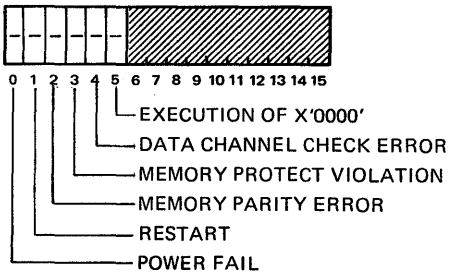
c. Detection of a memory parity error (even parity condition) when a word is fetched from memory.

d. An attempt to write into a protected memory location.

e. The detection of a direct memory data channel check error during a chaining operation (paragraph 4.9).

f. Execution of command X'0000'.

The internal interrupt level cannot be masked, but the execution of an XIO, BSI, or BOSC skip or jump instruction prevents an internal interrupt for one instruction. The interrupt level status word for this level is reset when it is sensed to identify the interrupting condition. The format of the status word is as follows:



4.13 TRACE INTERRUPT LEVEL

The trace interrupt level is a standard feature of the computer. It has the lowest priority of all interrupt levels. It occurs after every instruction except an XIO, BSI, or BOSC skip or jump instruction if the TRACE switch on the programmer's console is set to the lower (on) position. This interrupt has no interrupt level status word and cannot be masked.

4.14 EXTERNAL INTERRUPT LEVELS

External interrupt levels 0 through 5 are part of the standard interrupt feature of the computer. Levels 6 through 58 are optional. All external interrupt levels are unassigned and can be used for any purpose. Each level may have its own interrupt level status word, and all levels can be masked or unmasked under program control. Additionally, each external interrupt level can be activated by the program.

4.15 INTERRUPT CONTROL

The operation of the interrupt system is primarily controlled by the Execute I/O and Branch Out of Interrupt instructions. The following control operations are performed by these instructions:

a. An XIO instruction can be used to mask or unmask the external interrupt levels (the internal and trace levels cannot be masked).

b. An XIO instruction can be used to initiate an interrupt on an external interrupt level from within the program.

c. After an interrupt level becomes active (excluding trace), an XIO instruction is used to sense the interrupt level status word to identify the interrupt source.

d. After the interrupt source is identified, an XIO instruction is used to sense the device or process interrupt status word to determine the interrupting condition.

e. After an interrupt has been serviced, a Branch Out of Interrupt instruction is used to reset the active interrupt level to permit the servicing of equal or lower priority interrupts.

4.16 INTERRUPT LEVEL MASKING

Each external interrupt level can be masked or unmasked by the program, and all levels are automatically masked when the RESET switch on the programmer's console is pressed. When a level is masked, it is prevented from being activated and is effectively removed from the priority string until it is unmasked. One XIO instruction can mask and unmask 16 interrupt levels at a time; four instructions are required for all 59 levels. The IOCC format for masking and unmasking external interrupt levels is described in paragraph 4.3.

4.17 PROGRAM-INITIATED INTERRUPTS

External interrupt levels can be activated from within a program by the execution of an XIO control instruction. Only those levels that are unmasked at the time the forced interrupt BSI instruction occurs can be triggered. Only one interrupt level can be triggered by one XIO instruction. The IOCC format for program-initiated interrupts is described in paragraph 4.3.

4.18 STATUS WORDS

The interrupt system of the computer uses three types of status words: device status words (DSW's), process interrupt status words (PISW's), and interrupt level status words (ILSW's). The device and process interrupt status words are essentially identical except in the way they originate. Device status words originate from status indicators in an I/O device or system feature, while process interrupt status words originate from status indicators within a controlled process. However, both types of status words provide indications to the program about the operational status of the device or process. Those status indicators in a device or process that are connected to interrupt levels initiate interrupt requests on their levels when they are turned on.

The XIO sense device instruction is used to read a device status word or process interrupt status word into the accumulator where the individual status bits can then be tested. When the XIO instruction is used to sense a device status word, bit position 15 of the IOCC specifies whether or not the status indicators in the device are reset when the status word is read into the accumulator. A 1-bit resets the indicators. When the XIO instruction is used to sense a process interrupt status word, the status indicators in the process are unconditionally reset as the status word is read into the accumulator regardless of the status of bit 15 of the IOCC.

Each interrupt level (except the trace level) may have a 16-bit interrupt level status word associated with it. Each bit position of the interrupt level status word can be turned on by the interrupt status indicators of a device status word or a process interrupt status word as shown in figure 4-1. When any of the bits in the interrupt level status word are turned on, an interrupt is requested on that level. After the interrupt request is recognized, an XIO Sense Interrupt Level instruction can be used to read the interrupt level status word into the accumulator where it can then be tested to determine the source device or process. The XIO Sense Interrupt Level instruction does not specify a particular interrupt level status word since, when this instruction is executed, the status word for the highest priority interrupt level requesting service is automatically read into the accumulator. If an interrupt level status word has only one device or process interrupt status word assigned to it, the interrupt level status word need not be sensed to identify the interrupt source since only one source is possible.

4.19 INTERRUPT OPERATION

An interrupt request can cause an interrupt only if it has been polled and acknowledged. In general, polling occurs at the beginning of a machine cycle, and acknowledgment occurs at the end of an instruction. (If the computer is in idle mode, a pseudo end-of-instruction is generated at the end of each machine cycle.)

Polling is inhibited during the first two cycles of the forced BSI instruction caused by an interrupt.

Interrupt acknowledgement is inhibited when the computer is executing an XIO, BSI, or BOSC (skip or jump) instruction.

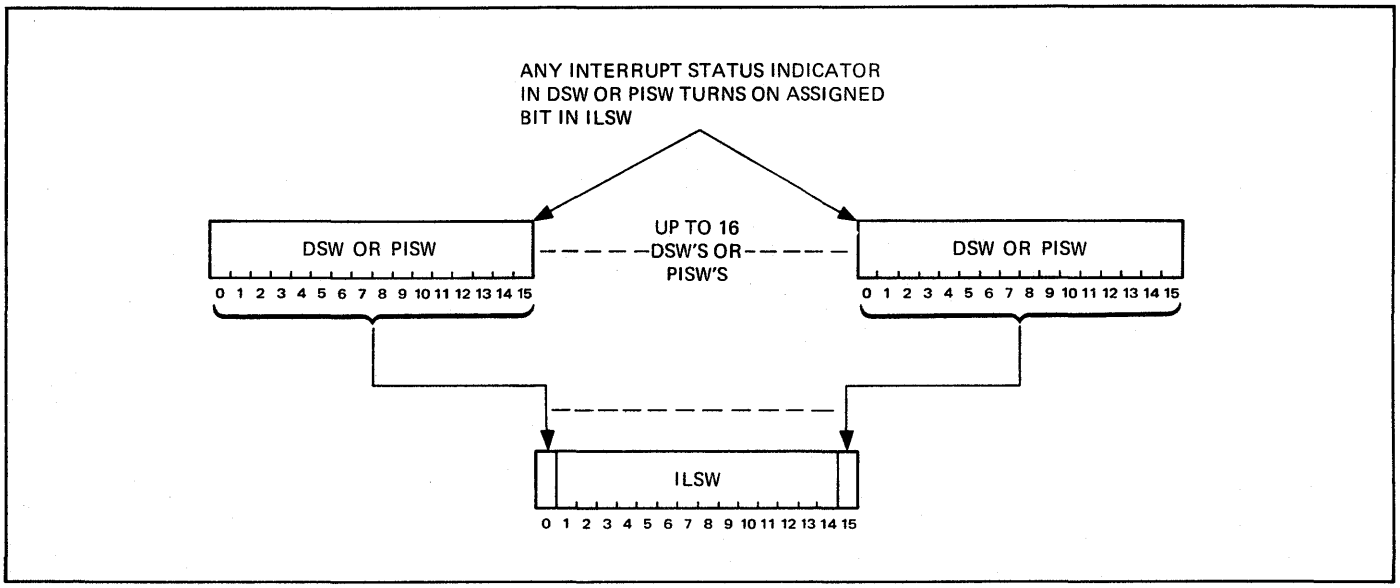


Figure 4-1. Status Word Relationships

4.20 INTERRUPT SERVICING ROUTINE PROGRAMMING

The way in which interrupts are assigned to the interrupt levels largely determines the method of programming used to service the interrupts. For example:

a. If several interrupt requests are assigned to the same level, the interrupt level status word is normally sensed first to determine the requesting source, and then the interrupt status word associated with the requesting source is sensed after that.

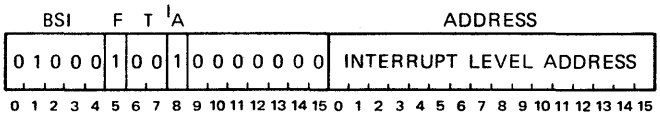
b. If only one interrupt request is assigned to an interrupt level, the status word is normally sensed to determine the interrupting condition.

When an interrupt request is initiated, it is recognized at the end of the instruction currently being executed unless the level is masked, the request is on the same or a lower priority level than an interrupt currently being serviced, or an interrupt has just occurred and polling has not yet been resumed (paragraph 4.19). Interrupt requests that occur on a masked level are retained by the source, not the interrupt system, for recognition when the level again becomes unmasked.

The operation of the interrupt system when the interrupt is recognized is as follows (see figure 4-2).

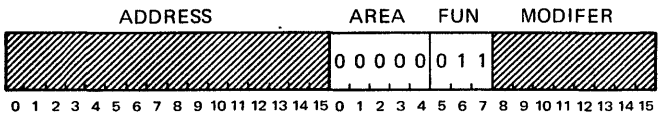
1. The interrupt request is polled into the interrupt request register.

2. The interrupt system forces the execution of an indirect BSI instruction that contains the address of the requesting interrupt level (table 4-1) in its address field. The format of the forced BSI instruction is as follows:



3. The forced BSI instruction stores the contents of the I-register (return link address) at the effective address of the BSI instruction. The effective address is the address stored at the memory location assigned to the interrupt level. The program then branches to the interrupt identification routine at the effective address plus one.

4. In the interrupt identification routine an XIO sense interrupt level instruction is executed to read the interrupt level status word into the accumulator. The IOCC for this instruction specifies only the sense interrupt level function; no other portions of the IOCC are used. The following format is used for this IOCC:



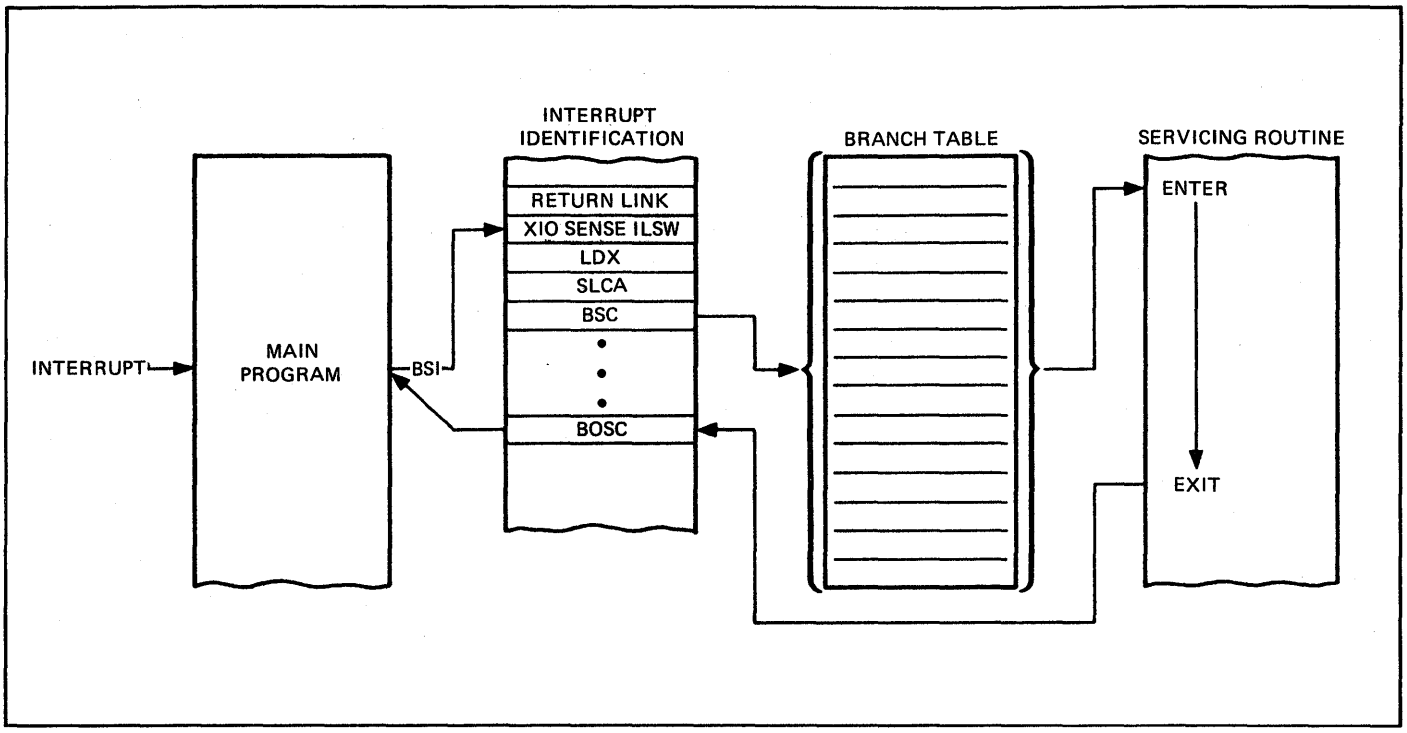


Figure 4-2. Interrupt Programming

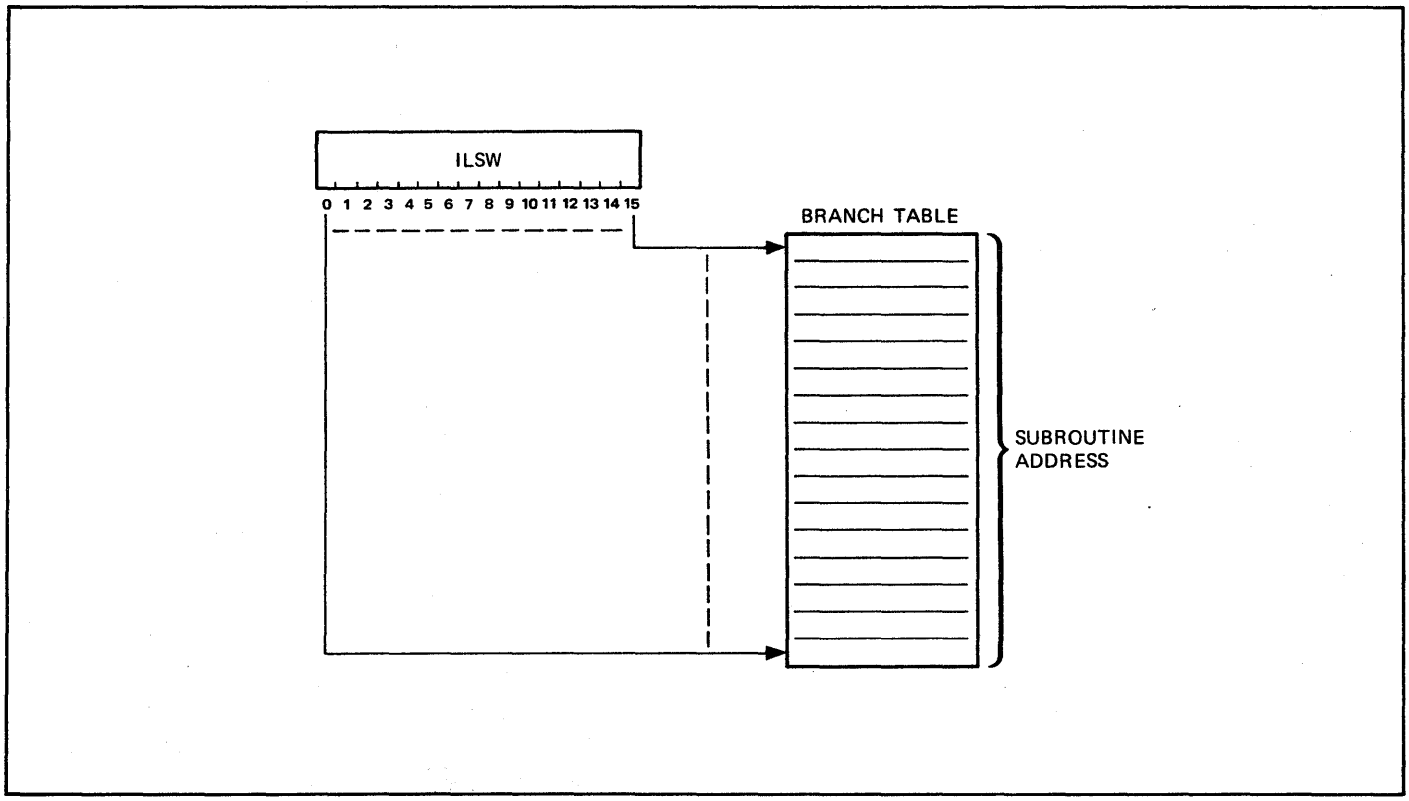


Figure 4-3. Interrupt Branch Table

5. A Load Index Register (LDX) instruction is then executed to load an index register with the number of interrupts assigned to the interrupt level.

6. A Shift Left and Count A (SLCA) instruction is then executed to determine which bit in the status word is causing the interrupt. The count in the index register when the shift stops corresponds to the first 1-bit in the status word.

7. An indirect and indexed BSC (branch on Carry) instruction is then executed. This instruction picks up an address from a branch table which contains the addresses of subroutines for servicing the interrupts assigned to the level. The highest location in the table corresponds to the interrupt represented by bit 0 of the interrupt level status word as shown in figure 4-3. The count in the index register determines which address in the branch table is to be used as the effective address of the BSC instruction. The execution of the BSC instruction causes the program to branch to the selected servicing routine.

8. The last instruction in the servicing routine is a Branch Out of Interrupt (BOSC) instruction that returns control to the main program through the return link address stored by the forced BSI instruction. The BOSC instruction also resets the interrupt level so that interrupt requests on the same or lower priority levels can be recognized.

SECTION V
PROGRAMMER'S CONSOLE

5.1 GENERAL

The programmer's console (figure 5-1) contains switches and indicators for controlling and monitoring all primary operations of the computer. Among the operational safety features included in the console design are two key-operated switches: CONSOLE ENABLE and WSPB (Write Storage Protect Bit). These switches provide an extra measure of safety in ensuring that the operational security of the system is not violated when the system is online and operating.

5.2 CONSOLE CONTROLS

5.3 DATA SWITCHES (0 THROUGH 15)

The 16 data switches are used to enter data or instructions into the computer either manually or under program control. These are two-position

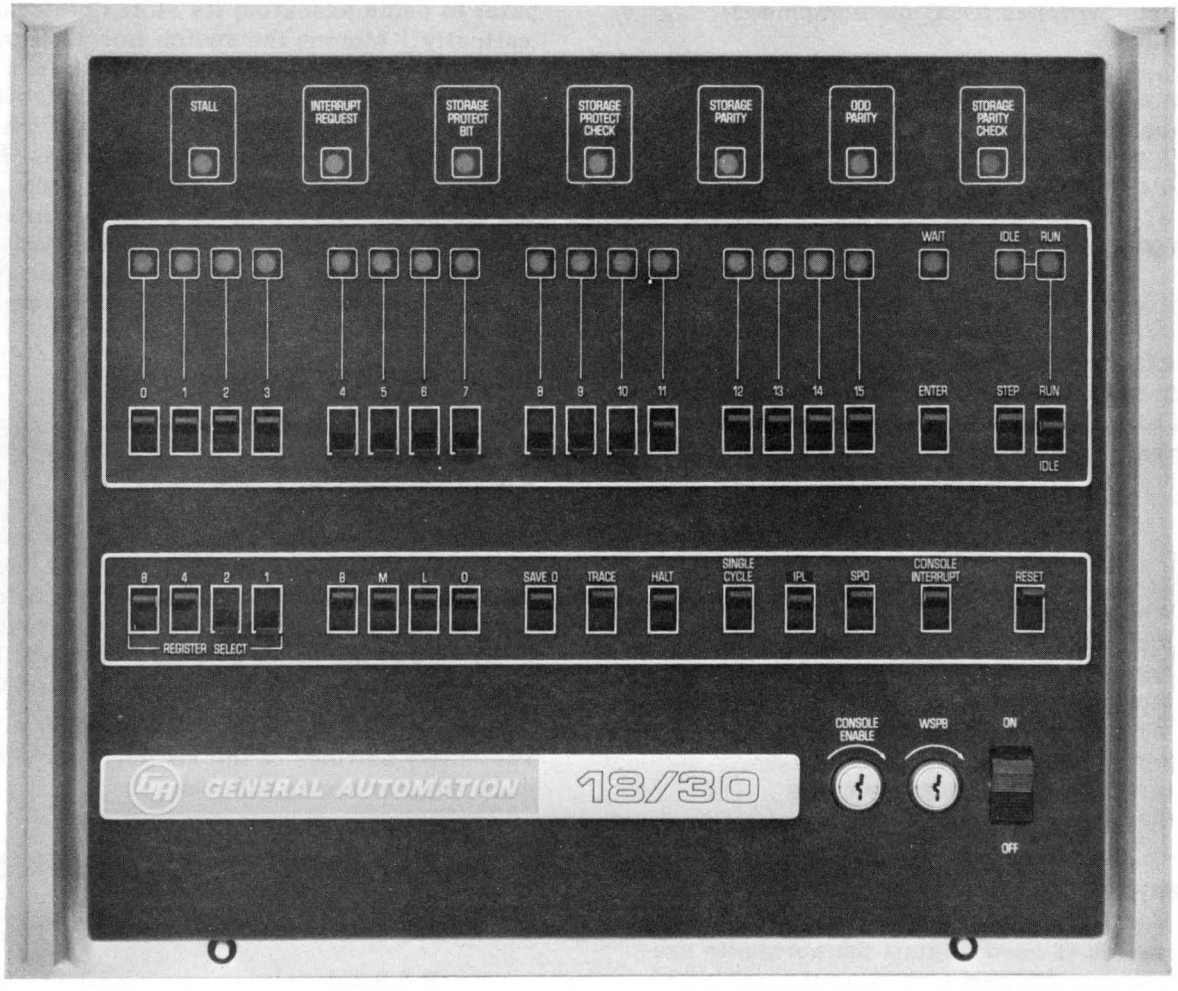


Figure 5-1. Programmer's Console

toggle switches with the upper position representing a binary 0 and the lower position, a binary 1. The switches are arranged in groups of four for ease of working with hexadecimal notation and are numbered to correspond with the bit positions of the computer registers and memory locations.

Information is entered into the computer from the data switches in any of three ways:

a. The information set into the data switches can be entered into 1 of 18 registers by momentarily pressing the ENTER switch. The register into which the data is entered is specified by the REGISTER SELECT switches or the B or O switch. Information can be entered in this way only when the CONSOLE ENABLE switch is unlocked and the computer is in idle mode.

b. The information set into the data switches can be entered into the accumulator under program control by executing an XIO instruction with a Sense Console Data Switches IOCC (paragraph 4.3).

c. The information set into the data switches can be entered into a specified memory location under program control by executing an XIO instruction with a Read Console Data Switches IOCC (paragraph 4.3).

5.4 ENTER SWITCH

The ENTER switch is used to enter information from the data switches into the register specified by the B, O, or REGISTER SELECT switches (paragraph 5.3). The switch is spring-loaded to return to the upper position and is operable only when the computer is in idle mode and the CONSOLE ENABLE switch is unlocked.

NOTE

When the ENTER switch is used to load a register other than the B-register, the information copied into the specified register is simultaneously copied into the B-register. The M- and L-registers cannot be loaded by using the ENTER switch.

5.5 STEP SWITCH

The STEP switch is used to start the automatic execution of a stored program or to step through a program one instruction or one memory cycle at a time. It is spring-loaded to return to the upper position, and is operable only when the CONSOLE ENABLE switch is unlocked.

Pressing the STEP switch with the RUN-IDLE switch set to IDLE and the SINGLE CYCLE switch set to the upper (off) position causes the computer to execute one complete instruction and then return to idle mode. Pressing the STEP switch with the RUN-IDLE switch set to IDLE and the SINGLE CYCLE switch set to the lower (on) position causes the computer to execute one machine cycle of an instruction and then stop. The HALT switch must be set to the lower position for single cycle operation.

The STEP switch is extremely useful in evaluating program or processor operation by stepping through a program one instruction or one cycle at a time while observing console indications.

5.6 RUN-IDLE SWITCH

The RUN-IDLE switch controls the operating mode of the computer. It is a two-position toggle switch, operable only when the CONSOLE ENABLE switch is unlocked. Setting the switch to the RUN position and then pressing the STEP switch causes the computer to begin executing its stored program automatically. Moving the switch from RUN to IDLE while the computer is executing a program in run mode causes the computer to cease operation and return to idle mode either at the end of the current instruction or the current memory cycle. If the SINGLE CYCLE switch is in the upper (off) position the computer stops at the end of the current instruction; if the SINGLE CYCLE switch is in the lower (on) position (and the HALT switch is on) the computer stops at the end of the current memory cycle.

5.7 REGISTER SELECT SWITCHES

The four REGISTER SELECT switches (8, 4, 2, 1) are used to select any one of 16 operational registers within the computer. The contents of the selected register are displayed by the 16 register display lamps above the data switches and can be altered using the data switches and the ENTER switch. The REGISTER SELECT switches are two-position toggle switches with the upper position representing a binary 0 and the lower position, a binary 1. They are operable only when the B, M, L, and O switches are in the upper (off) position.

The binary settings of the switches for selecting each register are as follows:

Switch Setting 8 4 2 1	Register Selected
0 0 0 0	I-register
0 0 0 1	XR1
0 0 1 0	XR2
0 0 1 1	XR3

Switch Setting 8 4 2 1	Register Selected
0 1 0 0	A-register
0 1 0 1	Q-register
0 1 1 0	CAR1
0 1 1 1	SCR1
1 0 0 0	CAR2
1 0 0 1	SCR2
1 0 1 0	CAR3
1 0 1 1	SCR3
1 1 0 0	CAR4
1 1 0 1	SCR4
1 1 1 0	CAR5
1 1 1 1	SCR5

5.8 B, M, L, O SWITCHES

The four switches labeled B, M, L, and O are used to display the contents of the buffer (B) register, memory data (M) register, memory address (L) register, or operation (O) register in the register display lamps above the data switches. When the operation register or the buffer register is selected for display, its contents can be changed using the data switches and the ENTER switch.

These four switches are two-position toggle switches with the upper position off and the lower position on. They are interlocked in the order O, B, M, L; that is, switches B, M, and L are inoperable when switch O is on; switches M and L are inoperable when switch B is on, and switch L is inoperative when switch M is on. All four of these switches must be off for the REGISTER SELECT switches to be operable.

5.9 SAVE O SWITCH

The SAVE O switch is used to prevent the contents of the operation register from being changed during the single step execution of an instruction in idle mode. It is a two-position toggle switch with the upper position off and the lower position on. It is operable only when the computer is in idle mode, the CONSOLE ENABLE switch is unlocked, and the HALT switch is in the lower (on) position.

The SAVE O switch is used primarily to display the contents of sequential locations in memory from the console or to store information into sequential locations in memory from the console. To gain access to a block of locations in memory requires that the proper instruction (Load Accumulator or Store Accumulator) be placed in the operation register while the instruction register is continuously incremented. The SAVE O switch prevents the instruction in the operation register from being lost each time the instruction register is advanced.

NOTE

The operation register is modified during the execution of shift, multiply, divide, and MDX modify memory instructions regardless of the setting of the SAVE O switch.

The procedures for storing information into sequential locations in memory and for displaying the contents of sequential locations in memory from the programmer's console are described in paragraphs 5.39 and 5.40 respectively.

5.10 TRACE SWITCH

The TRACE switch is a two-position toggle switch operable only when the CONSOLE ENABLE switch is unlocked. When the switch is set to the lower position (on), a trace interrupt occurs after the execution of each instruction except an XIO, BSI, or BOSC (skip or jump) instruction. Since the trace interrupt has the lowest priority of all interrupts, it cannot be accepted while other interrupts are being serviced. Once accepted, it causes the processor to execute a forced BSI instruction which transfers program control to the routine whose starting address is stored in memory location X'0009'. No other trace interrupts can occur until the trace interrupt has been cleared by a BOSC instruction.

5.11 HALT SWITCH

The HALT switch is used to inhibit the operation of the interval timers, direct memory data channels, and interrupts. If the switch is set to the lower position (on) while an operation is in progress, the operation continues to completion, but further operations are then inhibited.

This two-position toggle switch is operable only when the CONSOLE ENABLE switch is unlocked and the RUN-IDLE switch is set to IDLE. The HALT switch must be in the lower position for the SINGLE CYCLE, RESET, and SAVE O switches to be operative.

5.12 SINGLE CYCLE SWITCH

The SINGLE CYCLE switch is used during single step operations to stop instruction execution after one memory cycle. If the switch is in the upper (off) position when the STEP switch is pressed while the computer is in idle mode, the computer executes one complete instruction and then stops. If the switch is in the lower (on) position when the step operation is performed, the computer executes one memory cycle of the instruction and then stops. The SINGLE CYCLE switch also determines the point at which the computer stops when the RUN-IDLE switch is moved from RUN to IDLE while the computer is executing a program. Placing the computer in idle mode with the SINGLE CYCLE switch off stops operation at the end of the current instruction. Entering idle mode with the SINGLE CYCLE switch on stops operation at the end of the current memory cycle.

The SINGLE CYCLE switch is operable only when the CONSOLE ENABLE switch is unlocked, the HALT switch is in the lower (on) position, and the RUN-IDLE switch is set to IDLE.

5.13 IPL (INITIAL PROGRAM LOAD) SWITCH

The use of the IPL (initial program load) switch is described in the manuals for I/O devices with IPL ability.

5.14 SPO (STORAGE PROTECT OVERRIDE) SWITCH

The SPO (Storage Protect Override) switch is used to enable writing into protected memory locations. When the switch is in the lower (on) position and the WSPB (Write Storage Protect Bits) switch is unlocked, the storage protection circuits are deactivated.

5.15 CONSOLE INTERRUPT SWITCH

The CONSOLE INTERRUPT switch permits program operation to be interrupted from the programmer's console. Pressing the switch causes an interrupt request on the level to which the console interrupt is assigned. (The console interrupt is not preassigned to any particular level and can be assigned to fit the application.)

5.16 RESET SWITCH

The RESET switch is used to reset the operating conditions of the computer and all I/O devices. It is spring-loaded to return to the upper position and is operable only when the CONSOLE ENABLE switch is unlocked and the HALT and RUN-IDLE switches are in the lower position. Pressing the switch resets the system and inhibits all external interrupt levels by setting all bits of the interrupt mask register to 1.

5.17 WSPB (WRITE STORAGE PROTECT BIT) SWITCH

The WSPB (Write Storage Protect Bit) switch is used to enable the writing or clearing of the memory storage protection bits. It is a key-operation, two-position switch. The switch must be unlocked (clockwise) to permit the storage protection bits to be changed. The key can be removed from the switch when it is either locked or unlocked.

5.18 CONSOLE ENABLE SWITCH

The CONSOLE ENABLE switch is used to enable or disable all controls (except the data switches and POWER, CONSOLE INTERRUPT, and REGISTER SELECT switches) on the programmer's console. It is a two-position, key-operated switch. The switch must be unlocked (clockwise) to enable the console controls. The key can be removed from the switch with the switch either locked or unlocked.

5.19 POWER SWITCH

The POWER switch controls the application of ac power to the computer. It is a two-position toggle switch.

5.20 CONSOLE DISPLAYS

5.21 STALL INDICATOR

The STALL indicator lights to indicate that the stall alarm timer has timed out and the stall alarm has been activated. The stall alarm can be reset under program control or by unlocking the CONSOLE ENABLE switch.

5.22 INTERRUPT REQUEST INDICATOR

The INTERRUPT REQUEST indicator lights to indicate that an interrupt is being requested or is active. The trace interrupt is the only one that does not turn on the indicator.

5.23 STORAGE PROTECT BIT INDICATOR

The STORAGE PROTECT BIT indicator lights whenever a word is read from a protected memory location.

5.24 STORAGE PROTECT CHECK INDICATOR

The STORAGE PROTECT CHECK indicator lights whenever an attempt is made to write into a protected memory location with the storage protection circuits active. A memory protection violation also turns on bit 3 of the internal interrupt level status word to activate the internal interrupt level.

5.25 STORAGE PARITY INDICATOR

The STORAGE PARITY indicator lights whenever the parity bit in a word read from memory is set to 1.

5.26 ODD PARITY INDICATOR

The ODD PARITY indicator lights whenever the combination of 1-bits shown on the register display indicators, the STORAGE PROTECT BIT indicator, and the STORAGE PARITY indicator is odd. The indication given by this indicator does not necessarily represent storage parity. This is true when the register display indicators are displaying the contents of a register and not the contents of the memory data register.

5.27 STORAGE PARITY CHECK INDICATOR

The STORAGE PARITY CHECK indicator lights whenever a parity error (even number of 1-bits) is detected in a word read out of memory. The parity error also turns on bit 2 of the internal interrupt level status word to activate the internal interrupt level.

5.28 REGISTER DISPLAY INDICATORS (0 THROUGH 15)

The 16 register display indicators display the contents of the memory data bus. They are located directly above the data switches and, like the data switches, are arranged in groups of four for ease of working with hexadecimal notation. Each indicator is lit when the corresponding bit position of the memory data bus contains a 1-bit.

When the computer is operating in idle mode, the display indicators display the contents of the register specified by the settings of the REGISTER SELECT switches or the B, M, L, and O switches (since, in idle mode, the selected register is connected to the memory data bus).

5.29 WAIT INDICATOR

The WAIT indicator lights when the RUN-IDLE switch is set to RUN, but the computer is in idle mode. The indicator remains lit until the STEP switch is pressed to place the computer in run mode, or until an interrupt causes the computer to return to run mode.

5.30 IDLE INDICATOR

The IDLE indicator lights when the RUN-IDLE switch is set to IDLE and goes off when the switch is set to RUN.

5.31 RUN INDICATOR

The RUN indicator lights when the RUN-IDLE switch is set to RUN and the computer is operating in run mode. The indicator goes off if the computer enters idle mode while the RUN-IDLE switch is still in the RUN position.

5.32 OPERATING PROCEDURES

5.33 APPLYING POWER (WITHOUT AUTOMATIC RESTART)

The following procedure is used to apply power to a computer that is not equipped with the power failure detection and automatic restart option:

1. Verify that the computer power cable is connected to 115V, 60-Hz electrical service.

2. Unlock the CONSOLE ENABLE switch.

3. Set the RUN-IDLE switch to IDLE.

4. Set the HALT switch to the lower (on) position.

5. Set all other console switches to the upper position.

6. Set the POWER switch on.

7. Momentarily press the RESET switch to initialize system operating conditions.

With the completion of step 8 the computer is in idle mode with all console switches operable. If a program is already stored in memory, it can be executed using the procedure given in paragraph 5.41. If there is no program in memory, information can be entered into the computer by

using the procedures given in paragraph 5.36 or 5.39.

5.34 APPLYING POWER (WITH AUTOMATIC RESTART)

When power is applied to a computer equipped with the power failure detection and automatic restart option, bit 1 of the internal interrupt level status word is automatically turned on to activate the internal level and force the execution of an indirect BSI instruction with an address of X'0008'. The BSI instruction picks up the address stored at this location (normally the address of a power startup routine) and then transfers program control to the instruction at that address. If the startup routine is not present or if there is some doubt that it is present, use the procedure in paragraph 5.33 to apply power; otherwise, proceed as follows:

1. Verify that the computer power cable is connected to 115V, 60-Hz electrical service.

2. Set the POWER switch on. The computer automatically generates the restart interrupt and begins executing the startup routine.

5.35 REMOVING POWER

If the computer contains the marginal power detection and automatic restart option, power is removed simply by setting the POWER switch off. If the option is not present, remove power as follows:

1. Unlock the CONSOLE ENABLE switch.

2. Set the RUN-IDLE switch to IDLE. The computer enters the idle mode at the end of the current instruction.

3. Set the HALT switch to the lower (on) position.

4. Press the STEP switch.

5. Set the POWER switch off.

5.36 INITIAL LOADING

If the computer is not equipped with an I/O device that has the initial load feature, the procedure given in paragraph 5.39 must be used for initial program loading. If the computer is equipped with an I/O device that has the initial load feature, initial loading is performed by following the instructions in the user's manual for the I/O device.

5.37 ALTERING REGISTER CONTENTS

The following procedure is used to alter the contents of any of the selectable registers from the programmer's console:

1. Unlock the CONSOLE ENABLE switch.

2. Place the computer in idle mode by setting the RUN-IDLE switch to IDLE.

3. Set the REGISTER SELECT or B and O switches to select the register to be altered.

4. Set the data to be entered into the register into data switches 0 through 15 (upper position represents 0, lower position represents 1).

5. Momentarily press the ENTER switch.

NOTE

When the ENTER switch is pressed, the data is not only entered into the selected register, but is also automatically entered into the B-register. If the B-register contains data that should not be altered, the data must be reloaded into the B-register after the selected register contents are altered.

6. Verify that the contents of the register have been changed by observing the register display indicators.

7. To restart the program, set the RUN-IDLE switch to RUN and press the STEP switch.

5.38 DISPLAYING REGISTER CONTENTS

The contents of any of the selectable registers can be displayed by performing steps 1 through 3 of paragraph 5.37. To restart the program after displaying register contents, perform step 7 of paragraph 5.37.

5.39 ALTERING MEMORY CONTENTS

The following procedure is used to alter the contents of memory locations from the programmer's console:

1. Unlock the CONSOLE ENABLE switch.

2. Place the computer in idle mode by setting the RUN-IDLE switch to IDLE.

3. Set the HALT switch to the lower (on) position.

4. Verify that the SAVE O switch is off (upper position).

5. Set the O switch to the lower position (on).

6. Set X'D000' (Store Accumulator instruction) into data switches 0 through 15.

7. Momentarily press the ENTER switch to enter the instruction into the operation register.

8. Set the SAVE O switch to the lower (on) position.

9. Verify that B, M, L, and O switches are in the upper (off) position, and then set the REGISTER SELECT switches to X'0' to select the instruction counter.

10. Set data switches 0 through 15 to one less than the address of the memory location to be altered.

11. Momentarily press the ENTER switch to load the address into the instruction counter.

12. Set the REGISTER SELECT switches to X'4' to select the accumulator.

13. Set the data that is to be stored in the memory location into data switches 0 through 15.

14. Momentarily press the ENTER switch to load the data into the accumulator.

15. Momentarily press the STEP switch. The new information is now stored in the memory location whose address is currently in the instruction counter.

16. To continue storing information in consecutive memory locations, repeat steps 13 through 15.

5.40 DISPLAYING MEMORY CONTENTS

The following procedure is used to display the contents of memory locations from the programmer's console:

1. Unlock the CONSOLE ENABLE switch.

2. Place the computer in idle mode by setting the RUN-IDLE switch to IDLE.

3. Set the HALT switch to the lower (on) position.

4. Verify that the SAVE O switch is off (upper position).

5. Set the O switch to the lower (on) position.

6. Set X'C000' (Load Accumulator instruction) into data switches 0 through 15.

7. Momentarily press the ENTER switch to enter the instruction into the operation register.

8. Set the SAVE O switch to the lower (on) position.

9. Verify that the B, M, L, and O switches are in the upper (off) position, and then set the REGISTER SELECT switches to X'0' to select the instruction counter.

10. Set data switches 0 through 15 to one less than the address of the memory location to be displayed.

11. Momentarily press the ENTER switch to load the address into the instruction counter.

12. Set the REGISTER SELECT switches to X'4' to select the accumulator.

13. Momentarily press the STEP switch. Register display indicators 0 through 15 now display the contents of the memory location whose address is currently in the instruction counter.

14. To continue displaying the contents of consecutive memory locations, repeat step 13.

5.41 INITIATING PROGRAM EXECUTION

The following procedure is used to begin executing a program that is already stored in memory:

1. Unlock the CONSOLE ENABLE switch.

2. Place the computer in idle mode by setting the RUN-IDLE switch to IDLE.

3. Set the HALT switch to the lower (on) position.

4. Verify that the SAVE O switch is in the upper (off) position.

5. Set the O switch to the lower (on) position.

6. Verify that the B, M, L, and O switches are in the upper (off) position, and then set the REGISTER SELECT switches to X'0' to select the instruction counter.

7. Set data switches to the address of the memory location where program execution is to begin.

8. Momentarily press the ENTER switch to load the address into the instruction counter.

9. Set the RUN-IDLE switch to RUN.

10. Momentarily press the STEP switch. The computer now begins executing the program with the instruction stored in the starting memory location.

11. Lock the CONSOLE ENABLE switch.

5.42 CLEARING MEMORY PROTECT BIT

The following procedure may be used to clear the memory protect bit.

1. Load the following program

```
L  2D40
   0000
   5111
   70FC
   70FB
```

2. Set the WSPB switch ON.

3. Insert L in the I register.

4. Set RUN-IDLE switch to RUN.

5. Press the STEP switch.

The computer will execute the program, clearing all of the memory protect bits.

APPENDIX A
CONVERSION TABLES

This appendix contains the following reference tables:

HEXADECIMAL ARITHMETIC

ADDITION TABLE

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10
2	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11
3	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12
4	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13
5	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14
6	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15
7	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16
8	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17
9	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18
A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19
B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A
C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B
D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C
E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E

MULTIPLICATION TABLE

1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2	04	06	08	0A	0C	0E	10	12	14	16	18	1A	1C	1E
3	06	09	0C	0F	12	15	18	1B	1E	21	24	27	2A	2D
4	08	0C	10	14	18	1C	20	24	28	2C	30	34	38	3C
5	0A	0F	14	19	1E	23	28	2D	32	37	3C	41	46	4B
6	0C	12	18	1E	24	2A	30	36	3C	42	48	4E	54	5A
7	0E	15	1C	23	2A	31	38	3F	46	4D	54	5B	62	69
8	10	18	20	28	30	38	40	48	50	58	60	68	70	78
9	12	1B	24	2D	36	3F	48	51	5A	63	6C	75	7E	87
A	14	1E	28	32	3C	46	50	5A	64	6E	78	82	8C	96
B	16	21	2C	37	42	4D	58	63	6E	79	84	8F	9A	A5
C	18	24	30	3C	48	54	60	6C	78	84	90	9C	A8	B4
D	1A	27	34	41	4E	5B	68	75	82	8F	9C	A9	B6	C3
E	1C	2A	38	46	54	62	70	7E	8C	9A	A8	B6	C4	D2
F	1E	2B	3C	4B	5A	69	78	87	96	A5	B4	C3	D2	E1

TABLE OF POWERS OF SIXTEEN$_{10}$

16^n	n	16^{-n}
1	0	0.10000 00000 00000 00000 × 10
16	1	0.62500 00000 00000 00000 × 10^{-1}
256	2	0.39062 50000 00000 00000 × 10^{-2}
4 096	3	0.24414 06250 00000 00000 × 10^{-3}
65 536	4	0.15258 78906 25000 00000 × 10^{-4}
1 048 576	5	0.95367 43164 06250 00000 × 10^{-6}
16 777 216	6	0.59604 64477 53906 25000 × 10^{-7}
268 435 456	7	0.37252 90298 46191 40625 × 10^{-8}
4 294 967 296	8	0.23283 06436 53869 62891 × 10^{-9}
68 719 476 736	9	0.14551 91522 83668 51807 × 10^{-10}
1 099 511 627 776	10	0.90949 47017 72928 23792 × 10^{-12}
17 592 186 044 416	11	0.56843 41886 08080 14870 × 10^{-13}
281 474 976 710 656	12	0.35527 13678 80050 09294 × 10^{-14}
4 503 599 627 370 496	13	0.22204 46049 25031 30808 × 10^{-15}
72 057 594 037 927 936	14	0.13877 78780 78144 56755 × 10^{-16}
1 152 921 504 606 846 976	15	0.86736 17379 88403 54721 × 10^{-18}

TABLE OF POWERS OF 10_{16}

10^n	n	10^{-n}
1	0	1.0000 0000 0000 0000
A	1	0.1999 9999 9999 999A
64	2	0.28F5 C28F 5C28 F5C3 × 16^{-1}
3E8	3	0.4189 374B C6A7 EF9E × 16^{-2}
2710	4	0.68DB 8BAC 710C B296 × 16^{-3}
1 86A0	5	0.A7C5 AC47 1B47 8423 × 16^{-4}
F 4240	6	0.10C6 F7A0 B5ED 8D37 × 16^{-4}
98 9680	7	0.1AD7 F29A BCAF 4858 × 16^{-5}
5F5 E100	8	0.2AF3 1DC4 6118 73BF × 16^{-6}
3B9A CA00	9	0.44B8 2FA0 9B5A 52CC × 16^{-7}
2 540B E400	10	0.6DF3 7F67 5EF6 EADF × 16^{-8}
17 4876 E800	11	0.AFEB FF0B CB24 AAFF × 16^{-9}
E8 D4A5 1000	12	0.1197 9981 2DEA 1119 × 16^{-9}
918 4E72 A000	13	0.1C25 C268 4976 81C2 × 16^{-10}
5AF3 107A 4000	14	0.2D09 370D 4257 3604 × 16^{-11}
3 8D7E A4C6 8000	15	0.480E BE7B 9D58 566D × 16^{-12}
23 8652 6FC1 0000	16	0.734A CA5F 6226 F0AE × 16^{-13}
163 4578 5D8A 0000	17	0.B877 AA32 36A4 B449 × 16^{-14}
DE0 B6B3 A764 0000	18	0.1272 5DD1 D243 ABA1 × 16^{-14}
8AC7 2304 89E8 0000	19	0.1D83 C94F B6D2 AC35 × 16^{-15}

HEXADECIMAL-DECIMAL INTEGER CONVERSION

The table below provides for direct conversions between hexadecimal integers in the range 0—FFF and decimal integers in the range 0—4095. For conversion of larger integers, the table values may be added to the following figures:

Hexadecimal	Decimal	Hexadecimal	Decimal
01 000	4 096	20 000	131 072
02 000	8 192	30 000	196 608
03 000	12 288	40 000	262 144
04 000	16 384	50 000	327 680
05 000	20 480	60 000	393 216
06 000	24 576	70 000	458 752
07 000	28 672	80 000	524 288
08 000	32 768	90 000	589 824
09 000	36 864	A0 000	655 360
0A 000	40 960	B0 000	720 896
0B 000	45 056	C0 000	786 432
0C 000	49 152	D0 000	851 968
0D 000	53 248	E0 000	917 504
0E 000	57 344	F0 000	983 040
0F 000	61 440	100 000	1 048 576
10 000	65 536	200 000	2 097 152
11 000	69 632	300 000	3 145 728
12 000	73 728	400 000	4 194 304
13 000	77 824	500 000	5 242 880
14 000	81 920	600 000	6 291 456
15 000	86 016	700 000	7 340 032
16 000	90 112	800 000	8 388 608
17 000	94 208	900 000	9 437 184
18 000	98 304	A00 000	10 485 760
19 000	102 400	B00 000	11 534 336
1A 000	106 496	C00 000	12 582 912
1B 000	110 592	D00 000	13 631 488
1C 000	114 688	E00 000	14 680 064
1D 000	118 784	F00 000	15 728 640
1E 000	122 880	1 000 000	16 777 216
1F 000	126 976	2 000 000	33 554 432

Hexadecimal fractions may be converted to decimal fractions as follows:

1. Express the hexadecimal fraction as an integer times 16^{-n}, where n is the number of significant hexadecimal places to the right of the hexadecimal point.

$$0.\,CA9BF3_{16} = CA9\,BF3_{16} \times 16^{-6}$$

2. Find the decimal equivalent of the hexadecimal integer

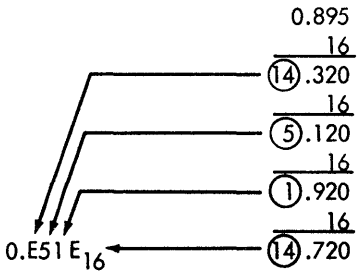
$$CA9\,BF3_{16} = 13\,278\,195_{10}$$

3. Multiply the decimal equivalent by 16^{-n}

$$\begin{array}{r} 13\,278\,195 \\ \times\ 596\,046\,448 \times 10^{-16} \\ \hline 0.791\,442\,096_{10} \end{array}$$

Decimal fractions may be converted to hexadecimal fractions by successively multiplying the decimal fraction by 16_{10}. After each multiplication, the integer portion is removed to form a hexadecimal fraction by building to the right of the hexadecimal point. However, since decimal arithmetic is used in this conversion, the integer portion of each product must be converted to hexadecimal numbers.

Example: Convert 0.895_{10} to its hexadecimal equivalent



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000	0000	0001	0002	0003	0004	0005	0006	0007	0008	0009	0010	0011	0012	0013	0014	0015
010	0016	0017	0018	0019	0020	0021	0022	0023	0024	0025	0026	0027	0028	0029	0030	0031
020	0032	0033	0034	0035	0036	0037	0038	0039	0040	0041	0042	0043	0044	0045	0046	0047
030	0048	0049	0050	0051	0052	0053	0054	0055	0056	0057	0058	0059	0060	0061	0062	0063
040	0064	0065	0066	0067	0068	0069	0070	0071	0072	0073	0074	0075	0076	0077	0078	0079
050	0080	0081	0082	0083	0084	0085	0086	0087	0088	0089	0090	0091	0092	0093	0094	0095
060	0096	0097	0098	0099	0100	0101	0102	0103	0104	0105	0106	0107	0108	0109	0110	0111
070	0112	0113	0114	0115	0116	0117	0118	0119	0120	0121	0122	0123	0124	0125	0126	0127
080	0128	0129	0130	0131	0132	0133	0134	0135	0136	0137	0138	0139	0140	0141	0142	0143
090	0144	0145	0146	0147	0148	0149	0150	0151	0152	0153	0154	0155	0156	0157	0158	0159
0A0	0160	0161	0162	0163	0164	0165	0166	0167	0168	0169	0170	0171	0172	0173	0174	0175
0B0	0176	0177	0178	0179	0180	0181	0182	0183	0184	0185	0186	0187	0188	0189	0190	0191
0C0	0192	0193	0194	0195	0196	0197	0198	0199	0200	0201	0202	0203	0204	0205	0206	0207
0D0	0208	0209	0210	0211	0212	0213	0214	0215	0216	0217	0218	0219	0220	0221	0222	0223
0E0	0224	0225	0226	0227	0228	0229	0230	0231	0232	0233	0234	0235	0236	0237	0238	0239
0F0	0240	0241	0242	0243	0244	0245	0246	0247	0248	0249	0250	0251	0252	0253	0254	0255

HEXADECIMAL-DECIMAL INTEGER CONVERSION (Cont.)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
100	0256	0257	0258	0259	0260	0261	0262	0263	0264	0265	0266	0267	0268	0269	0270	0271
110	0272	0273	0274	0275	0276	0277	0278	0279	0280	0281	0282	0283	0284	0285	0286	0287
120	0288	0289	0290	0291	0292	0293	0294	0295	0296	0297	0298	0299	0300	0301	0302	0303
130	0304	0305	0306	0307	0308	0309	0310	0311	0312	0313	0314	0315	0316	0317	0318	0319
140	0320	0321	0322	0323	0324	0325	0326	0327	0328	0329	0330	0331	0332	0333	0334	0335
150	0336	0337	0338	0339	0340	0341	0342	0343	0344	0345	0346	0347	0348	0349	0350	0351
160	0352	0353	0354	0355	0356	0357	0358	0359	0360	0361	0362	0363	0364	0365	0366	0367
170	0368	0369	0370	0371	0372	0373	0374	0375	0376	0377	0378	0379	0380	0381	0382	0383
180	0384	0385	0386	0387	0388	0389	0390	0391	0392	0393	0394	0395	0396	0397	0398	0399
190	0400	0401	0402	0403	0404	0405	0406	0407	0408	0409	0410	0411	0412	0413	0414	0415
1A0	0416	0417	0418	0419	0420	0421	0422	0423	0424	0425	0426	0427	0428	0429	0430	0431
1B0	0432	0433	0434	0435	0436	0437	0438	0439	0440	0441	0442	0443	0444	0445	0446	0447
1C0	0448	0449	0450	0451	0452	0453	0454	0455	0456	0457	0458	0459	0460	0461	0462	0463
1D0	0464	0465	0466	0467	0468	0469	0470	0471	0472	0473	0474	0475	0476	0477	0478	0479
1E0	0480	0481	0482	0483	0484	0485	0486	0487	0488	0489	0490	0491	0492	0493	0494	0495
1F0	0496	0497	0498	0499	0500	0501	0502	0503	0504	0505	0506	0507	0508	0509	0510	0511
200	0512	0513	0514	0515	0516	0517	0518	0519	0520	0521	0522	0523	0524	0525	0526	0527
210	0528	0529	0530	0531	0532	0533	0534	0535	0536	0537	0538	0539	0540	0541	0542	0543
220	0544	0545	0546	0547	0548	0549	0550	0551	0552	0553	0554	0555	0556	0557	0558	0559
230	0560	0561	0562	0563	0564	0565	0566	0567	0568	0569	0570	0571	0572	0573	0574	0575
240	0576	0577	0578	0579	0580	0581	0582	0583	0584	0585	0586	0587	0588	0589	0590	0591
250	0592	0593	0594	0595	0596	0597	0598	0599	0600	0601	0602	0603	0604	0605	0606	0607
260	0608	0609	0610	0611	0612	0613	0614	0615	0616	0617	0618	0619	0620	0621	0622	0623
270	0624	0625	0626	0627	0628	0629	0630	0631	0632	0633	0634	0635	0636	0637	0638	0639
280	0640	0641	0642	0643	0644	0645	0646	0647	0648	0649	0650	0651	0652	0653	0654	0655
290	0656	0657	0658	0659	0660	0661	0662	0663	0664	0665	0666	0667	0668	0669	0670	0671
2A0	0672	0673	0674	0675	0676	0677	0678	0679	0680	0681	0682	0683	0684	0685	0686	0687
2B0	0688	0689	0690	0691	0692	0693	0694	0695	0696	0697	0698	0699	0700	0701	0702	0703
2C0	0704	0705	0706	0707	0708	0709	0710	0711	0712	0713	0714	0715	0716	0717	0718	0719
2D0	0720	0721	0722	0723	0724	0725	0726	0727	0728	0729	0730	0731	0732	0733	0734	0735
2E0	0736	0737	0738	0739	0740	0741	0742	0743	0744	0745	0746	0747	0748	0749	0750	0751
2F0	0752	0753	0754	0755	0756	0757	0758	0759	0760	0761	0762	0763	0764	0765	0766	0767
300	0768	0769	0770	0771	0772	0773	0774	0775	0776	0777	0778	0779	0780	0781	0782	0783
310	0784	0785	0786	0787	0788	0789	0790	0791	0792	0793	0794	0795	0796	0797	0798	0799
320	0800	0801	0802	0803	0804	0805	0806	0807	0808	0809	0810	0811	0812	0813	0814	0815
330	0816	0817	0818	0819	0820	0821	0822	0823	0824	0825	0826	0827	0828	0829	0830	0831
340	0832	0833	0834	0835	0836	0837	0838	0839	0840	0841	0842	0843	0844	0845	0846	0847
350	0848	0849	0850	0851	0852	0853	0854	0855	0856	0857	0858	0859	0860	0861	0862	0863
360	0864	0865	0866	0867	0868	0869	0870	0871	0872	0873	0874	0875	0876	0877	0878	0879
370	0880	0881	0882	0883	0884	0885	0886	0887	0888	0889	0890	0891	0892	0893	0894	0895
380	0896	0897	0898	0899	0900	0901	0902	0903	0904	0905	0906	0907	0908	0909	0910	0911
390	0912	0913	0914	0915	0916	0917	0918	0919	0920	0921	0922	0923	0924	0925	0926	0927
3A0	0928	0929	0930	0931	0932	0933	0934	0935	0936	0937	0938	0939	0940	0941	0942	0943
3B0	0944	0945	0946	0947	0948	0949	0950	0951	0952	0953	0954	0955	0956	0957	0958	0959
3C0	0960	0961	0962	0963	0964	0965	0966	0967	0968	0969	0970	0971	0972	0973	0974	0975
3D0	0976	0977	0978	0979	0980	0981	0982	0983	0984	0985	0986	0987	0988	0989	0990	0991
3E0	0992	0993	0994	0995	0996	0997	0998	0999	1000	1001	1002	1003	1004	1005	1006	1007
3F0	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1020	1021	1022	1023

HEXADECIMAL-DECIMAL INTEGER CONVERSION (Cont.)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
400	1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039
410	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1050	1051	1052	1053	1054	1055
420	1056	1057	1058	1059	1060	1061	1062	1063	1064	1065	1066	1067	1068	1069	1070	1071
430	1072	1073	1074	1075	1076	1077	1078	1079	1080	1081	1082	1083	1084	1085	1086	1087
440	1088	1089	1090	1091	1092	1093	1094	1095	1096	1097	1098	1099	1100	1101	1102	1103
450	1104	1105	1106	1107	1108	1109	1110	1111	1112	1113	1114	1115	1116	1117	1118	1119
460	1120	1121	1122	1123	1124	1125	1126	1127	1128	1129	1130	1131	1132	1133	1134	1135
470	1136	1137	1138	1139	1140	1141	1142	1143	1144	1145	1146	1147	1148	1149	1150	1151
480	1152	1153	1154	1155	1156	1157	1158	1159	1160	1161	1162	1163	1164	1165	1166	1167
490	1168	1169	1170	1171	1172	1173	1174	1175	1176	1177	1178	1179	1180	1181	1182	1183
4A0	1184	1185	1186	1187	1188	1189	1190	1191	1192	1193	1194	1195	1196	1197	1198	1199
4B0	1200	1201	1202	1203	1204	1205	1206	1207	1208	1209	1210	1211	1212	1213	1214	1215
4C0	1216	1217	1218	1219	1220	1221	1222	1223	1224	1225	1226	1227	1228	1229	1230	1231
4D0	1232	1233	1234	1235	1236	1237	1238	1239	1240	1241	1242	1243	1244	1245	1246	1247
4E0	1248	1249	1250	1251	1252	1253	1254	1255	1256	1257	1258	1259	1260	1261	1262	1263
4F0	1264	1265	1266	1267	1268	1269	1270	1271	1272	1273	1274	1275	1276	1277	1278	1279
500	1280	1281	1282	1283	1284	1285	1286	1287	1288	1289	1290	1291	1292	1293	1294	1295
510	1296	1297	1298	1299	1300	1301	1302	1303	1304	1305	1306	1307	1308	1309	1310	1311
520	1312	1313	1314	1315	1316	1317	1318	1319	1320	1321	1322	1323	1324	1325	1326	1327
530	1328	1329	1330	1331	1332	1333	1334	1335	1336	1337	1338	1339	1340	1341	1342	1343
540	1344	1345	1346	1347	1348	1349	1350	1351	1352	1353	1354	1355	1356	1357	1358	1359
550	1360	1361	1362	1363	1364	1365	1366	1367	1368	1369	1370	1371	1372	1373	1374	1375
560	1376	1377	1378	1379	1380	1381	1382	1383	1384	1385	1386	1387	1388	1389	1390	1391
570	1392	1393	1394	1395	1396	1397	1398	1399	1400	1401	1402	1403	1404	1405	1406	1407
580	1408	1409	1410	1411	1412	1413	1414	1415	1416	1417	1418	1419	1420	1421	1422	1423
590	1424	1425	1426	1427	1428	1429	1430	1431	1432	1433	1434	1435	1436	1437	1438	1439
5A0	1440	1441	1442	1443	1444	1445	1446	1447	1448	1449	1450	1451	1452	1453	1454	1455
5B0	1456	1457	1458	1459	1460	1461	1462	1463	1464	1465	1466	1467	1468	1469	1470	1471
5C0	1472	1473	1474	1475	1476	1477	1478	1479	1480	1481	1482	1483	1484	1485	1486	1487
5D0	1488	1489	1490	1491	1492	1493	1494	1495	1496	1497	1498	1499	1500	1501	1502	1503
5E0	1504	1505	1506	1507	1508	1509	1510	1511	1512	1513	1514	1515	1516	1517	1518	1519
5F0	1520	1521	1522	1523	1524	1525	1526	1527	1528	1529	1530	1531	1532	1533	1534	1535
600	1536	1537	1538	1539	1540	1541	1542	1543	1544	1545	1546	1547	1548	1549	1550	1551
610	1552	1553	1554	1555	1556	1557	1558	1559	1560	1561	1562	1563	1564	1565	1566	1567
620	1568	1569	1570	1571	1572	1573	1574	1575	1576	1577	1578	1579	1580	1581	1582	1583
630	1584	1585	1586	1587	1588	1589	1590	1591	1592	1593	1594	1595	1596	1597	1598	1599
640	1600	1601	1602	1603	1604	1605	1606	1607	1608	1609	1610	1611	1612	1613	1614	1615
650	1616	1617	1618	1619	1620	1621	1622	1623	1624	1625	1626	1627	1628	1629	1630	1631
660	1632	1633	1634	1635	1636	1637	1638	1639	1640	1641	1642	1643	1644	1645	1646	1647
670	1648	1649	1650	1651	1652	1653	1654	1655	1656	1657	1658	1659	1660	1661	1662	1663
680	1664	1665	1666	1667	1668	1669	1670	1671	1672	1673	1674	1675	1676	1677	1678	1679
690	1680	1681	1682	1683	1684	1685	1686	1687	1688	1689	1690	1691	1692	1693	1694	1695
6A0	1696	1697	1698	1699	1700	1701	1702	1703	1704	1705	1706	1707	1708	1709	1710	1711
6B0	1712	1713	1714	1715	1716	1717	1718	1719	1720	1721	1722	1723	1724	1725	1726	1727
6C0	1728	1729	1730	1731	1732	1733	1734	1735	1736	1737	1738	1739	1740	1741	1742	1743
6D0	1744	1745	1746	1747	1748	1749	1750	1751	1752	1753	1754	1755	1756	1757	1758	1759
6E0	1760	1761	1762	1763	1764	1765	1766	1767	1768	1769	1770	1771	1772	1773	1774	1775
6F0	1776	1777	1778	1779	1780	1781	1782	1783	1784	1785	1786	1787	1788	1789	1790	1791

HEXADECIMAL-DECIMAL INTEGER CONVERSION (Cont.)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
700	1792	1793	1794	1795	1796	1797	1798	1799	1800	1801	1802	1803	1804	1805	1806	1807
710	1808	1809	1810	1811	1812	1813	1814	1815	1816	1817	1818	1819	1820	1821	1822	1823
720	1824	1825	1826	1827	1828	1829	1830	1831	1832	1833	1834	1835	1836	1837	1838	1839
730	1840	1841	1842	1843	1844	1845	1846	1847	1848	1849	1850	1851	1852	1853	1854	1855
740	1856	1857	1858	1859	1860	1861	1862	1863	1864	1865	1866	1867	1868	1869	1870	1871
750	1872	1873	1874	1875	1876	1877	1878	1879	1880	1881	1882	1883	1884	1885	1886	1887
760	1888	1889	1890	1891	1892	1893	1894	1895	1896	1897	1898	1899	1900	1901	1902	1903
770	1904	1905	1906	1907	1908	1909	1910	1911	1912	1913	1914	1915	1916	1917	1918	1919
780	1920	1921	1922	1923	1924	1925	1926	1927	1928	1929	1930	1931	1932	1933	1934	1935
790	1936	1937	1938	1939	1940	1941	1942	1943	1944	1945	1946	1947	1948	1949	1950	1951
7A0	1952	1953	1954	1955	1956	1957	1958	1959	1960	1961	1962	1963	1964	1965	1966	1967
7B0	1968	1969	1970	1971	1972	1973	1974	1975	1976	1977	1978	1979	1980	1981	1982	1983
7C0	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999
7D0	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
7E0	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031
7F0	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047
800	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063
810	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079
820	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095
830	2096	2097	2098	2099	2100	2101	2102	2103	2104	2105	2106	2107	2108	2109	2110	2111
840	2112	2113	2114	2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127
850	2128	2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143
860	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156	2157	2158	2159
870	2160	2161	2162	2163	2164	2165	2166	2167	2168	2169	2170	2171	2172	2173	2174	2175
880	2176	2177	2178	2179	2180	2181	2182	2183	2184	2185	2186	2187	2188	2189	2190	2191
890	2192	2193	2194	2195	2196	2197	2198	2199	2200	2201	2202	2203	2204	2205	2206	2207
8A0	2208	2209	2210	2211	2212	2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223
8B0	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239
8C0	2240	2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254	2255
8D0	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271
8E0	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2287
8F0	2288	2289	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303
900	2304	2305	2306	2307	2308	2309	2310	2311	2312	2313	2314	2315	2316	2317	2318	2319
910	2320	2321	2322	2323	2324	2325	2326	2327	2328	2329	2330	2331	2332	2333	2334	2335
920	2336	2337	2338	2339	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351
930	2352	2353	2354	2355	2356	2357	2358	2359	2360	2361	2362	2363	2364	2365	2366	2367
940	2368	2369	2370	2371	2372	2373	2374	2375	2376	2377	2378	2379	2380	2381	2382	2383
950	2384	2385	2386	2387	2388	2389	2390	2391	2392	2393	2394	2395	2396	2397	2398	2399
960	2400	2401	2402	2403	2404	2405	2406	2407	2408	2409	2410	2411	2412	2413	2414	2415
970	2416	2417	2418	2419	2420	2421	2422	2423	2424	2425	2426	2427	2428	2429	2430	2431
980	2432	2433	2434	2435	2436	2437	2438	2439	2440	2441	2442	2443	2444	2445	2446	2447
990	2448	2449	2450	2451	2452	2453	2454	2455	2456	2457	2458	2459	2460	2461	2462	2463
9A0	2464	2465	2466	2467	2468	2469	2470	2471	2472	2473	2474	2475	2476	2477	2478	2479
9B0	2480	2481	2482	2483	2484	2485	2486	2487	2488	2489	2490	2491	2492	2493	2494	2495
9C0	2496	2497	2498	2499	2500	2501	2502	2503	2504	2505	2506	2507	2508	2509	2510	2511
9D0	2512	2513	2514	2515	2516	2517	2518	2519	2520	2521	2522	2523	2524	2525	2526	2527
9E0	2528	2529	2530	2531	2532	2533	2534	2535	2536	2537	2538	2539	2540	2541	2542	2543
9F0	2544	2545	2546	2547	2548	2549	2550	2551	2552	2553	2554	2555	2556	2557	2558	2559

HEXADECIMAL-DECIMAL INTEGER CONVERSION (Cont.)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A00	2560	2561	2562	2563	2564	2565	2566	2567	2568	2569	2570	2571	2572	2573	2574	2575
A10	2576	2577	2578	2579	2580	2581	2582	2583	2584	2585	2586	2587	2588	2589	2590	2591
A20	2592	2593	2594	2595	2596	2597	2598	2599	2600	2601	2602	2603	2604	2605	2606	2607
A30	2608	2609	2610	2611	2612	2613	2614	2615	2616	2617	2618	2619	2620	2621	2622	2623
A40	2624	2625	2626	2627	2628	2629	2630	2631	2632	2633	2634	2635	2636	2637	2638	2639
A50	2640	2641	2642	2643	2644	2645	2646	2647	2648	2649	2650	2651	2652	2653	2654	2655
A60	2656	2657	2658	2659	2660	2661	2662	2663	2664	2665	2666	2667	2668	2669	2670	2671
A70	2672	2673	2674	2675	2676	2677	2678	2679	2680	2681	2682	2683	2684	2685	2686	2687
A80	2688	2689	2690	2691	2692	2693	2694	2695	2696	2697	2698	2699	2700	2701	2702	2703
A90	2704	2705	2706	2707	2708	2709	2710	2711	2712	2713	2714	2715	2716	2717	2718	2719
AA0	2720	2721	2722	2723	2724	2725	2726	2727	2728	2729	2730	2731	2732	2733	2734	2735
AB0	2736	2737	2738	2739	2740	2741	2742	2743	2744	2745	2746	2747	2748	2749	2750	2751
AC0	2752	2753	2754	2755	2756	2757	2758	2759	2760	2761	2762	2763	2764	2765	2766	2767
AD0	2768	2769	2770	2771	2772	2773	2774	2775	2776	2777	2778	2779	2780	2781	2782	2783
AE0	2784	2785	2786	2787	2788	2789	2790	2791	2792	2793	2794	2795	2796	2797	2798	2799
AF0	2800	2801	2802	2803	2804	2805	2806	2807	2808	2809	2810	2811	2812	2813	2814	2815
B00	2816	2817	2818	2819	2820	2821	2822	2823	2824	2825	2826	2827	2828	2829	2830	2831
B10	2832	2833	2834	2835	2836	2837	2838	2839	2840	2841	2842	2843	2844	2845	2846	2847
B20	2848	2849	2850	2851	2852	2853	2854	2855	2856	2857	2858	2859	2860	2861	2862	2863
B30	2864	2865	2866	2867	2868	2869	2870	2871	2872	2873	2874	2875	2876	2877	2878	2879
B40	2880	2881	2882	2883	2884	2885	2886	2887	2888	2889	2890	2891	2892	2893	2894	2895
B50	2896	2897	2898	2899	2900	2901	2902	2903	2904	2905	2906	2907	2908	2909	2910	2911
B60	2912	2913	2914	2915	2916	2917	2918	2919	2920	2921	2922	2923	2924	2925	2926	2927
B70	2928	2929	2930	2931	2932	2933	2934	2935	2936	2937	2938	2939	2940	2941	2942	2943
B80	2944	2945	2946	2947	2948	2949	2950	2951	2952	2953	2954	2955	2956	2957	2958	2959
B90	2960	2961	2962	2963	2964	2965	2966	2967	2968	2969	2970	2971	2972	2973	2974	2975
BA0	2976	2977	2978	2979	2980	2981	2982	2983	2984	2985	2986	2987	2988	2989	2990	2991
BB0	2992	2993	2994	2995	2996	2997	2998	2999	3000	3001	3002	3003	3004	3005	3006	3007
BC0	3008	3009	3010	3011	3012	3013	3014	3015	3016	3017	3018	3019	3020	3021	3022	3023
BD0	3024	3025	3026	3027	3028	3029	3030	3031	3032	3033	3034	3035	3036	3037	3038	3039
BE0	3040	3041	3042	3043	3044	3045	3046	3047	3048	3049	3050	3051	3052	3053	3054	3055
BF0	3056	3057	3058	3059	3060	3061	3062	3063	3064	3065	3066	3067	3068	3069	3070	3071
C00	3072	3073	3074	3075	3076	3077	3078	3079	3080	3081	3082	3083	3084	3085	3086	3087
C10	3088	3089	3090	3091	3092	3093	3094	3095	3096	3097	3098	3099	3100	3101	3102	3103
C20	3104	3105	3106	3107	3108	3109	3110	3111	3112	3113	3114	3115	3116	3117	3118	3119
C30	3120	3121	3122	3123	3124	3125	3126	3127	3128	3129	3130	3131	3132	3133	3134	3135
C40	3136	3137	3138	3139	3140	3141	3142	3143	3144	3145	3146	3147	3148	3149	3150	3151
C50	3152	3153	3154	3155	3156	3157	3158	3159	3160	3161	3162	3163	3164	3165	3166	3167
C60	3168	3169	3170	3171	3172	3173	3174	3175	3176	3177	3178	3179	3180	3181	3182	3183
C70	3184	3185	3186	3187	3188	3189	3190	3191	3192	3193	3194	3195	3196	3197	3198	3199
C80	3200	3201	3202	3203	3204	3205	3206	3207	3208	3209	3210	3211	3212	3213	3214	3215
C90	3216	3217	3218	3219	3220	3221	3222	3223	3224	3225	3226	3227	3228	3229	3230	3231
CA0	3232	3233	3234	3235	3236	3237	3238	3239	3240	3241	3242	3243	3244	3245	3246	3247
CB0	3248	3249	3250	3251	3252	3253	3254	3255	3256	3257	3258	3259	3260	3261	3262	3263
CC0	3264	3265	3266	3267	3268	3269	3270	3271	3272	3273	3274	3275	3276	3277	3278	3279
CD0	3280	3281	3282	3283	3284	3285	3286	3287	3288	3289	3290	3291	3292	3293	3294	3295
CE0	3296	3297	3298	3299	3300	3301	3302	3303	3304	3305	3306	3307	3308	3309	3310	3311
CF0	3312	3313	3314	3315	3316	3317	3318	3319	3320	3321	3322	3323	3324	3325	3326	3327

HEXADECIMAL-DECIMAL INTEGER CONVERSION (Cont.)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
D00	3328	3329	3330	3331	3332	3333	3334	3335	3336	3337	3338	3339	3340	3341	3342	3343
D10	3344	3345	3346	3347	3348	3349	3350	3351	3352	3353	3354	3355	3356	3357	3358	3359
D20	3360	3361	3362	3363	3364	3365	3366	3367	3368	3369	3370	3371	3372	3373	3374	3375
D30	3376	3377	3378	3379	3380	3381	3382	3383	3384	3385	3386	3387	3388	3389	3390	3391
D40	3392	3393	3394	3395	3396	3397	3398	3399	3400	3401	3402	3403	3404	3405	3406	3407
D50	3408	3409	3410	3411	3412	3413	3414	3415	3416	3417	3418	3419	3420	3421	3422	3423
D60	3424	3425	3426	3427	3428	3429	3430	3431	3432	3433	3434	3435	3436	3437	3438	3439
D70	3440	3441	3442	3443	3444	3445	3446	3447	3448	3449	3450	3451	3452	3453	3454	3455
D80	3456	3457	3458	3459	3460	3461	3462	3463	3464	3465	3466	3467	3468	3469	3470	3471
D90	3472	3473	3474	3475	3476	3477	3478	3479	3480	3481	3482	3483	3484	3485	3486	3487
DA0	3488	3489	3490	3491	3492	3493	3494	3495	3496	3497	3498	3499	3500	3501	3502	3503
DB0	3504	3505	3506	3507	3508	3509	3510	3511	3512	3513	3514	3515	3516	3517	3518	3519
DC0	3520	3521	3522	3523	3524	3525	3526	3527	3528	3529	3530	3531	3532	3533	3534	3535
DD0	3536	3537	3538	3539	3540	3541	3542	3543	3544	3545	3546	3547	3548	3549	3550	3551
DE0	3552	3553	3554	3555	3556	3557	3558	3559	3560	3561	3562	3563	3564	3565	3566	3567
DF0	3568	3569	3570	3571	3572	3573	3574	3575	3576	3577	3578	3579	3580	3581	3582	3583
E00	3584	3585	3586	3587	3588	3589	3590	3591	3592	3593	3594	3595	3596	3597	3598	3599
E10	3600	3601	3602	3603	3604	3605	3606	3607	3608	3609	3610	3611	3612	3613	3614	3615
E20	3616	3617	3618	3619	3620	3621	3622	3623	3624	3625	3626	3627	3628	3629	3630	3631
E30	3632	3633	3634	3635	3636	3637	3638	3639	3640	3641	3642	3643	3644	3645	3646	3647
E40	3648	3649	3650	3651	3652	3653	3654	3655	3656	3657	3658	3659	3660	3661	3662	3663
E50	3664	3665	3666	3667	3668	3669	3670	3671	3672	3673	3674	3675	3676	3677	3678	3679
E60	3680	3681	3682	3683	3684	3685	3686	3687	3688	3689	3690	3691	3692	3693	3694	3695
E70	3696	3697	3698	3699	3700	3701	3702	3703	3704	3705	3706	3707	3708	3709	3710	3711
E80	3712	3713	3714	3715	3716	3717	3718	3719	3720	3721	3722	3723	3724	3725	3726	3727
E90	3728	3729	3730	3731	3732	3733	3734	3735	3736	3737	3738	3739	3740	3741	3742	3743
EA0	3744	3745	3746	3747	3748	3749	3750	3751	3752	3753	3754	3755	3756	3757	3758	3759
EB0	3760	3761	3762	3763	3764	3765	3766	3767	3768	3769	3770	3771	3772	3773	3774	3775
EC0	3776	3777	3778	3779	3780	3781	3782	3783	3784	3785	3786	3787	3788	3789	3790	3791
ED0	3792	3793	3794	3795	3796	3797	3798	3799	3800	3801	3802	3803	3804	3805	3806	3807
EE0	3808	3809	3810	3811	3812	3813	3814	3815	3816	3817	3818	3819	3820	3821	3822	3823
EF0	3824	3825	3826	3827	3828	3829	3830	3831	3832	3833	3834	3835	3836	3837	3838	3839
F00	3840	3841	3842	3843	3844	3845	3846	3847	3848	3849	3850	3851	3852	3853	3854	3855
F10	3856	3857	3858	3859	3860	3861	3862	3863	3864	3865	3866	3867	3868	3869	3870	3871
F20	3872	3873	3874	3875	3876	3877	3878	3879	3880	3881	3882	3883	3884	3885	3886	3887
F30	3888	3889	3890	3891	3892	3893	3894	3895	3896	3897	3898	3899	3900	3901	3902	3903
F40	3904	3905	3906	3907	3908	3909	3910	3911	3912	3913	3914	3915	3916	3917	3918	3919
F50	3920	3921	3922	3923	3924	3925	3926	3927	3928	3929	3930	3931	3932	3933	3934	3935
F60	3936	3937	3938	3939	3940	3941	3942	3943	3944	3945	3946	3947	3948	3949	3950	3951
F70	3952	3953	3954	3955	3956	3957	3958	3959	3960	3961	3962	3963	3964	3965	3966	3967
F80	3968	3969	3970	3971	3972	3973	3974	3975	3976	3977	3978	3979	3980	3981	3982	3983
F90	3984	3985	3986	3987	3988	3989	3990	3991	3992	3993	3994	3995	3996	3997	3998	3999
FA0	4000	4001	4002	4003	4004	4005	4006	4007	4008	4009	4010	4011	4012	4013	4014	4015
FB0	4016	4017	4018	4019	4020	4021	4022	4023	4024	4025	4026	4027	4028	4029	4030	4031
FC0	4032	4033	4034	4035	4036	4037	4038	4039	4040	4041	4042	4043	4044	4045	4046	4047
FD0	4048	4049	4050	4051	4052	4053	4054	4055	4056	4057	4058	4059	4060	4061	4062	4063
FE0	4064	4065	4066	4067	4068	4069	4070	4071	4072	4073	4074	4075	4076	4077	4078	4079
FF0	4080	4081	4082	4083	4084	4085	4086	4087	4088	4089	4090	4091	4092	4093	4094	4095

HEXADECIMAL-DECIMAL FRACTION CONVERSION

Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal
.00 00 00 00	.00000 00000	.40 00 00 00	.25000 00000	.80 00 00 00	.50000 00000	.C0 00 00 00	.75000 00000
.01 00 00 00	.00390 62500	.41 00 00 00	.25390 62500	.81 00 00 00	.50390 62500	.C1 00 00 00	.75390 62500
.02 00 00 00	.00781 25000	.42 00 00 00	.25781 25000	.82 00 00 00	.50781 25000	.C2 00 00 00	.75781 25000
.03 00 00 00	.01171 87500	.43 00 00 00	.26171 87500	.83 00 00 00	.51171 87500	.C3 00 00 00	.76171 87500
.04 00 00 00	.01562 50000	.44 00 00 00	.26562 50000	.84 00 00 00	.51562 50000	.C4 00 00 00	.76562 50000
.05 00 00 00	.01953 12500	.45 00 00 00	.26953 12500	.85 00 00 00	.51953 12500	.C5 00 00 00	.76953 12500
.06 00 00 00	.02343 75000	.46 00 00 00	.27343 75000	.86 00 00 00	.52343 75000	.C6 00 00 00	.77343 75000
.07 00 00 00	.02734 37500	.47 00 00 00	.27734 37500	.87 00 00 00	.52734 37500	.C7 00 00 00	.77734 37500
.08 00 00 00	.03125 00000	.48 00 00 00	.28125 00000	.88 00 00 00	.53125 00000	.C8 00 00 00	.78125 00000
.09 00 00 00	.03515 62500	.49 00 00 00	.28515 62500	.89 00 00 00	.53515 62500	.C9 00 00 00	.78515 62500
.0A 00 00 00	.03906 25000	.4A 00 00 00	.28906 25000	.8A 00 00 00	.53906 25000	.CA 00 00 00	.78906 25000
.0B 00 00 00	.04296 87500	.4B 00 00 00	.29296 87500	.8B 00 00 00	.54296 87500	.CB 00 00 00	.79296 87500
.0C 00 00 00	.04687 50000	.4C 00 00 00	.29687 50000	.8C 00 00 00	.54687 50000	.CC 00 00 00	.79687 50000
.0D 00 00 00	.05078 12500	.4D 00 00 00	.30078 12500	.8D 00 00 00	.55078 12500	.CD 00 00 00	.80078 12500
.0E 00 00 00	.05468 75000	.4E 00 00 00	.30468 75000	.8E 00 00 00	.55468 75000	.CE 00 00 00	.80468 75000
.0F 00 00 00	.05859 37500	.4F 00 00 00	.30859 37500	.8F 00 00 00	.55859 37500	.CF 00 00 00	.80859 37500
.10 00 00 00	.06250 00000	.50 00 00 00	.31250 00000	.90 00 00 00	.56250 00000	.D0 00 00 00	.81250 00000
.11 00 00 00	.06640 62500	.51 00 00 00	.31640 62500	.91 00 00 00	.56640 62500	.D1 00 00 00	.81640 62500
.12 00 00 00	.07031 25000	.52 00 00 00	.32031 25000	.92 00 00 00	.57031 25000	.D2 00 00 00	.82031 25000
.13 00 00 00	.07421 87500	.53 00 00 00	.32421 87500	.93 00 00 00	.57421 87500	.D3 00 00 00	.82421 87500
.14 00 00 00	.07812 50000	.54 00 00 00	.32812 50000	.94 00 00 00	.57812 50000	.D4 00 00 00	.82812 50000
.15 00 00 00	.08203 12500	.55 00 00 00	.33203 12500	.95 00 00 00	.58203 12500	.D5 00 00 00	.83203 12500
.16 00 00 00	.08593 75000	.56 00 00 00	.33593 75000	.96 00 00 00	.58593 75000	.D6 00 00 00	.83593 75000
.17 00 00 00	.08984 37500	.57 00 00 00	.33984 37500	.97 00 00 00	.58984 37500	.D7 00 00 00	.83984 37500
.18 00 00 00	.09375 00000	.58 00 00 00	.34375 00000	.98 00 00 00	.59375 00000	.D8 00 00 00	.84375 00000
.19 00 00 00	.09765 62500	.59 00 00 00	.34765 62500	.99 00 00 00	.59765 62500	.D9 00 00 00	.84765 62500
.1A 00 00 00	.10156 25000	.5A 00 00 00	.35156 25000	.9A 00 00 00	.60156 25000	.DA 00 00 00	.85156 25000
.1B 00 00 00	.10546 87500	.5B 00 00 00	.35546 87500	.9B 00 00 00	.60546 87500	.DB 00 00 00	.85546 87500
.1C 00 00 00	.10937 50000	.5C 00 00 00	.35937 50000	.9C 00 00 00	.60937 50000	.DC 00 00 00	.85937 50000
.1D 00 00 00	.11328 12500	.5D 00 00 00	.36328 12500	.9D 00 00 00	.61328 12500	.DD 00 00 00	.86328 12500
.1E 00 00 00	.11718 75000	.5E 00 00 00	.36718 75000	.9E 00 00 00	.61718 75000	.DE 00 00 00	.86718 75000
.1F 00 00 00	.12109 37500	.5F 00 00 00	.37109 37500	.9F 00 00 00	.62109 37500	.DF 00 00 00	.87109 37500
.20 00 00 00	.12500 00000	.60 00 00 00	.37500 00000	.A0 00 00 00	.62500 00000	.E0 00 00 00	.87500 00000
.21 00 00 00	.12890 62500	.61 00 00 00	.37890 62500	.A1 00 00 00	.62890 62500	.E1 00 00 00	.87890 62500
.22 00 00 00	.13281 25000	.62 00 00 00	.38281 25000	.A2 00 00 00	.63281 25000	.E2 00 00 00	.88281 25000
.23 00 00 00	.13671 87500	.63 00 00 00	.38671 87500	.A3 00 00 00	.63671 87500	.E3 00 00 00	.88671 87500
.24 00 00 00	.14062 50000	.64 00 00 00	.39062 50000	.A4 00 00 00	.64062 50000	.E4 00 00 00	.89062 50000
.25 00 00 00	.14453 12500	.65 00 00 00	.39453 12500	.A5 00 00 00	.64453 12500	.E5 00 00 00	.89453 12500
.26 00 00 00	.14843 75000	.66 00 00 00	.39843 75000	.A6 00 00 00	.64843 75000	.E6 00 00 00	.89843 75000
.27 00 00 00	.15234 37500	.67 00 00 00	.40234 37500	.A7 00 00 00	.65234 37500	.E7 00 00 00	.90234 37500
.28 00 00 00	.15625 00000	.68 00 00 00	.40625 00000	.A8 00 00 00	.65625 00000	.E8 00 00 00	.90625 00000
.29 00 00 00	.16015 62500	.69 00 00 00	.41015 62500	.A9 00 00 00	.66015 62500	.E9 00 00 00	.91015 62500
.2A 00 00 00	.16406 25000	.6A 00 00 00	.41406 25000	.AA 00 00 00	.66406 25000	.EA 00 00 00	.91406 25000
.2B 00 00 00	.16796 87500	.6B 00 00 00	.41796 87500	.AB 00 00 00	.66796 87500	.EB 00 00 00	.91796 87500
.2C 00 00 00	.17187 50000	.6C 00 00 00	.42187 50000	.AC 00 00 00	.67187 50000	.EC 00 00 00	.92187 50000
.2D 00 00 00	.17578 12500	.6D 00 00 00	.42578 12500	.AD 00 00 00	.67578 12500	.ED 00 00 00	.92578 12500
.2E 00 00 00	.17968 75000	.6E 00 00 00	.42968 75000	.AE 00 00 00	.67968 75000	.EE 00 00 00	.92968 75000
.2F 00 00 00	.18359 37500	.6F 00 00 00	.43359 37500	.AF 00 00 00	.68359 37500	.EF 00 00 00	.93359 37500
.30 00 00 00	.18750 00000	.70 00 00 00	.43750 00000	.B0 00 00 00	.68750 00000	.F0 00 00 00	.93750 00000
.31 00 00 00	.19140 62500	.71 00 00 00	.44140 62500	.B1 00 00 00	.69140 62500	.F1 00 00 00	.94140 62500
.32 00 00 00	.19531 25000	.72 00 00 00	.44531 25000	.B2 00 00 00	.69531 25000	.F2 00 00 00	.94531 25000
.33 00 00 00	.19921 87500	.73 00 00 00	.44921 87500	.B3 00 00 00	.69921 87500	.F3 00 00 00	.94921 87500
.34 00 00 00	.20312 50000	.74 00 00 00	.45312 50000	.B4 00 00 00	.70312 50000	.F4 00 00 00	.95312 50000
.35 00 00 00	.20703 12500	.75 00 00 00	.45703 12500	.B5 00 00 00	.70703 12500	.F5 00 00 00	.95703 12500
.36 00 00 00	.21093 75000	.76 00 00 00	.46093 75000	.B6 00 00 00	.71093 75000	.F6 00 00 00	.96093 75000
.37 00 00 00	.21484 37500	.77 00 00 00	.46484 37500	.B7 00 00 00	.71484 37500	.F7 00 00 00	.96484 37500
.38 00 00 00	.21875 00000	.78 00 00 00	.46875 00000	.B8 00 00 00	.71875 00000	.F8 00 00 00	.96875 00000
.39 00 00 00	.22265 62500	.79 00 00 00	.47265 62500	.B9 00 00 00	.72265 62500	.F9 00 00 00	.97265 62500
.3A 00 00 00	.22656 25000	.7A 00 00 00	.47656 25000	.BA 00 00 00	.72656 25000	.FA 00 00 00	.97656 25000
.3B 00 00 00	.23046 87500	.7B 00 00 00	.48046 87500	.BB 00 00 00	.73046 87500	.FB 00 00 00	.98046 87500
.3C 00 00 00	.23437 50000	.7C 00 00 00	.48437 50000	.BC 00 00 00	.73437 50000	.FC 00 00 00	.98437 50000
.3D 00 00 00	.23828 12500	.7D 00 00 00	.48828 12500	.BD 00 00 00	.73828 12500	.FD 00 00 00	.98828 12500
.3E 00 00 00	.24218 75000	.7E 00 00 00	.49218 75000	.BE 00 00 00	.74218 75000	.FE 00 00 00	.99218 75000
.3F 00 00 00	.24609 37500	.7F 00 00 00	.49609 37500	.BF 00 00 00	.74609 37500	.FF 00 00 00	.99609 37500

HEXADECIMAL-DECIMAL FRACTION CONVERSION (Cont.)

Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal
.00 00 00 00	.00000 00000	.00 40 00 00	.00097 65625	.00 80 00 00	.00195 31250	.00 C0 00 00	.00292 96875
.00 01 00 00	.00001 52587	.00 41 00 00	.00099 18212	.00 81 00 00	.00196 83837	.00 C1 00 00	.00294 49462
.00 02 00 00	.00003 05175	.00 42 00 00	.00100 70800	.00 82 00 00	.00198 36425	.00 C2 00 00	.00296 02050
.00 03 00 00	.00004 57763	.00 43 00 00	.00102 23388	.00 83 00 00	.00199 89013	.00 C3 00 00	.00297 54638
.00 04 00 00	.00006 10351	.00 44 00 00	.00103 75976	.00 84 00 00	.00201 41601	.00 C4 00 00	.00299 07226
.00 05 00 00	.00007 62939	.00 45 00 00	.00105 28564	.00 85 00 00	.00202 94189	.00 C5 00 00	.00300 59814
.00 06 00 00	.00009 15527	.00 46 00 00	.00106 81152	.00 86 00 00	.00204 46777	.00 C6 00 00	.00302 12402
.00 07 00 00	.00010 68115	.00 47 00 00	.00108 33740	.00 87 00 00	.00205 99365	.00 C7 00 00	.00303 64990
.00 08 00 00	.00012 20703	.00 48 00 00	.00109 86328	.00 88 00 00	.00207 51953	.00 C8 00 00	.00305 17578
.00 09 00 00	.00013 73291	.00 49 00 00	.00111 38916	.00 89 00 00	.00209 04541	.00 C9 00 00	.00306 70166
.00 0A 00 00	.00015 25878	.00 4A 00 00	.00112 91503	.00 8A 00 00	.00210 57128	.00 CA 00 00	.00308 22753
.00 0B 00 00	.00016 78466	.00 4B 00 00	.00114 44091	.00 8B 00 00	.00212 09716	.00 CB 00 00	.00309 75341
.00 0C 00 00	.00018 31054	.00 4C 00 00	.00115 96679	.00 8C 00 00	.00213 62304	.00 CC 00 00	.00311 27929
.00 0D 00 00	.00019 83642	.00 4D 00 00	.00117 49267	.00 8D 00 00	.00215 14892	.00 CD 00 00	.00312 80517
.00 0E 00 00	.00021 36230	.00 4E 00 00	.00119 01855	.00 8E 00 00	.00216 67480	.00 CE 00 00	.00314 33105
.00 0F 00 00	.00022 88818	.00 4F 00 00	.00120 54443	.00 8F 00 00	.00218 20068	.00 CF 00 00	.00315 85693
.00 10 00 00	.00024 41406	.00 50 00 00	.00122 07031	.00 90 00 00	.00219 72656	.00 D0 00 00	.00317 38281
.00 11 00 00	.00025 93994	.00 51 00 00	.00123 59619	.00 91 00 00	.00221 25244	.00 D1 00 00	.00318 90869
.00 12 00 00	.00027 46582	.00 52 00 00	.00125 12207	.00 92 00 00	.00222 77832	.00 D2 00 00	.00320 43457
.00 13 00 00	.00028 99169	.00 53 00 00	.00126 64794	.00 93 00 00	.00224 30419	.00 D3 00 00	.00321 96044
.00 14 00 00	.00030 51757	.00 54 00 00	.00128 17382	.00 94 00 00	.00225 83007	.00 D4 00 00	.00323 48632
.00 15 00 00	.00032 04345	.00 55 00 00	.00129 69970	.00 95 00 00	.00227 35595	.00 D5 00 00	.00325 01220
.00 16 00 00	.00033 56933	.00 56 00 00	.00131 22558	.00 96 00 00	.00228 88183	.00 D6 00 00	.00326 53808
.00 17 00 00	.00035 09521	.00 57 00 00	.00132 75146	.00 97 00 00	.00230 40771	.00 D7 00 00	.00328 06396
.00 18 00 00	.00036 62109	.00 58 00 00	.00134 27734	.00 98 00 00	.00231 93359	.00 D8 00 00	.00329 58984
.00 19 00 00	.00038 14697	.00 59 00 00	.00135 80322	.00 99 00 00	.00233 45947	.00 D9 00 00	.00331 11572
.00 1A 00 00	.00039 67285	.00 5A 00 00	.00137 32910	.00 9A 00 00	.00234 98535	.00 DA 00 00	.00332 64160
.00 1B 00 00	.00041 19873	.00 5B 00 00	.00138 85498	.00 9B 00 00	.00236 51123	.00 DB 00 00	.00334 16748
.00 1C 00 00	.00042 72460	.00 5C 00 00	.00140 38085	.00 9C 00 00	.00238 03710	.00 DC 00 00	.00335 69335
.00 1D 00 00	.00044 25048	.00 5D 00 00	.00141 90673	.00 9D 00 00	.00239 56298	.00 DD 00 00	.00337 21923
.00 1E 00 00	.00045 77636	.00 5E 00 00	.00143 43261	.00 9E 00 00	.00241 08886	.00 DE 00 00	.00338 74511
.00 1F 00 00	.00047 30224	.00 5F 00 00	.00144 95849	.00 9F 00 00	.00242 61474	.00 DF 00 00	.00340 27099
.00 20 00 00	.00048 82812	.00 60 00 00	.00146 48437	.00 A0 00 00	.00244 14062	.00 E0 00 00	.00341 79687
.00 21 00 00	.00050 35400	.00 61 00 00	.00148 01025	.00 A1 00 00	.00245 66650	.00 E1 00 00	.00343 32275
.00 22 00 00	.00051 87988	.00 62 00 00	.00149 53613	.00 A2 00 00	.00247 19238	.00 E2 00 00	.00344 84863
.00 23 00 00	.00053 40576	.00 63 00 00	.00151 06201	.00 A3 00 00	.00248 71826	.00 E3 00 00	.00346 37451
.00 24 00 00	.00054 93164	.00 64 00 00	.00152 58789	.00 A4 00 00	.00250 24414	.00 E4 00 00	.00347 90039
.00 25 00 00	.00056 45751	.00 65 00 00	.00154 11376	.00 A5 00 00	.00251 77001	.00 E5 00 00	.00349 42626
.00 26 00 00	.00057 98339	.00 66 00 00	.00155 63964	.00 A6 00 00	.00253 29589	.00 E6 00 00	.00350 95214
.00 27 00 00	.00059 50927	.00 67 00 00	.00157 16552	.00 A7 00 00	.00254 82177	.00 E7 00 00	.00352 47802
.00 28 00 00	.00061 03515	.00 68 00 00	.00158 69140	.00 A8 00 00	.00256 34765	.00 E8 00 00	.00354 00390
.00 29 00 00	.00062 56103	.00 69 00 00	.00160 21728	.00 A9 00 00	.00257 87353	.00 E9 00 00	.00355 52978
.00 2A 00 00	.00064 08691	.00 6A 00 00	.00161 74316	.00 AA 00 00	.00259 39941	.00 EA 00 00	.00357 05566
.00 2B 00 00	.00065 61279	.00 6B 00 00	.00163 26904	.00 AB 00 00	.00260 92529	.00 EB 00 00	.00358 58154
.00 2C 00 00	.00067 13867	.00 6C 00 00	.00164 79492	.00 AC 00 00	.00262 45117	.00 EC 00 00	.00360 10742
.00 2D 00 00	.00068 66455	.00 6D 00 00	.00166 32080	.00 AD 00 00	.00263 97705	.00 ED 00 00	.00361 63330
.00 2E 00 00	.00070 19042	.00 6E 00 00	.00167 84667	.00 AE 00 00	.00265 50292	.00 EE 00 00	.00363 15917
.00 2F 00 00	.00071 71630	.00 6F 00 00	.00169 37255	.00 AF 00 00	.00267 02880	.00 EF 00 00	.00364 68505
.00 30 00 00	.00073 24218	.00 70 00 00	.00170 89843	.00 B0 00 00	.00268 55468	.00 F0 00 00	.00366 21093
.00 31 00 00	.00074 76806	.00 71 00 00	.00172 42431	.00 B1 00 00	.00270 08056	.00 F1 00 00	.00367 73681
.00 32 00 00	.00076 29394	.00 72 00 00	.00173 95019	.00 B2 00 00	.00271 60644	.00 F2 00 00	.00369 26269
.00 33 00 00	.00077 81982	.00 73 00 00	.00175 47607	.00 B3 00 00	.00273 13232	.00 F3 00 00	.00370 78857
.00 34 00 00	.00079 34570	.00 74 00 00	.00177 00195	.00 B4 00 00	.00274 65820	.00 F4 00 00	.00372 31445
.00 35 00 00	.00080 87158	.00 75 00 00	.00178 52783	.00 B5 00 00	.00276 18408	.00 F5 00 00	.00373 84033
.00 36 00 00	.00082 39746	.00 76 00 00	.00180 05371	.00 B6 00 00	.00277 70996	.00 F6 00 00	.00375 36621
.00 37 00 00	.00083 92333	.00 77 00 00	.00181 57958	.00 B7 00 00	.00279 23583	.00 F7 00 00	.00376 89208
.00 38 00 00	.00085 44921	.00 78 00 00	.00183 10546	.00 B8 00 00	.00280 76171	.00 F8 00 00	.00378 41796
.00 39 00 00	.00086 97509	.00 79 00 00	.00184 63134	.00 B9 00 00	.00282 28759	.00 F9 00 00	.00379 94384
.00 3A 00 00	.00088 50097	.00 7A 00 00	.00186 15722	.00 BA 00 00	.00283 81347	.00 FA 00 00	.00381 46972
.00 3B 00 00	.00090 02685	.00 7B 00 00	.00187 68310	.00 BB 00 00	.00285 33935	.00 FB 00 00	.00382 99560
.00 3C 00 00	.00091 55273	.00 7C 00 00	.00189 20898	.00 BC 00 00	.00286 86523	.00 FC 00 00	.00384 52148
.00 3D 00 00	.00093 07861	.00 7D 00 00	.00190 73486	.00 BD 00 00	.00288 39111	.00 FD 00 00	.00386 04736
.00 3E 00 00	.00094 60449	.00 7E 00 00	.00192 26074	.00 BE 00 00	.00289 91699	.00 FE 00 00	.00387 57324
.00 3F 00 00	.00096 13037	.00 7F 00 00	.00193 78662	.00 BF 00 00	.00291 44287	.00 FF 00 00	.00389 09912

HEXADECIMAL-DECIMAL FRACTION CONVERSION (Cont.)

Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal
.00 00 00 00	.00000 00000	.00 00 40 00	.00000 38146	.00 00 80 00	.00000 76293	.00 00 C0 00	.00001 14440
.00 00 01 00	.00000 00596	.00 00 41 00	.00000 38743	.00 00 81 00	.00000 76889	.00 00 C1 00	.00001 15036
.00 00 02 00	.00000 01192	.00 00 42 00	.00000 39339	.00 00 82 00	.00000 77486	.00 00 C2 00	.00001 15633
.00 00 03 00	.00000 01788	.00 00 43 00	.00000 39935	.00 00 83 00	.00000 78082	.00 00 C3 00	.00001 16229
.00 00 04 00	.00000 02384	.00 00 44 00	.00000 40531	.00 00 84 00	.00000 78678	.00 00 C4 00	.00001 16825
.00 00 05 00	.00000 02980	.00 00 45 00	.00000 41127	.00 00 85 00	.00000 79274	.00 00 C5 00	.00001 17421
.00 00 06 00	.00000 03576	.00 00 46 00	.00000 41723	.00 00 86 00	.00000 79870	.00 00 C6 00	.00001 18017
.00 00 07 00	.00000 04172	.00 00 47 00	.00000 42319	.00 00 87 00	.00000 80466	.00 00 C7 00	.00001 18613
.00 00 08 00	.00000 04768	.00 00 48 00	.00000 42915	.00 00 88 00	.00000 81062	.00 00 C8 00	.00001 19209
.00 00 09 00	.00000 05364	.00 00 49 00	.00000 43511	.00 00 89 00	.00000 81658	.00 00 C9 00	.00001 19805
.00 00 0A 00	.00000 05960	.00 00 4A 00	.00000 44107	.00 00 8A 00	.00000 82254	.00 00 CA 00	.00001 20401
.00 00 0B 00	.00000 06556	.00 00 4B 00	.00000 44703	.00 00 8B 00	.00000 82850	.00 00 CB 00	.00001 20997
.00 00 0C 00	.00000 07152	.00 00 4C 00	.00000 45299	.00 00 8C 00	.00000 83446	.00 00 CC 00	.00001 21593
.00 00 0D 00	.00000 07748	.00 00 4D 00	.00000 45895	.00 00 8D 00	.00000 84042	.00 00 CD 00	.00001 22189
.00 00 0E 00	.00000 08344	.00 00 4E 00	.00000 46491	.00 00 8E 00	.00000 84638	.00 00 CE 00	.00001 22785
.00 00 0F 00	.00000 08940	.00 00 4F 00	.00000 47087	.00 00 8F 00	.00000 85234	.00 00 CF 00	.00001 23381
.00 00 10 00	.00000 09536	.00 00 50 00	.00000 47683	.00 00 90 00	.00000 85830	.00 00 D0 00	.00001 23977
.00 00 11 00	.00000 10132	.00 00 51 00	.00000 48279	.00 00 91 00	.00000 86426	.00 00 D1 00	.00001 24573
.00 00 12 00	.00000 10728	.00 00 52 00	.00000 48875	.00 00 92 00	.00000 87022	.00 00 D2 00	.00001 25169
.00 00 13 00	.00000 11324	.00 00 53 00	.00000 49471	.00 00 93 00	.00000 87618	.00 00 D3 00	.00001 25765
.00 00 14 00	.00000 11920	.00 00 54 00	.00000 50067	.00 00 94 00	.00000 88214	.00 00 D4 00	.00001 26361
.00 00 15 00	.00000 12516	.00 00 55 00	.00000 50663	.00 00 95 00	.00000 88810	.00 00 D5 00	.00001 26957
.00 00 16 00	.00000 13113	.00 00 56 00	.00000 51259	.00 00 96 00	.00000 89406	.00 00 D6 00	.00001 27553
.00 00 17 00	.00000 13709	.00 00 57 00	.00000 51856	.00 00 97 00	.00000 90003	.00 00 D7 00	.00001 28149
.00 00 18 00	.00000 14305	.00 00 58 00	.00000 52452	.00 00 98 00	.00000 90599	.00 00 D8 00	.00001 28746
.00 00 19 00	.00000 14901	.00 00 59 00	.00000 53048	.00 00 99 00	.00000 91195	.00 00 D9 00	.00001 29342
.00 00 1A 00	.00000 15497	.00 00 5A 00	.00000 53644	.00 00 9A 00	.00000 91791	.00 00 DA 00	.00001 29938
.00 00 1B 00	.00000 16093	.00 00 5B 00	.00000 54240	.00 00 9B 00	.00000 92387	.00 00 DB 00	.00001 30534
.00 00 1C 00	.00000 16689	.00 00 5C 00	.00000 54836	.00 00 9C 00	.00000 92983	.00 00 DC 00	.00001 31130
.00 00 1D 00	.00000 17285	.00 00 5D 00	.00000 55432	.00 00 9D 00	.00000 93579	.00 00 DD 00	.00001 31726
.00 00 1E 00	.00000 17881	.00 00 5E 00	.00000 56028	.00 00 9E 00	.00000 94175	.00 00 DE 00	.00001 32322
.00 00 1F 00	.00000 18477	.00 00 5F 00	.00000 56624	.00 00 9F 00	.00000 94771	.00 00 DF 00	.00001 32918
.00 00 20 00	.00000 19073	.00 00 60 00	.00000 57220	.00 00 A0 00	.00000 95367	.00 00 E0 00	.00001 33514
.00 00 21 00	.00000 19669	.00 00 61 00	.00000 57816	.00 00 A1 00	.00000 95963	.00 00 E1 00	.00001 34110
.00 00 22 00	.00000 20265	.00 00 62 00	.00000 58412	.00 00 A2 00	.00000 96559	.00 00 E2 00	.00001 34706
.00 00 23 00	.00000 20861	.00 00 63 00	.00000 59008	.00 00 A3 00	.00000 97155	.00 00 E3 00	.00001 35302
.00 00 24 00	.00000 21457	.00 00 64 00	.00000 59604	.00 00 A4 00	.00000 97751	.00 00 E4 00	.00001 35898
.00 00 25 00	.00000 22053	.00 00 65 00	.00000 60200	.00 00 A5 00	.00000 98347	.00 00 E5 00	.00001 36494
.00 00 26 00	.00000 22649	.00 00 66 00	.00000 60796	.00 00 A6 00	.00000 98943	.00 00 E6 00	.00001 37090
.00 00 27 00	.00000 23245	.00 00 67 00	.00000 61392	.00 00 A7 00	.00000 99539	.00 00 E7 00	.00001 37686
.00 00 28 00	.00000 23841	.00 00 68 00	.00000 61988	.00 00 A8 00	.00001 00135	.00 00 E8 00	.00001 38282
.00 00 29 00	.00000 24437	.00 00 69 00	.00000 62584	.00 00 A9 00	.00001 00731	.00 00 E9 00	.00001 38878
.00 00 2A 00	.00000 25033	.00 00 6A 00	.00000 63180	.00 00 AA 00	.00001 01327	.00 00 EA 00	.00001 39474
.00 00 2B 00	.00000 25629	.00 00 6B 00	.00000 63776	.00 00 AB 00	.00001 01923	.00 00 EB 00	.00001 40070
.00 00 2C 00	.00000 26226	.00 00 6C 00	.00000 64373	.00 00 AC 00	.00001 02519	.00 00 EC 00	.00001 40666
.00 00 2D 00	.00000 26822	.00 00 6D 00	.00000 64969	.00 00 AD 00	.00001 03116	.00 00 ED 00	.00001 41263
.00 00 2E 00	.00000 27418	.00 00 6E 00	.00000 65565	.00 00 AE 00	.00001 03712	.00 00 EE 00	.00001 41859
.00 00 2F 00	.00000 28014	.00 00 6F 00	.00000 66161	.00 00 AF 00	.00001 04308	.00 00 EF 00	.00001 42455
.00 00 30 00	.00000 28610	.00 00 70 00	.00000 66757	.00 00 B0 00	.00001 04904	.00 00 F0 00	.00001 43051
.00 00 31 00	.00000 29206	.00 00 71 00	.00000 67353	.00 00 B1 00	.00001 05500	.00 00 F1 00	.00001 43647
.00 00 32 00	.00000 29802	.00 00 72 00	.00000 67949	.00 00 B2 00	.00001 06096	.00 00 F2 00	.00001 44243
.00 00 33 00	.00000 30398	.00 00 73 00	.00000 68545	.00 00 B3 00	.00001 06692	.00 00 F3 00	.00001 44839
.00 00 34 00	.00000 30994	.00 00 74 00	.00000 69141	.00 00 B4 00	.00001 07288	.00 00 F4 00	.00001 45435
.00 00 35 00	.00000 31590	.00 00 75 00	.00000 69737	.00 00 B5 00	.00001 07884	.00 00 F5 00	.00001 46031
.00 00 36 00	.00000 32186	.00 00 76 00	.00000 70333	.00 00 B6 00	.00001 08480	.00 00 F6 00	.00001 46627
.00 00 37 00	.00000 32782	.00 00 77 00	.00000 70929	.00 00 B7 00	.00001 09076	.00 00 F7 00	.00001 47223
.00 00 38 00	.00000 33378	.00 00 78 00	.00000 71525	.00 00 B8 00	.00001 09672	.00 00 F8 00	.00001 47819
.00 00 39 00	.00000 33974	.00 00 79 00	.00000 72121	.00 00 B9 00	.00001 10268	.00 00 F9 00	.00001 48415
.00 00 3A 00	.00000 34570	.00 00 7A 00	.00000 72717	.00 00 BA 00	.00001 10864	.00 00 FA 00	.00001 49011
.00 00 3B 00	.00000 35166	.00 00 7B 00	.00000 73313	.00 00 BB 00	.00001 11460	.00 00 FB 00	.00001 49607
.00 00 3C 00	.00000 35762	.00 00 7C 00	.00000 73909	.00 00 BC 00	.00001 12056	.00 00 FC 00	.00001 50203
.00 00 3D 00	.00000 36358	.00 00 7D 00	.00000 74505	.00 00 BD 00	.00001 12652	.00 00 FD 00	.00001 50799
.00 00 3E 00	.00000 36954	.00 00 7E 00	.00000 75101	.00 00 BE 00	.00001 13248	.00 00 FE 00	.00001 51395
.00 00 3F 00	.00000 37550	.00 00 7F 00	.00000 75697	.00 00 BF 00	.00001 13844	.00 00 FF 00	.00001 51991

HEXADECIMAL-DECIMAL FRACTION CONVERSION (Cont.)

Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal	Hexadecimal	Decimal
.00 00 00 00	.00000 00000	.00 00 00 40	.00000 00149	.00 00 00 80	.00000 00298	.00 00 00 C0	.00000 00447
.00 00 00 01	.00000 00002	.00 00 00 41	.00000 00151	.00 00 00 81	.00000 00300	.00 00 00 C1	.00000 00449
.00 00 00 02	.00000 00004	.00 00 00 42	.00000 00153	.00 00 00 82	.00000 00302	.00 00 00 C2	.00000 00451
.00 00 00 03	.00000 00006	.00 00 00 43	.00000 00155	.00 00 00 83	.00000 00305	.00 00 00 C3	.00000 00454
.00 00 00 04	.00000 00009	.00 00 00 44	.00000 00158	.00 00 00 84	.00000 00307	.00 00 00 C4	.00000 00456
.00 00 00 05	.00000 00011	.00 00 00 45	.00000 00160	.00 00 00 85	.00000 00309	.00 00 00 C5	.00000 00458
.00 00 00 06	.00000 00013	.00 00 00 46	.00000 00162	.00 00 00 86	.00000 00311	.00 00 00 C6	.00000 00461
.00 00 00 07	.00000 00016	.00 00 00 47	.00000 00165	.00 00 00 87	.00000 00314	.00 00 00 C7	.00000 00463
.00 00 00 08	.00000 00018	.00 00 00 48	.00000 00167	.00 00 00 88	.00000 00316	.00 00 00 C8	.00000 00465
.00 00 00 09	.00000 00020	.00 00 00 49	.00000 00169	.00 00 00 89	.00000 00318	.00 00 00 C9	.00000 00467
.00 00 00 0A	.00000 00023	.00 00 00 4A	.00000 00172	.00 00 00 8A	.00000 00321	.00 00 00 CA	.00000 00470
.00 00 00 0B	.00000 00025	.00 00 00 4B	.00000 00174	.00 00 00 8B	.00000 00323	.00 00 00 CB	.00000 00472
.00 00 00 0C	.00000 00027	.00 00 00 4C	.00000 00176	.00 00 00 8C	.00000 00325	.00 00 00 CC	.00000 00474
.00 00 00 0D	.00000 00030	.00 00 00 4D	.00000 00179	.00 00 00 8D	.00000 00328	.00 00 00 CD	.00000 00477
.00 00 00 0E	.00000 00032	.00 00 00 4E	.00000 00181	.00 00 00 8E	.00000 00330	.00 00 00 CE	.00000 00479
.00 00 00 0F	.00000 00034	.00 00 00 4F	.00000 00183	.00 00 00 8F	.00000 00332	.00 00 00 CF	.00000 00481
.00 00 00 10	.00000 00037	.00 00 00 50	.00000 00186	.00 00 00 90	.00000 00335	.00 00 00 D0	.00000 00484
.00 00 00 11	.00000 00039	.00 00 00 51	.00000 00188	.00 00 00 91	.00000 00337	.00 00 00 D1	.00000 00486
.00 00 00 12	.00000 00041	.00 00 00 52	.00000 00190	.00 00 00 92	.00000 00339	.00 00 00 D2	.00000 00488
.00 00 00 13	.00000 00044	.00 00 00 53	.00000 00193	.00 00 00 93	.00000 00342	.00 00 00 D3	.00000 00491
.00 00 00 14	.00000 00046	.00 00 00 54	.00000 00195	.00 00 00 94	.00000 00344	.00 00 00 D4	.00000 00493
.00 00 00 15	.00000 00048	.00 00 00 55	.00000 00197	.00 00 00 95	.00000 00346	.00 00 00 D5	00000 00495
.00 00 00 16	.00000 00051	.00 00 00 56	.00000 00200	.00 00 00 96	.00000 00349	00 00 00 D6	.00000 00498
.00 00 00 17	.00000 00053	.00 00 00 57	.00000 00202	.00 00 00 97	.00000 00351	.00 00 00 D7	.00000 00500
.00 00 00 18	.00000 00055	.00 00 00 58	.00000 00204	.00 00 00 98	.00000 00353	.00 00 00 D8	.00000 00502
.00 00 00 19	.00000 00058	.00 00 00 59	.00000 00207	.00 00 00 99	.00000 00356	.00 00 00 D9	.00000 00505
.00 00 00 1A	.00000 00060	.00 00 00 5A	.00000 00209	.00 00 00 9A	.00000 00358	.00 00 00 DA	.00000 00507
.00 00 00 1B	.00000 00062	.00 00 00 5B	.00000 00211	.00 00 00 9B	.00000 00360	.00 00 00 DB	.00000 00509
.00 00 00 1C	.00000 00065	.00 00 00 5C	.00000 00214	.00 00 00 9C	.00000 00363	.00 00 00 DC	.00000 00512
.00 00 00 1D	.00000 00067	.00 00 00 5D	.00000 00216	.00 00 00 9D	.00000 00365	.00 00 00 DD	.00000 00514
.00 00 00 1E	.00000 00069	.00 00 00 5E	.00000 00218	.00 00 00 9E	.00000 00367	00 00 00 DE	.00000 00516
.00 00 00 1F	.00000 00072	.00 00 00 5F	.00000 00221	.00 00 00 9F	.00000 00370	.00 00 00 DF	.00000 00519
.00 00 00 20	.00000 00074	.00 00 00 60	.00000 00223	.00 00 00 A0	.00000 00372	.00 00 00 E0	.00000 00521
.00 00 00 21	.00000 00076	.00 00 00 61	.00000 00225	.00 00 00 A1	.00000 00374	.00 00 00 E1	.00000 00523
.00 00 00 22	.00000 00079	.00 00 00 62	.00000 00228	.00 00 00 A2	.00000 00377	.00 00 00 E2	.00000 00526
.00 00 00 23	.00000 00081	.00 00 00 63	.00000 00230	.00 00 00 A3	.00000 00379	00 00 00 E3	.00000 00528
.00 00 00 24	.00000 00083	.00 00 00 64	.00000 00232	.00 00 00 A4	.00000 00381	.00 00 00 E4	00000 00530
.00 00 00 25	.00000 00086	.00 00 00 65	.00000 00235	.00 00 00 A5	.00000 00384	.00 00 00 E5	.00000 00533
.00 00 00 26	.00000 00088	.00 00 00 66	.00000 00237	.00 00 00 A6	.00000 00386	00 00 00 E6	.00000 00535
.00 00 00 27	.00000 00090	.00 00 00 67	.00000 00239	.00 00 00 A7	.00000 00388	.00 00 00 E7	.00000 00537
.00 00 00 28	.00000 00093	.00 00 00 68	.00000 00242	.00 00 00 A8	.00000 00391	.00 00 00 E8	.00000 00540
.00 00 00 29	.00000 00095	.00 00 00 69	.00000 00244	.00 00 00 A9	.00000 00393	.00 00 00 E9	.00000 00542
.00 00 00 2A	.00000 00097	.00 00 00 6A	.00000 00246	.00 00 00 AA	.00000 00395	.00 00 00 EA	.00000 00544
.00 00 00 2B	.00000 00100	.00 00 00 6B	.00000 00249	.00 00 00 AB	.00000 00398	.00 00 00 EB	.00000 00547
.00 00 00 2C	.00000 00102	.00 00 00 6C	00000 00251	.00 00 00 AC	.00000 00400	00 00 00 EC	.00000 00549
.00 00 00 2D	.00000 00104	.00 00 00 6D	.00000 00253	.00 00 00 AD	.00000 00402	.00 00 00 ED	.00000 00551
.00 00 00 2E	.00000 00107	.00 00 00 6E	.00000 00256	.00 00 00 AE	.00000 00405	.00 00 00 EE	00000 00554
.00 00 00 2F	.00000 00109	.00 00 00 6F	.00000 00258	.00 00 00 AF	.00000 00407	00 00 00 EF	.00000 00556
.00 00 00 30	.00000 00111	.00 00 00 70	.00000 00260	.00 00 00 B0	.00000 00409	00 00 00 F0	.00000 00558
.00 00 00 31	.00000 00114	.00 00 00 71	.00000 00263	.00 00 00 B1	.00000 00412	00 00 00 F1	.00000 00561
.00 00 00 32	.00000 00116	.00 00 00 72	.00000 00265	.00 00 00 B2	.00000 00414	.00 00 00 F2	.00000 00563
.00 00 00 33	.00000 00118	.00 00 00 73	.00000 00267	.00 00 00 B3	.00000 00416	.00 00 00 F3	.00000 00565
.00 00 00 34	.00000 00121	.00 00 00 74	.00000 00270	.00 00 00 B4	.00000 00419	.00 00 00 F4	.00000 00568
.00 00 00 35	.00000 00123	.00 00 00 75	.00000 00272	.00 00 00 B5	.00000 00421	.00 00 00 F5	.00000 00570
.00 00 00 36	.00000 00125	.00 00 00 76	.00000 00274	.00 00 00 B6	.00000 00423	.00 00 00 F6	.00000 00572
.00 00 00 37	.00000 00128	.00 00 00 77	.00000 00277	.00 00 00 B7	.00000 00426	.00 00 00 F7	.00000 00575
.00 00 00 38	.00000 00130	.00 00 00 78	.00000 00279	.00 00 00 B8	.00000 00428	.00 00 00 F8	.00000 00577
.00 00 00 39	.00000 00132	.00 00 00 79	.00000 00281	.00 00 00 B9	.00000 00430	.00 00 00 F9	.00000 00579
.00 00 00 3A	.00000 00135	.00 00 00 7A	.00000 00284	.00 00 00 BA	.00000 00433	.00 00 00 FA	.00000 00582
.00 00 00 3B	.00000 00137	.00 00 00 7B	.00000 00286	.00 00 00 BB	.00000 00435	.00 00 00 FB	.00000 00584
.00 00 00 3C	.00000 00139	.00 00 00 7C	.00000 00288	.00 00 00 BC	.00000 00437	.00 00 00 FC	.00000 00586
.00 00 00 3D	.00000 00142	.00 00 00 7D	.00000 00291	.00 00 00 BD	.00000 00440	.00 00 00 FD	.00000 00589
.00 00 00 3E	.00000 00144	.00 00 00 7E	.00000 00293	.00 00 00 BE	.00000 00442	.00 00 00 FE	.00000 00591
.00 00 00 3F	.00000 00146	.00 00 00 7F	.00000 00295	.00 00 00 BF	.00000 00444	.00 00 00 FF	.00000 00593

POWERS OF TWO

2^n	n	2^{-n}
1	0	1.0
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.062 5
32	5	0.031 25
64	6	0.015 625
128	7	0.007 812 5
256	8	0.003 906 25
512	9	0.001 953 125
1 024	10	0.000 976 562 5
2 048	11	0.000 488 281 25
4 096	12	0.000 244 140 625
8 192	13	0.000 122 070 312 5
16 384	14	0.000 061 035 156 25
32 768	15	0.000 030 517 578 125
65 536	16	0.000 015 258 789 062 5
131 072	17	0.000 007 629 394 531 25
262 144	18	0.000 003 814 697 265 625
524 288	19	0.000 001 907 348 632 812 5
1 048 576	20	0.000 000 953 674 316 406 25
2 097 152	21	0.000 000 476 837 158 203 125
4 194 304	22	0.000 000 238 418 579 101 562 5
8 388 608	23	0.000 000 119 209 289 550 781 25
16 777 216	24	0.000 000 059 604 644 775 390 625
33 554 432	25	0.000 000 029 802 322 387 695 312 5
67 108 864	26	0.000 000 014 901 161 193 847 656 25
134 217 728	27	0.000 000 007 450 580 596 923 828 125
268 435 456	28	0.000 000 003 725 290 298 461 914 062 5
536 870 912	29	0.000 000 001 862 645 149 230 957 031 25
1 073 741 824	30	0.000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0.000 000 000 465 661 287 307 739 257 812 5
4 294 967 296	32	0.000 000 000 232 830 643 653 869 628 906 25
8 589 934 592	33	0.000 000 000 116 415 321 826 934 814 453 125
17 179 869 184	34	0.000 000 000 058 207 660 913 467 407 226 562 5
34 359 738 368	35	0.000 000 000 029 103 830 456 733 703 613 281 25
68 719 476 736	36	0.000 000 000 014 551 915 228 366 851 806 640 625
137 438 953 472	37	0.000 000 000 007 275 957 614 183 425 903 320 312 5
274 877 906 944	38	0.000 000 000 003 637 978 807 091 712 951 660 156 25
549 755 813 888	39	0.000 000 000 001 818 989 403 545 856 475 830 078 125
1 099 511 627 776	40	0.000 000 000 000 909 494 701 772 928 237 915 039 062 5
2 199 023 255 552	41	0.000 000 000 000 454 747 350 886 464 118 957 519 531 25
4 398 046 511 104	42	0.000 000 000 000 227 373 675 443 232 059 478 759 765 625
8 796 093 022 208	43	0.000 000 000 000 113 686 837 721 616 029 739 379 882 812 5
17 592 186 044 416	44	0.000 000 000 000 056 843 418 860 808 014 869 689 941 406 25
35 184 372 088 832	45	0.000 000 000 000 028 421 709 430 404 007 434 844 970 703 125
70 368 744 177 664	46	0.000 000 000 000 014 210 854 715 202 003 717 422 485 351 562 5
140 737 488 355 328	47	0.000 000 000 000 007 105 427 357 601 001 858 711 242 675 781 25
281 474 976 710 656	48	0.000 000 000 000 003 552 713 678 800 500 929 355 621 337 890 625
562 949 953 421 312	49	0.000 000 000 000 001 776 356 839 400 250 464 677 810 668 945 312 5
1 125 899 906 842 624	50	0.000 000 000 000 000 888 178 419 700 125 232 338 905 334 472 656 25
2 251 799 813 685 248	51	0.000 000 000 000 000 444 089 209 850 062 616 169 452 667 236 328 125
4 503 599 627 370 496	52	0.000 000 000 000 000 222 044 604 925 031 308 084 726 333 618 164 062 5
9 007 199 254 740 992	53	0.000 000 000 000 000 111 022 302 462 515 654 042 363 166 809 082 031 25
18 014 398 509 481 984	54	0.000 000 000 000 000 055 511 151 231 257 827 021 181 583 404 541 015 625
36 028 797 018 963 968	55	0.000 000 000 000 000 027 755 575 615 628 913 510 590 791 702 270 507 812 5
72 057 594 037 927 936	56	0.000 000 000 000 000 013 877 787 807 814 456 755 295 395 851 135 253 906 25
144 115 188 075 855 872	57	0.000 000 000 000 000 006 938 893 903 907 228 377 647 697 925 567 626 953 125
288 230 376 151 711 744	58	0.000 000 000 000 000 003 469 446 951 953 614 188 823 848 962 783 813 476 562 5
576 460 752 303 423 488	59	0.000 000 000 000 000 001 734 723 475 976 807 094 411 924 481 391 906 738 281 25
1 152 921 504 606 846 976	60	0.000 000 000 000 000 000 867 361 737 988 403 547 205 962 240 695 953 369 140 625
2 305 843 009 213 693 952	61	0.000 000 000 000 000 000 433 680 868 994 201 773 602 981 120 347 976 684 570 312 5
4 611 686 018 427 387 904	62	0.000 000 000 000 000 000 216 840 434 497 100 886 801 490 560 173 988 342 285 156 25
9 223 372 036 854 775 808	63	0.000 000 000 000 000 000 108 420 217 248 550 443 400 745 280 086 994 171 142 578 125

MATHEMATICAL CONSTANTS

Constant	Decimal Value	Hexadecimal Value
π	3.14159 26535 89793	3.243F 6A89
π^{-1}	0.31830 98861 83790	0.517C C1B7
$\sqrt{\pi}$	1.77245 38509 05516	1.C5BF 891C
$\ln \pi$	1.14472 98858 49400	1.250D 048F
e	2.71828 18284 59045	2.B7E1 5163
e^{-1}	0.36787 94411 71442	0.5E2D 58D9
$\sqrt{e}$	1.64872 12707 00128	1.A612 98E2
$\log_{10} e$	0.43429 44819 03252	0.6F2D EC55
$\log_2 e$	1.44269 50408 88963	1.7154 7653
γ	0.57721 56649 01533	0.93C4 67E4
$\ln \gamma$	-0.54953 93129 81645	-0.8CAE 9BC1
$\sqrt{2}$	1.41421 35623 73095	1.6A09 E668
$\ln 2$	0.69314 71805 59945	0.B172 17F8
$\log_{10} 2$	0.30102 99956 63981	0.4D10 4D42
$\sqrt{10}$	3.16227 76601 68379	3.298B 075C
$\ln 10$	2.30258 40929 94046	2.4D76 3777

INDEX

M

N

O

P

Q

R