RM61136-00A

REFERENCE MANUAL
DC 6024/4
COMPUTER SYSTEM

August, 1973

Revised
October, 1973

Datacraft Corporation
1200 N.W. 70th Street • P.O. Box 23550 • Fort Lauderdale, Florida 33307 • (305) 974-1700

LIST OF EFFECTIVE PAGES

TOTAL NUMBER OF PAGES IN THIS PUBLICATION IS 260
CONSISTING OF THE FOLLOWING:

Page No.	Change No.	Page No.	Change No.	Page No.	Change No.
Title	A				
A	A				
i thru iv	Original				
v	A				
1-1 thru 1-12	Original				
2-1 thru 2-6	Original				
2-7	A				
2-8 thru 2-29	Original				
3-1 thru 3-194	Original				
4-1 thru 4-4	Original				
4-5 & 4-6	A				
4-7 thru 4-15	Original				
A-1 thru A-3	Original				

Upon receipt of the second and subsequent changes to this technical document, personnel responsible for maintaining this publication in current status will ascertain that all previous changes have been received and incorporated. Action should be taken promptly if the publication is incomplete.

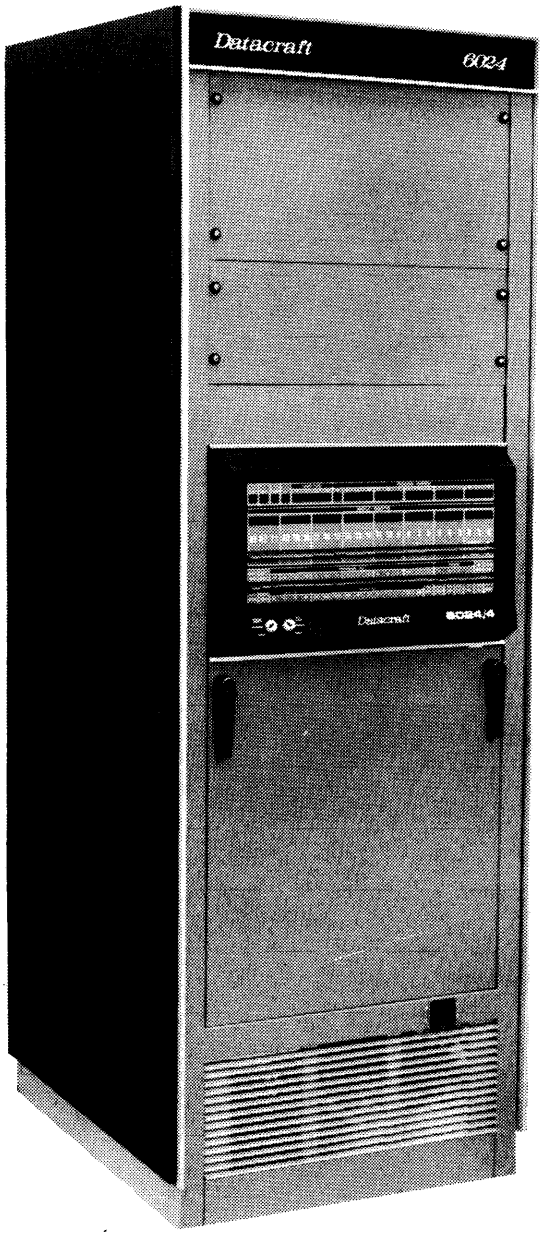
CONTENTS

CONTENTS (CONT'D.)

CONTENTS (CONT'D.)

ILLUSTRATIONS

TABLES



HT436

DC 6024/4 Computer System

SECTION I
GENERAL DESCRIPTION

1-1 INTRODUCTION

The DC 6024/4 Computer System, shown on the opposite page, is a medium-scale general purpose digital computer manufactured by the Datacraft Corporation. This machine was designed with the power requirements to match the present trend toward larger and more sophisticated input/output (I/O) equipment. The DC 6024/4 has a 750 nanosecond full cycle time, and a fixed word length of 24 bits. These and other features (found in Table 1-1 at the end of this section) provide rapid memory access, ease of programming, and a real-time control capability approaching that of a multiprocessor or large-scale system.

1-2 SYSTEM DESCRIPTION

The DC 6024/4 is the fourth model in a line of compatible (I/O and software) 24-bit digital computers. Field-proven logic design of the three other machines has been retained, but implemented with the advantages of increased medium scale integration techniques, utilization of a planer core memory, and a more efficient physical form factor.

The computer employs a magnetic core memory, a multi-access bus structure, fully parallel binary arithmetic, fully buffered I/O channels, and single address capability. Two's complement arithmetic is used on parallel, fixed-point binary operands.

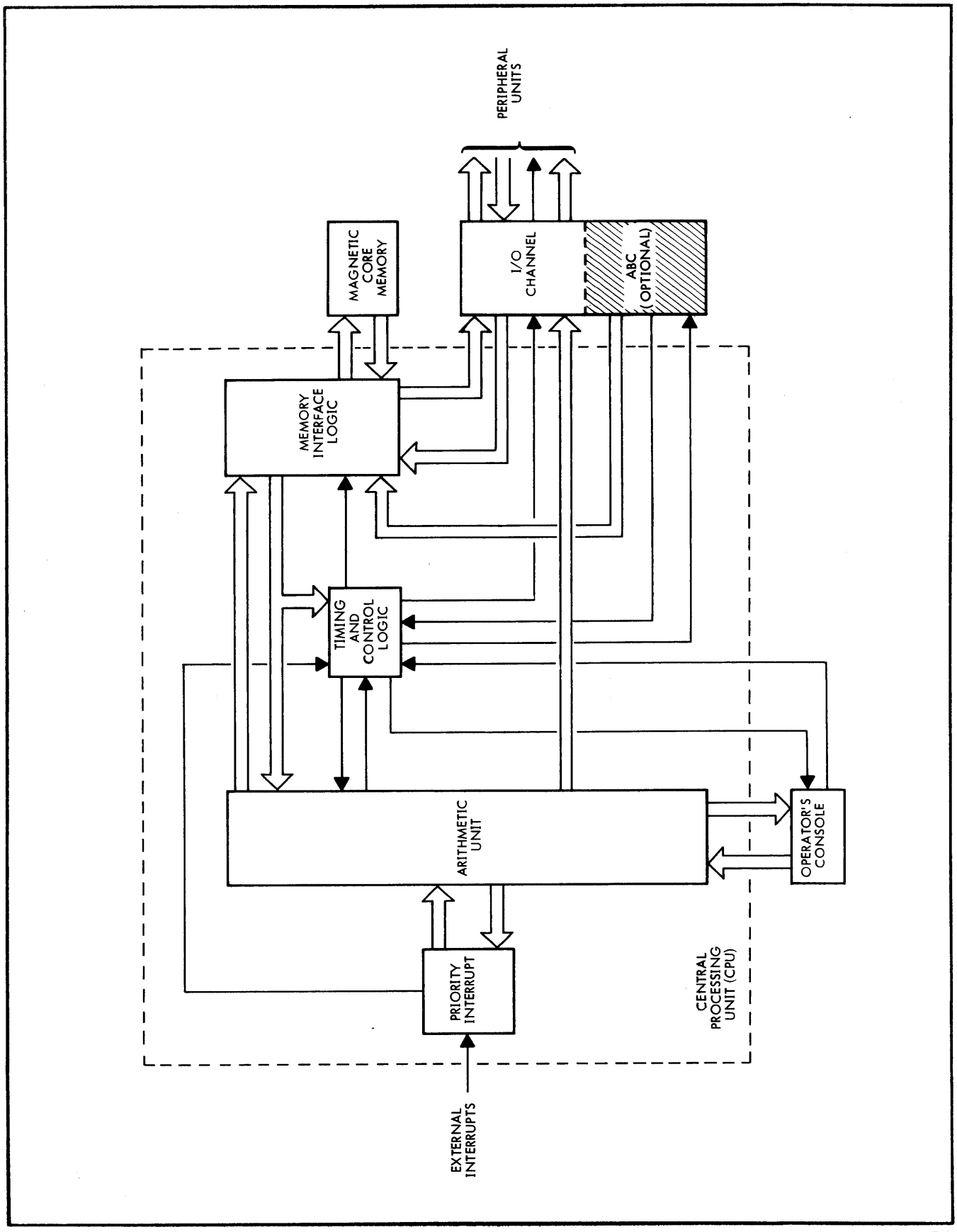
Hardware multiply/divide/square root, four priority interrupt levels, and a basic software package are furnished as part of the basic computer system. Five 24-bit general purpose registers, three of which may be used for indexing, are also included.

Memory capacity of the basic DC 6024/4 is 8,192 (or 8K) words, and may be expanded to 256K words in 8K word increments. Memory may be accessed at the word, double word, byte, or bit levels. In memory configurations employing more than 32K words, memory is divided into sectors - or "maps". The addressing technique used permits up to 32K words per map to be directly addressed and allows up to 256K words to be accessed by indirect and index references.

The instruction set includes more than 600 instructions which offer the programmer considerable flexibility and convenience. The instruction repertoire features an extensive list of operand, register-to-register and byte manipulation, a versatile set of arithmetic and logical operations, and total control over external interrupts and I/O functions.

1-3 SYSTEM ORGANIZATION

The DC 6024/4 digital computer operates on, and from, 24-bit data and instruction words and is also capable of byte manipulation and bit processing. The major functional sections of the computer are the Central Processing Unit (CPU), the magnetic core memory, and the I/O structure. Figure 1-1 is a general block diagram of the computer system.



BD60-066-972

Figure 1-1. DC 6024/4 Computer System - General Block Diagram

1-3.1 Central Processing Unit (CPU)

The CPU consists of an arithmetic unit, a control unit, interface elements for the magnetic core memory, I/O channel interface, and the operator's console.

The arithmetic unit contains a 24-bit parallel adder and a number of general and special purpose registers. Inputs to the adder are from any of the arithmetic registers via the adder input bus or from the operand register by way of the transfer bus.

The control unit, or timing and control logic, determines the exact sequences to be followed during the execution of any given instruction. This logic includes a clock generator driven by a crystal-controlled oscillator that provides precise timing for all gating pulses and commands.

Instruction words are retrieved from memory and retained in an instruction register for the duration of the instruction cycle. The control unit decodes these instruction words and provides the internal commands and timing necessary for the execution of these instructions.

The memory interface unit is comprised of the memory data register, the memory address register, the memory input bus, the memory address bus, module selection logic, and address drivers. The 24-bit (plus two parity bits) memory data register retains the data to be written into the memory as well as data read from the cores. When data is to be written into memory, the source is the parallel adder, program address register, or I/O structure. When data is read from memory, the destination is either the instruction register or an arithmetic register, or the I/O structure. The timing and control signals determine whether the memory word is an instruction word or a data word.

The memory address register is an 18-bit register that retains the address word defining the specific location into which data is to be written or from which data is to be read. For instruction words, the source is the program address register. For data words, the source is generally the parallel adder. Address input to the register may also be supplied from the ABC's or XBC's in the I/O structure. The module selection logic uses the contents of the memory address register to select one of 32 memory modules.

Communications between the CPU and the I/O structure are conducted via the channel interface circuits. Although the I/O structure conducts channel-to-unit and unit-to-channel operations independently from the CPU, channel/unit activation, function commands, and status testing are initiated under program control.

The control console allows the manual control of various system functions and provides the means for display of important system parameters. The console facilities make use of portions of the CPU timing and control logic, particularly the clock circuitry and the start-stop control logic.

The CPU is described more fully in Section II of this manual.

1-3.2 Magnetic Core Memory

The memory stores both the data and instruction words used by the computer. The storage elements are ferrite cores. Up to 32 memory modules may be included in a DC 6024/4 computer system. Each module is comprised of two identical 8,192 x 13 bit (12 data bits plus one parity bit) memory boards. These boards are installed by pairs to provide the basic 8K word

by 26-bit memory increment. In addition to the magnetic cores, each module contains the X and Y drive circuits, sense amplifiers and timing and control circuits.

Each 26-bit word constitutes a memory address. Data is stored at a given address by magnetizing the cores in either the set (storing a ONE) or reset (storing a ZERO) states. The difference between the two states is the direction (clockwise or counterclockwise) of the flux field.

A core is magnetized by coincident currents flowing in the X and Y drive lines that threaded through each core. The direction of the current determines the direction of the flux field. Selecting a core (or a group of cores comprising an address) involves using the address information from the memory interface elements in the CPU to select the proper X and Y driver circuits.

1-3.3 Input/Output Interface Structure

I/O operations within the DC 6024/4 Computer System consists of the transfer of data, commands, and status information between the CPU and peripheral devices through the I/O channel interface logic. All these operations are initiated under program control and are conducted via the facilities of the computer I/O structure.

The I/O structure employs data channels to act as buffers between the CPU and peripheral units and to control the flow of information in and out. Up to 24 data channel cards may be installed in a DC 6024/4 system and all may be active concurrently. Each channel can communicate with a maximum of 16 units, one at a time.

An I/O operation is initiated by selecting and activating a data channel and one of its assigned peripheral units through the execution of a computer Input/Output instruction. The DC 6024/4 repertoire includes six such instructions. Once activated, the channel provides complete functional control over the operation.

The specific operation may involve setting up a peripheral unit for a subsequent operation, determining the operational status of the unit or transferring data between the unit and the CPU. Data may be transferred on a single word basis (i.e., one data word per instruction) or in blocks (n words) through the use of an Automatic Block Controller (ABC) or an External Block Controller (XBC). Assuming the peripheral device is equipped with both input and output capabilities, data may be transferred in either direction between the CPU and the unit.

Communications between the I/O structure and the CPU may also be conducted on an interrupt basis through the use of interrupt logic in the channel and unit. This logic may be controlled by the program, or peripheral unit functions may be connected directly to the CPU priority interrupt system.

Further information concerning the computer I/O structure can be found in Section IV of this manual.

1-3.4 Computer Console

The DC 6024/4 Computer System operator's console includes the physical controls for starting and halting operations, entering data into memory and the various CPU registers, and selecting registers for display and/or entry. Indicators on the console provide display for the

contents of the CPU registers and memory, the status of the CPU, and other important system functions. Refer to Section II for more detailed information on the operation of the DC 6024/4 computer console.

1-4 OPTIONAL FEATURES

A wide variety of optional features is available with DC 6024/4 Computer Systems. A basic, generalized description of each option is presented in the following paragraphs and are summarized in Table 1-1 at the end of this section. A more comprehensive description of each option is provided at the end of Section II of this reference manual.

1-4.1 Program Restrict and Instruction Trap

The Program Restrict system allows areas of memory to be selected under program control and protected from unauthorized access and guarded against accidental modification. The Instruction Trap provides the means for preventing the execution of a privileged group of instructions when the system is in a restricted environment. Two registers, two executive trap interrupt trigger circuits, and the associated control logic comprise the option. Control of the option is by two instructions: Transfer Double register to Limit register (TDL) and Transfer Limit register to Double register (TLD).

1-4.2 Stall Alarm

The Stall Alarm option monitors all CPU instructions and operations in the interrupt/prohibiting category when enabled. There are specific conditions and operations within the DC 6024/4 instruction set that prohibits the recognition of external interrupts. A series of these instruction routines, when initiated could produce a situation where interrupts are excluded for a considerable period. By employing the Stall Alarm option, this situation cannot occur.

If, for example, a string of aforementioned instructions/CPU conditions has not been completed before a predetermined time elapses, an executive trap interrupt is generated. The resultant interrupt-processing routine may then examine the situation and take corrective action as needed. (This interrupt is capable of interrupting an indirect chain or a halt condition.)

1-4.3 Programmable Interval Timer

The Programmable Interval Timer option is an interval CPU timer that provides a method for regulating time segments for operating programs and recording other time intervals through the use of the CPU clock. The Interval Timer is under program control and clocks either CPU time or real time, depending on the instruction used for timer activation. In addition to its timing applications, the Interval Timer option provides the user with an additional 24-bit register that may be accessed through the standard instruction set. Characteristics are as follows:

Time Range

- Starting at full scale negative count:

  1:2.9145385 = 1.0485766 minutes

- Starting at zero:

  2:5.8291125 = 2.0971518 minutes

- Resolution:

  7.5 microseconds/count (real time)

1-4.4 Power Failure Shutdown and Restart

This option provides protection for operating programs in the event of a power failure and restoration when power levels return to normal.

1-4.5 Address Trap

The Address Trap option is an on-line debugging aid for use in applications such as breakpoint tracing. An address may be defined under program control so that when address is referenced, an interrupt will be generated at the assigned executive level. The Address Trap may be disabled under program control.

1-4.6 Automatic Block Controller (ABC)

The ABC implements and controls automatic data transfers between memory and a selected peripheral unit. The ABC provides high-speed, fully-buffered data transfer operations between these units and memory without program intervention. Addressing and block length is under program control of the CPU.

1-4.7 Channel Interrupt Generator

This interrupt option permits programmed triggering of four external interrupts. The option is a special option of the 8-bit I/O channel integrated controller card.

1-4.8 120-Hertz Clock

The 120-Hz clock option is required when the computer is operated under the Disc Monitor System (DMS) software package. Continuously generated interrupt triggers are software controlled by enabling or disabling the associated interrupt level which allows the clock to be used for various timing operations. The clock continuously transmits 120 mainframe interrupt signals per second.

1-4.9 Hardware Bootstrap Program

The hardware bootstrap operation provides automatic loading of the appropriate bootstrap program into the computer through the use of switches on the console. The bootstrap, once entered into the appropriate area in memory, will automatically load a minimum of one record from a selected peripheral unit.

1-4.10 Priority Interrupts

The priority interrupt system provides added program control of I/O operations and computations in the DC 6024/4. Recognition of special external conditions is performed on a basis of predetermined priority. The system consists of three separate interrupt groups - Group 0, 1 and 2. Group 0 contains eight optional executive traps, and Groups 1 and 2 up to 24 external interrupt levels in each group (with four levels being supplied as standard equipment). The external interrupts are available in increments of one and are totally under program control.

1-4.11 External Block Controller (XBC)

Like the ABC, the XBC provides high-speed, fully-buffered data transfer operations between these units and memory without program intervention. Control over the addressing and block length is provided by an external device.

1-4.12 Scientific Arithmetic Unit (SAU)

The SAU provides concurrent floating-point arithmetic capability independent of the CPU. Two additional registers are included with the option, an arithmetic register and a condition register. The SAU has a unique set of 47 instructions for performing double-provision, floating-point computations.

1-5 PERIPHERAL UNITS

A full line of peripheral units is available for use with the DC 6024/4 Computer System. An Automatic Block Controller (ABC) or an External Block Controller (XBC) is also available for block transfer operations requiring direct memory access. The units available are explained in detail in the User's Guide, AA61658.

- Teletypewriters
- Paper Tape Readers
- Paper Tape Punches
- Card Readers
- Card Punches
- Line Printers
- Moving Head Discs
- Fixed Head Discs
- Cartridge Discs
- Magnetic Tape Transports
- CRT Displays
- Communications Interfaces
- Digital Plotters

1-6 DC 6024/4 COMPUTER SOFTWARE SYSTEM

A comprehensive, fully-integrated, well-documented and completely checked-out program preparation, library, debugging, and utility and cross reference system is supplied with the DC 6024/4 Computer System. Two basic types of program preparation are provided: a Macro Assembler and an optional full FORTRAN compiling system. (A Resident Operating System is standard on the basic computer system.) The FORTRAN language specified for this system is the standard ANSI X 3.9 FORTRAN IV language, providing compatibility with FORTRAN IV systems supplied by other manufacturers. Several useful extensions have been included.

All of the supplied software packages are written in a modular form and are contained in a special software Reference Manual, SD61319.

The standard software documentation supplied to a customer is written in a general specification format; the basic package contains the following manuals:

- Series 6000 Macro Assembler, AA61649
- Series 6000 Resident Operating System (ROS), AA61501
- Series 6000 FORTRAN Support Library, AA61517
- Series 6000 Utility Package, AA61515
- Series 6000 Cross Reference Program, AA61514

Also available with the DC 6024/4 are optional software programs with supporting documentation as follows:

- Series 6000 FORTRAN Compiler, AA61516
- Series 6000 Disc Operation System (DOS-II), AA61568
- Series 6000 Object Time Trace, AA61544
- Series 6000 Disc Monitor System (DMS), AA61600
- Series 6000 Tape Operating System (TOS), AA61560

1-7 SYSTEM CHARACTERISTICS SUMMARY

The major operating characteristics and pertinent technical specifications of the DC 6024/4 are presented in Table 1-1.

Table 1-1. DC 6024/4 Computer System Specifications

Characteristic	Specifications		
Type	General purpose, with full parallel binary arithmetic, single address, multi-access bus structure, and fully buffered I/O channels.		
Word Length	26 bits (24 data bits plus 2 parity bits)		
Arithmetic	Fully parallel, binary, two's complement number representation, fixed-point. Hardware multiply, divide, and square root.		
Cycle Time	750 nanoseconds		
Memory:			
. Type	Planar core array; 2 memory boards per 8K words.		
. Minimum Size	8K words.		
. Maximum Size	256K words.		
. Increment	8K words.		
Addressing Modes	Direct to 32K. (Some Branch instructions address 65K words.) Indirect (data only) to 256K words. Indexed (three registers) to 65K words. Byte indexed to 192K bytes. Immediate.		
Instruction Timing: (in microseconds)		Register-to-Register	Memory Access
	Add/Subtract	0.75	1.5
	Multiply	6.0	6.0
	Divide	11.25	11.25
	Register Manipulation	0.75	1.5
	I/O	0.75	0.75
	Compare	0.75	1.50
	Square Root	5.25	----
Registers:			
. A Register	Main arithmetic register.		
. E Register	Main arithmetic register extension.		
. D Register	47-bit pseudo-register provides double precision capability (A and E registers).		
. I Register	24-bit general purpose or index register.		
. J Register	24-bit general purpose or index register.		

Table 1-1. DC 6024/4 Computer System Specifications (Cont'd.)

Characteristics	Specifications
. K Register	24-bit general purpose or index register.
. B Register	Byte register - bits 0-7 of the A register which provides byte manipulation capability. The B register is a pseudo-register.
. C Register	4-bit condition register used to indicate status of arithmetic operations and data transfers.
. P Register	16-bit program address register for storage of current instruction address.
. Instruction Register	24-bit register that holds instructions while they are being executed.
. Operand Register	24-bit register used for holding operands in arithmetic execution and for entry from the control panel.
. T Register	24-bit optional general purpose register usable as an inteval timer.
. Q Register	16-bit address query register that stores selected program address; used with Address Trap option.
. Shift Control Register	8-bit register used for certain arithmetic and shift operations.
. Memory Address Register	18-bit register used to hold memory address.
. Memory Data Register	26-bit register used to hold memory data and parity bits.
. PI Control Registers	Four 24-bit registers that contain the Armed/Disarmed, Enabled/ Disabled, Request, and Active states of the discrete interrupt levels within a specified group. 1 set for each external Group (1 & 2).
. Program Restrict Registers	Two 16-bit registers used for holding program restrict limits (with Program Restrict option). These registers hold the upper (U register) and lower (L register) limits of an area in memory that is restricted from unauthorized access.
. X Register	47-bit SAU arithmetic register contains the 39-bit double-precision floating point mantissa and the 8-bit exponent of the operand.
. W Register	8-bit pseudo-register consisting of bits 0-7 of the X register. Used to hold the SAU operand exponent.
. H Register	Single bit register for the Bit Processor.
. V Register	18-bit register used to hold a base address of an EMA. Used for the Bit Processor.

Table 1-1. DC 6024/4 Computer System Specifications (Cont'd.)

Characteristic	Specifications
Input/Output Capabilities:	
. Programmed Data Transfer	Single word to/from A Register. 8 or 24 bits. External control lines. External sense lines. Maximum of 12 channels; 16 units per channel (in basic chassis). Maximum of 24 channels; 16 units per channel (with I/O Expansion chassis).
. Automatic Data Transfer	Maximum of 12 Automatic Block Controllers (ABC) or External Block Controller (XBC) for automatic memory access.
. Input Rate	Up to 2,666,666 bytes/second (2 words every 3 cycles) on single ABC.
. Output Dump	Full memory rate (1,333,333) words/second on one ABC. (No "handshaking".)
. Output Normal	Up to 444,444 words/second on single ABC.
. Maximum Transfer Rate	1,333,333 words/second multiple ABCs, interleaved.
. ABC Setup Time	1.5 microseconds
. Block Access	Every word or every other word.
. CPU Cycles stolen per Word Transferred	1 cycle.
I/O Command Modes:	
. Normal	Normal Channel Operation command - raised by bits 5 and 4 of the OCW being zeros. Operation identical to existing DC 6024/1, DC 6024/3, and DC 6024/5 modes.
. Multiplex	CPU releases channel to master/slave pair of peripherals - raised by bits 5 and 4 of OCW being in a "0, 1" configuration.
. Offline	Command same as Multiplex, except I/O devices in the channel are turned off, allowing second CPU to share peripherals without need of peripheral switches (assumes control of I/O bus). Bits 5 and 4 of OCW are in a "1, 0" configuration.
. Reset	Command operates the same as a Normal command, but resets the channel out of either Multiplex or Offline mode. (Channel restored on-line, unit selected.) Bits 5 and 4 of OCW are in a "1, 1" configuration.

Table 1-1. DC 6024/4 Computer System Specifications (Cont'd.)

Characteristic	Specifications
Priority Interrupt System	Maximum of 8 executive traps and 48 external interrupts available (4 standard to 48 in 1-level increments). All external interrupt levels (program controlled) may be individually enabled, disabled, armed or disarmed; dedicated memory address for each level.
Operating Environments: . Temperature . Relative Humidity	 15° to 45°C (59°F to 113°F) 20% to 90% (non-condensing)
Power Requirements: . Line	 115 VAC, ±10%, 47 to 63 Hz, 15 amps, (basic), single phase
Power Failure Protection: . Standard . Optional	 Memory Data Retention. Power Failure Shutdown and Restart
CPU Options	Automatic Block Controller (ABC) External Block Controller (XBC) External Interrupts Hardware Bootstrap (up to 4 units) Programmable Interval Timer Power Failure Shutdown and Restart Program Restrict/Instruction Trap 120-Hz Clock Stall Alarm Address Trap Channel Interrupt Generator Scientific Arithmetic Unit (SAU)

SECTION II
CENTRAL PROCESSING UNIT

2-1 GENERAL DESCRIPTION

The DC 6024/4 Central Processor Unit (CPU) is a single address, fully parallel, 24-bit word-oriented, stored-program unit. Operations performed by the CPU include data transfers, arithmetic computation, and logical manipulation. These operations are defined by instructions retrieved from memory. The specified operation is performed on single word, double word, or byte operands stored in memory or contained in one of the CPU's five registers. Data word formats, as defined by both hardware and software are illustrated in Figure 2-1.

In addition to the five general registers, the CPU contains an Arithmetic section to perform the actual computation on operands and a Control section that retrieves and decodes instructions from memory and directs the functional processes of the system. The CPU also contains an Operator's Console to allow manual control of the computer. The functional capability of the CPU can be expanded by the addition of the Program Restrict System, Stall Alarm, Interval Timer, Address Trap, and other options.

2-2 CENTRAL PROCESSING UNIT REGISTERS

The CPU contains arithmetic and control registers and two pseudo registers. Six optional registers are available that provide additional control and arithmetic functions.

2-2. 1 Registers Available to the Programmer

The programmable and otherwise accessible registers of the CPU are illustrated on Figure 2-2 and described in the following paragraphs.

A. A Register

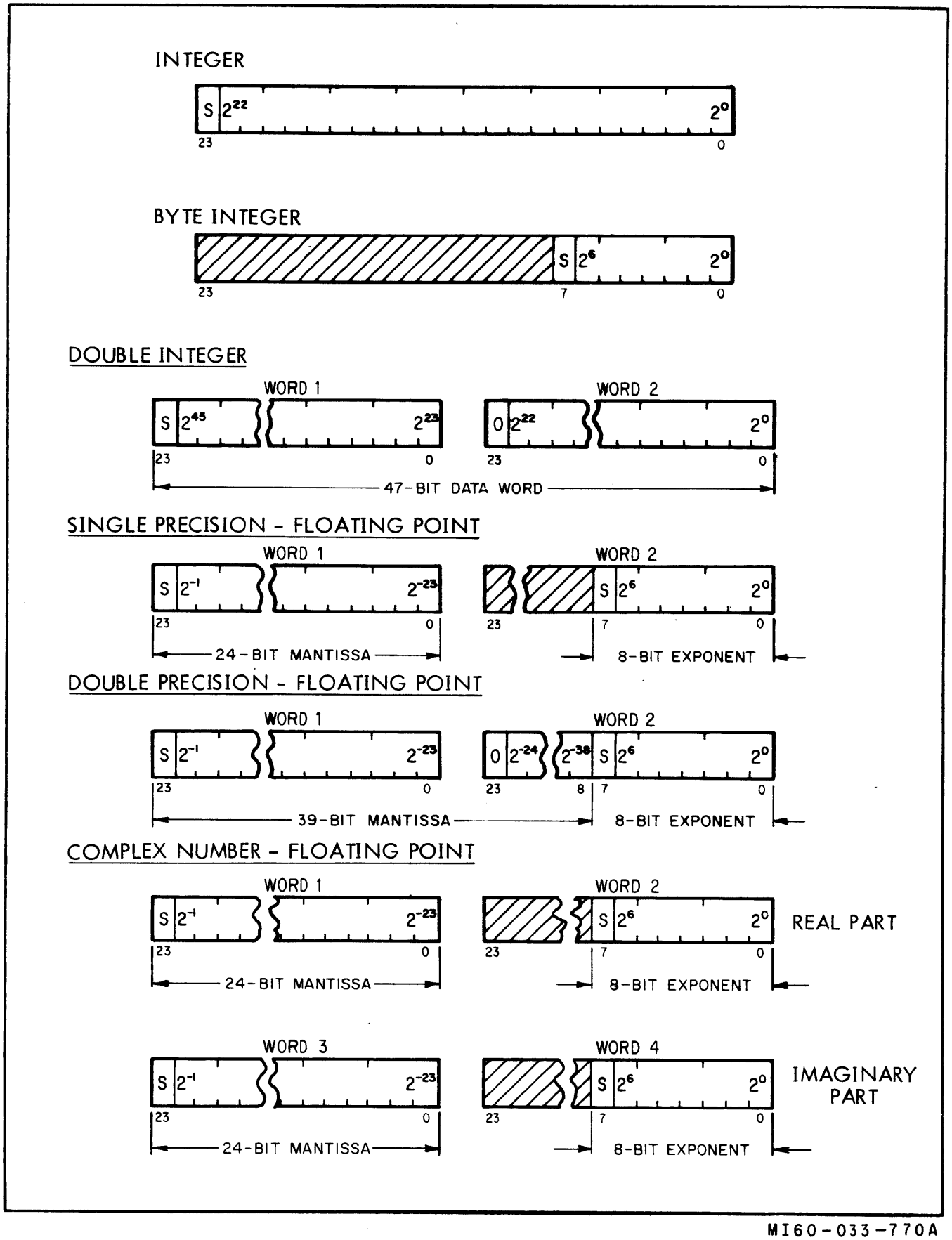
The 24-bit A register is the main arithmetic register. It contains complete arithmetic and shift capabilities and is used as the input/output communication register.

B. E Register

The E register is a 24-bit general register. It is also used as an extension of the A register for additional shifting and arithmetic capability.

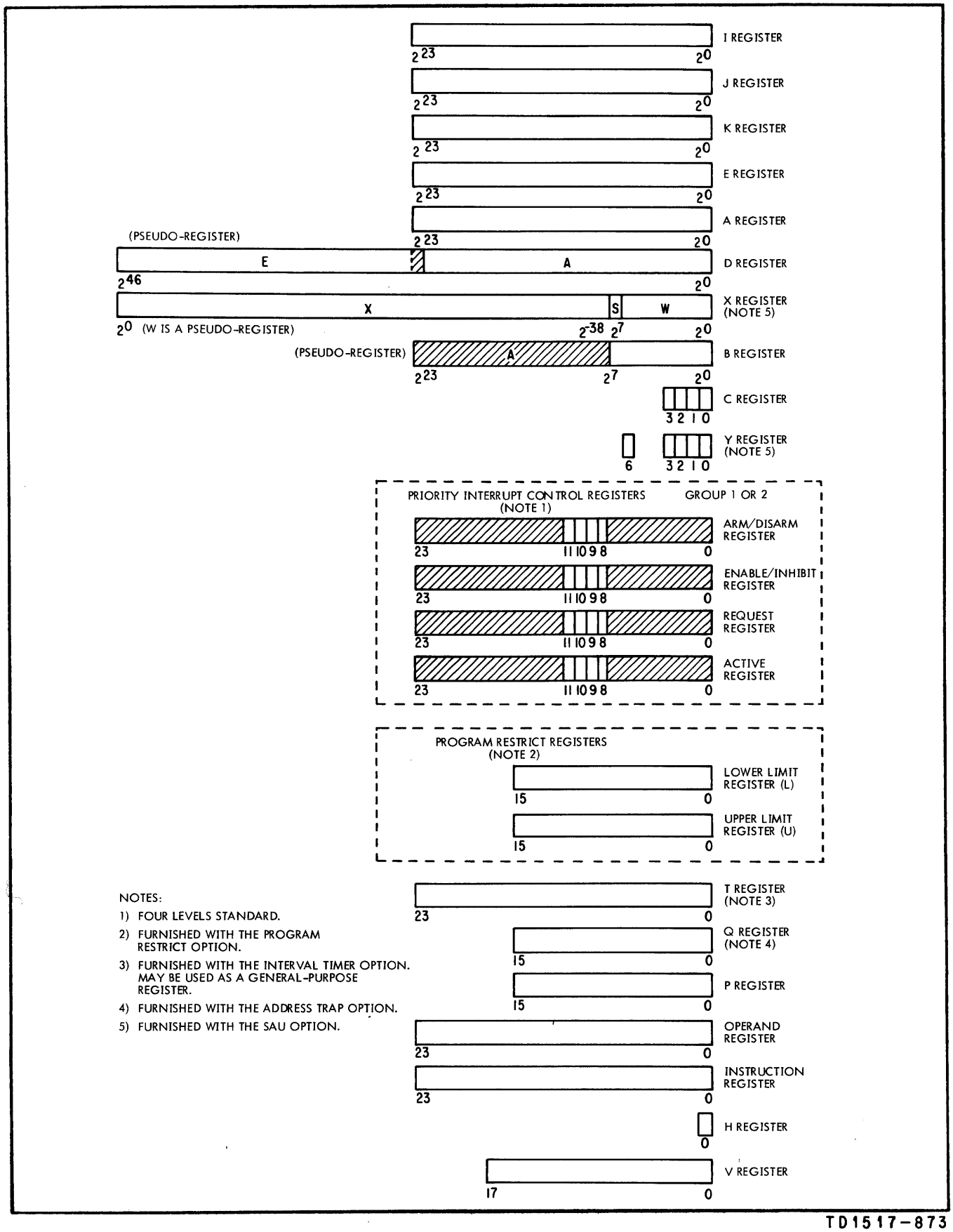
C. D Register

The D (Double) register is a 47-bit pseudo-register consisting of the A and E registers. It provides double-precision capability. The A register contains the 23 least significant bits and the E register contains the 24 most significant bits.



MI60-033-770A

Figure 2-1. Data Word Formats



TD1517-873

Figure 2-2. Central Processing Unit Registers

D. I, J, and K Registers

The I, J, and K registers are 24-bit general registers that may be used as index registers for address modification.

E. B Register

The B (Byte) register is a pseudo-register consisting of bits 0-7 of the A register. It provides byte manipulation capability.

F. C Register

The C (Condition) register is a 4-bit register that stores and displays the results of specific operations. The following conditions are displayed:

Bit 0 - Indicates Overflow (logic 1) or No Overflow (logic 0)
Bit 1 - Indicates Negative (logic 1) or Not Negative (Logic 0)
Bit 2 - Indicates Zero (logic 1) or Not Zero (logic 0)
Bit 3 - Indicates Positive (logic 1) or Not Positive (logic 0)

NOTE

Bits 1, 2, and 3 are mutually exclusive only when displaying the results of an operation.

G. H Register

A single bit register associated with the Bit Processor feature. This register permits bit manipulation.

H. V Register

The 18-bit V register is provided with the Bit Processor feature. This register is used to store a base address of an effective memory location containing the bit to be manipulated.

2-2.2 Registers Not Available to the Programmer

Several registers within the CPU are not directly available to the programmer. These registers are described in the following paragraphs and illustrated in Figure 2-2.

A. P Register

The 16-bit P (Program address) register contains the address in memory from which the current instruction was taken for processing. A maximum of 65,536 memory locations can be accessed via the P register.

B. Operand Register

The 24-bit Operand register temporarily retains operands being transferred between major sections of the CPU. The Operand register is not directly programmable, but functions as a console entry register.

C. Instruction Register

The 24-bit Instruction register contains instruction words after they have been accessed from memory and while they are being decoded. The instruction register is not programmable.

2-2.3 Optional Registers

Up to 14 additional registers can be added to the CPU with the incorporation of available options. These registers are described in the following paragraphs and are illustrated in Figure 2-2.

A. Priority Interrupt Control Registers

The Priority Interrupt Control registers are four 24-bit registers that contain the Armed/Disarmed, and Enable/Disable, Active and Request states of the discrete interrupt levels within a specified group (1 or 2).

B. Program Restrict Registers

Two 16-bit registers are provided with the Program Restrict Option. These registers hold the upper (U register) and lower (L register) limits to an area in memory that is restricted from unauthorized access.

C. T Register

The T (Timer) register is a 24-bit general register included with the Interval Timer option. This register may also be used as additional 24-bit general register that is accessed via the standard instruction set.

D. Q Register

The 16-bit Q (address Query) register is supplied with the Address Trap option. It stores a selected program address and when that address is reached in the program, a interrupt is generated.

E. X Register

The X register is the arithmetic register of the SAU. The register holds the 39-bit double-precision floating-point mantissa and the 8-bit exponent of the operand.

F. W Register

The W Register is a pseudo-register consisting of bits 0-7 of the X register and is used to hold the exponent of the operand.

G. Y Register

The Y register is the SAU condition register and operates in the same manner as the CPU C register (Paragraph 2-2. 1F). Another bit associated with but not a part of the Y register indicates the status of the SAU interrupt. This bit is associated with bit position 6 in the mainframe's A register. When this bit is a logic 1, it indicates the interrupt is enabled.

2-3 OPERATOR'S CONSOLE

2-3. 1 General Description

The DC 6024/4 operator's console includes the physical controls for starting and halting operation, entering data in memory and the various CPU registers, and selecting registers for display and/or entry. Indicators on the console provide display for the contents of the CPU registers and memory, the status of the CPU and other important functions.

For information concerning single and multi-address addressing, basic program execution, and program (bootstrap) loading - both manual and hardware, refer to the DC 6024/4 Operator's Manual, OM61398

2-3. 2 Controls and Indicators

Figure 2-3 illustrates the controls and indicators for the operator's console. Table 2-1 provides a description of the function of each. The call-outs on the figure are used in the table for reference.

2-3. 3 PROGRAM HALT Function

Normal operation of the PROGRAM HALT function involves: a) selecting an address on the PROGRAM HALT switches; and b) raising the PH ENA switch. When the program subsequently references the selected address. The CPU will automatically halt.

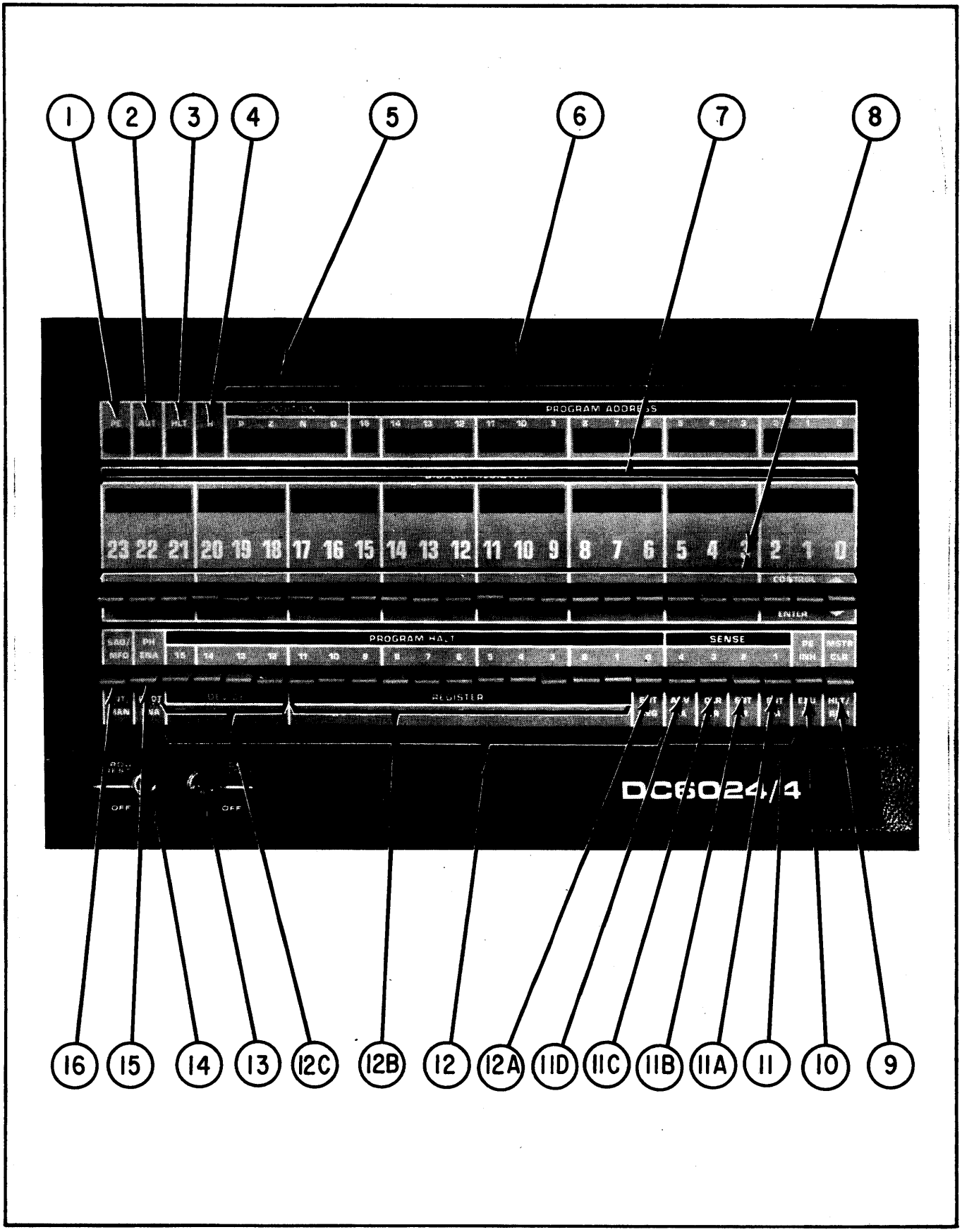
However, special techniques are required when using the PROGRAM HALT feature in conjunction with any Branch instruction. The basic rule is:

> If the PROGRAM HALT switches are set to an address that is referenced by a Branch instruction, the halt will not take place at the selected address.

For example, if the following coding appeared in the program.

Program Location	Instruction, Operand
100	BUC '200

and the PROGRAM HALT switches were also set to '200, the program would branch to location '200 and continue execution. In order to have the program halt occur at location '200, the PROGRAM HALT switches must be set to the location immediately following that of the Branch; e. g., '101. In this situation, the program would branch to location '200 and then halt.



MI 1521-1073

Figure 2-3. Operator's Console, DC 6024/4 Digital Computer

Table 2-1. Controls and Indicators, DC 6024/4 Operator's Console

Reference (Figure 2-3)	Position	Control/Indicator	Function
1		PE Indicator	Indicates the detection of a memory Parity Error.
2		AUT Indicator	Indicates the CPU is in the AUTomatic mode.
3		HLT Indicator	Indicates a CPU HALT condition.
4		H Indicator (Bit Processor)	Displays the contents of the H register. (Illuminated = ONE).
5		CONDITION Indicators P, Z, N, & O	Displays the contents of the CONDITION register. P = Positive, Z = Zero, N = Negative, and O = Overflow. (Illuminated = condition true).
6		PROGRAM ADDRESS Indicators	Displays the memory location from which the current instruction will be called for processing; i.e., the contents of the Program Counter. (Illuminated = ONE).
7		DISPLAY REGISTER Indicators	Displays the contents of the register selected by the REGISTER switches (see Reference 12B). (Illuminated = ONE).
8	UP	Switch Register = ONE	Retains a 24-bit operand for subsequent transfer to a CPU register upon execution of appropriate instruction. (TSI, TSJ, TSK, TSE, TSA, or TST).
	Center	Switch Register = ZERO	
	Down*	ENTER Switches	Enter ONEs in the corresponding bit positions of the Operand Register (OR).
9	Up	MSTR CLR Switch	MaSTeR CLeaR - Clears all CPU registers and initializes all control logic.
	Center	- - - - - - -	No function.
	Down	HLT/RUN switch	Toggles the CPU into a HaLT or RUN condition. HLT - Stops all processing and activates all console controls (unless disables by CP LOCK). RUN - Starts program execution and disables all console controls except HLT/RUN and MSTR CLR.

*Momentary Contact.

Table 2-1. Controls and Indicators, DC 6024/4 Operator's Console (Cont'd.)

Reference (Figure 2-3)	Position	Control/Indicator	Position
10	Up	PE INH switch	Parity Error INHibit - Inhibits halting when a Parity Error is detected.
	Center	- - - -	No function.
	Down*	EXU IR switch	EXecUte Instruction Register - Executes the instruction in the Instruction Register (IR) and Operand Register (OR), and loads the next instruction in the IR and OR.
11	Up	SENSE Switches = ONE	May be set and queried under program control (QSS instruction).
	Center	SENSE Switches = ZERO	
11A	Down*	ENT PA switch	ENTer Program Address - Replaces the contents of the PROGRAM ADDRESS register with the contents of bits 15-0 of the Operand Register (OR); accesses the instruction at the new PROGRAM ADDRESS; and loads this instruction into the Instruction Register (IR) and Operand Register (OR).
11B	Down*	ENT IR Switch	ENTer Instruction Register - Replaces the contents of the Instruction Register (IR) with the contents of the Operand Register (OR).
11C	Down *	CLR OR switch	CLeaR Operand Register - Clears the Operand Register (i. e., resets all OR bits to ZERO).
11D	Down*	ADV PA Switch	ADVance Program Address - Increments the PROGRAM ADDRESS by one.
12	Up	PROGRAM HALT = ONE	Selects the Program Address for the PROGRAM HALT function (See Reference 15-PH ENA).
	Center	PROGRAM HALT = ZERO	
12A	Down*	ENT REG switch	MANual mode - Replaces the contents of the selected register(s) or memory location with the contents of the Operand Register (OR). AUTomatic mode - Replaces the contents of the selected memory location with the contents of the OR. The PROGRAM ADDRESS is automatically advanced and the OR is cleared.

*Momentary Contact.

Table 2-1. Controls and Indicators, DC 6024/4 Operator's Console (Cont'd.)

Reference (Figure 2-3)	Position	Control/Indicator	Function
12B	Down	REGISTER Switches	Each switch selects a specific register for display and/or entry. When several switches are set down at one time, data is entered into all the selected registers. The DISPLAY REGISTER may indicate the logical-OR of the selected register or may give an erroreous indication, depending on the combination of registers selected. OR - Operand Register M - Memory location specified by the PROGRAM ADDRESS. I - I Register J - J Register K - K Register E - E Register A - A Register Y/T - Y Register or T Register as selected by SAU/MFO switch. X_R/U - X_R Register or U Register as selected by SAU/MFO switch. X_L/L - X_L Register or L Register as selected by SAU/MFO switch. IR - Instruction Register.
12C	Down	DEVICE Switches	Selects the peripheral device bootstrap routine that will be entered into memory by the BOOT ENA switch (See Reference 15) MT = Magnetic Tape, D = Disc, CR = Card Reader, PT = Paper Tape.
13	On	CP LOCK key switch	Enables the Control Panel lock and de-activates all console switches.
14	On	PROG REST key switch	Enables the PROGram RESTrict system (Optional).
15	Up	PH ENA switch	Program Halt ENAble - Causes the CPU to halt when the PROGRAM ADDRESS matches the address selected on the PROGRAM HALT switches. (See Reference 12).
	Center	- - - -	No function.
	Down*	BOOT ENA switch	Causes the hardware bootstrap routine selected by the DEVICE switches to be loaded into memory and executed.

*Momentary Contact

Table 2-1. Controls and Indicators, DC 6024/4 Operator's Console (Cont'd.)

Reference (Figure 2-3)	Position	Control/Indicator	Function
16	Up	SAU Switch	Selects the Scientific Arithmetic Unit (SAU) X_L, X_R and Y registers for display. The specific register is selected by the X_L/L, X_R/U or Y/T switch (see Reference 12B).
	Center	MFO switch	Selects the Mainframe Option registers (L, U, and T) for display and/or entry. The specific register is selected by the X_L/L, X_R/U or Y/T switch (see Reference 12B).
	Down*	AUT/MAN switch	Toggles the operating mode into an AUTomatic or MANual condition. (See Reference 12A - ENT REG switch).

*Momentary Contact.

2-4 BIT PROCESSOR FEATURE

The bit processor consists of the single-bit H register, a 16-bit V register (Base register); and the associated control logic. The bit processor provided the capability to selectively change a bit in memory, store a bit in memory, or test a bit.

2-4.1 Operational Description

The 18-bit V register is loaded with a base address which specifies a memory location to be manipulated. This is accomplished by transferring a 18-bit memory address from the K register. The instruction word further defines the memory location, the specific bit, and the operation to be performed.

After the operation is performed on the selected bit, the results are displayed in the Condition register. Refer to Section III, Paragraph 3-17.

2-4.2 Program Control

Two instruction formats are associated with bit processor operations. The first specifies a displacement (bits 0-7 of the instruction word) from the base address to specify the location to be used. Five bits in the instruction words (bits 8-12) select the specific bit to be processed. The remaining (bits 13-23) contain the OP CODE which specifies the operation to be performed.

The second format is not used for specifying a bit in memory but for bit movement or transfer operations where a specific bit is not required. Bits 12-23 contain a 12-bit OP CODE and bits 0-11 are undefined.

2-5 PRIORITY INTERRUPT SYSTEM

The Priority Interrupt system for the DC 6024/4 Computer System provides added program control of I/O operations and computations, internal CPU operations, and immediate recognition of special external conditions on the basis of predetermined priority. Receipt of an interrupt trigger by the Priority Interrupt system diverts the normal program flow to an interrupt subroutine so that the interrupt can be serviced. Figure 2-4 is a general block diagram of the priority interrupt system.

The Priority Interrupt system consists of up to three separate interrupt groups (0, 1 and 2). Group 0 is reserved for internal CPU functions and is comprised of up to eight optional executive trap interrupt levels. Groups 1 and 2 are reserved for external interrupt functions. Four interrupt levels on group 1 are standard. The optional external interrupts are available in increments of one and are totally under program control (with a maximum of 48).

2-5.1 Operational Description

Operation of the Priority Interrupt system is initiated by an interrupt trigger. Both the external interrupts and executive traps operate in a similar manner. The primary difference is that in order for an external interrupt to become active, it must be armed and enabled. Executive traps are hardwired in an armed and enabled state. The armed and enabled status of the system is stored in the control latches. Two latches are required for each external interrupt. One latch stores the armed/disarmed state of the interrupt and the other latch stores the enable/disable state.

2-5.2 Executive Traps (Group 0) Operation

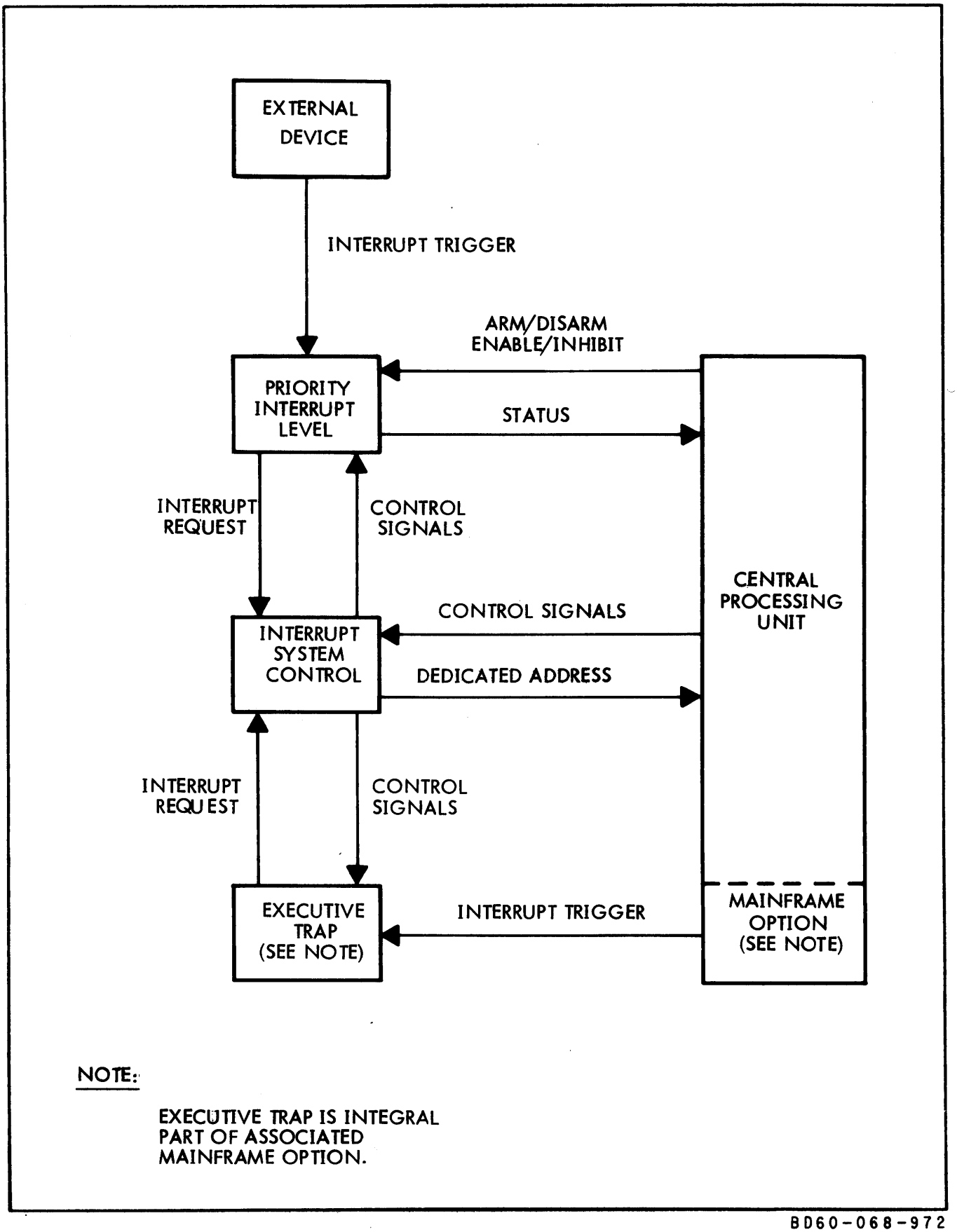
The optional executive traps consist of eight hardwired, armed and enabled interrupt levels. They are associated exclusively with available mainframe options and are therefore permanently assigned.

NOTE

All executive trap levels (and external interrupt levels) are assigned a unique priority number. These interrupt levels descend in order of priority from Group 0, Level 0 (executive trap 0) to Group 2, Level 23. Group 0 has priority over Group 1; Level 0 has priority over Level 23.

Once triggered by an internal source, the executive trap becomes active immediately if no higher priority interrupt is active also. Interrupt level assignments on the executive traps are as follows:

Level	Function	
0	Power Down	Power Failure Shutdown/Restart
1	Power Up	
2	Program Restrict	
3	Instruction Trap	
4	Stall Alarm	
5	Internal Timer	
6	SAU	
7	Address Trap	



BD60-068-972

Figure 2-4. Priority Interrupt System, General Block Diagram

2-5.3 External Interrupts Operation (Groups 1 and 2)

Since the need for and application of interrupts are so varied, maximum flexibility is obtained by providing a number of interrupts that can be program controlled and not permanently assigned. The DC 6024/4 provides a total of 48 external interrupt levels, with four levels (8-11) standard, and the remaining 44 optional.

All interrupt triggers occur asynchronous with respect to CPU timing. Synchronization of the interrupt with CPU operations occurs within the interrupt logic. Whereas an interrupt can become active at anytime, it will only be acknowledged after the execution of the current instruction is completed.

Each external interrupt operates in three distinct states: Inactive, Waiting, and Active. In the Inactive state, the interrupt level has not received an external trigger. When the trigger is received, the interrupt level is set to the Waiting state if that level is armed. If the interrupt level is enabled and no higher priority interrupts are active, the interrupt level becomes Active (state).

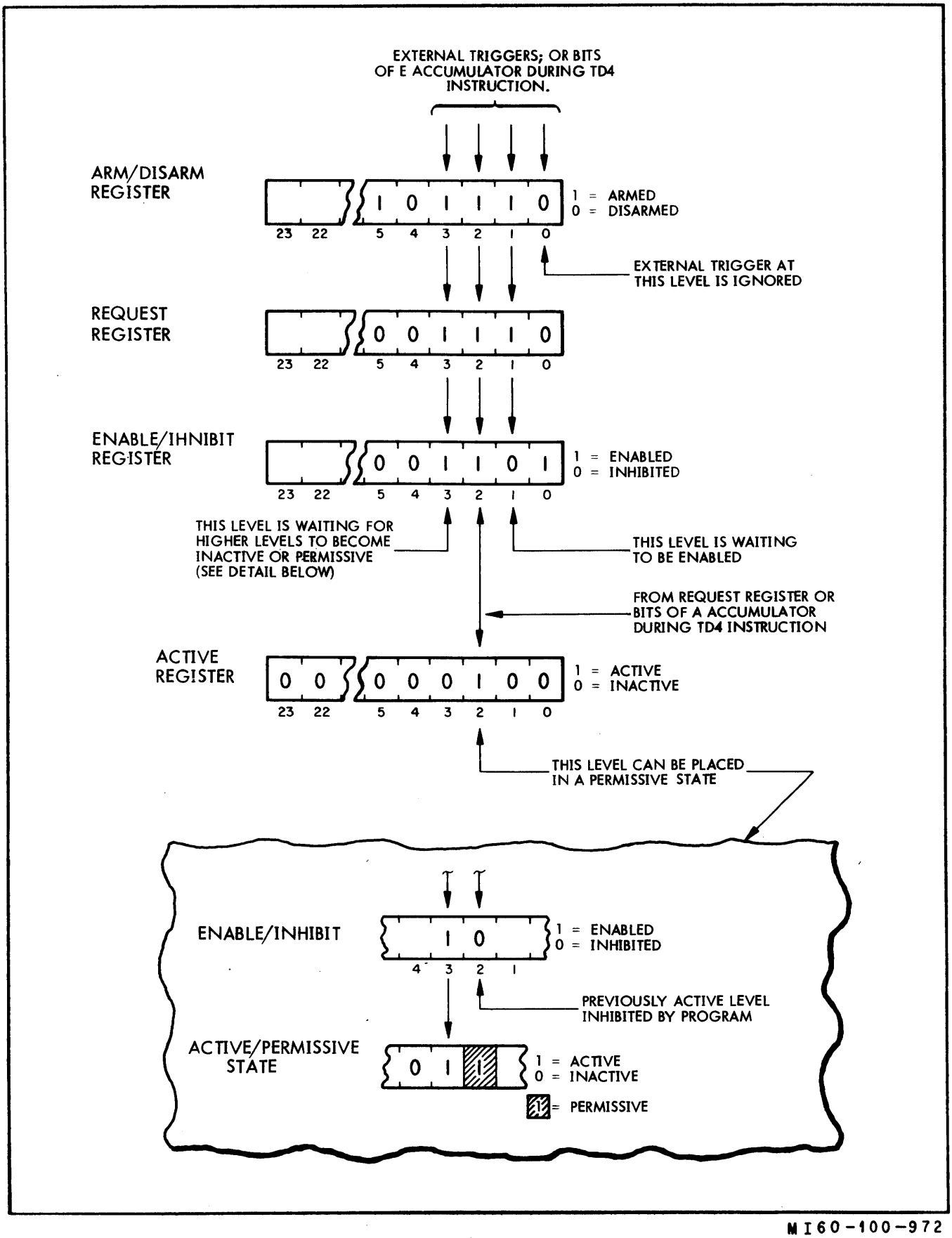
Four 24-bit registers are associated with each external interrupt group (1 and 2). The first of these is the arm/disarm register, which contains the armed/disarmed status of each interrupt level in the group. The second register is the enable/inhibit register, which contains the enable/inhibited status of each interrupt level within the group. The third register is the request register, which (depending on the status of the arm/disarm register) will be triggered by an external source or by a TD4 or TD5 instruction. The fourth register is the active register, which contains the active/inactive status of each interrupt level within the group.

Interrupt levels are triggered by an external source or by a TD4 or TD5 instruction. The contents of the arm/disarm and enable/inhibit registers determine whether the triggered interrupt will be ignored, held, or allowed to become active. A given interrupt level must be both armed and enabled before it becomes active. If the interrupt level is disarmed, the trigger will be ignored. If an interrupt level is armed but inhibited (i.e., not enabled), it is held in the waiting state until such time as it is enabled under program control. A level can become active by a TD4 instruction.

Once an interrupt request (i.e., external trigger) passes both the arm/disarm and enable/inhibit tests, it becomes active as soon as the instruction being executed is completed assuming no higher priority level is active and that external interrupts are not being held by a Hold eXternal Interrupts (HXI) instruction.

Several instructions are privileged. Should an interrupt occur during the execution of one of these instruction, it will not be allowed to become active until the completion of the instruction following the privileged instruction. The privileged instructions are: BSL (Branch and Save return-long), HTI, HTJ, HTK, (Hold interrupts and Transfer I, J, or K register to memory, RXI (Release eXternal Interrupts, EXN (EXecute Memory), TRM (Transfer Registers to Memory), TMR (Transfer Memory to registers), USP (Update Stack Pointer), TDG (Transfer Double Register to Group), UA1 or UA2 (Unitarily Arm Group 1 or 2), UD1 or UD2 (Unitarily Disarm Group 1 or 2), UE1 or UE2 (Unitarily Enable Group 1 or 2), UI1 or UI2 (Unitarily Inhibit Group 1 or 2).

After an interrupt becomes active, it is possible to inhibit that level under program control and permit an armed and enabled lower priority level to become active. If, for example, active level 2 (in Figure 2-5) is inhibited by the program, level 3 becomes active immediately. When the inhibited level 2 is reenabled, it is restored to its normal priority. This permissive state of the interrupts permits on-line modification of the priority interrupt structure.



MI60-100-972

Figure 2-5. DC 6024/4 External Interrupt Control

Program control of external interrupts is afforded by several instructions in the DC 6024/4 repertoire. Individual levels can be selectively (unitarily) armed, disarmed, enabled, or inhibited or an entire group of interrupts can be simultaneously controlled. The HXI and RXI (Hold and Release eXternal Interrupts) instructions can prohibit and restore the activation of any external interrupt. The HXI instruction will positively prohibit the activation of any external interrupt (other than the levels currently active) regardless of that interrupt's armed/enabled status. This prohibition would ensure that an interrupt processing routine would not be interrupted. This Hold condition can only be released by an RXI instruction. For a detailed description of all priority interrupt instructions associated with the DC 6024/4 refer to the appropriate portion of Section III of this reference manual.

2-5. 4 Interrupt Processing Considerations

Each interrupt level has a memory location assigned for its exclusive use. This applies to both the Executive Traps (Group 0) and the external interrupts (Groups 1 and 2). Table 2-2 lists the dedicated memory locations for the interrupt system.

Table 2-2. DC 6024/4 Dedicated Memory Locations

Addresses (Octal)	Function
60-67	EXECUTIVE TRAPS (GROUP 0)
70-117	EXTERNAL INTERRUPTS (Group 1)
120-147	EXTERNAL INTERRUPTS (Group 2)

Table 2-3 lists a typical designation for the group and level of interrupt, its associated octal address, and its suggested assignment. For example, Group 1/Level 8 is octal address 100 and is assigned to the input of the console teletypewriter. Note that the executive trap assignments are predetermined and are not reassignable.

Although any instruction may be stored in the dedicated location, the Branch and Save return-Long (BSL) instruction provides a means for returning to the point of interrupt plus restoring the original machine status (i. e., the contents of the Condition register). Thus, the BSL instruction normally provides the entry for the interrupt processing routine.

A means of exit from the interrupt routine is the Branch and Reset interrupt - Long (BRL) instruction. Normally, the BRL instruction would make use of an indirect reference (*) to the address previously referenced by the BSL instruction upon entering the routine. If this is done, the Condition register is restored to its original contents (at the time the interrupt occurred).

The BRL instruction resets the highest active (not in permissive state) trap or external interrupt level provided that external interrupts are not being "held" (HXI instruction). Active traps are always reset by the BRL instruction. Active interrupts can only be reset by the BRL instruction, a TD1 or TD2 instruction, or by master clearing the CPU.

NOTE

A BRL instruction will not reset an interrupt that is in the permissive state.

Table 2-3. DC 6024/4 Computer System Interrupt Assignments

Group	Level	Address (Octal)	Function	
0	0	60	Power Down	Executive Traps
0	1	61	Power Up	
0	2	62	Program Restrict	
0	3	63	Instruction Trap	
0	4	64	Stall Alarm	
0	5	65	Interval Timer	
0	6	66	SAU	
0	7	67	Address Trap	
1	0-7	70-77	Not Assigned	External Interrupts
1	8	100	Console Typewriter (Input)	
1	9	101	Console Typewriter (Output)	
1	10	102	Card Reader (Input)	
1	11	103	Card Punch (Output)	
1	12	104	Paper Tape Reader (Input)	
1	13	105	Paper Tape Punch (Output)	
1	14	106	Disc Status	
1	15	107	Magnetic Tape Status	
1	16-23	110-117	Not Assigned	
2	0-23	120-147	Not Assigned	

2-6 PROGRAM RESTRICT/INSTRUCTION TRAP OPTION

The Program Restrict/Instruction Trap option allows areas of memory to be defined and selected under program control and protected from unauthorized access. The instruction trap provides a means for preventing the execution of a privileged group of instructions. Two registers (16-bit), two executive trap interrupt trigger circuits, and the associated control logic comprise this option. Control is maintained by two instructions: Transfer Double register to Limit register (TDL) and Transfer Limit register to Double register (TLD).

2-6.1 Operational Description

The CPU operates with the program restrict system enabled or disabled depending on the position of the PROGram RESTrict key switch. When the restrict system is disabled, all memory is accessible.

When the restrict system is enabled, the computer operates in three distinct modes as established under program control. The three program restrict modes are defined below:

a. Unrestricted (Mode 0) - Programs working in this mode may access and alter any location in memory.

b. Restricted/Privileged (Mode 1) - Programs operating in this mode may access and load from any location in memory; however, Mode 1 programs may not alter the contents of, or transfer control to, any memory location outside the specified limits.

c. Restricted/Unprivileged (Mode 2) - Programs operating in this mode may not reference, in any manner, any memory location outside the specified limits.

Once established, the restrict mode is maintained by two flags operating concurrently. The mode flags can be set to one of three significant states to establish Mode 0, 1 or 2.

Control over the restrict system is maintained by the program restricted flag (PRF). The PRF operates under the conditions and in the manner outlined below:

a. The PRF may be set only when the restrict system is enabled (i.e., when the PROGram RESTrict key switch is on).

b. When in Mode 1 or 2, the PRF is set when an instruction transfers control into the restricted area of memory.

c. When MaSTer CLeaR is depressed, the PRF is set and Mode 2 is established automatically. MaSTer CLeaR also clears the limit registers. Once a violation is made, the only way of recovering manually is by disabling the PROGram RESTrict mode switch with the key and then depressing the MaSTer CLeaR switch.

d. Priority interrupts reset (turn off) the PRF, rendering the mode flags ineffective until control is returned to the restricted area and the PRF is set again.

e. When the PRF is set, executive trap interrupt 02 (Group Q Level 2) occurs if a restrict violation takes place, except when operating in the Halt mode. (A

restrict violation consists of any attempt to violate the conditions established by the modes.) The one exception to this is the Branch and Link Unrestricted (BLU) instruction. The BLU instruction has been implemented as an executive call to allow restricted programs to communicate directly with the resident operating system without being trapped. The BLU instruction resets the PRF and transfers control unconditionally to the address specified by the instruction.

f. When the restrict system is enabled and in Mode 1 or 2, the console HaLT switch can be activated only if the PRF is set. (This prohibits halting the machine during an interrupt routine.) If the CPU is operating in Mode 0, the only method of halting operation is to disable the restrict system (turn the key switch off)and activate the HaLT switch.

g. If the CPU is being controlled manually (Halt mode) and the restrict switch is in Mode 1 or 2 (PRF set), violations will be treated as follows:

(1) If the violating instruction is a Branch or an attempt to transfer an operand into memory, it will be treated as a No OPeration (NOP) instruction.

(2) If the violation attempts to transfer an operand from memory, all ZEROs will be retrieved and loaded into the destination register (Mode 2 only).

(3) If the normal advance of the Program Address causes the violation, all ZEROs will be loaded into the Operand Register. The Instruction Register will remain the same.

(4) If an attempt is made to enter an address into the Program Address Register that is outside the bounds specified by the Limit Registers, the Program Address Register will not be loaded, and ZEROS will be retrieved from memory.

2-6.2 Program Control

The restricted area of memory is defined by two special registers, the Lower limit (L) and Upper limit (U) registers. Each register retains a 16-bit address that defines one limit of the restricted area.

Two instructions, Transfer Double to Limit registers (TDL) and Transfer Limit registers to Double (TLD), are provided for operating the limit registers. The limits are defined by executing a TDL instruction where D = E and A, bits E_{15}-E_0 specify the lower limit and A_{15}-A_0 specify the upper limit. The TDL instruction also establishes the restrict mode by setting the mode flags with bits A_{21} and A_{22}. The bit configuration determines which mode will be established as shown in Table 2-4.

Table 2-4. Mode Bit Configuration

Bit A_{22}	Bit A_{21}	Mode
0	0	0
0	1	1
1	0	2
1	1	0

NOTE

If an attempt is made to execute the TDL instruction while in Mode 1 or 2, the instruction trap is activated.

The TLD instruction provides a method for saving the contents of the limit registers plus the status of the mode flags. The contents of the L register are transferred to bits E_{15} - E_0 and the contents of the U register are transferred to A_{15} - A_0. The mode flag bit configuration is retained in bits A_{22} and A_{21}.

2-6.3 Instruction Trap

The instruction trap is enabled and disabled by the PROGram RESTrict key switch. When the PRF is off, the instruction trap is inhibited.

If an attempt is made to execute any of the instructions listed below with the PRF on, an interrupt occurs at Group 0, Level 03. The interrupt routine may then examine the trapped instruction and determine what action is to be taken. The affected instructions are:

a. Halt (HLT)
b. Output Data Word (OCW)
c. Input Data Word (IDW)
d. Output Command Word (OCW)
e. Input Status Word (ISW)
f. Output Address Word (OAW)
g. Input Address Word (IAW)
h. Hold eXternal Interrupts (RXI)
i. Release eXternal Interrupts (RXI)
j. Unitarily Arm group 1 interrupts (UA1)
k. Unitarily Arm group 2 interrupts (UA2)
l. Unitarily Disarm group 1 interrupts (UD1)
m. Unitarily Disarm group 2 interrupts (UD2)
n. Unitarily Enable group 1 interrupts (UE1)
o. Unitarily Enable group 2 interrupts (UE2)
p. Unitarily Inhibit group 1 interrupts (UI1)
q. Unitarily Inhibit group 2 interrupts (UI2)
r. Transfer Double to group 1 (TD1)
s. Transfer Double to group 2 (TD2)
t. Transfer Double to group 1 (TD4)
u. Transfer Double to group 2 (TD5)
v. Transfer Double to Limit Registers (TDL)

2-7 STALL ALARM OPTION

The Stall Alarm option is enabled and disabled by a toggle switch, which is located on the mainframe option circuit card. With the stall alarm disabled, normal CPU operations take place. When the option is enabled, a 128-cycle counter is activated by decoding any of a group of designated instructions or by the presence of certain operating conditions. The counter is incremented once each machine cycle until another instruction (not the designated group) is decoded or the special operating condition is removed. If, after 128 cycles, neither event has occurred, an executive trap is generated at Group 0, Level 4.

The following instructions and/or CPU conditions will activate the stall alarm counter:

a. Unitarily Arm group 1 interrupts (UA1)
b. Unitarily Arm group 2 interrupts (UA2)
c. Unitarily Disarm group 1 interrupts (UD1)
d. Unitarily Disarm group 2 interrupts (UD2)
e. Unitarily Enable group 1 interrupts (UE1)
f. Unitarily Enable group 2 interrupts (UE2)
g. Unitarily Inhibit group 1 interrupts (UI1)
h. Unitarily Inhibit group 2 interrupts (UI2)
i. Transfer Double to group 1 interrupts (TD1)
j. Transfer Double to group 2 interrupts (TD2)
k. Transfer Double to group 1 interrupts (TD4)
l. Transfer Double to group 2 interrupts (TD5)
m. Update Stack Pointer (USP)
n. Branch and Save return - Long (BSL)
o. Hold eXternal Interrupts (HXI)
p. Hold interrupts and Transfer I, J, or K to memory (HTI, HTJ, or HTK)
q. EXecute Memory (EXM)
r. Release eXternal Interrupts (RXI)
s. Transfer Registers to Memory (TRM)
t. Transfer Memory to Registers (TMR)
u. A halt condition
v. An indirect memory cycle

Each of the preceding instructions prohibits the recognition of external interrupts for a period of one cycle following completion of the instruction. Executing a number of these instructions in series will lock out external interrupts for the entire series. Multilevel indirect addressing can produce a similar effect, since the instruction must satisfy all address references before completion. A halt condition — whether as a result of a programmed halt, operator action, or memory parity error — also prohibits external interrupt recognition by the CPU.

If a power failure occurs, the stall alarm becomes disabled. However, when power is restored, the stall alarm is re-enabled and operations continue in o normal routine.

With the exception of an EXM instruction or an indirect cycle, the monitored operation is allowed to complete its sequence before the executive trap assumes control. An EXM chain — where an EXM instruction references another EXM which, in turn, specifies a third, etc. — has the same overall effect as an indirect chain in that all references must be completed before the sequence is complete. Therefore, if an EXM or indirect cycle is in process when the executive trap is generated, the stall alarm logic automatically terminates the sequence. If an Automatic Block Controller (ABC) is transferring data into memory when the executive trap interrupt is generated, the current cycle is completed before termination occurs and the trap takes control. If a halt condition is in effect when the executive trap interrupt is generated, the stall alarm logic automatically forces the CPU into a run mode.

2-8 PROGRAMMABLE INTERVAL TIMER OPTION

The Programmable Interval Timer option consists of a 24-bit register (T register) and the associated control logic. The timer measures elapsed CPU time or real time as desired. A time interval may be preset in the interval timer under program control. When not employed as an interval timer, the T register can be used as a general purpose register.

2-8.1 Operational Description

Generally, a negative count is loaded into the T register, which is then incremented every 10 machine cycles in all timing operations. The timer operates in one of two modes (processor time or clock time) depending on which timing signal is used to strobe the timer (T register).

In the processor time mode, the CPU's "T" timing pulses are used as the clock for the interval timer. This mode measures elapsed time of selected computations or operations within the CPU. When this mode is selected, the interval timer operates only during the time the CPU is running, and not during an ABC cycle steal.

The clock time mode measures real time by counting down the time present in the T register. A selected time interval is preset in the T register under program control and the timer is counted down by using the CPU's "C" timing pulses as a strobe. In this mode, the interval timer is operable anytime power is applied to the computer.

When used as an interval timer, an executive trap interrupt is generated when the preset time is counted down to zero. The executive trap used is interrupt Group 0, Level 5. Generation of this interrupt does not inhibit interval timer operation.

2-8.2 Program Control

Interval timer operation is controlled with three instructions; Hold Interval Timer (HIT), Release Processor Time (RPT), or Release Clock Time (RCT). The HIT instruction inhibits interval timer operation in that a timing operation is prevented from starting or is halted while in progress. The timer remains halted until it is released by a RPT or RCT instruction. The RPT instruction enables the interval timer in the processor time mode to allow the timer to measure elapsed CPU time. Time used for Automatic Block Controller operation is not measured as CPU time. Clock time mode operation is enabled with the RCT instruction. The interval timer counts CPU "C" timing pulses continuously.

Executive trap 05 interrupt is triggered when the contents of the T register are decremented to zero. Generation of the interrupt does not inhibit the timer.

When used as a general purpose register, the T register recognizes the various arithmetic and transfer instructions associated with T register operation.

2-9 ADDRESS TRAP OPTION

The Address Trap option queries each referenced memory address and compares it to the address preset in the 16-bit query register. Coincidence between the two addresses causes the executive trap interrupt to be generated. Besides the Q register, the option includes a 16-bit address comparator, an interrupt trigger circuit, and associated control logic.

2-9.1 Operational Description

The Query register is loaded with a 16-bit address. This address corresponds to a memory location that possesses some significance to the program. Each time this memory address is referenced, executive trap interrupt 07 is generated to inform the computer.

Operation of the address trap can be enabled or disabled with bit 23 of the 24-bit memory word used to load the Query Register. If bit 23 is set (ONE), the address trap is disabled. The address trap is enabled when bit 23 is reset or ZERO. Disabling the trap inhibits the executive trap interrupt.

2-9.2 Program Control

Control of the address trap is with the Transfer Memory to Query register (TMQ) instruction. This instruction transfers the contents of the selected memory location from memory via the Operand register to the Query register. This location contains a 16-bit address word and the enable/disable control bit (bit 23). With the Query register loaded and the address trap enabled, an interrupt is generated each time a reference is made to the memory location corresponding to the address stored in the Query register. If it is desired that a reference to the selected memory location be recognized only once, a second TMQ instruction should be executed following the first interrupt with bit 23 set to ONE. This disables the address trap.

2-10 HARDWARE BOOTSTRAP OPTION

The Hardware Bootstrap option automatically stores in memory a bootstrap program that permits a more complex program to be stored. Any program can be stored as long as it is in bootstrap format; however, the most common application is to load a loader program which allows other programs, operating systems, diagnostics, or other data to be stored in selected memory locations. Four sources (bootstrap options) are selectable for transferring a program to memory via a selected peripheral device: paper tape, disc, card reader, or magnetic tape. The operation of the bootstrap option is controlled at the operator's console. The specific device may be selected by actuating the appropriate DEVICE switch and the BOOT ENA switch, which stores the program into memory. A description of the bootstrap operation and individual bootstrap programs are documented in the DC 6024/4 Operator's Manual, OM61398.

2-11 POWER FAILURE SHUTDOWN/RESTART OPTION

This option saves the operating program in the event of a power failure and the subsequent restart when power levels return to normal. The power failure shutdown/restart circuits inhibit the start of a new memory cycle during the power interruption.

2-11.1 Operational Description

The Power Failure Shutdown/Restart option monitors the input AC power source for the computer power supplies for fluctuations in power level. A decrease in the AC voltage level of approximately 10 percent causes an executive trap 00 interrupt (Group 0, Level 0) to be generated.

A Master Clear signal is generated 1 millisecond after the executive trap is generated to complete the shutdown process. This period allows the program time to save the data being processed.

When the AC power level returns to 95 percent of its nominal level, another Master Clear signal is generated to begin the restore process. The restart signal is generated approximately 5 µsec after Master Clear. The restart signal generates an interrupt on executive trap 01 (Group 0, Level 1).

2-12 120-HERTZ CLOCK OPTION

This option continuously transmits 120 mainframe interrupt signals per second. This interrupt signal is controlled completely by enabling (or disabling) the assigned mainframe interrupt level. The first mainframe interrupt following an Enable signal will occur in less than 1/120 of a second because the clock never stops transmitting signals; however, all subsequent interrupts will be precisely 1/120 seconds apart.

The accuracy in using this optional clock is a function of the user interrupt routine logic. For example, if the clock is used to update a "time-in-seconds" counter by adding one every 120 interrupts, the "current time" at any given query will be accurate within 1 second. If, however, the counter is updated each interrupt (1/120 second) and divided by 120 when "current time" is queried, the accuracy will be within 1/120 of 1 second.

A simple example of coding, where the clock is assigned to priority interrupt Group 1, Level 22, is as follows:

```
*.....                          . Initialize Clock Routine
INITCT      TMA = B22           . (A) = Bit 22
            TME = B22           . (E) = Bit 22
            UA1                 . Arm L22/G1
            UE2                 . Enable L22/G1
            TZM  CLOCK T        . Zero Clock Time
            BUC  Ø, J
*.....                          . Interrupt Routine
CLOCK IR    ***                 . Enter
            AUM CLOCK T         . Increment Clock Time
            BRL* CLOCK IR       . Restore C Register and Exit
*.....                          . Current Time Routine
CTIME       TMA CLOCK T
            ESA
            DVO  120
            BUC Ø, J
                                . Return: (A) = Seconds
                                          (E) = Remainder
```

2-13 AUTOMATIC BLOCK CONTROLLER (ABC) OPTION

The ABC implements and controls automatic data transfers between memory and a selected peripheral unit. The ABC provides high-speed, fully-buffered data transfer operations (24-bit words) between these units and computer memory without program intervention. Automatic block transfers are accomplished by an I/O channel circuit card, which contains the timing and

control logic that processes the block transfer. As stated, a block transfer is initiated under program control and proceeds from that point until the transfer is terminated. When an ABC channel is used in a DC 6024/4 Computer System configuration, no special considerations in the unit controller have to be made. The unit will respond with the same signals that are used in all other I/O channels.

2-13.1 Operational Description

The ABC is initialized by loading the address of the first of a pair of block transfer parameters in the Transfer Address Register (TAR) with an Output Address Word (OAW) instruction. The two parameter word formats are shown in Figure 2-6.

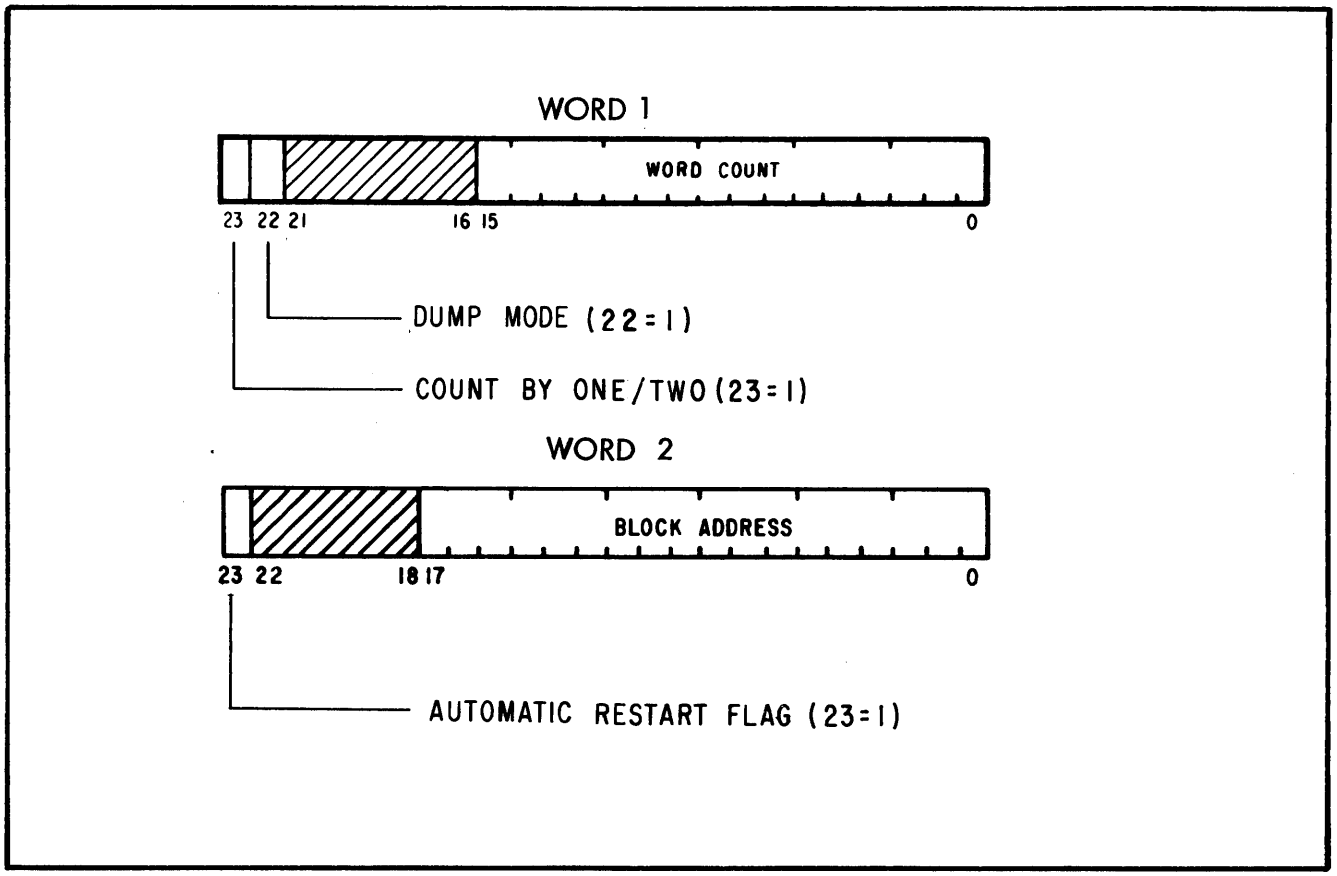


Figure 2-6. ABC Parameter Word Formats

The first parameter word (Word 1) is the word count (65K maximum) or number of sequential words to be transferred. The word count is contained in bits 0-15. If bit 23 of Word 1 is set, the address contained in the TAR during the block transfer will be incremented by 2 for each word transferred to or from the unit, except after the last word transferred which will be incremented by 1. If bit 23 of Word 1 is set, the contents of the computer memory will be dumped into the connected unit at full memory cycle rate without normal "handshaking" between the I/O channel and unit. The Output Data Here (ODH) signal will be a pulse to indicate to the unit that data is available. The Output Data Accept (ODA) signal will not be used in this mode of operation. The computer will be completely "locked" out of memory once the ABC has started its block and until the sequence is terminated.

The second parameter (Word 2) is the starting address (256K maximum) of the block to be transferred. The starting address is contained in bits 0-17. If bit 23 of Word 2 is set (Automatic Restart Flag), a new transfer sequence is initiated without program intervention.

During each output transfer, the ABC logic will access data at the current memory location stored in the TAR. The output buffer is loaded and data is transferred to the unit controller over the standard I/O channel lines. The TAR is incremented and the WCR decremented. After the unit controller responds, the transfer sequence is automatically repeated.

During each input transfer, the ABC logic will transfer data to the current memory location stored in the TAR whenever the unit controller generates the proper signal. The TAR is incremented and the WCR decremented. The sequence is repeated as soon as the unit controller responds.

The OCW instruction, which initiates the block transfer. can also be used to terminate the transfer sequence as a result of the override bit (bit 11) being set in the OCW format.

The OCW instruction selects and activates the channel and one of its assigned units. This instruction contains a nine-bit code which specifies the unit/channel combination. Five bits comprise the channel code, while four bits designate the unit. The four bit unit code is strobed to the unit controller by the I/O channel timing and control logic.

2-13.2 ABC Functional Summary

After the ABC is initialized, words are transferred between the selected peripheral unit and computer memory via the ABC. During the data word transfer, the Transfer Address Register (TAR) is incremented and the Word Count Register (WCR) is decremented. Subsequent data words are transferred by repetition of this cycle.

When the word count is decremented to zero, the block transfer is terminated unless bit 23 of the address register is set. When bit 23 is set, a new block transfer is automatically initiated by the ABC without program intervention.

2-14 EXTERNAL BLOCK CONTROLLER (XBC) OPTION

An XBC channel is similar to an ABC in that both types of channels provide for automatic data transfer between memory and a selected peripheral device. The essential difference between the ABC and XBC channels is that the latter receives all address inputs from an external source. Programming considerations and applications are largely controlled by the external devices with which the XBC is operating. All interrupts are also a function of the external device.

2-14.1 XBC Functional Summary

All data transfers are preceded by a request. The XBC itself has a higher priority than a unit. A unit, having a data word ready for transfer into memory or ready to receive data from memory, will generate a request. No other inputs (address, data, etc.) will be allowed to become active until the XBC responds to the request.

Upon sampling all eight unit request lines, and selecting the unit assigned the highest priority, the XBC will place the unit code of the selected device on the Word Transfer Unit

Code Bus where it will be presented to all eight units. (The unit code placed on the bus will be the one previously specified by an instruction.) The designated unit will respond by providing an ACCPT level. If the designated unit is currently involved in a data transfer request, it must respond to the command before the request will be honored and the transfer allowed to take place. A separate 3-bit Status code is sent to all units, and the designated unit will respond by placing its status (8-bits, maximum) on the appropriate input lines to the XBC.

2-15 CHANNEL INTERRUPT GENERATOR OPTION

This option permits programmed triggering of four external interrupt pulses via an Output Address Word (OAW) instruction. The four interrupt pulses are generated in an 8-bit I/O channel integrated controller card.

The four pulses generated are 625 nanoseconds in width (positive "true") and may be routed for use as interrupts in another CPU or any external peripheral unit.

The Channel Interrupt Generator responds to the particular OAW instruction with the proper channel code. The four least significant bits (0 - 3) of the A register, during the OAW instruction, will trigger the pulse from the generator. The pulse remains at the "true" level for the 625-nanoseconds cycle and then is restored to the "false" state. There is no response to the mainframe C (condition) register during the execution on this OAW, i. e., if the C register were tested, it would indicate "not zero".

In summary, if an interrupt pulse is to be generated, the following coding could be applied:

```
TOA   B0, B1, B2, B3    (Unitary bits; one for each interrupt pulse line.)
OAW   CU
```

2-16 SCIENTIFIC ARITHMETIC UNIT (SAU) OPTION

2-16.1 Operational Description

The SAU provides double precision floating-point arithmetic capability for the DC 6024/4 computers. The SAU has 47 instructions in its repetoire. Of these 47 instructions, 19 permit concurrent CPU/SAU operations. An executive trap (Group 0, Level 6) is provided with the SAU. The trap is used to detect overflow/underflow conditions.

The SAU has three registers in addition to memory, available to the programmer:

The X Register (mantissa and exponent)

The W Register (exponent)

The Y Register (condition)

The W Register is the least significant 8 bits of the X register and can be modified independently of the mantissa register as required.

The Y register is set according to the result of the SAU instruction being executed, i. e., positive, zero, negative and/or overflow. General instructions set the overflow bit as well as one or more of the remaining bits in an error occurs.

The A register and D register of the CPU are available to theSAU via several instructions. Operations with these CPU registers do not affect the CPU condition register (C register).

2-16.2 Programming Considerations

The SAU and CPU will operate concurrently for one or more machine cycles, depending on the SAU instruction being executed. In order to take advantage of the available cycles, CPU and SAU instructions must be intermixed.

If the instruction sequence contains several consecutive SAU instructions, the CPU will wait for the SAU, i. e., if an SAU instruction is in progress and another SAU instruction follows it, the CPU must wait until the second instruction has started (or completed, if there is no time-sharing) before executing any non-SAU instruction. For example the sequence:

```
TMX   A  (3 cycles)
MMX   B  (7 cycles, 3 to initiate, 4 for concurrent operation)
DMX   C  (16 cycles, 3 to initiate, 13 for concurrent operation)
TXM   D  (3 cycles)
```

will not permit execution of non-SAU instructions for 29 cycles. Note, however, that there are 17 cycles available in the sequence for execution non-SAU instructions. The following sequence makes use of the available CPU cycles:

```
TMA   A
MMX   B
TMD   X    (3 cycles)  }
TOI   30   (1 cycle)   }  4 Concurrent MMX cycles
DMX   C
AMD   Y    (3 cycles)  }
TMD   Z    (3 cycles)  }
AMI   J    (2 cycles)  }
TIA        (1 cycle)   }  13 Concurrent DMX cycles
NII        (1 cycle)   }
AAM   K    (3 cycles)  }
TXM   D
```

2-16.3 Interrupt Considerations

The executive trap (Group 0, Level 6) provided with the SAU is used to detect overflow/underflow conditions resulting from the execution of SAU instructions. The trap is controlled by two SAU instructions and the Hold/Release external interrupt instructions of the CPU.

The SAU instructions which control the trap are:

HSI - Hold SAU overflow interrupt

RSI - Release SAU overflow interrupt.

The trap, when enabled, is triggered by the overflow bit (bit 0) of the SAU condition register (Y register). In order to start SAU operation and enable the trap the following sequence may be used:

```
TOY    0        or      TMX    OPERAND
RSI                     RSI
```

Either sequence clears the overflow bit and prevents an extraneous interrupt.

When the SAU trap is enabled and an overflow occurs, the SAU is set to a busy condition, preventing the execution of any other SAU instruction except a HSI. This allows the program to determine the location of the SAU instruction which caused the overflow. The SAU interrupt processing routine must execute an HSI as its first SAU instruction. Prior to exiting the service routine, bit 0 of the Y register must be cleared and an RSI instruction performed to rearm the SAU trap. A typical entry/exit sequence is:

```
SAUPI   ***
        HSI
        .
        .
        .
        TOY   Ø
        RSI
        BRL*  SAUPI
```

Note that an overflow can be caused by program control with the sequence:

```
HSI
RSI
TOY   1
```

It should be noted that the content of the Program Counter at the time of the interrupt does not necessarily have a direct relation to the location of the SAU instruction which caused the overflow. This is due to the concurrent processing capability, the occurrence of other interrupts, the execution of the HXI/RXI instructions and the way in which the SAU and CPU instructions are intermixed.

When it is a requirement to know exactly where the instruction causing the overflow is located, careful coding is mandatory if the concurrent operation capability is to be used. It is recommended that, in cases where overflow is likely, the SAU instructions be written consecutively to simplify the procedure for finding which SAU instruction caused the overflow.

SECTION III
INSTRUCTION SET

3-1 INTRODUCTION

The DC 6024/4 Computer System instruction set consists of several functional groups or families of instructions. Among these are: Arithmetic; Branch; Compare; Input/Output; Logical; Shift; Transfer; etc. Each group, in turn, is composed of individual instructions that perform specific functions.

Through the application of the instruction set, the programmer has access to each memory location and major register in the CPU. In addition, the instruction set provides for the alteration and control of program flow, manipulation and modification (arithmetic and logical) of data, servicing of priority interrupts and control of I/O operations.

3-2 INSTRUCTION FORMATS

Each instruction is decoded from a 24-bit memory word. The instruction word bits define the operation to be performed and the manner in which it is to be performed. All instruction formats contain an operation code (OP CODE) that defines the general process that is to be undertaken (Add, Subtract, Interchange, etc.). The OP CODE may contain either six or 12 bits.

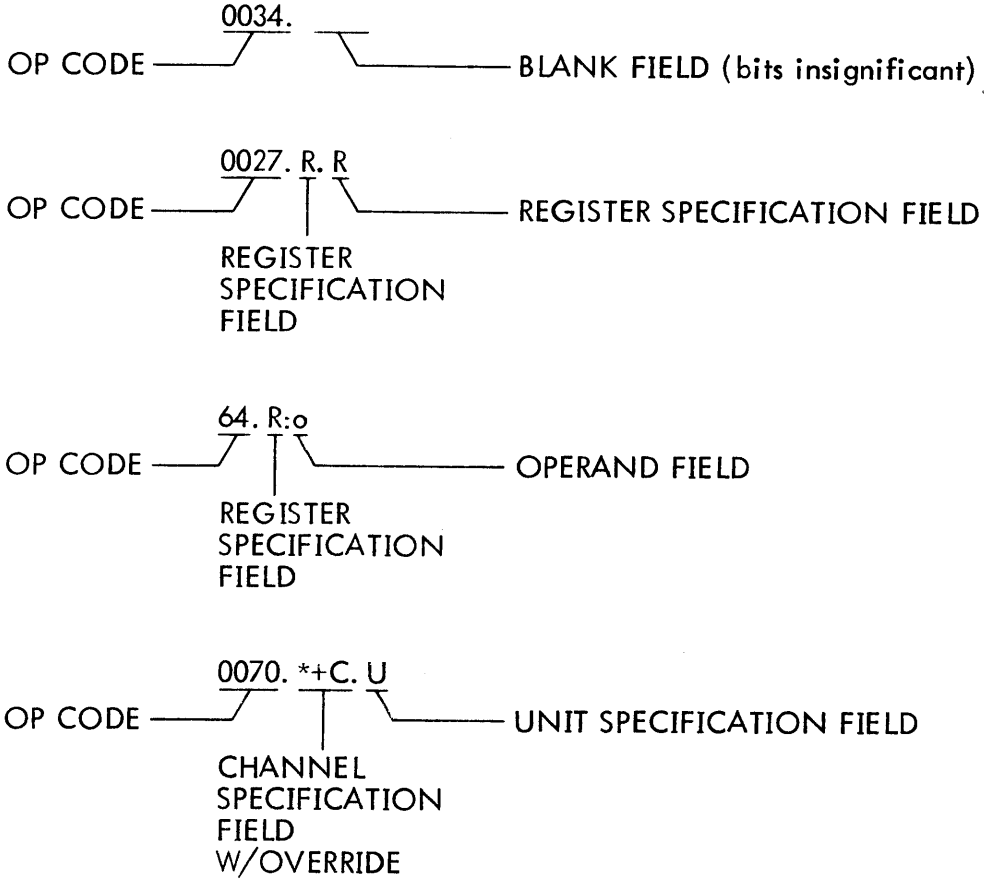
Additional bits in the instruction word specify how the general operation is to be performed. For example, when adding the contents of one register to the contents of another, the additional bits indicate which registers are involved.

Some instructions access memory and use formats that specify an address. The address bits are sometimes supplemented by special bits (indirect, index) in the instruction word. In other cases, the additional bits are not used for address modification, but are used to define a condition under which the specified memory location will be accessed or to indicate which of the CPU registers will be used in the operation. The appropriate formats are provided with the individual instruction descriptions.

3-3 INSTRUCTION FORMULA

The instruction formula, presented with each instruction description, provides a graphic representation of a 24-bit instruction word. The formula expresses an instruction word as a concatenation of its various fields where each field is represented by one or more octal digits. For example, the formula 21. *+X:a expresses a memory reference branch where: 21 represents a 6-bit (2 octal digits) OP CODE, * and X are additive quantities defining the indirect (*) and index (X) field, and "a" is a memory reference in a 15-bit address field.

The period (.) and colon (:) provide field separation in the formula, with the colon indicating right/left justification. All digits or references to the left of the colon are left-justified, and those to the right are right-justified in their respective fields. The absence of a colon indicates that all digits or references are left-justified in their fields. Examples of instruction formulas are shown on the following page.



3-4 ADDRESSING TECHNIQUE

When memory exceeds 32,768 words, the DC 6024/4 automatic memory mapping scheme comes into consideration. Memory from 0 to 32,767 words comprise Map 0, while Map 1 consists of memory from 32,768 to 65,535 words.

The addressing technique considers the most significant bit of the Program Address register (P_{15}) as the Map bit. $P_{15} = 0$ specifies that the current map is Map 0, and $P_{15} = 1$ indicates Map 1. By performing a logical OR function between the Map bit and the immediate address reference, a program may directly address up to 32,768 words in its own map.

In Map 0 ($P_{15} = 0$), immediate address references may be indexed to access up to 65,536 words since ORing the Map bit would not alter the effective address. However, when in Map 1, Map 0 may not be referenced in this manner since all immediate references are biased by 100000_8.

NOTE

It should be noted that when the last location in Map 0 is used for an address reference, the Program Counter will have advanced by the time the effective address is computed and the address reference in this location will be biased by 100000_8 which places the address in Map 1.

Indirect references are not affected by the Map bit. This allows multilevel indirect references, indexed at any level, to access up to 256K words irrespective of the current map.

A group of Long Branch instructions have a 16-bit immediate address field, allowing Map 0 or Map 1 programs to directly address up to 65,536 words. The Long Branch instructions are not affected by the Map bit.

The basic memory reference formats and their instruction formulas are illustrated in Figure 3-1. Figure 3-2 illustrates the memory referencing logic.

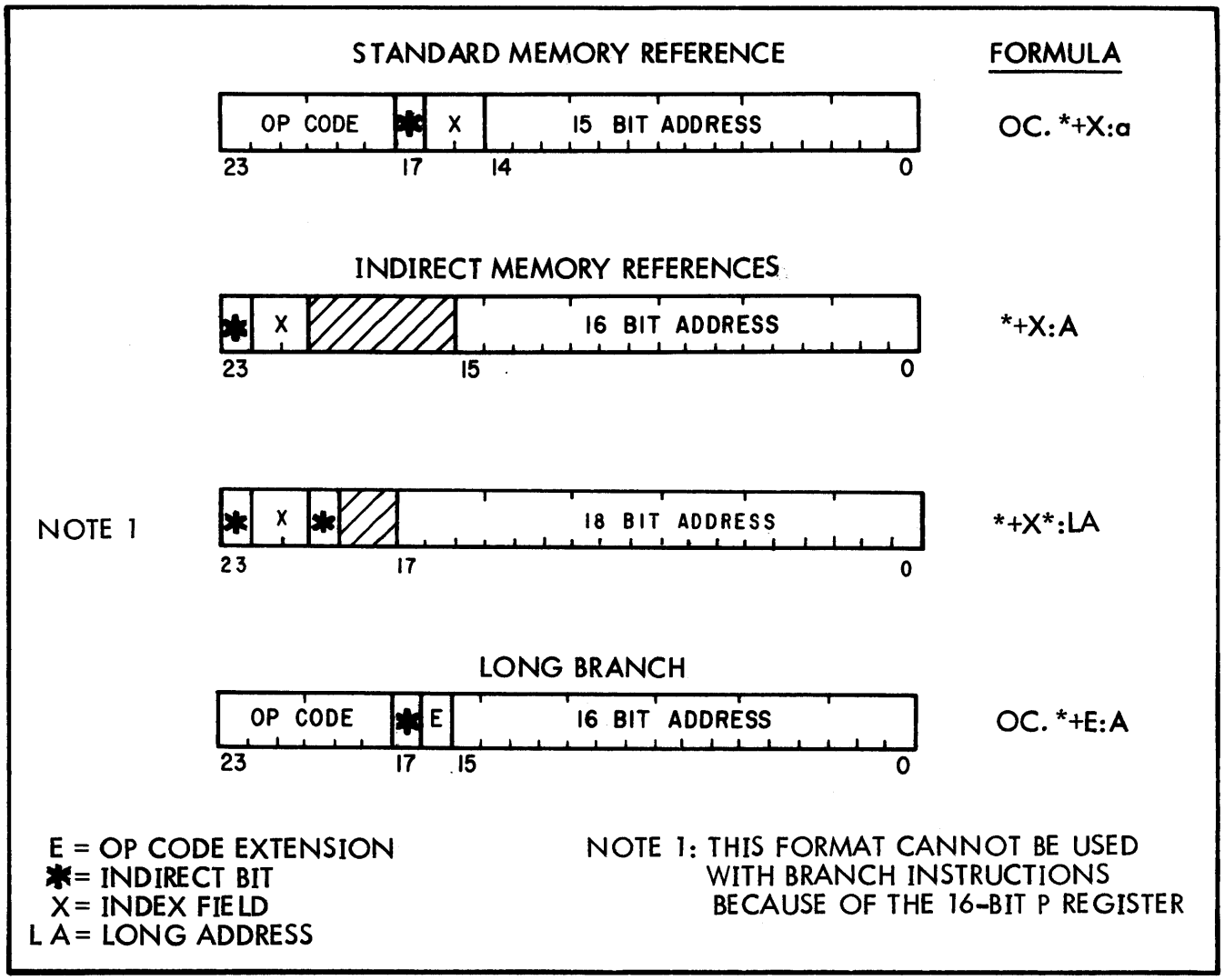
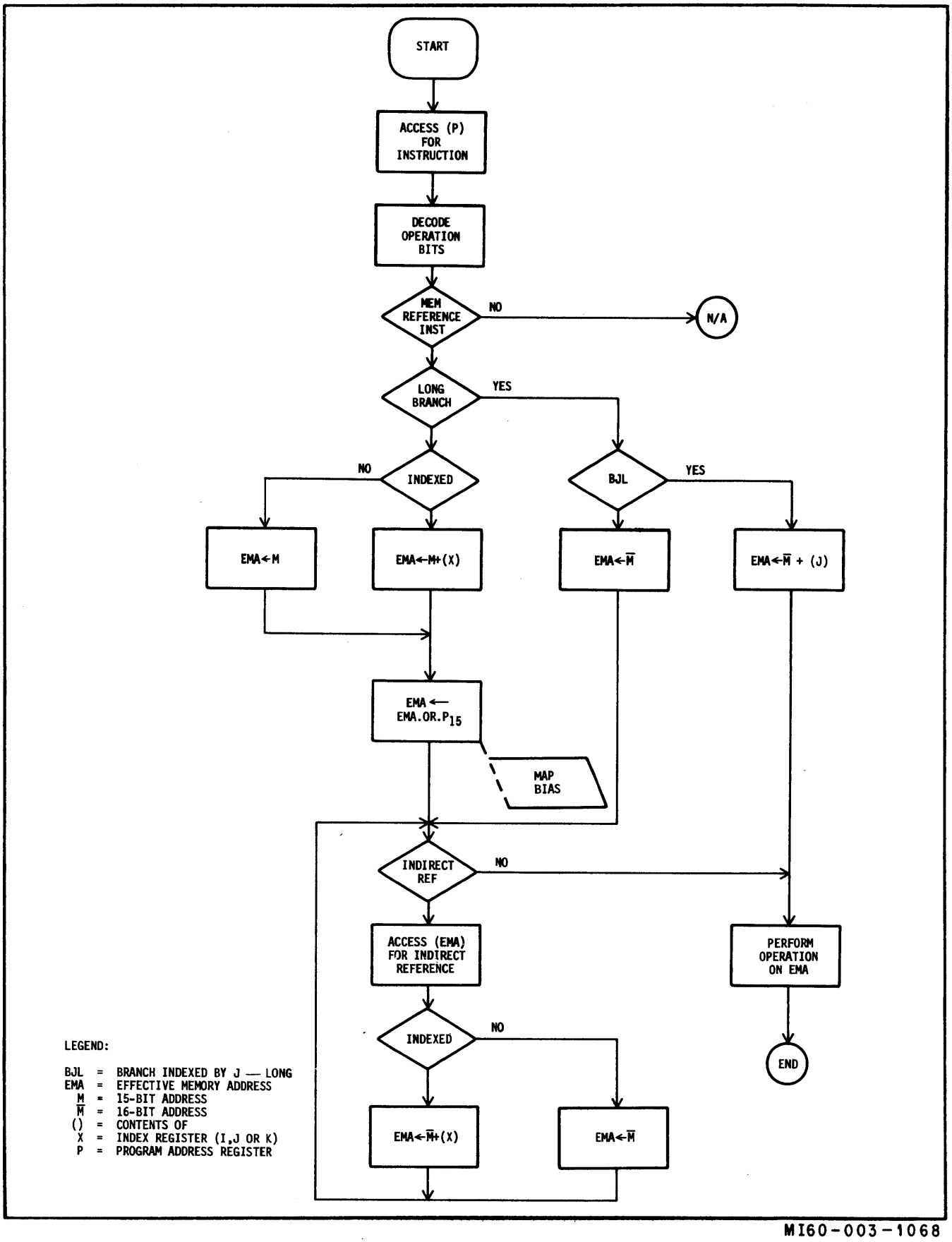


Figure 3-1. DC 6024/4 Memory Reference Formats and Formulas

3-5 INSTRUCTION DESCRIPTIONS

The following paragraphs describe, in detail, the various instructions in the DC 6024/4 repertiore. The instructions have been arranged by functional groups (Arithmetic, Branch, Compare, etc.). General information pertaining to the entire group is presented in the introductory paragraphs.



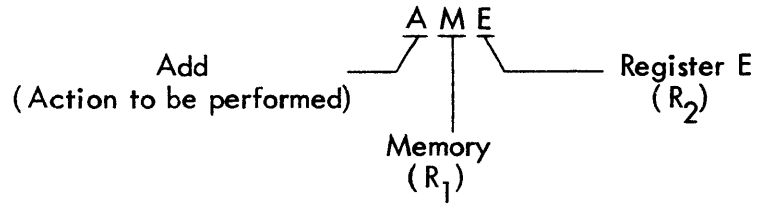
MI60-003-1068

Figure 3-2. Memory Reference Logic

3-6 ARITHMETIC INSTRUCTIONS

The DC 6024/4 Arithmetic instruction group includes the standard arithmetic operations —— addition, subtraction, multiplication and division —— as well as square root, normalization and sign extension instructions. Also included are several register-to-register operations which compute the absolute value, negate or round off the contents, or negate the sign of one register and subsequently transfer its contents to a second register.

The arithmetic instruction mnemonics provide a brief definition of specific operations to be performed. The first letter of the mnemonic signifies the action or type of operation to be performed, the second letter identifies the first quantity or reference (R_1) to be used in the operation, and the third letter identifies the second reference (R_2). For example:



In the majority of arithmetic instructions, the result of the operation remains in R_2 leaving R_1 unchanged (except where R_1 and R_2 are the same). Certain instructions —— notably, those performing multiplication, division, sign extension and square root computation —— do not comply with the R_1/R_2 conventions stated above. These instances are described thoroughly in the individual instruction descriptions.

Unless noted otherwise, each arithmetic operation causes the Condition (C) register to be set reflecting the status of the result. The various arithmetic conditions are defined below:

(a) Positive —— Result is arithmetically greater than zero, indicated by a ONE in bit position 3 of the C register. A ZERO in bit position 3 indicates "Not Positive".

(b) Zero —— All bits of the quantity under consideration are ZEROs, indicated by a ONE in bit position 2 of the C register. A ZERO in bit position 2 indicates "Not Zero".

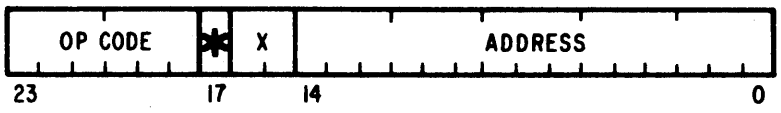
(c) Negative —— Result is arithmetically less than zero, indicated by a ONE in bit position 1 of the C register. A ZERO in bit position 1 indicates "Not Negative".

(d) Overflow —— An Overflow results from an operation instead of displaying the status of an operand. As a general rule, an arithmetic Overflow will occur when a bit is carried into the designated sign bit position and not carried out or vice versa. An Overflow condition is indicated by a ONE in bit position 0 of the C register. A ZERO in bit position 0 indicates "No Overflow".

The following instructions are included in the Arithmetic Group.

MNEMONIC	INSTRUCTION	PAGE
AAA	Add A to A	3-16
AAE	Add A to E	3-16
AAI	Add A to I	3-16
AAJ	Add A to J	3-16
AAK	Add A to K	3-16
AAM	Add A to Memory	3-14
AAT	Add A to T	3-16
AEA	Add E to A	3-16
AEE	Add E to E	3-16
AEI	Add E to I	3-16
AEJ	Add E to J	3-16
AEK	Add E to K	3-16
AEM	Add E to Memory	3-14
AET	Add E to T	3-16
AIA	Add I to A	3-16
AIE	Add I to E	3-16
AII	Add I to I	3-16
AIJ	Add I to J	3-16
AIK	Add I to K	3-16
AIM	Add I to Memory	3-13
AIT	Add I to T	3-16
AJA	Add J to A	3-16
AJE	Add J to E	3-16
AJI	Add J to I	3-16
AJJ	Add J to J	3-16
AJK	Add J to K	3-16
AJM	Add J to Memory	3-13
AJT	Add J to T	3-16
AKA	Add K to A	3-16
AKE	Add K to E	3-16
AKI	Add K to I	3-16
AKJ	Add K to J	3-16
AKK	Add K to K	3-16
AKM	Add K to Memory	3-13
AKT	Add K to T	3-16
AMA	Add Memory to A	3-12
AMB	Add Memory to Byte	3-13
AMD	Add Memory to Double	3-12
AME	Add Memory to E	3-12
AMI	Add Memory to I	3-11
AMJ	Add Memory to J	3-11
AMK	Add Memory to K	3-11
AOA	Add Operand to A	3-14
AOB	Add Operand to Byte	3-15
AOE	Add Operand to E	3-14
AOI	Add Operand to I	3-14
AOJ	Add Operand to J	3-14
AOK	Add Operand to K	3-14
AOM	Add Operand to Memory	3-15

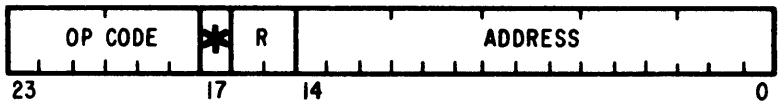
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
30. *+X:a	Add Unity to Memory	M,C	AUM



The contents of the effective memory address are incremented by one.

Cycles ..... 3 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-5

41. *+1:a	Add Memory to I	I,C	AMI
41. *+2:a	J	J,C	AMJ
41. *+3:a	K	K,C	AMK



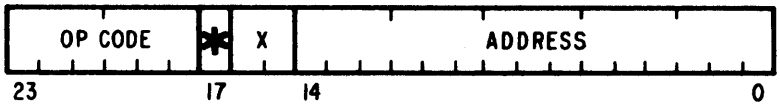
The contents of the effective memory address are algebraically added to the contents of register I, J or K.

NOTE

The immediate memory reference cannot be indexed; however, indexing of indirect references is permitted.

Cycles ..... 2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-5

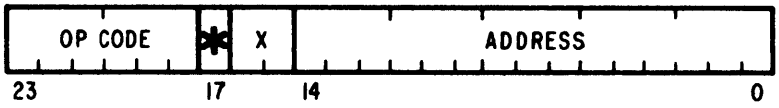
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
42. *+X:a	Add Memory to E	E,C	AME
43. *+X:a	A	A,C	AMA



The contents of the effective memory address are algebraically added to the contents of register E or A.

Cycles ..... 2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-5

44. *+X:a	Add Memory to Double	E,A,C	AMD



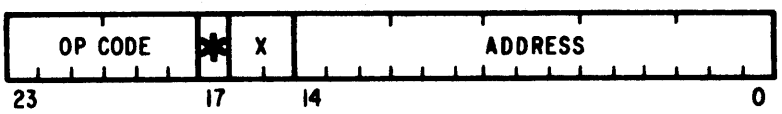
The contents of the effective memory address (EMA) and the next sequential memory address (EMA +1) are algebraically added to the contents of register D according to the double integer format defined in Section II.

NOTE

Bit A_{23} must be ZERO.

Cycles ..... 3 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-5

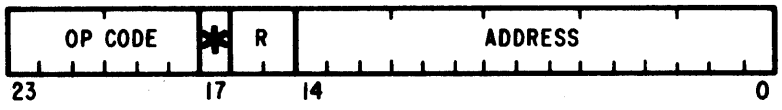
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
45. *+X:a	Add Memory to Byte	A,C	AMB



Bits 0-7 of the contents of the effective memory address are algebraically added to the contents of register B (A_0-A_7). Bits 8-23 of register A are unchanged.

Cycles .....2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-5

46. *+1:a	Add I to Memory	M,C	AIM
46. *+2:a	J	M,C	AJM
46. *+3:a	K	M,C	AKM



The 24-bit contents of register I, J or K are algebraically added to the contents of the effective memory address.

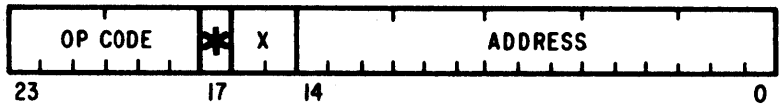
NOTE

The immediate memory reference cannot be indexed; however, indexing of indirect references is permitted, e.g.,

```
      AJM*   X
       .
       .
X     DAC    Y,K
```

Cycles ..... 3 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-5

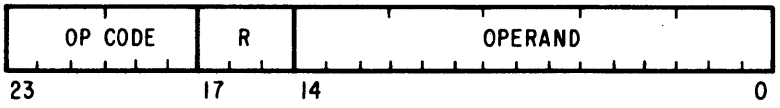
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
47. *+X:a	Add E to Memory	M, C	AEM
50. *+X:a	A	M, C	AAM



The contents of register E or A are algebraically added to the contents of the effective memory address.

Cycles .....3 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-5

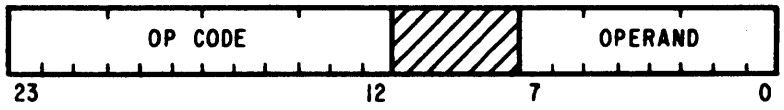
64. 1:o	Add Operand to I	I, C	AOI
64. 2:o	J	J, C	AOJ
64. 3:o	K	K, C	AOK
64. 4:o	E	E, C	AOE
64. 5:o	A	A, C	AOA
64. 6:o	T	T, C	AOT



The 15-bit unsigned operand is algebraically added to the contents of the specified register.

Cycles 1
Reference pages 3-1, 3-5

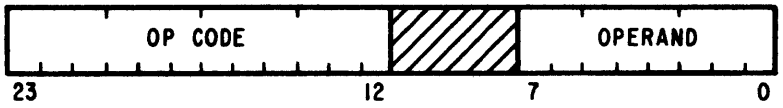
INSTRUCTION FORMULA		FUNCTION	REGISTERS AFFECTED	MNEMONIC
0012:o		Add Operand to Byte	A, C	AOB

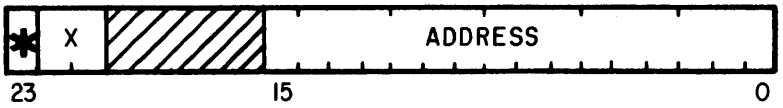


The 8-bit signed operand is algebraically added to the contents of the B register. (A_0-A_7). Bits 8-23 of register A are unchanged.

Cycles 1
References pages 3-1, 3-5

0074:o	(word 1)	Add Operand to Memory	M, C	AOM (n)
*+X:A	(word 2)			DAC (m)





The 8-bit signed operand (n) is algebraically added to the contents of the effective memory address (m).

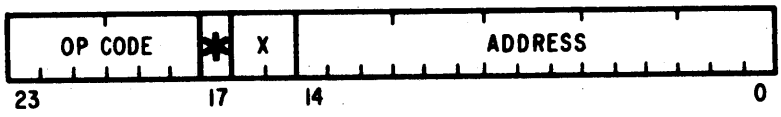
Cycles ..... 4 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0020.01.01	Add I to I	I,C	AII
0020.01.02	J	J,C	AIJ
0020.01.04	K	K,C	AIK
0020.01.10	E	E,C	AIE
0020.01.20	A	A,C	AIA
0020.01.40	T	T,C	AIT
0020.02.01	Add J to I	I,C	AJI
0020.02.02	J	J,C	AJJ
0020.02.04	K	K,C	AJK
0020.02.10	E	E,C	AJE
0020.02.20	A	A,C	AJA
0020.02.40	T	T,C	AJT
0020.04.01	Add K to I	I,C	AKI
0020.04.02	J	J,C	AKJ
0020.04.04	K	K,C	AKK
0020.04.10	E	E,C	AKE
0020.04.20	A	A,C	AKA
0020.04.40	T	T,C	AKT
0020.10.01	Add E to I	I,C	AEI
0020.10.02	J	J,C	AEJ
0020.10.04	K	K,C	AEK
0020.10.10	E	E,C	AEE
0020.10.20	A	A,C	AEA
0020.10.40	T	T,C	AET
0020.20.01	Add A to I	I,C	AAI
0020.20.02	J	J,C	AAJ
0020.20.04	K	K,C	AAK
0020.20.10	E	E,C	AAE
0020.20.20	A	A,C	AAA
0020.20.40	T	T,C	AAT
0020.40.01	Add T to I	I,C	ATI
0020.40.02	J	J,C	ATJ
0020.40.04	K	K,C	ATK
0020.40.10	E	E,C	ATE
0020.40.20	A	A,C	ATA
0020.40.40	T	T,C	ATT

The contents of R_1 are algebraically added to the contents of R_2.

Cycles 1
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
57. *+X:a	DiVide by Memory	E,A,C	DVM



The double-precision contents of register D (E and A) are algebraically divided by the single-precision contents of the effective memory address. The signed, single-precision, quotient is left in A and the remainder is left in E. The remainder will have the same sign as the original dividend and the Condition register will be set according to the status of the quotient.

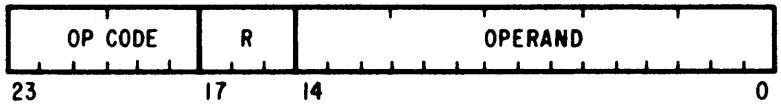
NOTES

(1) If it is desired to divide a single-precision number in A by memory, an Extend Sign of A (ESA) instruction should be executed prior to the DVM. This will establish the proper format for the dividend.

(2) If the contents of E are equal to, or greater than, the contents of memory, an Overflow condition will result and the Condition register will be set accordingly.

Cycles ..... 15 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
61.0:o	DiVide by Operand	E,A,C	DVO



The double-precision contents of register D (E and A) are algebraically divided by the 15-bit unsigned operand. The signed, single-precision, quotient is left in A and the remainder is left in E. The remainder will have the same sign as the original dividend and the Condition register will be set according to the status of the quotient.

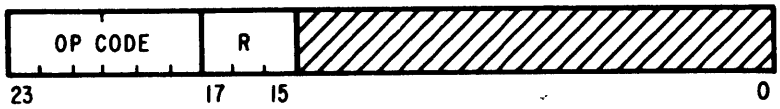
NOTES

(1) If it is desired to divide a single-precision number in A by the operand, an Extend Sign of A (ESA) instruction should be executed prior to the DVO. This will establish the proper format for the dividend.

(2) If the contents of E are equal to, or greater than, the operand, an Overflow condition will result and the Condition register will be set accordingly.

Cycles 15
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
61.1	DiVide by I	E,A,C	DVI
61.2	J	E,A,C	DVJ
61.3	K	E,A,C	DVK
61.6	T	E,A,C	DVT



The double-precision contents of register D (E and A) are algebraically divided by the contents of the specified register. The signed, single-precision, quotient is left in A and the remainder is left in E. The remainder will have the same sign as the original dividend and the Condition register will be set according to the status of the quotient.

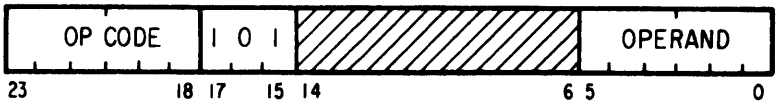
NOTES

(1) If it is desired to divide a single-precision number in A by by the contents of the specified register, an Extend Sign of A (ESA) instruction should be executed prior to the divide instruction. This will establish the proper format for the dividend.

(2) If the contents of E are equal to, or greater than, the contents of the specified register, an Overflow condition will result and the Condition register will be set accordingly.

Cycles 15
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
61.5:0	DiVide by 2	E	DV2

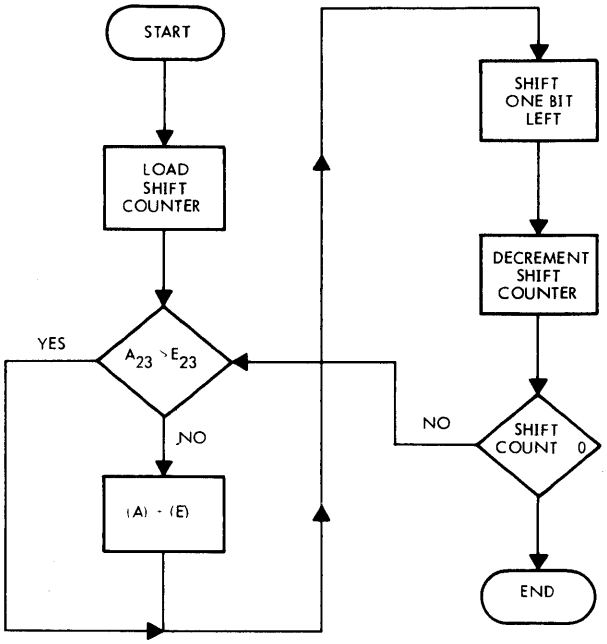


The DV2 instruction divides the contents of the E accumulator by the contents of the A accumulator, except that the arithmetic operation will be Modulo 2 (exclusive OR) instead of 2's complement arithmetic. The 6-bit operand contained in the instruction specifies the number of shifts.

NOTE

The specified number of shifts must be an even number.

This instruction is used for generating and checking error codes based on polynomial coding techniques. The polynomial and the operand to be implemented must be left-justified in the A and E accumulators. The result will be placed in the E accumulator while the polynomial will remain in the A accumulator.

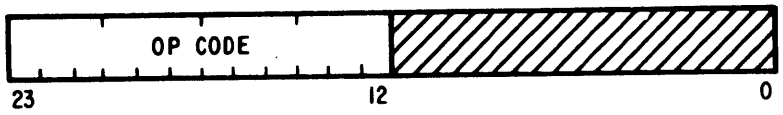


Cycles.....3 minimum. Cycles = $3+\left\lceil \frac{n-4}{2} \right\rceil$

Reference Pages.... 3-1, 3-5

Where: n = number of shifts

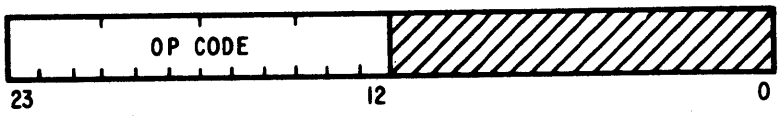
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0037.	Extend Sign of A	E, C, A	ESA



The state of the sign bit (A_{23}) of register A is copied into all 24 bit positions of register E and bit A_{23} is then set to zero. This forms a double-precision number in E and A.

Cycles 1
Reference pages 3-1, 3-5

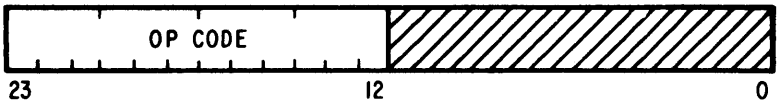
0010.	Extend Sign of Byte	A, C	ESB



The state of the register B sign bit (A_7) is copied into bit positions A_8–A_{23}, forming a sign extension of the byte.

Cycles 1
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0054.	Floating NOrmalize	E,A,I,C	FNO



The contents of register D (E and A) are shifted left arithmetically until bit E_{22} differs from E_{23}. The negative shift count (i.e., the number of shifts performed) replaces the contents of register I.

Example: Convert a double-precision integer in D to double-precision floating point format.

```
TOC   Ø     Clear Overflow
FNO         Normalize
TIB         Position exponent in byte (A0 - A7).
BOZ  *+2    If result is zero, no exponent adjustment is necessary.
AOB   46    Adjust shift count.
```

NOTES

There are four special cases where the shifting process differs from that described above.

(1) If the binary pattern 11000...0 is detected in register D, normalization is terminated to avoid creating the invalid pattern 10000...0.

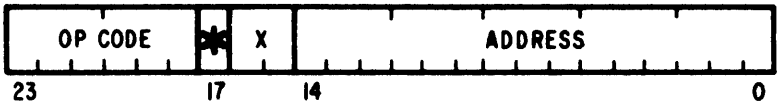
(2) If the invalid binary pattern 10000...0 is detected, it is shifted right one position, producing the pattern 11000...0. The shift count is adjusted accordingly.

(3) If the pattern 00000...0 is detected, the shift count is set to -177_8, making a zero less significant than any other value.

(4) If an Overflow condition is present when beginning the operation, the contents of register D are arithmetically shifted right one position. The shift count is set to ONE and the sign of D is complemented.

Cycles .....2+[(n-1)/4]
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
56. *+X:a	MultiplY by Memory	E,A,C	MYM



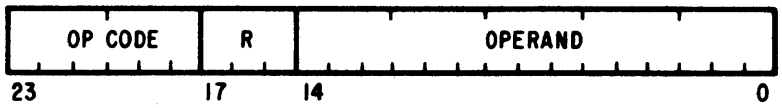
The contents of register A are algebraically multiplied by the contents of the effective memory address. The double-precision product replaces the previous contents of register D (E and A).

NOTE

An Overflow will result if the full-scale negative number (1000....00) is used as both the multiplier and multiplicand.

Cycles .....8 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-5

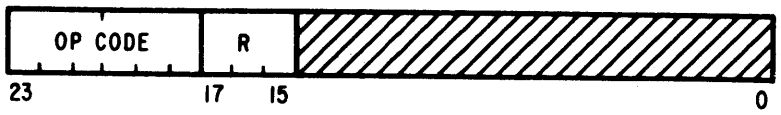
60. 0:o	MultiplY by Operand	E,A,C	MYO



The contents of register A are algebraically multiplied by the 15-bit unsigned operand in the instruction word. The double-precision product replaces the previous contents of register D (E and A).

Cycles 8
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
60.1	MultiplY by I	E,A,C	MYI
60.2	J	E,A,C	MYJ
60.3	K	E,A,C	MYK
60.4	E	E,A,C	MYE
60.5	A	E,A, C	MYA
60.6	T	E,A,C	MYT



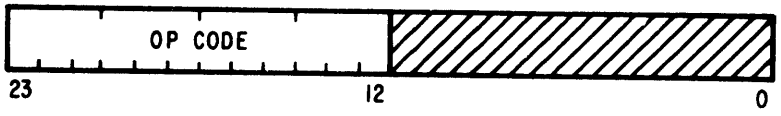
The contents of register A are algebraically multiplied by the contents of the specified register. The double-precision product replaces the previous contents of register D (E and A).

NOTE

An Overflow will result if the full-scale negative number (1000....00) is used as both the multiplier and multiplicand.

Cycles 8
Reference pages 3-1, 3-5

0005. Negate of Byte to Byte A,C NBB



The contents of register B (A_0-A_7) are two's complemented. Bit positions A_8-A_{23} are unchanged.

NOTE

An Overflow will result when negating 2^7 (full-scale negative byte).

Cycles 1
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0022.01.01	Negate of I to I	I,C	NII
0022.01.02	J	J,C	NIJ
0022.01.04	K	K,C	NIK
0022.01.10	E	E,C	NIE
0022.01.20	A	A,C	NIA
0022.01.40	T	T,C	NIT
0022.02.01	Negate of J to I	I,C	NJI
0022.02.02	J	J,C	NJJ
0022.02.04	K	K,C	NJK
0022.02.10	E	E,C	NJE
0022.02.20	A	A,C	NJA
0022.02.40	T	T,C	NJT
0022.04.01	Negate of K to I	I,C	NKI
0022.04.02	J	J,C	NKJ
0022.04.04	K	K,C	NKK
0022.04.10	E	E,C	NKE
0022.04.20	A	A,C	NKA
0022.04.40	T	T,C	NKT
0022.10.01	Negate of E to I	I,C	NEI
0022.10.02	J	J,C	NEJ
0022.10.04	K	K,C	NEK
0022.10.10	E	E,C	NEE
0022.10.20	A	A,C	NEA
0022.10.40	T	T,C	NET
0022.20.01	Negate of A to I	I,C	NAI
0022.20.02	J	J,C	NAJ
0022.20.04	K	K,C	NAK
0022.20.10	E	E,C	NAE
0022.20.20	A	A,C	NAA
0022.20.40	T	T,C	NAT
0022.40.01	Negate of T to I	I,C	NTI
0022.40.02	J	J,C	NTJ
0022.40.04	K	K,C	NTK
0022.40.10	E	E,C	NTE
0022.40.20	A	A,C	NTA
0022.40.40	T	T,C	NTT

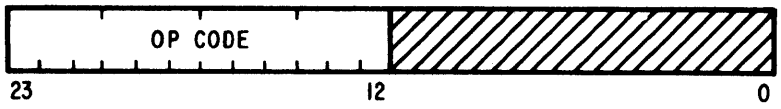
The two's complement of the contents of R_1 replace the previous contents of R_2.

NOTE

An Overflow will result when negating 2^{23} (full-scale negative number).

Cycles 1
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0033.	Negate of Double to Double	E,A,C	NDD



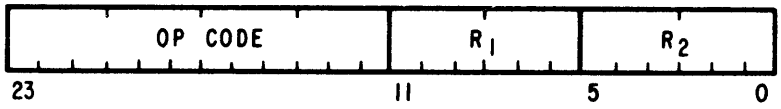
The contents of register D (E and A), in double-precision format, are two's complemented.

NOTE

An Overflow will result when negating 2^{46} (full-scale negative double integer).

Cycles 2
Reference pages 3-1, 3-5

0032.01.01	Negate Sign of I	I,C	NSI
0032.02.02	J	J,C	NSJ
0032.04.04	K	K, C	NSK
0032.10.10	E	E,C	NSE
0032.20.20	A	A,C	NSA
0032.40.40	T	T,C	NST



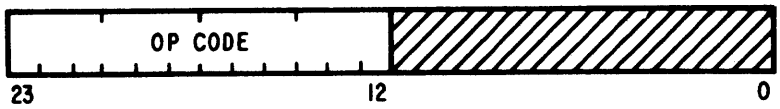
The sign bit of the specified register is complemented.

NOTE

An overflow will result when negating a full-scale negative.

Cycles 1
Reference pages 3-1, 3-5

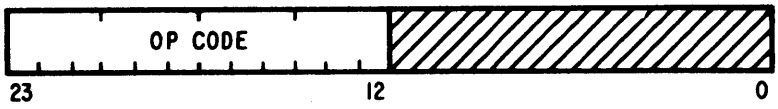
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0006.	Positive of Byte to Byte	A,C	PBB



The absolute value of the contents of register B (A_0-A_7) is placed in register B.

Cycles 1
Reference pages 3-1, 3-5

0034.	Positive of Double to Double	E,A,C	PDD



The absolute value of the contents of register D is placed in register D according to the double integer format defined in Section II.

Cycles 2
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0023.01.01	Positive of I to I	I, C	PII
0023.01.02	J	J, C	PIJ
0023.01.04	K	K, C	PIK
0023.01.10	E	E, C	PIE
0023.01.20	A	A, C	PIA
0023.01.40	T	T, C	PIT
0023.02.01	Positive of J to I	I, C	PJI
0023.02.02	J	J, C	PJJ
0023.02.04	K	K, C	PJK
0023.02.10	E	E, C	PJE
0023.02.20	A	A, C	PJA
0023.02.40	T	T, C	PJT
0023.04.01	Positive of K to I	I, C	PKI
0023.04.02	J	J, C	PKJ
0023.04.04	K	K, C	PKK
0023.04.10	E	E, C	PKE
0023.04.20	A	A, C	PKA
0023.04.40	T	T, C	PKT
0023.10.01	Positive of E to I	I, C	PEI
0023.10.02	J	J, C	PEJ
0023.10.04	K	K, C	PEK
0023.10.10	E	E, C	PEE
0023.10.20	A	A, C	PEA
0023.10.40	T	T, C	PET
0023.20.01	Positive of A to I	I, C	PAI
0023.20.02	J	J, C	PAJ
0023.20.04	K	K, C	PAK
0023.20.10	E	E, C	PAE
0023.20.20	A	A, C	PAA
0023.20.40	T	T, C	PAT
0023.40.01	Positive of T to I	I, C	PTI
0023.40.02	J	J, C	PTJ
0023.40.04	K	K, C	PTK
0023.40.10	E	E, C	PTE
0023.40.20	A	A, C	PTA
0023.40.40	T	T, C	PTT

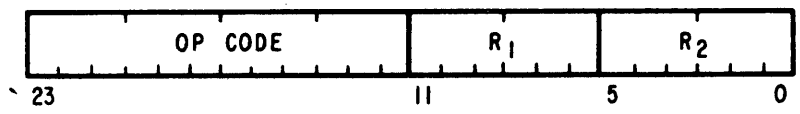
The absolute value of the contents of R_1 replaces the previous contents of R_2.

NOTE

An Overflow will result when negating a full-scale negative.

Cycles 1
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0075.01.01	Round of I to I	I, C	RII
0075.01.02	J	J, C	RIJ
0075.01.04	K	K, C	RIK
0075.01.10	E	E, C	RIE
0075.01.20	A	A, C	RIA
0075.01.40	T	T, C	RIT
0075.02.01	Round of J to I	I, C	RJI
0075.02.02	J	J, C	RJJ
0075.02.04	K	K, C	RJK
0075.02.10	E	E, C	RJE
0075.02.20	A	A, C	RJA
0075.02.40	T	T, C	RJT
0075.04.01	Round of K to I	I, C	RKI
0075.04.02	J	J, C	RKJ
0075.04.04	K	K, C	RKK
0075.04.10	E	E, C	RKE
0075.04.20	A	A, C	RKA
0075.04.40	T	T, C	RKT
0075.10.01	Round of E to I	I, C	REI
0075.10.02	J	J, C	REJ
0075.10.04	K	K, C	REK
0075.10.10	E	E, C	REE
0075.10.20	A	A, C	REA
0075.10.40	T	T, C	RET
0075.40.01	Round of T to I	I, C	RTI
0075.40.02	J	J, C	RTJ
0075.40.04	K	K, C	RTK
0075.40.10	E	E, C	RTE
0075.40.20	A	A, C	RTA
0075.40.40	T	T, C	RTT

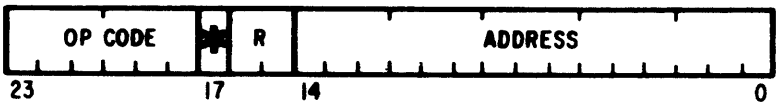


Round the contents of R_1 as a function of A and place the result in R_2.

If bit A_{22} is a ONE, the contents of R_1 +1 are transferred to R_2. If A_{22} is ZERO, the contents of R_1 replace the previous contents of R_2. In either case, R_1 is unchanged except when the same as R_2.

Cycles 1
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
51. *+1:a	Subtract Memory from I	I,C	SMI
51. *+2:a	J	J,C	SMJ
51. *+3:a	K	K,C	SMK



The contents of the effective memory address are algebraically subtracted from the contents of register I, J or K.

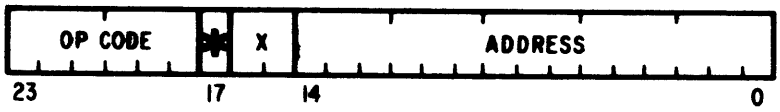
NOTE

The immediate memory reference cannot be indexed; however, indexing of indirect references is permitted, e.g.,

```
      SMI*   X
       .
       .
       .
X     DAC    Y,J
```

Cycles ..... 2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-5

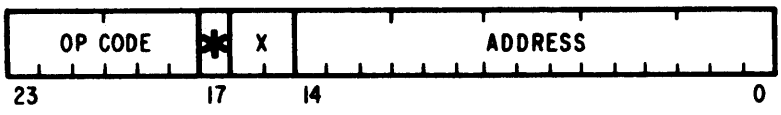
52. *+X:a	Subtract Memory from E	E,C	SME
53. *+X:a	A	A,C	SMA



The contents of the effective memory address are algebraically subtracted from the contents of register E or A.

Cycles ..... 2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
54. *+X:a	Subtract Memory from Double	E,A,C	SMD



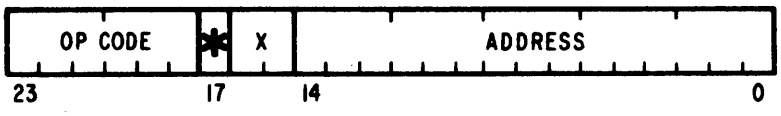
The contents of the effective memory address (EMA) and the next sequential address (EMA +1) are algebraically subtracted from the contents of register D (E and A), according to the double integer format defined in Section II.

NOTE

Failure to adhere to the double integer format will provide unpredictable results. Bit A_{23} must be ZERO.

Cycles .....3 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-5

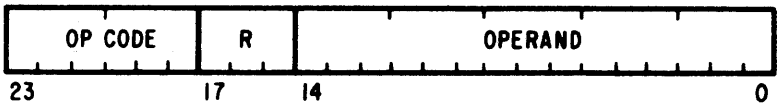
55. *+X:a	Subtract Memory from Byte	A,C	SMB



The contents of bits 0-7 of the effective memory address are algebraically subtracted from register B (A_0-A_7). Bits A_8-A_{23} are unaffected.

Cycles .....2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-5

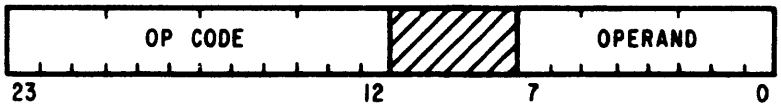
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
65.1:o	Subtract Operand from I	I, C	SOI
65.2:o	J	J, C	SOJ
65.3:o	K	K, C	SOK
65.4:o	E	E, C	SOE
65.5:o	A	A, C	SOA
65.6:o	T	T, C	SOT



The 15-bit unsigned operand is algebraically subtracted from the contents of the specified register.

Cycles 1
Reference pages 3-1, 3-5

0013:o Subtract Operand from Byte A, C SOB



The 8-bit signed operand is algebraically subtracted from the contents of register B (A_0-A_7). Bits A_8-A_{23} are unaffected.

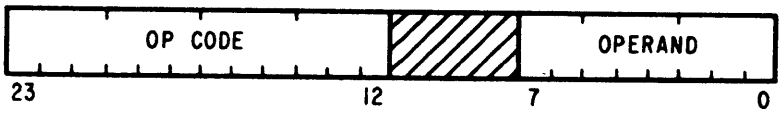
Cycles 1
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0021.01.02	Subtract I from J	J,C	SIJ
0021.01.04	K	K,C	SIK
0021.01.10	E	E,C	SIE
0021.01.20	A	A,C	SIA
0021.01.40	T	T,C	SIT
0021.02.01	Subtract J from I	I,C	SJI
0021.02.04	K	K,C	SJK
0021.02.10	E	E,C	SJE
0021.02.20	A	A,C	SJA
0021.02.40	T	T,C	SJT
0021.04.01	Subtract K from I	I,C	SKI
0021.04.02	J	J,C	SKJ
0021.04.10	E	E,C	SKE
0021.04.20	A	A,C	SKA
0021.04.40	T	T,C	SKT
0021.10.01	Subtract E from I	I,C	SEI
0021.10.02	J	J,C	SEJ
0021.10.04	K	K,C	SEK
0021.10.20	A	A,C	SEA
0021.10.40	T	T,C	SET
0021.20.01	Subtract A from I	I,C	SAI
0021.20.02	J	J,C	SAJ
0021.20.04	K	K,C	SAK
0021.20.10	E	E,C	SAE
0021.20.40	T	T,C	SAT
0021.40.01	Subtract T from I	I,C	STI
0021.40.02	J	J,C	STJ
0021.40.04	K	K,C	STK
0021.40.10	E	E,C	STE
0021.40.20	A	A,C	STA

The contents of R_1 are algebraically subtracted from R_2.

Cycles 1
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0076:014	Square Root	E,A,C	SRT
0076:027	Square Root — Extended	E,A,C	SRE



The contents of register A are treated as a 23-bit positive integer. The square root of this quantity is placed in register A, right justified, and the remainder is placed in register E so that:

$$root^2 + remainder = original\ integer.$$

If the sign bit (23) of register A is set, the Condition register will be set to OVERFLOW.

SRT generates a root of 12 significant bits; i.e., the true integer root of any positive integer in register A.

SRE generates a root of 23 significant bits. This extended significance is obtained by assuming 22 zeros to the right of bit A_0; effectively, multiplying the contents of A by 2^{22} and, consequently, the root by 2^{11}.

Consider the following examples where: A_n implies a binary point to the right of bit n.

Positive Integer	Instruction	Root (Octal)
2 at A_0	SRT	1 at A_0
2 at A_0	SRE	1.3240 at A_{11}
2 at A_{20}	SRT	1.3240 at A_{10}
2 at A_{20}	SRE	1.3240474 at A_{21}

Cycles: SRT 14
SRE 25
Reference pages 3-1, 3-5

3-7 BRANCH INSTRUCTIONS

The Branch group of instructions can be divided into two basic types: conditional and unconditional branches. Conditional branches cause control to be transferred to a specified address upon detection of a certain machine condition as indicated by the contents of the Condition register. Unconditional branches cause control to be transferred unconditionally to a specified address. Only Long Branch instructions (BJL, BLL, BRL, BSL, BUL, BLU) should be used on the last location of each 32K Memory Map. Use of any other Branch instruction will cause a Branch to the opposite Memory Map.

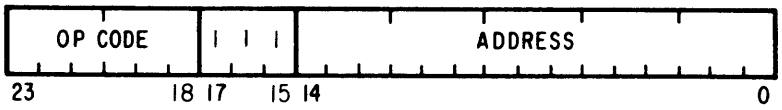
The following instructions are included in the Branch group.

MNEMONIC	INSTRUCTION	PAGE
BBI	Branch when Byte Address +1 in I $\neq$ 0	3-36
BBJ	Branch when Byte Address +1 in J $\neq$ 0	3-36
BJL*	Branch indexed by J —— Long	3-40
BLI	Branch and Link I	3-41
BLJ	Branch and Link J	3-41
BLK	Branch and Link K	3-41
BLL*	Branch and Link (J) —— Long	3-41
BLU	Branch and Link —— Unrestricted	3-44
BNN	Branch on Not Negative	3-39
BNO	Branch on No Overflow	3-39
BNP	Branch on Not Positive	3-39
BNZ	Branch on Not Zero	3-39
BON	Branch On Negative	3-39
BOO	Branch On Overflow	3-39
BOP	Branch On Positive	3-39
BOZ	Branch On Zero	3-39
BRL*	Branch and Reset interrupts —— Long	3-43
BSL*	Branch and Save return —— Long	3-42
BUC	Branch UnConditionally	3-38
BUL*	Branch Unconditionally —— Long	3-38
BWI	Branch When I+1 $\neq$ 0	3-40
BWJ	Branch When J+1 $\neq$ 0	3-40
BWK	Branch When K+1 $\neq$ 0	3-40

*Long Branches disregard the automatic memory mapping logic.

The instruction definitions all refer to a 16-bit effective memory address. The address is derived from the address specified in the instruction and the automatic mapping process.

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
60.7:a	Branch when Byte Address +1 in I $\neq$ 0	I	BBI
61.7:a	Branch when Byte Address +1 in J $\neq$ 0	J	BBJ



The contents of bits 22 and 23 of the specified index register (I or J) is incremented by one. If the result of this addition (in bits 22 and 23) is not 00_2, then the contents of register P (current PROGRAM ADDRESS) is replaced by the 15-bit effective memory address. If the result of the addition to bits 22 and 23 is 00_2, then bits 22 and 23 are set to 01_2 and bits 0-21 are incremented by one. If the resultant sum in bits 0-21 is zero, then register P advances to the next sequential program location and the index register is set to 40000000_8. Otherwise, the contents of register P are replaced by the 15-bit effective memory address.

In general, the BBI and BBJ instructions are used as special index register increments in order to sequentially reference consecutive bytes in memory via the EMB and RBM instructions. Consider the following example which will move 11 consecutive bytes starting from the third byte at location '200 to the first byte at location '300.

Example:

```
TMJ   = '60000200
TMI   = '20000300
TNK   11
EMB   0
RBM   0
BBI   *+1
BBJ   *+1
BWK   *-4
```

Occasionally, it is possible to use the address of a portion of index register I or J as a byte counter as well as a word pointer. This may be illustrated by the following example which will set the buffer starting at byte 3 of location '100 through byte 3 of location '102 to blanks.

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC

Example:

```
TOB   "6"
TMI   = '77777775      bits 22 and 23 = 3, bits 0-21 = -3
RBM   '100+3
BBI   *-1
```

However, it should be noted this technique of using the index register as both a byte counter and word pointer may be used only in certain instances. Specifically, when the following relationship is true.

$$R\left(\frac{4\text{-b.n.}}{3}\right) = R\left(\frac{CT}{3}\right)$$

where: R ( ) = remainder

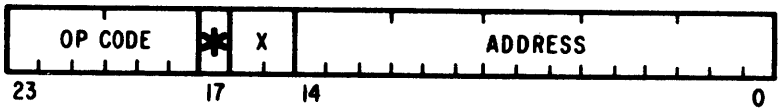
b.n. = the starting byte number (1, 2, or 3)

CT = the number of bytes to be referenced.

Cycles.....1

Reference Pages. . 3-1, 3-2, 3-35

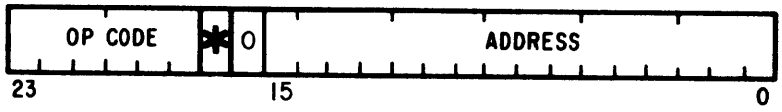
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
21. *+X:a	Branch UnConditionally	P	BUC



The contents of register P (current PROGRAM ADDRESS) are replaced by the 16-bit effective memory address.

Cycles ..... 1 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-35

26. *+0:A	Branch Unconditionally — Long	P	BUL



The contents of register P (current PROGRAM ADDRESS) are replaced by the 16-bit effective memory address.

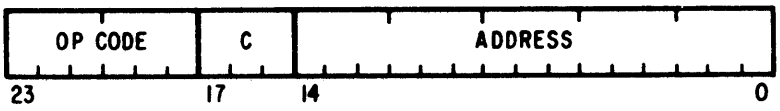
NOTE

The immediate memory reference cannot be indexed; however, indexing of indirect references is permitted, e.g.,

```
        BUL*   X
        .
        .
        .
X       DAC    Y,I
```

Cycles ..... 1 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-35

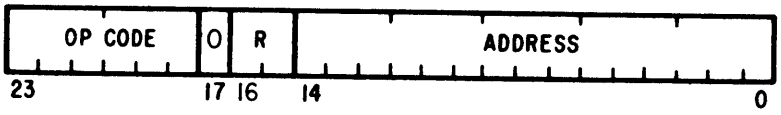
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
22.0:a	Branch On Overflow	P	BOO
22.1:a	On Negative	P	BON
22.2:a	On Zero	P	BOZ
22.3:a	On Positive	P	BOP
22.4:a	on No Overflow	P	BNO
22.5:a	on Not Negative	P	BNN
22.6:a	on Not Zero	P	BNZ
22.7:a	on Not Positive	P	BNP



The contents of the Condition register are tested for the specified condition. If the condition is present, the contents of the P register (current PROGRAM ADDRESS) are replaced by the 16-bit effective memory address. If the specified condition is not present, the program advances to the next sequential location (PROGRAM ADDRESS +1).

Cycles 1
Reference pages 3-1, 3-2, 3-35

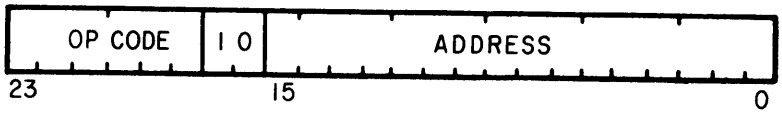
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
23. 1:a	Branch When I+1 ≠ 0	I,P	BWI
23. 2:a	J+1 ≠0	J,P	BWJ
23. 3:a	K+1 ≠ 0	K,P	BWK



The contents of the specified register are incremented by one and then tested for zero. If the contents are not zero, the contents of register P (current PROGRAM ADDRESS) are replaced by the 16-bit effective memory address. If the contents are zero, the program advances to the next sequential location (PROGRAM ADDRESS +1).

Cycles 1
Reference pages 3-1, 3-2, 3-35

23. 4:A Branch indexed by J — Long P BJL



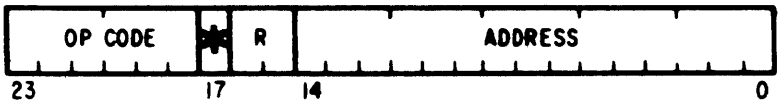
The contents of register P (current PROGRAM ADDRESS) are replaced by the 16-bit effective memory address.

NOTE

The immediate memory reference is automatically indexed by J.

Cycles 1
Reference pages 3-1, 3-2, 3-35

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
24. *+1:a	Branch and Link I	I, P	BLI
24. *+2:a	J	J,P	BLJ
24. *+3:a	K	K,P	BLK



The contents of register I, J or K are replaced by the next sequential address (PROGRAM ADDRESS +1) and the contents of register P (Current PROGRAM ADDRESS) are replaced by the 16-bit effective memory address.

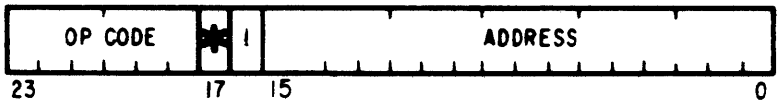
NOTE

The immediate memory reference cannot be indexed; however, indexing of indirect references is permitted, e.g.

```
        BLI*   X
        .
        .
        .
X       DAC    Y,J
```

Cycles ..... 1 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-35

26. *+2:a — Branch and Link (J) — Long — J,P — BLL



The contents of register J are replaced by the next sequential address (PROGRAM ADDRESS +1) and the contents of register P (current PROGRAM ADDRESS) are replaced by the 16-bit effective memory address.

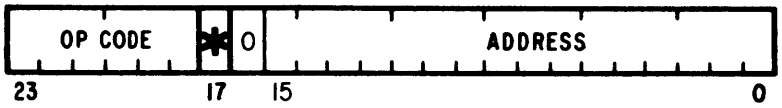
NOTE

The immediate memory reference cannot be indexed; however, indexing of indirect references is permitted, e.g.

```
        BLI*
        .
        .
        .
X       DAC
```

Cycles ..... 1 (+1 per indirect references)
Reference pages 3-1, 3-2, 3-35

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
25. *+0:A	Branch and Save return — Long	P	BSL



The next sequential address (PROGRAM ADDRESS +1), along with the contents of the Condition register are stored in the 16-bit effective memory address (EMA). The contents of register P (current PROGRAM ADDRESS) are then replaced by the address following the effective memory address (EMA +1).

This instruction is used to enter an interrupt subroutine because it provides a means of returning to the main program at the point of interrupt and saves the machine status (Condition) at the time of the interrupt.

NOTES

(1) The contents of the Condition register are stored in bit positions 16-19 of the EMA and the return address (PROGRAM ADDRESS +1) is stored in bits 0-15. The remaining bits are set to ZEROs; however, refer to note 2 for variation on bit 20.

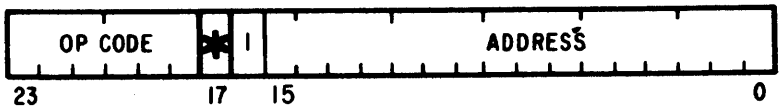
(2) If a "Power Down" interrupt occurs when the CPU is halted, bit 20 is set to ONE. If the CPU is running when the "Power Down" interrupt occurs, bit 20 is set to ZERO.

(3) The immediate memory reference cannot be indexed; however, indexing of indirect references is permitted.

(4) External interrupts are prohibited for the period of one instruction following the execution of this instruction.

Cycles ..... 2 (+1 per indirect reference)
Reference pages 2-15, 3-1, 3-2, 3-35

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
25. *+2:a	Branch and Reset interrupt —Long	C,P	BRL



The highest-level active interrupt is reset (i.e., returned to the inactive state) and the contents of register P (current PROGRAM ADDRESS) are replaced by the 16-bit effective memory address.

BRL is nornally used to exit an interrupt subroutine. If BRL contains an indirect reference, the last word in the indirect address chain contains the previous status (i.e., C register contents at the time of the interrupt) in bit positions M_{16}-M_{19} and the return address in bit positions M_0-M_{15} as a result of the BSL instruction. The C register is restored and the program branches to the return address (restarting the machine to the pre-interrupt status).

Example:

```
          .
          .
          TMA
L         AMA
          SMA        Interrupt occurs (EXM K).
          .
          .
K         BSL  M     Dedicated interrupt location.

M         ***        M becomes L+1 as a result of BSL at K.
          :          The C register contents are stored in
          :          M16-M19.
          BRL* M     Restore C register and return to L+1.
```

NOTES

(1) The BRL will not reset the interrupt if external interrupts have been held by an HXI instruction. Control will be returned to the effective memory address.

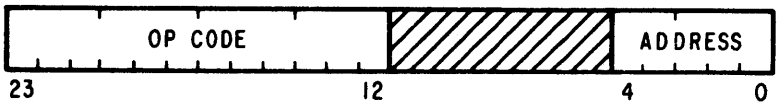
(2) Those executive traps, which are not affected by the HXI instruction, will be reset by the BRL.

(3) The immediate memory reference cannot be indexed; however, indexing indirect references is permitted, e.g.

```
          BRL*   X
          .
          .
X         DAC    Y,K
```

Cycles ..... 1 (+1 per indirect reference)
Reference pages 2-15, 3-1, 3-2, 3-35

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0067:a	Branch and Link — Unrestricted	J,P	BLU



The next sequential address (PROGRAM ADDRESS +1) replaces the contents of register J and the contents of register P (current PROGRAM ADDRESS) are replaced by the 5-bit immediate memory address.

NOTE

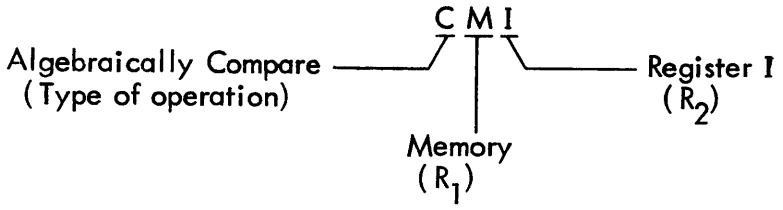
If the CPU is equipped with the Program Restrict system, the Program Restricted Flag (PRF) comes into consideration. Execution of the BLU instruction will turn OFF the PRF. If the computer is in a Halt condition and single cycle operation is attempted when the Program Restrict is active, the BLU instruction will be treated as an NOP instruction.

Cycles 1
Reference pages 2-17, 3-1, 3-2, 3-35

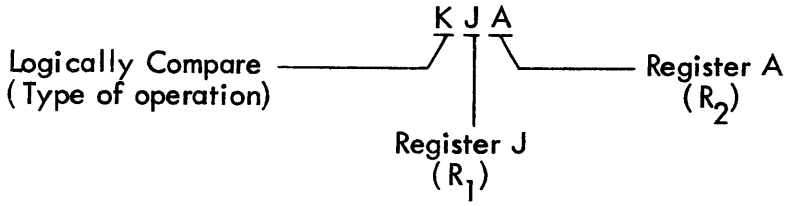
3-8 COMPARE INSTRUCTIONS

The Compare group of instructions is composed of two basic types of operations: algebraic and logical comparisions. Both types of instructions compare two referenced quantities and set the Condition register according to the result. Algebraic comparisons treat the references as signed (+ or -) quantities, while logical comparisons assume the references are unsigned quantities.

Algebraic comparisons are identified by the letter "C" as the first letter in the instruction mnemonic (e.g., CAI). Logical comparisons use a mnemonic code beginning with the letter "K" (KAI). The second letter of the mnemonic code designates the first of the compared quantities (R_1) and the last letter designates the second quantity (R_2). For example:



or



Both algebraic and logical comparisons are performed according to the formula:

$R_2 - R_1 =$ C (positive, zero or negative)

Therefore, $R_2 > R_1$, $R_2 < R_1$ and $R_2 = R_1$ will set the Condition register (C) to positive (+), negative (-) and zero (0), respectively.

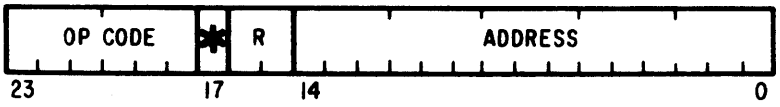
The following instructions are included in the DC 6024/5 Compare group.

MNEMONIC	INSTRUCTION	PAGE
CAE	Compare A and E	3-52
CAI	Compare A and I	3-52
CAJ	Compare A and J	3-52
CAK	Compare A and K	3-52
CAT	Compare A and T	3-52
CEA	Compare E and A	3-52
CEI	Compare E and I	3-52
CEJ	Compare E and J	3-52

MNEMONIC	INSTRUCTION	PAGE
CEK	Compare E and K	3-52
CET	Compare E and T	3-52
CIA	Compare I and A	3-52
CIE	Compare I and E	3-52
CIJ	Compare I and J	3-52
CIK	Compare I and K	3-52
CIT	Compare I and T	3-52
CJA	Compare J and A	3-52
CJE	Compare J and E	3-52
CJI	Compare J and I	3-52
CJK	Compare J and K	3-52
CJT	Compare J and T	3-52
CKA	Compare K and A	3-52
CKE	Compare K and E	3-52
CKI	Compare K and I	3-52
CKJ	Compare K and J	3-52
CKT	Compare K and T	3-52
CMA	Compare Memory and A	3-48
CMB	Compare Memory and Byte	3-49
CME	Compare Memory and E	3-48
CMI	Compare Memory and I	3-48
CMJ	Compare Memory and J	3-48
CMK	Compare Memory and K	3-48
COB	Compare Operand and Byte	3-50
CTA	Compare T and A	3-52
CTE	Compare T and E	3-52
CTI	Compare T and I	3-52
CTJ	Compare T and J	3-52
CTK	Compare T and K	3-52
CZA	Compare Zero and A	3-50
CZD	Compare Zero and Double	3-51
CZE	Compare Zero and E	3-50
CZI	Compare Zero and I	3-50
CZJ	Compare Zero and J	3-50
CZK	Compare Zero and K	3-50
CZM	Compare Zero and Memory	3-49
CZT	Compare Zero and T	3-50
KAE	Kompare A and E	3-53
KAI	Kompare A and I	3-53
KAJ	Kompare A and J	3-53
KAK	Kompare A and K	3-53
KAT	Kompare A and T	3-53
KEA	Kompare E and A	3-53
KEI	Kompare E and I	3-53
KEJ	Kompare E and J	3-53
KEK	Kompare E and K	3-53
KET	Kompare E and T	3-53
KIA	Kompare I and A	3-53
KIE	Kompare I and E	3-53
KIJ	Kompare I and J	3-53

MNEMONIC	INSTRUCTION	PAGE
KIK	Kompare I and K	3-53
KIT	Kompare I and T	3-53
KJA	Kompare J and A	3-53
KJE	Kompare J and E	3-53
KJI	Kompare J and I	3-53
KJK	Kompare J and K	3-53
KJT	Kompare J and T	3-53
KKA	Kompare K and A	3-53
KKE	Kompare K and E	3-53
KKI	Kompare K and I	3-53
KKJ	Kompare K and J	3-53
KKT	Kompare K and T	3-53
KOB	Kompare Operand and Byte	3-54
KTA	Kompare T and A	3-53
KTE	Kompare T and E	3-53
KTI	Kompare T and I	3-53
KTJ	Kompare T and J	3-53
KTK	Kompare T and K	3-53

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
31. *+1:a	Compare Memory and I	C	CMI
31. *+2:a	J	C	CMJ
31. *+3:a	K	C	CMK



The contents of the effective memory address and the contents of register I, J, or K are algebraically compared and the Condition register is set to the status of the result.

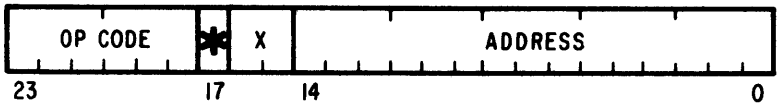
NOTE

The immediate memory reference cannot be indexed; however, indexing of indirect references is permitted, e.g.,

```
        CMI*    X
         .
         .
X       DAC     Y,K
```

Cycles ..... 2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-45

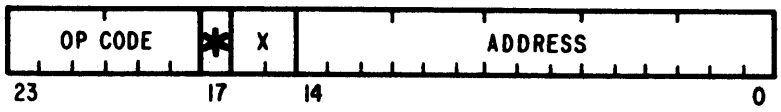
32. *+X:a	Compare Memory and A	C	CME
33. *+X:a	E	C	CMA



The contents of the effective memory address and the contents of register E or A are algebraically compared and the Condition register is set to the status of the result.

Cycles ..... 2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-45

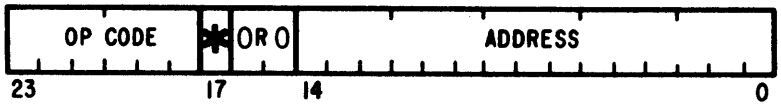
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
34. *+X:a	Compare Memory and Byte	C	CMB



The contents of register B (A_0-A_7) and the contents of the effective memory address (M_0-M_7) are algebraically compared and the Condition register is set to the status of the result.

Cycles .....2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-45

41. *+0:a — Compare Zero and Memory — C — CZM



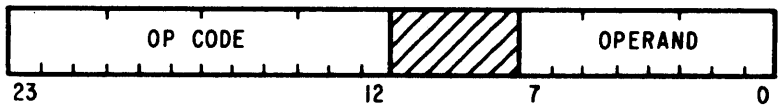
The contents of the effective memory address and zero are algebraically compared and the Condition register is set to the status of the result.

NOTE

The immediate memory reference cannot be indexed; however, indexing of indirect references is permitted.

Cycles .....2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-45

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0014:o	Compare Operand and Byte	C	COB



The 8-bit signed operand and the contents of register B (A_0-A_7) are algebraically compared and the Condition register is set to the status of the result.

Cycles 1
Reference pages 3-1, 3-45

0024.00.01	Compare Zero and I	C	CZI
0024.00.02	J	C	CZJ
0024.00.04	K	C	CZK
0024.00.10	E	C	CZE
0024.00.20	A	C	CZA
0024.00.40	T	C	CZT

The contents of the specified register and zero are algebraically compared and the Condition register is set to the status of the result.

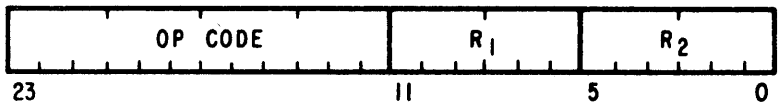
Cycles 1
Reference pages 3-1, 3-45

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0024.00.30	Compare Zero and Double	C	CZD

The contents of register D (E and A) and zero are algebraically compared and the Condition register is set to the status of the result.

Cycles 1
Reference pages 3-1, 3-45

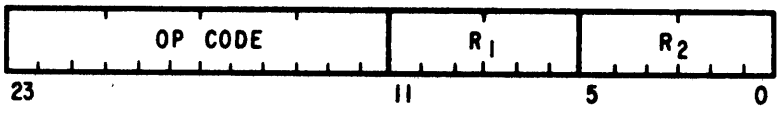
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0024.01.02	Compare I and J	C	CIJ
0024.01.04	K	C	CIK
0024.01.10	E	C	CIE
0024.01.20	A	C	CIA
0024.01.40	T	C	CIT
0024.02.01	Compare J and I	C	CJI
0024.02.04	K	C	CJK
0024.02.10	E	C	CJE
0024.02.20	A	C	CJA
0024.02.40	T	C	CJT
0024.04.01	Compare K and I	C	CKI
0024.04.02	and J	C	CKJ
0024.04.10	and E	C	CKE
0024.04.20	and A	C	CKA
0024.04.40	and T	C	CKT
0024.10.01	Compare E and I	C	CEI
0024.10.02	J	C	CEJ
0024.10.04	K	C	CEK
0024.10.20	A	C	CEA
0024.10.40	T	C	CET
0024.20.01	Compare A and I	C	CAI
0024.20.02	J	C	CAJ
0024.20.04	K	C	CAK
0024.20.10	E	C	CAE
0024.20.40	T	C	CAT
0024.40.01	Compare T and I	C	CTI
0024.40.02	J	C	CTJ
0024.40.04	K	C	CTK
0024.40.10	E	C	CTE
0024.40.20	A	C	CTA



The contents of R_1 and the contents of R_2 are algebraically compared and the Condition register is set to the status of the result.

Cycles 1
Reference pages 3-1, 3-45

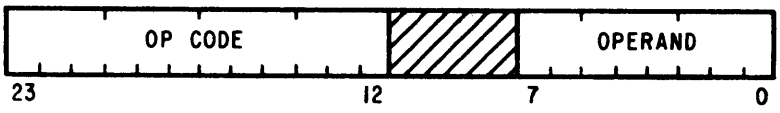
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0025.01.02	Kompare I and J	C	KIJ
0025.01.04	and K	C	KIK
0025.01.10	and E	C	KIE
0025.01.20	and A	C	KIA
0025.01.40	and T	C	KIT
0025.02.01	Kompare J and I	C	KJI
0025.02.04	and K	C	KJK
0025.02.10	and E	C	KJE
0025.02.20	and A	C	KJA
0025.02.40	and T	C	KJT
0025.04.01	Kompare K and I	C	KKI
0025.04.02	and J	C	KKJ
0025.04.10	and E	C	KKE
0025.04.20	and A	C	KKA
0025.04.40	and T	C	KKT
0025.10.01	Kompare E and I	C	KEI
0025.10.02	and J	C	KEJ
0025.10.04	and K	C	KEK
0025.10.20	and A	C	KEA
0025.10.40	and T	C	KET
0025.20.01	Kompare A and I	C	KAI
0025.20.02	and J	C	KAJ
0025.20.04	and K	C	KAK
0025.20.10	and E	C	KAE
0025.20.40	and T	C	KAT
0025.40.01	Kompare T and I	C	KTI
0025.40.02	and J	C	KTJ
0025.40.04	and K	C	KTK
0025.40.10	and E	C	KTE
0025.40.20	and A	C	KTA



The contents of R_1 and R_2 are logically compared and the Condition register is set according to the result.

Cycles 1
Reference pages 3-1, 3-45

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0015:o	Kompare Operand and Byte	C	KOB



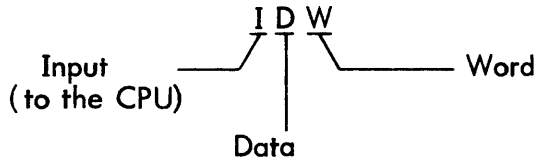
The 8-bit operand and the contents of register B (A_0-A_7) are logically compared and the Condition register is set according to the result.

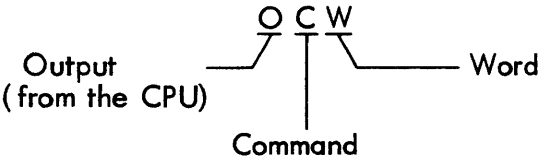
Cycles 1
Reference pages 3-1, 3-45

3-9 INPUT/OUTPUT INSTRUCTIONS

The Input/Out (I/O) instructions provide the required control for all communications between the CPU and the input/output structure. In addition to controlling data transfers between the CPU and peripheral units, the I/O instruction repertoire allows peripheral unit command functions and status testing to be placed under program control.

The specific I/O operation can be identified by examination of the individual instruction mnemonics. All I/O instruction mnemonics use the letter "W" to indicate that a full word is to be transferred between the CPU and the I/O structure. The first letter of the mnemonic indicates the direction of the transfer (input or output). The second letter indicates the type of word to be transferred. For example:





There is no "I/O hold", or delay, imposed by the hardware. All I/O instructions are executed unconditionally; i.e., the CPU is not forced to wait for a response from the I/O structure in order to complete the instruction execution cycle.

Although there is no built-in hold/delay provision, a programmed delay can be implemented if desired. At the beginning of each I/O instruction cycle, the Condition register is cleared. At the end of the execution phase of each I/O instruction, bit 2 (ZERO) is set to "ZERO" if the selected channel was ready and accepted the command. If the selected channel was not ready, bit 2 of the Condition register remains set to "NOT ZERO". The program can test the "NOT ZERO" state of bit 2 with a branch instruction following the I/O instruction. When bit 2 is set to "NOT ZERO+, a programmed delay is implemented. For example:

```
ODW     '0103       Output word to Channel 1, Unit 3
BNZ     *-1         Delay if not ready
---                 Continue if ready
```

Two examples of a channel being not ready are that the peripheral units data transfer capability is slower than that of the program loop and therefore cannot accept data as it is available from the channel. The other example occurs in a channel/multiunit environment where the channel is connected to peripheral unit A and peripheral unit B is selected for a data transfer. In this instance, the channel remains not ready until a disconnect/connect sequence is performed and peripheral unit B is connected to the channel. Two cycles are required for the disconnect/connect sequence.

NOTE

Status returned to the Condition register immediately after completion of an I/O instruction refers to channel status only. A ready ("ZERO") condition indicates the channel accepted the I/O command. This does not imply the I/O operation was completed with the selected peripheral unit.

If the program selects a non-existent channel or unit, the channel accepts the command or data and leaves bit 2 of the Condition register set to "NOT ZERO" to indicate not ready. The channel will remain not ready for any subsequent commands.

If the system is equipped with the Program Restrict/Instruction Trap option, all I/O instructions will be affected.

The I/O command modes are determined by the configuration of bits 5 and 4 of the OCW instruction and are as follows:

(1) Normal - The Normal Channel Operation command is raised by bits 5 and 4 of the OCW being ZEROs (0,0).

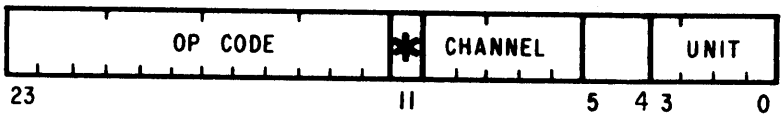
(2) Multiplex - This command is raised by bits 5 and 4 of the OCW being in a ZERO, ONE (0,1) configuration. (The CPU releases the channel to a master/slave pair of peripheral units.)

(3) Offline - This command is the same as the Multiplex command, except the I/O drivers in the channel are turned off, allowing the second CPU to share peripherals without need of peripheral switches. (Assumes control of I/O bus.) The command is raised by bits 5 and 4 being in a ONE, ZERO (1, 0) configuration.

(4) Reset - This command operates the same as a Normal command, but resets the channel out of either the Multiplex or Offline mode. (Channel restored on-line, unit selected.) This command is raised by bits 5 and 4 being in a ONE, ONE (1, 1) configuration.

The following instructions are included in the Input/Output group.

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0070.*+C.U	Output Command Word	C	OCW



An 8-bit or a 24-bit command word is transferred from the A register to the specified channel/unit combination and the Condition register is set to "ZERO". If the selected channel is not ready, the Condition register remains set to "NOT ZERO" which allows a programmed delay if desired.

Bits 0-3 of the OCW instruction form a 4-bit paralleled unit code that is used to select a particular peripheral unit. The configuration of bits 4 and 5 determines the Multiplex or Offline mode (explained on page 3-56 and in Table 1-1, Section I) for a particular channel. The configuration of bits 6-10 determines which channel is to be selected. Bit 11 is the Override bit, and bits 12-23 define the general process that is to be performed.

If the Override bit (*) is set (ONE), the command word assumes immediate control over the channel. The contents of the A register are transferred to the channel and a disconnect/connect sequence is initiated. The Condition register is set to "ZERO" to indicate the channel has accepted but not necessarily executed the command. Upon completion of the disconnect/connect sequence, the channel transfers the command word to the unit.

If the Override bit is not set (ZERO) and the OCW specifies a unit other than the unit connected to the channel and the channel is ready, the command word is accepted by the channel. The Condition register is set to "NOT ZERO" to indicate the channel is not ready. A disconnect/connect sequence is performed and the command is transferred to the unit. The Condition register is reset to "ZERO" to indicate ready.

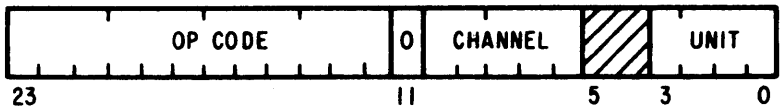
NOTE

Following the execution of an OCW, the channel remains not ready until the peripheral unit accepts the data.

If the selected channel is an ABC channel and is actively engaged in a block transfer, executing an OCW with the Override bit set terminates the transfer sequence leaving the contents of the TAR and WCR intact. If the Override bit is not set and the ABC channel is engaged in a block transfer, the OCW instruction will be ignored. The Condition register will remain set to "NOT ZERO". Once an ABC channel is activated it will not accept an OCW with the Override bit not set until the word count is complete; i.e., all words in the block have been transferred and WCR equals zero.

Cycles..... 1
Reference pages 3-1, 3-55, 3-56

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0073.00+C.U	Input Status Word	A,C	ISW



A status word is transferred from the specified channel/unit combination to register A and the Condition register is set to "Zero".

If the addressed channel/unit combination is not ready (see notes 1 and 2) or a status word is not available, the Condition register is set to "Not Zero" to allow a programmed delay.

NOTES

(1) If the selected channel is in the process of executing a command (resulting from a previous OCW), the channel indicates not ready (Condition register remains set to "NOT ZERO") and ignores the ISW instruction until the peripheral unit accepts the OCW command. The channel indicates ready (Condition register set to "ZERO") and accepts the ISW when it is executed again.

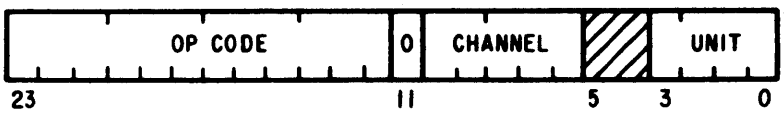
(2) If the ISW specifies a unit other than the unit connected to the channel, the channel indicates not ready and ignores the command. A disconnect/connect is initiated.

(3) If the selected channel is an ABC channel engaged in a block transfer, the Condition register is set to "ZERO" and a 24-bit status word is transferred to the A register. Bits 0 through 7 contain the unit status and bit 23 contains the ABC word count status.

(4) If the selected unit is receiving data as the result of an ODW instruction, the ISW will be accepted and the Condition register is set to "ZERO".

Cycles 1
Reference pages 3-1, 3-55, 3-56

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0071.00+C.U	Output Data Word	C	ODW



A data word is transferred from register A to the specified channel/unit combination and the Condition register is set to "Zero".

If the channel is busy and cannot accept the data word, the Condition register is set to "Not Zero" to allow a programmed delay.

NOTES

(1) Although a 24-bit word is transferred to the channel, the peripheral unit accepts only a predetermined number of bits (dictated by peripheral unit design).

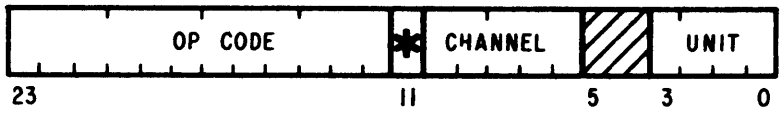
(2) For character-oriented units and units accepting data words of less than 24 bits, the data for transfer must be right-justified in the A register prior to executing the ODW instruction.

(3) If ODW instruction specifies a unit other than the unit connected to the channel and the channel is ready, the channel accepts the ODW, sets the Condition register to "ZERO", and initiates a disconnect/connect sequence. After completion of the disconnect/connect sequence, the ODW is transferred to the unit. The channel indicates ready to subsequent I/O instructions.

(4) If the ODW instruction specifies an ABC channel that is engaged in a block transfer, the Condition register remains set to "NOT ZERO" and the ODW is ignored. An ABC channel, once activated, will not accept an ODW instruction until the word count is complete; i.e., all words in the block have been transferred and WCR equals zero.

Cycles 1
Reference pages 3-1, 3-55, 3-56

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0072. *+C. U	Input Data Word	A, C	IDW



A data word is transferred from the specified channel/unit combination to register A and the Condition register is set to "Zero".

If the channel is not ready or data from the specified unit is not available, the Condition register is set to "Not Zero" to allow a programmed delay.

NOTES

(1) If the selected unit is in the process of executing a command as the result of a previous OCW instruction, the channel indicates not ready (Condition register remains set to "NOT ZERO") and the IDW is ignored. At the completion of the OCW, the Condition register is set to "ZERO" and the IDW instruction will be accepted by the channel.

(2) If the selected unit is in the process of receiving data as a result of an ODW instruction and data is available from the unit, an ODW will be accepted and the Condition register set to "ZERO".

(3) If the IDW instruction specifies a unit other than the unit connected to the channel, the channel indicates not ready (Condition register remains set to "NOT ZERO"), ignores the instruction, and initiates a disconnect/connect sequence.

(4) If a IDW instruction specifies an ABC channel that is engaged in a block transfer, the Condition register remains set to "NOT ZERO" (channel not ready) and the instruction is ignored. An ABC channel, once activated, will not accept an IDW instruction until the word count is complete; i.e., all words in the block have been transferred and WCR equals zero.

If the Merge bit (*) is ZERO, register A is cleared prior to the data transfer. Input data is right-justified in register A.

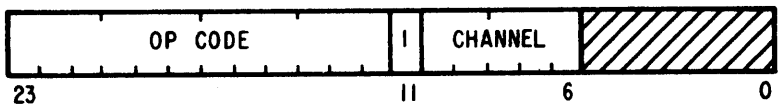
If the Merge bit is a ONE, an OR is performed between the previous contents of register A and the incoming data word. This feature, in conjunction with a shift operation, allows input data characters to be packed in register A.

Example: Two 12-bit data characters are to be packed in register A.

```
IDW     '0102     Clear A and load first character from
                  Channel 01, Unit 02
BNZ     *-1       Wait if busy
LLA     12        Shift the contents of A left 12 bits
IDW*    '0102     Merge second character
BNZ     *-1       Wait if busy
...               Continue
```

Cycles 1
Reference pages 3-1, 3-55, 3-56

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0071.40+C	Output Address Word	C	OAW



The contents of register A are transferred to the Transfer Address Register (TAR) in the specified ABC channel. The Condition register is set to "Zero".

NOTE

An ABC channel will always indicate ready for an OAW instruction. However, if the OAW specifies an invalid channel number, it will receive a "not ready" indication and the Condition register remains set to "Not Zero".

The OAW instruction does not activate an ABC channel. It transfers the starting memory address of the first of two ABC parameter words from the A register to the TAR in the selected ABC channel.

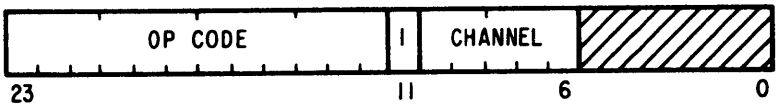
If an OAW instruction addresses an ABC channel during a block transfer sequence, the sequence will be terminated.

If the OAW instruction addresses a standard channel, the Condition register remains set to "NOT ZERO" indicating channel not ready.

If the OAW instruction addresses on IOC-IC channel containing the Channel Interrupt Generator option, the four least significant bits (0-3) of the A register are transferred to the Channel Interrupt Generator logic. These bits (unitarily) control the triggering of (one to four) a 575 nanosecond interrupt pulse(s). The Condition register remains set to "Not Zero".

Cycles 1
Reference pages 3-1, 3-55, 3-56

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0073.40+C	Input Address Word	A,C	IAW



The current contents of the Transfer Address Register (TAR) in the specified ABC channel are transferred to register A and the Condition register is set to "Zero".

NOTES

(1) An ABC channel always indicates ready to an IAW instruction.

(2) If the IAW instruction specifies an invalid channel or a standard channel, the Condition register remains set to "NOT ZERO" indicating channel not ready.

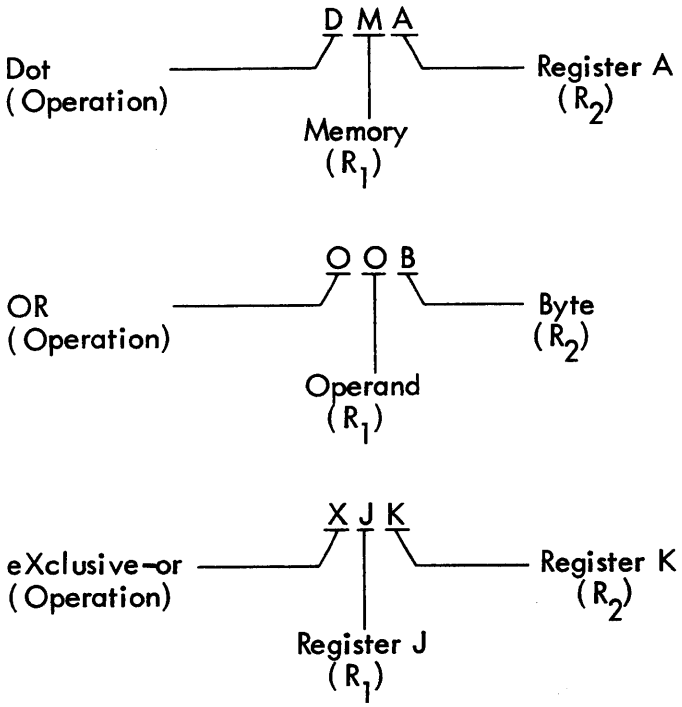
(3) Refer to Section IV for a description of an Automatic Block Transfer.

Cycles 1
Reference pages 3-1, 3-55, 3-56

3-10 LOGICAL INSTRUCTIONS

The Logical group of instructions includes AND (DOT product), OR and EXCLUSIVE-OR operations. All three types use two quantities to produce a logical result. The AND instructions use a mnemonic code beginning with the letter "D", for "Dot". The OR instructions use a mnemonic beginning with the letter "O", while EXCLUSIVE-OR instructions are distinguished by the letter "X".

The second letter of the mnemonic code identifies the first of the two quantities (R_1). The third letter signifies the second quantity (R_2). Some examples are listed below:



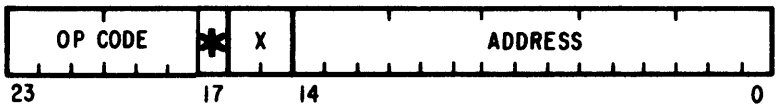
Unless specifically noted otherwise in the individual descriptions, the result of the logical operation replaces the previous contents of R_2 while R_1 is unchanged. The Condition register is set to the status of the result (Positive, Negative or Zero) after the operation. The various logical operations are illustrated in the following table.

R_1	R_2	R_1 .AND. R_2	R_1 .OR. R_2	R_1 .XOR. R_2
1	1	1	1	0
0	1	0	1	1
1	0	0	1	1
0	0	0	0	0

The following instructions are included in the Logical group.

MNEMONIC	INSTRUCTION	PAGE
DAE	Dot A with E	3-69
DAI	Dot A with I	3-69
DAJ	Dot A with J	3-69
DAK	Dot A with K	3-69
DAT	Dot A with T	3-69
DEA	Dot E with A	3-69
DEI	Dot E with I	3-69
DEJ	Dot E with J	3-69
DEK	Dot E with K	3-69
DET	Dot E with T	3-69
DIA	Dot I with A	3-69
DIE	Dot I with E	3-69
DIJ	Dot I with J	3-69
DIK	Dot I with K	3-69
DIT	Dot I with T	3-69
DJA	Dot J with A	3-69
DJE	Dot J with E	3-69
DJI	Dot J with I	3-69
DJK	Dot J with K	3-69
DJT	Dot J with T	3-69
DKA	Dot K with A	3-69
DKE	Dot K with E	3-69
DKI	Dot K with I	3-69
DKJ	Dot K with J	3-69
DKT	Dot K with T	3-69
DMA	Dot Memory with A	3-68
DOB	Dot Operand with Byte	3-68
DTA	Dot T with A	3-69
DTE	Dot T with E	3-69
DTI	Dot T with I	3-69
DTJ	Dot T with J	3-69
DTK	Dot T with K	3-69
OAE	Or A with E	3-71
OAI	Or A with I	3-71
OAJ	Or A with J	3-71
OAK	Or A with K	3-71
OAT	Or A with T	3-71
OEA	Or E with A	3-71
OEI	Or E with I	3-71
OEJ	Or E with J	3-71
OEK	Or E with K	3-71
OET	Or E with T	3-71
OIA	Or I with A	3-71
OIE	Or I with E	3-71
OIJ	Or I with J	3-71
OIK	Or I with K	3-71
OIT	Or I with T	3-71
OJA	Or J with A	3-71
OJE	Or J with E	3-71

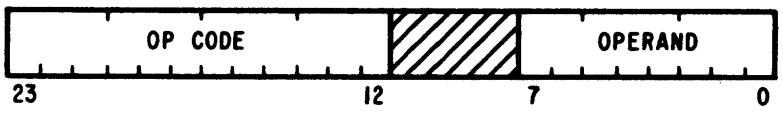
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
36. *+X:a	Dot Memory with A	A,C	DMA



A logical AND is performed between the contents of the effective memory address and the contents of register A.

Cycles .....2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-65

0016:o	Dot Operand with Byte	A,C	DOB



A logical AND is performed between the 8-bit operand and the contents of register B (A_0-A_7). Bits A_8-A_{23} are unchanged.

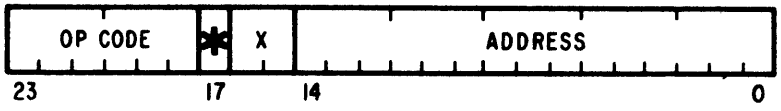
Cycles 1
Reference pages 3-1, 3-65

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0026.01.02	Dot I with J	J,C	DIJ
0026.01.04	K	K,C	DIK
0026.01.10	E	E,C	DIE
0026.01.20	A	A,C	DIA
0026.01.40	T	T,C	DIT
0026.02.01	Dot J with I	I,C	DJI
0026.02.04	K	K,C	DJK
0026.02.10	E	E,C	DJE
0026.02.20	A	A,C	DJA
0026.02.40	T	T,C	DJT
0026.04.01	Dot K with I	I,C	DKI
0026.04.02	J	J,C	DKJ
0026.04.10	E	E,C	DKE
0026.04.20	A	A,C	DKA
0026.04.40	T	T,C	DKT
0026.10.01	Dot E with I	I,C	DEI
0026.10.02	J	J,C	DEJ
0026.10.04	K	K,C	DEK
0026.10.20	A	A,C	DEA
0026.10.40	T	T,C	DET
0026.20.01	Dot A with I	I,C	DAI
0026.20.02	J	J,C	DAJ
0026.20.04	K	K,C	DAK
0026.20.10	E	E,C	DAE
0026.20.40	T	T,C	DAT
0026.40.01	Dot T with I	I,C	DTI
0026.40.02	J	J,C	DTJ
0026.40.04	K	K,C	DTK
0026.40.10	E	E,C	DTE
0026.40.20	A	A,C	DTA

A logical AND is performed between the contents of R_1 and R_2.

Cycles 1
Reference pages 3-1, 3-65

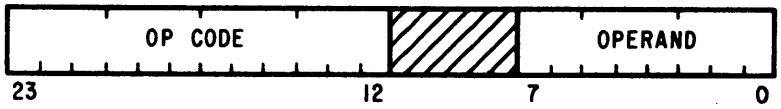
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
35. *+X:a	Or Memory with A	A, C	OMA



A logical OR is performed between the contents of the effective memory address and the contents of register A.

Cycles .....2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-65

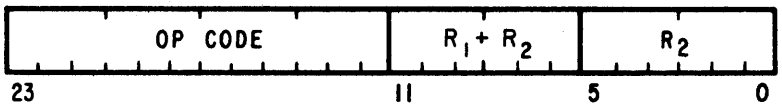
0004:o	Or Operand with Byte	A, C	OOB



A logical OR is performed between the 8-bit operand and the contents of register B (A_0-A_7). Bits A_8-A_{23} are unchanged.

Cycles 1
Reference pages 3-1, 3-65

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0030. 03. 02	Or I with J	J,C	OIJ
0030. 05. 04	K	K,C	OIK
0030. 11. 10	E	E,C	OIE
0030. 21. 20	A	A,C	OIA
0030. 41. 40	T	T,C	OIT
0030. 03. 01	Or J with I	I,C	OJI
0030. 06. 04	K	K,C	OJK
0030. 12. 10	E	E,C	OJE
0030. 22. 20	A	A,C	OJA
0030. 42. 40	T	T,C	OJT
0030. 05. 01	Or K with I	I,C	OKI
0030. 06. 02	J	J,C	OKJ
0030. 14. 10	E	E,C	OKE
0030. 24. 20	A	A,C	OKA
0030. 44. 40	T	T,C	OKT
0030. 11. 01	Or E with I	I,C	OEI
0030. 12. 02	J	J,C	OEJ
0030. 14. 04	K	K,C	OEK
0030. 30. 20	A	A,C	OEA
0030. 50. 40	T	T,C	OET
0030. 21. 01	Or A with I	I,C	OAI
0030. 22. 02	J	J,C	OAJ
0030. 24. 04	K	K,C	OAK
0030. 30. 10	E	E,C	OAE
0030. 60. 40	T	T,C	OAT
0030. 41. 01	Or T with I	I,C	OTI
0030. 42. 02	J	J,C	OTJ
0030. 44. 04	K	K,C	OTK
0030. 50. 10	E	E,C	OTE
0030. 60. 20	A	A,C	OTA



A logical OR is performed between the contents of R_1 and R_2.

NOTE

The general instruction formula for this group of Logical instructions is:

$0030. R_1 + R_2 \cdot R_2$

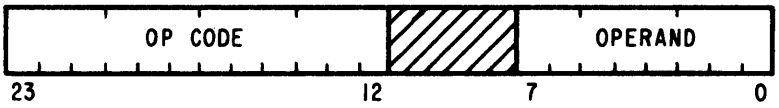
Cycles 1
Reference pages 3-1, 3-65

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
37. *+X:a	eXclusive-or Memory with A	A,C	XMA

An EXCLUSIVE-OR operation is performed between the contents of the effective memory address and the contents of register A.

Cycles .....2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-65

0017:o	eXclusive-or Operand with Byte	A,C	XOB



An EXCLUSIVE-OR operation is performed between the 8-bit operand and the contents of register B (A_0-A_7). Bits A_8-A_{23} are unchanged.

Cycles 1
Reference pages 3-1, 3-65

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0027.01.02	eXclusive-or I with J	J,C	XIJ
0027.01.04	K	K,C	XIK
0027.01.10	E	E,C	XIE
0027.01.20	A	A,C	XIA
0027.01.40	T	T,C	XIT
0027.02.01	eXclusive-or J with I	I,C	XJI
0027.02.04	K	K,C	XJK
0027.02.10	E	E,C	XJE
0027.02.20	A	A,C	XJA
0027.02.40	T	T,C	XJT
0027.04.01	eXclusive-or K with I	I,C	XKI
0027.04.02	J	J,C	XKJ
0027.04.10	E	E,C	XKE
0027.04.20	A	A,C	XKA
0027.04.40	T	T,C	XKT
0027.10.01	eXclusive-or E with I	I,C	XEI
0027.10.02	J	J,C	XEJ
0027.10.04	K	K,C	XEK
0027.10.20	A	A,C	XEA
0027.10.40	T	T,C	XET
0027.20.01	eXclusive-or A with I	I,C	XAI
0027.20.02	J	J,C	XAJ
0027.20.04	K	K,C	XAK
0027.20.10	E	E,C	XAE
0027.20.40	T	T,C	XAT
0027.40.01	eXclusive-or T with I	I,C	XTI
0027.40.02	J	J,C	XTJ
0027.40.04	K	K,C	XTK
0027.40.10	E	E,C	XTE
0027.40.20	A	A,C	XTA

An EXCLUSIVE-OR function is performed between the contents of R_1 and R_2.

Cycles 1
Reference pages 3-1, 3-65

3-11 PRIORITY INTERRUPT INSTRUCTIONS

The Priority Interrupt instruction group provides the means for program control of external interrupts. External interrupts may be selectively armed, disarmed, enabled or inhibited under program control. Other instructions provide the means for holding and releasing external interrupts, while others are available for transferring control upon interrupt detection. For a detailed description of the DC 6024/4 Priority Interrupt System, refer to Section II of this manual.

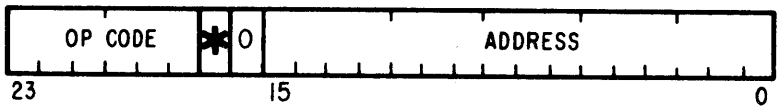
If the DC 6024/4 system is equipped with theoptional Program Restrict System and Instruction Trap, the following Priority Interrupt instructions will be affected:

a) Hold eXternal Interrupts (HXI)
b) Release eXternal Interrupts (RXI)
c) Unitarily Arm group 1 interrupts (UA1)
d) Unitarily Arm group 2 interrupts (UA2)
e) Unitarily Disarm group 1 interrupts (UD1)
f) Unitarily Disarm group 2 interrupts (UD2)
g) Unitarily Enable group 1 interrupts (UE1)
h) Unitarily Enable group 2 interrupts (UE2)
i) Unitarily Inhibit group 1 interrupts (UI1)
j) Unitarily Inhibit group 2 interrupts (UI2)
k) Transfer Double to group 1 (TD1)
l) Transfer Double to group 2 (TD2)
m) Transfer Double to group 1 (TD4)
n) Transfer Double to group 2 (TD5)

The following instructions are included in the Priority Interrupt group:

MNEMONIC	INSTRUCTION	PAGE
UE1	Unitarily Enable group 1	3-85
UE2	Unitarily Enable group 2	3-85
UI1	Unitarily Inhibit group 1	3-86
UI2	Unitarily Inhibit group 2	3-86

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
25. *+0:A	Branch and Save return — Long	P	BSL



The next sequential address (PROGRAM ADDRESS +1), along with the contents of the Condition register, are stored in the 16-bit effective memory address (EMA). The contents of register P (current PROGRAM ADDRESS) are then replaced by the address following the effective memory address (EMA +1).

This instruction is normally used to enter an interrupt subroutine because it provides a means of returning to the main program at the point of interrupt and saves the machine status (Condition) at the time of the interrupt.

NOTES

(1) The contents of the Condition register are stored in bit positions 16-19 of the EMA and the return address (PROGRAM ADDRESS +1) is stored in bits 0-15. The remaining bits are set to ZEROs, however, refer to note 2 for variation on bit 20.

(2) If a "Power Down" interrupt occurs when the CPU is halted, bit 20 is set to ONE. If the CPU is running when the "Power Down" interrupt occurs, bit 20 is set to a ZERO.

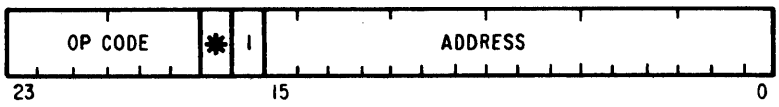
(3) The immediate memory reference cannot be indexed; however, indexing of indirect references is permitted, e.g.,

```
      BSL*   X
      .
      .
X     DAC    Y,I
```

(4) External interrupts are prohibited for the period of one instruction following the execution of this instruction.

Cycles ..... 2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-75

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
25. *+2:A	Branch and Reset interrupt —Long	C,P	BRL



The highest-level active interrupt is reset (i.e., returned to the inactive state) and the contents of register P (current PROGRAM ADDRESS) are replaced by the 16-bit effective memory address.

BRL is normally used to exit an interrupt subroutine. If BRL contains an indirect reference, the last word in the indirect address chain contains the previous status (i.e., C register contents at the time of the interrupt) in bit positions M_{16}-M_{19} and the return address in bit positions M_0-M_{15} as a result of the BSL instruction. The C register is restored and the program branches to the return address (restarting the machine to the preinterrupt status).

Example:

```
          .
          .
          TMA
L         AMA
          SMA              Interrupt occurs (EXM K).
          .
          .
K         BSL    M         Dedicated interrupt location.

M         ***              M becomes L+1 as a result of BSL at K.
          .                The C register contents are stored in
          .                M16-M19.
          .
          BRL*             Restore C register and return to L+1.
```

NOTES

(1) The BRL will not reset the interrupt if external interrupts have been held by an HXI instruction. Control will be returned to the effective memory address.

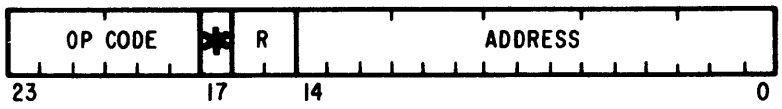
(2) Those executive traps, which are not affected by the HXI instruction, will be reset by the BRL.

(3) The immediate memory reference cannot be indexed; however, indexing indirect references is permitted, e.g.,

```
          BRL*   X
          .
          .
X         DAC    Y,K
```

Cycles ..... 1 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-75

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
27. *+1:a	Hold interrupts and Transfer I to memory	M	HTI
27. *+2:a	Transfer J to memory	M	HTJ
27. *+3:a	Transfer K to memory	M	HTK



The contents of register I, J or K replace the previous contents of the effective memory address and external interrupts are prohibited for the period of one instruction following the execution of this instruction.

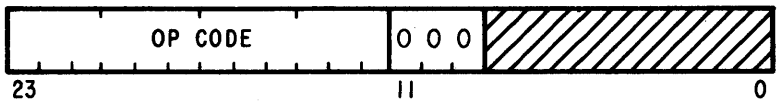
NOTE

The immediate memory reference cannot be indexed; however, indexing of indirect references is permitted, e.g.,

```
      HTI*   M
       .
       .
M     DAC    A,K
```

Cycles ..... 2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-75

0066.0 Hold eXternal Interrupts HXI



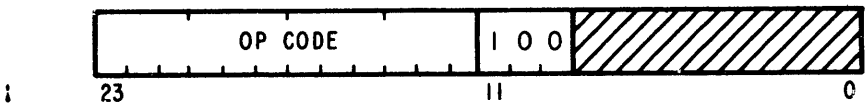
The activation of any external interrupt is prohibited. The prohibition is effective immediately upon execution of the instruction and lasts until the interrupts are released (see RXI instruction). Executive traps (Group 0, Levels 4-7) are prohibited from becoming active while the HXI is in effect.

NOTE

Only the two executive traps mentioned are affected by this instruction.

Cycles 1
Reference pages 3-1, 3-75

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0066.4	Release eXternal Interrupts		RXI



The prohibition imposed by the HXI instruction is removed, allowing any external interrupt to be activated 1 cycle after this instruction. This permits the next sequential instruction to be executed without external interruption.

If any of the affected executive traps have been triggered while an HXI was in effect, the highest level will come in first after the RXI instruction.

Cycles 1
Reference pages 3-1, 3-75

0064.01	Transfer Double to group 1	1 A/D, 1 E/I	TD1
0064.02	group 2	2 A/D, 2E/I	TD2

The contents of register D (E and A) replace the previous contents of the Arm/Disarm (A/D) and Enable/Inhibit (E/I) registers of the specified interrupt group. The contents of E are transferred to the A/D register and the contents of A are transferred to the E/I register.

NOTE

The external interrupt structure is cleared by the execution of these instructions.

Cycles 1
Reference pages 3-1, 3-75

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0065.01	Transfer group 1 to Double	E,A	T1D
0065.02	group 2	E,A	T2D

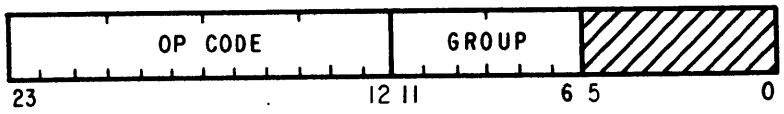
The contents of the Arm/Disarm (A/D) and Enable/Inhibit (E/I) registers of the specified interrupt group replace the previous contents of register D (E and A). The contents of the A/D register are transferred to register E and the contents of the E/I register are transferred to register A.

NOTE

The states of the external interrupts are not affected by the execution of these instructions.

Cycles 1
Reference pages 3-1, 3-75

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0065.41	Transfer group 1 to Double	E, A	T4D
0065.42	group 2	E, A	T5D

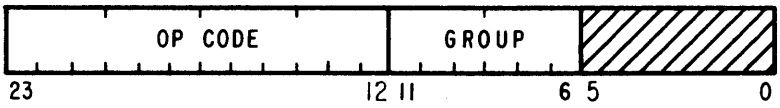


The contents of the request and active registers of the interrupt group replace the previous contents of register D (E and A). The contents of the request register are transferred to E, and the contents of the active register are transferred to A.

Cycles 1

Reference pages 3-1, 3-75

INSTRUCTION FORMULA	FUNCTION		REGISTERS AFFECTED	MNEMONIC
0064.41	Transfer Double to	group 1	1 Request, Active	TD4
0064.42		group 2	1 Request, Active	TD5

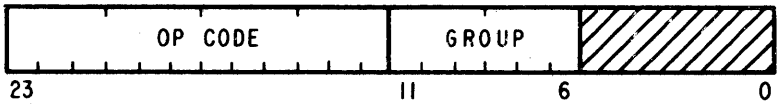


The contents of register D (E and A) are ORed with the current contents of the request and active registers of the interrupt group. The contents of E are ORed with the request register and the contents of A are ORed with the active register.

Cycles.....1

Reference pages.....3-1, 3-75

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0060.01	Unitarily Arm group 1 interrupts	1 A/D	UA1
0060.02	group 2	2 A/D	UA2



Any number of the 24 interrupt levels in the specified group are selectively armed; i.e., the selected bit(s) of the Arm/Disarm (A/D) register are set to ONE.

The corresponding bit(s) of register A must be set to select the appropriate level(s) prior to executing these instructions.

Example: Arm levels 1 and 3, group 1

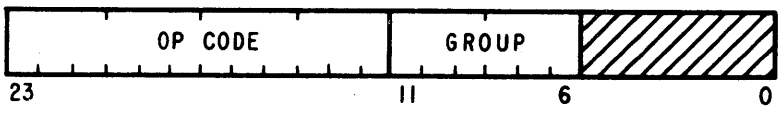
```
TOA   B1B3    Select levels 1 and 3
              (set bits 1 and 3 of A)
UA1           Arm selected levels of group 1
```

NOTE

Execution of these instructions does not clear the interrupt structure and, therefore, does not affect any interrupt levels other than those selected. If a level selected for arming is already armed, it is not cleared by the execution of these instructions.

Cycles 1
Reference pages 3-1, 3-75

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0061.01	Unitarily Disarm group 1 interrupts	1 A/D	UD1
0061.02	group 2	2 A/D	UD2



Any number of the 24 interrupts levels in the specified group are selectively disarmed; i.e., the selected bits of the Arm/Disarm (A/D) register are reset to ZERO.

The corresponding bit(s) of register A must be set to select the appropriate level(s) prior to executing these instructions.

Example: Disarm level 2, group 1

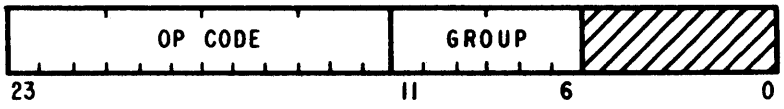
```
TOA  B2     Select level 2 (set bit 2 of A)
UD1         Disarm selected level of group 1
```

NOTE

Execution of these instructions will clear only those levels which are selected. The remaining levels will not be affected.

Cycles 1
Reference pages 3-1, 3-75

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0062.01	Unitarily Enable group 1 interrupts	1 E/I	UE1
0062.02	group 2	2 E/I	UE2



Any number of the 24 interrupt levels in the specified group are selectively enabled; i.e., the selected bits of the Enable/Inhibit (E/I) register are set to ONE.

The corresponding bit(s) of register A must be set to select the appropriate level(s) prior to executing these instructions.

Example: Enable levels 0, 2 and 5, group 1

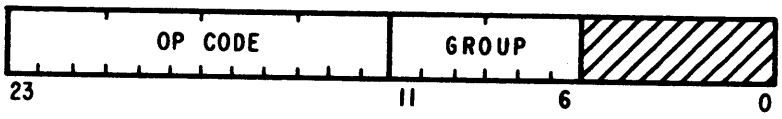
```
TOA   B0B2B5    Select levels 0, 2, 5
                (set bits 0, 2 and 5 of A)
UE1             Enable selected levels of group 1
```

NOTE

Execution of these instructions does not clear the interrupt structure and, therefore, does not affect any interrupt levels other than those selected. If a level selected for enabling is already enabled, it is not cleared by the execution of these instructions.

Cycles 1
Reference pages 3-1, 3-75

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0063.01	Unitarily Inhibit group 1 interrupts	1 E/I	UI1
0063.02	group 2	2 E/I	UI2



Any number of the 24 interrupt levels in the specified group are selectively inhibited; i.e., the selected bits of the Enable/Inhibit (E/I) register are reset to ZERO.

The corresponding bit(s) of register A must be set to select the appropriate level(s) prior to executing these instructions.

Example: Inhibit levels 1, 4 and 7 of group 1

```
TOA   B1B4B7    Select levels 1, 4, 7
                (set bits 1, 4 and 7 of A)
UI1             Inhibit selected levels of group 1
```

NOTE

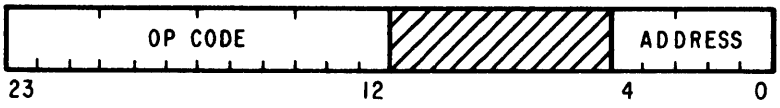
Execution of these instructions does not clear the interrupt structure and, therefore, does not affect any interrupt levels other than those selected. If one or more of the selected levels is active upon execution of these instructions, the level(s) will be placed in a "permissive" state.

Cycles 1
Reference pages 3-1, 3-75

3-12 PROGRAM RESTRICT INSTRUCTIONS

The following instructions provide control for the optional Program Restrict System.

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0067:a	Branch and Link Unrestricted	J,P	BLU



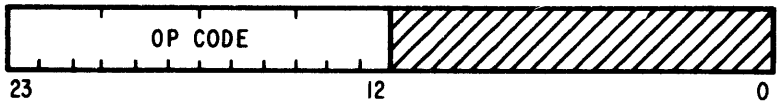
The next sequential address (PROGRAM ADDRESS +1) replaces the contents of register J and the contents of register P (current PROGRAM ADDRESS) are replaced by the 5-bit immediate memory address.

NOTE

Execution of the BLU instruction will turn OFF the Program Restricted Flag (PRF). If the computer is in a HALT condition and the PRF is ON, the BLU instruction will be treated as a NOP instruction.

Cycles 1
Reference pages 2-17, 2-18, 3-1

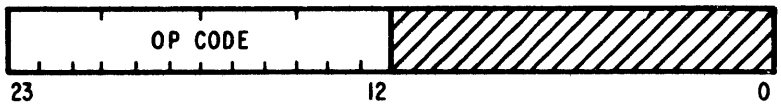
0056.	Transfer Double to Limit registers	LL,UL	TDL



The contents of bits E_0-E_{15} replace the previous contents of the Lower Limit (LL) register and the contents of bits A_0-A_{15} replace the previous contents of the Upper Limit (UL) register. Bits A_{21} and A_{22} set the restrict mode flags.

Cycles 1
Reference pages 2-17, 2-18, 3-1

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0057.	Transfer Limit registers to Double	E,A	TLD



The contents of the Limit registers replace the previous contents of register D (E and A). The Upper Limit register contents are transferred to bits A_0-A_{15} and the contents of the Lower Limit register are transferred to E_0-E_{15}. The states of the restrict mode flags are transferred to bits A_{21} and A_{22}. All other bits in E and A are reset to ZERO.

Cycles 1
Reference pages 2-17, 2-18, 3-1

This Page Left Blank Intentionally.

3-13 SHIFT INSTRUCTIONS

The Shift instruction group consists of arithmetic and logical shifts. The arithmetic shifts cause the contents of a register to be shifted left or right a specified number of times, while preserving the original sign. The logical shifts are similar to the arithmetic shifts, except that the sign bit is shifted along with the other bits.

With both types of shift instructions, any number of shifts from 1 to 255 may be programmed without restriction. The number of shifts (n) are specified in bits 0-7 of the instruction word.

At the conclusion of any shift operation, the Condition register is set to the status of the affected register's contents (Positive, Negative, Zero).

The time required to complete a Shift instruction is determined by the formula:

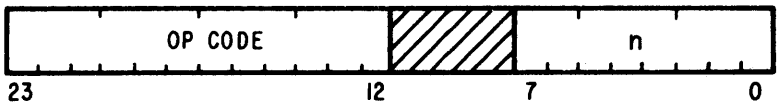
where $n>5$	where $n \leq 5$
$t = 2+(\frac{n-1}{4})$	$t = 2$

t = time in cycles
n = number of shifts

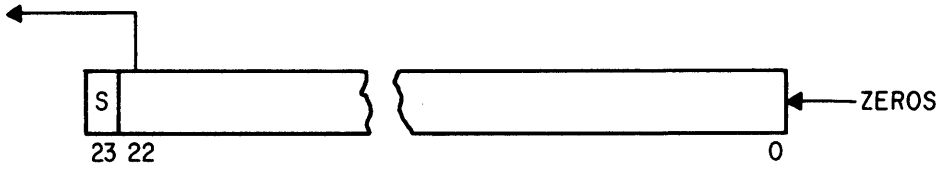
The following instructions are included in the Shift group.

MNEMONIC	INSTRUCTION	PAGE
LAA	Left shift Arithmetic A	3-92
LAD	Left shift Arithmetic Double	3-93
LLA	Left shift Logical A	3-94
LLD	Left shift Logical Double	3-94
LRA	Left Rotate A	3-95
LRD	Left Rotate Double	3-95
RAA	Right shift Arithmetic A	3-96
RAD	Right shift Arithmetic Double	3-96
RLA	Right shift Logical A	3-97
RLD	Right shift Logical Double	3-97
RRA	Right Rotate A	3-98
RRD	Right Rotate Double	3-98

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0040:n	Left shift Arithmetic A	A,C	LAA



Bits A_{22}-A_0 are shifted left n places, with the most significant n bits being lost and n ZEROs being shifted into the least significant bit positions. The sign bit (A_{23}) is unchanged.

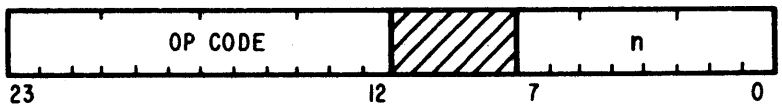


NOTE

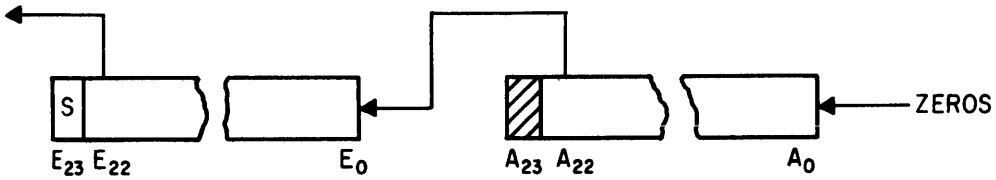
If a bit shifted off from A_{22} differs from the sign bit, the Condition register will be set to OVERFLOW. (This is in addition to the Positive/Negative/Zero status.)

Cycles ..... See Paragraph 3-13.
Reference pages 3-1, 3-91

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0046:n	Left shift Arithmetic Double	E,A,C	LAD



Bits E_{22}-E_0 and A_{22}-A_0 are shifted ——as one register—— left n places. The most significant n bits are lost and the least significant n bits are replaced with ZEROs. Bits E_{23} and A_{23} are bypassed. E_{23} is the register D sign bit and A_{23} is not used in the double-precision format.

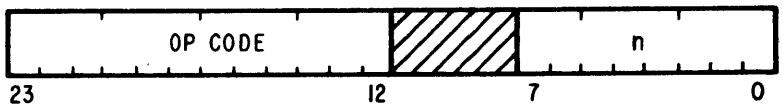


NOTE

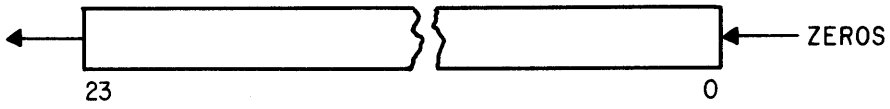
If a bit shifted off from E_{22} differs from the sign bit, the Condition register will be set to OVERFLOW. (This is in addition to the Positive/Negative/Zero status.)

Cycles ..... See Paragraph 3-13.
Reference pages 3-1, 3-91

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0042:n	Left shift Logical A	A,C	LLA

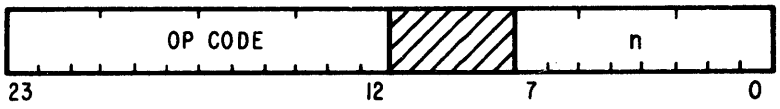


Bits A_{23}-A_0 are shifted left n places, with the most significant n bits being lost and the least significant n bits replaced by ZEROs.

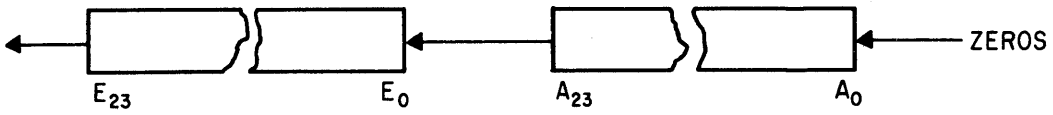


Cycles ..... See Paragraph 3-13.
Reference pages 3-1, 3-91

0050:n	Left shift Logical Double	E,A,C	LLD

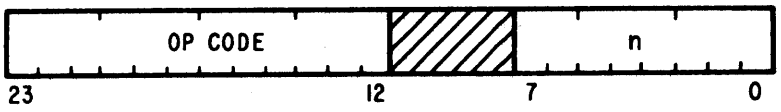


Bits E_{23}-E_0 and A_{23}-A_0 are shifted ——as one register—— left n places. The most significant n bits are lost and the least significant n bits are replaced with ZEROs.

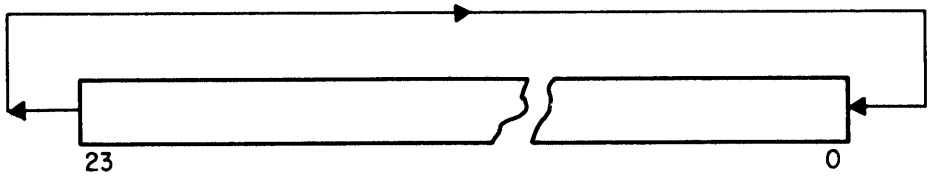


Cycles ..... See Paragraph 3-13.
Reference pages 3-1, 3-91

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0044:n	Left Rotate A	A, C	LRA

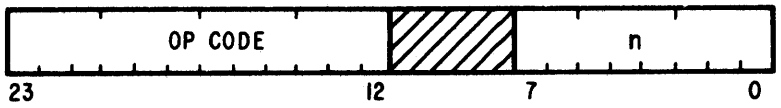


Bits A_{23}-A_0 are rotated left n places. No bits are lost.

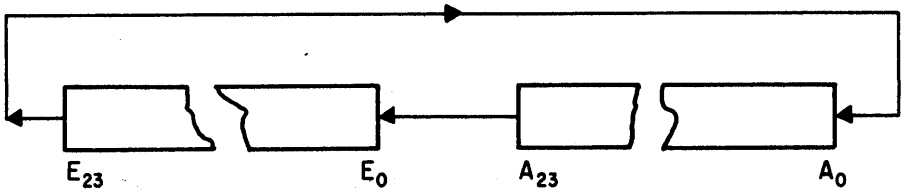


Cycles ..... See Paragraph 3-13.
Reference pages 3-1, 3-91

0052:n Left Rotate Double E, A, C LRD

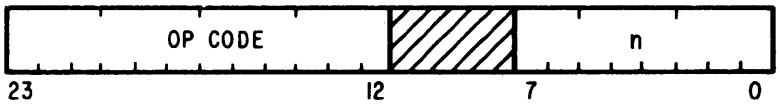


Bits E_{23}-E_0 and A_{23}-A_0 are rotated ——as one register—— left n places, with E_{23} replacing A_0 and A_{23} replacing E_0 as each shift takes place. No bits are lost.

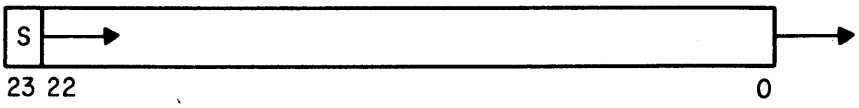


Cycles ..... See Paragraph 3-13.
Reference pages 3-1, 3-91

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0041:n	Right shift Arithmetic A	A,C	RAA

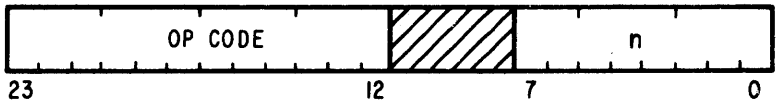


Bits A_{22}-A_0 are shifted right n places. The least significant n bits are lost and the most significant n bits are replaced by an extension of the sign bit (A_{23}). The sign bit is not changed.

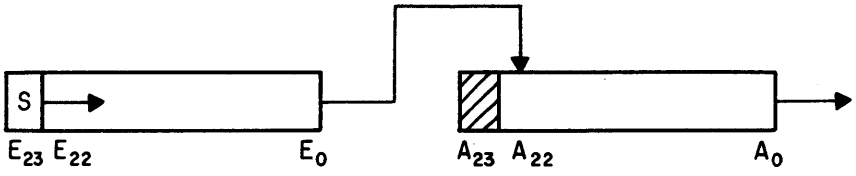


Cycles .....See Paragraph 3-13.
Reference pages 3-1, 3-91

0047:n	Right shift Arithmetic Double	E,A,C	RAD

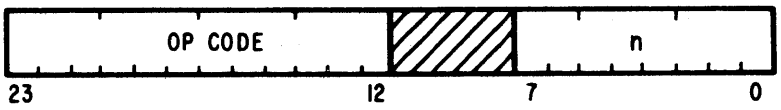


Bits E_{22}-E_0 and A_{22}-A_0 are shifted ——as one register—— right n places. The least significant n bits are lost and the most significant n bits are replaced by an extension of the sign bit (E_{23}). Bit A23 is bypassed.

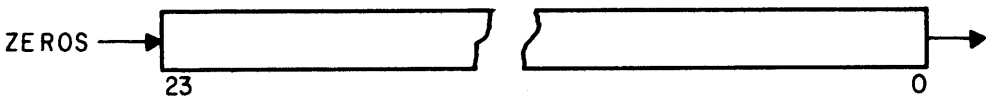


Cycles .....See Paragraph 3-13.
Reference pages 3-1, 3-91

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0043:n	Right shift Logical A	A,C	RLA

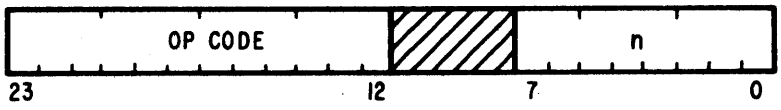


Bits A_{23}-A_0 are shifted right n places. The least significant n bits are lost and the most significant n bits are replaced by ZEROs.

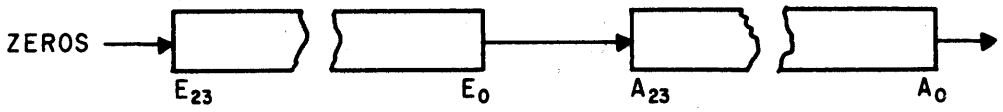


Cycles .....See Paragraph 3-13.
Reference pages 3-1, 3-91

0051:n	Right shift Logical Double	E,A,C	RLD

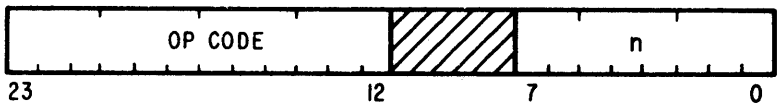


Bits E_{23}-E_0 and A_{23}-A_0 are shifted ——as one register—— right n places. The least significant n bits are lost and the most significant n bits are replaced by ZEROs.

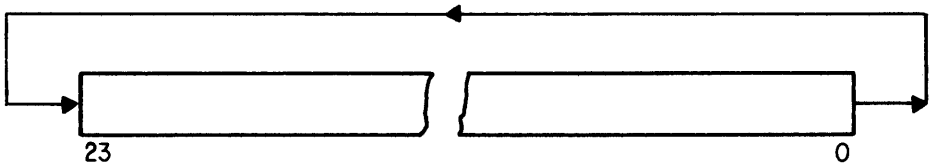


Cycles .....See Paragraph 3-13
Reference pages 3-1, 3-91

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0045:n	Right Rotate A	A,C	RRA

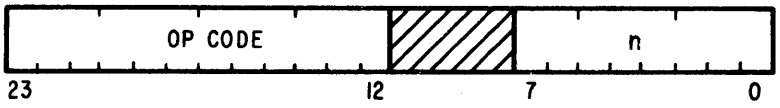


Bits A_{23}-A_0 are rotated right n places. No bits are lost.

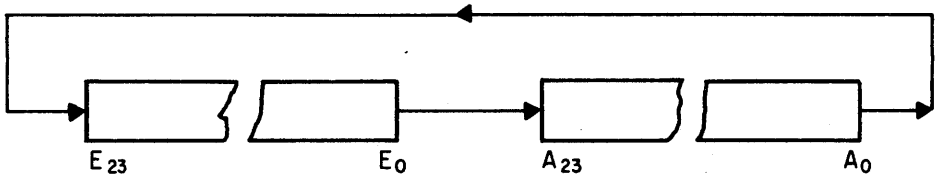


Cycles ..... See Paragraph 3-13.
Reference pages 3-1, 3-91

0053:n Right Rotate Double E,A,C RRD



Bits E_{23}-E_0 and A_{23}-A_0 are rotated ——as one register—— right n places, with E_0 replacing A_{23} and A_0 replacing E_{23} as each shift takes place. No bits are lost.

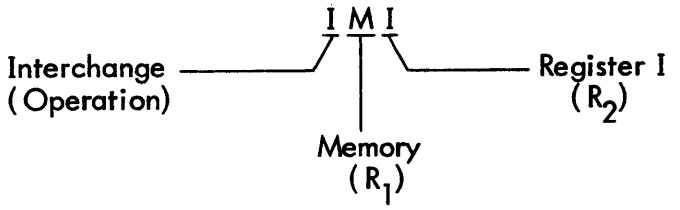


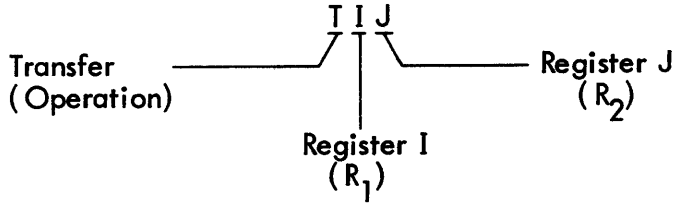
Cycles ..... See Paragraph 3-13.
Reference pages 3-1, 3-91

3-14 TRANSFER INSTRUCTIONS

The Transfer instruction group includes various types of operations. Among these are: interchanges between memory and a specified register, interchanges between registers, memory-to-register and register-to-memory transfers,and register-to-register transfers.

The mnemonic code for the Transfer instruction describes the individual operation. The first letter of the mnemonic indicates what action is to be taken; "I" for Interchange or "T" for Transfer. The second and third letters specify the source (R_1) and destination (R_2) respectively. Some examples are listed below:





With the exception of the Interchange instructions, the transfer source (R_1) is not altered by the execution of any instructions in the Transfer group.

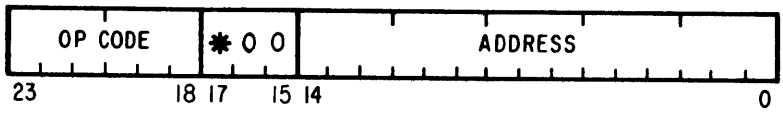
The Condition register is always set to reflect the status (Positive, Negative or Zero) of the contents of R_2 at the completion of the instruction.

The following instructions are included in the Transfer group.

MNEMONIC	INSTRUCTION	PAGE
IJE	Interchange J and E	3-105
IJI	Interchange J and I	3-105
IJK	Interchange J and K	3-105
IJT	Interchange J and T	3-105
IKA	Interchange K and A	3-105
IKE	Interchange K and E	3-105
IKI	Interchange K and I	3-105
IKJ	Interchange K and J	3-105
IKT	Interchange K and T	3-105
IMA	Interchange M and A	3-104
IME	Interchange M and E	3-104
IMI	Interchange M and I	3-104
IMJ	Interchange M and J	3-104
IMK	Interchange M and K	3-104
ITA	Interchange T and A	3-105
ITE	Interchange T and E	3-105
ITI	Interchange T and I	3-105
ITJ	Interchange T and J	3-105
ITK	Interchange T and K	3-105
RBM	Replace Byte in Memory	3-106
T1D	Transfer group 1 to Double	3-119
T2D	Transfer group 2 to Double	3-119
T4D	Transfer group 1 to Double	3-121
T5D	Transfer group 2 to Double	3-121
TAE	Transfer A to E	3-117
TAI	Transfer A to I	3-117
TAJ	Transfer A to J	3-117
TAK	Transfer A to K	3-117
TAM	Transfer A to Memory	3-116
TAT	Transfer A to T	3-117
TBM	Transfer Byte to Memory	3-114
TD1	Transfer Double to group 1	3-119
TD2	Transfer Double to group 2	3-119
TD4	Transfer Double to group 1	3-120
TD5	Transfer Double to group 2	3-120
TDL	Transfer Double to Limit registers	3-118
TDM	Transfer Double to Memory	3-114
TEA	Transfer E to A	3-117
TEB	Transfer E to Byte	3-113
TEI	Transfer E to I	3-117
TEJ	Transfer E to J	3-117
TEK	Transfer E to K	3-117
TEM	Transfer E to Memory	3-116
TET	Transfer E to T	3-117
TFM	Transfer Flag to Memory	3-115
TIA	Transfer I to A	3-117
TIB	Transfer I to Byte	3-113
TIE	Transfer I to E	3-117
TIJ	Transfer I to J	3-117
TIK	Transfer I to K	3-117
TIM	Transfer I to Memory	3-116
TIT	Transfer I to T	3-117
TJA	Transfer J to A	3-117
TJB	Transfer J to Byte	3-113

MNEMONIC	INSTRUCTION	PAGE
TJE	Transfer J to E	3-117
TJI	Transfer J to I	3-117
TJK	Transfer J to K	3-117
TJM	Transfer J to Memory	3-116
TJT	Transfer J to T	3-117
TKA	Transfer K to A	3-117
TKB	Transfer K to Byte	3-113
TKE	Transfer K to E	3-117
TKI	Transfer K to I	3-117
TKJ	Transfer K to J	3-117
TKM	Transfer K to Memory	3-116
TKT	Transfer K to T	3-117
TLD	Transfer Limit registers to Double	3-118
TLO	Transfer Long Operand to K	3-112
TMA	Transfer Memory to A	3-109
TMB	Transfer Memory to Byte	3-107
TMD	Transfer Memory to Double	3-107
TME	Transfer Memory to E	3-109
TMI	Transfer Memory to I	3-109
TMJ	Transfer Memory to J	3-109
TMK	Transfer Memory to K	3-109
TMQ	Transfer Memory to Query register	3-108
TMR	Transfer Memory to Registers	3-109
TNA	Transfer Negative operand to A	3-110
TNE	Transfer Negative operand to E	3-110
TNI	Transfer Negative operand to I	3-110
TNJ	Transfer Negative operand to J	3-110
TNK	Transfer Negative operand to K	3-110
TNT	Transfer Negative operand to T	3-110
TOA	Transfer Operand to A	3-111
TOB	Transfer Operand to Byte	3-110
TOC	Transfer Operand to Condition Register	3-111
TOE	Transfer Operand to E	3-111
TOI	Transfer Operand to I	3-111
TOJ	Transfer Operand to J	3-111
TOK	Transfer Operand to K	3-111
TOT	Transfer Operand to T	3-111
TRM	Transfer Registers to Memory	3-116
TSA	Transfer Switches to A	3-112
TSE	Transfer Switches to E	3-112
TSI	Transfer Switches to I	3-112
TSJ	Transfer Switches to J	3-112
TSK	Transfer Switches to K	3-112
TST	Transfer Switches to T	3-112
TTA	Transfer T to A	3-117
TTB	Transfer T to Byte	3-113
TTE	Transfer T to E	3-117
TTI	Transfer T to I	3-117
TTJ	Transfer T to J	3-117
TTK	Transfer T to K	3-117
TZA	Transfer Zero to A	3-113

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
31.*+0:a	Extract Memory Byte	B,C	EMB



The effective memory address is added to the contents of register J, producing the word address which contains the byte to be extracted. The selected byte, as determined by the contents of bits 22 and 23 of index register J, is then placed in register B. The following table shows the correspondence between bits 22 and 23 of J and the byte to be extracted:

Bits 22 and 23 J Register	Byte Selection
01	Leftmost byte (bits 16-23 of EMA+J)
10	Middle byte (bits 8-15 of EMA+J)
11	Rightmost byte (bits 0-7 of EMA+J)
00	Rightmost byte (bits 0-7 of EMA+J)

The final address of any indirect index sequence should not be indexed since implied indexing on register J takes place. If indexing is specified on the final address, then the specified index register will be logically ORed with register J prior to the add function with the EMA.

Example:

If J = '4ØØØØØ3Ø

and K = 'ØØØØØØ1Ø when the following is executed:

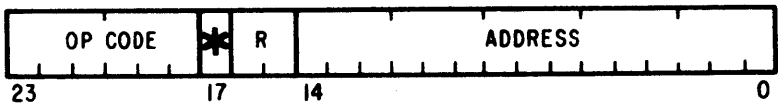
```
        EMB*   '40
'4Ø     DAC*   '5Ø,K
'42     DATA   "XYZ"
'6Ø     DAC    '12
```

Then the character Y will be placed in register B. Note that the effective address of the indirect/index sequence is '12. However, '12 plus bits 0-15 of index register J ('3Ø) yields the final address of '42. Since a byte specification of 10_2 was made in bits 22-23 of index register J, then the second byte (bits 8-15) of memory location '42 is placed in register B.

Cycles 2

Reference Pages 3-1, 3-2, 3-99

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
66. *+1:a	Interchange Memory and I	M,I,C	IMI
66. *+2:a	J	M,J,C	IMJ
66. *+3:a	K	M,K,C	IMK



The contents of the effective memory address and register I, J, or K are interchanged.

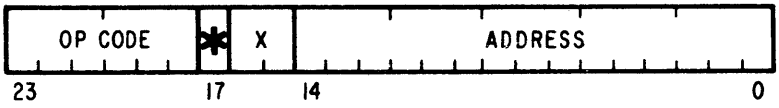
NOTE

The immediate memory reference cannot be indexed; however, indexing of indirect references is permitted, e.g.,

```
      IMK*   X
       .
       .
X     DAC    Y,J
```

Cycles .....3 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-99

67. *+X:a	Interchange Memory and E	M,E,C	IME
70. *+X:a	A	M,A,C	IMA



The contents of the effective memory address and the specified register are interchanged.

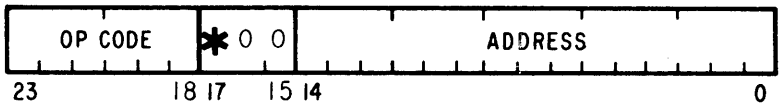
Cycles .....3 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-99

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0035.01.02	Interchange I and J	I,J,C	IIJ
0035.01.04	K	I,K,C	IIK
0035.01.10	E	I,E,C	IIE
0035.01.20	A	I,A,C	IIA
0035.01.40	T	I,T,C	IIT
0035.02.01	Interchange J and I	J,I,C	IJI
0035.02.04	K	J,K,C	IJK
0035.02.10	E	J,E,C	IJE
0035.02.20	A	J,A,C	IJA
0035.02.40	T	J,T,C	IJT
0035.04.01	Interchange K and I	K,T,C	IKI
0035.04.02	J	K,J,C	IKJ
0035.04.10	E	K,E,C	IKE
0035.04.20	A	K,A,C	IKA
0035.04.40	T	K,T,C	IKT
0035.10.01	Interchange E and I	E,I,C	IEI
0035.10.02	J	E,J,C	IEJ
0035.10.04	K	E,K,C	IEK
0035.10.20	A	E,A,C	IEA
0035.10.40	T	E,T,C	IET
0035.20.01	Interchange A and I	A,I,C	IAI
0035.20.02	J	A,J,C	IAJ
0035.20.04	K	A,K,C	IAK
0035.20.10	E	A,E,C	IAE
0035.20.40	T	A,T,C	IAT
0035.40.01	Interchange T and I	T,I,C	ITI
0035.40.02	J	T,J,C	ITJ
0035.40.04	K	T,K,C	ITK
0035.40.10	E	T,E,C	ITE
0035.40.20	A	T,A,C	ITA

The contents of R_1 and R_2 are interchanged.

Cycles 1
Reference pages 3-1, 3-99

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
27.*+0:a	Replace Byte in Memory	M	RBM



The effective memory address is added to the contents of register I producing the word address which contains the byte to be replaced. The selected byte, as determined by the contents of bits 22 and 23 of index register I, is then replaced by the contents of register B. The following table shows the correspondence between bits 22 and 23 of I and the byte to be replaced.

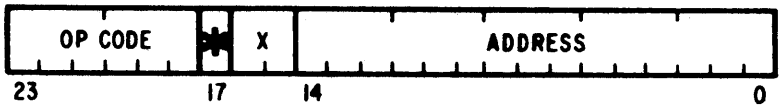
Bits 23 and 22 I Register	Byte Selection
01	Leftmost byte (bits 16-23 of EMA+I)
10	Middle byte (bits 8-15 of EMA+I)
11	Rightmost byte (bits 0-7 of EMA+I)
00	Causes no operation.

The final address of any indirect/index sequence should not be indexed since implied indexing on register I takes place. If indexing is specified on the final address, then the specified index register will be logically ORed with register I prior to the add function with the EMA.

Cycles 3

Reference Pages 3-1, 3-2, 3-99

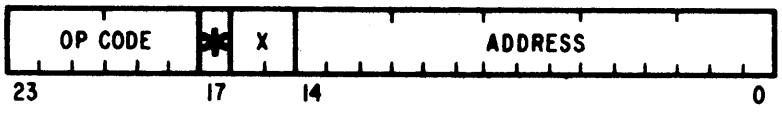
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
07. *+X:a	Transfer Memory to Byte	A, C	TMB



The 8 least significant bits (0-7) of the contents of the effective memory address replace the previous contents of register B (A_0-A_7). Bits A_8-A_{23} are unaffected.

Cycles .....2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-99

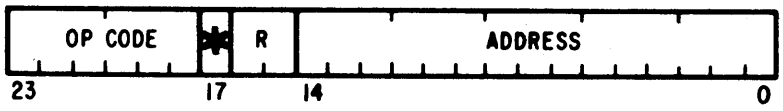
06. *+X:a	Transfer Memory to Double	E, A, C	TMD



The contents of the effective memory address (EMA) and the next sequential address (EMA +1) replace the previous contents of register D (E and A). EMA and EMA +1 are transferred to E and A, respectively.

Cycles .....3 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-99

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
51. *+0:a	Transfer Memory to Query register	Q	TMQ



Bits 0-15 of the contents of the effective memory address replace the previous contents of the Query register.

Executing this instruction will cause the Address Trap to be enabled or disabled, depending on the state of bit 23 of the effective memory address.

Bit 23 = ONE = Disable Address Trap
Bit 23 = ZERO = Enable Address Trap

Example:

```
        TMQ   OA
        ...
OA      DAC   ADDR     Enable Address Trap

        or

OA      DAC*  O        Disable Address Trap
```

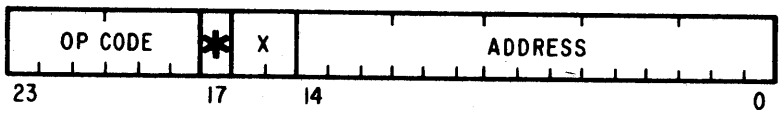
NOTE

The immediate memory reference cannot be indexed; however, indexing of indirect references is permitted, e.g.,

```
        TMQ*  X
        .
        .
X       DAC   Y,I
```

Cycles ..... 2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-99

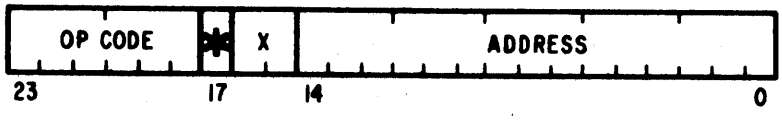
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
01. *+X:a	Transfer Memory to I	I, C	TMI
02. *+X:a	J	J, C	TMJ
03. *+X:a	K	K, C	TMK
04. *+X:a	E	E, C	TME
05. *+X:a	A	A, C	TMA



The contents of the effective memory address replace the previous contents of the specified register.

Cycles ..... 2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-99

10. *+X:a	Transfer Memory to Registers	I, J, K, E, A	TMR



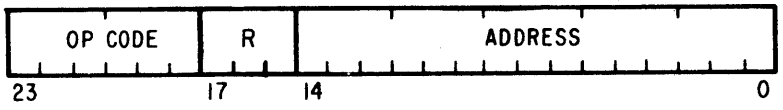
Registers I, J, K, E and A are loaded from consecutive memory addresses beginning with the effective memory address.

NOTE

External interrupts are prohibited for the period of one instruction following the execution of this instruction.

Cycles ..... 6 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-99

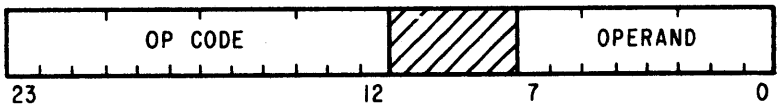
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
63.1:o	Transfer Negative operand to I	I,C	TNI
63.2:o	J	J,C	TNJ
63.3:o	K	K,C	TNK
63.4:o	E	E,C	TNE
63.5:o	A	A,C	TNA
63.6:o	T	T,C	TNT



The two's complement of the 15-bit unsigned operand replaces the previous contents of bits 0-23 of the specified register.

Cycles 1
Reference pages 3-1, 3-2, 3-99

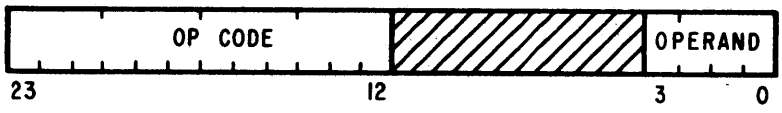
0003:o	Transfer Operand to Byte	A,C	TOB



The 8-bit operand replaces the previous contents of register B (A_0-A_7). Bits A_8-A_{23} are unaffected.

Cycles 1
Reference pages 3-1, 3-99

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0036:o	Transfer Operand to Condition register	C	TOC



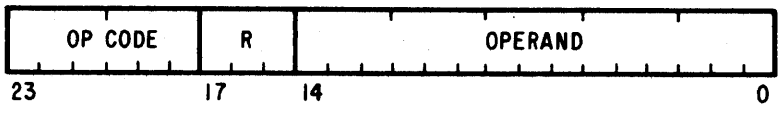
The 4-bit operand replaces the previous contents of the Condition register.

Operand definition is as follows:

bit 0 - ONE = Overflow,	ZERO = No Overflow
bit 1 - ONE = Negative,	ZERO = Not Negative
bit 2 - ONE = Zero,	ZERO = Not Zero
bit 3 - ONE = Positive,	ZERO = Not Positive

Cycles 1
Reference pages 3-1, 3-99

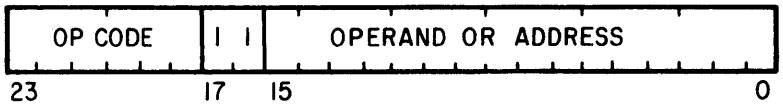
62.1:o	Transfer Operand to I	I, C	TOI
62.2:o	J	J, C	TOJ
62.3:o	K	K, C	TOK
62.4:o	E	E, C	TOE
62.5:o	A	A, C	TOA
62.6:o	T	T, C	TOT



The 15-bit operand replaces the previous contents of bits 0-23 of the specified register.

Cycles 1
Reference pages 3-1, 3-99

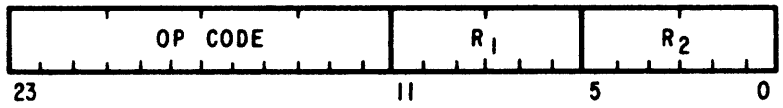
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
23.6:A	Transfer Long Operand to K	K	TLO



The 16-bit operand (or address) replaces the previous contents of bits 0-23 of register K.

Cycles 1
Reference pages 3-1, 3-2, 3-99

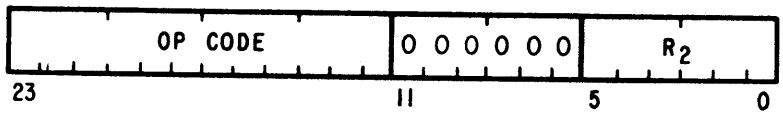
0031.00.01	Transfer Switches to I	I,C	TSI
0031.00.02	J	J,C	TSJ
0031.00.04	K	K,C	TSK
0031.00.10	E	E,C	TSE
0031.00.20	A	A,C	TSA
0031.00.40	T	T,C	TST



The states (set = ONE) of the console control switches (i.e., switch register) are transferred to the corresponding bit positions of the specified register.

Cycles 1
Reference pages 3-1, 3-99

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0030.00.01	Transfer Zero to I	I,C	TZI
0030.00.02	J	J,C	TZJ
0030.00.04	K	K,C	TZK
0030.00.10	E	E,C	TZE
0030.00.20	A	A,C	TZA
0030.00.40	T	T,C	TZT
0030.00.30	Double	E,A,C	TZD



The previous contents of the specified register are replaced with ZEROs.

Cycles 1
Reference pages 3-1, 3-99

0002.01	Transfer I to Byte	A	TIB
0002.02	J	A	TJB
0002.04	K	A	TKB
0002.10	E	A	TEB
0002.40	T	A	TTB

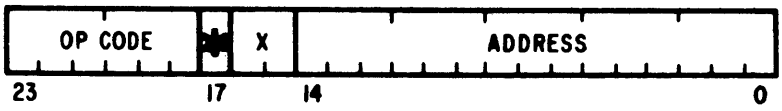
The least significant 8 bits (0-7) of the contents of the specified register replace the previous contents of register B (A_0-A_7). Bits A_8-A_{23} are unchanged.

NOTE

The Condition register is not affected.

Cycles 1
Reference pages 3-1, 3-99

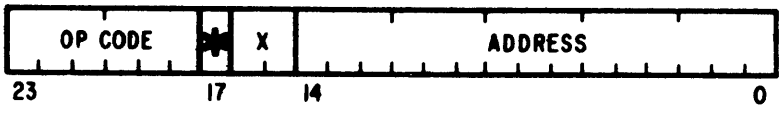
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
17. *+X:a	Transfer Byte to Memory	M	TBM



The contents of register B (A_0-A_7) replace the 8 least significant bits (0-7) of the contents of the effective memory address. Bits 8-23 of the memory word are unaffected.

Cycles .....3 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-99

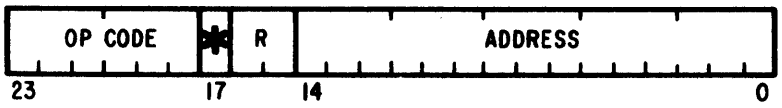
16. *+X:a	Transfer Double to Memory	M	TDM



The contents of register D (E and A) replace the previous contents of the effective memory address (EMA) and the next sequential address (EMA +1). The contents of E and A are transferred to EMA and EMA +1, respectively.

Cycles .....3 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-99

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
46. *+0:a	Transfer Flag to Memory	M, C	TFM



The previous contents of the effective memory address are replaced by ONEs.

NOTES

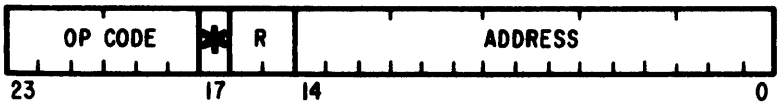
(1) The Condition (C) register is set to the status of memory prior to the transter.

(2) The immediate memory reference cannot be indexed; however, indexing of indirect is permitted, e.g.,

```
        TFM*    X
         .
         .
X       DAC     Y,I
```

Cycles ..... 3 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-99

66. *+0:a Transfer Zero to Memory M, C TZM



The previous contents of the effective memory address are replaced by ZEROs.

NOTES

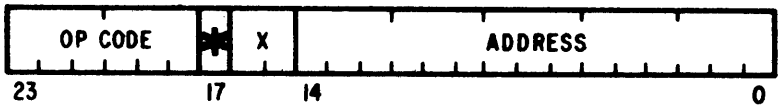
(1) The Condition (C) register is set to the status of memory prior to the transfer.

(2) The immediate memory reference cannot be indexed; however, indexing of indirect references is permitted, e.g.,

```
        TZM*    X
         .
         .
X       DAC     Y,I
```

Cycles ..... 3 (+1 per indirect reference)
Reference pages..... 3-1, 3-2, 3-99

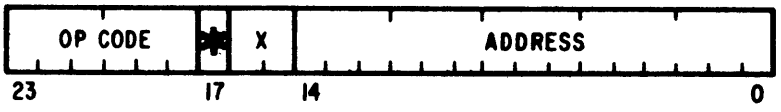
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
11. *+X:a	Transfer I to Memory	M	TIM
12. *+X:a	J	M	TJM
13. *+X:a	K	M	TKM
14. *+X:a	E	M	TEM
15. *+X:a	A	M	TAM



The contents of the specified register replace the previous contents of the effective memory address.

Cycles .....2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-99

20. *+X:a Transfer Registers to Memory M TRM



The contents of registers I, J, K, E and A are stored in consecutive memory locations beginning with the effective memory address.

NOTE

External interrupts are prohibited for the period of one instruction following the execution of this instruction.

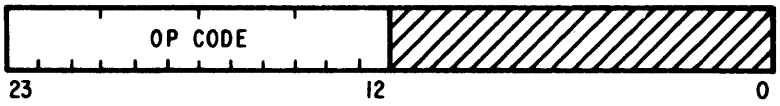
Cycles .....6 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-99

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0030.01.02	Transfer I to J	J,C	TIJ
0030.01.04	K	K,C	TIK
0030.01.10	E	E,C	TIE
0030.01.20	A	A,C	TIA
0030.01.40	T	T,C	TIT
0030.02.01	Transfer J to I	I,C	TJI
0030.02.04	K	K,C	TJK
0030.02.10	E	E,C	TJE
0030.02.20	A	A,C	TJA
0030.02.40	T	T,C	TJT
0030.04.01	Transfer K to I	I,C	TKI
0030.04.02	J	J,C	TKJ
0030.04.10	E	E,C	TKE
0030.04.20	A	A,C	TKA
0030.04.40	T	T,C	TKT
0030.10.01	Transfer E to I	I,C	TEI
0030.10.02	J	J,C	TEJ
0030.10.04	K	K,C	TEK
0030.10.20	A	A,C	TEA
0030.10.40	T	T,C	TET
0030.20.01	Transfer A to I	I,C	TAI
0030.20.02	J	J,C	TAJ
0030.20.04	K	K,C	TAK
0030.20.10	E	E,C	TAE
0030.20.40	T	T,C	TAT
0030.40.01	Transfer T to I	I,C	TTI
0030.40.02	J	J,C	TTJ
0030.40.04	K	K,C	TTK
0030.40.10	E	E,C	TTE
0030.40.20	A	A,C	TTA

The contents of R_1 replace the previous contents of R_2.

Cycles 1
Reference pages 3-1, 3-99

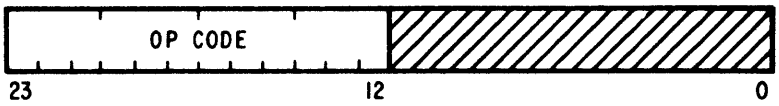
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0056.	Transfer Double to Limit registers	LL,UL	TDL



The contents of bits E_0-E_{15} replace the previous contents of the Lower Limit (LL) register and the contents of bits A_0-A_{15} replace the previous contents of the Upper Limit (UL) register. Bits A_{21} and A_{22} set the restrict mode flags.

Cycles 1
Reference pages 3-1, 3-99

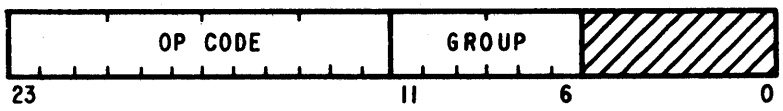
0057.	Transfer Limit registers to Double	E,A	TLD



The contents of the Limit registers replace the previous contents of register D (E and A). The Upper Limit register contents are transferred to bits A_0-A_{15} and the contents of the Lower Limit register are transferred to E_0-E_{15}. The states of the restrict mode flags are transferred to bits A_{21} and A_{22}. All other bits in E and A are reset to ZERO.

Cycles 1
Reference pages 3-1, 3-99

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0064.01	Transfer Double to group 1	1 A/D, 1 E/I	TD1
0064.02	group 2	2 A/D, 2 E/I	TD2



The contents of register D (E and A) replace the previous contents of the Arm/Disarm (A/D) and Enable/Inhibit (E/I) registers of the specified interrupt group. The contents of E are transferred to the A/D register and the contents of A are transferred to the E/I register.

NOTE

The external interrupt structure is cleared by the execution of these instructions.

Cycles 1
Reference pages 3-1, 3-99

0065.01	Transfer group 1 to Double	E,A	T1D
0065.02	group 2	E,A	T2D

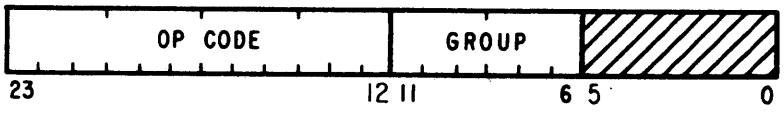
The contents of the Arm/Disarm (A/D) and Enable/Inhibit (E/I) registers of the specified interrupt group replace the previous contents of register D (E and A). The contents of the A/D register are transferred to register E and the contents of the E/I register are transferred to register A.

NOTE

The states of the external interrupts are not affected by the execution of these instructions.

Cycles 1
Reference pages 3-1, 3-99

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0064.41	Transfer Double to Group 1	1 Request, Active	TD4
0064.42	Group 2	2 Request, Active	T5D

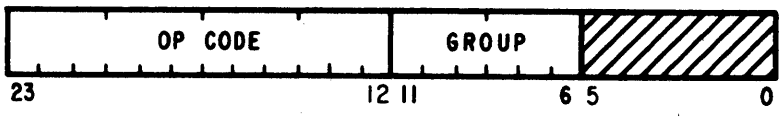


The contents of register D (E and A) are ORed with the current contents of the request and active registers of the specified interrupt group. The contents of E are ORed with the request register and the contents of A are ORed with the active register.

Cycles.....1

Reference pages.....3-1, 3-75

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0065.41	Transfer Group 1 to Double	E, A	T4D
0065.42	Group 2	E, A	T5D



The contents of the request and active registers of the specified interrupt group replace the previous contents of register D (E and A). The contents of the request register are transferred to E, and the contents of the active register are transferred to A.

Cycles 1

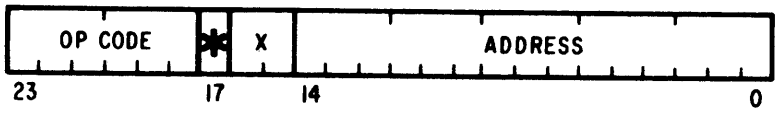
Reference pages 3-1, 3-75

This Page Left Blank Intentionally.

3-15 BYTE PROCESSING INSTRUCTIONS

The Byte Processing group of instructions permits program manipulation of all three bytes within the computer word (24 bits); e. g., extract, replace, etc. The following instructions are inclusive of byte processing operations:

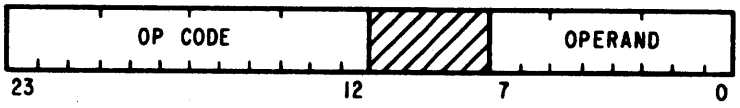
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
45. *+X:a	Add Memory to Byte	A, C	AMB



Bits 0-7 of the contents of the effective memory address are algebraically added to the contents of register B (A_0-A_7). Bits 8-23 of register A are unchanged.

Cycles ..... 2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-5

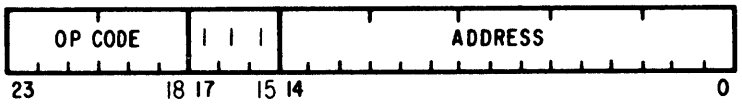
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0012. o	Add Operand to Byte	A,C	AOB



The 8-bit signed operand is algebraically added to the contents of the B register. (A_0-A_7). Bits 8-23 of register A are unchanged.

Cycles 1
References pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
60.7:a	Branch when Byte Address +1 in I $\neq$ 0	I	BBI
61.7:a	Branch when Byte Address +1 in J $\neq$ 0	J	BBJ



The contents of bits 22 and 23 of the specified index register (I or J) is incremented by one. If the result of this addition (in bits 22 and 23) is not 00_2, then the contents of register P (current PROGRAM ADDRESS) is replaced by the 15-bit effective memory address. If the result of the addition to bits 22 and 23 is 00_2, then bits 22 and 23 are set to 01_2 and bits 0-21 are incremented by one. If the resultant sum in bits 0-21 is zero, then register P advances to the next sequential program location and the index register is set to 40000000_8. Otherwise, the contents of register P are replaced by the 15-bit effective memory address.

In general, the BBI and BBJ instructions are used as special index register increments in order to sequentially reference consecutive bytes in memory via the EMB and RBM instructions. Consider the following example which will move 11 consecutive bytes starting from the third byte at location '200 to the first byte at location '300.

Example:

```
TMJ   = '60000200
TMI   = '20000300
TNK   11
EMB   0
RBM   0
BBI   *+1
BBJ   *+1
BWK   *-4
```

Occasionally, it is possible to use the address of a portion of index register I or J as a byte counter as well as a word pointer. This may be illustrated by the following example which will set the buffer starting at byte 3 of location '100 through byte 3 of location '102 to blanks.

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC

Example:

```
TOB   "Ø"
TMI   = '77777775      bits 22 and 23 = 3, bits 0-21 = -3
RBM   '100+3
BBI   *-1
```

However, it should be noted this technique of using the index register as both a byte counter and word pointer may be used only in certain instances. Specifically, when the following relationship is true.

$$R\left(\frac{4-b.n.}{3}\right) = R\left(\frac{CT}{3}\right)$$

where: R ( ) = remainder

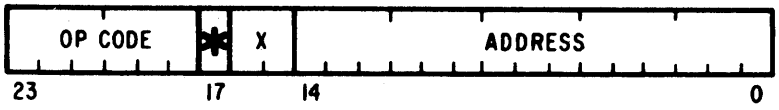
b.n. = the starting byte number (1, 2, or 3)

CT = the number of bytes to be referenced.

Cycles. 1

Reference Pages. . 3-1, 3-2, 3-35

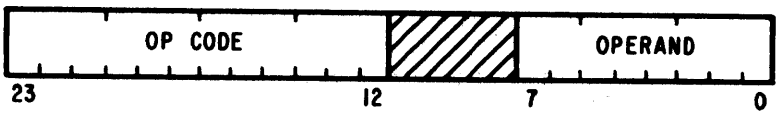
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
34. *+X:a	Compare Memory and Byte	C	CMB



The contents of register B (A_0-A_7) and the contents of the effective memory address (M_0-M_7) are algebraically compared and the Condition register is set to the status of the result.

Cycles ..... 2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-45

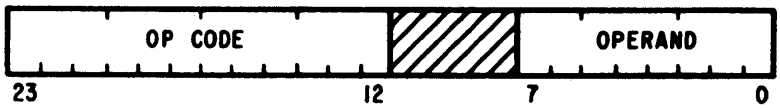
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0014:o	Compare Operand and Byte	C	COB



The 8-bit signed operand and the contents of register B (A_0-A_7) are algebraically compared and the Condition register is set to the status of the result.

Cycles 1
Reference pages 3-1, 3-45

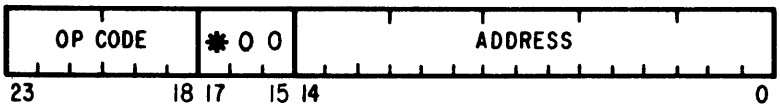
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0016:o	Dot Operand with Byte	A,C	DOB



A logical AND is performed between the 8-bit operand and the contents of register B (A_0-A_7). Bits A_8-A_{23} are unchanged.

Cycles 1
Reference pages 3-1, 3-65

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
31. *+0:a	Extract Memory Byte	B,C	EMB



The effective memory address is added to the contents of register J, producing the word address which contains the byte to be extracted. The selected byte, as determined by the contents of bits 22 and 23 of index register J, is then placed in register B. The following table shows the correspondence between bits 22 and 23 of J and the byte to be extracted:

Bits 22 and 23 J Register	Byte Selection
01	Leftmost byte (bits 16-23 of EMA+J)
10	Middle byte (bits 8-15 of EMA+J)
11	Rightmost byte (bits 0-7 of EMA+J)
00	Rightmost byte (bits 0-7 of EMA+J)

The final address of any indirect index sequence should not be indexed since implied indexing on register J takes place. If indexing is specified on the final address, then the specified index register will be logically ORed with register J prior to the add function with the EMA.

Example:

If J = '4ØØØØØ3Ø

and K = 'ØØØØØ1Ø when the following is executed:

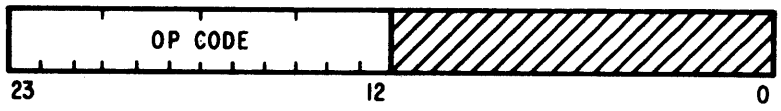
```
        EMB*   '40
'4Ø     DAC*   '5Ø,K
'42     DATA   "XYZ"
'6Ø     DAC    '12
```

Then the character Y will be placed in register B. Note that the effective address of the indirect/index sequence is '12. However, '12 plus bits 0-15 of index register J ('3Ø) yields the final address of '42. Since a byte specification of 10_2 was made in bits 22-23 of index register J, then the second byte (bits 8-15) of memory location '42 is placed in register B.

Cycles 2

Reference Pages 3-1, 3-2, 3-99

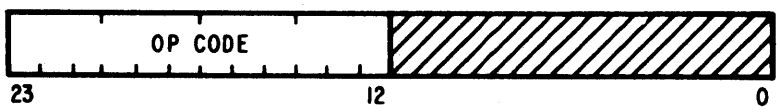
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0010.	Extend Sign of Byte	A,C	ESB



The state of the register B sign bit (A_7) is copied into bit positions A_8–A_{23}, forming a sign extension of the byte.

Cycles 1
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0007.	Extend Zeros from Byte	A	EZB



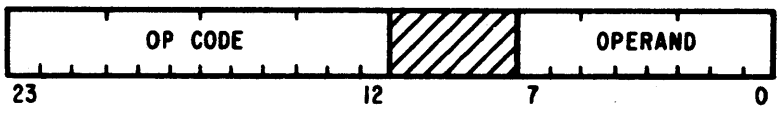
Bit positions A_8-A_{23} are set to ZERO. The contents of register B (A_0-A_7) are not affected.

NOTE

The Condition register is not affected.

Cycles 1
Reference page 3-1

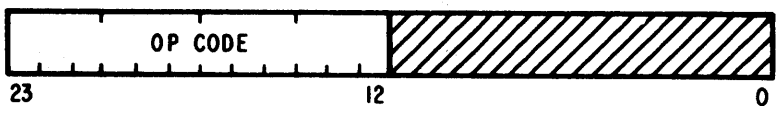
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0015:o	Kompare Operand and Byte	C	KOB



The 8-bit operand and the contents of register B (A_0-A_7) are logically compared and the Condition register is set according to the result.

Cycles 1
Reference pages 3-1, 3-45

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0005.	Negate of Byte to Byte	A, C	NBB



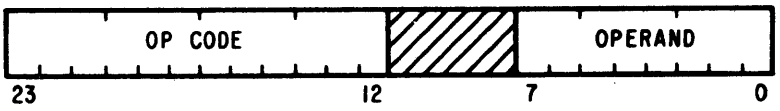
The contents of register B (A_0-A_7) are two's complemented. Bit positions A_8-A_{23} are unchanged.

NOTE

An Overflow will result when negating 2^7 (full-scale negative byte).

Cycles 1
Reference pages 3-1, 3-5

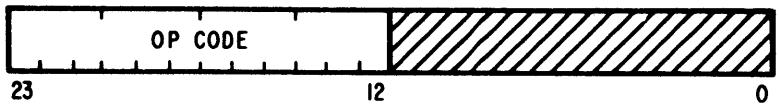
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0004:o	Or Operand with Byte	A,C	OOB



A logical OR is performed between the 8-bit operand and the contents of register B (A_0-A_7). Bits A_8-A_{23} are unchanged.

Cycles 1
Reference pages 3-1, 3-65

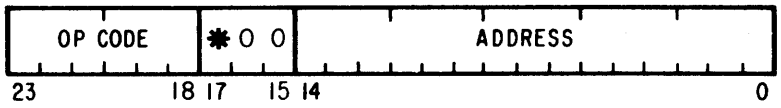
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0006.	Postive of Byte to Byte	A,C	PBB



The absolute value of the contents of register B (A_0-A_7) is placed in register B.

Cycles 1
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
27.*+0:a	Replace Byte in Memory	M	RBM



The effective memory address is added to the contents of register I producing the word address which contains the byte to be replaced. The selected byte, as determined by the contents of bits 22 and 23 of index register I, is then replaced by the contents of register B. The following table shows the correspondence between bits 22 and 23 of I and the byte to be replaced.

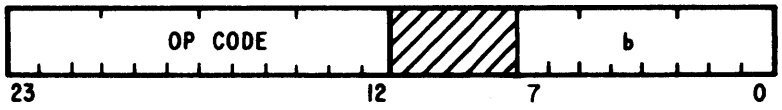
Bits 23 and 22 I Register	Byte Selection
01	Leftmost byte (bits 16-23 of EMA+I)
10	Middle byte (bits 8-15 of EMA+I)
11	Rightmost byte (bits 0-7 of EMA+I)
00	Causes no operation.

The final address of any indirect/index sequence should not be indexed since implied indexing on register I takes place. If indexing is specified on the final address, then the specified index register will be logically ORed with register I prior to the add function with the EMA.

Cycles 3

Reference Pages 3-1, 3-2, 3-99

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0011:b	Query Bits of Byte	C	QBB



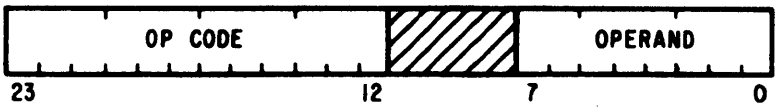
A logical AND is performed between operand bits 0-7 and the contents of register B. The Condition register is set according to the status of the result; i.e., positive, negative, or zero.

Examples:

```
1)  TOA     B7      A = '00000200     C = Positive
    . . .
    . . .
    QBB     B7                        C = Negative
2)  TOA     B6      A = '00000100     C = Positive
    . . .
    . . .
    QBB     B6                        C = Positive
3)  TNA     1       A = '77777777     C = Negative
    . . .
    . . .
    DMA     MASK    A = '40000000     C = Negative
    . . .
    . . .
MASK DATA   '40000000
```

Cycles 1
Reference page 3-1

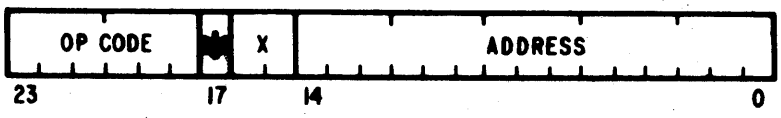
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0013:o	Subtract Operand from Byte	A,C	SOB



The 8-bit signed operand is algebraically subtracted from the contents of register B (A_0-A_7). Bits A_8-A_{23} are unaffected.

Cycles 1
Reference pages 3-1, 3-5

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
17. *+X:a	Transfer Byte to Memory	M	TBM



The contents of register B (A_0-A_7) replace the 8 least significant bits (0-7) of the contents of the effective memory address. Bits 8-23 of the memory word are unaffected.

Cycles .....3 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-99

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0002.01	Transfer I to Byte	A	TIB
0002.02	J	A	TJB
0002.04	K	A	TKB
0002.10	E	A	TEB
0002.40	T	A	TTB

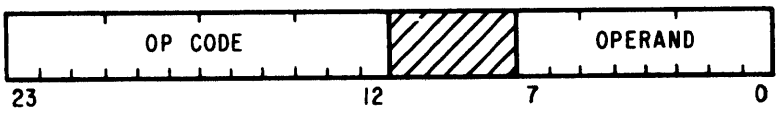
The least significant 8 bits (0-7) of the contents of the specified register replace the previous contents of register B (A_0-A_7). Bits A_8-A_{23} are unchanged.

NOTE

The Condition register is not affected.

Cycles 1
Reference pages 3-1, 3-99

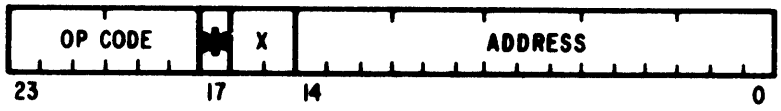
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0003:o	Transfer Operand to Byte	A,C	TOB



The 8-bit operand replaces the previous contents of register B (A_0-A_7). Bits A_8-A_{23} are unaffected.

Cycles 1
Reference pages 3-1, 3-99

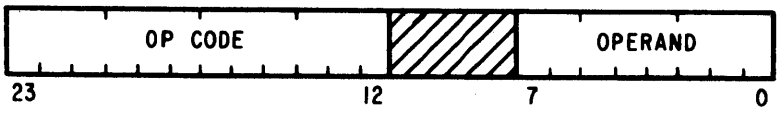
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
07. *+X:a	Transfer Memory to Byte	A, C	TMB



The 8 least significant bits (0-7) of the contents of the effective memory address replace the previous contents of register B (A_0-A_7). Bits A_8-A_{23} are unaffected.

Cycles ..... 2 (+1 per indirect reference)
Reference pages 3-1, 3-2, 3-99

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
00017:o	eXclusive-or Operand with Byte	A,C	XOB



An EXCLUSIVE-OR operation is performed between the 8-bit operand and the contents of register B (A_0-A_7). Bits A_8-A_{23} are unchanged.

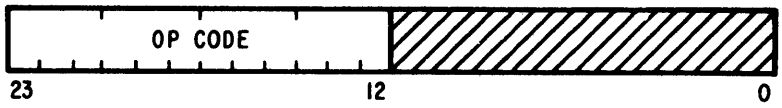
Cycles 1
Reference pages 3-1, 3-65

This page left blank intentionally.

3-16 MISCELLANEOUS INSTRUCTIONS

The following instructions are included in the Miscellaneous group because they do not fall into one of the previously defined functional groups.

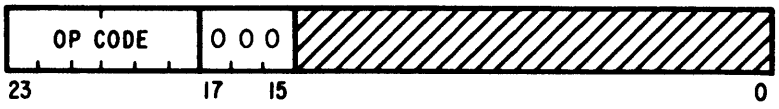
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0000.	HaLT	-	HLT



The PROGRAM ADDRESS (i.e., the contents of the P register) is advanced by one and program execution is terminated. When the RUN switch is depressed, execution will begin at the location defined by the PROGRAM ADDRESS.

Cycles 1
Reference page 3-1

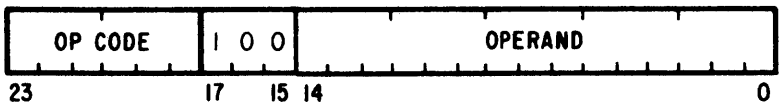
62.0	No OPeration	-	NOP



The PROGRAM ADDRESS is advanced by one and program execution continues with the next instruction.

Cycles 1
Reference page. 3-1

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
24.4:o	Generate Argument Pointer	I,J	GAP



The contents of register J are assumed to be the first address in an indirect memory reference sequence. The effective memory address derived from this indirect sequence replaces the previous contents of register I. The contents of register J and the 15-bit operand are added, and the result is placed in register J.

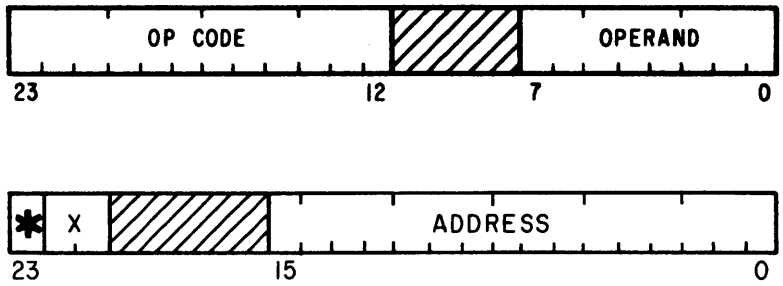
The purpose of a GAP instruction is to generate an effective memory address which points to one or more data words not directly available to a subroutine. This is illustrated in the following example where subroutine B requires the data contained in location Y.

Example:

```
A     BLJ    B     (J) = C, (P) = B
C     DAC*   X
D     ...         RETURN
      ...
X     DAC    Y
Y     DATA   2
      ...
B     GAP    1     (I) = Y, (J) = (J)+1
      TMA    0,I   (A) = 2
      BUC    0,J   (P) = D
```

Cycles .....2 (+1 per indirect reference)
Reference page 3-1

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0055:o *+X:A	Update Stack Pointer	K, C	USP



The contents of register K are replaced by the contents of the effective memory address. The 8-bit signed operand is then added to the contents of the effective memory address.

Example:

```
        BLJ     ENT     Call re-entrant routine
        ...
ENT     TRM*    SP      Save registers in stack
        USP     5       Update Stack Pointer  [(K) = stack, (SP) = stack+5]
        DAC     SP
        ...
        HTK     SP      Reset stack pointer
        TMR*    SP      Restore registers
        BUC     0,J     Return
SP      DAC     STACK   Stack pointer
STACK   BLOK    5N      Where N represents maximum number of
                        re-entrant levels
```

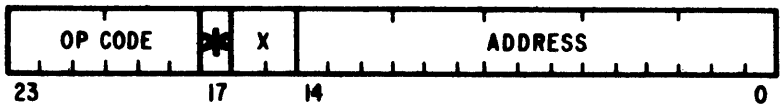
NOTES

(1) The Condition register is set to reflect the result of the operand addition.

(2) External interrupts are prohibited for the period of one instruction following this instruction.

Cycles .....4 (+1 per indirect reference)
Reference pages 3-1, 3-2

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
40. *+X:a	EXecute Memory	See Note 1	EXM



The instruction located in the effective memory address is executed as though it were at the address of the EXM.

In the case that the referenced instruction is a two word instruction, the second word must follow the EXM.

Example:

```
      EXM     M
      DAC     L       Second word
      ...
M     AOM     10      Two word instruction
      AOM     20
      AOM     30
```

NOTES

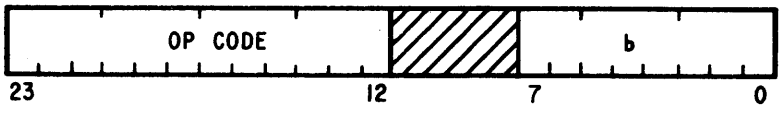
(1) The registers affected will depend on the instruction in the effective memory address.

(2) External interrupts are prohibited for the period of one instruction following the execution of this instruction.

Cycles ..... 1 (+1 per indirect reference) plus the time required to execute the referenced instruction

Reference pages 3-1, 3-2

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0011:b	Query Bits of Byte	C	QBB



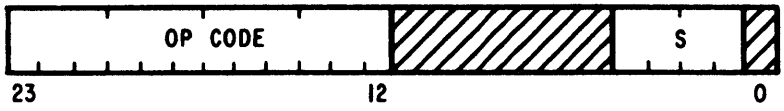
A logical AND is performed between operand bits 0-7 and the contents of register B. The Condition register is set according to the status of the result; i.e., positive, negative, or zero.

Examples:

```
1)  TOA     B7      A = '00000200     C = Positive
    ...
    ...
    QBB     B7                        C = Negative
2)  TOA     B6      A = '00000100     C = Positive
    ...
    ...
    QBB     B6                        C = Positive
3)  TNA     1       A = '77777777     C = Negative
    ...
    ...
    DMA     MASK    A = '40000000     C = Negative
    ...
    ...
MASK DATA   '40000000
```

Cycles 1
Reference page 3-1

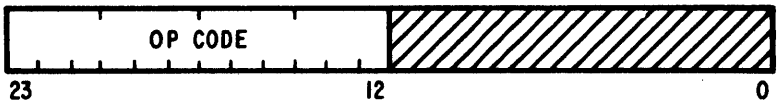
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0001:s	Query Sense Switches	C	QSS



A logical AND is performed between operand bits 1-4 and the state(s) of the sense switches. The Condition register is set accordingly.

Cycles 1
Reference page 3-1

0007.	Extend Zeros from Byte	A	EZB



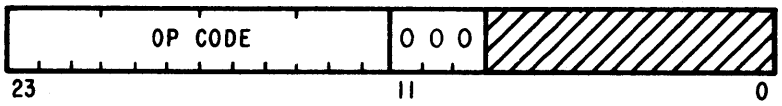
Bit positions A_8-A_{23} are set to ZERO. The contents of register B (A_0-A_7) are not affected.

NOTE

The Condition register is not affected.

Cycles 1
Reference page 3-1

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0077.0	Hold Interval Timer	-	HIT



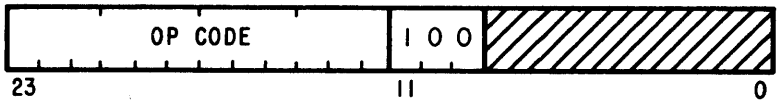
The CPU's Interval Timer is halted and will remain so until released by an RPT or RCT instruction.

NOTE

The 10-cycle counter associated with the Interval Timer is reset to zero by this instruction.

Cycles 1
Reference page 3-1

0077.4	Release Processor Time	-	RPT



The CPU's Interval Timer is started; i.e., allowed to begin its counting sequence. The Processor Time mode allows the Interval Timer to count only CPU time and not time stolen by an ABC operation.

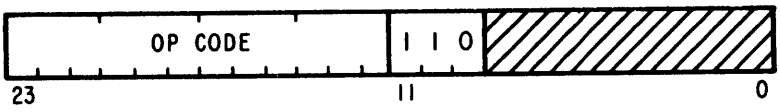
Once started, the timer counts until held by an HIT instruction or until the CPU is halted. At each 10-cycle interval, the contents of register T are decremented by one and tested for zero. If the contents of T are zero, an executive interrupt is triggered.

NOTE

The timer is not stopped by the executive interrupt.

Cycles 1
Reference page 3-1

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
0077.6	Release Clock Time	-	RCT



The CPU's Interval Timer is started; i.e., allowed to begin its counting sequence. The Clock Time mode causes the Interval Timer to count continuously.

Once started, the timer will count until held by an HIT instruction. At each 10-cycle interval, the contents of register T are decremented by one and tested for zero. If the contents of T are zero, an executive interrupt is triggered.

NOTE

The timer is not stopped by the interrupt.

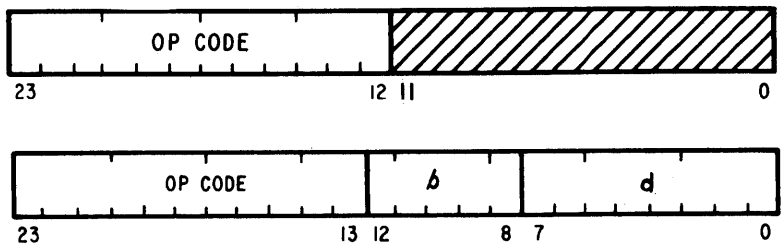
Cycles 1
Reference page 3-1

This Page Left Blank Intentionally.

3-17 BIT PROCESSOR INSTRUCTIONS

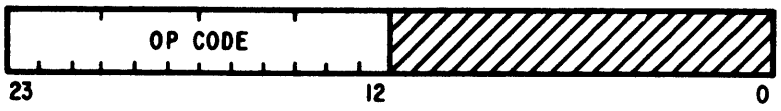
The bit (Boolean function) processor group of instructions include branches, logical manipulation, and interrogation of a specified bit selected from an effective memory address or the H register. In most instances, Bit 2 (ZERO/NOT ZERO) of the Condition register is used to display either the result of an operation or the status of a bit before the operation is performed.

The bit processor employs two instruction word formats. The first format uses an (bits 12-23) OP CODE to specify the operation to be performed. The remaining 12 bits (bits 0-11) are undefined. The second instruction format contains a displacement, bit specification, and an OP CODE. Eight bits (bits 0-7) are added to the base address contained in register V to obtain a displacement from the base address which is an effective memory address for the word containing the bit in question. Five bits (bits 8-12) are used to select a specific bit in the effective memory address for an operation as specified in the 11-bit (bits 13-23) OP CODE. Both instruction word formats are illustrated below.



The following instructions are included in the Bit Processor group.

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7742.	Transfer Zero to H	H,C	TZH



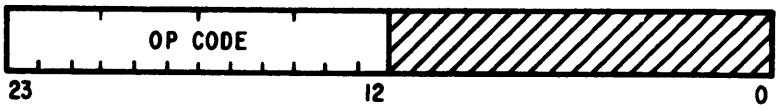
A ZERO is placed in register H. The Condition register is set to reflect the original contents of H.

NOTE

If the original contents of the H register was ZERO, Condition register Bit 2 is set to 1 (ZERO). If the contents was ONE, Bit 2 is set to 0 (NOT ZERO).

Cycles 1
Reference pages 3-1, 3-2, 3-157

7743.	Transfer Flag to H	H,C	TFH



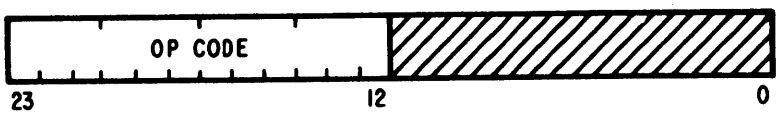
A ONE is placed in register H and the Condition register is set to reflect the original contents of H.

NOTE

If the original contents of the H register was ZERO, Condition register Bit 2 is set to 1 (ZERO). If the contents was ONE, Bit 2 is set to 0 (NOT ZERO).

Cycles 1
Reference pages 3-1, 3-2, 3-157

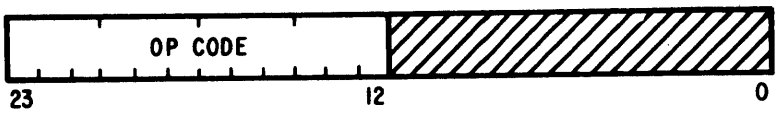
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7744.	Transfer K to V	V	TKV



The 18 least significant bits of register K replace the present contents of register V. The Condition register is unaffected.

Cycles 1
Reference pages 3-1, 3-2, 3-157

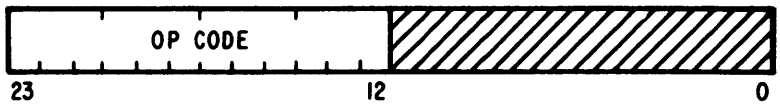
7745.	Transfer V to K	K	TVK



The contents of register V are transferred to the 18 least significant bit positions of register K. The Condition register is unaffected.

Cycles 1
Reference pages 3-1, 3-2, 3-157

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7746.	Query Bit of H	C	QBH



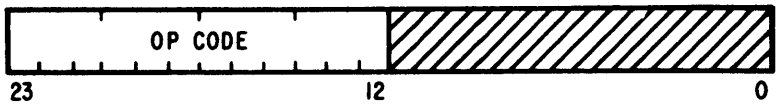
The H register bit is tested and the Condition register is set to display the result of the query.

NOTE

The Condition register is cleared. If the resultant content of the H register is ZERO, Condition register Bit 2 is set to 1 (ZERO). If the content is ONE, Bit 2 is set to 0 (NOT ZERO).

Cycles 1
Reference pages 3-1, 3-2, 3-157

7747.	Negate of H to H	H,C	NHH



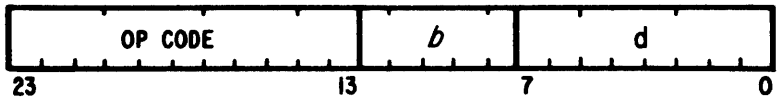
The current content of register H is complemented and returned to H. The Condition register is set to display the result.

NOTE

The Condition register is cleared. If the resultant content of the H register is ZERO, Condition register Bit 2 is set to 1 (ZERO). If the content is ONE, Bit 2 is set to 0 (NOT ZERO).

Cycles 1
Reference pages 3-1, 3-2, 3-157

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7750.*b*:d	Dot Memory with H	H,C	DMH



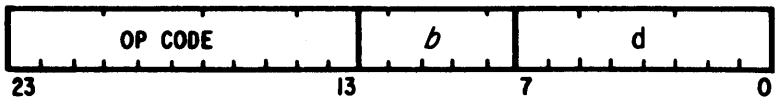
A logical AND is performed between the selected bit in the effective memory address and the contents of register H. The result is returned to the H register and the Condition register is set to display the result.

NOTE

The Condition register is cleared. If the resultant content of the H register is ZERO, Condition register Bit 2 is set to 1 (ZERO). If the content is ONE, Bit 2 is set to 0 (NOT ZERO).

Cycles 2
Reference pages 3-1, 3-2, 3-157

7752.*b*:d	Dot Not (memory) with H	H,C	DNH



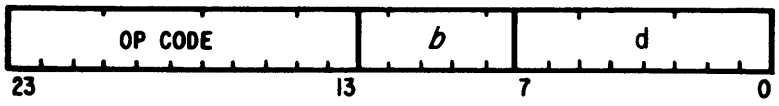
A logical AND is performed between the complement of the selected bit in the effective memory address and the content of register H. The result is returned to the H register and the Condition register is set to display the result.

NOTE

The Condition register is cleared. If the resultant content of the H register is ZERO, Condition register Bit 2 is set to 1 (ZERO). If the content is ONE, Bit 2 is set to 0 (NOT ZERO).

Cycles 2
Reference pages 3-1, 3-2, 3-157

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7754.b:d	Or Memory with H	H,C	OMH



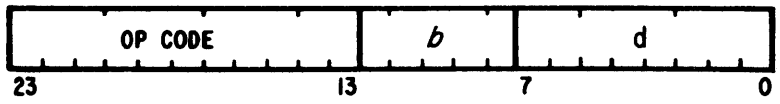
A logical OR is performed between the selected bit in the effective memory address and the content of register H. The Condition register is set to display the result.

NOTE

The Condition register is cleared. If the resultant content of the H register is ZERO, Condition register Bit 2 is set to 1 (ZERO). If the content is ONE, Bit 2 is set to 0 (NOT ZERO).

Cycles 2
Reference pages 3-1, 3-2, 3-157

7756.b :d Or Not (memory) with H H,C ONH



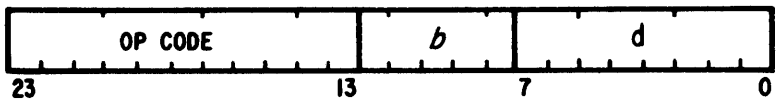
A logical OR is performed between the complement of the selected bit in the effective memory address and the content of register H. The Condition register is set to display the result.

NOTE

The Condition register cleared. If the resultant content of the H register is ZERO, Condition register Bit 2 is set to 1 (ZERO). If the content is ONE, Bit 2 is set to 0 (NOT ZERO).

Cycles 2
Reference pages 3-1, 3-2, 3-157

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7760.*b*:d	eXclusive-or Memory with H	H,C	XMH



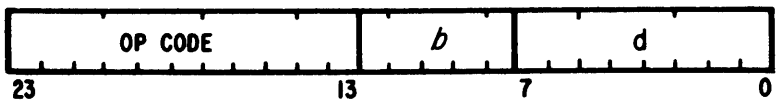
An EXCLUSIVE-OR function is performed between the selected bit in the effective memory address and the content of register H. The Condition register is set to display the result.

NOTE

The Condition register is cleared. If the resultant content of the H register is ZERO, Condition register Bit 2 is set to 1 (ZERO). If the content is ONE, Bit 2 is set to 0 (NOT ZERO).

Cycles 2
Reference pages 3-1, 3-2, 3-157

7762.*b*:d	eXclusive-or Not (memory) with H	H,C	XNH



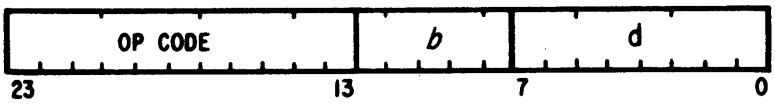
An EXCLUSIVE-OR function is performed between the complement of the selected bit in the effective memory address and the content of H register. The Condition register is set to display the result.

NOTE

The Condition register is cleared. If the resultant content of the H register is ZERO, Condition register Bit 2 is set to 1 (ZERO). If the content of ONE, Bit 2 is set to 0 (NOT ZERO).

Cycles 2
Reference pages 3-1, 3-2, 3-157

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7764.*b*:d	Transfer Memory to H	H,C	TMH



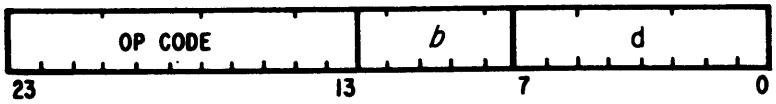
The selected bit in the effective memory address is transferred to register H. The Condition register is set to display the resultant content of the H register.

NOTE

The Condition register is cleared. If the resultant content of the H register is ZERO, Condition register Bit 2 is set to 1 (ZERO). If the resultant content is ONE, Bit 2 is set to 0 (NOT ZERO).

Cycles 2
Reference pages 3-1, 3-2, 3-157

7766.*b*:d	Query Bit of Memory	C	QBM



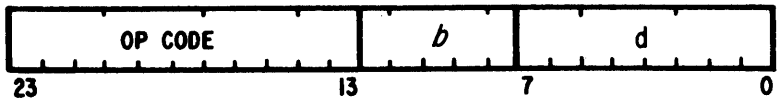
The selected bit in the effective memory address is tested and the Condition register is set to display the result of the query.

NOTE

The Condition register is cleared. If the resultant content of the H register is ZERO, Condition register Bit 2 is set to 1 (ZERO). If the resultant content is ONE, Bit 2 is set to 0 (NOT ZERO).

Cycles 2
Reference pages 3-1, 3-2, 3-157

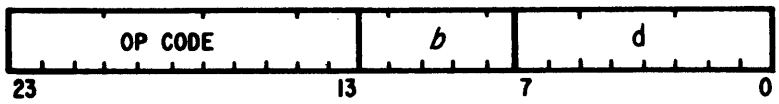
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7770.*b*:d	Transfer H to Memory	M	THM



The content of register H is placed in the selected bit position in the effective memory address. The Condition register is not affected.

Cycles 3
Reference pages 3-1, 3-2, 3-157

7772.*b*:d	Flag Bit of Memory	M,C	FBM



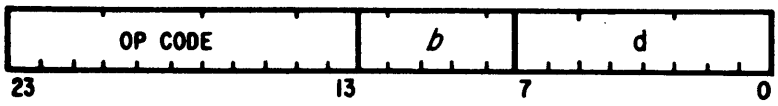
A ONE is placed in the selected bit position in the effective memory address. The Condition register is set to display the original state of the selected bit in memory.

NOTE

If the original state of the selected bit in memory was ZERO, Condition register Bit 2 is set to 1 (ZERO). If the original state was ONE, Bit 2 is set to 0 (NOT ZERO).

Cycles 3
Reference pages 3-1, 3-2, 3-157

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7774.*b*:d	Zero Bit of Memory	M,C	ZBM



A ZERO is transferred to the selected bit position in the effective memory address. The Condition register is set to display the original state of the selected bit in memory.

NOTE

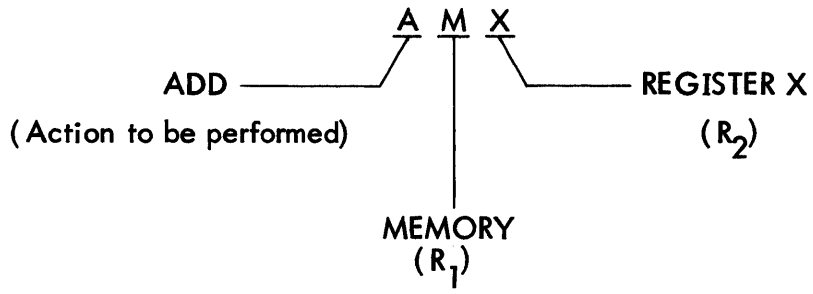
If the original state of the selected bit in memory was ZERO, Condition register Bit 2 is set to 1 (ZERO). If the original state was ONE, Bit 2 is set to 0 (NOT ZERO).

Cycles 3
Reference pages 3-1, 3-2, 3-157

3-18 SCIENTIFIC ARITHMETIC UNIT INSTRUCTIONS

The instruction set for the Scientific Arithmetic Unit is divided into five functional groups: arithmetic, transfer, branch, compare, and interrupt control. In the descriptions that follow, each instruction is defined and the time is given in cycles and concurrent cycles available. The concurrent cycles, if any, occur after the instruction has been initiated by the SAU. The SAU is designed to operate on normalized floating point numbers, and all descriptions of the arithmetic instructions are based on this fact. If an unnormalized operand is used in an arithmetic operation the results are not considered valid.

Standard arithmetic instructions - add, subtract, multiply, and divide - as well as square, square root, fix and float are included in the group. The instruction mnemonics provide a brief definition of specific operations to be performed. The first letter in the mnemonic specifies the action or type of operation that is to be performed. The second letter identifies the first quantity or reference (R_1) to be used in the operation and the third letter identifies the second reference (R_2). For example:



In the majority of SAU arithmetic instructions, the result of the operation remains in R_2 while R_1, remains unchanged (except where R_1 and R_2 are the same).

Unless otherwise noted, each arithmetic operation sets a bit in the SAU Condition (Y) register to reflect the status of the result. Various conditions are described below:

(a) Positive - The result is arithmetically greater than zero, indicated by a ONE in bit position 3 of the Y register. A ZERO in bit position 3 indicates "Not Positive".

(b) Zero -- All of the mantissa bits comprising the quantity under consideration are ZERO and the exponent is '201, indicated by a ONE in bit position 2 of the Y register. A ZERO in bit position 2 indicates "Not Zero".

(c) Negative - The result is arithmetically less than zero, indicated by a ONE in bit position 1 of the Y register. A ZERO in bit position ONE indicates "Not Negative".

(d) Overflow - An overflow results from an arithmetic operation which causes exponent overflow, i. e. ,an exponent greater than 2^7-1 (127) or less than -2^7 (-128).

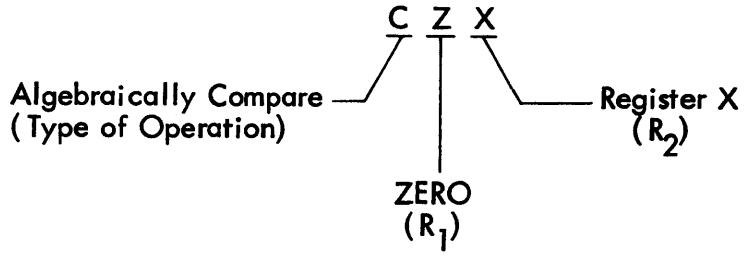
NOTE

If the SAU Overflow/Underflow executive trap is enabled, any instruction causing the overflow bit of the Y register to be set will cause an interrupt.

Bits 1, 2 and 3 (Negative, Zero, Positive) of the Y register are normally mutually exclusive. In certain instances it is desirable to know what operation caused an overflow, e.g., a division by zero. The following operations cause more than two bits to be set in the Y register:

Division by zero sets bits 0, 2, 3 ('15)
$\sqrt{-X}$ sets bits 0, 1, 2, 3 ('17)
Float to Fix, X>8388607 sets bits 0, 1, 3 ('13)

The algebraic compare instructions are included in the SAU instruction set which compare two referenced, signed (+ or -) quantities. The Y condition register is set according to the result of the comparison. Algebraic comparisons are identified by the letter "C" as the first letter in the instruction mnemonic (e.g., CZX). The second letter in the mnemonic code identifies the first of the compared quantities (R_1) and the remaining letter identifies the second quantity (R_2). For example:



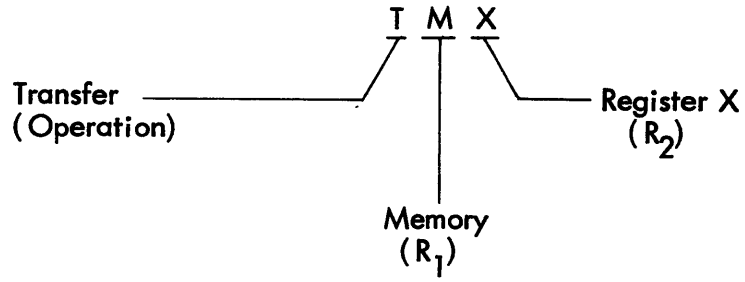
Comparisons are performed according to the following formula:

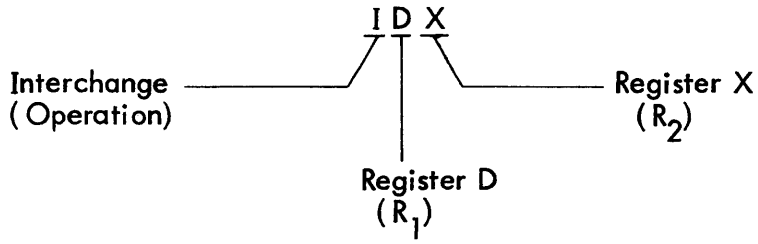
$R_2 - R_1 = Y$ (positive, zero, or negative)

Therefore, $R_2 > R_1$, $R_2 < R_1$, and $R_2 = R_1$, will set the condition (Y) register to positive (+), negative (-), and zero (0) respectively.

Two instructions provide control of the SAU interrupt. These instructions either release or hold the interrupt.

The transfer instruction group includes various types of operations. Among these are transfers between memory and registers, registers and memory, and register-to-register. The transfer operation mnemonic code describes the individual operation. When operation is to be performed is described by the first letter in the mnemonic; "T" for transfer and "I" for interchange. The second and third of the mnemonic specify the source (R_1) and destination (R_2) of the transfer respectively. Listed below are two examples:





With the exception of the interchange instruction, the transfer source (R_1) is not altered as a result of the execution of a transfer instruction.

The following instructions are included in the SAU group.

ARITHMETIC

Mnemonic	Instruction	Page
AAX	Add A register to X register	3-171
ADX	Add D register to X register	3-171
AMX	Add Memory to X register	3-172
AOW	Add Operand to W register	3-172
AOX	Add Operand to X register	3-173
DAX	Divide A register into X register	3-173
DDX	Divide D register into X register	3-174
DMX	Divide Memory into X register	3-174
DOX	Divide Operand into X register	3-175
FAX	Floating normalize of A register to X register	3-176
FXA	Fix of X register to A register	3-177
INX	INverse of X register	3-177
MAX	Multiple A register and X register	3-178
MDX	Multiple D register and X register	3-178
MMX	Multiple Memory and X register	3-179
MOX	Multiple Operand X register	3-179
NXX	Negative of X register to X register	3-180
PXX	Positive of X register to X register	3-180
SAX	Subtract A register from X register	3-181
SDX	Subtract D register from X register	3-181
SEX	SquarE X register	3-182
SMX	Subtract Memory from X register	3-182
SOX	Subtract Operand from X register	3-183
SRX	Square Root of X register	3-183

BRANCH

Mnemonic	Instruction	Page
BNR	Branch on Negative Reset	3-184
BNS	Branch on Negative Set	3-184
BOR	Branch on Overflow Reset	3-184
BOS	Branch on Overflow Set	3-184
BOX	Branch on SAU Ready	3-185

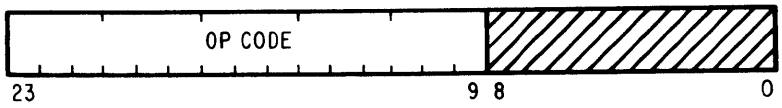
BRANCH (Cont'd.)

COMPARE

INTERRUPT

TRANSFER

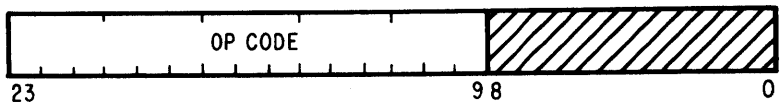
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7707.0	Add A register to X register	X,Y	AAX



The signed integer in the A register is converted to floating-point format and added to the number in the X register. The sum replaces the previous contents of the X register.

Cycles 2
Concurrent cycles available 1

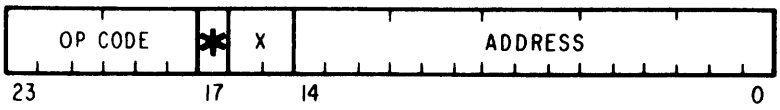
7710.0	Add D register to X register	X,Y	ADX



The floating-point number in the D register is added to the number in the X register. The sum replaces the previous contents of the X register.

Cycles 2
Concurrent cycles available 1

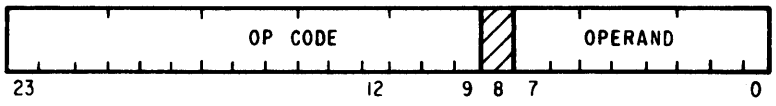
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
73. *+X· a	Add Memory to X register	X,Y	AMX



The contents of the Effective Memory Address (EMA) and the next sequential address (EMA +1) are added to the contents of the X register. The sum replaces the previous contents of the X register.

Cycles . . . . . 3 (+1 per indirect reference)

7701.2:o	Add Operand to W register (exponent)	W,Y	AOW



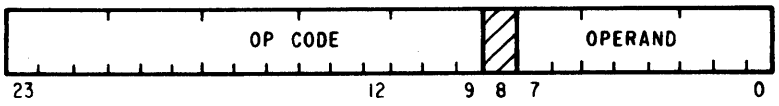
The 8-bit, signed operand is algebraically added to the contents of the W register. The sum replaces the previous contents of the W register.

NOTE

A subtraction may be accomplished by adding a negative operand.

Cycles 1

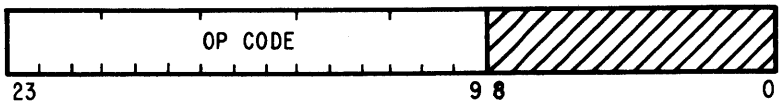
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7706.0:o	Add Operand to X register	X,Y	AOX



The signed, 8-bit integer operand is converted to floating point format and added to the contents of the X register. The sum replaces the previous contents of the X register.

Cycles 2
Concurrent cycles available 1

7707.3	Divide A register (integer) into X register	X,Y	DAX



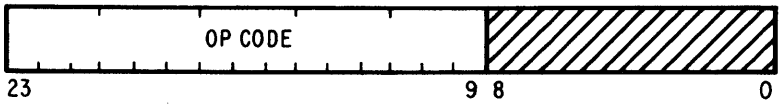
The signed integer in the A register is converted to floating point format. The contents of the X register are divided by the converted number. The quotient replaces the previous contents of the X register

NOTE

If division by zero occurs, the condition register (Y) is set to overflow, positive and zero, i.e., (Y) = '15

Cycles 15
Concurrent cycles available 14

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7710.3	Divide D register (floating-point) into X register	X,Y	DDX



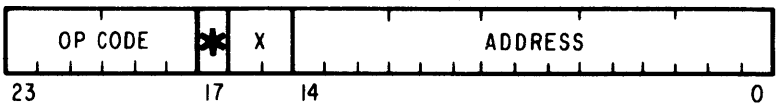
The floating-point contents of the D register are divided into the contents of the X register. The quotient replaces the previous contents of the X register.

NOTE

If division by zero occurs the condition register (Y) is set to overflow, positive and zero, i.e., (Y) = '15

Cycles 15
Concurrent cycles available 14

76. *+X: a	Divide Memory into X register	X,Y	DMX



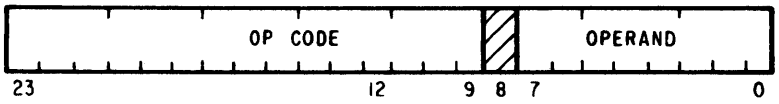
The contents of the X register are divided by the contents of the effective memory address (EMA) and the next sequential address (EMA +1). The quotient replaces the previous contents of the X register.

NOTE

If division by zero occurs, the condition register (Y) will be set to overflow, positive and zero, i.e., (Y) = '15

Cycles . . . . . 16(+ 1 per indirect reference)
Concurrent cycles Available 13

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7706.3:o	Divide Operand into X register	X,Y	DOX



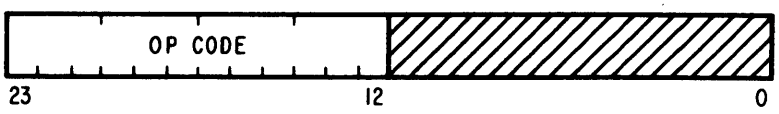
The signed, 8-bit integer operand is converted to floating-point and is divided into the contents of the X register. The quotient replaces the previous contents of the X register.

NOTE

If division by zero occurs, the condition register (Y) will be set to overflow, positive and zero, i.e., (Y) = ' 15

Cycles 15
Concurrent cycles available 14

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7703.	Floating normalize of A register to X register	X,Y	FAX



A positive normalized number will have as the sign and most significant bit the following pattern:

01

A negative normalized number (where the value is not -1) has the configuration

10

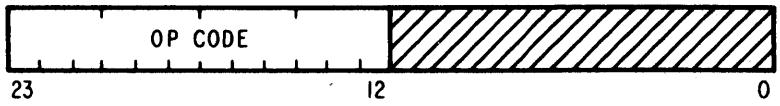
A -1 has the pattern

11

If the result is zero, the mantissa will be zero and the exponent will be set to a full scale negative value, i.e. (W) = '201.

Cycles 1

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7713.	Fix of X register to A register	A,Y	FXA



The integer portion of the positive mantissa in the X register replaces the previous contents of the A register.

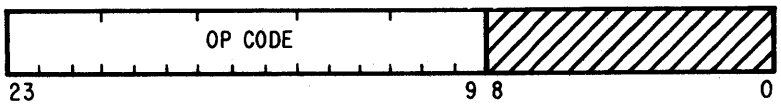
NOTE

If the exponent is greater than 23, the condition register (Y) will be set to overflow, negative and positive, i.e., (Y) = '13.

If the mantissa is negative, a one bit error may result.

Cycles 2

7705.0	INverse of X register	X,Y	INX



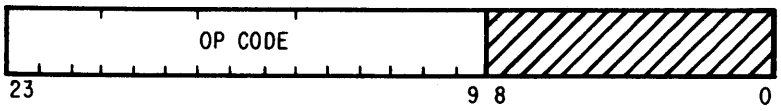
The inverse of the contents $\left[\frac{1}{(X)}\right]$ of the X register replaces the contents of the X register.

NOTE

If division by zero occurs, the condition register will be set to overflow, positive and zero, i.e., (Y) = '15.

Cycles 15
Concurrent Cycles Available 14

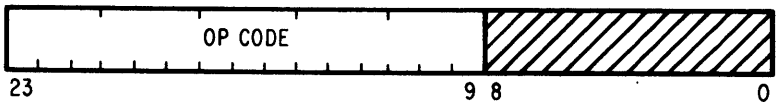
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7707.2	Multiply A register (integer) and X register	X,Y	MAX



The signed integer in the A register is converted to floating-point format and multiplied by the contents of the X register. The product replaces the previous contents of the X register.

Cycles 7
Concurrent cycles available. . . . 6

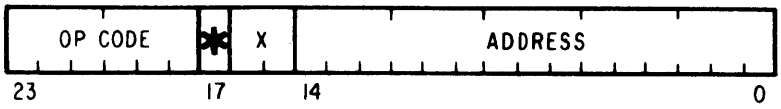
7710.2	Multiply D register (floating point) and X register	X,Y	MDX



The floating-point contents of the D register are multiplied by the contents of the X register. The product replaces the previous contents of the X register.

Cycles 7
Concurrent cycles available. 6

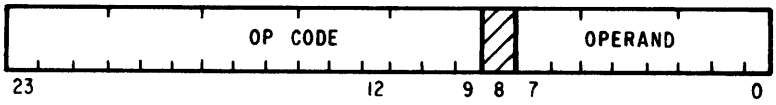
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
75. *+X:a	Multiply Memory and X register	X,Y	MMX



The contents of the X register are multiplied by the contents of the effective memory address (EMA) and the next sequential address (EMA+1). The product replaces the previous contents of the X register.

Cycles . . . . . 7 (+1 per indirect reference)
Concurrent Cycles Availalble . . . 4

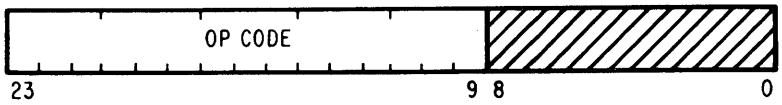
7706.2:o	Multiply Operand and X register	X,Y	MOX



The signed, 8-bit integer operand is converted to floating point format and is multiplied by the contents of the X register. The floating-point product replaces the previous contents of the X register.

Cycles 7
Concurrent Cycles Available 6

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7704.1	Negative of X register To X register	X,Y	NXX



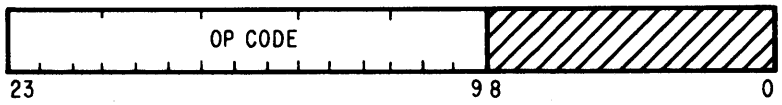
The mantissa in the X register is two's-complemented and the result is loaded into the X register. The Y register is changed to reflect the status of the new quantity.

NOTE

If the bit pattern of the mantissa is 100....0, a one bit error will occur in the result.

Cycles 1

7704.0	Positive of X register to X register	X,Y	PXX



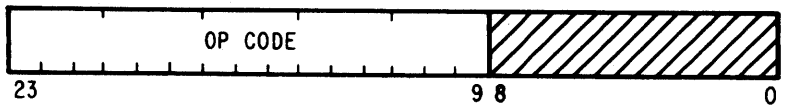
The absolute value of the contents of the X register replaces the previous contents of the X register.

NOTE

If the bit pattern of the mantissa is 100....0, a one bit error will occur in the result.

Cycles 1

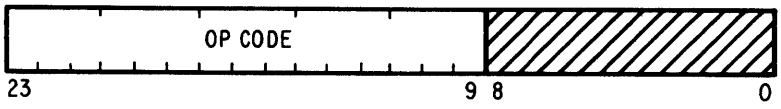
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7707.1	Subtract A register (integer) from X register	X,Y	SAX



The signed integer in the A register is converted to floating-point format and subtracted from the contents of the X register. The difference replaces the previous contents of the X register.

Cycles 2
Concurrent cycles available 1

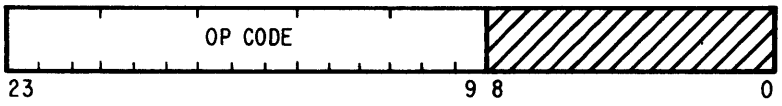
7710.1	Subtract D register (floating point) from X register	X,Y	SDX



The floating-point contents of register D are subtracted from the X register. The difference replaces the previous contents of the X register.

Cycles 2
Concurrent cycle available 1

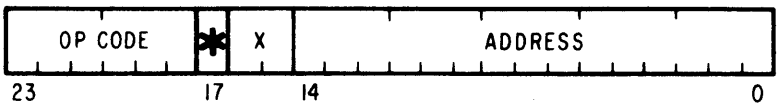
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7705. 1	SquarE X register	X,Y	SEX



The square of the contents of the X register replaces the previous contents of the X register. (i.e., register X is replaced by X times X.)

Cycles 7
Concurrent cycle available 6

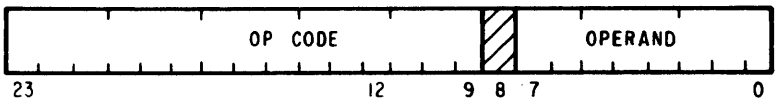
74. *+X:a — Subtract Memory from X register — X,Y — SMX



The contents of the effective memory address (EMA) and the next sequential address (EMA+1) are subtracted from the contents of the X register. The difference replaces the contents of the X register.

Cycles . . . . . .3 (+1 per indirect reference)

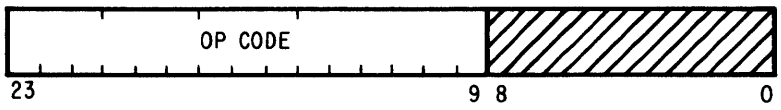
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7706.1:o	Subtract Operand from X register	X,Y	SOX



The signed, 8-bit integer operand is converted to floating-point format and subtracted from the contents of the X register. The difference replaces the previous contents of the X register.

Cycles 2
Concurrent cycles available 1

7705.2	Square Root of X register	X,Y	SRX



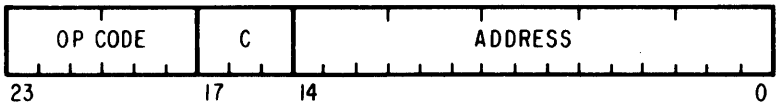
The square root of the contents of the X register replaces the previous contents of the X register.

NOTE

If the contents of register X is negative, the condition register is set to positive, zero, negative and overflow, i.e., (Y) = '17.

Cycles 13
Concurrent Cycles Available 12

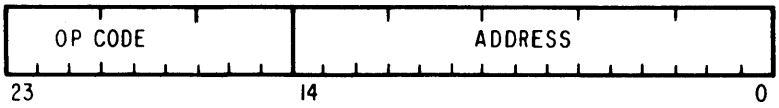
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
63.0:a	Branch on Negative Reset	P	BNR
63.7:a	Branch on Negative Set	P	BNS
64.0:a	Branch on Zero Reset	P	BZR
64.7:a	Branch on Zero Set	P	BZS
65.0:a	Branch on Positive Reset	P	BPR
65.7:a	Branch on Positive Set	P	BPS
77.2:a	Branch on Overflow Reset	P	BOR
77.3:a	Branch on Overflow Set	P	BOS



The contents of the Y condition register are tested for the specified condition. If the condition is present, the contents of the P register (current PROGRAM ADDRESS) are replaced by the effective memory address. If the specified condition is not present, the program address advances to the next sequential location (PROGRAM ADDRESS +1).

Cycles 1

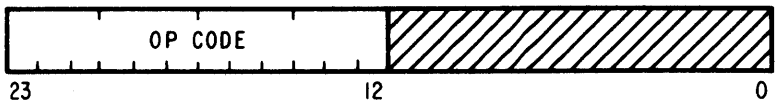
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
62.7:a	Branch on SAU Ready	P	BOX



A determination is made as to whether or not the SAU is processing an instruction (the SAU Busy latch is tested). If the SAU is able to process another instruction (i.e., ready) then the contents of the P register (current PROGRAM ADDRESS) are replaced by the effective memory address. If the SAU is currently processing an instruction (i.e., not ready) the program address advances to the next sequential location (PROGRAM ADDRESS +1).

Cycles 1

7712.	Compare D register to X register	Y	CDX



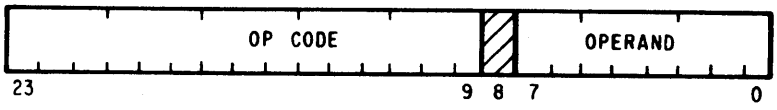
The contents of the D register and the contents of the X register are compared and the Y condition register is set to the status of the result.

Comparison results are as follows:

If X is greater than D;	Y = positive
If X is equal to D;	Y = zero
If X is less than D;	Y = negative

Cycles 2
Concurrent cycles available 1

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7701.3:o	Compare Operand to W register (exponent)	Y	COW



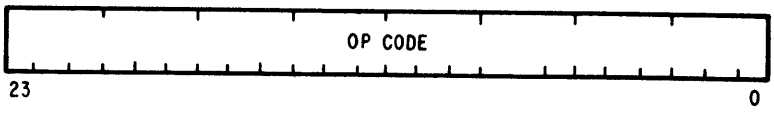
The 8-bit, signed operand and the contents of the W register are algebraically compared and the Y condition register is set to the status of the result.

Comparison results are as follows:

If W is greater than the operand; Y = positive
If W is equal to the operand; Y = zero
If W is less than the operand; Y = negative

Cycles 1

7706.0000	Compare Zero to X register	Y,X	CZX



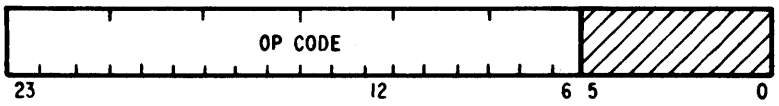
The contents of the X register and floating-point zero are compared and the Y condition register is set to the status of the result.

NOTE

Overflow will result if the mantissa and exponent have the pattern:
110....0/10....0
The least significant bit of X will be set to a 1.

Cycles 2
Concurrent cycles available . . .1

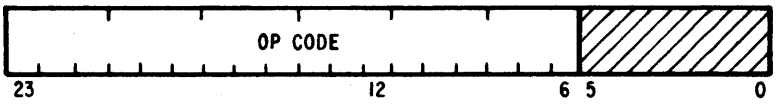
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7702.00	Hold SAU overflow Interrupt	None	HSI



This instruction disarms the overflow/underflow interrupt (Executive trap Group 0, Level 6). The trap remains disarmed until the execution of the release instruction.

Cycles 1

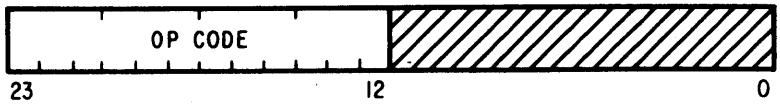
7702.01	Release SAU overflow Interrupt	None	RSI



This instruction arms the overflow/underflow interrupt (Executive Trap Group 0, Level 6). When the trap is armed, and not inhibited by an HXI instruction, any SAU operation which causes bit 0 of the Y register to be set (overflow) will generate an interrupt request.

Cycles 1

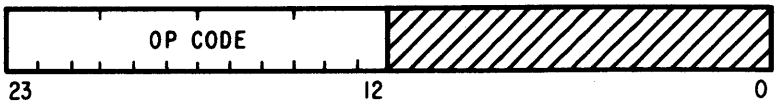
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7711.	Interchange D register and X register	D,X,Y	IDX



The contents of the X register and the D register are interchanged. The Y condition register is set to the status of the X register on completion of the instruction.

Cycles 2

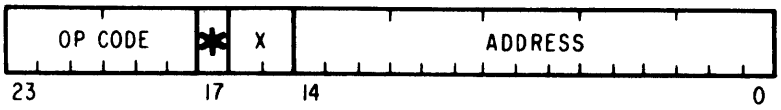
7714.	Transfer D register to X register	X,Y	TDX



The contents of the D register replace the previous contents of the X register.

Cycles 1

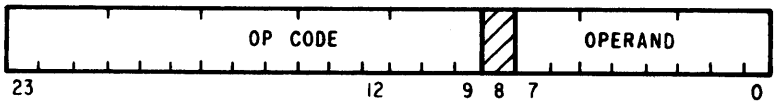
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
71.*+X;a	Transfer Memory to X register	X,Y	TMX



The contents of the effective memory address (EMA) and the next sequential address (EMA+1) replace the previous contents of the X register. EMA and EMA+1 replace the most significant and least significant part of X, respectively.

Cycles . . . . . 3 (+1 per indirect reference)

7701.1;o	Transfer Operand to W register (exponent)	W,Y	TOW

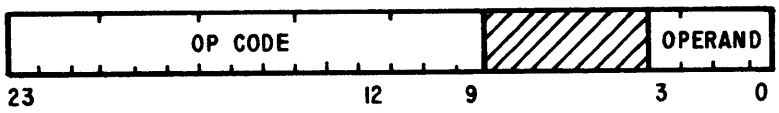


The 8-bit, signed operand replaces the previous contents of the W register. All other bits within the X register are unaffected.

The Y condition register is set to the status of the X and XW registers upon completion of the instruction.

Cycles 1

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7701.0:o	Transfer Operand to Y register	Y	TOY



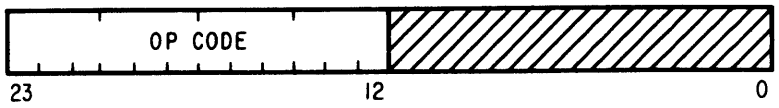
The four bit operand replaces the previous contents of the Y condition register.

Operand definition is as follows:

bit Ø-ONE = Overflow	ZERO = No overflow
bit 1-ONE = Negative	ZERO = Not negative
bit 2-ONE = Zero	ZERO = not zero
bit 3-ONE = Positive	ZERO = not positive

Cycles 1

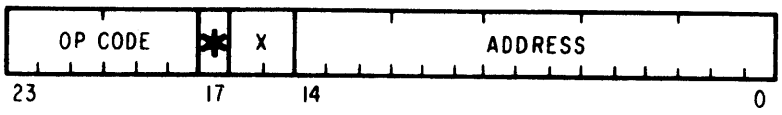
7715.	Transfer X register to D register	D	TXD



The contents of the X register replaces the previous contents of the D register. The X register is unchanged.

Cycles 2

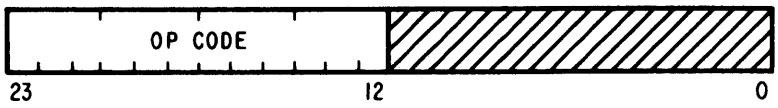
INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
72. *+X:a	Transfer X register to Memory	M	TXM



The contents of the X register replaces the previous contents of the effective memory address (EMA) and the next sequential address (EMA+1). The most and least significant portions of X are transferred to EMA and EMA+1, respectively.

Cycles . . . . . .3 (+1 per indirect reference)

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7700.	Transfer Y register to A register	A	TYA



The contents of the Y register are transferred to the A register and the status of the SAU overflow/underflow interrupt will be placed in bit position 6 in the A register.

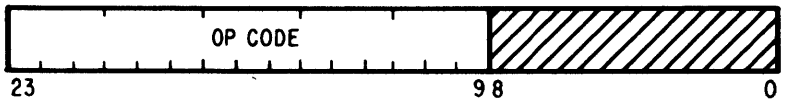
The following table shows the bit placements of the various Y condition register settings when transferred to the A register.

A Register	Bit Function
Bit Ø = 1	Overflow/Underflow
Bit 1 = 1	Negative
Bit 2 = 1	Zero
Bit 3 = 1	Positive
Bit 6 = Ø	SAU Interrupt Enabled
Bit 6 = 1	SAU Interrupt Disabled

All other bits within the A register are set to zero.

Cycles 1

INSTRUCTION FORMULA	FUNCTION	REGISTERS AFFECTED	MNEMONIC
7704.2	Transfer Zero to X register	X	TZX



The floating point representation of zero (0000000000000201) replaces the previous contents of the X register. The Y condition register is unaffected.

Cycles 1

This page left blank intentionally.

SECTION IV
INPUT/OUTPUT

4-1 INTRODUCTION

The DC 6024/4 Computer System input/output (I/O) structure combines the characteristic economy of unit I/O systems with the speed of a channel I/O system. This configuration, in conjunction with the I/O instruction repertoire, permits maximum flexibility in I/O communications.

The basic I/O structure allows single word data transfers between the Central Processing Unit (CPU) and a peripheral unit. It also allows I/O command and test operations to be program-controlled. An optional Automatic Block Controller (ABC) or an optional External Block Controller (XBC) may be used to control the transfer of blocks of data between the CPU and the peripheral units without program intervention.

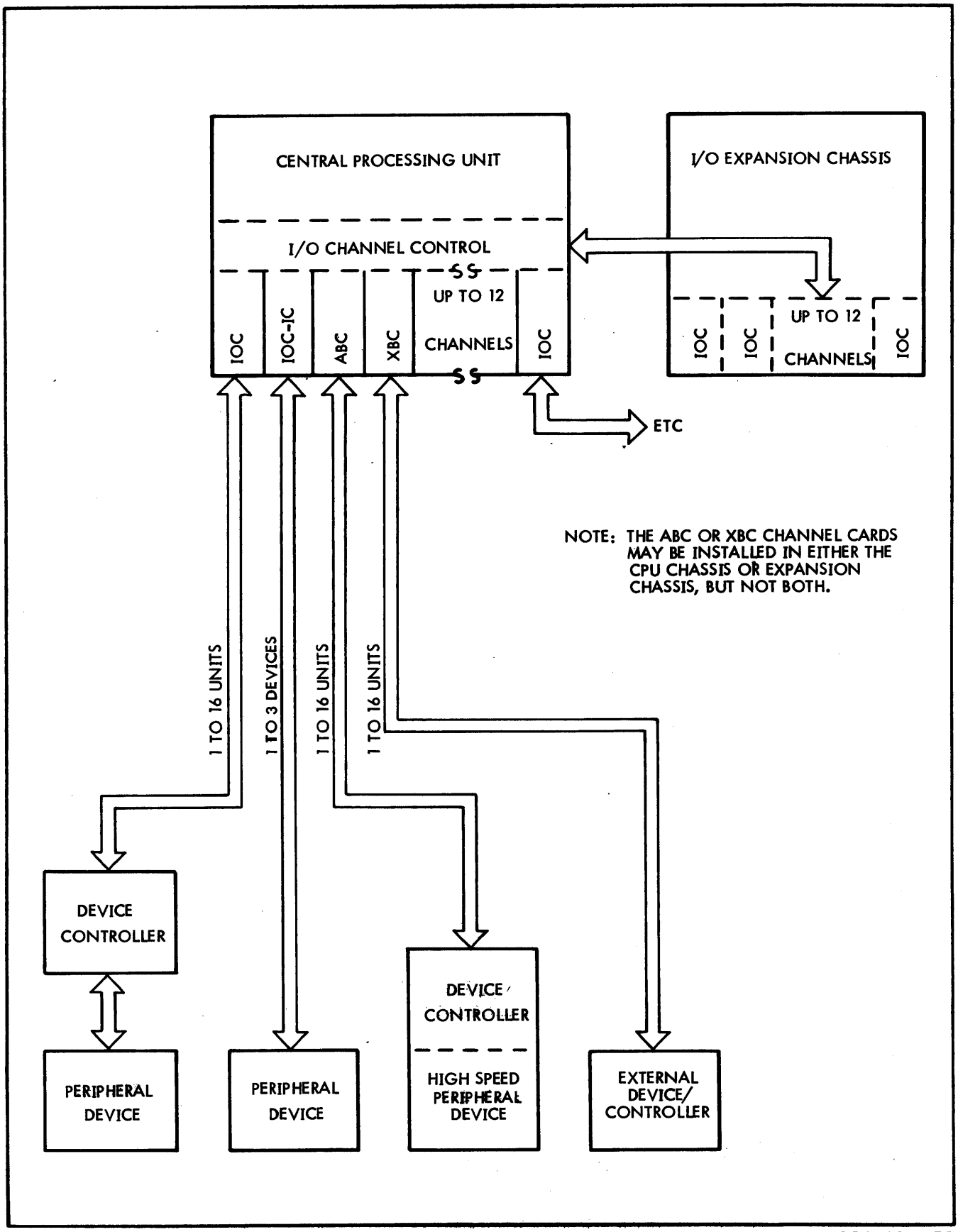
The relationship between the CPU and the I/O structure is illustrated on Figure 4-1. The elements comprising the I/O structure are described in the following paragraphs.

4-2 BASIC INPUT/OUTPUT STRUCTURE

The basic I/O structure involves communication (such as data transfers, addresses, and command/status information) between the CPU and a peripheral unit by way of a channel. The CPU communicates with a specific channel and the channel, in turn, communicates with a peripheral unit. The I/O structure accommodates up to 24 Input/Output Channel (IOC) cards (12 in basic chassis) for the DC 6024/4, all of which can be active concurrently. Each channel can communicate with from one to 16 peripheral units using the standard I/O instructions. Only one peripheral unit per channel can be connected; however, all 16 units can be active at any given time.

Communications between the I/O structure and the CPU may also be conducted on an interrupt basis. Logic in the channel and unit allows unit interrupts to be placed under program control and selectively enabled or disabled by executing the appropriate I/O instruction. An alternate method permits unit functions to be wired directly to the CPU priority interrupt structure and used as interrupt triggers.

The I/O interface is the link between each peripheral unit and its channel. The interface and its associated unit control facilities provide the physical means for connecting the peripheral device to the I/O structure and the logic capability that allows the unit to adopt the standard I/O controls to its specific requirements. The interface facilities and unit control logic are normally integrated with the peripheral unit. However, some controllers for the less complex devices are available as options to the Internal Controller (IC) IOC card.



BD1510-873

Figure 4-1. DC 6024/4 Computer I/O Structure Block Diagram

4-3 AUTOMATIC BLOCK TRANSFER

4-3.1 Automatic Block Controller (ABC)

The optional Automatic Block Controller (ABC) permits a block of data (n words) to be transferred between the CPU and peripheral units, without program intervention once activated. Data blocks of from one to 65,536 words may be automatically transferred. In addition, separate data blocks may be linked and transferred under control of the ABC.

Up to 12 ABCs may be used with the I/O structure. Each ABC services a single channel. Memory access by one or more ABCs is granted on a priority basis. Two or more ABCs may be interleaved depending on the transfer capability of the peripheral units involved.

Once initiated, the ABC channel transfer blocks of data to and from the computer memory on a memory cycle stealing basis. A block transfer is initiated under program control and proceeds from that point without further program intervention until the transfer is terminated. When an ABC channel is used in a system configuration, no special considerations in the unit controller or units have to be made. The unit will respond with the same signals that are used in all other I/O channels. The maximum combined transfer rate with the ABC is 1,333,333 words per second.

4-3.2 External Block Controller (XBC)

The External Block Controller (XBC) permits a block of data (n words) to be transferred between the CPU and an external device, without program intervention, once initialized. The operation is similar to the ABC except that the external device controls the addressing and interrupt requests to the XBC.

4-4 INPUT/OUTPUT COMMUNICATIONS

Input/output communications consist of the transfer of command, data, status or address words between the A register and the specified channel/unit combination upon the execution of appropriate I/O instructions. The channel/unit code in each I/O instruction, excluding Output Address Word (OAW) and Input Address Word (IAW) allows one ot 12 channels to be selected and one of up to 16 units to be connected to that channel. When an instruction to the same channel carries a different unit code, the previously specified unit is disconnected and the new unit is connected automatically. During this disconnect/connect sequence, the channel is busy and does not respond to I/O instructions until the sequence is completed. If a channel is in the process of transferring commands or data to a unit, an Input Status Word (ISW) or Input Data Word (IDW) instruction addressed to a different unit on the same channel receives a busy indication.

Command and data words from the CPU are transferred to the channel output buffer and subsequently to the connected peripheral unit. Data and status words are retained in the input buffer to the selected unit and transferred to the A register upon request (instruction) from the CPU. Address words are applicable only to those channels employing an ABC or an XBC.

4-4.1 I/O Commands

The Output Command Word (OCW) instruction transfers a command word to the specified channel/unit combination. The command word bits specify the unit control function(s) to be

performed and/or the I/O condition to be established. Following the execution of an OCW instruction, the channel remains busy until the command has been accepted by the addressed unit. The I/O command word format, shown in Figure 4-2, illustrates bit assignments for all standard peripherals. Figure 4-3 shows the format for a typical OCW instruction.

If the channel is busy or not ready when addressed by the OCW instruction, the Condition register is set to "Not Zero" to allow a programmed delay. The Override function causes the channel to go through a unit disconnect/connect sequence regardless of the unit code. This clears the channel/unit of any other activity and allows the current instruction to assume control of the channel unconditionally upon termination of the disconnect/connect sequence.

4-4.2 Command Word Interrupt Control

The OCW instruction may be used to selectively enable and disable two peripheral unit interrupts. The two interrupts are defined as Input and Output and are controlled by bits 0-2 of the command word. Table 4-1 illustrates the various bit configurations.

The terms "input interrupt" and "output interrupt" are applicable only to peripheral units that are equipped with both input and output data handling facilities. Input only devices may make use of the input interrupt and an alternate interrupt at the normal output level. Output only devices may make use of the output interrupt plus an alternate at the normal input level.

When the unit input interrupt has been previously enabled, an Input Interrupt signal will be generated when the input buffer in the unit is loaded (i.e., the same time the Input Data Available signal is generated).

4-4.3 I/O Status Word

The Input Status Word (ISW) instruction is used to test the operational status of a peripheral unit. When a channel is addressed by the ISW instruction, a 24-bit status word is transferred to the A register in the CPU. The basic I/O status word format is illustrated on Figure 4-2.

The quantity and significance of the status bits depends on the type of peripheral unit involved. The WC bit (bit 23) is applicable only to those channels employing an ABC.

4-4.4 Single Word Data Transfer

A. Input

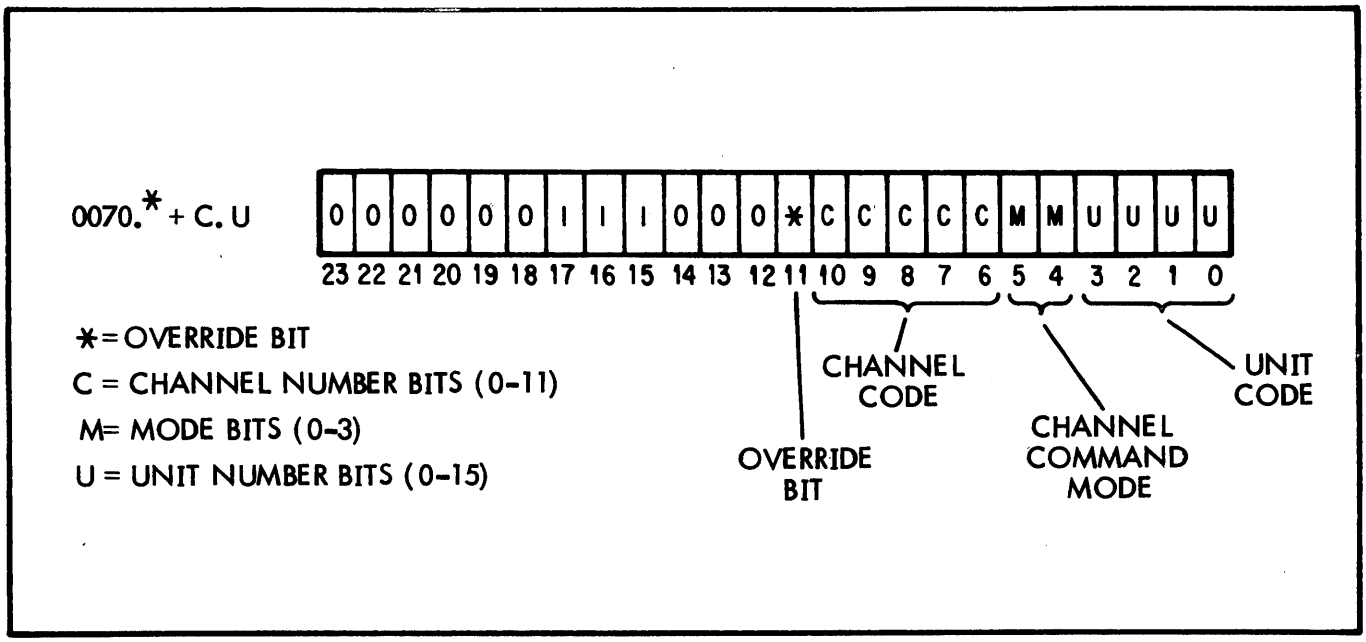
The Input Data Word (IDW) instruction is a request from the CPU to a specific channel/unit combination for a data word. If data is available, the data word is transferred immediately to the A register. If data is not available, the Condition register is set to "Not Zero" to allow a programmed delay.

Normally, the 24-bit input data word contains a single data character. The actual number of data bits per character depends on the peripheral unit involved in the transfer. For example, the console typewriter generates an 8-bit character and the card reader generates a 12-bit character. In any case, the character is right-justified in the A register with the unused bit positions set to ZEROs.

PERIPHERAL DEVICE		23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TELETYPE: MODELS 6001-1, 6001-2, 6001-3, 6001-5	COMMAND WORD																		CLEAR INPUT BUFFER	DISABLE INPUT	ENABLE KEYBOARD	ENABLE READER	SELECT OUTPUT INTERRUPT	SELECT INPUT INTERRUPT	INTERRUPT CONTROL
	STATUS WORD																				KEYBOARD ENABLED	READER ENABLED	OUTPUT BUFFER EMPTY	INPUT BUFFER FULL	UNIT ON-LINE
PAPER TAPE READER: MODELS 6002-10, 6002-20	COMMAND WORD																		CLEAR INPUT BUFFER	DISABLE READER	0	ENABLE READER	0	SELECT INPUT INTERRUPT	INTERRUPT CONTROL
	STATUS WORD																			GATE OPEN	0	READER ENABLED	0	INPUT BUFFER FULL	UNIT ON-LINE
PAPER TAPE READER: MODELS 6003-30, 6003-32	COMMAND WORD																	FORWARD (0) REVERSE (1)	CLEAR INPUT BUFFER	DISABLE READER	0	ENABLE READER	0	SELECT INPUT INTERRUPT	INTERRUPT CONTROL
	STATUS WORD																				FORWARD/ REVERSE SELECTED	READER ENABLED	0	INPUT BUFFER FULL	UNIT ON-LINE
PAPER TAPE PUNCH: MODEL 6003-1	COMMAND WORD																		CLEAR OUTPUT BUFFER	0	DISABLE PUNCH (POWER-OFF)	ENABLE PUNCH (POWER-ON)	SELECT OUTPUT INTERRUPT	0	INTERRUPT CONTROL
	STATUS WORD																			TAPE LOW	0	0	OUTPUT BUFFER EMPTY	0	UNIT ON (POWER ON)
PAPER TAPE PUNCH: MODELS 6003-30, 6003-34	COMMAND WORD																	FORWARD (0) REVERSE (1)	0	0	0	0	SELECT OUTPUT INTERRUPT	0	INTERRUPT CONTROL
	STATUS WORD																			TAPE LOW	TAPE HANDLING ERROR	0	OUTPUT BUFFER EMPTY	0	UNIT ON-LINE
CARD READER: MODELS 3010, 3020, 3030	COMMAND WORD																				EJECT CARD	FEED CARD	SELECT TROUBLE INTERRUPT	SELECT INPUT INTERRUPT	INTERRUPT CONTROL
	STATUS WORD																		COMMAND BUFFER FULL	NON-PICK	0	HOPPER CHECK	READER TROUBLE	INPUT BUFFER FULL	UNIT ON-LINE
CARTRIDGE DISC: MODELS 5201, 5210	COMMAND WORD	0 0 0 RESTORE; 0 0 1 SEEK; 0 1 0 WRITE EOF; 0 1 1 EXPANDED STATUS; 1 0 0 READ; 1 1 0 WRITE; 1 1 1 WRITE DATA/EOF			DRIVE ADDRESS		DISC SELECT	CYLINDER ADDRESS									0	0	HEAD ADDRESS	0	SECTOR ADDRESS					
	STATUS WORD										CYLINDER ADDRESS ERROR	HEAD ADDRESS ERROR	SECTOR ADDRESS ERROR	NO SYNC PATTERN	INVALID SECTOR ADDRESS	0	SEEK ERROR	BUSY (1) READY (0)	0	END OF FILE	CHECKSUM ERROR	EXPANDED STATUS ACTIVE	FILE UNSAFE	READ ONLY	SELECTED DRIVE ON-LINE
LINE PRINTER: MODELS 4005, 4006	COMMAND WORD																	PAPER FEED MODE	VERTICAL FORMAT CONTROL	UNCONDI-TIONAL TOP-OF-FORM	PRINT AND FILL COMMAND	RESET BUFFER	SELECT TROUBLE INTERRUPT	SELECT READY INTERRUPT	INTERRUPT CONTROL
	STATUS WORD																					BUSY	SLEWING/ DATA	PRINTING/ DATA	UNIT ON-LINE
LINE PRINTER: MODEL 4010	COMMAND WORD																	VFC SELECT	VFC MODE	0	PRINT	CLEAR PRINT BUFFER	SELECT TROUBLE INTERRUPT	SELECT READY INTERRUPT	INTERRUPT CONTROL
	STATUS WORD																						TROUBLE	BUSY	UNIT ON-LINE
LINE PRINTER: MODEL 4020	COMMAND WORD																				RESET CONTROLLER BUFFER	MODE SELECT	SELECT TROUBLE INTERRUPT	SELECT READY INTERRUPT	INTERRUPT CONTROL
	STATUS WORD																					INTERFACE VERIFICA-TION	TROUBLE CONDITION	PRINTER READY	UNIT ON-LINE
MOVING HEAD DISC: MODELS 5100, 5110, 5112, 5114	COMMAND WORD	0 0 SEEK; 0 1 WRITE EOF; 1 0 READ; 1 1 WRITE		0 1	DRIVE ADDRESS			CYLINDER ADDRESS									TRACK ADDRESS				SECTOR ADDRESS				
		0 0 RESTORE; 0 1 PRESENT; EXPANDED STATUS		0 1	DRIVE ADDRESS											1	1								
	STATUS WORD										CYLINDER ADDRESS MISMATCH	TRACK ADDRESS MISMATCH	SECTOR ADDRESS MISMATCH	NO SECTOR SYNC PATTERN	INVALID SECTOR ADDRESS	CYLINDER MISMATCH FROM DRIVE	SEEK ERROR	BUSY/ READY	READ/ WRITE UNDER-FLOW	END OF FILE	READ CHECKSUM ERROR	EXPANDED STATUS ACTIVE	DRIVE UNSAFE	READ ONLY	DRIVE ON-LINE
FIXED HEAD DISC: MODELS 5300, 7200B, H	COMMAND WORD	0 0 NO OPERATION; 0 1 WRITE EOF; 1 0 READ; 1 1 WRITE		0	UNIT ADDRESS			HEAD ADDRESS									SECTOR ADDRESS								
		0 PRESENT SECTOR STATUS		1	UNIT ADDRESS																				
	STATUS WORD																	BUSY	INVALID SECTOR ADDRESS	END OF FILE	CHECKSUM ERROR	SECTOR ADDRESS ERROR	HEAD ADDRESS ERROR	READ ONLY	DRIVE ON-LINE
CARD PUNCH: MODELS 3170, 3172	COMMAND WORD																		CLEAR PUNCH MEMORY	SKIP	PUNCH FEED	PUNCH OFF-LINE	SELECT READY INTERRUPT	0	INTERRUPT CONTROL
	STATUS WORD																				PUNCH READY	PUNCH ON-LINE	BUFFER READY	TROUBLE	CONTROLLER ON-LINE
MAGNETIC TAPE TRANSPORT: MODELS 6007, 6008, 6009, 6010	COMMAND WORD	ABC CONTROL								EDIT	PARITY	DENSITY SELECT		CHARACTERS PER WORD			OPERATION CODE				TRANSPORT ADDRESS			SELECT TROUBLE INTERRUPT	SELECT READY INTERRUPT	INTERRUPT CONTROL
	STATUS WORD																	BUSY	LOAD POINT	END OF FILE	END OF TAPE	READ OVERFLOW	TROUBLE	FILE PROTECTED	ON-LINE
		23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

MI60-101A-972

Figure 4-2. Standard Peripheral Unit Bit Assignments for Command and Status Words



TD1516-873

Figure 4-3. OCW Instruction Format

Table 4-1. Peripheral Unit Interrupt Control

Command Word Bit Configuration			Action
2	1	0	
0	0	0	No Action.
0	0	1	No Action.
0	1	0	Disable Input (or Alternate) Interrupt
0	1	1	Enable Input (or Alternate) Interrupt
1	0	0	Disable Output (or Alternate) Interrupt
1	0	1	Enable Output (or Alternate) Interrupt
1	1	0	Disable Both Interrupts
1	1	1	Enable Both Interrupts

Assuming the data character contains no more than 12 bits, more than one character may be packed in the A register through the use of the Merge feature. When a character Merge is employed, a logical OR is performed between the previous contents of the A register and the new input data word. Without the Merge, the previous contents of A are destroyed upon transfer of a new character to A. An illustration of the character Merge technique, as compared to a normal IDW instruction, is shown on Figure 4-4.

B. Output

When an Output Data Word (ODW) instruction is executed, a 24-bit data word is transferred from the A register to the specified channel. The data word is subsequently transferred from the channel to the unit that is currently connected. If the channel is busy or not ready to accept the data word, the Condition register is set to "Not Zero" to allow a programmed delay. If the unit is not ready to accept the data from the channel, the data remains in the channel buffer.

As soon as the peripheral unit is able to accept the data from the channel, the channel-to-unit transfer is made, thereby freeing the channel buffer for another data (or command) word from the CPU.

The number of data bits accepted by the peripheral unit varies according to the type of unit involved. Some peripheral units are word-oriented and accept the entire 24-bit word. Others are character-oriented and accept only a specific number of bits per character.

4-4.5 Automatic Block Transfer

A block data transfer is controlled by the Automatic Block Controller (ABC). A block transfer is initiated under program control and proceeds from that point without further program intervention until the transfer is completed or until another transfer is initiated.

Two ABC registers initialize and subsequently control a block transfer. They are the Transfer Address Register (TAR) and the Word Count Register (WCR). TAR contains the address of the first word in a block of data. The WCR defines the number of words to be transferred.

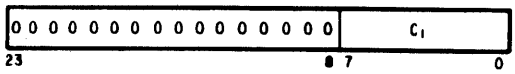
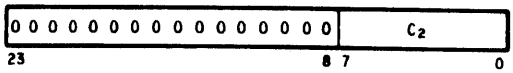
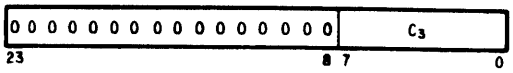
An Automatic Restart Flag permits the sequential transfer of more than one block of data. If this flag is set when the initial data block transfer is completed, the next data block will be transferred without program intervention.

The ABC is initialized by loading the address of the first of a pair of block transfer parameters in the TAR with an OAW instruction. These parameters are stored in sequential memory locations and are illustrated on Figure 4-5. The OAW instruction transfers the contents of register A to the TAR. An OAW instruction does not clear the I/O buffers nor does it initiate a block transfer.

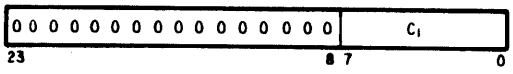
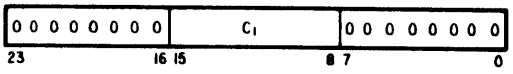
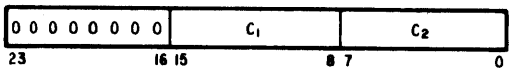
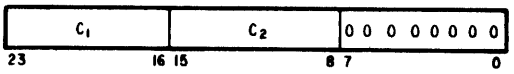
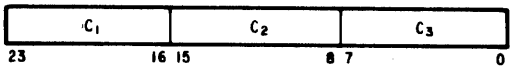
The first parameter word (Word 1) is the word count or number of sequential words to be transferred. They are contained in bits 0-15. If bit 23 of Word 1 is set, the address contained in the TAR during the block transfer will be incremented by 2 for each word transferred to or from the unit, except after the last word transferred which will be incremented by 1. If bit 22 of Word 1 is set, the contents of the computer memory will be dumped into the connected unit at full memory cycle rate without the normal "handshaking" between the I/O channel and unit. The Output Data Here (ODH) signal will be a pulse to indicate to the unit that data is available.

EXAMPLE: THREE 8-BIT DATA CHARACTERS ARE TO BE PACKED IN THE A REGISTER.

(a) NORMAL (WITHOUT MERGE)

CODING	COMMENTS	REGISTER A
IDW CU	BRING IN FIRST DATA CHARACTER	
. . .		
IDW CU	BRING IN SECOND CHARACTER	
. . .		
IDW CU	BRING IN THIRD CHARACTER	
. . .		

(b) MERGE

CODING	COMMENTS	REGISTER A
IDW CU	BRING IN FIRST DATA CHARACTER	
LLA 8	SHIFT LEFT 8 PLACES	
IDW* CU	BRING IN SECOND CHARACTER AND MERGE	
LLA 8	SHIFT LEFT 8 PLACES	
IDW* CU	BRING IN THIRD CHARACTER AND MERGE	
. . .		

MI60-005-770

Figure 4-4. IDW Instruction, Data Character Formatting

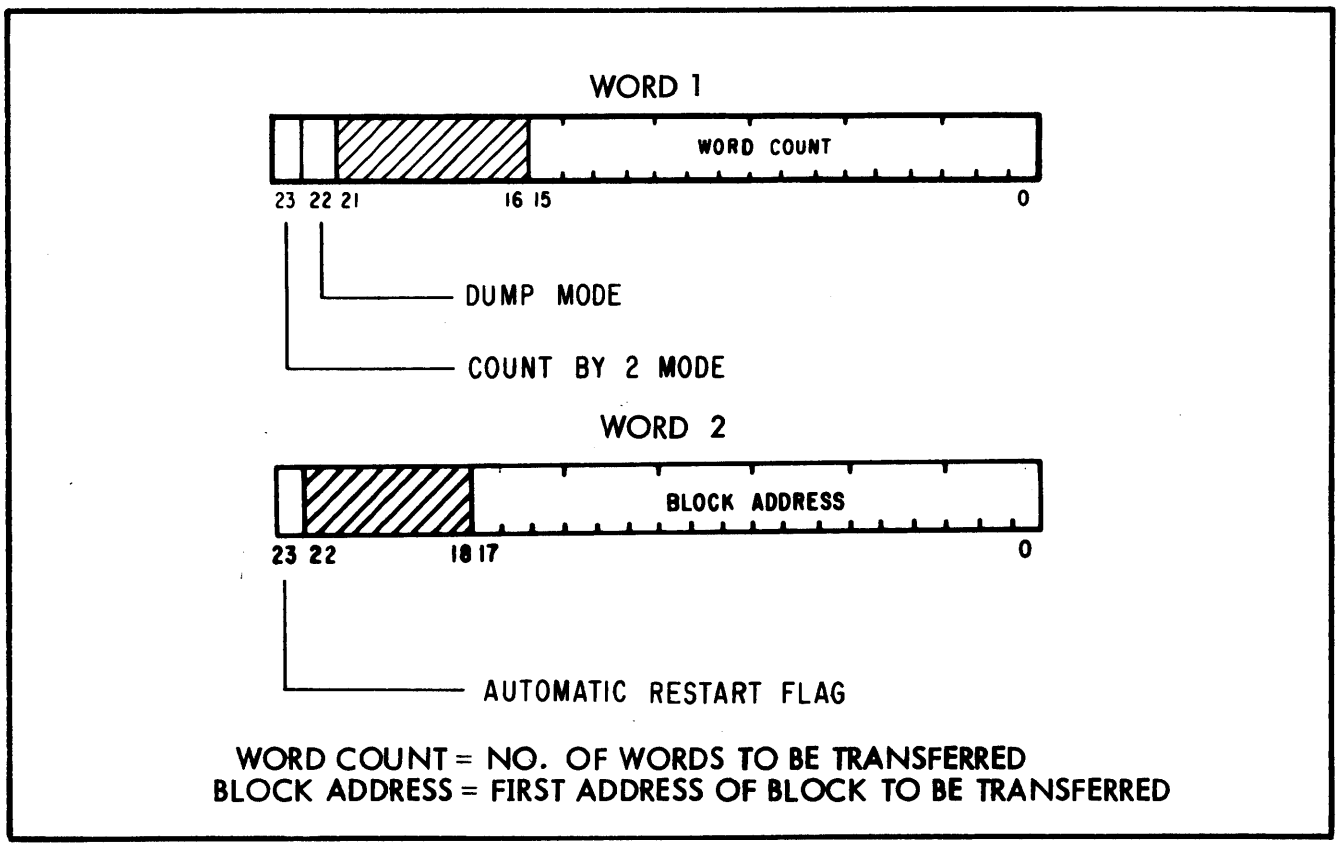


Figure 4-5. Block Transfer Parameters, Storage Format

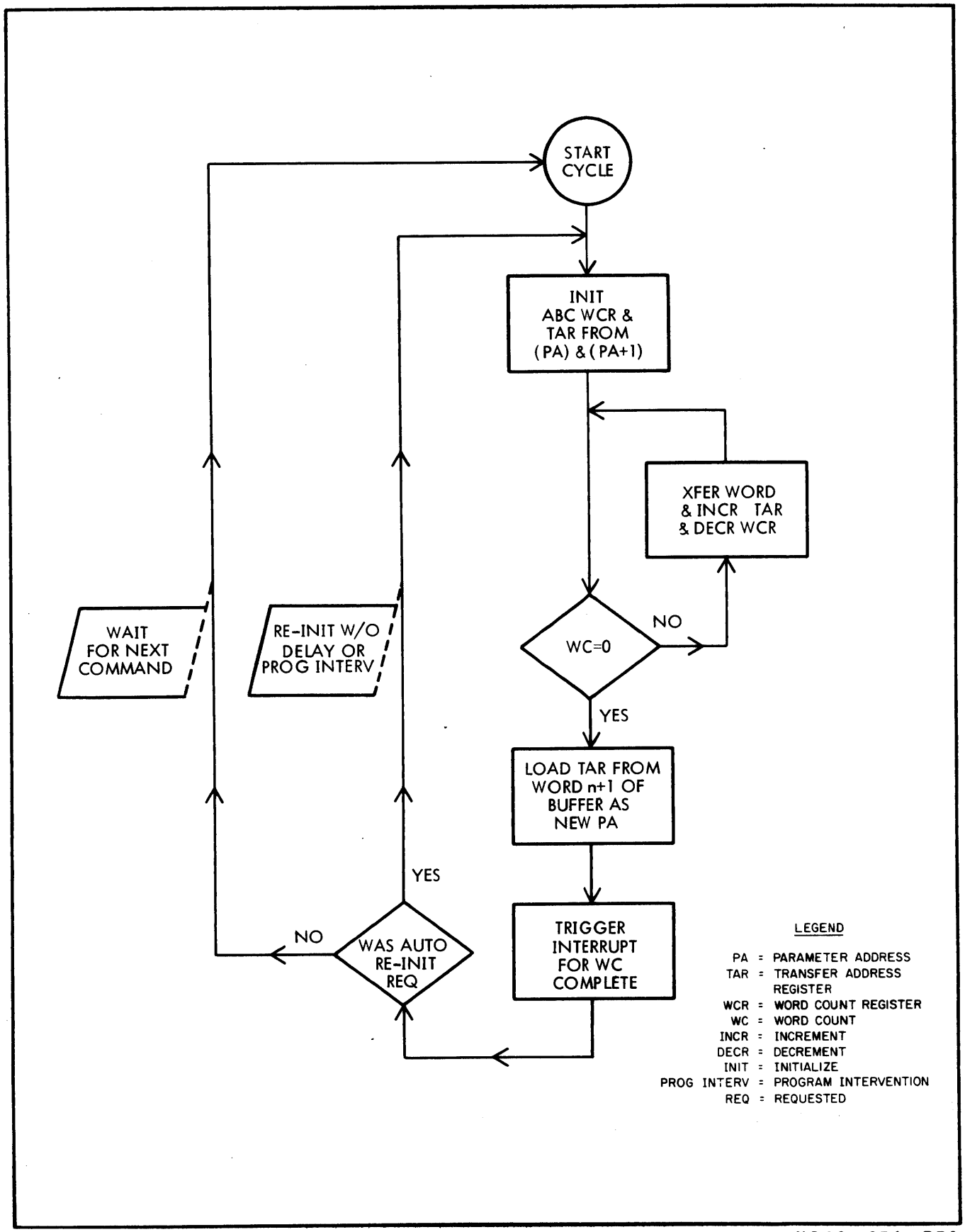
The Output Data Accepted signal will not be used in this mode of operation. The computer will be completely "locked" out of memory once the ABC has started its block and until the sequence is terminated.

The second parameter (Word 2) is the starting address of the block to be transferred. It is contained in bits 0-17. If bit 23 is set (Automatic Restart Flag), a new transfer sequence is initiated without program intervention.

During each output transfer, the ABC logic will access data at the current memory location stored in the TAR. The output buffer is loaded and data is transferred to the unit controller over the standard I/O channel lines. The TAR is incremented and the WCR decremented. After the unit controller responds, the transfer sequence is automatically repeated. (Refer to Figure 4-6.)

During each input transfer, the ABC logic will transfer data to the current memory location stored in the TAR whenever the unit controller generates the appropriate signal. The TAR is incremented and the WCR decremented. The sequence is repeated as soon as the unit controller responds with the signal.

The OCW instruction, which initiates the block transfer, can also be used to terminate the transfer sequence as a result of the Override bit (bit 11) being set in the OCW format.



MI60-036-770

Figure 4-6. ABC Logic - Flow Diagram

The OCW instruction selects and activates the channel and one of its assigned units. This instruction contains a 9-bit code which specifies the unit/channel combination. Five bits comprise the channel code, while four bits designate the unit. The four bit unit code is strobed to the unit controller by the I/O channel timing and control logic.

After the ABC is initialized, words are transferred between the selected peripheral unit and computer memory via the ABC. During the data word transfer, the Transfer Address Register (TAR) is incremented and the Word Count Register (WCR) is decremented. Subsequent data words are transferred by repetition of this cycle. When the word count is decremented to zero, the block transfer is terminated unless bit 23 of the address register is set. When bit 23 is set, a new block transfer is automatically initiated by the ABC.

4-5 ABC APPLICATIONS

The following examples illustrate three different ABC applications.

Example 1: Simple, single buffer input.

```
        TOA     PA      Parameter Address
        OAW     C       Initialize TAR
        TMA     CW      Command Word
        OCW     CU      Initiate transfer
        BNZ     *-1     Delay if channel is busy
        . . .
CW      DATA            Bit 23 and others as required by the I/O device
PA      DAC     n       Absolure Word Count
        DAC     BUFF    Address of Input buffer
BUFF    BLOK    n       Reserve n words. Word n+1 is of no
                        significance since the AR bit is not set.
```

Example 2: Multi-buffered output with automatic restart and buffer switching.

```
        TOA     PA1     Parameter Address 1
        OAW     C       Initialize TAR
        TMA     CW      Command Word
        OCW     CU      Initiate first transfer
        BNZ     *-1     Delay if channel is busy
        . . .
CW      DATA            Bits 23,22,and others as required by the I/O
                        device
PA1     DAC     n       Word Count
        DAC*    BUF1    Address of buffer 1 and the ARF (*)
PA2     DAC     n       Word Count
        DAC*    BUF2    Address ot buffer 2 and the ARF (*)
BUF1    BLOK    n       Reserve n words
        DAC     PA2     Automatic Reinitialization address for TAR,
                        to switch buffers
BUF2    BLOK    n       Reserve n words
        DAC     PA1     Automatic Reinitialization address for TAR,
                        to switch buffers
```

NOTE

Once this cycle is initiated it will continue, without program intervention, until a new command is received.

Example 3: Multi-buffer input with automatic buffer switching but without Automatic Restart.

```
        TOA     PA1     Parameter Address 1
        OAW     C       Initialize TAR
        TOI     BUF1    Setup pointer for buffer to be processed
        TIM     BA
        TOI     BUF2    Prepare for initial interchange
        TMA     CW      Command Word
        OCW     CU      Initiate first transfer
        BNZ     *-1     Wait until channel takes command
SBAR    IMI     BA      Switch process pointer
        TMA     CW      Command Word
        OCW     CU      Initiate next transfer
        BNZ     *-1     As soon as channel is free
        . . .           Process current buffer, whose base address is
        . . .           in register I, while the alternate buffer is
        . . .           being filled.
        . . .
        BUC     SBAR    To switch buffers and restart ABC
        . . .
CW      DATA            Bits 15, 14 and others required by I/O device
BA      ZZZ             Alternating buffer address
PA1     DAC     n       Word count only
        DAC     BUF1    Buffer 1 address
PA2     DAC     n       Word count
        DAC     BUF2    Buffer 2 address
BUF1    BLOK    n       Reserve n words
        DAC     PA2     Automatic Reinitialization address for TAR
BUF2    BLOK    n       Reserve n words
        DAC     PA1     Automatic Reinitialization address for TAR
```

NOTE

This example illustrates an application in which the processing rate is slower than the input rate via ABC.

4-6 I/O INSTRUCTION FORMAT SUMMARY

The I/O instructions used in conjunction with the DC 6024/4 are briefly discussed in the following paragraphs. Refer to the I/O portion of Section III for illustrations of the I/O formats.

4-6.1 Output Command Word (OCW)

An OCW instruction transfers a command word (either 8 or 24 bits) to the specified channel/unit combination. The command word normally specifies the unit control function(s) to be performed and/or the I/O condition to be established.

Bits 0-3 of the OCW instruction form a 4-bit paralleled unit code that is used to select a particular unit. The configuration of bits 4 and 5 determines the Multiplex or Offline mode for a particular channel. (Refer to Sections I and II for discussions on these two modes.) The configuration of bits 6-10 determines which channel is to be selected. Bit 11 —— the override bit —— when set (1) causes the command word to assume immediate control over the channel. When not set (0), the command word will not be accepted if the channel is currently performing an operation. Bits 12-23 (Op Code bits) define the general process that is to be performed.

4-6.2 Input Status Word (ISW)

An ISW instruction is used to test the operational status of a specified peripheral unit. When an ISW instruction is executed, the channel control logic gates either 6 or 8-bit status lines to the computer.

Bits 0-3, 6-10, and 12-23 are used in the ISW identically to the configuration of the OCW; that being the unit code, channel section, and operand code, respectively. Bit 11 will be reset (0) so that the ISW instruction can be distinguished from an IAW instruction. Also, bits 4 and 5 are not used.

4-6.3 Input Data Word (IDW)

An IDW instruction is a request from the computer to a specified channel/unit combination for a data word. If data is available, the data word is transferred to the computer; if data is not available, there is a programmed delay. The actual number of data bits depends on the type of peripheral unit involved in the transfer.

Bit 11 of the IDW, which is the Merge bit, is the only configuration change from that of the OCW. If the Merge bit is 0, the computer register is cleared prior to the data transfer. The input data is right-justified in the computer register. If the Merge bit is a 1, an OR function is performed between the previous contents of the computer register and the incoming word. (This feature permits the input data characters to be packed in the computer register.)

4-6.4 Output Data Word (ODW)

An ODW instruction is a register to transfer a data word from the computer to the unit that is currenily connected. If the channel is busy or not ready to accept the data, there is a programmed delay. If the unit is not ready to accept the data from the channel, the data remains in the channel output buffer; data transfer is made as soon as the unit is able to accept the data from the buffer.

Bit 11 of the ODW will be reset to distinguish the ODW from an OAW instruction. All other bit definitions are standard (as explained in the OCW discussion).

4-6.5 Input Address Word (IAW)

An IAW instruction transfers the current address of the Transfer Address Register (TAR) in the Automatic Block Controller (ABC) channel into the computer register. (This instruction is normally used only in ABC operations.)

The IAW instruction is directed only to the channel; therefore, bits 0-3 (which are normally used for unit decode) are ignored - as are bits 4 and 5. Bits 6-10 determine which channel is to be selected. The operand code (bits 12-23) defines the general process that is to be performed. Bit 11, when set (1), distinguishes the IAW instruction from an ISW (0) instruction.

4-6.6 Output Address Word (OAW)

An OAW instruction is normally used only in ABC operations. This instruction transfers the contents of the computer register to the ABC address register. This instruction is used to set up the address register with the address of the parameter list that is used by the ABC to obtain the starting address and word count for the block.

The OAW instruction is directed only to the channel; therefore, bits 0-3 (normally used for unit decode) are ignored - as are bits 4 and 5. Bits 6-10 determine which channel is to be selected. The operand code (bits 12-23) defines the general process that is to be performed. Bit 11, when set (1), distinguishes the OAW from an ODW instruction.

APPENDIX A
BOOTSTRAP PROGRAMS

```
                        ********** DISC BOOTSTRAP **********
        00000500    CU      EQIV  '500

000001 62500013             TDA   WC
000002 00714500             DAW   CU
000003 05000012             TMA   CW
000004 00700500             DCW   CU
000005 00730500             ISW   CU
000006 22600005             BNZ   *-1
000007 00110200             QBB   B7
000010 22600005             BNZ   *-3
000011 21000020             BUC   120
000012 40000000    CW       DATA  B23
000013 00000100    WC       DAC   1100
000014 00000020             DAC   120
000015 00400000             END


                        ********** CARD BOOTSTRAP **********
        00000400    CU      EQIV  1400

000001 00030010             TDB   110
000002 00700400             DCW   CU
000003 00720400             IDW   CU
000004 22600003             BNZ   *-1
000005 00420014             LLA   12
000006 00724400             IDW*  CU
000007 22600006             BNZ   *-1
000010 00240020             CZA
000011 22200020             BDZ   120
000012 15100020             TAM   120,I
000013 23100003             BWI   *-8
000014 00400000             END


                        ********** 7-TR. MAGNETIC TAPE BOOTSTRAP *****
        00000700    CU      EQIV  1700

000001 62500406             TDA   1406
000002 00700700             DCW   CU
000003 62500015             TDA   CW
000004 00714700             DAW   CU
000005 05000015             TMA   CW
000006 00700700             DCW   CU
000007 22600006             BNZ   *-1
000010 00730700             ISW   CU
000011 22600010             BNZ   *-1
000012 00110204             QBB   B7B2
000013 22600010             BNZ   *-3
000014 21000020             BUC   120
000015 40036506    CW       DATA  140036506
000016 00000020             DAC   120
000017 00400000             END
```

```
                        ********** ASR TAPE READER BOOTSTRAP *******

000001 00030110                 TOB     '110
000002 00700000                 OCW     0
000003 00720000                 IDW     0
000004 00140000                 COB     0
000005 22200003                 BOZ     *-2
000006 63200004                 TNJ     4
000007 00420006                 LLA     6
000010 00724000                 IDW*    0
000011 22600010                 BNZ     *-1
000012 23200007                 BWJ     *-3
000013 00240020                 CZA
000014 22200020                 BOZ     '20
000015 15100020                 TAM     '20,I
000016 23100006                 BWI     *-8
000017 00400000                 END

                        ********** HIGH SPEED PAPER TAPE BOOTSTRAP **
       00000100         CU      EQIV    '100

000001 00030110                 TOB     '110
000002 00700700                 OCW     CU
000003 00720700                 IDW     CU
000004 00140000                 COB     0
000005 22200003                 BOZ     *-2
000006 63200004                 TNJ     4
000007 00420006                 LLA     6
000010 00724700                 IDW*    CU
000011 22600010                 BNZ     *-1
000012 23200007                 BWJ     *-3
000013 00240020                 CZA
000014 22200020                 BOZ     '20
000015 15100020                 TAM     '20,I
000016 23100006                 BWI     *-8
000017 00400000                 END

                        ********** 9-TR, MAGNETIC TAPE BOOTSTRAP ****

       00000700         CU      EQIV    '700
000001 62500406                 TOA     '406
000002 00700700                 OCW     CU
000003 62500015                 TOA     CW
000004 00714700                 OAW     CU
000005 05000015                 TMA     CW
000006 00700700                 OCW     CU
000007 22600006                 BNZ     *-1
000010 00730700                 ISW     CU
000011 22600010                 BNZ     *-1
000012 00110204                 QBB     B7B2
000013 22600010                 BNZ     *-3
000014 21000020                 BUC     '20
000015 40034506         CW      DATA    '40034506
000016 00000020                 DAC     '20
000017 00400000                 END$
```

READER COMMENT SHEET

This "comment" mailer is included to help us evaluate and evolve our documentation. Datacraft Corporation solicits your comments.

Manual Title_____ Pub. No._____

FROM:

Name_____ Title _____

Equipment Application_____

Company_____

Staple or Tape

Fold

NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 3027, FT. LAUDERDALE, FLORIDA

POSTAGE WILL BE PAID BY

Manager, Marketing Communications
Datacraft Corporation
1200 Gateway Drive
Fort Lauderdale, Florida 33309

Fold

Staple or Tape