

digital

**PROGRAMMED DATA
PROCESSOR - 8/S
USERS HANDBOOK**

Copyright © 1966 by
Digital Equipment Corporation

PDP is a registered trademark
of Digital Equipment Corporation.

CONTENTS

| | |
|---|-----|
| Introduction | v |
| Chapter 1 Memory and Processor Functional Operation | 1 |
| Chapter 2 Memory and Processor Instructions..... | 7 |
| Chapter 3 Memory and Processor Basic Programming..... | 20 |
| Chapter 4 Program Interrupt | 27 |
| Chapter 5 Memory Parity | 29 |
| Chapter 6 Basic IOT Programming | 31 |
| Chapter 7 Teletype and Control..... | 35 |
| Chapter 8 High Speed Perforated Tape Reader and Control Type PC02 | 41 |
| Chapter 9 High Speed Tape Punch Control Type PC03 | 43 |
| Chapter 10 Analog-To Digital Converter Type 138E and Multiplexer Control Type 139E | 45 |
| Chapter 11 Digital-To-Analog Converter Type AA01A | 50 |
| Chapter 12 Incremental Plotter and Control Type 350B | 51 |
| Chapter 13 Card Reader and Control Type CR01C | 55 |
| Chapter 14 Card Reader and Control Type 451 | 59 |
| Chapter 15 Automatic Line Printer and Control Type 645 | 62 |
| Chapter 16 Standard PDP-8/S Operation | 66 |
| Chapter 17 PDP-8/S Input/Output Facilities | 74 |
| Chapter 18 Programmed Data Transfers | 76 |
| Chapter 19 Installation Planning | 95 |
| Appendix 1 Instructions | 98 |
| Appendix 2 Codes | 104 |
| Appendix 3 Scales of Notation | 108 |
| Appendix 4 Powers of Two | 109 |
| Appendix 5 Octal-Decimal Conversion..... | 110 |
| Appendix 6 Perforated Tape Loader Sequences | 117 |

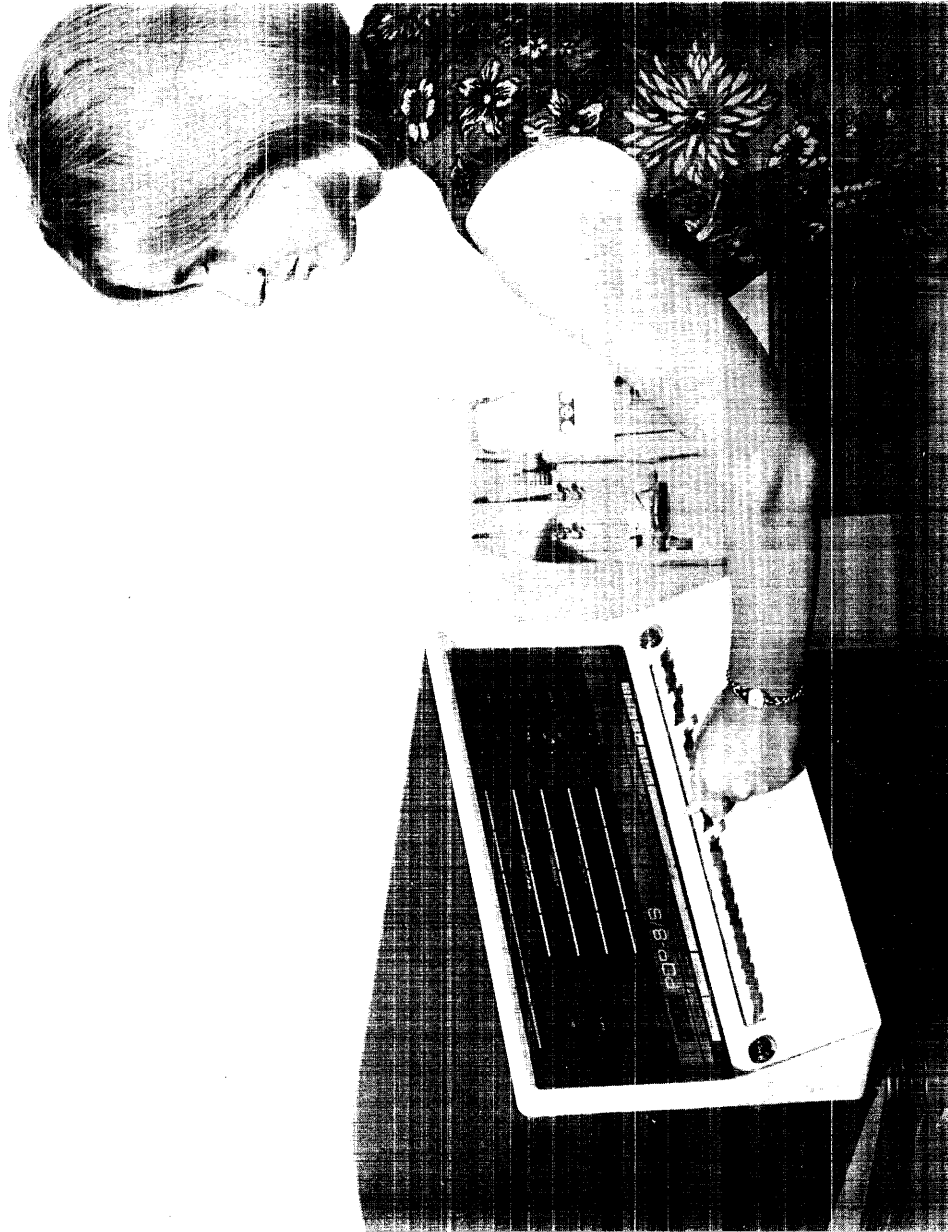


Figure 1. The PDP-8/S System

INTRODUCTION

Digital Equipment Corporation's Programmed Data Processor-8/S (PDP-8/S) is designed for use as a small scale general-purpose computer, an independent information handling facility in a larger computer system, or as a control element in a complex processing system. The PDP-8/S is a one-address, fixed word length, serial computer using a word length of 12-bits plus parity and two's complement arithmetic. Cycle time of the 4096-word random address magnetic core memory is 8 microseconds. Standard features of the system include indirect addressing and facilities for instruction skipping and program interruption as functions of input/output device conditions.

Addition is performed in 36 microseconds with one number in the accumulator. Subtraction is performed in 64 microseconds with the subtrahend in the accumulator. Multiplication is performed in approximately 5.35 milliseconds by a subroutine that operates on two signed 12-bit numbers that produce a 24-bit product, leaving the 12 most significant bits in the accumulator. Division of two signed 12-bit numbers is performed in approximately 7.38 milliseconds by a subroutine that produces a 12-bit remainder in core memory.

Flexible, high capacity, input/output capabilities of the computer operate a variety of peripheral equipment. In addition to standard teletype and perforated tape equipment, the system can operate in conjunction with all non-data break optional devices offered in the PDP-8 family line. Equipment of special design is easily adapted for connection into the PDP-8/S system. The computer is not modified with the addition of peripheral devices.

The PDP-8/S is completely self-contained requiring no special power sources or environmental conditions. A single source of a 115-volts, 60 cycle, single phase power operates the machine. An internal power supply produces all of the required operating voltages. Standard FLIP CHIP Modules utilizing hybrid silicon circuits insure reliable operation in ambient temperatures between 32 and 130 degrees Fahrenheit.

COMPUTER ORGANIZATION

The PDP-8/S consists of a central processor, a memory unit, and input/output equipment and facilities. All arithmetic, logic, and system control operations of the standard PDP-8/S are performed by the processor. The central processor and memory are independent and asynchronous. The processor requests memory cycles only when required, each memory cycle consisting of a read followed by a write operation. Input and output addresses and data buffering for the core memory are performed by registers of the processor, and operation of the memory is under control of its own timing logic.

Interface circuits for the processor allow bussed connections to a variety of peripheral equipment. Each input/output device is responsible for detecting its own selection code and for providing any necessary input or output gating. Individually programmed data transfers between the processor and peripheral equipment take place through the processor accumulator. Standard features of the PDP-8/S also allow peripheral equipment to perform certain control functions such as instruction skipping, and a transfer of program control when initiated by a program interrupt.

Standard peripheral equipment provided with each PDP-8/S system consists of a Teletype Model 33 Automatic Send/Receive set and a Teletype control. The Teletype unit is a standard machine operating from serial 11 unit code characters at a rate of 10 characters per second. The teletype provides a means of supplying data to the computer from perforated tape or by the keyboard, and supplies data as an output from the computer in the form of perforated tapes or typed copy. The Teletype control serves as a serial-to-parallel converter for Teletype inputs to the computer, and serves as a parallel-to-serial converter for computer output signals to the Teletype unit. The Teletype control functions are performed by two functional modules which are built from monolithic integrated circuits and mounted on two standard double size FLIP CHIP cards.

SYMBOLS

The following special symbols are used throughout this handbook to explain the function of equipment and instructions:

| <u>Symbol</u> | <u>Explanation</u> |
|------------------------|---|
| $A = > B$ | The content of register A is transferred into register B |
| $O = > A$ | Register A is cleared to contain all binary zeros |
| A_j | Any given bit in A |
| A_5 | The content of bit 5 of register A |
| $A_5(1)$ | Bit 5 of register A contains a 1 |
| $A_6 - 11$ | The content of bits 6 through 11 of register A |
| $A_6 - 11 = > B_0 - 5$ | The content of bits 6 through 11 of register A is transferred into bits 0 through 5 of register B |
| Y | The content of any core memory location |
| V | Inclusive OR |
| ∇ | Exclusive OR |
| \wedge | AND |
| \overline{A} | Ones complement of the content of A |

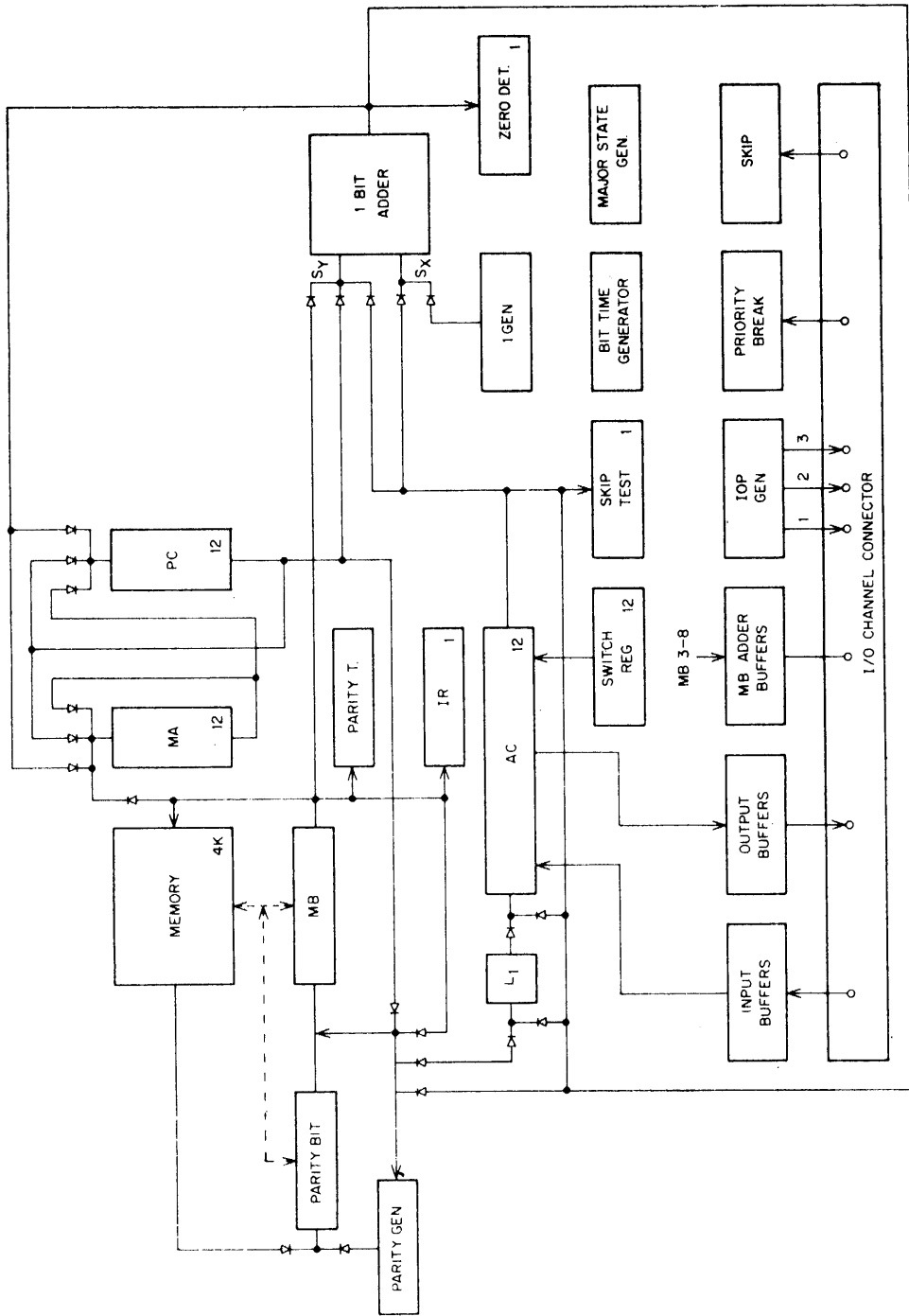


Figure 2. PDP-8/S Major Register Block Diagram

CHAPTER 1

MEMORY AND PROCESSOR FUNCTIONAL DESCRIPTION

MAJOR REGISTERS

To store, retrieve, control, and modify information and to perform the required logical, arithmetic, and data processing operations, the central processor employs the logic components shown in Figure 2 and described in the following paragraphs.

ACCUMULATOR (AC)

Arithmetic and logic operations are performed in this 12-bit register. Under programmed control, the AC can be cleared or complemented, its content can be rotated right or left with the link. The content can be rotated right or left with the link. The content of the memory buffer register can be added to the content of the AC and the result left in the AC. The content of both of these registers may be combined by logical operation and the result left in the AC. An inclusive OR may be performed between the AC and the switch register on the operator console and the results left in the AC. The accumulator also serves as an input/output register. All programmed information transfers between memory and an external device pass through the accumulator. In addition to these functions, the accumulator also serves as a buffer for the switch register during manual console operation. Loading an address into the program counter is accomplished by loading the content of the switch register into the accumulator and then serially transferring the content of the accumulator through the adder to the program counter register.

LINK (L)

This one-bit register is used to extend the arithmetic facilities of the accumulator. It is used as the carry register for two's complement arithmetic. Overflow into the L from the AC can be checked by the program to greatly simplify and speed up single and multiple precision arithmetic routines. Under program control the link can be cleared and/or complemented, and it can be rotated as part of the accumulator.

PROGRAM COUNTER (PC)

The program sequence (the order in which instructions are performed) is determined by the PC. This 12-bit register contains the address of the memory locations from which the next instruction will be taken. Information enters the PC from the memory via the memory address register and from the switch register on the operator console via the accumulator and adder. Information in the PC is transferred to the memory address register to determine the memory address from which each instruction is taken. Incrementation of the contents of the PC is performed through the adder and establishes the successive memory locations of the program and provides skipping of instructions based upon a programmed test of information or external conditions.

MEMORY ADDRESS REGISTER (MA)

The address in memory which is currently selected for reading or writing is contained in this 12-bit register. Therefore, 4096 words of memory can be addressed directly by this register. Data is transmitted to the MA from the PC or from the memory buffer register.

SWITCH REGISTER (SR)

Information can be manually set into the switch register for transfer into the PC as an address by means of the LOAD ADDRESS key, or into the MB if data is to be stored in core memory by means of the DEPOSIT key. The switch register may also be interrogated by programmed instruction and its contents placed in the accumulator. All transfers from SR occur through the accumulator.

MEMORY BUFFER REGISTER (MB)

All information transferred between the memory and the processor are temporarily held in the MB. Information can be transferred into the MB from the accumulator or program counter register. The MB can transmit or receive data from an external memory in either serial form or parallel form. Information is read from a memory location in 4 microseconds and rewritten in the same location in another 4 microseconds of one 8 microsecond memory cycle.

INSTRUCTION REGISTER (IR)

This 4-bit register contains the operation code of the instruction currently being performed by the machine. The three most significant bits of the current instruction plus the indirect address bit are loaded into the IR from the memory buffer register during a Fetch cycle. The content of the IR is decoded to produce the 8 basic instructions and affects the cycles and states entered in each step in the program.

PARITY BIT (PB)

Information stored in the memory contains 12 data bits plus one parity bit. The parity bit is transferred from the memory to the PB register and held until the data word is used. After the data word residing in the MB is used, the content of the PB is checked against the parity tester to determine whether bits have been dropped. New data sent to the MB has its parity stored in PB for writing along with the data word into memory.

PARITY TEST (PT)

The PT is a one-bit register which continually examines the output of the memory buffer register. Whenever MB is shifted, the parity test flip-flop generates a new parity bit for the 12-bit data word contained in the MB. The content of the PT is compared with the content of the PB, and if they are not the same, a parity error is generated.

PARITY GENERATOR (PG)

The Parity Generator examines all incoming data to the MB and generates a new parity bit. The parity bit is then transferred into the PB register just prior to requesting a write memory cycle.

SKIP TEST (SKP)

The skip test flip-flop examines the output of the AC during the microinstructions which test the content of the AC. If the SKP is set, the PC is incremented by one, causing the processor to skip the next instruction. An external input to the SKP allows peripheral hardware to generate a skip request during an IOT instruction.

SERIAL ADDER (SA)

The SA is a one-bit full serial adder and includes a carry flip-flop, which is used to hold the carry between successive bit additions, and a constant generator, which provides a + constant used in incrementing either the PC, the AC, or the MB. During a TAD instruction, the MB and AC are shifted bit by bit through the serial adder and the output of the serial adder is shifted into the front end of the AC and L. The incrementing and loading of the PC is performed through the adder. A zero detector (ZD) continuously observes the output of the adder during its use for instructions such as ISZ.

MAJOR STATE GENERATOR

Two more major states are entered serially to execute program instructions. The major state generator establishes one state for each computer timing cycle, every instruction requiring three basic states; the fetch state, the execute state, and the end state. In some cases the execute and end state of the instructions may occur simultaneously. This is so if the instruction does not require the use of the memory during the execute portion of the instruction, or if the use of the serial adder is required both during the execute and end states. In addition to these basic states, there are index and defer states and the break states. Entry into each state is produced as a function of the current instruction and the current state.

The major states of the machine are also referred to as word times, for during these times the data or instruction words are shifted between registers in the central processor.

BIT TIME GENERATOR

A major state or word time consists of a set of timing pulses provided by the bit time generator. There are 14 consecutive pulses to each word time. The first 12 of the 14 pulses BT 0 through 11, are used for inter-register shifting in the processor. The 13th pulse is used for housekeeping operations inside the central processor as checking parity, generating parity alarm, requesting memory cycles if necessary, and handling the link during a TAD, or IAC instruction. The 14th pulse is always the final pulse in any word time and is always responsible for the setting of the next word time as a function of the current instruction and the current word time.

DESCRIPTION OF MAJOR STATES (WORD TIMES)

Fetch Word Time (F)

During this state, the instruction word which was previously placed in the MB by a memory read request is shifted to the IR and the MA. The MB is also shifted end around on itself so that the contents of the MB remains intact. At the same time, the program counter is incremented through the adder in preparation for the next instruction. All of these operations occur simultaneously during bit times 0-11. On bit time 12, parity test is executed on the instruction and a memory read request is set up if required. Also during bit time 12 and group I operate instructions, the accumulator is cleared if required, the link is cleared if required, or the complement link function occurs if required. If the instruction which is now residing in the IR is a two cycle instruction, on bit time 13, word time X (execute state) and word time E state (end state) are entered simultaneously. If the instruction is a 3-cycle instruction, only word time X is entered. If the instruction is an indirect address instruction and does not refer to one of eight auto-index locations (10-17₈) then word time D (defer state) is entered. If the instruction contains an indirect address and does refer to one of the eight auto-indexed locations, then the index word time is entered (I state).

Execute Word Time (X)

All arithmetic and logical manipulations of the AC and MB are performed during word time X. Input/output transfers also occur during word time X. At the conclusion of the execute state, word time E (end state) is always entered.

End Word Time (E)

Regardless of whether or not word time E occurs simultaneously with word time X, its function is always to bring the next instruction to be executed out of the memory into the memory buffer. In addition, if a skip request is present during ISZ, JMS, operate group 2, or IOT instructions, the PC is incremented through the serial adder. Since the PC was already incremented during word time F, the processor effectively skips the next instruction. At the termination of the end state, word time F is always entered unless a priority interrupt is present on the interrupt bus and the interrupt is enabled in which case the break word time is entered.

Index Word Time (I)

During word time I (index) the content of the memory buffer, which holds one of the eight auto-index locations, (10₈-17₈) is incremented by one and replaced in the memory. The incremented content of the auto-index location is still in the memory buffer when the deferred state is entered. This word will be sent as a 12-bit word to the MA and used as the address of the instruction during the Defer Word time.

Defer Word Time (D)

Word time D is always entered after word time I, or after word time F when an indirect bit (IR3) is present. During the word time, the content of the memory buffer is

sent to the memory address register to be used as a 12-bit address for the instruction operand. On termination of the defer state, the execute state or the execute and end states are entered depending on whether the basic instruction was a two or three cycle instruction.

Break Word Time (B)

Should an interrupt request be present on the interrupt line at the completion of word time E, the break state is entered and the content of the program counter is transferred to the MB. A 1 is placed in the program counter and a 0 is placed in the MA. A memory write request is then initiated which will cause the content of the MB (previous C (PC)) to be saved in location 0. The interrupt control is disabled during the break state. At the termination of the break state, the end state is always entered, and the next instruction will be taken from the current content of the program counter or location 1 in the memory.

INTERFACE

The input/output portion of the PDP-8/S is extremely flexible and interfaces readily with special equipment, especially in real time data processing and control environments. The PDP-8/S utilizes a "bus" I/O system rather than the more conventional "radial" system. The "bus" system allows a single set of data and control lines to communicate with all I/O devices. The bus simply goes from one device to the next. No additional connections to the computer are required. A "radial" system requires that a different set of signals be transmitted to each device; and thus the computer must be modified when new devices are added. The PDP-8/S need not be modified when adding new peripheral devices.

External devices receive two types of information from the computer: data and control signals. Computer output data is present as static levels on 12 lines. These levels represent a 12-bit word to be transmitted in parallel to a device. Data signals are received at all devices but are sampled only by the appropriate one in response to a control signal. Control signals are of two types: levels and timing pulses. Six static levels and their complement are supplied by the MB on 12 lines. These lines contain a code representing the device from which action is required. Each device recognizes its own code and performs its function only when this code is present. There are three timing pulses which may be programmed to occur. These IOP pulses are separated in time by 1 μ sec and are brought to all devices on 3 lines. These pulses are used by a device only when it is selected by the appropriate code on the level lines. They may be used to perform sequential functions in an external register, such as clear and read, or any other function requiring one, two, or three sequential pulses.

Peripheral devices transmit information to the computer on four types of "busses". These are the information bus, the clear AC bus, the skip bus, and the program interrupt bus. The information bus consists of 12 lines normally held at -3 volts by load resistors within the computer. Whenever one of these lines is brought to ground, a binary 1 will be placed in the corresponding accumulator bit. Each device may use the input bus only when it is selected; and thus, these input lines are time shared among all of the connected devices. The skip bus is electrically identical to the information bus. However, when it is driven to ground the next sequential instruction will be skipped. It too can be used only by the device currently selected and is effectively time shares. The program interrupt bus may be driven to ground at any time by any device whether currently selected or not. When more than one device is connected to the interrupt bus they should also be connected to the skip bus so the program can identify the device requesting program interruption.

The transmission of device selection levels and timing pulses is completely under program control. A single instruction can select any one of 64 devices and transmit up to three IOP timing pulses. Since the timing pulses are individually programmable, one might be used to strobe data into an external device buffer, another to transmit data to the computer, and the third to test a status flip-flop and drive the skip bus to ground if it is in the enabling state.

CHAPTER TWO

MEMORY AND PROCESSOR INSTRUCTIONS

Instruction words are of two types: memory reference and augmented. Memory reference instructions store or retrieve data from core memory, while augmented instructions do not. All instructions utilize bits 0-2 to specify the operation code. Operation codes of 0_8 through 5_8 specify memory reference instructions, and codes of 6_8 and 7_8 specify augmented instructions. Since the PDP-8/S operates asynchronously from its memory machine timing is not a multiple of the memory cycle time. Every word time for major states represents a processor time of 10 microseconds. Thus a two cycle instruction requires 20 microseconds worth of processor time. If a major state requires a memory reference, another 8 microseconds is added to the processor cycle time. Example: a two cycle instruction requiring one memory reference consists of 20 microseconds worth of processor time plus 8 microseconds worth of memory cycle time for a total execution time of 28 microseconds. Indirect addressing increases the execution time for the memory reference instruction by 18 microseconds; i.e., 10 microseconds worth of processor time and 8 microseconds worth of memory time. Indirect addressing referencing one of the eight auto-index locations adds an additional 18 microseconds to an indirect memory reference instruction. Augmented instruction, input/output transfer and operate, are performed in two or three machine cycles each requiring one memory reference cycle. Therefore they require either 28 microseconds or 38 microseconds total time.

MEMORY REFERENCE INSTRUCTIONS

Since the PDP-8/S system contains a 4096-word core memory, 12 bits are required to address all locations. To simplify addressing, the core memory is divided into blocks, or pages, of 128 words (200_8 addresses). Pages are numbered 0_8 through 37_8 and each field of 4096-words of core memory uses 32 pages. The seven address bits (bits 5 through 11) of a memory reference instruction can address any location in the page on which the current instruction is located by placing a 1 in bit 4 of the instruction. By placing a 0 in bit 4 of the instruction, any location in page 0 can be addressed directly from any page of core memory. All other core memory locations can be addressed indirectly by placing a 1 in bit 3 and placing a 7-bit effective address in bits 5 through 11 of the instruction to specify the location in the current page or page 0 which contains the full 12-bit absolute address of the operand.

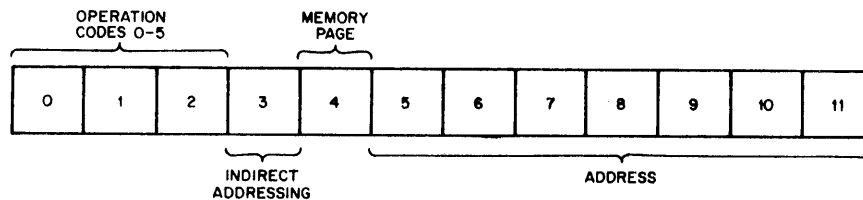


Figure 3. Memory Reference Instruction Bit Assignments

Word format of memory reference instructions is shown in Figure 3 and the instructions performed as follows:

LOGICAL AND (AND Y)

Octal Code: 0

Indicators: AND, FETCH, EXECUTE, END

Execution Time: 36 microseconds with direct addressing, 54 microseconds with indirect addressing, 72 microseconds with auto-indexing

Parity Test: instruction, operand

Operation: The AND operation is performed between the content of memory location Y and the contents of the AC. The result is left in the AC, the original content of the AC is lost, and the content of Y is restored. Corresponding bits of the AC and Y are operated upon independently. This instruction, often called extract or mask, can be considered as a bit-by-bit multiplication. Example:

| <u>Original</u> <u>AC_j</u> | <u>Y_j</u> | <u>Final</u> <u>AC_j</u> |
|--|----------------------|---------------------------------------|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Symbol: $AC_j \wedge Y_i = AC_j$

TWO'S COMPLEMENT ADD (TAD Y)

Octal Code: 1

Indicators: TAD, FETCH, EXECUTE, END

Execution Time: 36 microseconds with direct addressing, 54 microseconds with indirect addressing, 72 microseconds with auto-indexing

Parity Test: instruction, operand

Operation: The contents of memory location Y is added to the content of the AC in two's complement arithmetic. The result of this addition is held in the AC, the original content of the AC is lost, and the content of Y is restored. If there is a carry from ACO, the link is complemented. This feature is useful in multiple precision arithmetic.

Symbol: $ACO-11 + YO-11 = \rightarrow ACO-11$

INCREMENT AND SKIP IF ZERO (ISZ Y)

Octal Code: 2

Indicators: ISZ, FETCH, EXECUTE, END

Execution Time: 54 microseconds with direct addressing, 72 microseconds with indirect addressing, 90 microseconds with auto-indexing

Parity Test: instruction, operand

Operation: The content of memory location Y is incremented by one in two's complement arithmetic. If the resultant content of Y equals zero, the content of the PC is incremented by one and the next instruction is skipped. If the resultant content of Y does not equal zero, the program proceeds to the next instruction. The incremented content of Y is restored to memory. The content of the AC is not affected by this instruction.

Symbol: $Y + 1 \Rightarrow Y$ If resultant $Y_{0-11} = 0$, then $PC + 1 = > PC$

DEPOSIT AND CLEAR AC (DCA Y)

Octal Code: 3

Indicators: DCA, FETCH, EXECUTE, END

Execution Time: 46 microseconds with direct addressing, 64 microseconds with indirect addressing, 82 microseconds with auto-indexing

Parity Test: instruction

Operation: The content of the AC is deposited in core memory at address Y and the AC is cleared. The previous content of memory location Y is lost.

Symbol: $AC \Rightarrow Y$, then $0 \Rightarrow AC$

JUMP TO SUBROUTINE (JMS Y)

Octal Code: 4

Indicators: JMS, FETCH, EXECUTE, END

Execution Time: 46 microseconds with direct addressing, 64 microseconds with indirect addressing, 82 microseconds with auto-indexing.

Parity Test: instruction

Operation: The content of the PC is deposited in core memory location Y and the next instruction is taken from core memory location $Y + 1$. The content of the AC is not affected by this instruction.

Symbol: $PC + 1 \Rightarrow Y$

$Y + 1 \Rightarrow PC$

JUMP TO Y (JMP Y)

Octal Code: 5

Indicators: JMP, FETCH, EXECUTE, END

Execution Time: 28 microseconds with direct addressing, 46 microseconds with indirect addressing, 64 microseconds with auto-indexing.

Parity Test: instruction

Operation: Address Y is set into the PC so that the next instruction is taken from core memory address Y. The original content of the PC is lost. The content of the AC is not affected by this instruction.

Symbol: Y => PC

AUGMENTED INSTRUCTIONS

There are two augmented instructions which do not reference memory. They are the input/output transfer (IOT) which has an operation code of 6 and the operate (OPR) which has an operation code of 7. Bits 3-11 within these instructions function as an extension of the operation code and can be microprogrammed to perform several operations within one instruction. The IOT instruction is a 3 processor cycle instruction requiring a fetch, an execute, and an end cycle. Only one memory reference is required for the IOT instruction, the total time of the instruction therefore is 38 microseconds.

During the execute state, a series of three microprogrammed pulses may be generated starting with bit time 0 of word time X and separated from each other by one microsecond intervals. These pulses are used to initiate external functions and are designated as input/output pulses (IOP 1, 2, 4). There are two classes of operate instructions: Group 2 operate instructions are used for testing the accumulator and require 3 processor cycles and 1 memory reference cycle. These instructions therefore require 38 microseconds. Group 1 operate instructions are used for rotating the accumulator and incrementing the accumulator. These instructions require 2 processor cycles and 1 memory reference cycle and therefore execution time is 28 microseconds.

INPUT OUTPUT TRANSFER INSTRUCTION

Microinstructions of the input-output transfer (IOT) instruction initiate operation of peripheral equipment and effect information transfers between the processor and an I/O device. Specifically, when an operation code of 6 is detected, the IOP generator is enabled to produce IOP 1, IOP 2, and IOP 4 pulses as a function of the content of instruction bits 9 through 11. These pulses occur at 1 microsecond intervals designated as event times 3, 2, and 1 as follows:

| <u>Instruction</u> <u>Bit</u> | <u>IOP</u> <u>Pulse</u> | <u>IOT</u> <u>Pulse</u> | <u>Event</u> <u>Time</u> |
|----------------------------------|----------------------------|----------------------------|-----------------------------|
| 11 | IOP 1 | IOT 1 | 1 |
| 10 | IOP 2 | IOT 2 | 2 |
| 9 | IOP 4 | IOT 4 | 3 |

The IOP pulses are gated in the device selector of the program-selected equipment to produce IOT pulses that enact a data transfer or initiate a control operation. Selection of an equipment is accomplished by bits 3 through 8 of the IOT instruction. These bits form a 6-bit code that enables the device selector in a given device.

The format of the IOT instruction is shown in Figure 4. Operations performed by IOT microinstructions are explained in Section B of this handbook.

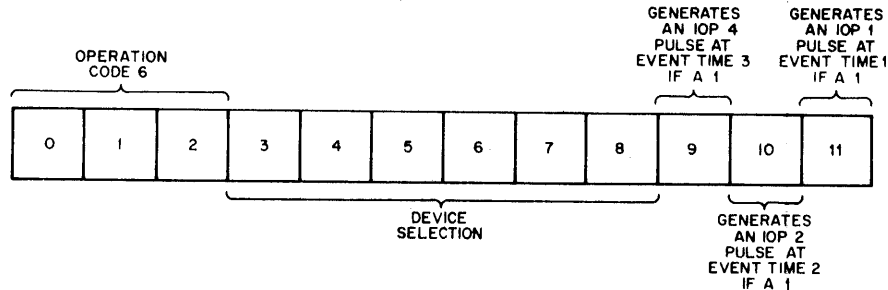


Figure 4 IOT Instruction Bit Assignments

OPERATE INSTRUCTION

The operate instruction consists of two groups of microinstructions. Group 1 (OPR 1) is principally for clear, complement, rotate, and increment operations and is designated by the presence of a 0 in bit 3. Group 2 (OPR 2) is used principally in checking the content of the accumulator and link and continuing to, or skipping, the next instruction based on the check. A 1 in bit 3 designates an OPR 2 microinstruction.

Group 1 operate microinstruction format is shown in Figure 5 and the microinstructions are explained in the succeeding paragraphs. Any logical combination of bits within this group can be combined into one microinstruction. For example, it is possible to assign ones to bits 5, 6, and 11; but it is not logical to assign ones to bits 8 and 9 simultaneously since they specify conflicting operations. The only restriction on combining OPR 1 operations within one instruction, other than logical conflicts, is that a rotate operation (bits 8, 9, or 10) may not be combined with the increment AC operation (bit 11) since they are executed during the same bit times.

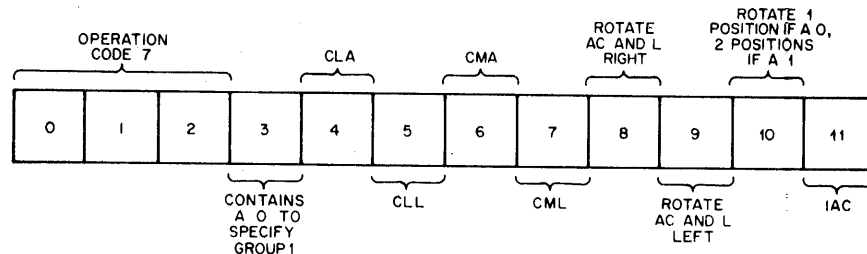


Figure 5 Group 1 Operate Instruction Bit Assignments

NO OPERATION (NOP)

Octal Code: 7000

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 28 microseconds

Parity Test: instruction

Operation: This command causes a 1 instruction delay in the program (28 microseconds) and then the next sequential instruction is initiated. This command is used to add execution time to a program, such as to synchronize subroutine or loop timing with peripheral equipment timing.

Symbol: None

INCREMENT ACCUMULATOR (IAC)

Octal Code: 7001

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 28 microseconds

Parity Test: instruction

Operation: The content of the AC is incremented by one in two's complement arithmetic.

Symbol: $AC + 1 \Rightarrow AC$

ROTATE ACCUMULATOR LEFT (RAL)

Octal Code: 7004

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 28 microseconds

Parity Test: instruction

Operation: The content of the AC is rotated one binary position to the left with the content of the link. The content of bits AC1-11 are shifted to the next greater significant bit, the content of AC0 is shifted into the L, and the content of the L is shifted in AC11.

Symbol: $AC_j \Rightarrow AC_j - 1$

$AC_0 \Rightarrow L$

$L = AC_{10}$

ROTATE TWO LEFT (RTL)

Octal Code: 7006

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 28 microseconds

Parity Test: instruction

Operation: The content of the AC is rotated two binary positions to the left with the content of the link. This instruction is logically equal to two successive RAL operations.

Symbol: $AC_j \Rightarrow AC_j - 2$

$AC1 \Rightarrow L$

$AC0 \Rightarrow AC11$

$L \Rightarrow AC10$

ROTATE ACCUMULATOR RIGHT (RAR)

Octal Code: 7010

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 28 microseconds

Parity Test: instruction

Operation: The content of the AC is rotated one binary position to the right with the content of the link. The content of bits AC0-10 are shifted to the next less significant bit, the content of AC11 is shifted into the L, and the content of the L is shifted into AC0.

Symbol: $AC_j \Rightarrow AC_j + 1$

$AC11 \Rightarrow L$ $L \Rightarrow AC0$

ROTATE TWO RIGHT (RTR)

Octal Code: 7012

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 28 microseconds

Parity Test: instruction

Operation: The content of the AC is rotated two binary positions to the right with the content of the link. This instruction is logically equal to two successive RAR operations.

Symbol: $AC_j \Rightarrow AC_j + 2$ $AC11 = AC0$

$AC10 = L$ $L \Rightarrow AC1$

COMPLEMENT LINK (CML)

Octal Code: 7020

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 28 microseconds

Parity Test: instruction

Operation: The content of the L is complemented.

Symbol: $\overline{L} \Rightarrow L$

COMPLEMENT ACCUMULATOR (CMA)

Octal Code: 7040

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 28 microseconds

Parity Test: instruction

Operation: The content of the AC is set to the one's complement of the current contents of the AC. The content of each bit of the AC is complemented individually.

Symbol: $\overline{AC_j} \Rightarrow AC_j$

COMPLEMENT AND INCREMENT ACCUMULATOR (CIA)

Octal Code: 7041

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 28 microseconds

Parity Test: instruction

Operation: The content of the AC is converted from a binary value to its equivalent two's complement number. This conversion is accomplished by combining the CMA and IAC commands, thus the one's complement of the content of the AC is incremented by one during bit time 0-11.

Symbol: $AC_j \Rightarrow AC_j,$
then $AC + 1 \Rightarrow AC$

CLEAR LINK (CLL)

Octal Code: 7100

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 28 microseconds

Parity Test: instruction

Operation: The content of the L is cleared to contain a 0.

Symbol: 0 => L

SET LINK (STL)

Octal Code: 7120

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 28 microseconds

Parity Test: instruction

Operation: The L is set to contain a binary 1. This instruction is logically equal to combining the CLL and CML commands.

Symbol: 1 => L

CLEAR ACCUMULATOR (CLA)

Octal Code: 7200

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 28 microseconds

Parity Test: instruction

Operation: The content of each bit of the AC is cleared to contain a binary 0.

Symbol: 0 => AC

SET ACCUMULATOR (STA)

Octal Code: 7240

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 28 microseconds

Parity Test: instruction

Operation: Each bit of the AC is set to contain a binary 1. This operation is logically equal to combining the CLA and CMA commands.

Symbol: 1 ⇒ ACj

Group 2 operate microinstruction format is shown in Figure 6 and the primary microinstructions are explained in the following paragraphs. Any logical combination of bits within this group can be composed into one microinstruction. (The instructions constructed by most logical command combinations are listed in Appendix 1.)

If skips are combined in a single instruction the inclusive OR of the conditions determines the skip when bit 8 is a 0; and the AND of the inverse of the conditions determines the skip when bit 8 is a 1. For example, if ones are designated in bits 6 and 7 (SZA and SNL), the next instruction is skipped if either the content of the AC = 0, or the content of L = 1. If ones are contained in bits 5, 7, and 8, the next instruction is skipped if the AC contains a positive number and the L contains a 0.

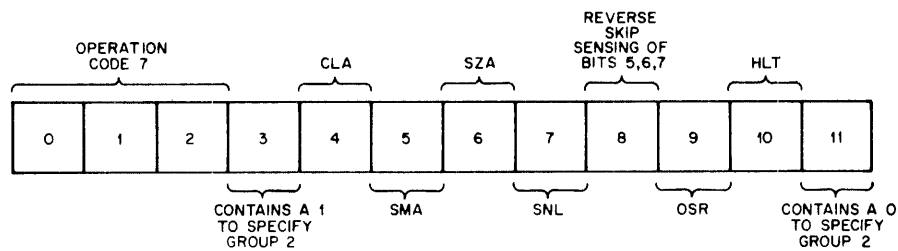


Figure 6 Group 2 Operate Instruction Bit Assignments

HALT (HLT)

Octal Code: 7402

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: Clears the RUN flip-flop at the end of word time E, just after memory request for the next instruction is executed so that the program stops at the conclusion of the current instructions. This command can be combined with others in the OPR 2 group. In such combinations the operations are executed prior to the program stop.

Symbol: 0 => RUN

OR WITH SWITCH REGISTER (OSR)

Octal Code: 7404

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The inclusive OR operation is performed between the content of the AC and the content of the SR. The result is left in the AC, the original content of the AC is lost, and the content of the SR is unaffected by this command. When combined with the CLA command, the OSR performs a transfer of the content of the SR in the AC.

Symbol: $AC_j \vee SR_j = >AC_j$

SKIP, UNCONDITIONAL (SKP)

Octal Code: 7410

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of the PC is incremented by one so that the next sequential instruction is skipped.

Symbol: $PC + 1 = >PC$

SKIP ON NON-ZERO LINK (SNL)

Octal Code: 7420

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of the L is sampled, and if it contains a 1 the content of the PC is incremented by one so that the next sequential instruction is skipped. If the L contains a 0, no operation occurs and the next sequential instruction is initiated.

Symbol: If $L = 1$, then $PC + 1 = >PC$

SKIP ON ZERO LINK (SZL)

Octal Code: 7430

Event Time: 1

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of the L is sampled, and if it contains a 0 the content of the PC is incremented by one so that the next sequential instruction is skipped. If the L contains a 1, no operation occurs and the next sequential instruction is initiated.

SKIP ON ZERO ACCUMULATOR (SZA)

Symbol: If $L = 0$, then $PC + 1 = PC$

Octal code: 7440

Indicators: OPR, FETCH, EXECUTE, END

Execution TIME: 38 microseconds

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of each bit of the AC is sampled, and if each bit contains a 0 the content of the PC is incremented by one so that the next sequential instruction is skipped. If any bit of the AC contains a 1, no operation occurs and the next sequential instruction is initiated.

Symbol: If $ACO - 11 = 0$, then $PC + 1 \Rightarrow PC$

SKIP ON NON-ZERO ACCUMULATOR (SNA)

Octal Code: 7450

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of each bit of the AC is sampled, and if any bit contains a 1 the content of the PC is incremented by one so that the next sequential instruction is skipped. If all bits of the AC contain a 0, no operation occurs and the next sequential instruction is initiated.

Symbol: If $ACO - 11 \neq 0$, then $PC + 1 \Rightarrow PC$

38 microseconds

SKIP ON MINUS ACCUMULATOR (SMA)

Octal Code: 7500

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of the most significant bit of the AC is sampled, and if it contains a 1, indicating the AC contains a negative two's complement number, the content of the PC is incremented by one so that the next sequential instruction is skipped. If the AC contains a positive number no operation occurs and program control advances to the next sequential instruction.

Symbol: If $ACO = 1$, then $PC + 1 \Rightarrow PC$

SKIP ON POSITIVE ACCUMULATOR (SPA)

Octal Code: 7510

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of the most significant bit of the AC is sampled, and if it contains a 0, indicating a positive (or zero) two's complement number, the content of the PC is incremented by one so that the next sequential instruction is skipped. If the AC contains a negative number, no operation occurs and program control advances to the next sequential instruction.

Symbol: If $ACO = 0$, then $PC + 1 \Rightarrow PC$

CLEAR ACCUMULATOR (CLA)

Octal Code: 7600

Indicators: OPR, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: Each bit of the AC is cleared to contain a binary 0.

Symbol: $0 \Rightarrow AC$

CHAPTER 3

MEMORY AND PROCESSOR BASIC PROGRAMMING

MEMORY ADDRESSING

The following terms are used in memory address programming:

| <u>Term</u> | <u>Definition</u> |
|-------------------|---|
| Page | A block of 128 core memory locations (200_8 addresses). |
| Current Page | The page containing the instruction being executed; as determined by bits 0 through 4 of the program counter. |
| Page Address | An 8-bit number contained in bits 4 through 11 of an instruction which designates one of 256 core memory locations. Bit 4 of a page address indicates that the location is in the current page when a 1, or indicates it is in page 0 when a 0. Bits 5 through 11 designate one of the 128 locations in the page determined by bit 4. |
| Absolute Address | A 12-bit number used to address any location in core memory. |
| Effective Address | The address of the operand. When the address of the operand is in the current page or in page 0, the effective address is a page address. Otherwise, the effective address is an absolute address stored in the current page or page 0 and obtained by indirect addressing. |

Organization of the standard core memory or any 4096-word field of extended memory is summarized as follows:

| | |
|--|-------|
| Total locations (decimal) | 4096 |
| Total addresses (octal) | 7777 |
| Number of pages (decimal) | 32 |
| Page designations (octal) | 0-37 |
| Number of locations per page (decimal) | 128 |
| Addresses within a page (octal) | 0-177 |

Four methods of obtaining the effective address are used as specified by combinations of bits 3 and 4.

| Bit 3 | Bit 4 | Effective Address |
|-------|-------|--|
| 0 | 0 | The operand is in page 0 at the address specified by bits 5 through 11. |
| 0 | 1 | The operand is in the current page at the address specified by bits 5 through 11. |
| 1 | 0 | The absolute address of the operand is taken from the content of the location in page 0 designated by bits 5 through 11. |
| 1 | 1 | The absolute address of the operand is taken from the content of the location in the current page designated by bits 5 through 11. |

The following example indicates the use of bits 3 and 4 to address any location in core memory. Suppose it is desired to add the content of locations A, B, C, and D to the content of the accumulator by means of a routine stored in page 2. The instructions in this example indicate the operation code, the content of bit 4, the content of bit 3, and a 7-bit address. This routine would take the following form:

| Page 0 | | Page 1 | | Page 2 | | Remarks |
|----------|---------|----------|---------|----------|---------|--|
| Location | Content | Location | Content | Location | Content | |
| | | | | R TAD 00 | A | DIRECT TO DATA IN PAGE 0 |
| | | | | S TAD 01 | B | DIRECT TO DATA IN SAME PAGE |
| | | | | T TAD 10 | M | INDIRECT TO ADDRESS SPECIFIED IN PAGE 0 |
| | | | | U TAD 11 | N | INDIRECT TO ADDRESS SPECIFIED IN SAME PAGE |
| | | | | . | . | . |
| A | xxxx | C | xxxx | B | xxxx | |
| M | C | D | xxxx | N | D | |

Routines using 128 instructions, or less, can be written in one page using direct addresses for looping and using indirect addresses for data stored in other pages. When planning the location of instructions and data in core memory, remember that the following locations are reserved for special purposes:

| <u>Address</u> | <u>Purpose</u> |
|---|--|
| 0 ₈ | Stores the contents of the program counter following a program interrupt. |
| 1 ₈ | Stores the first instruction to be executed following a program interrupt. |
| 10 ₈ through 17 ₈ | Auto-indexing. |

INDIRECT ADDRESSING

When indirect addressing is specified, the address part (bits 5-11) of a memory reference instruction is interpreted as the address of a location containing not the operand, but containing the address of the operand. Consider the instruction TAD A. Normally, A is interpreted as the address of the location containing the quantity to be added to the content of the AC. Thus, if location 100 contains the number 5432, the instruction TAD 100 causes the quantity 5432 to be added to the content of the AC. Now suppose that location 5432 contains the number 6543. The instruction TAD I 100 (where I signifies indirect addressing) causes the computer to take the number 5432, which is in location 100, as the effective address of the instruction and the number in location 5432 as the operand. Hence, this instruction results in the quantity 6543 being added to the content of the AC.

AUTO-INDEXING

When a location between 10₈ and 17₈ in page 0 of any core memory field is addressed indirectly (by an instruction in which bit 3 is a 1) the content of that location is read, incremented by one, rewritten in the same location, and then taken as the effective address of the instruction. This feature is called auto-indexing. If location 12₈ contains the number 5432 and the instruction DCA I Z 12 is given, the number 5433 is stored in location 12, and the content of the accumulator is deposited in core memory location 5433.

STORING AND LOADING

Data is stored in any core memory location by use of the DCA Y instruction. This instruction clears the AC to simplify loading of the next datum. If the data deposited is required in the AC for the next program operation, the DCA must be followed by a TAD Y for the same address.

All loading of core memory information into the AC is accomplished by means of the TAD Y instruction, preceded by an instruction that clears the AC such as CLA or DCA.

Storing and Loading of information in sequential core memory locations can make excellent use of an auto-indexing register to specify the core memory address.

Program Control

Transfer of program control to any core memory location uses the JMP or JMS instructions. The JMP I (indirect address, 1 in bit 3) is used to transfer program control to any location in core memory which is not in the current page or page 0.

The JMS Y is used to enter a subroutine which starts at location $Y + 1$ in the current page or page 0. The content of the PC + 1 is stored in the specified address Y, and address $Y + 1$ is transferred into the PC. To exit a subroutine the last instruction is a JMP I Y, which returns program control to the location stored in Y.

Indexing Operations

External events can be counted by the program and the number can be stored in core memory. The core memory location used to store the event count can be initialized (cleared) by a CLA command followed by a DCA instruction. Each time the event occurs, the event count can be advanced by a sequence of commands such as CLA, TAD, IAC, and DCA.

The ISZ instruction is used to count repetitive program operations or external events without disturbing the content of the accumulator. Counting a specified number of operations is performed by storing a two's complement negative number equal to the number of iterations to be counted. Each time the operation is performed, the ISZ instruction is used to increment the content of this stored number and check the result. When the stored number becomes zero, the specified number of operations have occurred and the program skips out of the loop and back to the main sequence.

This instruction is also used for other routines in which the content of a memory location is incremented without disturbing the content of the accumulator, such as storing information from an I/O device in sequential memory locations or using core memory locations to count I/O device events.

LOGIC OPERATIONS

The PDP-8/S instruction list includes the logic instruction, AND Y. From this instruction short routines can be written to perform the inclusive OR and exclusive OR operations.

Logical AND

The logic AND operation between the content of the accumulator and the content of a core memory location Y is performed directly by means of the AND Y instruction. The result remains in the AC, the original content of the AC is lost, and the content of Y is unaffected.

Inclusive OR

Assuming value A is in the AC and value B is stored in a known core memory address, the following sequence performs the inclusive OR. The sequence is stated as a utility subroutine called IOR.

```

/CALLING SEQUENCE                                     JMS IOR
/                                                       (ADDRESS OF B)
/                                                       (RETURN)
/ENTER WITH ARGUMENT IN AC; EXIT WITH LOGICAL RESULT IN AC
      IOR,      0
              DCA TEM1
              TAD I IOR
              DCA TEM2
              TAD TEM1
              CMA
              AND I TEM2
              TAD TEM1
              ISZ IOR
              JMP I IOR
      TEM1,    0
      TEM2,    0
```

Exclusive OR

The exclusive OR operation for two numbers, A and B, can be performed by a subroutine called by the mnemonic code XOR. In the following general purpose XOR subroutine, the value A is assumed to be in the AC, and the address of the value B is assumed to be stored in a known memory location.

```

/CALLING SEQUENCE                                     JMS XOR
/                                                       (ADDRESS OF B)
/                                                       (RETURN)
/ENTER WITH ARGUMENT IN AC; EXIT WITH LOGICAL RESULT IN AC
      XOR,      0
              DCA TEM1
              TAD I XOR
              DCA TEM2
              TAD TEM1
              AND I TEM2
              CMA IAC
              CLL RAL
              TAD TEM1
              TAD I TEM2
              ISZ XOR
              JMP I XOR
      TEM1,    0
      TEM2,    0
```


An XOR subroutine can be written using fewer core memory locations by making use of the IOR subroutine; however, such a subroutine takes more time to execute. A faster XOR subroutine can be written by storing the value B in the second instruction of the calling sequence instead of the address of B; however, the resulting subroutine is not as utilitarian as the routine given here.

ARITHMETIC OPERATIONS

One arithmetic instruction is included in the PDP-8/S order code the two's complement add: TAD Y. Using this instruction, routines can easily be written to perform addition, subtraction, multiplication, and division in two's complement arithmetic.

Two's Complement Arithmetic

In two's complement arithmetic addition, subtraction, multiplication, and division of binary numbers is performed in accordance with the common rules of binary arithmetic PDP-8/S, as in other machines utilizing complementation techniques, negative numbers are represented as the complement of positive numbers, and subtraction is achieved by complement addition. Representation of negative values in one's complement arithmetic is slightly different from that in two's complement arithmetic.

The one's complement of a number is the complement of the absolute positive value; that is, all ones are replaced by zeros and all zeros are replaced by ones. The two's complement of a number is equal to the one's complement of the positive value plus one.

In one's complement arithmetic a carry from the sign bit (most significant bit) is added to the least significant bit in an end-around carry. In two's complement arithmetic a carry from the sign bit complements the link (a carry would set the link to 1 if it were properly cleared before the operation), and there is no end-around carry.

A one's complement representation of a negative number is always one less than the two's complement representation of the same number. Differences between one's and two's complement representations are indicated in the following list:

| <u>Number</u> | <u>1's Complement</u> | <u>2's Complement</u> |
|---------------|-----------------------|-----------------------|
| +5 | 00000000101 | 00000000101 |
| +4 | 00000000100 | 00000000100 |
| +3 | 00000000011 | 00000000011 |
| +2 | 00000000010 | 00000000010 |
| +1 | 00000000001 | 00000000001 |
| +0 | 00000000000 | 00000000000 |
| -0 | 11111111111 | Nonexistent |
| -1 | 11111111110 | 11111111111 |
| -2 | 11111111101 | 11111111110 |
| -3 | 11111111100 | 11111111101 |
| -4 | 11111111011 | 11111111100 |
| -5 | 11111111010 | 11111111011 |

Note that in two's complement there is only one representation for the number which has the value zero, while in one's complement there are two representations. Note also that complementation does not interfere with sign notation in either one's complement or two's complement arithmetic; bit 0 remains a 0 for positive numbers and a 1 for negative numbers.

To form the two's complement of any number in the PDP-8/S the one's complement is formed, and the result is incremented by one. This is accomplished by the instruction CMA combined with an IAC instruction. Since both of these instructions are functions of the OPR 1 instruction and the actions occur at different event times, they can be combined to form the instruction CIA, Complement and Increment AC.

ADDITION

The addition of a number contained in a core memory location and the number contained in the accumulator is performed directly by using the TAD Y instruction, assuming that the binary point is in the same position and that both numbers are properly represented in two's complement arithmetic. Addition can be performed without regard for the sign of either the augend or the addend. Overflow is possible, in which case the result will have an incorrect sign, although the 11 least significant bits will be correct. Following the addition a test for overflow can be made by using the SZL command.

SUBTRACTION

Subtraction is performed by complementing the subtrahend and adding the minuend. As in addition, if both numbers are represented by their two's complement, subtraction can be performed without regard for the sign of either number. Assuming that both numbers are stored in core memory, a routine to find the value of A-B follows:

```
CLA
TAD B          /LOAD SUBTRAHEND INTO AC
CIA           /COMPLEMENT AND INCREMENT B
TAD A          /AC = A - B
```

CHAPTER 4

PROGRAM INTERRUPT

The program interrupt feature allows certain external conditions to interrupt the computer program. It is used to speed the information processing of input-output devices or to allow certain alarms to halt the program in progress and initiate another routine. When a program interrupt request is made the computer completes execution of the instruction in progress before acknowledging the request and entering the interrupt mode. A program interrupt is similar to a JMS to location 0; that is, the content of the program counter is stored in location 0, and the program resumes operation in location 1. The interrupt program commencing in location 1 is responsible for identifying the signal causing the interruption, for removing the interrupt condition, and for returning to the original program. Exit from the interrupt program, back to the original program, can be accomplished by a JMP I Z 0 instruction.

INSTRUCTIONS

The two instructions associated with the program interrupt synchronization element are IOT microinstructions that do not use the IOP generator. These instructions are:

INTERRUPT TURN ON (ION)

Octal Code: 6001

Event Time: Not applicable

Indicators: IOT, FETCH, EXECUTE, END, ION

Execution Time: 38 microseconds

Parity Test: instructions

Operation: This command enables the computer to respond to a program interrupt request. If the interrupt is disabled when this instruction is given, the computer executes the next instruction, then enables the interrupt. The additional instruction allows exit from the interrupt subroutines before allowing another interrupt to occur. This instruction has no affect upon the condition of the interrupt circuits if it is given when the interrupt is enabled.

Symbol: 1 = > INT. ENABLE

INTERRUPT TURN OFF (IOF)

Octal Code: 6002

Event Time: Not applicable

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instructions

Operation: This command disables the program interrupt synchronization element to prevent interruption of the current program.

Symbol: 0 => INT. ENABLE, INT. DELAY

PROGRAMMING

When an interrupt request is acknowledged, the interrupt is automatically disabled by the program interrupt synchronization circuits (not by instructions). The break state is entered after completion of the current instruction, followed by the end state when the next instruction is taken from core memory location 1. A total of 36 microseconds is therefore needed to complete the break. Usually the instruction stored in location 1 is a JMP, which transfers program control to a subroutine which services the interrupt. At some time during this subroutine an ION instruction must be given. The ION can be given at the end of the subroutine to allow other interrupts to be serviced after program control is transferred back to the original program. In this application, the ION instruction immediately precedes the last instruction in the routine. A delay of one instruction (regardless of the execution time of the instruction in progress at the time the request is made) is inherent in the ION instruction to allow transfer of program control back to the original program before enabling the interrupt. Usually exit from the subroutine is accomplished by a JMP I O instruction.

The ION command can be given during the subroutine as soon as it has determined the I/O device causing the interrupt. This latter method allows the subroutine which is handling a low priority interrupt to be interrupted, possibly by a high priority device. Programming of an interrupt subroutine which checks for priority and allows itself to be interrupted, must make provisions to relocate the content of the program counter stored in location 0; so that if interrupted, the content of the PC during the subroutine is stored in location 0, and the content of the PC during the original program is not lost.

CHAPTER 5

MEMORY PARITY

Checking of each word written in and read from memory is provided by the parity checking hardware. Thus each word stored in memory is a 13 bit word consisting of 12 data bits and one parity bit. The parity bit is made either a 0 or a 1 so that an even number of binary ones is always present in every word in memory. Every word which enters the memory buffer from the processor has a new parity bit generated, for writing in the memory. Every word entering the memory buffer from the memory carries with it a parity bit. Every word shifted from the memory buffer into the processor has its parity tested and checked against the parity bit which was placed in the processor by the memory. The memory parity error flag is connected to the interrupt bus and will generate an interrupt if the interrupt is enabled and a parity error occurs.

INSTRUCTIONS

SKIP ON NO MEMORY PARITY ERROR (SMP)

Octal Code: 6101

Event Time: 3

Indicator: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instructions

Operation: The memory parity error flag is sensed and if it contains a 0 (signifying no error has been detected) the PC is incremented so that the next successive instruction is skipped.

Symbol: If Memory Parity Error Flag = 0, then $PC + 1 = > PC$

CLEAR MEMORY PARITY ERROR FLAG (CMP)

Octal Code: 6104

Event Time: 2

Indicator: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instructions

Operation: The memory parity error flag is cleared.

Symbol: 0 = Memory Parity Error Flag

PROGRAMMING

Both instructions for this option are used in the program interrupt subroutine and in diagnostic maintenance programs. The SMP command is used as a programmed check for memory parity error. In the program interrupt subroutine this command can be followed by a jump to a portion of the routine that services memory parity as described previously. The CMP command is used to initialize memory parity in preparation for normal programmed operation of the computer.

CHAPTER 6

BASIC IOT PROGRAMMING

Two basic modes of programming can be used to transfer information or control signals between input/output devices and the PDP-8/S. These modes are programmed data transfers and program interrupt. To understand the use which can be made of each of these control modes assume that the PDP-8/S is connected as the primary control element in a system which contains several furnaces or kilns, continuous-belt conveyors which transport products through the furnaces and a visual monitor panel. Assume that each furnace or zone of an oven contains a controller which has a digital temperature readout, and overtemperature and undertemperature alarms which can be set by a digital readin. Next, assume that the conveyor for each oven is automatically loaded with blocks of products spaced at given distances on the conveyor and that the conveyor has a normal speed, a high emergency speed, and a stop control. The following explanation of the use of each of the two control modes in programming this hypothetical system can easily be translated into examples of processing or other control system programs.

PROGRAMMED DATA TRANSFERS

All peripheral equipment transfers information to or from the computer by programmed instructions. This means of communication can be used as the sole method of transferring information or can be used to initialize equipment using the program interrupt or data break facilities. This mode of operation utilizes the IOT instruction which is divided into three parts. Bits 0, 1, and 2 contain an operation code of 6 to specify the IOT microinstruction. Bits 3 through 8 serve as a device selection code which is transmitted to all peripheral equipment and which activates only the equipment designated by a specific code number contained within these bits. Bits 9, 10, and 11 control the IOP generator within the processor and enable or disable generation of IOP1, IOP2, and IOP4 pulses during each IOT instruction. A device selector within each peripheral equipment monitors the device selection lines and enables pulse amplifiers when its assigned select code has been detected within bits 3 through 8 of an IOT instruction. When enabled in this manner the pulse amplifiers produce positive or negative IOT pulses when triggered by an associated IOP pulse. The IOT pulses, in turn, perform data transfers to or from the computer or perform control functions within the peripheral equipment.

Each peripheral equipment can contain one or more device selector. A device selector can consist of a Type W103 Device Selector FLIP CHIP module, or can be constructed of three FLIP-CHIP modules such as the Type R603 Pulse Amplifier, R111 Diode Gate, and R002 Diode Cluster. Regardless of its circuit components, a device selector consists of a 6-input negative diode NAND gate which is enabled only when the select code of the specified device is contained in the instruction. The output from this NAND gate enables gating circuits at the input of each of three pulse amplifiers which are triggered by the IOP1, IOP2, or IOP4 pulse.

SENSE FOR DEVICE READY

In preparation for a normal data transfer the computer program normally checks the ready status of the transmitting or receiving device by means of a skip instruction. This skip instruction can skip on either the ready or not ready status of the device, depending upon the internal operations of that device. In our imaginary PDP-8/S control system, assume that the control for each furnace contains an "up to temperature flag" and an associated

instruction which provides a Skip pulse to the program counter control element when the operating temperature range has been attained. Therefore the programming can contain this skip on up to temperature instruction followed by a JMP instruction which transfers program control back in a loop so that it repeatedly checks the up to temperature flag until the acceptable temperature range has been attained. Then the next step in the program proceeds with an operation, such as initiate loading of the conveyor and movement of the conveyor at the normal speed. In some instances the program will proceed to assemble data so that it can be transferred into or out of the computer

ASSEMBLE DATA

The ready control described under the previous heading can indicate that a device is operating in a known control mode or that data is ready for transmission. In our example the former condition is true so that data assembly follows the sense for ready operation. To assemble data the program issues commands in one or more IOT instructions to clear and load a buffer with data to be transferred from the accumulator, or to clear and load the accumulator for data to be transferred to a peripheral equipment buffer. In the imaginary control system described earlier, data assembly could consist of an instruction which generates an IOT1 pulse to clear an information buffer, generates an IOT2 pulse to load a temperature setting into the six least significant bits of this buffer and loads the status of the undertemperature and overtemperature flags into the two most significant bits of this buffer.

EFFECT A TRANSFER

Actual data transfer between peripheral equipment and the PDP-8/S accumulator is performed by an IOT instruction. Usually a transfer into the PDP-8/S is performed by an instruction in which an IOT1 pulse is generated to clear the accumulator and a later IOT pulse is generated to strobe the content of an external buffer into the accumulator. In transfers from the accumulator to an external buffer the static accumulator data lines are used to condition gates at the input of a static buffer, then an IOT pulse is generated to actuate the gates and transfer the static conditions into the buffer. In the case of our hypothetical process control system an IOT instruction can be developed to clear the accumulator, then read the content of the data buffer into the accumulator.

The entire sequence of operations described under Programmed Data Transfers is summarized in the following program example, using legitimate memory reference and operate instructions, and using artificial IOT instructions.

| | | | |
|---|---|------------|---|
| Initialize | } | CLA | |
| | | TAD | /LOAD OVERTEMPERATURE VALUE INTO AC |
| | | LOT | /CLEAR AND LOAD OVERTEMPERATURE SET /POINT |
| | } | CLA | |
| | | TAD | /LOAD UNDERTEMPERATURE VALUE INTO AC |
| | | LUT | /CLEAR AND LOAD UNDERTEMPERATURE SET /POINT |
| Sense for device ready (one or more devices) | } | TON | /TURN ON FURNACE |
| | | SUT | /SKIP ON DEVICE #1 UP TO TEMP. |
| | | JMP,-1 | /LOOP IF NOT UP TO TEMP |
| Assemble Data Effect Transfer | } | IPC | /INITIATE PRODUCT TRANSPORT TO CONVEYOR /AND CONVEYOR OPERATION AT NORMAL /SPEED |
| | | LDB | /CLEAR AND LOAD DEVICE BUFFER WITH DATA /AND STATUS |
| | | RDB | /CLEAR AC AND READ DATA BUFFER INTO AC |
| Process Data | } | RAL | /ROTATE OVERTEMP. CONTROL STATUS FROM /ACO INTO L |
| | | SNL OR SZL | /SENSE OVERTEMP. CONTROL |
| | | JMS | /JUMP TO OVERTEMP. SUBROUTINE WHICH /TURNS OFF FURNACE, STOPS PRODUCT /LOADING INTO CONVEYOR, ADVANCES /CONVEYOR AT EMERGENCY SPEED TO /REMOVE PRODUCTS FROM FURNACE, THEN /STOPS CONVEYOR |
| | | RAL | /ROTATE UNDERTEMP. CONTROL STATUS /INTO L |
| | | SNL OR SZL | /SENSE UNDERTEMP. CONTROL |
| | | JMS | /JUMP TO SUBROUTINE WHICH STOPS /CONVEYOR MOTION AND JUMPS BACK TO /THE BEGINNING OF THE MAIN ROUTINE TO /WAIT FOR UP TO TEMP. |
| | | RTR | /RELOCATE DATA |
| | | DCA | /STORE TEMPERATURE DATA |

PROGRAM INTERRUPT

Urgent requirements for programmed data transfer or programmed control functions by peripheral equipment can be satisfied through use of the program interrupt facility. This facility allows an external device to cause the main computer program to be interrupted and a subroutine to be initiated to service the interrupting device. Use of this facility simplifies basic programming by eliminating the need for checking alarm conditions and allows the alarm conditions themselves to activate corrective operations, rather than waiting for cyclic checking by the main routine.

When the program interrupt feature is used address 0001 is automatically specified as the first address of a subroutine that checks and services the interrupt condition. Usually the instruction stored in this address is a jump to a location where the subroutine really begins. As designated in Chapter 4 of Section A of this handbook the program interrupt subroutine must locate the device causing the interruption, take some corrective action, restore or enable the program interrupt synchronization element of the computer by execution of an ION instruction, and return program control to the main program at

the point at which the interrupt occurred. If only one device is connected to the program interrupt facility no checking is required to locate the interrupting device. However, if many devices are connected to the program interrupt bus (as normally is the case) the interrupt subroutine must perform repeated skip instructions to test the condition of the various devices. This testing should be accomplished by a program-established priority system so that the devices which need servicing in the least amount of time or which require servicing most frequently are checked first, depending upon the application.

In our hypothetical process control system if the overtemperature alarms for each furnace are connected to the program interrupt bus the interrupt subroutine can skip on the condition of the overtemperature flag to a portion of the routine which de-energizes the appropriate furnace and performs any required operations in the control of the conveyor for that oven (such as inhibiting additional loading of the conveyor, removing products from the oven by high speed advance of the conveyor, or by initiating other shut down procedures). The sequence of testing for the furnace cause the temperature alarm can proceed from the first furnace to the last furnace if it is most important to prevent unfired products from entering the defective oven, can be performed from the last to the first furnace if over firing cannot occur, or can be performed in the sequence determined by the various furnace temperatures.

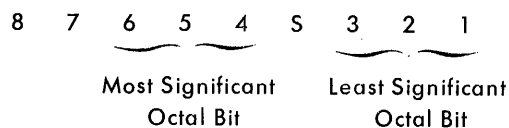
CHAPTER 7

TELETYPE AND CONTROL

TELETYPE MODEL 33 ASR

The standard Teletype Model 33 ASR (automatic send-receive) can be used to type in or print out information at a rate of up to ten characters per second, or to read in or punch out perforated paper tape at a ten characters per second rate. Signals transferred between the 33 ASR and the control logic are standard serial, 11 unit code Teletype signals. The signals consist of marks and spaces which correspond to idle and bias current in the Teletype and to zeros and ones in the control and computer. The start mark and subsequent eight character bits are one unit of time duration and are followed by the stop mark which is two units.

The 8-bit code used by the Model 33 ASR Teletype unit is the American Standard Code for Information Interchange (ASCII) modified. To convert the ASCII code to Teletype code add 200 octal ($ASCII + 200_8 = \text{Teletype}$). This code is read in the reverse of the normal octal form used in the PDP-8/S since bits are numbered from right to left, from 1 through 8, with bit 1 having the least significance. Therefore perforated tape is read:



The Model 33 ASR set can generate all assigned codes except 340 through 374 and 376. Generally codes 207, 212, 215, 240 through 337, and 377 are sufficient for Teletype operation. The Model 33 ASR set can detect all characters, but does not interpret all of the codes that it can generate as commands. The standard number of characters printed per line is 72. The sequence for proceeding to the next line is a carriage return followed by a line feed (as opposed to a line feed followed by a carriage return).

Appendix 2 lists the character code for the Teletype. Punched tape format is as follows:

| | Tape Channel | | |
|-------------|--------------|-----|-------|
| | 87 | 654 | S 321 |
| Binary Code | | | |
| (Punch = 1) | 10 | 110 | 100 |
| Octal Code | 2 | 6 | 4 |

TELETYPE CONTROL

Serial information read or written by the Teletype unit is assembled or disassembled by the control for parallel transfer to the accumulator of the processor. The control also provides the program flags which cause a program interrupt or an instruction skip based upon the availability of the Teletype and the processor as a function of the program.

In all programmed operation, the Teletype unit and control are considered as a Teletype in (TTI) as a source of input intelligence from the keyboard or the perforated-tape reader and is considered a Teletype out (TTO) for computer output information to be printed and/or punched on tape. Therefore, two device selectors are used; the select code of 03 initiates operations associated with the keyboard/reader, and the device selector, assigned the select code of 04, performs operations associated with the teleprinter/punch. Parallel input and output functions are performed by corresponding IOT pulses produced by the two device selectors. Pulses produced by IOP1 pulse trigger skip gates; pulses produced by the IOP2 pulse clear the control flags and/or the accumulator; and pulses produced by the IOP4 pulse initiate data transfers to or from the control.

KEYBOARD/READER

The keyboard and tape reader control contains an 8-bit buffer (TTI) which assembles and holds the code for the last character struck on the keyboard or read from the tape. Teletype characters from the keyboard/reader are received serially by the 8-bit shift register TTI. The character code of a Teletype character is loaded into the TTI so that spaces correspond with binary zeros and marks correspond to binary ones. Upon program command the content of the TTI is transferred in parallel to the accumulator. When a Teletype character starts to enter the TTI the control de-energizes a relay in the Teletype unit to release the tape feed latch. When released, the latch mechanism stops tape motion only when a complete character has been sensed, and before sensing of the next character is started. A keyboard flag is set to one, and causes a program interrupt when an 8-bit computer character has been assembled in the TTI from a Teletype character. The program senses the condition of this flag with a KSF microinstruction and issues a KRB microinstruction which clears the AC, clears the keyboard flag, transfers the content of the TTI into the AC, and enables advance of the tape feed mechanism. Instructions for use in supplying data to the computer from the Teletype are:

SKIP ON KEYBOARD FLAG (KSF)

Octal Code: 6031

Event Time: 1

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The keyboard flag is sensed, and if it contains a binary 1 the content of the PC is incremented by one so that the next sequential instruction is skipped.

Symbol: If Keyboard Flag = 1, then $PC + 1 = > PC$

CLEAR KEYBOARD FLAG (KCC)

Octal Code: 6032

Event Time: 2

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: Both the AC and the keyboard flag are cleared in preparation for transferring a Teletype character into the AC.

Symbol: 0 = > AC

0 = > Keyboard Flag

READ KEYBOARD BUFFER STATIC (KRS)

Octal Code: 6034

Event Time: 3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of the TTI is transferred into bits 4 through 11 of the AC. This is a static command in that neither the AC nor the keyboard flag is cleared.

Symbol: TTI V AC 4-11 = > AC 4-11

READ KEYBOARD BUFFER DYNAMIC (KRB)

Octal Code: 6036

Event Time: 2.3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The AC and the keyboard flag are both cleared, then the content of the TTI is transferred into bits 4 through 11 of the AC.

Symbol: 0 = >AC, Keyboard Flag
TTI V AC 4-11 = >AC 4-11

A program sequence loop to read input information into the computer from the Teletype keyboard or tape reader can be written as follows:

```
LOOK,      KSF      /SKIP WHEN TTI IS FULL
            JMP LOOK
            KRB      /READ TTI INTO AC
```

TELEPRINTER/PUNCH

Eight-bit computer characters from the accumulator are loaded in parallel into the 8-bit flip-flop shift register TTO for transmission to the Teletype unit. The control generates the start space, then shifts the eight character bits into the printer selector magnets of the Teletype unit, and then produces the stop mark. This transfer of information from the TTO into the Teletype unit is accomplished in a serial manner at the normal Teletype rate. A teleprinter flag in the teleprinter control is set when the last bit of the Teletype code has been sent to the teleprinter/punch, indicating that the TTO is ready to receive a new character from the AC. The flag is connected to both the program interrupt synchronization element and the PC control (instruction skip) element. Upon detecting the set (binary one) condition of the flag by means of the TSF microinstruction the program issues a TLS microinstruction which clears the flag and loads a new computer character into the TTO.

The instruction list for printing or punching is:

SKIP ON TELEPRINTER FLAG (TSF)

Octal Code: 6041

Event Time: 1

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The teleprinter flag is sensed, and if it contains a binary 1 the content of the PC is incremented by one so that the next sequential instruction is skipped.

Symbol: If Teleprinter Flag = 1, then $PC + 1 = >PC$

CLEAR TELEPRINTER FLAG (TCF)

Octal Code: 6042

Event Time: 2

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The teleprinter flag is cleared to 0.

Symbol: 0 = >Teleprinter Flag

LOAD TELEPRINTER AND PRINT (TPC)

Octal Code: 6044

Event Time: 3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The TTO is loaded from the content of bits 4 through 11 of the AC; then the Teletype character just loaded is selected, and punched and/or printed.

Symbol: AC 4-11 = > TTO

LOAD TELEPRINTER SEQUENCE (TLS)

Octal Code: 6046

Event Time: 2, 3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The teleprinter flag is cleared; then a Teletype character code is transferred from the content of AC 4-11 into the TTO, the character is selected and punched and/or printed.

Symbol: 0 = > Teleprinter Flag
AC 4-11 = >TTO

A program sequence loop to print and/or punch a character when the TTO is free can be written as follows:

| | | |
|-------|----------|---------------------------|
| FREE, | TSF | /SKIP WHEN FREE |
| | JMP FREE | |
| | TLS | /LOAD TTO, PRINT OR PUNCH |

TELETYPE SYSTEM TYPE LT08

The Teletype facility of the basic computer can be expanded to accommodate several Model 33 or Model 35 Automatic Send Receive or Keyboard Send Receive units by addition of the Type LT08 option. Each Teletype line added to the PDP-8/S system contains logic elements that are functionally identical to those of the basic Teletype control. Therefore, instructions and programming for each line of the LT08 equipment are similar to those described previously for the basic Teletype unit. The following device select codes have been assigned for five lines of LT08 equipment:

| <u>Line</u> <u>Unit</u> | <u>Select</u> <u>Codes</u> |
|----------------------------|-------------------------------|
| 1 | 40 and 41 |
| 2 | 42 and 43 |
| 3 | 44 and 45 |
| 4 | 46 and 47 |
| 5 | 11 and 12 |

Instruction mnemonics for Teletype equipment in the LT08 system are not recognized by the program assembler (PAL III) and must be defined by the programmer. Mnemonic codes can be defined by the mnemonic code of the comparable basic Teletype microinstruction, suffixed with "LT" and the line number. For example, the following instructions can be defined for line 3:

| <u>Mnemonic</u> | <u>Octal</u> | <u>Operation</u> |
|-----------------|--------------|---|
| TSFLT3 | 6441 | Skip if teleprinter 3 flag is a 1. |
| TCPLT3 | 6442 | Clear teleprinter 3 flag. |
| TPCLT3 | 6444 | Load teleprinter 3 buffer (TT03) from the content of AC4-11 and print and/or punch the character. |
| TLSLT3 | 6446 | Load TT03 from the content of AC4-11, clear teleprinter 3 flag, and print and/or punch the character. |
| KSFLT3 | 6451 | Skip if keyboard 3 flag is a 1. |
| KCCLT3 | 6452 | Clear AC and clear keyboard 3 flag. |
| KRSLT3 | 6454 | Read keyboard 3 buffer (TT13) static. The content of TT13 is loaded into AC4-11 by an OR transfer. |
| KRBLT3 | 6456 | Clear the AC, clear keyboard 3 flag, and read the content of TT13 into AC4-11. |

CHAPTER 8

HIGH SPEED PERFORATED TAPE READER AND CONTROL TYPE PC02

This device senses 8-hole perforated paper or Mylar tape photoelectrically at 300 characters per second. The reader control requests reader movement, transfers data from the reader into the reader buffer (RB), and signals the computer when incoming data is present. Reader tape movement is started by a reader control request to simultaneously release the brake and engage the clutch. The 8-bit reader buffer sets the reader flag to 1 when it has been filled from the reader and transfers data into bits 4 through 11 of the accumulator under program control. The reader flag is connected to the computer program interrupt and instruction skip facilities, and is cleared by IOT pulses. Tape format is as described for the Teletype unit. Computer instructions for the reader are:

SKIP ON READER FLAG (RSF)

Octal Code: 6011

Event Time: 1

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The reader flag is sensed, and if it contains a binary 1 the content of the PC is incremented by one so that the next sequential instruction is skipped.

Symbol: If Reader Flag = 1, then $PC + 1 = > PC$

READ READER BUFFER (RRB)

Octal Code: 6012

Event Time: 2

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of the reader buffer is transferred into bits 4 through 11 of the AC and the reader flag is cleared. This command does not clear the AC.

Symbol: $RB \vee AC 4-11 = > AC 4-11$

0 = $>$ Reader Flag

READER FETCH CHARACTER (RFC)

Octal Code: 6014

Event Time: 3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The reader flag and the reader buffer are both cleared, one character is loaded into the reader buffer from tape, and the reader flag is set when this operation is completed.

Symbol: 0 = >Reader Flag, RB

Tape Data = >RB

1 = >Reader Flag when done

A program sequence loop to read a character from perforated tape can be written as follows:

| | | |
|-------|----------|----------------------------|
| | RFC | /FETCH CHARACTER FROM TAPE |
| LOOK, | RSF | /SKIP WHEN RB FULL |
| | JMP LOOK | |
| | CLA | |
| | RRB | /LOAD AC FROM RB |

CHAPTER 9

HIGH SPEED TAPE PUNCH CONTROL TYPE PC03

This option consists of a Royal McBee paper tape punch that perforates 8-hole tape at a rate of 50 characters per second. Information to be punched on a line of tape is loaded in an 8-bit punch buffer (PB) from AC bits 4 through 11. The punch flag becomes a 1 at the completion of punching action, signaling that new information may be transferred into the punch buffer, and punching initiated. The punch flag is connected to the computer program interrupt and instruction skip facility. Tape format is as described in Chapter 2. The punch instructions are:

SKIP ON PUNCH FLAG (PSF)

Octal Code: 6021

Event Time: 1

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The punch flag is sensed, and if it contains a binary 1 the content of the PC is incremented by one so that the next sequential instruction is skipped.

Symbol: If Punch Flag = 1, then $PC + 1 = > PC$

CLEAR PUNCH FLAG (PCF)

Octal Code: 6022

Event Time: 2

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: Both the punch flag and the punch buffer are cleared in preparation for receiving a new character from the computer.

Symbol: $0 = > \text{Punch Flag, PB}$

LOAD PUNCH BUFFER AND PUNCH CHARACTER (PPC)

Octal Code: 6024

Event Time: 3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: An 8-bit character is transferred from bits 4 through 11 of the AC into the punch buffer and then this character is punched. This command does not clear the punch flag or the punch buffer.

Symbol: AC4-11 V PB = > PB

LOAD PUNCH BUFFER SEQUENCE (PLS)

Octal Code: 6026

Event Time: 2, 3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The punch flag and punch buffer are both cleared the contents of bits 4 through 11 of the AC is transferred into the punch buffer, the character in the PB is punched in tape, and the punch flag is set when the operation is completed.

Symbol: 0 =>Punch Flag, PB
AC4-11 =>PB
1 =>Punch Flag when done

A program sequence loop to punch a character when the punch buffer is "free" can be written as follows:

| | | |
|-------|----------|--|
| FREE, | PSF | /SKIP WHEN FREE |
| | JMP FREE | |
| | PLS | /LOAD PB FROM AC AND PUNCH /CHARACTER |

CHAPTER 10

ANALOG-TO DIGITAL CONVERTER TYPE 138E AND MULTIPLEXER CONTROL TYPE 139E

The Type 138E/139E General-purpose Analog-to-Digital Converter and Multiplexer Control combines a versatile, multipurpose converter with a multiplexer to provide a fast, automatic, multichannel scanning and conversion capability. It is intended for use in systems in which computers sample and process analog data from sensors or other external signal sources at high rates. For example, analog data on each of 64 channels can be accepted and converted into 12-bit digital numbers 415 times per second. Switching point accuracy in this instance is 99.975 per cent, with an additional quantization error of half the least significant bit (LSB). If less resolution and accuracy is required, all 64 channels can be scanned and the analog signals on them converted into 6-bit digital numbers 1,360 times each second.** Switching point accuracy in this case is 99.2 per cent, again with the the additional quantization error of half the digital value of the LSB.

The Type 139 Multiplexer Control can include from 1 to 32 series A100 Multiplexer modules determined by the user. Each module addresser one of two channels for maximum of 64 channels per Type 139E control. In the Individual Address mode, the Type 139 routes the data from any selected channel to the Type 138E converter input. In the Sequential Address mode, the multiplexer advances its channel address by one each time it receives an increment command, returning to channel zero after scanning the last channel. Sequenced operations can be short-cycled when the number of channels in use is less than the maximum available.*

* Conversion rate = $[(35+2.5)(10^{-6})(64)]^{-1} = 415$ cycles/sec

** Conversion rate = $[(9+2.5)(10^{-6})(64)]^{-1} = 1360$ cycles/sec

TABLE 1 ANALOG-TO-DIGITAL CONVERTER TYPE 138E CHARACTERISTICS

| Word Length (in bits) | Switching Point Error*** (in percent) | Total Conversion Time (in microseconds) | Conversion Rate (in kc) |
|--------------------------|---|--|-------------------------------|
| 6 | ±1.6 | 9.0 | 110.0 |
| 7 | ±0.8 | 10.5 | 95.0 |
| 8 | ±0.4 | 12.0 | 83.0 |
| 9 | ±0.2 | 13.5 | 74.0 |
| 10 | ±0.1 | 17.0 | 58.5 |
| 11 | ±0.05 | 25.0 | 40.0 |
| 12 | ±0.025 | 35.0 | 28.5 |

*** $\pm 1/2$ LSB for quantizing error.

The Type 138E is a successive approximation converter that measures a 0 to 10 volt analog input signal and provides a binary output indication of the amplitude of the input signal. Output indication accuracy is a function of the conversion time, and is determined by a switch on the front panel. Each of the seven positions of the rotary switch establishes an output word length, conversion accuracy, and conversion time for operation of the converter. Overall conversion error equals switching point error plus a quantization of $\pm 1/2$ the digital value of the LSB. Converter characteristics selected for each switch position are specified in Table 1.

CONVERTER SPECIFICATIONS

Monotonicity: Guaranteed for all settings

Aperture Time: Same as conversion time

Converter Recovery Time: None

Analog Input: 0 to -10 volts is standard. Bipolar or specific amplitude range input can be accommodated on special request. If a different voltage range is desired, it is recommended that an amplifier be used at the source, since this will also provide a low driving impedance and reduce the possibilities of noise pickup between the source and the converter.

Input Loading: ± 1 microampere and 125 picofarads for the standard 0 to -10 volt input.

Digital Output: A signed 6 to 12-bit binary number in 2's complement notation. A 0 volt input yields a digital output number of 4000_8 ; a -5 volt input produces 0000_8 ; and a -10 volt input gives an output of 3777_8 .

Controls: Binary readout indicators and a seven-position rotary switch for selecting converter characteristics are provided on the front panel.

The Type 139E Multiplexer Control is intended for use with the Type 138E or Type 189 analog-to-digital conversion system in applications where the PDP-8/S must process sampled analog data from multiple sources at high speeds. Under program control the multiplexer can select from 2 to 64 analog input signal channels for connection to the input of an analog-to-digital converter. Channel selection is provided by Type A100, A101, A102, or A103 Multiplex Switch FLIP CHIP modules. These module types each have slightly different timing, impedance, and power characteristics so that multiplexers can be built for wide differences in application by selecting the appropriate module type. Each module contains two independent, floating, transistor switches letting the user select any multiple of two channels to a maximum of 64. In the individual address mode, the Type 139E routes the analog data from any program-selected channel to the converter input. In the sequential address mode, the multiplexer advances the channel address by one each time it receives an incrementing command, returning to channel zero after scanning the last channel. Sequenced operations can be short-cycled when the number of channels in use is less than the maximum available.

A 6-bit channel address register (CAR) specifies a channel number from $0-77_8$. A channel address may be chosen in one of two ways. It can be specified by the content of bits 6-11 of the AC or by incrementing the content of the CAR.

MULTIPLEXER SPECIFICATIONS

Indicators: Six binary indicators on the front panel give visual indication of the selected channel.

Multiplexer Switching Time: The time required to switch from one channel to any program-specified channel, or to select the next adjacent channel when the content of the CAR is incremented is 2.5 microseconds. This time is measured from when either a select or increment command is received.

Both the converter and multiplexer circuits are constructed entirely of FLIP CHIP modules. Both the Type 138E converter and the Type 139E Multiplex Control (implemented to 24 input channels) circuits can be contained in one standard 64-connector module mounting panel.

The following IOT commands have been assigned to the Type 138E/139E converter system:

SKIP ON A-D FLAG (ADSF)

Octal Code: 6531

Event Time: 1

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The A-D converter flag is sensed, and if it contains a binary 1 (indicating that the conversion is complete) the content of the PC is incremented by one so that the next instruction is skipped.

Symbol: If A-D Flag = 1, then $PC + 1 = >PC$

CONVERT ANALOG VOLTAGE TO DIGITAL VALUE (ADCV)

Octal Code: 6532

Event Time: 2

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The A-D converter flag is cleared, the analog input voltage is converted to a digital value, and then the A-D converter flag is set to 1. The number of binary bits in the digital-value word and the accuracy of the word is determined by the preset switch position.

Symbol: 0 = > A-D Flag at start of conversion, then
1 = > A-D Flag when conversion is done.

READ A-D CONVERTER BUFFER (ADRB)

Octal Code: 6534

Event Time: 3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The converted number contained in the converter buffer (ADCB) is transferred into the AC as a normalized word (shifted into the most significant bits), unused bits of the AC are cleared, and the A-D converter flag is cleared.

Symbol: ADCB = > AC

0 = > A-D Converter Flag

CLEAR MULTIPLEXER CHANNEL (ADCC)

Octal Code: 6541

Event Time: 1

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The channel address register (CAR) of the multiplexer is cleared in preparation for setting of a new channel.

Symbol: 0 = > CAR

SET MULTIPLEXER CHANNEL (ADSC)

Octal Code: 6542

Event Time: 2

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The channel address register of the multiplexer is set to the channel specified by bits 6 through 11 of the AC. A maximum of 64 single-ended or 32 differential input channels can be used.

Symbol: AC 6-11 = > CAR

INCREMENT MULTIPLEXER CHANNEL (ADIC)

Octal Code: 6544

Event Time: 3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of the channel address register of the multiplexer is incremented by one. If the maximum address is contained in the register when this command is given, the minimum address (00) is selected.

Symbol: $CAR + 1 = > CAR$

A program to cycle through all channels of the converter a given number of times, storing the conversion values at successive core memory locations can be written as follows:

```
LOOP,      ADIC      /INCREMENT CAR
           ADCV      /INITIATE CONVERSION
           ADSF      /NEEDED FOR CONVERSION
           JMP.-1    /TIMES GREATER THAN
                    /28.5 us
           ADRB      /READ A-D CONVERTER BUFFER
           DCA I Z 10 /STORE RESULT IN ADDRESS SPECIFIED
                    /BY AUTO-INDEX REGISTER 10
           ISZ CNTR  /INCREMENT CYCLE COUNTER
           JMP LOOP  /REPEAT CYCLE
                    /END OF LOOP
```

Execution of this program loop takes 278 microseconds.

CHAPTER 11

DIGITAL-TO-ANALOG CONVERTER TYPE AA01A

The general purpose Digital-to-Analog Converter Type AA01A converts 12-bit binary computer output numbers to analog voltages. The basic option consists of three channels, each containing a 12-bit digital buffer register and a digital-to-analog converter (DAC). Digital input to all three registers is provided, in common, by one 12-bit input channel which receives bussed output connections from the PDP-8/S accumulator. Appropriate precision voltage reference supplies are provided for the converters.

One IOT microinstruction simultaneously selects a channel and transfers a digital number into the selected register. Each converter operates continuously on the content of the associated register to provide an analog output voltage.

Type AA01A options can be specified in a wide range of basic configurations: e.g., with from one to three channels, with or without output operational amplifiers, and with internally or externally supplied reference voltages. Configurations with double buffer registers in each channel are also available.

Each single-buffer channel of the equipment is operated by a single IOT command. Select codes of 55, 56, and 57 are assigned to the AA01A, making it possible to operate nine single-buffered channels or various configurations of double-buffered channels. A typical instruction for the AA01A is:

LOAD DIGITAL-TO-ANALOG CONVERTER 1 (DAL1)

Octal Code: 6551

Event Time: 1

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of the accumulator is loaded into the digital buffer register of channel 1.

Symbol: AC = > DAC1

The analog output voltage of a standard converter is from ground to -9.9976 volts (other voltages are available in equipment containing output operational amplifiers). All binary input numbers are assumed to be 12 bits in length with negative numbers represented in 2's complement notation. An input of 4000, yields an output of ground potential; an input of 0000_8 yields an output of -5 volts; and an input of 1777_8 yields an output of -10 volts minus the analog value of the least significant digital bit. Output accuracy is $\pm 0.0125\%$ of full scale and resolution is 0.025% of full scale value. Response time, measured directly at the converter output, is 3 microseconds for a full-scale step change to 1 least significant bit accuracy. Maximum buffer register loading rate is 2 megacycles.

CHAPTER 12

INCREMENTAL PLOTTER AND CONTROL TYPE 350B

Two models of California Computer Products Digital Incremental Recorder can be operated from a DEC Type 350 Increment Plotter Control. Characteristics of the four recorders are:

| <u>CCP Model</u> | <u>Step Size (inches)</u> | <u>Speed (steps/minute)</u> | <u>Paper Width (inches)</u> |
|----------------------|-----------------------------------|---------------------------------|-------------------------------------|
| 563 | 0.01 or 0.005 | 12,000 | 31 |
| 565 | 0.01 or 0.005 | 18,000 | 12 |

The principles of operation are the same for the two models of Digital Incremental Recorders. Bidirectional rotary step motors are employed for both the X and Y axes. Recording is produced by movement of a pen relative to the surface of the graph paper, with each instruction causing an incremental step. X-axis deflection is produced by motion of the drum; Y-axis deflection, by motion of the pen carriage. Instructions are used to raise and lower the pen from the surface of the paper. Each incremental step can be in any one of eight directions through appropriate combinations of the X and Y axis instructions. All recording (discrete points, continuous curves, or symbols) is accomplished by the incremental stepping action of the paper drum and pen carriage. Front panel controls permit single-step or continuous-step manual operation of the drum and carriage, and manual control of the pen solenoid. The recorder and control are connected to the computer program interrupt and instruction skip facility.

Instructions for the recorder and control are:

SKIP ON PLOTTER FLAG (PLSF)

Octal Code: 6501

Event Time: 1

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instructions

Operation: The plotter flag is sensed, and if it contains a 1 the content of the PC is incremented by one so the next sequential instruction is skipped.

Symbol: If plotter Flag = 1, then $PC + 1 = > PC$

CLEAR PLOTTER FLAG (PLCF)

Octal Code: 6502

Event Time: 2

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The plotter flag is cleared in preparation for issuing a plotter operation command.

Symbol: 0 = > Plotter Flag

PEN UP (PLPU)

Octal Code: 6504

Event Time: 3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The plotter pen is raised from the surface of the paper.

Symbol: None

PEN RIGHT (PLPR)

Octal Code: 6511

Event Time: 1

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The plotter pen is move to the right in either the raised or lower position.

Symbol: None

DRUM UP (PLDU)

Octal Code: 6512

Event Time: 2

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The plotter paper drum is moved upward. This command can be combined with the PLPR and PLDD commands.

Symbol: None

DRUM DOWN (PLDD)

Octal Code: 6514

Event Time: 3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The plotter paper drum is moved downward.

Symbol: None

PEN LEFT (PLPL)

Octal Code: 6521

Event Time: 1

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The plotter pen is moved to the left in either the raised or lowered position.

Symbol: None

DRUM UP (PLUD)

Octal Code: 6522

Event Time: 2

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The plotter paper drum is moved upward. This command is similar to command 6512 except that it can be combined with the PLPL or PLPD commands.

Symbol: None

PEN DOWN (PLPD)

Octal Code: 6524

Event Time: 3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The plotter pen is lowered to the surface of the paper.

Symbol: None

Program sequence must assume that the pen location is known at the start of a routine since there is no means of specifying an absolute pen location in an incremental plotter. Pen location can be preset by the manual controls on the recorder. During a subroutine, the PDP-8/S can track the location of the pen on the paper by counting the instructions that increment position of the pen and the drum.

CHAPTER 13

CARD READER AND CONTROL TYPE CR01C

The Card Reader and Control Type CR01C reads standard 12-row, 80-column punched cards at a maximum rate of 100 cards per minute. Cards are read by column, beginning with column 1. One select instruction starts the card moving past the read station. Once a card is in motion, all 80 columns are read. Data in a card column is sensed by mechanical star wheels which close an electrical contact when a hole (binary 1) is detected. Column information is read in one of two program selected modes: alphanumeric and binary. In the alphanumeric mode the 12 information bits in one column are automatically decoded and transferred into the least significant half of the accumulator as a 6-bit Hollerith code. Appendix 2 lists the Hollerith card codes. In the binary mode the 12 bits of a column are transferred directly into the accumulator so that the top row (12) is transferred into AC0 and the bottom row (9) is transferred into AC11. A punched hole is interpreted as a binary 1 and no hole is interpreted as a binary 0.

Three program flags indicate card reader conditions to the computer. The data ready flag rises and requests a program interrupt when a column of information is ready to be transferred into the AC. A read alphanumeric or read binary command must be issued within 1.5 milliseconds after the data ready flag rises to prevent data loss. The card done flag rises and requests a program interrupt when the card leaves the read station. A new select command must be issued within 25 milliseconds after the card done flag rises to keep the reader operating at maximum speed. Sensing of this flag can eliminate the need for counting columns, or combined with column counting can provide a check for data loss. The reader-not-ready flag can be sensed by a skip command to provide indication of card reader power off, no card in the read station, or that a reader failure has been detected. When this flag is raised the reader cannot be selected and select commands are ignored. The reader-not-ready flag is not connected to the program interrupt facility and cannot be cleared under program control. Manual intervention is required to clear the reader-not-ready flag. Instructions for the CR01C are:

SKIP ON DATA READY (RCSF)

Octal Code: 6631

Event Time: 1

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of the data ready flag is sensed, and if it contains a 1 (indicating that information for one card column is ready to be read) the content of the PC is incremented by one so the next sequential instruction is skipped.

Symbol: If Data Ready Flag = 1, then $PC + 1 = > PC$

READ ALPHANUMERIC (RCRA)

Octal Code: 6632

Event Time: 2

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The 6-bit Hollerith code for the 12 bits of a card column are transferred into bits 6 through 11 of the AC, and the data ready flag is cleared.

Symbol: AC6-11 V Hollerith Code = > AC6-11
0 = > Data Ready Flag

READ BINARY (RCRB)

Octal Code: 6634

Event Time: 3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instructions

Operation: The 12-bit binary code for a card column is transferred directly into the AC, and the data ready flag is cleared. Information from the card column is transferred into the AC so that card row 12 enters AC0, row 11 enters AC1, row 0 enters AC2, . . . and row 9 enters AC 11.

Symbol: AC V Binary Code = > AC

0 = > Data Ready Flag

SKIP ON CARD DONE FLAG (RCSP)

Octal Code: 6671

Event Time: 1

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of the card done flag is sensed, and if it contains a 1 (indicating that the card has passed the read station) the content of the PC is incremented to skip the next sequential instruction.

Symbol: If Card Done Flag = 1, then $PC + 1 = > PC$

SELECT CARD READER AND SKIP IF READY (RCSE)

Octal Code: 6672

Event Time: 2

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of the reader-not-ready flag is sensed and if it contains a 1 (indicating that the card reader is ready for programmed operation) the PC is incremented to skip the next sequential instruction; a card is started towards the read station from the feed hopper; and the card done flag is cleared. If the reader-not-ready flag contains a 0 (indicating power is off or no card is in the read station) card selection (motion) does not occur and the skip does not occur.

Symbol: If Reader-Not-Ready Flag = 1, then $PC + 1 = > PC$

0 = > Card Done Flag

CLEAR CARD DONE FLAG (RCRD)

Octal Code: 6674

Event Time: 3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The card done flag is cleared. This command allows a program to stop reading at any point in a card deck.

Symbol: 0 = > Card Done Flag

A logical instruction sequence to read cards is:

| | | |
|--------|-------------|--------------------------------------|
| START, | RCSE | /START CARD MOTION AND SKIP IF READY |
| | JMP NOT RDY | /JUMP TO SUBROUTINE THAT TYPES OUT |
| | | /"CARD READER MANUAL INTERVENTION |
| | | /REQUIRED" OR HALTS |

| | | |
|-------|--------------|--------------------------------------|
| NEXT, | RCSF | /DATA READY? |
| | JUMP .-1 | /NO, KEEP WAITING |
| | RCRA or RCRB | /YES, READ ONE CHARACTER OR ONE |
| | | /COLUMN |
| | DCA I STR | /STORE DATA |
| | RCSD | /END OF CARD? |
| | JMP NEXT | /NO, READ NEXT COLUMN |
| | JMP OUT | /YES, JUMP TO SUBROUTINE THAT CHECKS |
| | | /CARD COUNT OR REPEATS AT START FOR |
| | | /NEXT CARD |

No validity or registration checking is performed by the CR01C. A programmed validity check can be made by reading each card column in both the alphanumeric and the binary mode (within the 1.5 millisecond time limitation), then performing a comparison check.

Before commencing a card reading program energize the reader, load the feed hopper with cards, and manually feed the first card to the read station. The function of the manual controls and indicators are as follows (as they appear from the right to left on the card reader):

| <u>Control or Indicator</u> | <u>Function</u> |
|-----------------------------|--|
| ON/OFF switch | Controls the application of primary power to the reader. When power is applied, the reader is ready to respond to operation of the other keys or programmed commands. |
| AUTO/MAN switch | Controls card reading. In the manual position this switch disables the card feed mechanism so that cards must be manually placed on the read table and registered by pressing the REG key. In the automatic position card motion from the feed hopper through the read station is under program control. |
| REG switch | When the AUTO/MAN switch is in the AUTO position the REG key is used to feed the first card to the read station. When the AUTO/MAN switch is in the MAN position the REG key is used to feed a card manually placed on the read table. |
| SKIP switch | This key is not connected on the CR01C and has no effect on equipment operation. |
| CHECK READER indicator | This lamp is not connected on the CR01C. |
| READY indicator | Lights when the reader is energized and cards are present in the feed hopper. The plastic card cover should always be used on top of a deck of cards to assure that the ready switch and indicator are activated. |
| CARD RELEASE pushbutton | When pressed, this pushbutton (adjacent to the read station) releases a card already in the read station. |

CHAPTER 14

CARD READER AND CONTROL TYPE 451

The Card Reader and Control Type 451A operates at a rate of 200 cards per minute. Cards are read column by column. Column information is read in either alphanumeric or binary mode. The alphanumeric mode converts the 12-bit Hollerith code of one column into the 6-bit binary coded decimal code with code validity checking. The binary mode reads a 12-bit column directly into the PDP-8/S.

The control of the card reader differs from the control of other input devices, in that the timing of the read in sequence is dictated by the device. When the command to fetch a card is given, the card reader reads all 80 columns of information in sequence. To read a column, the program must respond to a flag set as each new column is started. The instruction to read the column must come within 2.2 msec of the flag at 200 cards per minute. The commands for the card reader are:

SKIP ON CARD READER FLAG (CRSF)

Octal Code: 6632

Event Time: 2

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instructions

Operation: The content of the card reader flag is sensed, and if it contains a 1 (indicating that a card column is present for reading) the content of the PC is incremented by one so that the next sequential instruction is skipped.

Symbol: If card reader flag = 1, then $PC + 1 = >PC$

READ CARD EQUIPMENT STATUS (CERS)

Octal Code: 6634

Event Time: 3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instructions

Operation: The content of the card reader flag and status levels are transferred into the content of bits AC6 through AC9. The AC bit assignments are:

AC6 = Flag is set to 1 (the flag rises after reading each of the 80 rows).
AC7 = Card done.
AC8 = Not ready (covers not in place, power is off, START pushbutton has not been pressed, hopper is empty, stacker is empty, stacker is full, a card is jammed, a validity check error has been detected, or the read circuit is defective).
AC9 = End of the file (EOF) (hopper is empty and operator has pushed EOF pushbutton).

Symbol: Status = > AC6-9

READ CARD READER BUFFER (CRRB)

Octal Code: 6671

Event Time: 1

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of the card column buffer (CCB) is transferred into the AC and the card reader flag is cleared. One CRRB command reads either alphanumeric or binary information.

Symbol: CCB = > AC

SELECT ALPHANUMERIC (CRSA)

Octal Code: 6672

Event Time: 2

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The card reader alphanumeric mode is selected and a card is started moving past the read heads. Information read into the CCB is in 6-bit alphanumeric form (the Hollerith code representing the decoded 12 row character in one column).

Symbol: None

SELECT BINARY (CRSB)

Octal Code: 6674

Event Time: 3

Indicators: IOT, FETCH, EXECUTE, END

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The card reader alphanumeric mode is selected and a card is started moving past the read heads. Information read into the CCB is in 12-bit binary form.

Symbol: None

Upon instruction to read the card reader buffer, the content of the 12-bit CCB is transferred into the AC. In the alphanumeric mode a 6-bit Hollerith code is transferred into AC6 through AC11 and AC0 through AC5 are cleared. In the binary mode the binary content of the 12 bits (or rows) in a card column are transferred into the AC so that row X is read into AC0, row Y into AC1, row 0 into AC2 . . and row 9 into AC11. The mode is specified by either the CRSA or CRSB command and can be changed while the card is being read.

CHAPTER 15

AUTOMATIC LINE PRINTER AND CONTROL TYPE 645

The line printer can print 300 lines of 120 characters per minute. Each character is selected from a set of 64 available, by a 6-bit binary code (Appendix 2 lists the ASCII character specified for each code). Each 6-bit code is loaded separately into a core storage printing buffer (LPB) from bits 6 through 11 of the AC. The LPB is divided into two 120-character sections. To load one section of the LPB requires 120 load instructions. A print command causes the characters specified by the last loaded section of the LPB to be printed on one line. As printing of one section of the LPB is in progress, the other section can be reloaded. After the last character in a line is printed, the section of the LPB from which characters were just printed is cleared automatically. The section of the LPB that is loaded and printed is alternated automatically within the printer and is not program specified.

The line printer can load characters into the LPB at a 10 μ sec rate, clears one section of the LPB in 3 to 6 msec, and moves paper at the rate of one line every 18 msec. When transfer of one code into the LPB is completed, the line printer done flag rises to indicate that the printer is ready to receive another code. When printing of the last character of a section of the LPB is completed, the line printer done flag rises and causes a program interrupt to request reloading of that section of the LPB. A line printer error flag rises and causes a program interrupt if the line printer detects an inoperative condition (printer power off, control circuits not reset, paper supply low, etc.).

A 3-bit format register (FR) in the printer is loaded from bits 9 through 11 of the AC during a print command. This register selects one of eight channels of a perforated tape in the printer to control spacing of the paper. The tape moves in synchronism with the paper until a hole is sensed in the selected channel to halt paper advance. A recommended tape has the following characteristics:

| FR Code (Octal) | Paper Spacing | Tape Track |
|--------------------|----------------------|---------------|
| 0 | 1 line | 2 |
| 1 | 2 lines | 3 |
| 2 | 3 lines | 4 |
| 3 | 6 lines (1/4 page) | 5 |
| 4 | 11 lines (1/2 page) | 6 |
| 5 | 22 lines (3/4 page) | 7 |
| 6 | 33 lines (line feed) | 8 |
| 7 | top of form | 1 |

The IOT microinstructions which command the line printer are:

SKIP ON LINE PRINTER ERROR (LSE)

Octal Code: 6651

Event Time: 1

Indicators: IOT, FETCH, EXECUTE

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of line printer error flag is sensed, and if it contains a binary 1, indicating that an error has been detected, the content of the PC is incremented by one so that the next sequential instruction is skipped.

Symbol: If line printer error flag = 1, then $PC + 1 = > PC$

CLEAR PRINTER BUFFER (LCB)

Octal Code: 6652

Event Time: 2

Indicators: IOT, FETCH, EXECUTE

Execution Time: 38 microseconds

Parity Test: instruction

Operation: Both sections of the line printer buffer are cleared in preparation for receiving new character information.

Symbol: $0 = > LPB$

LOAD PRINTER BUFFER (LLB)

Octal Code: 6654

Event Time: 3

Indicators: IOT, FETCH, EXECUTE

Execution Time: 38 microseconds

Parity Test: instruction

Operation: A section of the printer buffer is loaded from the content of bits 6 through 11 of the AC, then the AC is cleared.

Symbol: $AC6 - 11 = > LPB$, then $0 = > AC$

SKIP ON LINE PRINTER DONE FLAG (LSD)

Octal Code: 6661

Event Time: 1

Indicators: IOT, FETCH, EXECUTE

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The content of the line printer done flag is sensed and if it contains a binary 1 the content of the PC is incremented by one so that the next sequential instruction is skipped.

Symbol: If line printer done flag = 1, then $PC + 1 \Rightarrow PC$

CLEAR LINE PRINTER FLAGS (LCF)

Octal Code: 6662

Event Time: 2

Indicators: IOT, FETCH, EXECUTE

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The line printer done and error flags are cleared.

Symbol: 0 \Rightarrow Line printer done flag
0 \Rightarrow Line printer error flag

CLEAR FORMAT REGISTER (LPR)

Octal Code: 6654

Event Time: 3

Indicators: IOT, FETCH, EXECUTE

Execution Time: 38 microseconds

Parity Test: instruction

Operation: The line printer format register (FR) is cleared then loaded from the content of bits 9 through 11 of the AC, and the AC is cleared. The line contained in the section of the printer buffer (LPB) loaded last is printed. Paper is advanced in accordance with the selected channel of the format tape if the content of AC8 is a 1. If AC8 is a 0 paper advance is inhibited.

Symbol: 0 = > FR

AC9 - 11 = > FR

0 = > AC

The content of half of the LPB is printed.

If AC8 = 1, then advance paper according to format tape channel FR.

The following routine demonstrates the use of these commands in a sequence which prints an unspecified number of 120-character lines. This sequence assumes that the printer is not in operation, that the paper is manually positioned for the first line of print, and that one-character words are stored in sequential core memory locations beginning at 2000. The PRINT location starts the routine.

| | | |
|--------|----------|-----------------------------------|
| PRINT, | LCB | /INITIALIZE PRINTER BUFFER |
| | CLA | |
| | TAD LOC | /LOAD INITIAL CHARACTER ADDRESS |
| | DCA 10 | /STORE IN AUTO-INDEX REGISTER |
| LRPT, | TAD CNT | /INITIALIZE CHARACTER COUNTER |
| | DCA TEMP | |
| LOOP, | LD\$ | /WAIT UNTIL PRINTING BUFFER READY |
| | JMP LOOP | |
| | LCF | /CLEAR LINE PRINTER FLAG |
| | TAD I 10 | /LOAD AC FROM CURRENT CHARACTER |
| | | /ADDRESS |
| | LLB | /LOAD PRINTING BUFFER |
| | ISZ TEMP | /TEST FOR 120 CHARACTERS LOADED |
| | JMP LOOP | |
| | TAD FRM | /LOAD SPACING CONTROL AND |
| | LPR | /PRINT A LINE |
| | JMP LRPT | /JUMP TO PRINT ANOTHER LINE |
| LOC. | 1777 | /INITIAL CHARACTER ADDRESS -1 |
| CNT. | -170 | /CHARACTER COUNTER = 120 DECIMAL |
| TEMP. | 0 | /CURRENT CHARACTER ADDRESS |
| FRM. | 10 | /SPACING CONTROL AND FORMAT |

CHAPTER 16

STANDARD PDP-8/S OPERATION

CONTROLS AND INDICATORS

Manual control of the PDP-8/S is exercised by means of keys and switches on the operator console. Visual indications of the machine status and the content of major registers and control flip-flops is also given on this console. Indicator lamps light to denote the presence of a binary 1 in specific register bits and in control flip-flops. The function of these controls and indicators is listed in Table 2, and their location is shown in Figure 7. The functions of all controls and indicators of the Model 33 ASR Teletype unit are described in Table 4, as they apply to operation of the computer. The Teletype console is shown in Figure 8.

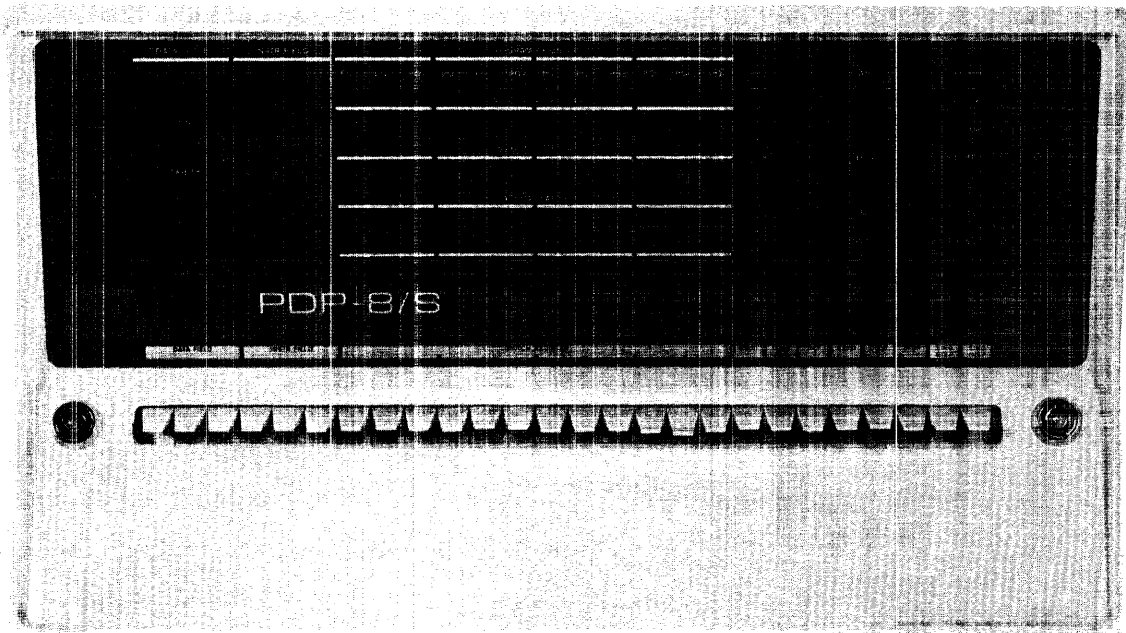


Figure 7 PDP-8/S Operator Console

TABLE 2 OPERATOR CONSOLE CONTROLS AND INDICATORS

| Control or Indicator | Function |
|----------------------|--|
| PANEL LOCK switch | With this key-operated switch turned clockwise, all keys and switches except the SWITCH REGISTER switches on the operator console are disabled. In this condition the program can not be disturbed by inadvertent key operation. The program can, however, monitor the content of the SR by execution of the OSR instruction. With this switch turned counterclockwise all operator console keys and switches function normally. |

TABLE 2 OPERATOR CONSOLE CONTROLS AND INDICATORS (continued)

| <u>Control or Indicator</u> | <u>Function</u> |
|-----------------------------|---|
| POWER switch | In the counterclockwise position this key-operated switch removes primary power from the computer, and in the clockwise position it applies power. |
| START key | Starts the computer program by turning off the program interrupt circuits; clearing the AC, L, MB, and IR; setting the End state, transferring the content of the PC into the MA; and setting the RUN flip-flop requesting a memory cycle. Therefore, the word stored at the address currently held by the PC is taken as the first instruction. |
| LOAD ADDRESS key | Pressing this key sets the content of the SR into the AC, sets the RUN flip-flop for one cycle, transfers the content of AC into PC serially, sets the content of the INST FIELD switches into the IF, and sets the content of the DATA FIELD switches into the DF. |
| DEPOSIT key | Lifting this key sends the contents of PC to MA serially, and sets the content of the SR into the AC from where it is shifted into the MB during a single machine cycle, and a memory write request is initiated causing the data in MB to be stored at the current content of the MA. The content of the PC is then incremented by one, to allow storing of information in sequential memory addresses by repeated operation of the DEPOSIT key. |
| EXAMINE key | Pressing this key sends the contents of PC to MA serially and sets the content of core memory, at the address specified by the content of the MA, into the MB. The content of the PC is then incremented by one to allow examination of the content of sequential core memory addresses by repeated operation of the EXAMINE key. |
| CONTINUE key | Pressing this key sets the RUN flip-flop to continue the program in the state and instruction designated by the lighted console indicators, at the address currently specified by the PC. |
| STOP key | Causes the RUN flip-flop to be cleared at the end of the FETCH cycle in progress at the time the key is pressed. |
| SINGLE STEP switch | The switch is off in the down position. In the up position, the switch causes the RUN flip-flop to be cleared to disable the timing circuits at the end of one cycle of operation, i.e., the end of each major state. Thereafter, repeated operation of the CONTINUE key steps the program one major state at a time so that the content of registers can be observed in each state. (Note: The machine stops after memory cycles are completed.) |
| SINGLE INSTRUCTION switch | The switch is off in the up position. In the down position, the switch causes the RUN flip-flop to be cleared at the beginning of the next instruction execution. The computer will always appear to stay in the Fetch state with the MB containing the instruction to be executed. Repeated operation of the CONTINUE key steps the program one instruction at a time. |

TABLE 2 OPERATOR CONSOLE CONTROLS AND INDICATORS (continued)

| <u>Control or Indicator</u> | <u>Function</u> |
|---|---|
| SWITCH REGISTER switches | Provide a means of manually setting a 12-bit word into the machine. Switches in the down position correspond to binary zeros, up to ones. The content of this register is loaded into the PC by the LOAD ADDRESS key or into the MB and core memory by the DEPOSIT key. The content of the switch register can be set into the AC under program control of the OSR instruction. |
| DATA FIELD indicators and switches* | The indicators denote the content of the data field register (DF) and the switches serve as an extension of the SR to load the DF by means of the LOAD ADDRESS key. The DF determines the core memory field of data storage and retrieval. |
| INST FIELD indicators and switches* | The indicators denote the content of the instruction field register (IF) and the switches serve as an extension of the SR to load the IF by means of the LOAD ADDRESS key. The IF determines the core memory field from which instructions are to be taken. |
| PROGRAM COUNTER indicators | Indicate the content of the PC. When the machine is stopped, the content of the PC indicates the core memory address of the first instruction to be executed when the START key is operated. The instruction to be executed after the CONTINUE key is operated is currently sitting in the MB and PC indicates the next instruction to be executed. |
| MEMORY ADDRESS indicators | Indicate the content of the MA. Usually the content of the MA denotes the core memory address of the word currently or previously read or written. After operation of either the DEPOSIT or EXAMINE key, the content of the MA indicates the core memory address at which information was just written or read. |
| MEMORY BUFFER indicators | Indicate the content of the MB. Usually the content of the MB designates the word just read or written at the core memory address held in the MA. |
| ACCUMULATOR indicators | Indicates the content of the AC. |
| LINK indicator | Indicates the content of the L. |
| Instruction indicators (AND, TAD, ISZ, DCA, JMS, JMP, IOT, OPR) | Indicate the decoded output of the IR as the instruction currently in progress. |
| FETCH, INDEX, DEFER, EXECUTE, END, and BREAK indicators | Indicate the primary control state of the machine and that the current processor cycle is a Fetch, Index, Defer, Execute, End or Break cycle, respectively. |
| ION indicator | Indicates the 1 status of the INT. ENABLE flip-flop. When lit, the program in progress can be interrupted by receipt of a Program Interrupt Request signal from an I/O device. |

* Activated only on systems containing the Memory Extension Control option.

TABLE 2 OPERATOR CONSOLE CONTROLS AND INDICATORS (continued)

| <u>Control or Indicator</u> | <u>Function</u> |
|-----------------------------|---|
| PAUSE indicator | Indicates the 1 status of the PAUSE flip-flop when lit. A memory request sets the PAUSE flip-flop to inhibit advance of the processor timing generator. The PAUSE flip-flop is automatically reset by the memory when a memory cycle has been completed and the data has either been removed from the memory buffer for storing in memory or loaded from memory into the memory buffer. |
| RUN indicator | Indicates the 1 status of the RUN flip-flop. When lit, the internal timing circuits are enabled and the machine performs instructions. |

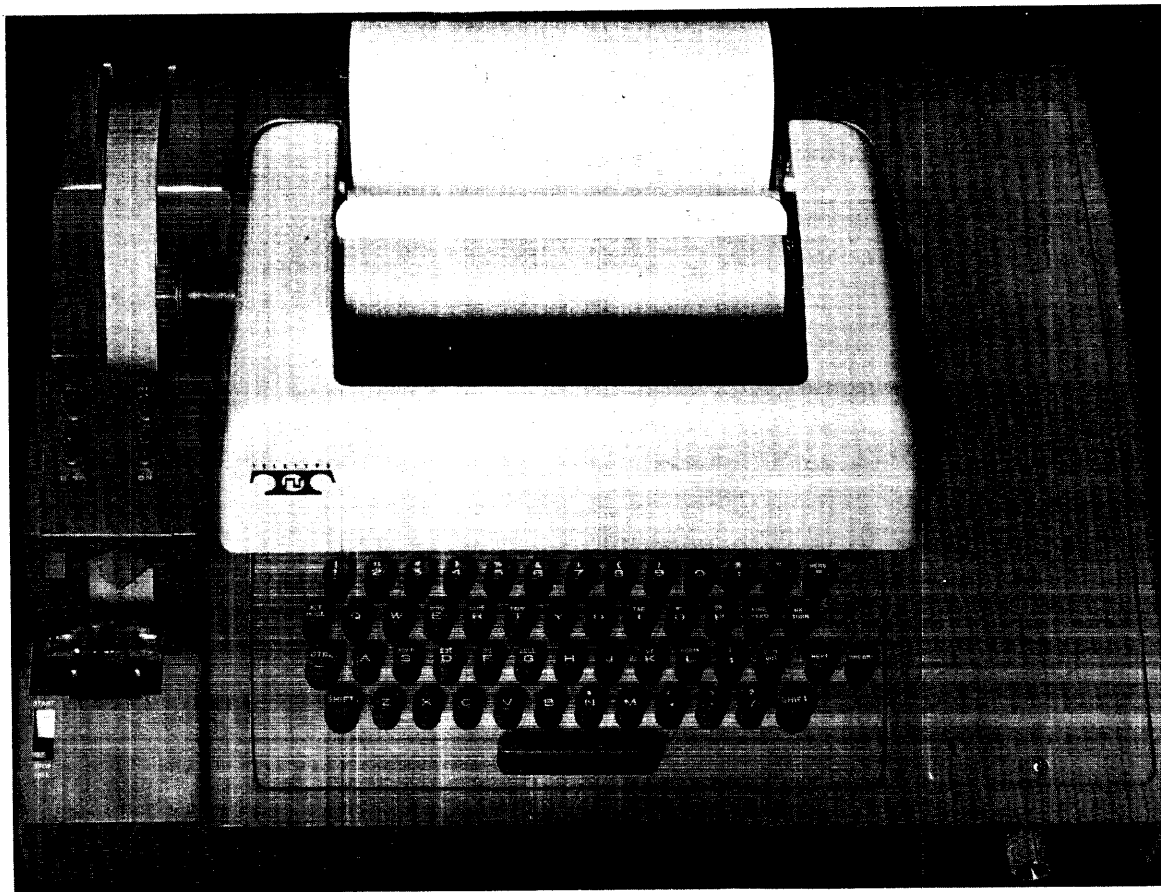


Figure 8 Teletype Model 33 ASR Console

TABLE 3 TELETYPE CONTROLS AND INDICATORS

| Control or Indicator | Function |
|------------------------|--|
| REL. pushbutton | Disengages the tape in the punch to allow tape removal or tape loading. |
| B. SP. pushbutton | Backspaces the tape in the punch by one space, allowing manual correction or rub out of the character just punched. |
| OFF and ON pushbuttons | Control use of the tape punch with operation of the Teletype keyboard/printer. |
| START/STOP/FREE switch | Controls use of the tape reader with operation of the Teletype. In the lower FREE position the reader is disengaged and can be loaded or unloaded. In the center STOP position the reader mechanism is engaged but de-energized. In the upper START position the reader is engaged and operated under program control. |
| Keyboard | Provides a means of printing on paper in use as a typewriter and punching tape when the punch ON pushbutton is pressed, and provides a means of supplying input data to the computer when the LINE/OFF/LOCAL switch is in the LINE position. |

LINE/OFF/LOCAL switch Controls application of primary power in the Teletype and controls data connection to the processor. In the LINE position the Teletype is energized and connected as an I/O device of the computer. In the OFF position the Teletype is de-energized. In the LOCAL position the Teletype is energized for off-line operation, and signal connections to the processor are broken. Both line and local use of the Teletype require that the computer be energized through the POWER switch.

OPERATING PROCEDURES

Many means are available for loading and unloading PDP-8/S information. The means used are, of course, dependent upon the form of the information, time limitations, and the peripheral equipment connected to the computer. The following procedures are basic to any use of the PDP-8/S, and although they may be used infrequently as the programming and use of the computer become more sophisticated, they are valuable in preparing the initial programs and learning the function of machine input and output transfers.

Manual Data Storage and Modification

Programs and data can be stored or modified manually by means of the facilities on the operator console. Chief use of manual data storage is made to load the readin mode loader program into the computer core memory. The readin mode (RIM) loader is a program used to automatically load programs into PDP-8/S from perforated tape in RIM format. This program and the RIM tape format are described in Appendix 6 of this handbook and in Digital Program Library descriptions. The RIM program listed in the Appendix can be used as an exercise in manual data storage. To store data manually in the PDP-8/S core memory:

1. Turn the PANEL LOCK switch counterclockwise and turn the POWER switch clockwise.
2. Set the bit switches of the SWITCH REGISTER (SR) to correspond with the address bits of the first word to be stored. Press the LOAD ADDRESS key and observe that the address set by the SR is held in the PC, as designated by lighted PROGRAM COUNTER indicators corresponding to switches in the 1 (up) position and unlighted indicators corresponding to switches in the 0 (down) position.
3. Set the SR to correspond with the data or instruction word to be stored at the address just set into the PC. Lift the DEPOSIT key and observe that the MB, and hence the core memory, hold the word set by the SR.

Also, observe that the PC has been incremented by one so that additional data can be stored at sequential addresses by repeated SR setting and DEPOSIT key operation.

To check the content of an address in core memory, set the address into the PC as in step 2, then press the EXAMINE key. The content of the address is then designed by the MEMORY BUFFER and ACCUMULATOR indicators. The content of the PC is incremented by one with operation of the EXAMINE key, so the content of sequential addresses can be examined by repeated operation after the original (or starting) address is loaded. The content of any address can be modified by repeating both steps 2 and 3.

Loading Data Under Program Control

Information can be stored or modified in the computer automatically only by enacting programs previously stored in core memory. For example, having the RIM loader stored in core memory allows RIM format tapes to be loaded as follows:

1. Turn the PANEL LOCK switch counterclockwise and turn the POWER switch clockwise.
2. Set the Teletype LINE/OFF/LOCAL switch to the LINE position.
3. Load the tape in the Teletype reader by setting the START/STOP/FREE switch to the FREE position, releasing the cover guard by means of the latch at the right, loading the tape so that the sprocket wheel teeth engage the feed holes in the tape, closing the cover guard, and setting the switch to the STOP position. Tape is loaded in the back of the reader so that it moves toward the front as it is read. Proper positioning of the tape in the reader finds three bit positions being sensed to the left of the sprocket wheel and five bit positions being sensed to the right of the sprocket wheel.
4. Load the starting address of the RIM loader program (not the address of the program to be loaded) into the PC by means of the SR and the LOAD ADDRESS key.
5. Press the computer START key and set the 3-position Teletype reader switch to the START position. The tape will be read automatically.

Automatic storing of the binary loader (BIN) program is performed by means of the RIM loader program as previously described. With the BIN loader stored in core memory, program tapes assembled in the program assembly language (PAL III) binary format can be stored as described in the previous procedure except that the starting address of the BIN loader (usually 7777) is used in step 4. When storing a program in this manner, the computer stops and the AC should contain all zeros if the program is stored properly. If the computer stops with a number other than zero in the AC, a checksum error has been detected. When the program has been stored, it can be initiated by loading the program starting address (usually designated on the leader of the tape) into the PC by means of the SR and LOAD ADDRESS key, then pressing the START key.

Off-Line Teletype Operation

The Teletype can be used separately from the PDP-8/S for typing, punching tape, or duplicating tapes. To use the Teletype in this manner:

1. Assure that the computer PANEL LOCK switch is turned counterclockwise and turn the POWER switch clockwise.
2. Set the Teletype LINE/OFF/LOCAL switch to the LOCAL position.
3. If the punch is to be used, load it by raising the cover, manually feeding the tape from the top of the roll into the guide at the back of the punch, advancing the tape through the

punch by manually turning the friction wheel, and then closing the cover. Energize the punch by pressing the ON pushbutton, and produce about two feet of leader. The leader-trailer can be code 200 or 377. To produce the code 200 leader, simultaneously press and hold the CTRL and SHIFT keys with the left hand; press and hold the REPT key; press and release the @ key. When the required amount of leader has been punched release all keys. To produce the 377 code, simultaneously press and hold both the REPT and RUB OUT keys until a sufficient amount of leader has been punched.

If an incorrect key is struck while punching a tape, the tape can be corrected as follows; if the error is noticed after typing and punching N characters, press the punch B. SP. (backspace) pushbutton N + 1 times and strike the keyboard RUB OUT key N + 1 times. Then continue typing and punching with the character which was in error.

To duplicate and obtain a listing of an existing tape: Perform the procedure under the current heading. Then load the tape to be duplicated as described in step 2 of the procedure under Loading Data Under Program Control. Initiate tape duplication by setting the reader START/STOP/FREE switch in the START position. The punch and teleprinter stop when the tape being duplicated is completely read.

Corrections to insert or delete information on a perforated tape can be made by duplicating the correct portion of the tape, and manually punching additional information or inhibiting punching of information to be deleted. This is accomplished by duplicating the tape and carefully observing the information being typed as the tape is read. In this manner the reader START/STOP/FREE switch can be set to the STOP position just before the point of the correction is typed. Information to be inserted can then be punched manually by means of the keyboard. Information can be deleted by pressing the punch OFF pushbutton and operating the reader until the portion of the tape to be deleted has been typed. It may be necessary to backspace and rub out one or two characters on the new tape if the reader is not stopped precisely on time. The number of characters to be rubbed out can be determined exactly by the typed copy. Be sure to count spaces when counting typed characters. Continue duplicating the tape in the normal manner after making the corrections.

New, duplicated, or corrected perforated tapes should be verified by reading them off line and carefully proofreading the typed copy.

CHAPTER 17

PDP-8/S INPUT/OUTPUT FACILITIES

Since the processing power of a computer system depends largely upon the range and number of peripheral devices that can be connected to it, the PDP-8/S has been designed to interface readily with a broad variety of external equipment.

The simple I/O techniques of the PDP-8/S, the availability of DEC's FLIP CHIP logic circuit modules, and DEC's policy of giving assistance wherever possible allow inexpensive, straight-forward device interfaces to be realized. Should questions arise relative to the computer interface characteristics, the design of device interfaces using DEC modules, or installation planning, customers are invited to telephone the main plant in Maynard, Massachusetts, or any of the sales offices. Digital Equipment Corporation makes no representation that the interconnection of its circuit modules in the manner described herein will not infringe on existing or future patent rights. Nor do the descriptions contained herein imply the granting of license to use, manufacture, or sell equipment constructed in accordance therewith.

The basic PDP-8/S contains a processor and core memory composed of FLIP CHIP circuit modules. These hybrid silicon circuits have an operating temperature range exceeding the limits of 32° to 130°F, so no air-conditioning is required at the computer site. Standard 115v, 60-cps power operates an internal solid-state power supply that produces all required voltages and currents.

High-capacity, high-speed I/O capabilities of the PDP-8/S allow it to operate a variety of peripheral devices in addition to the standard Teletype keyboard/printer, tape reader, and tape punch. DEC options, consisting of an interface and normal data processing equipment, are available for connecting into the computer system. These options include card equipment, line printers, magnetic tape transports, magnetic drums, analog-to-digital converters, CRT displays, and digital plotters. The PDP-8/S system can also accept other types of instruments or hardware devices that have an appropriate interface. Up to 61 devices requiring three programmed command pulses, or up to 193 devices requiring one programmed command pulse can be connected to the computer.

Interfacing of any devices to the computer requires no modifications to the processor and can be achieved in the field.

LOGIC SYMBOLS

Figure 9 defines the symbols used in this handbook to express signals and digital logic circuits.

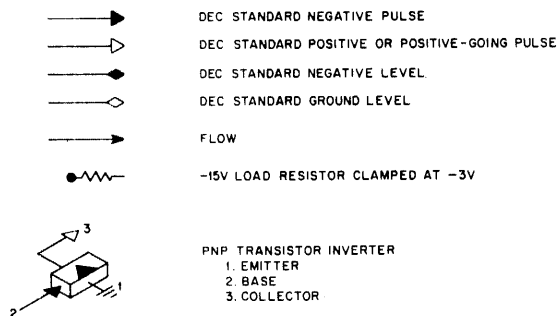
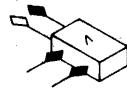
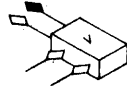


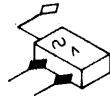
Figure 9 Logic Symbols



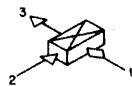
LOGIC AND GATE FOR
NEGATIVE SIGNALS
WITH COMPLEMENTARY
OUTPUT SIGNALS



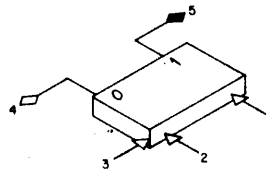
LOGIC OR GATE FOR
GROUND LEVEL SIGNALS
WITH COMPLEMENTARY
OUTPUT SIGNALS



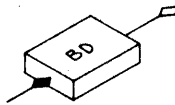
LOGIC NAND GATE FOR
NEGATIVE SIGNALS



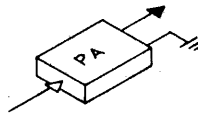
DIODE-CAPACITOR-DIODE GATE
1. CONDITIONING LEVEL INPUT
2. TRIGGERING PULSE INPUT
3. PULSE OUTPUT



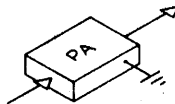
FLIP-FLOP (BISTABLE MULTIVIBRATOR)
1. GATED SET-TO-1 INPUT
2. GATED CLEAR-TO-0 INPUT
3. DIRECT CLEAR-TO-0 INPUT
4,5. OUTPUTS



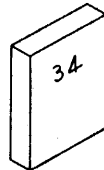
INVERTING BUS DRIVER



B OR W SERIES
PULSE AMPLIFIER. OUTPUT
CAN BE MADE POSITIVE OR
NEGATIVE BY REVERSING
GROUND AND SIGNAL OUTPUT
TERMINALS



R OR S SERIES PULSE AMPLIFIER
OUTPUT ALWAYS POSITIVE,
REFERENCED TO -3V.



DEVICE SELECTOR
LOGIC AS USED FOR ONE
SELECT CODE

Figure 9 Logic Symbols (continued)

CHAPTER 18

PROGRAMMED DATA TRANSFERS

The majority of I/O transfers take place under control of the PDP-8/S program, taking advantage of control elements built into the computer. The maximum data transfer rate for programmed operations of 12-bit words is 6 kc when no status checking, end transfer check, etc., is done. This speed is well beyond the normal rate required for typical laboratory or process control instrumentation.

The PDP-8/S is a parallel-transfer machine that distributes and collects data in bytes of up to twelve bits. All programmed data transfers take place through the accumulator, the 12-bit arithmetic register of the computer. The computer program controls the loading of information into the accumulator (AC) for an output transfer, and for storing information in core memory from the AC for an input transfer. Output information in the AC is power amplified and supplied to the interface connectors for bussed connection to many peripheral devices. Then the program-selected device can sample these signal lines to strobe AC information into a control or information register. Input data arrives at the AC as pulses received at the interface connectors from bussed outputs of many devices. Gating circuits of the program-selected device produce these pulses. Command pulses generated by the device flow to the input/output skip facility (IOS) to sample the condition of I/O device flags. The IOS allows branching of the program based upon the condition or availability of peripheral equipment, effectively making programmed decisions to continue the current program or jump to another part of the program, such as a subroutine that services an I/O device.

The bussed system of input/output data transfers imposes the following requirements on peripheral equipment:

- a. The ability of each device to sample the select code generated by the computer during IOT instructions and, when selected, to be capable of producing sequential IOT command pulses in accordance with computer-generated IOP pulses. Circuits which perform these functions in the peripheral device are called the device selector (DS).
- b. Each device receiving output data from the computer must contain gating circuits at the input of a receiving register capable of strobing the AC signal information into the register when triggered by a command pulse from the DS.
- c. Each device which supplies input data to the computer must contain gating circuits at the output of the transmitting register capable of sampling the information in the output register and supplying a pulse to the computer input bus when triggered by a command pulse from the DS.
- d. Each device should contain a busy/done flag (flip-flop) and gating circuits which can pulse the computer input/output skip bus upon command from the DS when the flag is set in the binary 1 state to indicate that the device is ready to transfer another byte of information.

Figure 10 shows the information flow within the computer which effects a programmed data transfer with input/output equipment. All instructions stored in core memory as a

program sequence are read into the memory buffer register (MB) for execution. The transfer of the operation code in the three most significant bits (bits 0, 1, and 2) of the instruction into the instruction register (IR) takes place and is decoded to produce appropriate control signals. The computer, upon recognition of the operation code as an IOT instruction, enters an 10 μ sec WTX cycle and enables the IOP generator to produce time sequenced IOP pulses as determined by the three least significant bits of the instruction (bits 9, 10, and 11 in the MB). These IOP pulses and the buffered output of the select code from bits 3-8 of the instruction word in the MB are bussed to device selectors in all peripheral equipment. Figure 10 indicates the timing of programmed data transfers and Figure 11 shows the decoding of the IOT instruction.

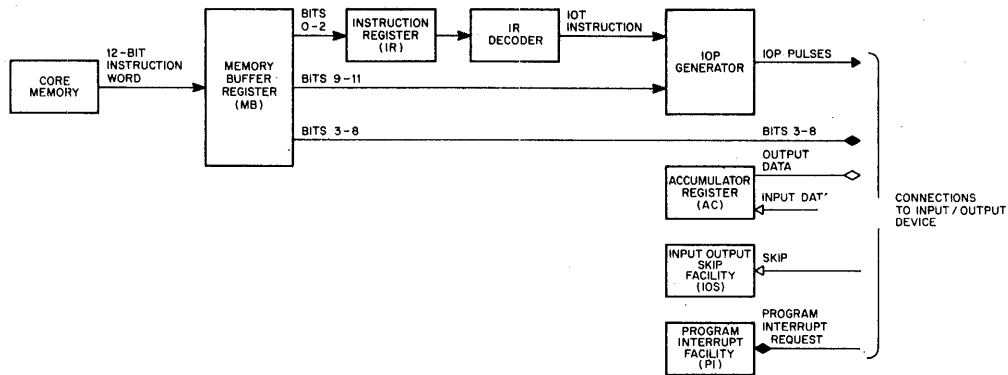


Figure 10 Programmed Data Transfer Interface Block Diagram

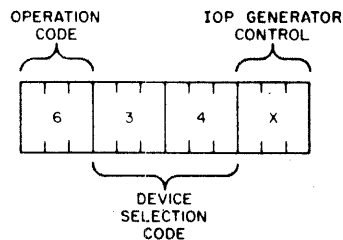


Figure 11 Typical IOT Instruction Decoding

Devices which require immediate service from the computer program, or which take an exorbitant amount of computer time to discontinue the main program until transfer needs are met, can use the program interrupt (PI) facility. In this mode of operation, the computer can initiate operation of I/O equipment and continue the main program until the device requests servicing. A signal input to the PI requesting a program interrupt causes storing of the conditions of the main program and initiates a subroutine to service the device. At the conclusion of this subroutine, the main program is reinstated until another interrupt request occurs.

TIMING AND IOP GENERATOR

When the IR decoder detects an operation code of 6₈, it identifies an IOT instruction and initiates operation of the IOP generator. The logic circuits of the IOP generator are shown in Figure 12 to consist of three similar channels; each channel consisting of a gated delay, a gated pulse amplifier, and an output pulse amplifier. Operation of the first channel is triggered by BT0 of WTX and operation of the other two channels is triggered by the pulse output of the delay in the previous channel. Series connection of the delays produces sequential operation of the three channels.

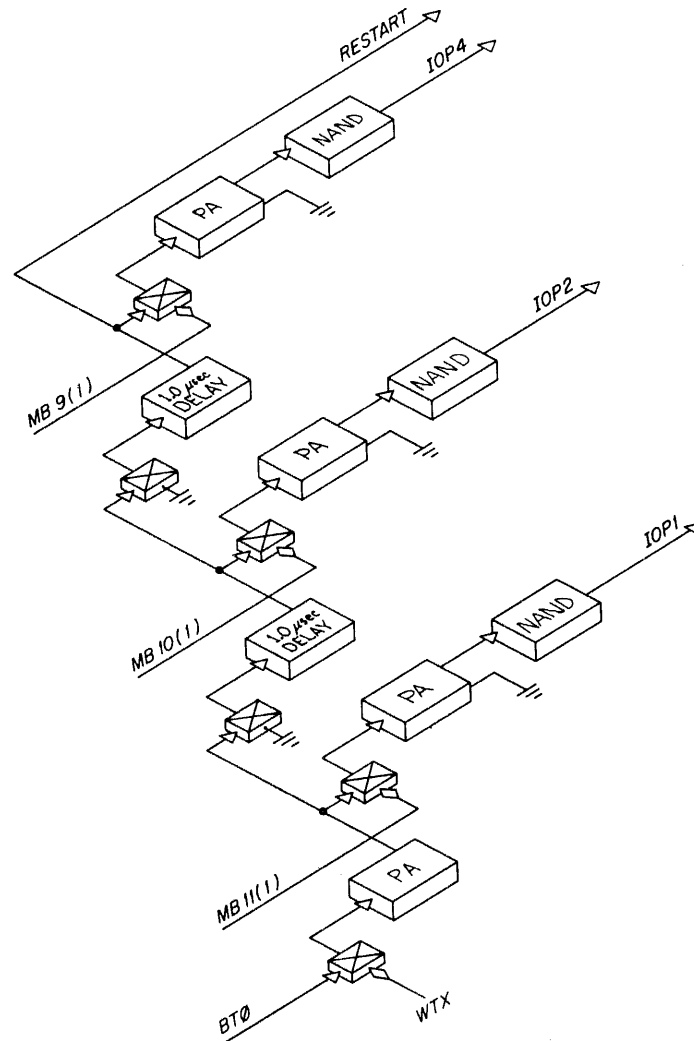


Figure 12 IOP Generator Logic

The gated pulse amplifier of each channel samples the content of one bit of the instruction when the delay output pulse occurs. If the sampled bit contains a binary 1, the gated pulse amplifier is triggered and the output pulse amplifier is operated to produce an IOP pulse. A diode-capacitor-diode (DCD) gate at the input of each gated pulse amplifier serves as a 2-input AND gate. The binary 1 status of one of the least significant bits of the instruction in the MB supplies the conditioning level of each of these gates. The output of the gated pulse amplifier initiates operation of the output amplifier to generate an IOP pulse which is available at the interface connector as a DEC standard 0.4 μ sec negative pulse. This configuration allows each IOP pulse to be individually programmed, permits a sequence of up to three events to occur within each instruction, and provides 1 μ sec between events for normal device circuit set-up times. The instruction bit that enables or disables generation of each IOP pulse, the corresponding number of the IOT pulse produced in the DS from the IOP pulse, and the event time for each IOP pulse is:

| <u>Instruction Bit</u> | <u>IOP Pulse</u> | <u>IOT Pulse</u> | <u>Event Time</u> |
|----------------------------|----------------------|----------------------|-----------------------|
| 11 | IOP 1 | IOT 1 | 1 |
| 10 | IOP 2 | IOT 2 | 2 |
| 9 | IOP 4 | IOT 4 | 3 |

DEVICE SELECTOR (DS)

Bits 3 through 8 of an IOT instruction serve as a device or subdevice select code. Bus drivers in the processor buffer both the binary 1 and 0 output signals of MB3-8 and distribute them to the interface connectors for bussed connection to all device selectors. Each DS is assigned a select code and is enabled only when the assigned code is present in the MB. When enabled, a DS regenerates IOP pulses as IOT command pulses and transmits these pulses to skip, input, or output gates within the device and/or to the processor to clear the AC.

Each group of three command pulses requires a separate DS channel (W103 module), and each DS channel requires a different select code (or I/O device address). One I/O device can, therefore, use several DS channels. Note that the processor produces the pulses identified as IOP 1, IOP 2, and IOP 4 and supplies them to all device selectors. The device selector produces pulses IOT 1, IOT 2, and IOT 4 which initiate a transfer or effect some control. Figure 13 shows generation of command pulses by several DC channels.

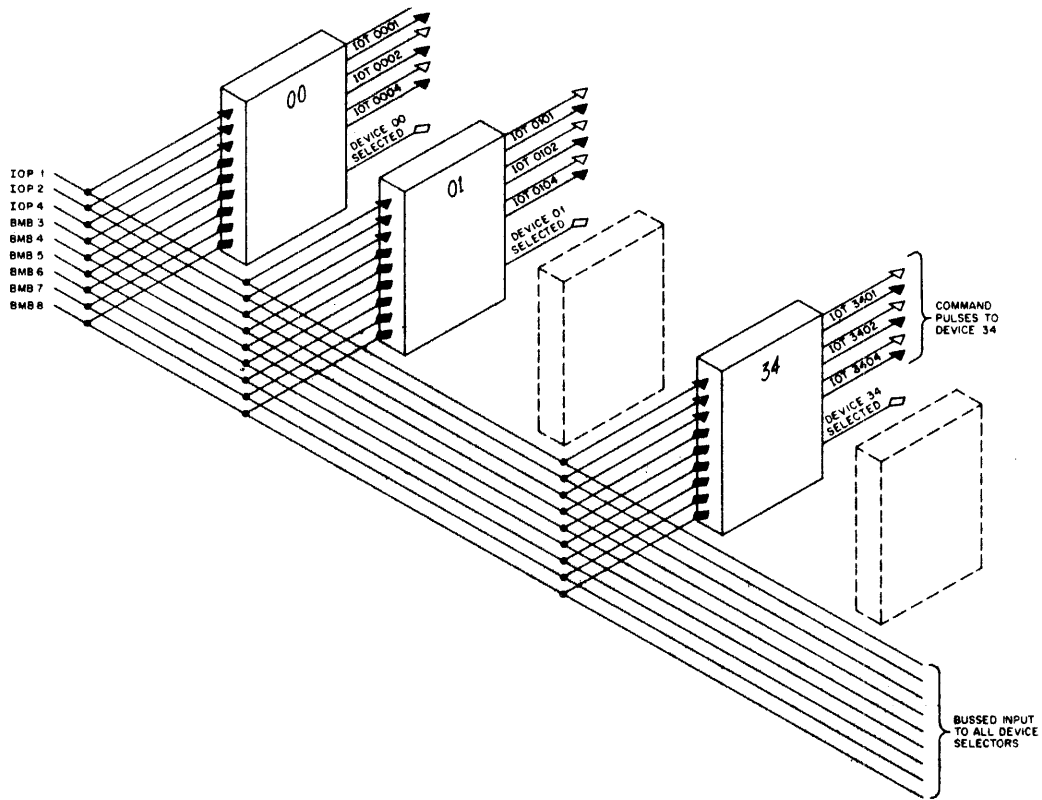


Figure 13 Generation Of IOT Command Pulses by Device Selectors

The logical representation for a typical channel of the DS, using channel 34, is shown in Figure 14. A 6-input NAND gate wired to receive the appropriate signal outputs from MB3-8 for select code 34 activates the channel. In the DS module, the NAND gate contains 14 diode input terminals; 12 of these connect to the complementary outputs of MB3-8, and 2 are open to receive subdevice or control condition signals as needed. Either the 1 or the 0 signal from each MB bit is disconnected by removing the appropriate diode from the NAND gate when establishing the select code. The ground level output of the NAND gate indicates when the IOT instruction selects the device, and can therefore enable circuit operations within the device. This output also enables three gating inverters, allowing them to trigger a pulse amplifier if an IOP pulse occurs. The positive output from each pulse amplifier is an IOT command pulse identified by the select code and the number of the initiating IOP pulse. Three inverters receive the positive IOT pulses to produce complementary IOT output pulses. A pulse amplifier module can be connected in each channel of the DS to provide greater output drive or to produce pulses of a specific duration required by the selected device.

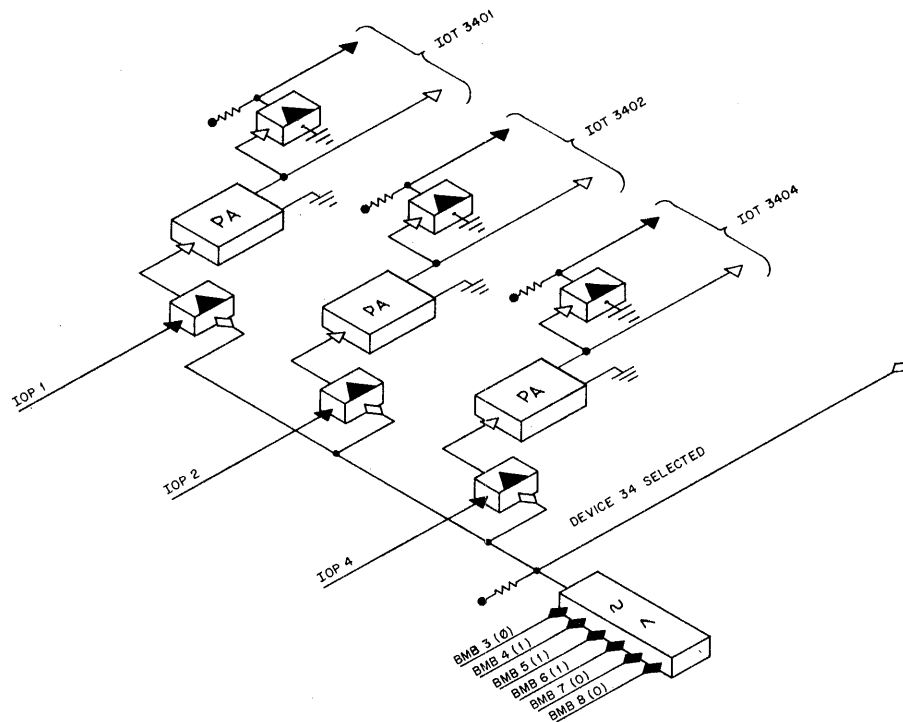


Figure 14 Typical Device Selector (Device 34)

INPUT/OUTPUT SKIP (IOS)

Generation of an IOT pulse can be used to test the condition or status of a device flag, and to continue to or skip the next sequential instruction based upon the results of this test. This operation is performed by a 2-input AND gate in the device connected as shown in Figure 15. One input of the skip gate receives the status level (flag output signal), the second input receives an IOT pulse, and the output drives the computer IOS bus to ground when the skip conditions are fulfilled. When the IOS bus is driven to ground, the content of the program counter is incremented by 1 to advance the program count without executing the instruction at the current program count. In this manner an IOT instruction can check the status of an I/O device flag and skip the next instruction if the device requires servicing. Programmed testing in this manner allows the routine to jump out of sequence to a subroutine that services the device tested.

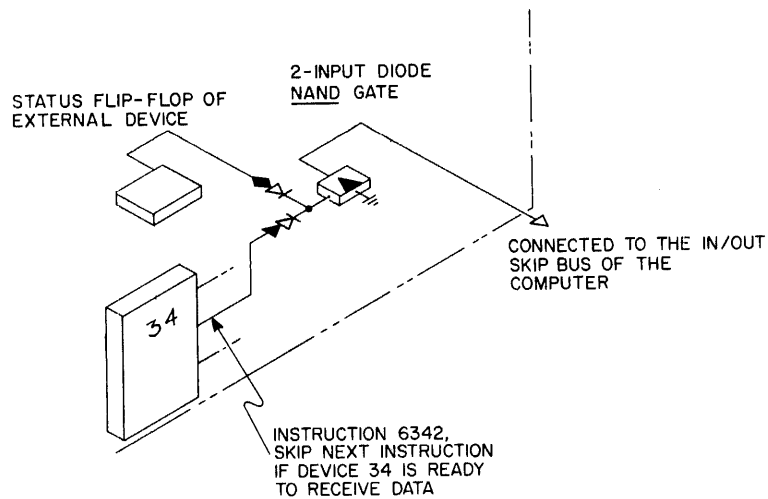


Figure 15 Use of IOS To Test The Status of an External Device

Assuming that a device is already operating, a possible program sequence to test its availability follows:

| <u>Address</u> | <u>Instruction</u> | <u>Remarks</u> |
|----------------|--------------------|--|
| . | . | . |
| 100, | 6342 | /SKIP IF DEVICE 34 IS READY |
| 101, | 5100 | /JUMP .-1 |
| 102, | 5XXX | /ENTER SERVICE ROUTINE FOR /DEVICE 34 |
| . | . | . |
| . | . | . |
| . | . | . |

When the program reaches address 100, it executes an instruction skip with 6342. The skip occurs only if device 34 is ready when the IOT 6342 command is given. If device 34 is not ready, the flag signal disqualifies the skip gate, and the skip pulse does not occur. Therefore, the program continues to the next instruction which is a jump back to the skip instruction. In this example, the program stays in the waiting loop until the device is ready to transfer data, at which time the skip gate in the device is enabled and the skip pulse is sent to the computer IOS facility. When the skip occurs, the instruction in location 102 transfers program control to a subroutine to service device 34. This subroutine can load the AC with data and transfer it to device 34, or can load the AC from a register in device 34 and store it in some known core memory address.

ACCUMULATOR

The binary 1 output signal of each flip-flop of the AC, buffered by an inverter, is available at the interface connectors. These computer data output lines are bus connected to all peripheral equipment receiving programmed data output information from the PDP-8/S. A direct-set terminal on each flip-flop of the AC is connected to the interface connectors for bussing to all peripheral equipment supplying programmed data input to the PDP-8/S. A pulse that drives the direct-set terminal to ground causes setting of an AC flip-flop to the binary 1 state. Output and input connections to the accumulator appear in Figure 16.

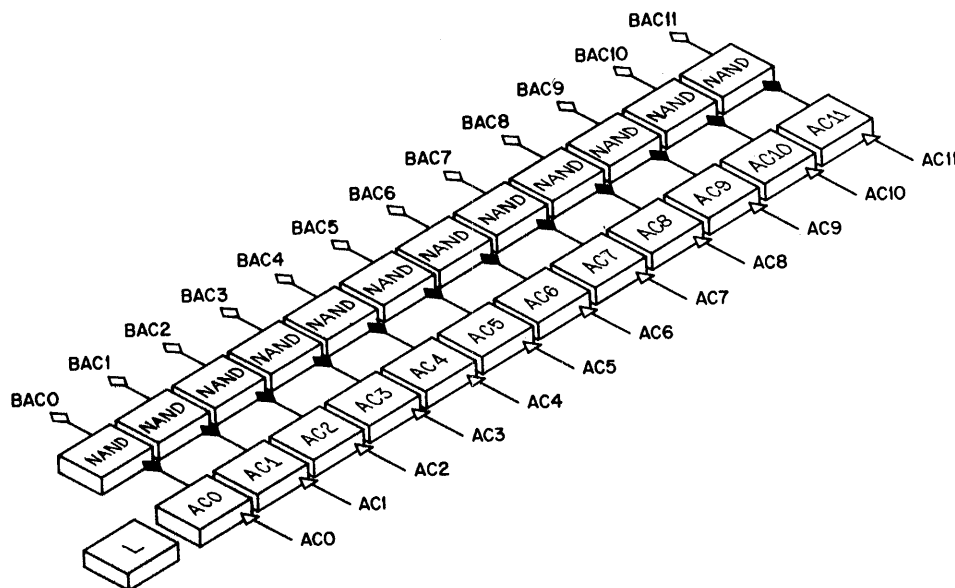


Figure 16 Accumulator Input and Output

Figure 16 illustrates the twelve bits of the accumulator and the link bit. The status of the link bit is not available to enter into transfers with peripheral equipment (unless it is rotated into the AC). An inverting driver continuously buffers the output signal from each AC flip-flop. These buffered accumulator (BAC) signals are available at the interface connectors.

INPUT DATA TRANSFERS

When ready to transfer data into the PDP-8/S accumulator, the device sets a flag connected to the IOS. The program senses the ready status of the flag and issues an IOT instruction to read the content of the external device buffer register into the AC. If the AC is not cleared before the transfer, the resultant word in the AC is the inclusive OR of the previous word in the AC and the word transferred from the device buffer register.

The illustration in Figure 17 shows that the accumulator has an input bus for each bit flip-flop. Setting a 1 into a particular bit of the accumulator necessitates grounding of the interface input bus by the standard DEC inverter. In the illustration, the 2-input AND

gates set various bits of the accumulator. In this case an IOT pulse is AND combined with the flip-flop state of the external device to conditionally set 1's into the accumulator. (The program must include a clear AC command prior to loading in this manner; otherwise an inclusive OR takes place between the previous content of the accumulator and the content of the data register being read.)

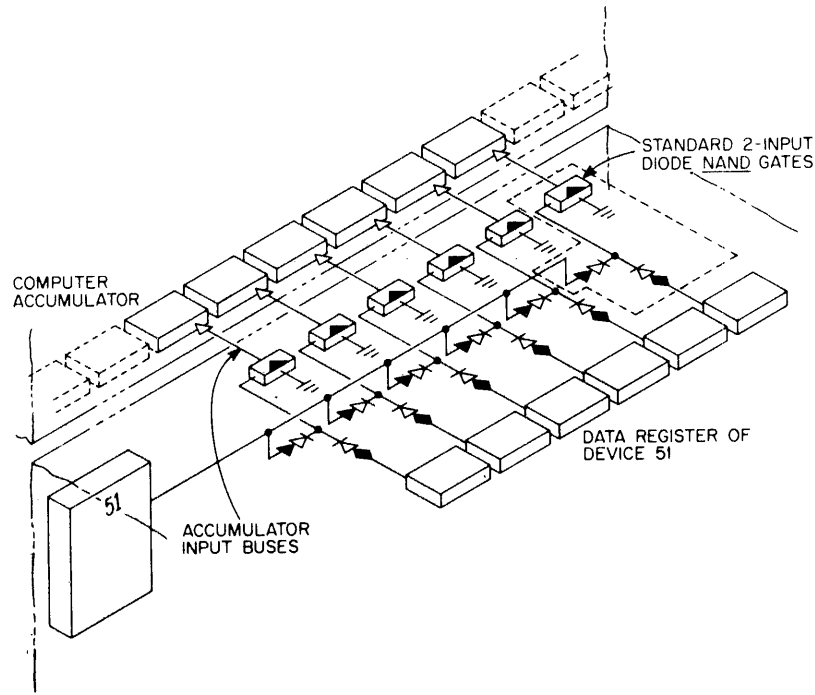


Figure 17 Loading Data into the Accumulator from an External Device

Following the transfer (possibly in the same instruction) the program can issue a command pulse to initiate further operation of the device and/or clear the device flag.

OUTPUT DATA TRANSFERS

The AC is loaded with a word (e.g., by a CLA TAD instruction sequence); then the IOT instruction is issued to transfer the word into the control or data register of the device by an IOT pulse (e.g., IOP 2), and operation of the device is initiated by another IOT pulse (e.g., IOP 4). The data word transferred in this manner can be a character to be operated upon, or can be a control word sampled by a status register to establish a control mode.

Since the BAC interface bus lines continually represent the status of the AC flip-flops, the receiving device can strobe them to sense the value in the accumulator. In Figure 30 a strobe pulse samples six bits of the accumulator to conditionally set an external 6-bit data register. Since this is not a jam transfer, it is necessary first to clear the external data register before setting 1's into it. The readin gates driving the external data register are part of the external device and are not supplied by the computer. The data register can contain any number of flip-flops up to a maximum of twelve. (If more than twelve flip-

flops are involved, two or more transfers must take place.) Obviously the clear pulse and the strobe pulse shown in Figure 30 must occur when the data to be placed in the external data register is held in the accumulator. These pulses therefore must be under computer control to effect synchronization with the operation or program of the computer.

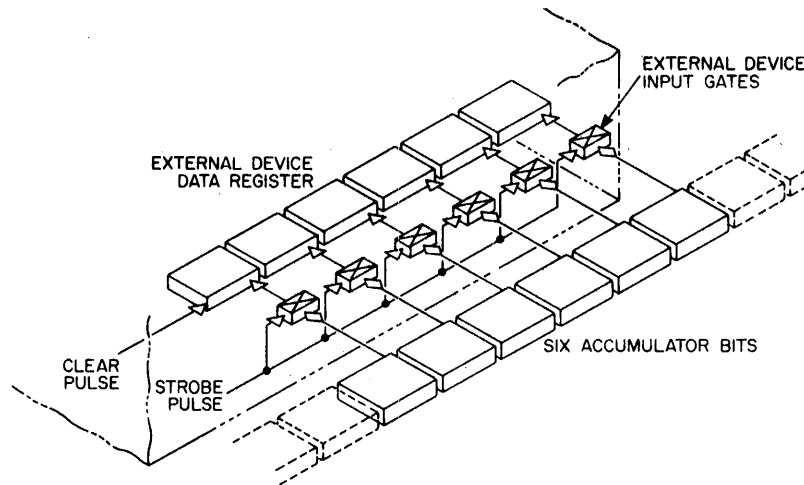


Figure 18 Loading a 6-Bit Word into an External Device from the Accumulator

Figure 19 illustrates the use of two of the pulses being gated by the device selected coded for "34." Pulse IOT 1 clears the data register and IOT 4 stromes the data from the accumulator into the data register. Note that the processor produces the IOP 1, IOP 2, and IOP 4 pulses and supplies them to all device selectors. The program-selected DS produces IOT 1, IOT 2, and IOT 4 pulses which initiate a transfer or effect some control. As indicated in Figure 19 this particular system adds two new microinstructions to the PDP-8/S repertoire. One generates a pulse to clear the data register of device number 34. The other microinstruction produces a pulse to load the data register of device number 34 with the content of the accumulator.

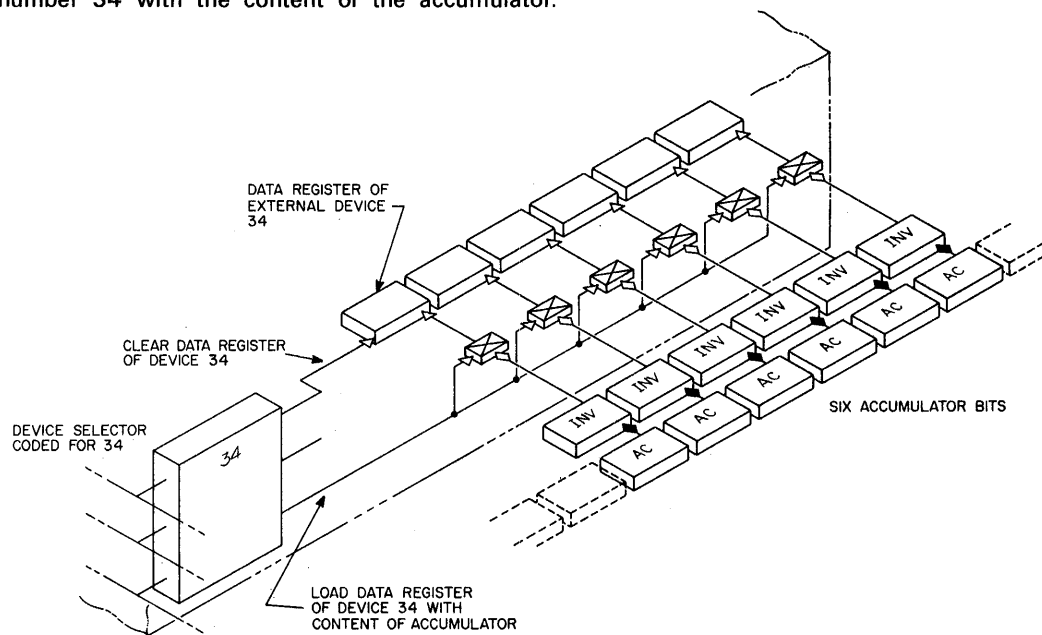


Figure 19 Use of a Device Selector for Activating and Controlling an External Device

PROGRAM INTERRUPT (PI)

When a large amount of computing is required, the program should initiate operation of an I/O device then continue the main program, rather than wait for the device to become ready to transfer data. The program interrupt facility, when enabled by the program, relieves the main program of the need for repeated flag checks by allowing the ready status of I/O device flags to automatically cause a program interrupt. When the program interrupt occurs, program control transfers to a subroutine that determines which device requested the interrupt and initiates an appropriate service routine.

In the example shown in Figure 20, a flag signal from a status flip-flop operates a standard inverter with no collector load. When the status flip-flop indicates the need for device service, the inverter drives the Program Interrupt Request bus to ground to request a program interrupt.

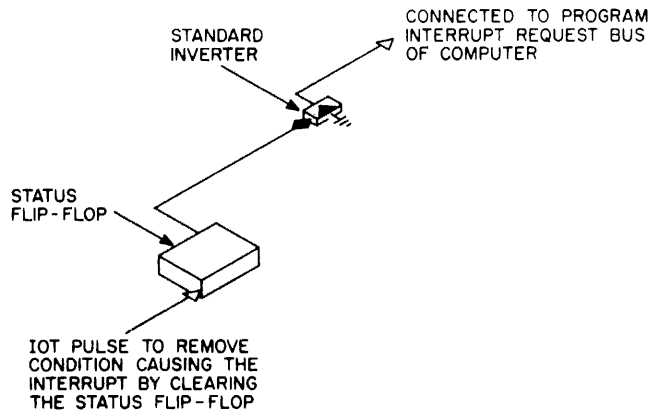


Figure 20 Program Interrupt Request Signal Origin

If only one device is connected to the PI facility, program control can be transferred directly to a routine that services the device when an interrupt occurs. This operation occurs as follows:

| <u>Tag</u> | <u>Address</u> | <u>Instruction</u> | <u>Remarks</u> |
|------------|----------------|--------------------|--|
| | 1000 | . | /MAIN PROGRAM |
| | 1001 | . | /MAIN PROGRAM CONTINUES |
| | 1002 | . | /INTERRUPT REQUEST OCCURS |
| | | | INTERRUPT OCCURS |
| | 0000 | | /PROGRAM COUNT (PC=1003) IS STORED IN 0000 |
| | 0001 | JMP SR | /ENTER SERVICE ROUTINE |
| SR | 2000 | . | /SERVICE SUBROUTINE FOR INTERRUPTING |
| | | . | /DEVICE AND SEQUENCE TO RESTORE |
| | 3001 | . | /AC, AND RESTORE L IF REQUIRED |
| | 3002 | ION | /TURN ON INTERRUPT |
| | 3003 | JMP I 0000 | /RETURN TO MAIN PROGRAM |
| | 1003 | . | /MAIN PROGRAM CONTINUES |
| | 1004 | . | |
| | | . | |
| | | . | |

In most PDP-8/S systems numerous devices are connected to the PI facility, so the routine beginning in core memory address 0001 must determine which device requested an interrupt. The interrupt routine determines the device requiring service by checking the flags of all equipment connected to the PI and transfers program control to a service routine for the first device encountered that has its flag in the state required to request a program interrupt. In other words, when program interrupt requests can originate in numerous devices, each device flag connected to the PI must also be connected to the IOS.

MULTIPLE USE OF IOS AND PI

In common practice, more than one device is connected to the PI facility. Therefore, since the computer receives a request that is the inclusive OR of requests from all devices connected to the PI, the IOS must identify the device making the request. When a program interrupt occurs, a routine is entered from address 0001 to sequentially check the status of each flag connected to the PI and to transfer program control to an appropriate service routine for the device whose flag is requesting a program interrupt. Figure 21 shows IOS and PI connections for three typical devices.

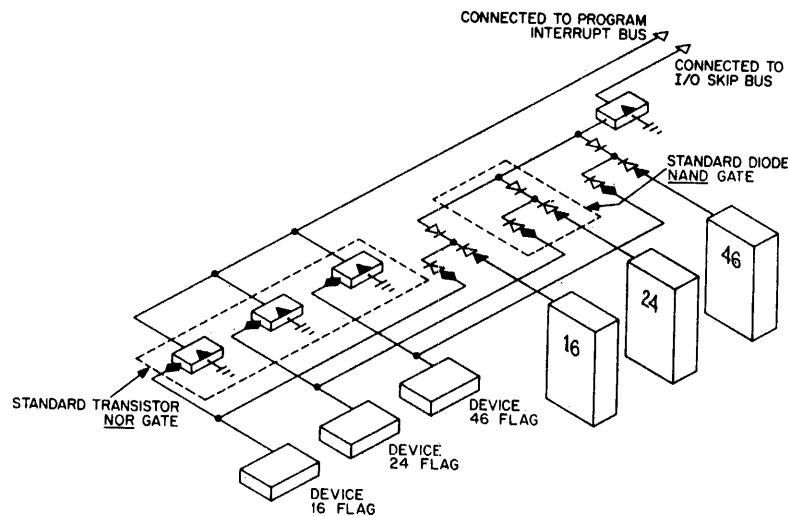


Figure 21 Multiple Inputs to IOS and PI Facilities

The following program example illustrates how the program interrupt routine determines the device requesting service:

| <u>Tag</u> | <u>Address</u> | <u>Instruction</u> | <u>Remarks</u> |
|------------|----------------|--------------------|---|
| | 1000 | . | /MAIN PROGRAM |
| | 1001 | . | /MAIN PROGRAM CONTINUES |
| | 1002 | . | /INTERRUPT REQUEST OCCURS |
| | | INTERRUPT OCCURS | |
| | 0000 | | /STORE PC (PC = 1003) |
| | 0001 | JMP FLG CK | /ENTER ROUTINE TO DETERMINE WHICH DEVICE /CAUSED INTERRUPT |
| FLG CK | | IOT 6341 | /SKIP IF DEVICE 34 IS REQUESTING |
| | | SKP | /NO - TEST NEXT DEVICE |
| | | JMP SR34 | /ENTER SERVICE ROUTINE 34 |
| | | IOT 6441 | /SKIP IF DEVICE 44 IS REQUESTING |
| | | SKP | /NO - TEST NEXT DEVICE |
| | | JMP SR44 | /ENTER SERVICE ROUTINE 44 |
| | | IOT 6541 | /SKIP IF DEVICE 54 IS REQUESTING |
| | | SKP | /NO - TEST NEXT DEVICE |
| | | JMP SR54 | /ENTER SERVICE ROUTINE 54 |
| | | . | |
| | | . | |
| | | . | |

Assume that the device that caused the interrupt is an input device (e.g., tape reader). The following example of a device service routine might apply:

| <u>Tag</u> | <u>Instruction</u> | <u>Remarks</u> |
|------------|--------------------|---|
| SR | DAC TEMP | /SAVE AC |
| | IOT XX | /TRANSFER DATA FROM DEVICE BUFFER TO AC |
| | DAC I 10 | /STORE IN MEMORY LIST |
| | ISZ COUNT | /CHECK FOR END |
| | SKP | /NOT END |
| | JMP END | /END. JUMP TO ROUTINE TO HANDLE END OF |
| | . | /LIST CONDITION |
| | . | |
| | . | |
| | | /RESTORE L AND EPC IF REQUIRED |
| | TAD TEMP | /RELOAD AC |
| | ION | /TURN ON INTERRUPT |
| | JMP I 0 | /RETURN TO PROGRAM |

If the device that caused the interrupt was essentially an output device (receiving data from computer), the IOT—then—DAC I 10 sequence might be replaced by a TAD I 10—then—IOT sequence.

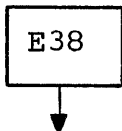
PDP-8/S PIN LOCATION LIST

E37



| NAME | FROM | PIN | MODULE TYPE/LOAD | Asser. |
|-------|------|-----|------------------|--------|
| | | A | | |
| | | B | | |
| | | C | | |
| IC 9 | A3J | D | R001/8ma | → |
| IC 10 | A3F | E | R001/8ma | → |
| | | F | | |
| IC 11 | A3D | H | R001/8ma | → |
| | | J | | |
| SKP | A9K | K | R107/11ma | ◇ |
| | | L | | |
| INT | A40E | M | R107/11ma | ◇ |
| | | N | | |
| CLA | A19R | P | R603 | → |
| | | R | | |
| | | S | | |
| | | T | | |
| | | U | | |
| | | V | | |

PDP-8/S PIN LOCATION LIST (cont.)












| NAME | FROM | PIN | MODULE TYPE/LOAD | Asser. |
|------|------|-----|------------------|--------|
| | | A | | |
| | | B | | |
| | | C | | |
| IC 0 | E3D | D | R001/8ma | → |
| IC 1 | E3F | E | R001/8ma | → |
| | | F | | |
| IC 2 | E3J | H | R001/8ma | → |
| | | J | | |
| IC 3 | E3L | K | R001/8ma | → |
| | | L | | |
| IC 4 | E3N | M | R001/8ma | → |
| | | N | | |
| IC 5 | A3T | P | R001/8ma | → |
| | | R | | |
| IC 6 | A3R | S | R001/8ma | → |
| IC 7 | A3N | T | R001/8ma | → |
| | | U | | |
| IC 8 | A3L | V | R001/8ma | → |

PDP-8/S PIN LOCATION LIST (cont.)

E39



| NAME | FROM | PIN | MODULE TYPE/LOAD | Asser. |
|-----------|------|-----|------------------|---|
| | | A | | |
| | | B | | |
| | | C | | |
| BAC 0 (1) | D5F | D | R107/18ma |  |
| BAC 1 (1) | D5J | E | R107/18ma |  |
| | | F | | |
| BAC 2 (1) | D5L | H | R107/18ma |  |
| | | J | | |
| BAC 3 (1) | D5N | K | R107/18ma |  |
| | | L | | |
| BAC 4 (1) | D5R | M | R107/18ma |  |
| | | N | | |
| BAC 5 (1) | D5T | P | R107/18ma |  |
| | | R | | |
| BAC 6 (1) | C5F | S | R107/18ma |  |
| BAC 7 (1) | C5J | T | R107/18ma |  |
| | | U | | |
| BAC 8 (1) | C5L | V | R107/18ma |  |

PDP-8/S PIN LOCATION LIST (cont.)

E40



| NAME | FROM | PIN | MODULE TYPE/LOAD | Asser. |
|-----------|------|-----|------------------|--------|
| | | A | | |
| | | B | | |
| | | C | | |
| | | D | | |
| | | E | | |
| | | F | | |
| | | H | | |
| | | J | | |
| BMB 3 (1) | B37F | K | R107/18ma | —◆ |
| | | L | | |
| BMB 3 (0) | B37J | M | R107/18ma | —◆ |
| | | N | | |
| BMB 4 (1) | B37L | P | R107/18ma | —◆ |
| | | R | | |
| BMB 4 (0) | B37N | S | R107/18ma | —◆ |
| BMB 5 (1) | B37R | T | R107/18ma | —◆ |
| | | U | | |
| BMB 5 (0) | B37T | V | R107/18ma | —◆ |

PDP-8/S PIN LOCATION LIST (cont.)

D40



| NAME | FROM | PIN | MODULE TYPE/LOAD | Asser. |
|-----------|------|-----|------------------|--------|
| | | A | | |
| | | B | | |
| | | C | | |
| BMB 6 (1) | A17D | D | R107/18ma | |
| BMB 6 (0) | A17F | E | R107/18ma | |
| | | F | | |
| BMB 7 (1) | A17R | H | R107/18ma | |
| | | J | | |
| BMB 7 (0) | C05T | K | R107/18ma | |
| | | L | | |
| BMB 8 (1) | A17T | M | R107/18ma | |
| | | N | | |
| BMB 8 (0) | C13L | P | R107/18ma | |
| | | R | | |
| | | S | | |
| | | T | | |
| | | U | | |
| | | V | | |

PDP-8/S PIN LOCATION LIST (cont.)

B39



| NAME | FROM | PIN | MODULE TYPE/LOAD | Asser. |
|---------------|------|-----|------------------|--------|
| | | A | | |
| | | B | | |
| | | C | | |
| BAC 9 (1) | C5N | D | R107/18ma | |
| BAC 10 (1) | C5R | E | R107/18ma | |
| | | F | | |
| BAC 11 (1) | C5D | H | R107/18ma | |
| | | J | | |
| IOP 1 | A40K | K | R107/18ma | |
| | | L | | |
| IOP 2 | A40L | M | R107/18ma | |
| | | N | | |
| IOP 4 | A40N | P | R107/18ma | |
| | | R | | |
| | | S | | |
| | | T | | |
| | | U | | |
| B Power Clear | A17L | V | R107/18ma | |

CHAPTER 19

INSTALLATION PLANNING

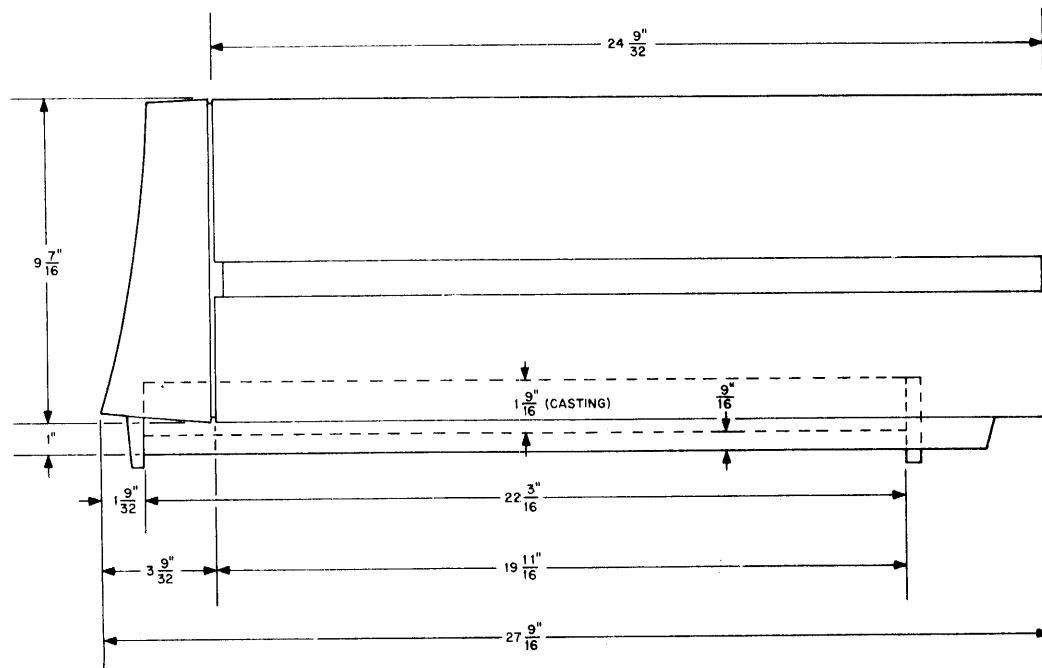
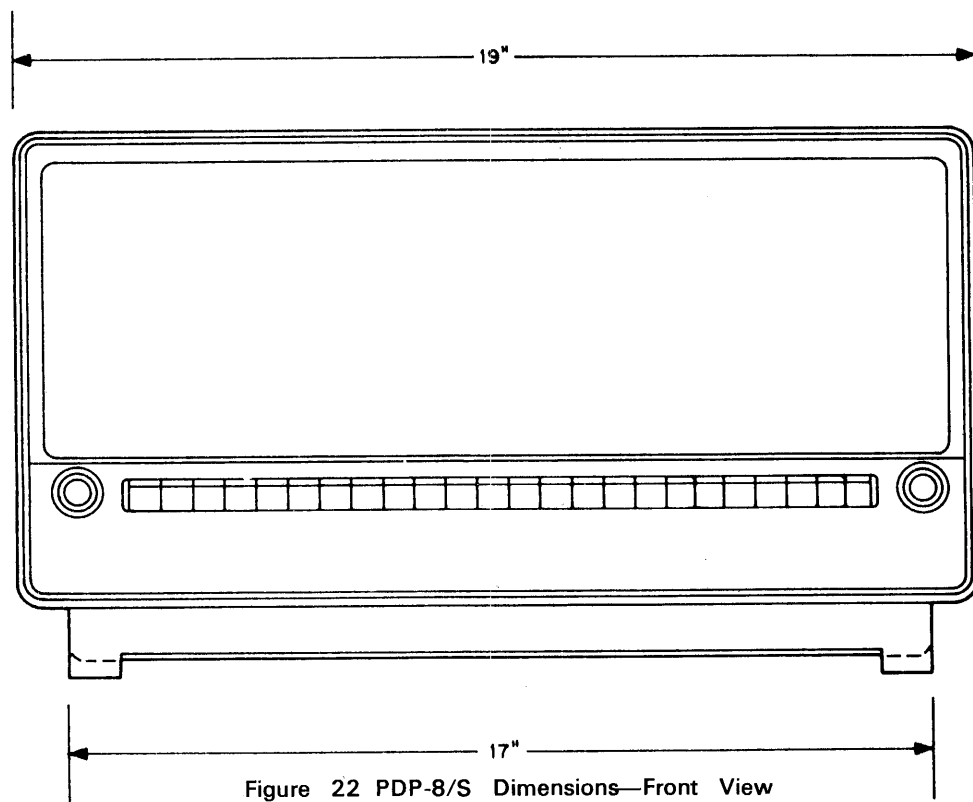


Figure 23 PDP-8/S—Side View

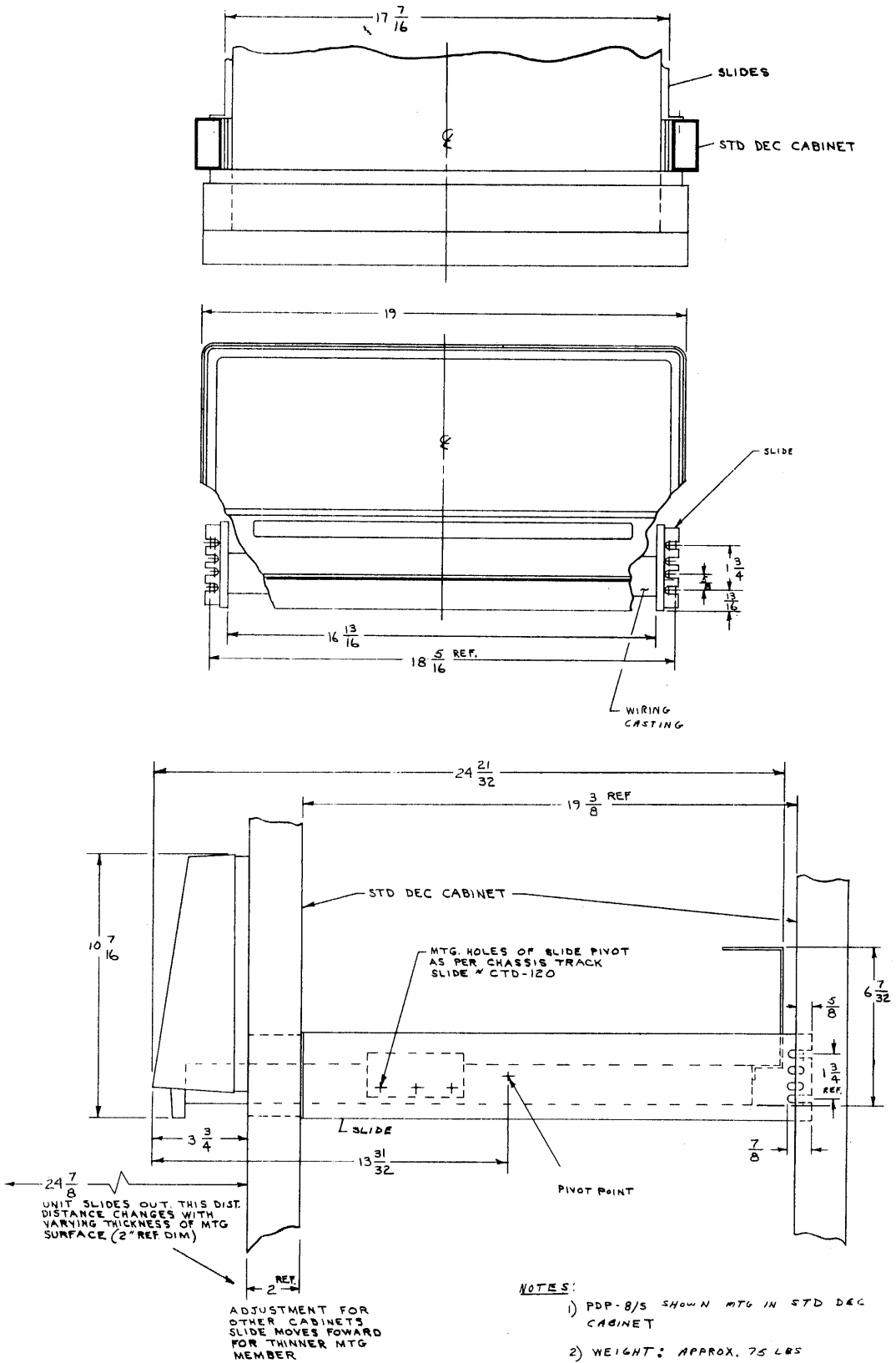


Figure 23A PDP-8/S Cabinet Model

POWER REQUIREMENTS

A source of 115v (\pm 17v), 60-cps (\pm 0.5 cps), single-phase power capable of supplying at least 10 amp must be provided to operate a standard PDP-8/S. To allow connection to the power cable of the computer, this source should be provided with a Hubbell 3-terminal, except for the basic table top PDP-8/S grounded-neutral flush receptacle (or its equivalent). A table-mounted PDP-8/S is provided with a 15-amp power plug; a rack-mounted PDP-8/S has a 20 amp twist-lock plug; and systems that draw more than 20 amps use a 30-amp twist-lock plug.

APPENDIX 1
INSTRUCTIONS

MEMORY REFERENCE INSTRUCTIONS

| Mnemonic Symbol | Operation Code | Direct Addr. | | Indirect Addr. | |
|-----------------|----------------|----------------|-----------------------|----------------|-----------------------|
| | | States Entered | Execution Time (μsec) | States Entered | Execution Time (μsec) |
| AND Y | 0 | F,X,ΛE | 36 | F,D,X,ΛE | 54 |
| TAD Y | 1 | F,X,ΛE | 36 | F,D,X,ΛE | 54 |
| ISZ Y | 2 | F,X,E | 54 | F,D,X,E | 72 |
| DCA Y | 3 | F,X,E | 46 | F,D,X,E | 64 |
| JMS Y | 4 | F,X,E | 46 | F,D,X,E | 64 |
| JMP Y | 5 | F,X,ΛE | 28 | F,D,X,ΛE | 46 |

BASIC IOT MICROINSTRUCTIONS

| Mnemonic | Octal | Operation |
|---|-------|--|
| PROGRAM INTERRUPT | | |
| ION | 6001 | Turn interrupt on and enable the computer to respond to an interrupt request. When this instruction is given, the computer executes the next instruction, then enables the interrupt. The additional instruction allows exit from the interrupt subroutine before allowing another interrupt to occur. |
| IOF | 6002 | Turn interrupt off i.e. disable the interrupt. |
| HIGH SPEED PERFORATED TAPE READER AND CONTROL TYPE PC02 | | |
| RSF | 6011 | Skip if reader flag is a 1. |
| RRB | 6012 | Read the content of the reader buffer and clear the reader flag. (This instruction does not clear the AC.) RB V AC 4-11 => AC 4-11 |
| RFC | 6014 | Clear reader flag and reader buffer, fetch one character from tape and load it into the reader buffer, and set the reader flag when done. |
| HIGH SPEED PERFORATED TAPE PUNCH AND CONTROL TYPE PC03 | | |
| PSF | 6021 | Skip if punch flag is a 1. |
| PCF | 6022 | Clear punch flag and punch buffer. |
| PPC | 6024 | Load the punch buffer from bits 4 through 11 of the AC and punch the character. This instruction does not clear the punch flag or punch buffer.) AC 4-11 V PB => PB |
| PLS | 6026 | Clear the punch flag, clear the punch buffer, load the punch buffer from the content of bits 4 through 11 of the accumulator, punch the character, and set the punch flag to 1 when done. |
| TELETYPE KEYBOARD/READER | | |
| KSF | 6031 | Skip if keyboard flag is a 1. |
| KCC | 6032 | Clear AC and clear keyboard flag. |
| KRS | 6034 | Read keyboard buffer static. (This is a static command in that neither the AC nor the keyboard flag is cleared.) TTI V AC 4-11 => AC 4-11 |
| KRB | 6036 | Clear AC, clear keyboard flag, and read the content of the keyboard buffer into the content of AC 4-11. |

BASIC IOT MICROINSTRUCTIONS (continued)

| Mnemonic | Octal | Operation |
|---|-------|--|
| TELETYPE TELEPRINTER/PUNCH | | |
| TSF | 6041 | Skip if teleprinter flag is a 1. |
| TCF | 6042 | Clear teleprinter flag. |
| TPC | 6044 | Load the TTO from the content of AC 4-11 and print and/or punch the character. |
| TLS | 6046 | Load the TTO from the content of AC 4-11, clear the teleprinter flag, and print and /or punch the character. |
| OSCILLOSCOPE DISPLAY TYPE 34D AND PRECISION CRT DISPLAY TYPE 30N | | |
| DCX | 6051 | Clear X coordinate buffer. |
| DXL | 6053 | Clear and load X coordinate buffer. AC 2-11 => XB |
| DIX | 6054 | Intensify the point defined by the content of the X and Y coordinate buffers. |
| DXS | 6057 | Executes the combined functions of DXL followed by DIX. |
| DCY | 6061 | Clear Y coordinate buffer. |
| DYL | 6063 | Clear and load Y coordinate buffer. AC 2-11 => YB |
| DIY | 6064 | Intensify the point defined by the content of the X and Y coordinate buffers. |
| DYS | 6067 | Executes the combined functions of DYL followed by DIY. |
| OSCILLOSCOPE DISPLAY TYPE 34D | | |
| DSB | 6075 | Set minimum brightness. |
| DSB | 6076 | Set medium brightness. |
| DSB | 6077 | Set maximum brightness. |
| LIGHT PEN TYPE 370 | | |
| DSF | 6071 | Skip if display flag is a 1. |
| DCF | 6072 | Clear the display flag. |
| MEMORY PARITY | | |
| SMP | 6101 | Skip if memory parity error flag = 0. |
| CMP | 6104 | Clear memory parity error flag. |

BASIC IOT MICROINSTRUCTIONS (continued)

| Mnemonic | Octal | Operation |
|--|-------|--|
| INCREMENTAL PLOTTER AND CONTROL TYPE 350B | | |
| PLSF | 6501 | Skip if plotter flag is a 1. |
| PLCF | 6502 | Clear plotter flag. |
| PLPU | 6504 | Plotter pen up. Raise pen off of paper. |
| PLPR | 6511 | Plotter pen right. |
| PLDU | 6512 | Plotter drum (paper) upward. |
| PLDD | 6514 | Plotter drum (paper) downward. |
| PLPL | 6521 | Plotter pen left. |
| PLUD | 6522 | Plotter drum (paper) upward. (Same as 6512.) |
| PLPD | 6524 | Plotter pen down. Lower pen on to paper. |
| GENERAL PURPOSE CONVERTER TYPE 138E AND MULTIPLEXER CONTROL TYPE 139E | | |
| ADSF | 6531 | Skip if A/D converter flag is a 1. |
| ADCV | 6532 | Clear A/D converter flag and convert input voltage to a digital number, flag will set to 1 at end of conversion. Number of bits in converted number determined by switch setting, 11 bits maximum. |
| ADRB | 6534 | Read A/D converter buffer into AC, left justified, and clear flag. |
| ADCC | 6541 | Clear multiplexer channel address register. |
| ADSC | 6542 | Set up multiplexer channel as per AC 6-11. Maximum of 64 single ended or 32 differential input channels. |
| ADIC | 6544 | Index multiplexer channel address (present address + 1). Upon reaching address limit, increment will cause channel 00 to be selected. |
| CARD READER AND CONTROL TYPE CRO1C | | |
| RCSF | 6631 | Skip if card reader data ready flag is a 1. |
| RCRA | 6632 | The alphanumeric code for the column is read into AC6-11, and the data ready flag is cleared. |
| RCRB | 6634 | The binary data in a card column is transferred into AC0-11, and the data ready flag is cleared. |
| RCSP | 6671 | Skip if card reader card done flag is a 1. |
| RCSE | 6672 | Clear the card done flag, select the card reader and start card motion towards the read station, and skip if the reader-not-ready flag is a 1. |
| RCRD | 6674 | Clear card done flag. |

GROUP 1 OPERATE MICROINSTRUCTIONS

| Mnemonic Symbol | Octal Code | Operation |
|--------------------|---------------|---|
| NOP | 7000 | No operation. Causes a 1.5 μ sec program delay. |
| IAC | 7001 | Increment AC. The content of the AC is incremented by one in two's complement arithmetic. |
| RAL | 7004 | Rotate AC and L left. The content of the AC and the L are rotated left one place. |
| RTL | 7006 | Rotate two places to the left. Equivalent to two successive RAL operations. |
| RAR | 7010 | Rotate AC and L right. The content of the AC and L are rotated right one place. |
| RTR | 7012 | Rotate two places to the right. Equivalent to two successive RAR operations. |
| CML | 7020 | Complement L. |
| CMA | 7040 | Complement AC. The content of the AC is set to the one's complement of its current content. |
| CIA | 7041 | Complement and increment accumulator. Used to form two's complement. |
| CLL | 7100 | Clear L. |
| CLL RAL | 7104 | Shift positive number one left. |
| CLL RTL | 7106 | Clear link, rotate two left. |
| CLL RAR | 7110 | Shift positive number one right. |
| CLL RTR | 7112 | Clear link, rotate two right. |
| STL | 7120 | Set link. The L is set to contain a binary 1. |
| CLA | 7200 | Clear AC. To be used alone or in OPR 1 combinations. |
| CLA IAC | 7201 | Set AC = 1 |
| GLK | 7204 | Get link. Transfer L into AC 11. |
| CLA CLL | 7300 | Clear AC and L. |
| STA | 7240 | Set AC = - 1. Each bit of the AC is set to contain a 1. |

GROUP 2 OPERATE MICROINSTRUCTIONS

| Mnemonic Symbol | Octal Code | Operation |
|--------------------|---------------|--|
| HLT | 7402 | Halt. Stops the program after completion of the cycle in process. If this instruction is combined with others in the OPR 2 group the other operations are completed before the end of the cycle. |
| OSR | 7404 | OR with switch register. The OR function is performed between the content of the SR and the content of the AC, with the result left in the AC. |
| SKP | 7410 | Skip, unconditional. The next instruction is skipped. |
| SNL | 7420 | Skip if $L \neq 0$. |
| SZL | 7430 | Skip if $L = 0$. |
| SZA | 7440 | Skip if $AC = 0$. |
| SNA | 7450 | Skip if $AC \neq 0$. |
| SZA SNL | 7460 | Skip if $AC = 0$, or $L = 1$, or both. |
| SNA SZL | 7470 | Skip if $AC \neq 0$ and $L = 0$. |
| SMA | 7500 | Skip on minus AC. If the content of the AC is a negative number, the next instruction is skipped. |
| SPA | 7510 | Skip on positive AC. If the content of the AC is a positive number, the next instruction is skipped. |
| SMA SNL | 7520 | Skip if $AC < 0$, or $L = 1$, or both. |
| SPA SZL | 7530 | Skip if $AC \geq 0$ and if $L = 0$. |
| SMA SZA | 7540 | Skip if $AC \geq 0$. |
| SPA SNA | 7550 | Skip if $AC > 0$. |
| CLA | 7600 | Clear AC. To be used alone or in OPR 2 combinations. |
| LAS | 7604 | Load AC with SR. |
| SZA CLA | 7640 | Skip if $AC = 0$, then clear AC. |
| SNA CLA | 7650 | Skip if $AC \neq 0$, then clear AC. |
| SMA CLA | 7700 | Skip if $AC < 0$, then clear AC. |
| SPA CLA | 7710 | Skip if $AC \geq 0$, then clear AC. |

APPENDIX 2 CODES

MODEL 33 ASR/KSR TELETYPE CODE (ASCII) IN OCTAL FORM

| Character | 8-Bit Code (in octal) | Character | 8-Bit Code (in octal) |
|-----------|--------------------------|-----------------|--------------------------|
| A | 301 | ! | 241 |
| B | 302 | " | 242 |
| C | 303 | # | 243 |
| D | 304 | \$ | 244 |
| E | 30 | % | 245 |
| F | 306 | & | 246 |
| G | 307 | ' | 247 |
| H | 310 | (| 250 |
| I | 311 |) | 251 |
| J | 312 | * | 252 |
| K | 313 | + | 253 |
| L | 314 | , | 254 |
| M | 315 | - | 255 |
| N | 316 | . | 256 |
| O | 317 | / | 257 |
| P | 320 | : | 272 |
| Q | 321 | ; | 273 |
| R | 322 | < | 274 |
| S | 323 | = | 275 |
| T | 324 | > | 276 |
| U | 325 | ? | 277 |
| V | 326 | @ | 300 |
| W | 327 | [| 333 |
| X | 330 | \ | 334 |
| Y | 331 |] | 335 |
| Z | 332 | ^ | 336 |
| | | ◀ | 337 |
| 0 | 260 | | |
| 1 | 261 | Leader/Trailer | 200 |
| 2 | 262 | Line-Feed | 212 |
| 3 | 263 | Carriage-Return | 215 |
| 4 | 264 | Space | 240 |
| 5 | 265 | Rub-out | 377 |
| 6 | 266 | Blank | 000 |
| 7 | 267 | | |
| 8 | 270 | | |
| 9 | 271 | | |

MODEL 33 ASR/KSR TELETYPE CODE (ASCII) IN BINARY FORM

1 = HOLE PUNCHED = MARK
0 = NO HOLE PUNCHED = SPACE

MOST SIGNIFICANT BIT
LEAST SIGNIFICANT BIT
8 7 6 5 4 3 2 1

| | @ | SPACE | NULL/IDLE | 8 7 6 5 4 3 2 1 |
|---------|---|-------|----------------------|-----------------|
| | A | ! | START OF MESSAGE | 0 0 0 0 0 |
| | B | " | END OF ADDRESS | 0 0 0 0 1 |
| | C | # | END OF MESSAGE | 0 0 0 1 0 |
| | D | \$ | END OF TRANSMISSION | 0 0 0 1 1 |
| | E | % | WHO ARE YOU | 0 0 1 0 0 |
| | F | & | ARE YOU | 0 0 1 0 1 |
| | G | ' | BELL | 0 0 1 1 0 |
| | H | (| BELL | 0 0 1 1 1 |
| | I | (| FORMAT EFFECTOR | 0 1 0 0 0 |
| | J |) | HORIZONTAL TAB | 0 1 0 0 1 |
| | K | * | LINE FEED | 0 1 0 1 0 |
| | L | + | VERTICAL TAB | 0 1 0 1 1 |
| | M | , | FORM FEED | 0 1 1 0 0 |
| | N | — | CARRIAGE RETURN | 0 1 1 0 1 |
| | O | . | SHIFT OUT | 0 1 1 1 0 |
| | P | / | SHIFT IN | 0 1 1 1 1 |
| | Q | 0 | DCO | 1 0 0 0 0 |
| | R | 1 | READER ON | 1 0 0 0 1 |
| | S | 2 | TAPE (AUX ON) | 1 0 0 1 0 |
| | T | 3 | READER OFF | 1 0 0 1 1 |
| | U | 4 | (AUX OFF) | 1 0 1 0 0 |
| | V | 5 | ERROR | 1 0 1 0 1 |
| | W | 6 | SYNCHRONOUS IDLE | 1 0 1 1 0 |
| | X | 7 | LOGICAL END OF MEDIA | 1 0 1 1 1 |
| | Y | 8 | S 0 | 1 1 0 0 0 |
| | Z | 9 | S 1 | 1 1 0 0 1 |
| | [| : | S 2 | 1 1 0 1 0 |
| | \ | ; | S 3 | 1 1 0 1 1 |
| |] | < | S 4 | 1 1 1 0 0 |
| | ^ | = | S 5 | 1 1 1 0 1 |
| | _ | > | S 6 | 1 1 1 1 0 |
| RUB OUT | ← | ? | S 7 | 1 1 1 1 1 |

| | |
|-------|------|
| 1 0 0 | SAME |
| 1 0 1 | SAME |
| 1 1 0 | SAME |
| 1 1 1 | SAME |

CARD READER CODE

| Card Code | | | Character | Card Code | | | |
|-----------|------|---------------|-----------|-----------------|------|---------------|---|
| Zone | Num. | Internal Code | | Zone | Num. | Internal Code | |
| — | — | 01 0000 | Blank | 11 | 0 | 10 1010 | ↑ |
| 12 | 8-3 | 11 1011 | . | 11 | 1 | 10 0001 | J |
| 12 | 8-4 | 11 1100 |) | 11 | 2 | 10 0010 | K |
| 12 | 8-5 | 11 1101 |] | 11 | 3 | 10 0011 | L |
| 12 | 8-6 | 11 1110 | < | 11 | 4 | 10 0100 | M |
| 12 | 8-7 | 11 1111 | ← | 11 | 5 | 10 0101 | N |
| 12 | — | 11 0000 | + | 11 | 6 | 10 0110 | O |
| 11 | 8-3 | 10 1011 | \$ | 11 | 7 | 10 0111 | P |
| 11 | 8-4 | 10 1100 | * | 11 | 8 | 10 1000 | Q |
| 11 | 8-5 | 10 1101 | [| 11 | 9 | 10 1001 | R |
| 11 | 8-6 | 10 1110 | > | 0 | 8-2 | 01 1010 | ; |
| 11 | 8-7 | 10 1111 | & | 0 | 2 | 01 0010 | S |
| 11 | — | 10 0000 | — | 0 | 3 | 01 0011 | T |
| 0 | 1 | 01 0001 | / | 0 | 4 | 01 0100 | U |
| 0 | 8-3 | 01 1011 | , | 0 | 5 | 01 0101 | V |
| 0 | 8-4 | 01 1100 | (| 0 | 6 | 01 0110 | W |
| 0 | 8-5 | 01 1101 | " | 0 | 7 | 01 0111 | X |
| 0 | 8-6 | 01 1110 | # | 0 | 8 | 01 1000 | Y |
| 0 | 8-7 | 01 1111 | % | 0 | 9 | 01 1001 | Z |
| — | 8-3 | 00 1011 | = | — | 0 | 00 1010 | 0 |
| — | 8-4 | 00 1100 | @ | — | 1 | 00 0001 | 1 |
| — | 8-5 | 00 1101 | ↑ | — | 2 | 00 0010 | 2 |
| — | 8-6 | 00 1110 | , | — | 3 | 00 0011 | 3 |
| — | 8-7 | 00 1111 | ~ | — | 4 | 00 0100 | 4 |
| 12 | 0 | 11 1010 | ? | — | 5 | 00 0101 | 5 |
| 12 | 1 | 11 0001 | A | — | 6 | 00 0110 | 6 |
| 12 | 2 | 11 0010 | B | — | 7 | 00 0111 | 7 |
| 12 | 3 | 11 0011 | C | — | 8 | 00 1000 | 8 |
| 12 | 4 | 11 0100 | D | — | 9 | 00 1001 | 9 |
| 12 | 5 | 11 0101 | E | All other codes | | 00 0000 | ← |
| 12 | 6 | 11 0110 | F | | | | |
| 12 | 7 | 11 0111 | G | | | | |
| 12 | 8 | 11 1000 | H | | | | |
| 12 | 9 | 11 1001 | I | | | | |

AUTOMATIC LINE PRINTER CODE

| Character (ASCII) | 6-Bit Code (in octal) | Character (ASCII) | 6-Bit Code (in octal) |
|----------------------|--------------------------|----------------------|--------------------------|
| @ | 0 | □ | 40 |
| A | 1 | ! | 41 |
| B | 2 | " | 42 |
| C | 3 | # | 43 |
| D | 4 | \$ | 44 |
| E | 5 | % | 45 |
| F | 6 | & | 46 |
| G | 7 | ' | 47 |
| H | 10 | (| 50 |
| I | 11 |) | 51 |
| J | 12 | * | 52 |
| K | 13 | + | 53 |
| L | 14 | , | 54 |
| M | 15 | - | 55 |
| N | 16 | . | 56 |
| O | 17 | / | 57 |
| P | 20 | ∅ | 60 |
| Q | 21 | 1 | 61 |
| R | 22 | 2 | 62 |
| S | 23 | 3 | 63 |
| T | 24 | 4 | 64 |
| U | 25 | 5 | 65 |
| V | 26 | 6 | 66 |
| W | 27 | 7 | 67 |
| X | 30 | 8 | 70 |
| Y | 31 | 9 | 71 |
| Z | 32 | : | 72 |
| [| 33 | ; | 73 |
| \ | 34 | < | 74 |
|] | 35 | = | 75 |
| ↑ | 36 | > | 76 |
| ← | 37 | ? | 77 |

APPENDIX 3 SCALES OF NOTATION

2^x IN DECIMAL

| x | 2 ^x | x | 2 ^x | x | 2 ^x |
|-------|---------------------|------|---------------------|-----|---------------------|
| 0.001 | 1.00069 33874 62581 | 0.01 | 1.00695 55500 56719 | 0.1 | 1.07177 34625 36293 |
| 0.002 | 1.00138 72557 11335 | 0.02 | 1.01395 94797 90029 | 0.2 | 1.14869 83549 97035 |
| 0.003 | 1.00208 16050 79633 | 0.03 | 1.02101 21257 07193 | 0.3 | 1.23114 44133 44916 |
| 0.004 | 1.00277 64359 01078 | 0.04 | 1.02811 38266 56067 | 0.4 | 1.31950 79107 72894 |
| 0.005 | 1.00347 17485 09503 | 0.05 | 1.03526 49238 41377 | 0.5 | 1.41421 35623 73095 |
| 0.006 | 1.00416 75432 38973 | 0.06 | 1.04246 57608 41121 | 0.6 | 1.51571 65665 10398 |
| 0.007 | 1.00486 38204 23785 | 0.07 | 1.04971 66836 23067 | 0.7 | 1.62450 47927 12471 |
| 0.008 | 1.00556 05803 98468 | 0.08 | 1.05701 80405 61380 | 0.8 | 1.74110 11265 92248 |
| 0.009 | 1.00625 78234 97782 | 0.09 | 1.06437 01824 53360 | 0.9 | 1.86606 59830 73615 |

10^{±n} IN OCTAL

| 10 ⁿ | n | 10 ⁻ⁿ | 10 ⁿ | n | 10 ⁻ⁿ |
|-----------------|---|------------------------------|----------------------------|----|------------------------------|
| 1 | 0 | 1.000 000 000 000 00 | 112 402 762 000 | 10 | 0.000 000 000 006 676 337 66 |
| 12 | 1 | 0.063 146 314 631 463 146 31 | 1 351 035 564 000 | 11 | 0.000 000 000 000 537 657 77 |
| 144 | 2 | 0.005 075 341 217 270 243 66 | 16 432 451 210 000 | 12 | 0.000 000 000 000 043 136 32 |
| 1 750 | 3 | 0.000 406 111 564 570 651 77 | 221 411 634 520 000 | 13 | 0.000 000 000 000 003 411 35 |
| 23 420 | 4 | 0.000 032 155 613 530 704 15 | 2 657 142 036 440 000 | 14 | 0.000 000 000 000 000 264 11 |
| 303 240 | 5 | 0.000 002 476 132 610 706 64 | 34 327 724 461 500 000 | 15 | 0.000 000 000 000 000 022 01 |
| 3 641 100 | 6 | 0.000 000 206 157 364 055 37 | 434 157 115 760 200 000 | 16 | 0.000 000 000 000 000 001 63 |
| 46 113 200 | 7 | 0.000 000 015 327 745 152 75 | 5 432 127 413 542 400 000 | 17 | 0.000 000 000 000 000 000 14 |
| 575 360 400 | 8 | 0.000 000 001 257 143 561 06 | 67 405 553 164 731 000 000 | 18 | 0.000 000 000 000 000 000 01 |
| 7 346 545 000 | 9 | 0.000 000 000 104 560 276 41 | | | |

n log₁₀ 2, n log₂ 10 IN DECIMAL

| n | n log ₁₀ 2 | n log ₂ 10 | n | n log ₁₀ 2 | n log ₂ 10 |
|---|-----------------------|-----------------------|----|-----------------------|-----------------------|
| 1 | 0.30102 99957 | 3.32192 80949 | 6 | 1.80617 99740 | 19.93156 85693 |
| 2 | 0.60205 99913 | 6.64385 61898 | 7 | 2.10720 99696 | 23.25349 66642 |
| 3 | 0.90308 99870 | 9.96578 42847 | 8 | 2.40823 99653 | 26.57542 47591 |
| 4 | 1.20411 99827 | 13.28771 23795 | 9 | 2.70926 99610 | 29.89735 28540 |
| 5 | 1.50514 99783 | 16.60964 04744 | 10 | 3.01029 99566 | 33.21928 09489 |

ADDITION AND MULTIPLICATION TABLES

Addition

Multiplication

Binary Scale

$$0 + 1 = \begin{array}{r} 0 + 0 = 0 \\ 1 + 0 = 1 \\ 1 + 1 = 10 \end{array}$$

$$0 \times 1 = \begin{array}{r} 0 \times 0 = 0 \\ 1 \times 0 = 0 \\ 1 \times 1 = 1 \end{array}$$

Octal Scale

| | | | | | | | |
|---|----|----|----|----|----|----|----|
| 0 | 01 | 02 | 03 | 04 | 05 | 06 | 07 |
| 1 | 02 | 03 | 04 | 05 | 06 | 07 | 10 |
| 2 | 03 | 04 | 05 | 06 | 07 | 10 | 11 |
| 3 | 04 | 05 | 06 | 07 | 10 | 11 | 12 |
| 4 | 05 | 06 | 07 | 10 | 11 | 12 | 13 |
| 5 | 06 | 07 | 10 | 11 | 12 | 13 | 14 |
| 6 | 07 | 10 | 11 | 12 | 13 | 14 | 15 |
| 7 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |

| | | | | | | |
|---|----|----|----|----|----|----|
| 1 | 02 | 03 | 04 | 05 | 06 | 07 |
| 2 | 04 | 06 | 10 | 12 | 14 | 16 |
| 3 | 06 | 11 | 14 | 17 | 22 | 25 |
| 4 | 10 | 14 | 20 | 24 | 30 | 34 |
| 5 | 12 | 17 | 24 | 31 | 36 | 43 |
| 6 | 14 | 22 | 30 | 36 | 44 | 52 |
| 7 | 16 | 25 | 34 | 43 | 52 | 61 |

MATHEMATICAL CONSTANTS IN OCTAL SCALE

| | | |
|-----------------------------------|------------------------------------|---------------------------------------|
| $\pi = 3.11037 \ 552421_8$ | $e = 2.55760 \ 521305_8$ | $\gamma = 0.44742 \ 147707_8$ |
| $\pi^{-1} = 0.24276 \ 301556_8$ | $e^{-1} = 0.27426 \ 530661_8$ | $\ln \gamma = -0.43127 \ 233602_8$ |
| $\sqrt{\pi} = 1.61337 \ 611067_8$ | $\sqrt{e} = 1.51411 \ 230704_8$ | $\log_2 \gamma = -0.62573 \ 030645_8$ |
| $\ln \pi = 1.11206 \ 404435_8$ | $\log_{10} e = 0.33626 \ 754251_8$ | $\sqrt{2} = 1.32404 \ 746320_8$ |
| $\log_2 \pi = 1.51544 \ 163223_8$ | $\log_2 e = 1.34252 \ 166245_8$ | $\ln 2 = 0.54271 \ 027760_8$ |
| $\sqrt{10} = 3.12305 \ 407267_8$ | $\log_2 10 = 3.24464 \ 741136_8$ | $\ln 10 = 2.23273 \ 067355_8$ |

APPENDIX 4 POWERS OF TWO

| 2^n | n | 2^{-n} |
|------------------------|----|--|
| 1 | 0 | 1.0 |
| 2 | 1 | 0.5 |
| 4 | 2 | 0.25 |
| 8 | 3 | 0.125 |
| 16 | 4 | 0.0625 |
| 32 | 5 | 0.03125 |
| 64 | 6 | 0.015625 |
| 128 | 7 | 0.0078125 |
| 256 | 8 | 0.00390625 |
| 512 | 9 | 0.001953125 |
| 1024 | 10 | 0.0009765625 |
| 2048 | 11 | 0.00048828125 |
| 4096 | 12 | 0.000244140625 |
| 8192 | 13 | 0.0001220703125 |
| 16384 | 14 | 0.00006103515625 |
| 32768 | 15 | 0.000030517578125 |
| 65536 | 16 | 0.0000152587890625 |
| 131072 | 17 | 0.00000762939453125 |
| 262144 | 18 | 0.000003814697265625 |
| 524288 | 19 | 0.0000019073486328125 |
| 1048576 | 20 | 0.00000095367431640625 |
| 2097152 | 21 | 0.000000476837158203125 |
| 4194304 | 22 | 0.0000002384185791015625 |
| 8388608 | 23 | 0.00000011920928955078125 |
| 16777216 | 24 | 0.000000059604644775390625 |
| 33554432 | 25 | 0.0000000298023223876953125 |
| 67108864 | 26 | 0.00000001490116119384765625 |
| 134217728 | 27 | 0.000000007450580596923828125 |
| 268435456 | 28 | 0.0000000037252902984619140625 |
| 536870912 | 29 | 0.00000000186264514923095703125 |
| 1073741824 | 30 | 0.000000000931322574615478515625 |
| 2147483648 | 31 | 0.0000000004656612873077392578125 |
| 4294967296 | 32 | 0.00000000023283064365386962890625 |
| 8589934592 | 33 | 0.00000000011641532182693481453125 |
| 17179869184 | 34 | 0.0000000000582076609134674072265625 |
| 34359738368 | 35 | 0.00000000002910383045673370361328125 |
| 68719476736 | 36 | 0.000000000014551915228366851806640625 |
| 137438953472 | 37 | 0.0000000000072759576141834259033203125 |
| 274877906944 | 38 | 0.00000000000363797880709171295166015625 |
| 549755813888 | 39 | 0.000000000001818989403545856475830078125 |
| 1099511627776 | 40 | 0.0000000000009094947017729282379150390625 |
| 2199023255552 | 41 | 0.00000000000045474735088646411895751953125 |
| 4398046511104 | 42 | 0.000000000000227373675443232059478759765625 |
| 8796093022208 | 43 | 0.0000000000001136868377216160297393798828125 |
| 17592186044416 | 44 | 0.00000000000005684341886080801486968994140625 |
| 35184372088832 | 45 | 0.0000000000000284217094304040074348449703125 |
| 70368744177664 | 46 | 0.0000000000000142108547152020037174224853515625 |
| 140737488355328 | 47 | 0.00000000000000710542735760100185871124267578125 |
| 281474976710656 | 48 | 0.00000000000000355271367880050092935621337890625 |
| 562949953421312 | 49 | 0.0000000000000017763568394002504646778106689453125 |
| 1125899906842624 | 50 | 0.00000000000000088817841970012523233890533447265625 |
| 2251799813685248 | 51 | 0.000000000000000444089209850062616169452667236328125 |
| 4503599627370496 | 52 | 0.0000000000000002220446049250313080847263336181640625 |
| 9007199254740992 | 53 | 0.00000000000000011102230246251565404236316680908203125 |
| 18014398509481984 | 54 | 0.000000000000000055511151231257827021181583404541015625 |
| 36028797018963968 | 55 | 0.0000000000000000277555756156289135105907917022705078125 |
| 72057594037927936 | 56 | 0.00000000000000001387778780781445675529539585113525390625 |
| 144115188075855772 | 57 | 0.000000000000000006938893903907228377647697925567626953125 |
| 288230376151711744 | 58 | 0.0000000000000000034694469519536141888238489627838134765625 |
| 576460752303423488 | 59 | 0.00000000000000000173472347597680709441192448139190673828125 |
| 1152921504606846976 | 60 | 0.000000000000000000867361737988403547205962240695953369140625 |
| 2305843009213693952 | 61 | 0.0000000000000000004336808689942017736029811203479766845703125 |
| 4611686018427387904 | 62 | 0.00000000000000000021684043449710088680149056017398834228515625 |
| 9223372036854775808 | 63 | 0.000000000000000000108420217248550443400745280086994171142578125 |
| 18446744073709551616 | 64 | 0.0000000000000000000542101086242752217003726400434970855712890625 |
| 36893488147419103232 | 65 | 0.0000000000000000000271050543121376108501863202174854278564453125 |
| 73786976294838206464 | 66 | 0.00000000000000000001355252715606880542509316001087427139282265625 |
| 147573952589676412928 | 67 | 0.0000000000000000000067762635780344027125465800054371356964111328125 |
| 295147905179352825856 | 68 | 0.00000000000000000000338813178901720135627329000271856784820556640625 |
| 590295810358705651712 | 69 | 0.000000000000000000001694065894508600678136645001359283924102783203125 |
| 1180591620717411303424 | 70 | 0.0000000000000000000008470329472543003390683225006796419620513916015625 |
| 2361183241434822606848 | 71 | 0.00000000000000000000042351647362715016953416125033982098102569580078125 |
| 4722366482869645213696 | 72 | 0.000000000000000000000211758236813575084767080625169910490512847900390625 |

OCTAL-DECIMAL INTEGER CONVERSION TABLE (continued)

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|------|------|------|------|------|------|------|------|------|
| 2000 | 1024 | 1025 | 1026 | 1027 | 1028 | 1029 | 1030 | 1031 |
| 2010 | 1032 | 1033 | 1034 | 1035 | 1036 | 1037 | 1038 | 1039 |
| 2020 | 1040 | 1041 | 1042 | 1043 | 1044 | 1045 | 1046 | 1047 |
| 2030 | 1048 | 1049 | 1050 | 1051 | 1052 | 1053 | 1054 | 1055 |
| 2040 | 1056 | 1057 | 1058 | 1059 | 1060 | 1061 | 1062 | 1063 |
| 2050 | 1064 | 1065 | 1066 | 1067 | 1068 | 1069 | 1070 | 1071 |
| 2060 | 1072 | 1073 | 1074 | 1075 | 1076 | 1077 | 1078 | 1079 |
| 2070 | 1080 | 1081 | 1082 | 1083 | 1084 | 1085 | 1086 | 1087 |
| 2100 | 1088 | 1089 | 1090 | 1091 | 1092 | 1093 | 1094 | 1095 |
| 2110 | 1096 | 1097 | 1098 | 1099 | 1100 | 1101 | 1102 | 1103 |
| 2120 | 1104 | 1105 | 1106 | 1107 | 1108 | 1109 | 1110 | 1111 |
| 2130 | 1112 | 1113 | 1114 | 1115 | 1116 | 1117 | 1118 | 1119 |
| 2140 | 1120 | 1121 | 1122 | 1123 | 1124 | 1125 | 1126 | 1127 |
| 2150 | 1128 | 1129 | 1130 | 1131 | 1132 | 1133 | 1134 | 1135 |
| 2160 | 1136 | 1137 | 1138 | 1139 | 1140 | 1141 | 1142 | 1143 |
| 2170 | 1144 | 1145 | 1146 | 1147 | 1148 | 1149 | 1150 | 1151 |
| 2200 | 1152 | 1153 | 1154 | 1155 | 1156 | 1157 | 1158 | 1159 |
| 2210 | 1160 | 1161 | 1162 | 1163 | 1164 | 1165 | 1166 | 1167 |
| 2220 | 1168 | 1169 | 1170 | 1171 | 1172 | 1173 | 1174 | 1175 |
| 2230 | 1176 | 1177 | 1178 | 1179 | 1180 | 1181 | 1182 | 1183 |
| 2240 | 1184 | 1185 | 1186 | 1187 | 1188 | 1189 | 1190 | 1191 |
| 2250 | 1192 | 1193 | 1194 | 1195 | 1196 | 1197 | 1198 | 1199 |
| 2260 | 1200 | 1201 | 1202 | 1203 | 1204 | 1205 | 1206 | 1207 |
| 2270 | 1208 | 1209 | 1210 | 1211 | 1212 | 1213 | 1214 | 1215 |
| 2300 | 1216 | 1217 | 1218 | 1219 | 1220 | 1221 | 1222 | 1223 |
| 2310 | 1224 | 1225 | 1226 | 1227 | 1228 | 1229 | 1230 | 1231 |
| 2320 | 1232 | 1233 | 1234 | 1235 | 1236 | 1237 | 1238 | 1239 |
| 2330 | 1240 | 1241 | 1242 | 1243 | 1244 | 1245 | 1246 | 1247 |
| 2340 | 1248 | 1249 | 1250 | 1251 | 1252 | 1253 | 1254 | 1255 |
| 2350 | 1256 | 1257 | 1258 | 1259 | 1260 | 1261 | 1262 | 1263 |
| 2360 | 1264 | 1265 | 1266 | 1267 | 1268 | 1269 | 1270 | 1271 |
| 2370 | 1272 | 1273 | 1274 | 1275 | 1276 | 1277 | 1278 | 1279 |

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|------|------|------|------|------|------|------|------|------|
| 3000 | 1536 | 1537 | 1538 | 1539 | 1540 | 1541 | 1542 | 1543 |
| 3010 | 1544 | 1545 | 1546 | 1547 | 1548 | 1549 | 1550 | 1551 |
| 3020 | 1552 | 1553 | 1554 | 1555 | 1556 | 1557 | 1558 | 1559 |
| 3030 | 1560 | 1561 | 1562 | 1563 | 1564 | 1565 | 1566 | 1567 |
| 3040 | 1568 | 1569 | 1570 | 1571 | 1572 | 1573 | 1574 | 1575 |
| 3050 | 1576 | 1577 | 1578 | 1579 | 1580 | 1581 | 1582 | 1583 |
| 3060 | 1584 | 1585 | 1586 | 1587 | 1588 | 1589 | 1590 | 1591 |
| 3070 | 1592 | 1593 | 1594 | 1595 | 1596 | 1597 | 1598 | 1599 |
| 3100 | 1600 | 1601 | 1602 | 1603 | 1604 | 1605 | 1606 | 1607 |
| 3110 | 1608 | 1609 | 1610 | 1611 | 1612 | 1613 | 1614 | 1615 |
| 3120 | 1616 | 1617 | 1618 | 1619 | 1620 | 1621 | 1622 | 1623 |
| 3130 | 1624 | 1625 | 1626 | 1627 | 1628 | 1629 | 1630 | 1631 |
| 3140 | 1632 | 1633 | 1634 | 1635 | 1636 | 1637 | 1638 | 1639 |
| 3150 | 1640 | 1641 | 1642 | 1643 | 1644 | 1645 | 1646 | 1647 |
| 3160 | 1648 | 1649 | 1650 | 1651 | 1652 | 1653 | 1654 | 1655 |
| 3170 | 1656 | 1657 | 1658 | 1659 | 1660 | 1661 | 1662 | 1663 |
| 3200 | 1664 | 1665 | 1666 | 1667 | 1668 | 1669 | 1670 | 1671 |
| 3210 | 1672 | 1673 | 1674 | 1675 | 1676 | 1677 | 1678 | 1679 |
| 3220 | 1680 | 1681 | 1682 | 1683 | 1684 | 1685 | 1686 | 1687 |
| 3230 | 1688 | 1689 | 1690 | 1691 | 1692 | 1693 | 1694 | 1695 |
| 3240 | 1696 | 1697 | 1698 | 1699 | 1700 | 1701 | 1702 | 1703 |
| 3250 | 1704 | 1705 | 1706 | 1707 | 1708 | 1709 | 1710 | 1711 |
| 3260 | 1712 | 1713 | 1714 | 1715 | 1716 | 1717 | 1718 | 1719 |
| 3270 | 1720 | 1721 | 1722 | 1723 | 1724 | 1725 | 1726 | 1727 |
| 3300 | 1728 | 1729 | 1730 | 1731 | 1732 | 1733 | 1734 | 1735 |
| 3310 | 1736 | 1737 | 1738 | 1739 | 1740 | 1741 | 1742 | 1743 |
| 3320 | 1744 | 1745 | 1746 | 1747 | 1748 | 1749 | 1750 | 1751 |
| 3330 | 1752 | 1753 | 1754 | 1755 | 1756 | 1757 | 1758 | 1759 |
| 3340 | 1760 | 1761 | 1762 | 1763 | 1764 | 1765 | 1766 | 1767 |
| 3350 | 1768 | 1769 | 1770 | 1771 | 1772 | 1773 | 1774 | 1775 |
| 3360 | 1776 | 1777 | 1778 | 1779 | 1780 | 1781 | 1782 | 1783 |
| 3370 | 1784 | 1785 | 1786 | 1787 | 1788 | 1789 | 1790 | 1791 |

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|------|------|------|------|------|------|------|------|------|
| 2400 | 1280 | 1281 | 1282 | 1283 | 1284 | 1285 | 1286 | 1287 |
| 2410 | 1288 | 1289 | 1290 | 1291 | 1292 | 1293 | 1294 | 1295 |
| 2420 | 1296 | 1297 | 1298 | 1299 | 1300 | 1301 | 1302 | 1303 |
| 2430 | 1304 | 1305 | 1306 | 1307 | 1308 | 1309 | 1310 | 1311 |
| 2440 | 1312 | 1313 | 1314 | 1315 | 1316 | 1317 | 1318 | 1319 |
| 2450 | 1320 | 1321 | 1322 | 1323 | 1324 | 1325 | 1326 | 1327 |
| 2460 | 1328 | 1329 | 1330 | 1331 | 1332 | 1333 | 1334 | 1335 |
| 2470 | 1336 | 1337 | 1338 | 1339 | 1340 | 1341 | 1342 | 1343 |
| 2500 | 1344 | 1345 | 1346 | 1347 | 1348 | 1349 | 1350 | 1351 |
| 2510 | 1352 | 1353 | 1354 | 1355 | 1356 | 1357 | 1358 | 1359 |
| 2520 | 1360 | 1361 | 1362 | 1363 | 1364 | 1365 | 1366 | 1367 |
| 2530 | 1368 | 1369 | 1370 | 1371 | 1372 | 1373 | 1374 | 1375 |
| 2540 | 1376 | 1377 | 1378 | 1379 | 1380 | 1381 | 1382 | 1383 |
| 2550 | 1384 | 1385 | 1386 | 1387 | 1388 | 1389 | 1390 | 1391 |
| 2560 | 1392 | 1393 | 1394 | 1395 | 1396 | 1397 | 1398 | 1399 |
| 2570 | 1400 | 1401 | 1402 | 1403 | 1404 | 1405 | 1406 | 1407 |
| 2600 | 1408 | 1409 | 1410 | 1411 | 1412 | 1413 | 1414 | 1415 |
| 2610 | 1416 | 1417 | 1418 | 1419 | 1420 | 1421 | 1422 | 1423 |
| 2620 | 1424 | 1425 | 1426 | 1427 | 1428 | 1429 | 1430 | 1431 |
| 2630 | 1432 | 1433 | 1434 | 1435 | 1436 | 1437 | 1438 | 1439 |
| 2640 | 1440 | 1441 | 1442 | 1443 | 1444 | 1445 | 1446 | 1447 |
| 2650 | 1448 | 1449 | 1450 | 1451 | 1452 | 1453 | 1454 | 1455 |
| 2660 | 1456 | 1457 | 1458 | 1459 | 1460 | 1461 | 1462 | 1463 |
| 2670 | 1464 | 1465 | 1466 | 1467 | 1468 | 1469 | 1470 | 1471 |
| 2700 | 1472 | 1473 | 1474 | 1475 | 1476 | 1477 | 1478 | 1479 |
| 2710 | 1480 | 1481 | 1482 | 1483 | 1484 | 1485 | 1486 | 1487 |
| 2720 | 1488 | 1489 | 1490 | 1491 | 1492 | 1493 | 1494 | 1495 |
| 2730 | 1496 | 1497 | 1498 | 1499 | 1500 | 1501 | 1502 | 1503 |
| 2740 | 1504 | 1505 | 1506 | 1507 | 1508 | 1509 | 1510 | 1511 |
| 2750 | 1512 | 1513 | 1514 | 1515 | 1516 | 1517 | 1518 | 1519 |
| 2760 | 1520 | 1521 | 1522 | 1523 | 1524 | 1525 | 1526 | 1527 |
| 2770 | 1528 | 1529 | 1530 | 1531 | 1532 | 1533 | 1534 | 1535 |

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|------|------|------|------|------|------|------|------|------|
| 3400 | 1792 | 1793 | 1794 | 1795 | 1796 | 1797 | 1798 | 1799 |
| 3410 | 1800 | 1801 | 1802 | 1803 | 1804 | 1805 | 1806 | 1807 |
| 3420 | 1808 | 1809 | 1810 | 1811 | 1812 | 1813 | 1814 | 1815 |
| 3430 | 1816 | 1817 | 1818 | 1819 | 1820 | 1821 | 1822 | 1823 |
| 3440 | 1824 | 1825 | 1826 | 1827 | 1828 | 1829 | 1830 | 1831 |
| 3450 | 1832 | 1833 | 1834 | 1835 | 1836 | 1837 | 1838 | 1839 |
| 3460 | 1840 | 1841 | 1842 | 1843 | 1844 | 1845 | 1846 | 1847 |
| 3470 | 1848 | 1849 | 1850 | 1851 | 1852 | 1853 | 1854 | 1855 |
| 3500 | 1856 | 1857 | 1858 | 1859 | 1860 | 1861 | 1862 | 1863 |
| 3510 | 1864 | 1865 | 1866 | 1867 | 1868 | 1869 | 1870 | 1871 |
| 3520 | 1872 | 1873 | 1874 | 1875 | 1876 | 1877 | 1878 | 1879 |
| 3530 | 1880 | 1881 | 1882 | 1883 | 1884 | 1885 | 1886 | 1887 |
| 3540 | 1888 | 1889 | 1890 | 1891 | 1892 | 1893 | 1894 | 1895 |
| 3550 | 1896 | 1897 | 1898 | 1899 | 1900 | 1901 | 1902 | 1903 |
| 3560 | 1904 | 1905 | 1906 | 1907 | 1908 | 1909 | 1910 | 1911 |
| 3570 | 1912 | 1913 | 1914 | 1915 | 1916 | 1917 | 1918 | 1919 |
| 3600 | 1920 | 1921 | 1922 | 1923 | 1924 | 1925 | 1926 | 1927 |
| 3610 | 1928 | 1929 | 1930 | 1931 | 1932 | 1933 | 1934 | 1935 |
| 3620 | 1936 | 1937 | 1938 | 1939 | 1940 | 1941 | 1942 | 1943 |
| 3630 | 1944 | 1945 | 1946 | 1947 | 1948 | 1949 | 1950 | 1951 |
| 3640 | 1952 | 1953 | 1954 | 1955 | 1956 | 1957 | 1958 | 1959 |
| 3650 | 1960 | 1961 | 1962 | 1963 | 1964 | 1965 | 1966 | 1967 |
| 3660 | 1968 | 1969 | 1970 | 1971 | 1972 | 1973 | 1974 | 1975 |
| 3670 | 1976 | 1977 | 1978 | 1979 | 1980 | 1981 | 1982 | 1983 |
| 3700 | 1984 | 1985 | 1986 | 1987 | 1988 | 1989 | 1990 | 1991 |
| 3710 | 1992 | 1993 | 1994 | 1995 | 1996 | 1997 | 1998 | 1999 |
| 3720 | 2000 | 2001 | 2002 | 2003 | 2004 | 2005 | 2006 | 2007 |
| 3730 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 | 2014 | 2015 |
| 3740 | 2016 | 2017 | 2018 | 2019 | 2020 | 2021 | 2022 | 2023 |
| 3750 | 2024 | 2025 | 2026 | 2027 | 2028 | 2029 | 2030 | 2031 |
| 3760 | 2032 | 2033 | 2034 | 2035 | 2036 | 2037 | 2038 | 2039 |
| 3770 | 2040 | 2041 | 2042 | 2043 | 2044 | 2045 | 2046 | 2047 |

2000 to 2777 (Octal) | 1024 to 1535 (Decimal)

Octal Decimal
10000 - 4096
20000 - 8192
30000 - 12288
40000 - 16384
50000 - 20480
60000 - 24576
70000 - 28672

3000 to 3777 (Octal) | 1536 to 2047 (Decimal)

OCTAL-DECIMAL INTEGER CONVERSION TABLE (continued)

| | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|-------------------------------|---------------------------------|------|------|------|------|------|------|------|------|------|
| 4000 to 4777 (Octal) | 2048 to 2559 (Decimal) | 4000 | 2048 | 2049 | 2050 | 2051 | 2052 | 2053 | 2054 | 2055 |
| | | 4010 | 2056 | 2057 | 2058 | 2059 | 2060 | 2061 | 2062 | 2063 |
| | | 4020 | 2064 | 2065 | 2066 | 2067 | 2068 | 2069 | 2070 | 2071 |
| | | 4030 | 2072 | 2073 | 2074 | 2075 | 2076 | 2077 | 2078 | 2079 |
| | | 4040 | 2080 | 2081 | 2082 | 2083 | 2084 | 2085 | 2086 | 2087 |
| | | 4050 | 2088 | 2089 | 2090 | 2091 | 2092 | 2093 | 2094 | 2095 |
| | | 4060 | 2096 | 2097 | 2098 | 2099 | 2100 | 2101 | 2102 | 2103 |
| | | 4070 | 2104 | 2105 | 2106 | 2107 | 2108 | 2109 | 2110 | 2111 |
| | | 4100 | 2112 | 2113 | 2114 | 2115 | 2116 | 2117 | 2118 | 2119 |
| | | 4110 | 2120 | 2121 | 2122 | 2123 | 2124 | 2125 | 2126 | 2127 |
| | | 4120 | 2128 | 2129 | 2130 | 2131 | 2132 | 2133 | 2134 | 2135 |
| | | 4130 | 2136 | 2137 | 2138 | 2139 | 2140 | 2141 | 2142 | 2143 |
| | | 4140 | 2144 | 2145 | 2146 | 2147 | 2148 | 2149 | 2150 | 2151 |
| | | 4150 | 2152 | 2153 | 2154 | 2155 | 2156 | 2157 | 2158 | 2159 |
| | | 4160 | 2160 | 2161 | 2162 | 2163 | 2164 | 2165 | 2166 | 2167 |
| | | 4170 | 2168 | 2169 | 2170 | 2171 | 2172 | 2173 | 2174 | 2175 |
| 4200 | 2176 | 2177 | 2178 | 2179 | 2180 | 2181 | 2182 | 2183 | | |
| 4210 | 2184 | 2185 | 2186 | 2187 | 2188 | 2189 | 2190 | 2191 | | |
| 4220 | 2192 | 2193 | 2194 | 2195 | 2196 | 2197 | 2198 | 2199 | | |
| 4230 | 2200 | 2201 | 2202 | 2203 | 2204 | 2205 | 2206 | 2207 | | |
| 4240 | 2208 | 2209 | 2210 | 2211 | 2212 | 2213 | 2214 | 2215 | | |
| 4250 | 2216 | 2217 | 2218 | 2219 | 2220 | 2221 | 2222 | 2223 | | |
| 4260 | 2224 | 2225 | 2226 | 2227 | 2228 | 2229 | 2230 | 2231 | | |
| 4270 | 2232 | 2233 | 2234 | 2235 | 2236 | 2237 | 2238 | 2239 | | |
| 4300 | 2240 | 2241 | 2242 | 2243 | 2244 | 2245 | 2246 | 2247 | | |
| 4310 | 2248 | 2249 | 2250 | 2251 | 2252 | 2253 | 2254 | 2255 | | |
| 4320 | 2256 | 2257 | 2258 | 2259 | 2260 | 2261 | 2262 | 2263 | | |
| 4330 | 2264 | 2265 | 2266 | 2267 | 2268 | 2269 | 2270 | 2271 | | |
| 4340 | 2272 | 2273 | 2274 | 2275 | 2276 | 2277 | 2278 | 2279 | | |
| 4350 | 2280 | 2281 | 2282 | 2283 | 2284 | 2285 | 2286 | 2287 | | |
| 4360 | 2288 | 2289 | 2290 | 2291 | 2292 | 2293 | 2294 | 2295 | | |
| 4370 | 2296 | 2297 | 2298 | 2299 | 2300 | 2301 | 2302 | 2303 | | |
| 4400 | 2304 | 2305 | 2306 | 2307 | 2308 | 2309 | 2310 | 2311 | | |
| 4410 | 2312 | 2313 | 2314 | 2315 | 2316 | 2317 | 2318 | 2319 | | |
| 4420 | 2320 | 2321 | 2322 | 2323 | 2324 | 2325 | 2326 | 2327 | | |
| 4430 | 2328 | 2329 | 2330 | 2331 | 2332 | 2333 | 2334 | 2335 | | |
| 4440 | 2336 | 2337 | 2338 | 2339 | 2340 | 2341 | 2342 | 2343 | | |
| 4450 | 2344 | 2345 | 2346 | 2347 | 2348 | 2349 | 2350 | 2351 | | |
| 4460 | 2352 | 2353 | 2354 | 2355 | 2356 | 2357 | 2358 | 2359 | | |
| 4470 | 2360 | 2361 | 2362 | 2363 | 2364 | 2365 | 2366 | 2367 | | |
| 4500 | 2368 | 2369 | 2370 | 2371 | 2372 | 2373 | 2374 | 2375 | | |
| 4510 | 2376 | 2377 | 2378 | 2379 | 2380 | 2381 | 2382 | 2383 | | |
| 4520 | 2384 | 2385 | 2386 | 2387 | 2388 | 2389 | 2390 | 2391 | | |
| 4530 | 2392 | 2393 | 2394 | 2395 | 2396 | 2397 | 2398 | 2399 | | |
| 4540 | 2400 | 2401 | 2402 | 2403 | 2404 | 2405 | 2406 | 2407 | | |
| 4550 | 2408 | 2409 | 2410 | 2411 | 2412 | 2413 | 2414 | 2415 | | |
| 4560 | 2416 | 2417 | 2418 | 2419 | 2420 | 2421 | 2422 | 2423 | | |
| 4570 | 2424 | 2425 | 2426 | 2427 | 2428 | 2429 | 2430 | 2431 | | |
| 4600 | 2432 | 2433 | 2434 | 2435 | 2436 | 2437 | 2438 | 2439 | | |
| 4610 | 2440 | 2441 | 2442 | 2443 | 2444 | 2445 | 2446 | 2447 | | |
| 4620 | 2448 | 2449 | 2450 | 2451 | 2452 | 2453 | 2454 | 2455 | | |
| 4630 | 2456 | 2457 | 2458 | 2459 | 2460 | 2461 | 2462 | 2463 | | |
| 4640 | 2464 | 2465 | 2466 | 2467 | 2468 | 2469 | 2470 | 2471 | | |
| 4650 | 2472 | 2473 | 2474 | 2475 | 2476 | 2477 | 2478 | 2479 | | |
| 4660 | 2480 | 2481 | 2482 | 2483 | 2484 | 2485 | 2486 | 2487 | | |
| 4670 | 2488 | 2489 | 2490 | 2491 | 2492 | 2493 | 2494 | 2495 | | |
| 4700 | 2496 | 2497 | 2498 | 2499 | 2500 | 2501 | 2502 | 2503 | | |
| 4710 | 2504 | 2505 | 2506 | 2507 | 2508 | 2509 | 2510 | 2511 | | |
| 4720 | 2512 | 2513 | 2514 | 2515 | 2516 | 2517 | 2518 | 2519 | | |
| 4730 | 2520 | 2521 | 2522 | 2523 | 2524 | 2525 | 2526 | 2527 | | |
| 4740 | 2528 | 2529 | 2530 | 2531 | 2532 | 2533 | 2534 | 2535 | | |
| 4750 | 2536 | 2537 | 2538 | 2539 | 2540 | 2541 | 2542 | 2543 | | |
| 4760 | 2544 | 2545 | 2546 | 2547 | 2548 | 2549 | 2550 | 2551 | | |
| 4770 | 2552 | 2553 | 2554 | 2555 | 2556 | 2557 | 2558 | 2559 | | |
| 5000 | 2560 | 2561 | 2562 | 2563 | 2564 | 2565 | 2566 | 2567 | | |
| 5010 | 2568 | 2569 | 2570 | 2571 | 2572 | 2573 | 2574 | 2575 | | |
| 5020 | 2576 | 2577 | 2578 | 2579 | 2580 | 2581 | 2582 | 2583 | | |
| 5030 | 2584 | 2585 | 2586 | 2587 | 2588 | 2589 | 2590 | 2591 | | |
| 5040 | 2592 | 2593 | 2594 | 2595 | 2596 | 2597 | 2598 | 2599 | | |
| 5050 | 2600 | 2601 | 2602 | 2603 | 2604 | 2605 | 2606 | 2607 | | |
| 5060 | 2608 | 2609 | 2610 | 2611 | 2612 | 2613 | 2614 | 2615 | | |
| 5070 | 2616 | 2617 | 2618 | 2619 | 2620 | 2621 | 2622 | 2623 | | |
| 5100 | 2624 | 2625 | 2626 | 2627 | 2628 | 2629 | 2630 | 2631 | | |
| 5110 | 2632 | 2633 | 2634 | 2635 | 2636 | 2637 | 2638 | 2639 | | |
| 5120 | 2640 | 2641 | 2642 | 2643 | 2644 | 2645 | 2646 | 2647 | | |
| 5130 | 2648 | 2649 | 2650 | 2651 | 2652 | 2653 | 2654 | 2655 | | |
| 5140 | 2656 | 2657 | 2658 | 2659 | 2660 | 2661 | 2662 | 2663 | | |
| 5150 | 2664 | 2665 | 2666 | 2667 | 2668 | 2669 | 2670 | 2671 | | |
| 5160 | 2672 | 2673 | 2674 | 2675 | 2676 | 2677 | 2678 | 2679 | | |
| 5170 | 2680 | 2681 | 2682 | 2683 | 2684 | 2685 | 2686 | 2687 | | |
| 5200 | 2688 | 2689 | 2690 | 2691 | 2692 | 2693 | 2694 | 2695 | | |
| 5210 | 2696 | 2697 | 2698 | 2699 | 2700 | 2701 | 2702 | 2703 | | |
| 5220 | 2704 | 2705 | 2706 | 2707 | 2708 | 2709 | 2710 | 2711 | | |
| 5230 | 2712 | 2713 | 2714 | 2715 | 2716 | 2717 | 2718 | 2719 | | |
| 5240 | 2720 | 2721 | 2722 | 2723 | 2724 | 2725 | 2726 | 2727 | | |
| 5250 | 2728 | 2729 | 2730 | 2731 | 2732 | 2733 | 2734 | 2735 | | |
| 5260 | 2736 | 2737 | 2738 | 2739 | 2740 | 2741 | 2742 | 2743 | | |
| 5270 | 2744 | 2745 | 2746 | 2747 | 2748 | 2749 | 2750 | 2751 | | |
| 5300 | 2752 | 2753 | 2754 | 2755 | 2756 | 2757 | 2758 | 2759 | | |
| 5310 | 2760 | 2761 | 2762 | 2763 | 2764 | 2765 | 2766 | 2767 | | |
| 5320 | 2768 | 2769 | 2770 | 2771 | 2772 | 2773 | 2774 | 2775 | | |
| 5330 | 2776 | 2777 | 2778 | 2779 | 2780 | 2781 | 2782 | 2783 | | |
| 5340 | 2784 | 2785 | 2786 | 2787 | 2788 | 2789 | 2790 | 2791 | | |
| 5350 | 2792 | 2793 | 2794 | 2795 | 2796 | 2797 | 2798 | 2799 | | |
| 5360 | 2800 | 2801 | 2802 | 2803 | 2804 | 2805 | 2806 | 2807 | | |
| 5370 | 2808 | 2809 | 2810 | 2811 | 2812 | 2813 | 2814 | 2815 | | |
| 5400 | 2816 | 2817 | 2818 | 2819 | 2820 | 2821 | 2822 | 2823 | | |
| 5410 | 2824 | 2825 | 2826 | 2827 | 2828 | 2829 | 2830 | 2831 | | |
| 5420 | 2832 | 2833 | 2834 | 2835 | 2836 | 2837 | 2838 | 2839 | | |
| 5430 | 2840 | 2841 | 2842 | 2843 | 2844 | 2845 | 2846 | 2847 | | |
| 5440 | 2848 | 2849 | 2850 | 2851 | 2852 | 2853 | 2854 | 2855 | | |
| 5450 | 2856 | 2857 | 2858 | 2859 | 2860 | 2861 | 2862 | 2863 | | |
| 5460 | 2864 | 2865 | 2866 | 2867 | 2868 | 2869 | 2870 | 2871 | | |
| 5470 | 2872 | 2873 | 2874 | 2875 | 2876 | 2877 | 2878 | 2879 | | |
| 5500 | 2880 | 2881 | 2882 | 2883 | 2884 | 2885 | 2886 | 2887 | | |
| 5510 | 2888 | 2889 | 2890 | 2891 | 2892 | 2893 | 2894 | 2895 | | |
| 5520 | 2896 | 2897 | 2898 | 2899 | 2900 | 2901 | 2902 | 2903 | | |
| 5530 | 2904 | 2905 | 2906 | 2907 | 2908 | 2909 | 2910 | 2911 | | |
| 5540 | 2912 | 2913 | 2914 | 2915 | 2916 | 2917 | 2918 | 2919 | | |
| 5550 | 2920 | 2921 | 2922 | 2923 | 2924 | 2925 | 2926 | 2927 | | |
| 5560 | 2928 | 2929 | 2930 | 2931 | 2932 | 2933 | 2934 | 2935 | | |
| 5570 | 2936 | 2937 | 2938 | 2939 | 2940 | 2941 | 2942 | 2943 | | |
| 5600 | 2944 | 2945 | 2946 | 2947 | 2948 | 2949 | 2950 | 2951 | | |
| 5610 | 2952 | 2953 | 2954 | 2955 | 2956 | 2957 | 2958 | 2959 | | |
| 5620 | 2960 | 2961 | 2962 | 2963 | 2964 | 2965 | 2966 | 2967 | | |
| 5630 | 2968 | 2969 | 2970 | 2971 | 2972 | 2973 | 2974 | 2975 | | |
| 5640 | 2976 | 2977 | 2978 | 2979 | 2980 | 2981 | 2982 | 2983 | | |
| 5650 | 2984 | 2985 | 2986 | 2987 | 2988 | 2989 | 2990 | 2991 | | |
| 5660 | 2992 | 2993 | 2994 | 2995 | 2996 | 2997 | 2998 | 2999 | | |
| 5670 | 3000 | 3001 | 3002 | 3003 | 3004 | 3005 | 3006 | 3007 | | |
| 5700 | 3008 | 3009 | 3010 | 3011 | 3012 | 3013 | 3014 | 3015 | | |
| 5710 | 3016 | 3017 | 3018 | 3019 | 3020 | 3021 | 3022 | 3023 | | |
| 5720 | 3024 | 3025 | 3026 | 3027 | 3028 | 3029 | 3030 | 3031 | | |
| 5730 | 3032 | 3033 | 3034 | 3035 | 3036 | 3037 | 3038 | 3039 | | |
| 5740 | 3040 | 3041 | 3042 | 3043 | 3044 | 3045 | 3046 | 3047 | | |
| 5750 | 3048 | 3049 | 3050 | 3051 | 3052 | 3053 | 3054 | 3055 | | |
| 5760 | 3056 | 3057 | 3058 | 3059 | 3060 | 3061 | 3062 | 3063 | | |
| 5770 | 3064 | 3065 | 3066 | 3067 | 3068 | 3069 | 3070 | 3071 | | |

OCTAL-DECIMAL INTEGER CONVERSION TABLE (continued)

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|------|------|------|------|------|------|------|------|------|
| 6000 | 3072 | 3073 | 3074 | 3075 | 3076 | 3077 | 3078 | 3079 |
| 6010 | 3080 | 3081 | 3082 | 3083 | 3084 | 3085 | 3086 | 3087 |
| 6020 | 3088 | 3089 | 3090 | 3091 | 3092 | 3093 | 3094 | 3095 |
| 6030 | 3096 | 3097 | 3098 | 3099 | 3100 | 3101 | 3102 | 3103 |
| 6040 | 3104 | 3105 | 3106 | 3107 | 3108 | 3109 | 3110 | 3111 |
| 6050 | 3112 | 3113 | 3114 | 3115 | 3116 | 3117 | 3118 | 3119 |
| 6060 | 3120 | 3121 | 3122 | 3123 | 3124 | 3125 | 3126 | 3127 |
| 6070 | 3128 | 3129 | 3130 | 3131 | 3132 | 3133 | 3134 | 3135 |
| 6100 | 3136 | 3137 | 3138 | 3139 | 3140 | 3141 | 3142 | 3143 |
| 6110 | 3144 | 3145 | 3146 | 3147 | 3148 | 3149 | 3150 | 3151 |
| 6120 | 3152 | 3153 | 3154 | 3155 | 3156 | 3157 | 3158 | 3159 |
| 6130 | 3160 | 3161 | 3162 | 3163 | 3164 | 3165 | 3166 | 3167 |
| 6140 | 3168 | 3169 | 3170 | 3171 | 3172 | 3173 | 3174 | 3175 |
| 6150 | 3176 | 3177 | 3178 | 3179 | 3180 | 3181 | 3182 | 3183 |
| 6160 | 3184 | 3185 | 3186 | 3187 | 3188 | 3189 | 3190 | 3191 |
| 6170 | 3192 | 3193 | 3194 | 3195 | 3196 | 3197 | 3198 | 3199 |
| 6200 | 3200 | 3201 | 3202 | 3203 | 3204 | 3205 | 3206 | 3207 |
| 6210 | 3208 | 3209 | 3210 | 3211 | 3212 | 3213 | 3214 | 3215 |
| 6220 | 3216 | 3217 | 3218 | 3219 | 3220 | 3221 | 3222 | 3223 |
| 6230 | 3224 | 3225 | 3226 | 3227 | 3228 | 3229 | 3230 | 3231 |
| 6240 | 3232 | 3233 | 3234 | 3235 | 3236 | 3237 | 3238 | 3239 |
| 6250 | 3240 | 3241 | 3242 | 3243 | 3244 | 3245 | 3246 | 3247 |
| 6260 | 3248 | 3249 | 3250 | 3251 | 3252 | 3253 | 3254 | 3255 |
| 6270 | 3256 | 3257 | 3258 | 3259 | 3260 | 3261 | 3262 | 3263 |
| 6300 | 3264 | 3265 | 3266 | 3267 | 3268 | 3269 | 3270 | 3271 |
| 6310 | 3272 | 3273 | 3274 | 3275 | 3276 | 3277 | 3278 | 3279 |
| 6320 | 3280 | 3281 | 3282 | 3283 | 3284 | 3285 | 3286 | 3287 |
| 6330 | 3288 | 3289 | 3290 | 3291 | 3292 | 3293 | 3294 | 3295 |
| 6340 | 3296 | 3297 | 3298 | 3299 | 3300 | 3301 | 3302 | 3303 |
| 6350 | 3304 | 3305 | 3306 | 3307 | 3308 | 3309 | 3310 | 3311 |
| 6360 | 3312 | 3313 | 3314 | 3315 | 3316 | 3317 | 3318 | 3319 |
| 6370 | 3320 | 3321 | 3322 | 3323 | 3324 | 3325 | 3326 | 3327 |

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|------|------|------|------|------|------|------|------|------|
| 6400 | 3328 | 3329 | 3330 | 3331 | 3332 | 3333 | 3334 | 3335 |
| 6410 | 3336 | 3337 | 3338 | 3339 | 3340 | 3341 | 3342 | 3343 |
| 6420 | 3344 | 3345 | 3346 | 3347 | 3348 | 3349 | 3350 | 3351 |
| 6430 | 3352 | 3353 | 3354 | 3355 | 3356 | 3357 | 3358 | 3359 |
| 6440 | 3360 | 3361 | 3362 | 3363 | 3364 | 3365 | 3366 | 3367 |
| 6450 | 3368 | 3369 | 3370 | 3371 | 3372 | 3373 | 3374 | 3375 |
| 6460 | 3376 | 3377 | 3378 | 3379 | 3380 | 3381 | 3382 | 3383 |
| 6470 | 3384 | 3385 | 3386 | 3387 | 3388 | 3389 | 3390 | 3391 |
| 6500 | 3392 | 3393 | 3394 | 3395 | 3396 | 3397 | 3398 | 3399 |
| 6510 | 3400 | 3401 | 3402 | 3403 | 3404 | 3405 | 3406 | 3407 |
| 6520 | 3408 | 3409 | 3410 | 3411 | 3412 | 3413 | 3414 | 3415 |
| 6530 | 3416 | 3417 | 3418 | 3419 | 3420 | 3421 | 3422 | 3423 |
| 6540 | 3424 | 3425 | 3426 | 3427 | 3428 | 3429 | 3430 | 3431 |
| 6550 | 3432 | 3433 | 3434 | 3435 | 3436 | 3437 | 3438 | 3439 |
| 6560 | 3440 | 3441 | 3442 | 3443 | 3444 | 3445 | 3446 | 3447 |
| 6570 | 3448 | 3449 | 3450 | 3451 | 3452 | 3453 | 3454 | 3455 |
| 6600 | 3456 | 3457 | 3458 | 3459 | 3460 | 3461 | 3462 | 3463 |
| 6610 | 3464 | 3465 | 3466 | 3467 | 3468 | 3469 | 3470 | 3471 |
| 6620 | 3472 | 3473 | 3474 | 3475 | 3476 | 3477 | 3478 | 3479 |
| 6630 | 3480 | 3481 | 3482 | 3483 | 3484 | 3485 | 3486 | 3487 |
| 6640 | 3488 | 3489 | 3490 | 3491 | 3492 | 3493 | 3494 | 3495 |
| 6650 | 3496 | 3497 | 3498 | 3499 | 3500 | 3501 | 3502 | 3503 |
| 6660 | 3504 | 3505 | 3506 | 3507 | 3508 | 3509 | 3510 | 3511 |
| 6670 | 3512 | 3513 | 3514 | 3515 | 3516 | 3517 | 3518 | 3519 |
| 6700 | 3520 | 3521 | 3522 | 3523 | 3524 | 3525 | 3526 | 3527 |
| 6710 | 3528 | 3529 | 3530 | 3531 | 3532 | 3533 | 3534 | 3535 |
| 6720 | 3536 | 3537 | 3538 | 3539 | 3540 | 3541 | 3542 | 3543 |
| 6730 | 3544 | 3545 | 3546 | 3547 | 3548 | 3549 | 3550 | 3551 |
| 6740 | 3552 | 3553 | 3554 | 3555 | 3556 | 3557 | 3558 | 3559 |
| 6750 | 3560 | 3561 | 3562 | 3563 | 3564 | 3565 | 3566 | 3567 |
| 6760 | 3568 | 3569 | 3570 | 3571 | 3572 | 3573 | 3574 | 3575 |
| 6770 | 3576 | 3577 | 3578 | 3579 | 3580 | 3581 | 3582 | 3583 |

6000 to 6777
(Octal) | 3072 to 3583
(Decimal)

Octal Decimal
10000 - 4096
20000 - 8192
30000 - 12288
40000 - 16384
50000 - 20480
60000 - 24576
70000 - 28672

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|------|------|------|------|------|------|------|------|------|
| 7000 | 3584 | 3585 | 3586 | 3587 | 3588 | 3589 | 3590 | 3591 |
| 7010 | 3592 | 3593 | 3594 | 3595 | 3596 | 3597 | 3598 | 3599 |
| 7020 | 3600 | 3601 | 3602 | 3603 | 3604 | 3605 | 3606 | 3607 |
| 7030 | 3608 | 3609 | 3610 | 3611 | 3612 | 3613 | 3614 | 3615 |
| 7040 | 3616 | 3617 | 3618 | 3619 | 3620 | 3621 | 3622 | 3623 |
| 7050 | 3624 | 3625 | 3626 | 3627 | 3628 | 3629 | 3630 | 3631 |
| 7060 | 3632 | 3633 | 3634 | 3635 | 3636 | 3637 | 3638 | 3639 |
| 7070 | 3640 | 3641 | 3642 | 3643 | 3644 | 3645 | 3646 | 3647 |
| 7100 | 3648 | 3649 | 3650 | 3651 | 3652 | 3653 | 3654 | 3655 |
| 7110 | 3656 | 3657 | 3658 | 3659 | 3660 | 3661 | 3662 | 3663 |
| 7120 | 3664 | 3665 | 3666 | 3667 | 3668 | 3669 | 3670 | 3671 |
| 7130 | 3672 | 3673 | 3674 | 3675 | 3676 | 3677 | 3678 | 3679 |
| 7140 | 3680 | 3681 | 3682 | 3683 | 3684 | 3685 | 3686 | 3687 |
| 7150 | 3688 | 3689 | 3690 | 3691 | 3692 | 3693 | 3694 | 3695 |
| 7160 | 3696 | 3697 | 3698 | 3699 | 3700 | 3701 | 3702 | 3703 |
| 7170 | 3704 | 3705 | 3706 | 3707 | 3708 | 3709 | 3710 | 3711 |
| 7200 | 3712 | 3713 | 3714 | 3715 | 3716 | 3717 | 3718 | 3719 |
| 7210 | 3720 | 3721 | 3722 | 3723 | 3724 | 3725 | 3726 | 3727 |
| 7220 | 3728 | 3729 | 3730 | 3731 | 3732 | 3733 | 3734 | 3735 |
| 7230 | 3736 | 3737 | 3738 | 3739 | 3740 | 3741 | 3742 | 3743 |
| 7240 | 3744 | 3745 | 3746 | 3747 | 3748 | 3749 | 3750 | 3751 |
| 7250 | 3752 | 3753 | 3754 | 3755 | 3756 | 3757 | 3758 | 3759 |
| 7260 | 3760 | 3761 | 3762 | 3763 | 3764 | 3765 | 3766 | 3767 |
| 7270 | 3768 | 3769 | 3770 | 3771 | 3772 | 3773 | 3774 | 3775 |
| 7300 | 3776 | 3777 | 3778 | 3779 | 3780 | 3781 | 3782 | 3783 |
| 7310 | 3784 | 3785 | 3786 | 3787 | 3788 | 3789 | 3790 | 3791 |
| 7320 | 3792 | 3793 | 3794 | 3795 | 3796 | 3797 | 3798 | 3799 |
| 7330 | 3800 | 3801 | 3802 | 3803 | 3804 | 3805 | 3806 | 3807 |
| 7340 | 3808 | 3809 | 3810 | 3811 | 3812 | 3813 | 3814 | 3815 |
| 7350 | 3816 | 3817 | 3818 | 3819 | 3820 | 3821 | 3822 | 3823 |
| 7360 | 3824 | 3825 | 3826 | 3827 | 3828 | 3829 | 3830 | 3831 |
| 7370 | 3832 | 3833 | 3834 | 3835 | 3836 | 3837 | 3838 | 3839 |

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|------|------|------|------|------|------|------|------|------|
| 7400 | 3840 | 3841 | 3842 | 3843 | 3844 | 3845 | 3846 | 3847 |
| 7410 | 3848 | 3849 | 3850 | 3851 | 3852 | 3853 | 3854 | 3855 |
| 7420 | 3856 | 3857 | 3858 | 3859 | 3860 | 3861 | 3862 | 3863 |
| 7430 | 3864 | 3865 | 3866 | 3867 | 3868 | 3869 | 3870 | 3871 |
| 7440 | 3872 | 3873 | 3874 | 3875 | 3876 | 3877 | 3878 | 3879 |
| 7450 | 3880 | 3881 | 3882 | 3883 | 3884 | 3885 | 3886 | 3887 |
| 7460 | 3888 | 3889 | 3890 | 3891 | 3892 | 3893 | 3894 | 3895 |
| 7470 | 3896 | 3897 | 3898 | 3899 | 3900 | 3901 | 3902 | 3903 |
| 7500 | 3904 | 3905 | 3906 | 3907 | 3908 | 3909 | 3910 | 3911 |
| 7510 | 3912 | 3913 | 3914 | 3915 | 3916 | 3917 | 3918 | 3919 |
| 7520 | 3920 | 3921 | 3922 | 3923 | 3924 | 3925 | 3926 | 3927 |
| 7530 | 3928 | 3929 | 3930 | 3931 | 3932 | 3933 | 3934 | 3935 |
| 7540 | 3936 | 3937 | 3938 | 3939 | 3940 | 3941 | 3942 | 3943 |
| 7550 | 3944 | 3945 | 3946 | 3947 | 3948 | 3949 | 3950 | 3951 |
| 7560 | 3952 | 3953 | 3954 | 3955 | 3956 | 3957 | 3958 | 3959 |
| 7570 | 3960 | 3961 | 3962 | 3963 | 3964 | 3965 | 3966 | 3967 |
| 7600 | 3968 | 3969 | 3970 | 3971 | 3972 | 3973 | 3974 | 3975 |
| 7610 | 3976 | 3977 | 3978 | 3979 | 3980 | 3981 | 3982 | 3983 |
| 7620 | 3984 | 3985 | 3986 | 3987 | 3988 | 3989 | 3990 | 3991 |
| 7630 | 3992 | 3993 | 3994 | 3995 | 3996 | 3997 | 3998 | 3999 |
| 7640 | 4000 | 4001 | 4002 | 4003 | 4004 | 4005 | 4006 | 4007 |
| 7650 | 4008 | 4009 | 4010 | 4011 | 4012 | 4013 | 4014 | 4015 |
| 7660 | 4016 | 4017 | 4018 | 4019 | 4020 | 4021 | 4022 | 4023 |
| 7670 | 4024 | 4025 | 4026 | 4027 | 4028 | 4029 | 4030 | 4031 |
| 7700 | 4032 | 4033 | 4034 | 4035 | 4036 | 4037 | 4038 | 4039 |
| 7710 | 4040 | 4041 | 4042 | 4043 | 4044 | 4045 | 4046 | 4047 |
| 7720 | 4048 | 4049 | 4050 | 4051 | 4052 | 4053 | 4054 | 4055 |
| 7730 | 4056 | 4057 | 4058 | 4059 | 4060 | 4061 | 4062 | 4063 |
| 7740 | 4064 | 4065 | 4066 | 4067 | 4068 | 4069 | 4070 | 4071 |
| 7750 | 4072 | 4073 | 4074 | 4075 | 4076 | 4077 | 4078 | 4079 |
| 7760 | 4080 | 4081 | 4082 | 4083 | 4084 | 4085 | 4086 | 4087 |
| 7770 | 4088 | 4089 | 4090 | 4091 | 4092 | 4093 | 4094 | 4095 |

7000 to 7777
(Octal) | 3584 to 4095
(Decimal)

OCTAL-DECIMAL FRACTION CONVERSION TABLE

| Octal | Decimal | Octal | Decimal | Octal | Decimal | Octal | Decimal |
|-------|---------|-------|---------|-------|---------|-------|---------|
| .000 | .000000 | .100 | .125000 | .200 | .250000 | .300 | .375000 |
| .001 | .001953 | .101 | .126953 | .201 | .251953 | .301 | .376953 |
| .002 | .003906 | .102 | .128906 | .202 | .253906 | .302 | .378906 |
| .003 | .005859 | .103 | .130859 | .203 | .255859 | .303 | .380859 |
| .004 | .007812 | .104 | .132812 | .204 | .257812 | .304 | .382812 |
| .005 | .009765 | .105 | .134765 | .205 | .259765 | .305 | .384765 |
| .006 | .011718 | .106 | .136718 | .206 | .261718 | .306 | .386718 |
| .007 | .013671 | .107 | .138671 | .207 | .263671 | .307 | .388671 |
| .010 | .015625 | .110 | .140625 | .210 | .265625 | .310 | .390625 |
| .011 | .017578 | .111 | .142578 | .211 | .267578 | .311 | .392578 |
| .012 | .019531 | .112 | .144531 | .212 | .269531 | .312 | .394531 |
| .013 | .021484 | .113 | .146484 | .213 | .271484 | .313 | .396484 |
| .014 | .023437 | .114 | .148437 | .214 | .273437 | .314 | .398437 |
| .015 | .025390 | .115 | .150390 | .215 | .275390 | .315 | .400390 |
| .016 | .027343 | .116 | .152343 | .216 | .277343 | .316 | .402343 |
| .017 | .029296 | .117 | .154296 | .217 | .279296 | .317 | .404296 |
| .020 | .031250 | .120 | .156250 | .220 | .281250 | .320 | .406250 |
| .021 | .033203 | .121 | .158203 | .221 | .283203 | .321 | .408203 |
| .022 | .035156 | .122 | .160156 | .222 | .285156 | .322 | .410156 |
| .023 | .037109 | .123 | .162109 | .223 | .287109 | .323 | .412109 |
| .024 | .039062 | .124 | .164062 | .224 | .289062 | .324 | .414062 |
| .025 | .041015 | .125 | .166015 | .225 | .291015 | .325 | .416015 |
| .026 | .042968 | .126 | .167968 | .226 | .292968 | .326 | .417968 |
| .027 | .044921 | .127 | .169921 | .227 | .294921 | .327 | .419921 |
| .030 | .046875 | .130 | .171875 | .230 | .296875 | .330 | .421875 |
| .031 | .048828 | .131 | .173828 | .231 | .298828 | .331 | .423828 |
| .032 | .050781 | .132 | .175781 | .232 | .300781 | .332 | .425781 |
| .033 | .052734 | .133 | .177734 | .233 | .302734 | .333 | .427734 |
| .034 | .054687 | .134 | .179687 | .234 | .304687 | .334 | .429687 |
| .035 | .056640 | .135 | .181640 | .235 | .306640 | .335 | .431640 |
| .036 | .058593 | .136 | .183593 | .236 | .308593 | .336 | .433593 |
| .037 | .060546 | .137 | .185546 | .237 | .310546 | .337 | .435546 |
| .040 | .062500 | .140 | .187500 | .240 | .312500 | .340 | .437500 |
| .041 | .064453 | .141 | .189453 | .241 | .314453 | .341 | .439453 |
| .042 | .066406 | .142 | .191406 | .242 | .316406 | .342 | .441406 |
| .043 | .068359 | .143 | .193359 | .243 | .318359 | .343 | .443359 |
| .044 | .070312 | .144 | .195312 | .244 | .320312 | .344 | .445312 |
| .045 | .072265 | .145 | .197265 | .245 | .322265 | .345 | .447265 |
| .046 | .074218 | .146 | .199218 | .246 | .324218 | .346 | .449218 |
| .047 | .076171 | .147 | .201171 | .247 | .326171 | .347 | .451171 |
| .050 | .078125 | .150 | .203125 | .250 | .328125 | .350 | .453125 |
| .051 | .080078 | .151 | .205078 | .251 | .330078 | .351 | .455078 |
| .052 | .082031 | .152 | .207031 | .252 | .332031 | .352 | .457031 |
| .053 | .083984 | .153 | .208984 | .253 | .333984 | .353 | .458984 |
| .054 | .085937 | .154 | .210937 | .254 | .335937 | .354 | .460937 |
| .055 | .087890 | .155 | .212890 | .255 | .337890 | .355 | .462890 |
| .056 | .089843 | .156 | .214843 | .256 | .339843 | .356 | .464843 |
| .057 | .091796 | .157 | .216796 | .257 | .341796 | .357 | .466796 |
| .060 | .093750 | .160 | .218750 | .260 | .343750 | .360 | .468750 |
| .061 | .095703 | .161 | .220703 | .261 | .345703 | .361 | .470703 |
| .062 | .097656 | .162 | .222656 | .262 | .347656 | .362 | .472656 |
| .063 | .099609 | .163 | .224609 | .263 | .349609 | .363 | .474609 |
| .064 | .101562 | .164 | .226562 | .264 | .351562 | .364 | .476562 |
| .065 | .103515 | .165 | .228515 | .265 | .353515 | .365 | .478515 |
| .066 | .105468 | .166 | .230468 | .266 | .355468 | .366 | .480468 |
| .067 | .107421 | .167 | .232421 | .267 | .357421 | .367 | .482421 |
| .070 | .109375 | .170 | .234375 | .270 | .359375 | .370 | .484375 |
| .071 | .111328 | .171 | .236328 | .271 | .361328 | .371 | .486328 |
| .072 | .113281 | .172 | .238281 | .272 | .363281 | .372 | .488281 |
| .073 | .115234 | .173 | .240234 | .273 | .365234 | .373 | .490234 |
| .074 | .117187 | .174 | .242187 | .274 | .367187 | .374 | .492187 |
| .075 | .119140 | .175 | .244140 | .275 | .369140 | .375 | .494140 |
| .076 | .121093 | .176 | .246093 | .276 | .371093 | .376 | .496093 |
| .077 | .123046 | .177 | .248046 | .277 | .373046 | .377 | .498046 |

OCTAL-DECIMAL FRACTION CONVERSION TABLE (continued)

| Octal | Decimal | Octal | Decimal | Octal | Decimal | Octal | Decimal |
|---------|---------|---------|---------|---------|---------|---------|---------|
| .000000 | .000000 | .000100 | .000244 | .000200 | .000488 | .000300 | .000732 |
| .000001 | .000003 | .000101 | .000247 | .000201 | .000492 | .000301 | .000736 |
| .000002 | .000007 | .000102 | .000251 | .000202 | .000495 | .000302 | .000740 |
| .000003 | .000011 | .000103 | .000255 | .000203 | .000499 | .000303 | .000743 |
| .000004 | .000015 | .000104 | .000259 | .000204 | .000503 | .000304 | .000747 |
| .000005 | .000019 | .000105 | .000263 | .000205 | .000507 | .000305 | .000751 |
| .000006 | .000022 | .000106 | .000267 | .000206 | .000511 | .000306 | .000755 |
| .000007 | .000026 | .000107 | .000270 | .000207 | .000514 | .000307 | .000759 |
| .000010 | .000030 | .000110 | .000274 | .000210 | .000518 | .000310 | .000762 |
| .000011 | .000034 | .000111 | .000278 | .000211 | .000522 | .000311 | .000766 |
| .000012 | .000038 | .000112 | .000282 | .000212 | .000526 | .000312 | .000770 |
| .000013 | .000041 | .000113 | .000286 | .000213 | .000530 | .000313 | .000774 |
| .000014 | .000045 | .000114 | .000289 | .000214 | .000534 | .000314 | .000778 |
| .000015 | .000049 | .000115 | .000293 | .000215 | .000537 | .000315 | .000782 |
| .000016 | .000053 | .000116 | .000297 | .000216 | .000541 | .000316 | .000785 |
| .000017 | .000057 | .000117 | .000301 | .000217 | .000545 | .000317 | .000789 |
| .000020 | .000061 | .000120 | .000305 | .000220 | .000549 | .000320 | .000793 |
| .000021 | .000064 | .000121 | .000308 | .000221 | .000553 | .000321 | .000797 |
| .000022 | .000068 | .000122 | .000312 | .000222 | .000556 | .000322 | .000801 |
| .000023 | .000072 | .000123 | .000316 | .000223 | .000560 | .000323 | .000805 |
| .000024 | .000076 | .000124 | .000320 | .000224 | .000564 | .000324 | .000808 |
| .000025 | .000080 | .000125 | .000324 | .000225 | .000568 | .000325 | .000812 |
| .000026 | .000083 | .000126 | .000328 | .000226 | .000572 | .000326 | .000816 |
| .000027 | .000087 | .000127 | .000331 | .000227 | .000576 | .000327 | .000820 |
| .000030 | .000091 | .000130 | .000335 | .000230 | .000579 | .000330 | .000823 |
| .000031 | .000095 | .000131 | .000339 | .000231 | .000583 | .000331 | .000827 |
| .000032 | .000099 | .000132 | .000343 | .000232 | .000587 | .000332 | .000831 |
| .000033 | .000102 | .000133 | .000347 | .000233 | .000591 | .000333 | .000835 |
| .000034 | .000106 | .000134 | .000350 | .000234 | .000595 | .000334 | .000839 |
| .000035 | .000110 | .000135 | .000354 | .000235 | .000598 | .000335 | .000843 |
| .000036 | .000114 | .000136 | .000358 | .000236 | .000602 | .000336 | .000846 |
| .000037 | .000118 | .000137 | .000362 | .000237 | .000606 | .000337 | .000850 |
| .000040 | .000122 | .000140 | .000366 | .000240 | .000610 | .000340 | .000854 |
| .000041 | .000125 | .000141 | .000370 | .000241 | .000614 | .000341 | .000858 |
| .000042 | .000129 | .000142 | .000373 | .000242 | .000617 | .000342 | .000862 |
| .000043 | .000133 | .000143 | .000377 | .000243 | .000621 | .000343 | .000865 |
| .000044 | .000137 | .000144 | .000381 | .000244 | .000625 | .000344 | .000869 |
| .000045 | .000141 | .000145 | .000385 | .000245 | .000629 | .000345 | .000873 |
| .000046 | .000144 | .000146 | .000389 | .000246 | .000633 | .000346 | .000877 |
| .000047 | .000148 | .000147 | .000392 | .000247 | .000637 | .000347 | .000881 |
| .000050 | .000152 | .000150 | .000396 | .000250 | .000640 | .000350 | .000885 |
| .000051 | .000156 | .000151 | .000400 | .000251 | .000644 | .000351 | .000888 |
| .000052 | .000160 | .000152 | .000404 | .000252 | .000648 | .000352 | .000892 |
| .000053 | .000164 | .000153 | .000408 | .000253 | .000652 | .000353 | .000896 |
| .000054 | .000167 | .000154 | .000411 | .000254 | .000656 | .000354 | .000900 |
| .000055 | .000171 | .000155 | .000415 | .000255 | .000659 | .000355 | .000904 |
| .000056 | .000175 | .000156 | .000419 | .000256 | .000663 | .000356 | .000907 |
| .000057 | .000179 | .000157 | .000423 | .000257 | .000667 | .000357 | .000911 |
| .000060 | .000183 | .000160 | .000427 | .000260 | .000671 | .000360 | .000915 |
| .000061 | .000186 | .000161 | .000431 | .000261 | .000675 | .000361 | .000919 |
| .000062 | .000190 | .000162 | .000434 | .000262 | .000679 | .000362 | .000923 |
| .000063 | .000194 | .000163 | .000438 | .000263 | .000682 | .000363 | .000926 |
| .000064 | .000198 | .000164 | .000442 | .000264 | .000686 | .000364 | .000930 |
| .000065 | .000202 | .000165 | .000446 | .000265 | .000690 | .000365 | .000934 |
| .000066 | .000205 | .000166 | .000450 | .000266 | .000694 | .000366 | .000938 |
| .000067 | .000209 | .000167 | .000453 | .000267 | .000698 | .000367 | .000942 |
| .000070 | .000213 | .000170 | .000457 | .000270 | .000701 | .000370 | .000946 |
| .000071 | .000217 | .000171 | .000461 | .000271 | .000705 | .000371 | .000949 |
| .000072 | .000221 | .000172 | .000465 | .000272 | .000709 | .000372 | .000953 |
| .000073 | .000225 | .000173 | .000469 | .000273 | .000713 | .000373 | .000957 |
| .000074 | .000228 | .000174 | .000473 | .000274 | .000717 | .000374 | .000961 |
| .000075 | .000232 | .000175 | .000476 | .000275 | .000720 | .000375 | .000965 |
| .000076 | .000236 | .000176 | .000480 | .000276 | .000724 | .000376 | .000968 |
| .000077 | .000240 | .000177 | .000484 | .000277 | .000728 | .000377 | .000972 |

OCTAL-DECIMAL FRACTION CONVERSION TABLE (continued)

| Octal | Decimal | Octal | Decimal | Octal | Decimal | Octal | Decimal |
|---------|---------|---------|---------|---------|---------|---------|---------|
| .000400 | .000976 | .000500 | .001220 | .000600 | .001464 | .000700 | .001708 |
| .000401 | .000980 | .000501 | .001224 | .000601 | .001468 | .000701 | .001712 |
| .000402 | .000984 | .000502 | .001228 | .000602 | .001472 | .000702 | .001716 |
| .000403 | .000988 | .000503 | .001232 | .000603 | .001476 | .000703 | .001720 |
| .000404 | .000991 | .000504 | .001235 | .000604 | .001480 | .000704 | .001724 |
| .000405 | .000995 | .000505 | .001239 | .000605 | .001483 | .000705 | .001728 |
| .000406 | .000999 | .000506 | .001243 | .000606 | .001487 | .000706 | .001731 |
| .000407 | .001003 | .000507 | .001247 | .000607 | .001491 | .000707 | .001735 |
| .000410 | .001007 | .000510 | .001251 | .000610 | .001495 | .000710 | .001739 |
| .000411 | .001010 | .000511 | .001255 | .000611 | .001499 | .000711 | .001743 |
| .000412 | .001014 | .000512 | .001258 | .000612 | .001502 | .000712 | .001747 |
| .000413 | .001018 | .000513 | .001262 | .000613 | .001506 | .000713 | .001750 |
| .000414 | .001022 | .000514 | .001266 | .000614 | .001510 | .000714 | .001754 |
| .000415 | .001026 | .000515 | .001270 | .000615 | .001514 | .000715 | .001758 |
| .000416 | .001029 | .000516 | .001274 | .000616 | .001518 | .000716 | .001762 |
| .000417 | .001033 | .000517 | .001277 | .000617 | .001522 | .000717 | .001766 |
| .000420 | .001037 | .000520 | .001281 | .000620 | .001525 | .000720 | .001770 |
| .000421 | .001041 | .000521 | .001285 | .000621 | .001529 | .000721 | .001773 |
| .000422 | .001045 | .000522 | .001289 | .000622 | .001533 | .000722 | .001777 |
| .000423 | .001049 | .000523 | .001293 | .000623 | .001537 | .000723 | .001781 |
| .000424 | .001052 | .000524 | .001296 | .000624 | .001541 | .000724 | .001785 |
| .000425 | .001056 | .000525 | .001300 | .000625 | .001544 | .000725 | .001789 |
| .000426 | .001060 | .000526 | .001304 | .000626 | .001548 | .000726 | .001792 |
| .000427 | .001064 | .000527 | .001308 | .000627 | .001552 | .000727 | .001796 |
| .000430 | .001068 | .000530 | .001312 | .000630 | .001556 | .000730 | .001800 |
| .000431 | .001071 | .000531 | .001316 | .000631 | .001560 | .000731 | .001804 |
| .000432 | .001075 | .000532 | .001319 | .000632 | .001564 | .000732 | .001808 |
| .000433 | .001079 | .000533 | .001323 | .000633 | .001567 | .000733 | .001811 |
| .000434 | .001083 | .000534 | .001327 | .000634 | .001571 | .000734 | .001815 |
| .000435 | .001087 | .000535 | .001331 | .000635 | .001575 | .000735 | .001819 |
| .000436 | .001091 | .000536 | .001335 | .000636 | .001579 | .000736 | .001823 |
| .000437 | .001094 | .000537 | .001338 | .000637 | .001583 | .000737 | .001827 |
| .000440 | .001098 | .000540 | .001342 | .000640 | .001586 | .000740 | .001831 |
| .000441 | .001102 | .000541 | .001346 | .000641 | .001590 | .000741 | .001834 |
| .000442 | .001106 | .000542 | .001350 | .000642 | .001594 | .000742 | .001838 |
| .000443 | .001110 | .000543 | .001354 | .000643 | .001598 | .000743 | .001842 |
| .000444 | .001113 | .000544 | .001358 | .000644 | .001602 | .000744 | .001846 |
| .000445 | .001117 | .000545 | .001361 | .000645 | .001605 | .000745 | .001850 |
| .000446 | .001121 | .000546 | .001365 | .000646 | .001609 | .000746 | .001853 |
| .000447 | .001125 | .000547 | .001369 | .000647 | .001613 | .000747 | .001857 |
| .000450 | .001129 | .000550 | .001373 | .000650 | .001617 | .000750 | .001861 |
| .000451 | .001132 | .000551 | .001377 | .000651 | .001621 | .000751 | .001865 |
| .000452 | .001136 | .000552 | .001380 | .000652 | .001625 | .000752 | .001869 |
| .000453 | .001140 | .000553 | .001384 | .000653 | .001628 | .000753 | .001873 |
| .000454 | .001144 | .000554 | .001388 | .000654 | .001632 | .000754 | .001876 |
| .000455 | .001148 | .000555 | .001392 | .000655 | .001636 | .000755 | .001880 |
| .000456 | .001152 | .000556 | .001396 | .000656 | .001640 | .000756 | .001884 |
| .000457 | .001155 | .000557 | .001399 | .000657 | .001644 | .000757 | .001888 |
| .000460 | .001159 | .000560 | .001403 | .000660 | .001647 | .000760 | .001892 |
| .000461 | .001163 | .000561 | .001407 | .000661 | .001651 | .000761 | .001895 |
| .000462 | .001167 | .000562 | .001411 | .000662 | .001655 | .000762 | .001899 |
| .000463 | .001171 | .000563 | .001415 | .000663 | .001659 | .000763 | .001903 |
| .000464 | .001174 | .000564 | .001419 | .000664 | .001663 | .000764 | .001907 |
| .000465 | .001178 | .000565 | .001422 | .000665 | .001667 | .000765 | .001911 |
| .000466 | .001182 | .000566 | .001426 | .000666 | .001670 | .000766 | .001914 |
| .000467 | .001186 | .000567 | .001430 | .000667 | .001674 | .000767 | .001918 |
| .000470 | .001190 | .000570 | .001434 | .000670 | .001678 | .000770 | .001922 |
| .000471 | .001194 | .000571 | .001438 | .000671 | .001682 | .000771 | .001926 |
| .000472 | .001197 | .000572 | .001441 | .000672 | .001686 | .000772 | .001930 |
| .000473 | .001201 | .000573 | .001445 | .000673 | .001689 | .000773 | .001934 |
| .000474 | .001205 | .000574 | .001449 | .000674 | .001693 | .000774 | .001937 |
| .000475 | .001209 | .000575 | .001453 | .000675 | .001697 | .000775 | .001941 |
| .000476 | .001213 | .000576 | .001457 | .000676 | .001701 | .000776 | .001945 |
| .000477 | .001216 | .000577 | .001461 | .000677 | .001705 | .000777 | .001949 |

APPENDIX 6

PERFORATED TAPE LOADER SEQUENCES

READIN MODE LOADER

The readin mode (RIM) loader is a minimum length, basic, perforated tape reader program for the 33 ASR. It is initially stored in memory by manual use of the operator console keys and switches. The loader is permanently stored in 18 locations on page 37.

A perforated tape to be read by the RIM loader must be in RIM format:

| <u>Tape Channel</u> 8 7 6 5 4 S 3 2 1 | <u>Format</u> |
|--|--|
| 1 0 0 0 0 . 0 0 0 | Leader-trailer code |
| 0 1 A1 . A2 0 0 A3 . A4 | Absolute address to contain next 4 digits |
| 0 0 X1 . X2 0 0 X3 . X4 | Content of previous 4-digit address |
| 0 1 A1 . A2 0 0 A3 . A4 | Address |
| 0 0 X1 . X2 0 0 X3 . X4 | Content |
| (Etc.) | (Etc.) |
| 1 0 0 0 0 . 0 0 0 | Leader-trailer code |

The RIM loader can only be used in conjunction with the 33 ASR reader (not the high-speed perforated tape reader). Because a tape in RIM format is, in effect, twice as long as it need be, it is suggested that the RIM loader be used only to read the binary loader when using the 33 ASR. (Note that PDP-8 diagnostic program tapes are in RIM format.)

The complete PDP-8 RIM loader (SA = 7756) is as follows:

| Absolute Address | Octal Content | Tag | Instruction I Z | Comments |
|------------------|-------------------------|------|-----------------|------------------------|
| 7756, | 6032 | BEG. | KCC | /CLEAR AC AND FLAG |
| 7757, | 6031 | | KSF | /SKIP IF FLAG = 1 |
| 7760, | 5357 | | JMP .-1 | /LOOKING FOR CHARACTER |
| 7761, | 6036 | | KRB | /READ BUFFER |
| 7762, | 7106 | | CLL RTL | |
| 7763, | 7006 | | RTL | /CHANNEL 8 IN ACO |
| 7764, | 7510 | | SPA | /CHECKING FOR LEADER |
| 7765, | 5357 | | JMP BEG+1 | /FOUND LEADER |
| 7766, | 7006 | | RTL | /OK, CHANNEL 7 IN LINK |
| 7767, | 6031 | | KSF | |
| 7770, | 5367 | | JMP .-1 | |
| 7771, | 6034 | | KRS | |
| 7772, | 7420 | | SNL | /READ, DO NOT CLEAR |
| 7773, | 3776 | | DCA I TEMP | /CHECKING FOR ADDRESS |
| 7774, | 3376 | | DCA TEMP | /STORE CONTENT |
| 7775, | 5356 | | JMP BEG | /STORE ADDRESS |
| 7776, | 0 | TEMP | 0 | /NEXT WORD |
| 7777, | JMP START OF BIN LOADER | | 0 | /TEMP STORAGE |

Placing the RIM loader in core memory by way of the operator console keys and switches is accomplished as follows:

1. Set the starting address 7756 in the switch register (SR).
2. Press LOAD ADDRESS key.
3. Set the first instruction (6032) in the SR.
4. Press the DEPOSIT key.
5. Set the next instruction (6031) in the SR.
6. Press DEPOSIT key.
7. Repeat steps 5 and 6 until all 16 instructions have been deposited.

To load a tape in RIM format, place the tape in the reader, set the SR to the starting address 7756 of the RIM loader (not of the program being read), press the LOAD ADDRESS key, press the START key, and start the Teletype reader.

Refer to Digital Program Library document Digital-8-1-U for additional information on the Readin Mode Loader program.

BINARY LOADER

The binary loader (BIN) is used to read machine language tapes (in binary format) produced by the program assembly language (PAL). A tape in binary format is about one half the length of the comparable RIM format tape. It can, therefore, be read about twice as fast as a RIM tape and is, for this reason, the most desirable format to use with the 10 cps 33 ASR reader or the Type 750C High Speed Perforated Tape Reader.

The format of a binary tape is as follows:

Leader: about 2 feet of leader-trailer codes.

BODY: characters representing the absolute, machine language program in easy-to-read binary (or octal) form. The section of tape may contain characters representing instructions (channels 8 and 7 not punched) or origin resettings (channel 8 not punched, channel 7 punched) and is concluded by 2 characters (channels 8 and 7 not punched) that represent a checksum for the entire section.

Trailer: same as leader

Example of the format of a binary tape:

| <u>Tape Channel</u> 8 7 6 5 4 3 2 1 | <u>Memory</u> <u>Location</u> | <u>Content</u> | <u>Comments</u> |
|--|----------------------------------|----------------|---------------------|
| 1 0 0 0 0 . 0 0 0 | | | leader-trailer code |
| 0 1 0 0 0 . 0 1 0 | | | |
| 0 0 0 0 0 . 0 0 0 | | 0200 | |
| 0 0 1 1 1 . 0 1 0 | | | |
| 0 0 0 0 0 . 0 0 0 | 0200 | CLA | origin setting |
| 0 0 0 0 1 . 0 1 0 | | | |
| 0 0 1 1 1 . 1 1 1 | 0201 | | TAD 277 |
| 0 0 0 1 1 . 0 1 0 | | | |
| 0 0 1 1 1 . 1 1 0 | 0202 | DCA 276 | |
| 0 0 1 1 1 . 1 0 0 | | | |
| 0 0 0 0 0 . 0 1 0 | 0203 | HLT | |
| 0 1 0 0 0 . 0 1 0 | | | |
| 0 0 1 1 1 . 1 1 1 | | 0277 | origin setting |
| 0 0 0 0 0 . 0 0 0 | | | |
| 0 0 1 0 1 . 0 1 1 | 0277 | 0053 | |
| 0 0 0 0 1 . 0 0 0 | | | |
| 0 0 0 0 0 . 1 1 1 | | 1007 | sum check |
| 1 0 0 0 0 . 0 0 0 | | | leader-trailer code |

After a BIN tape has been read in, one of the two following conditions exists:

- a. No checksum error: halt with AC = 0
- b. Checksum error: halt with AC = (computed checksum) — (tape checksum)

Operation of the BIN loader in no way depends upon or uses the RIM loader. To load a tape in BIN format place the tape in the reader, set the SR to 7777 (the starting address of the BIN loader), press the LOAD ADDRESS key, set SR switch 0 up for loading via the Teletype unit or down for loading via the high speed reader, then press the START key, and start the tape reader.

Refer to Digital Program Library document Digital-8-2-U-RIM for additional information on the Binary Loader program.

digital

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

Printed in U.S.A.

25-12/66